UNIVERSITY OF GRANADA

Department of Computer Architecture and
Computer Technology

PhD Thesis Dissertation:

Specific-Purpose Processing Architectures for Dynamic Artificial Vision Systems

by
Francisco Barranco Expósito

Advisors:
**Javier A. Díaz Alonso, Begoña del Pino Prieto,
Eduardo Ros Vidal**

Granada, July 2012

Editor: Editorial de la Universidad de Granada
Autor: Francisco Barranco Expósito
D.L.: GR 504-2013
ISBN: 978-84-9028-387-5

UNIVERSIDAD DE GRANADA

Specific-Purpose Processing Architectures for Dynamic Artificial Vision Systems

(Sistema Dinámico de Visión Artificial basado en Arquitecturas para Procesamiento de Propósito Específico)

Dissertation presented by:
Francisco Barranco Expósito

To apply for the:

European PhD degree in Computer Science

Signed. Francisco Barranco Expósito

D. Javier A. Díaz Alonso, Dña. Begoña del Pino Prieto y D. Eduardo Ros Vidal, Profesor Titular, Profesora Titular y Catedrático de Universidad respectivamente del Departamento de Arquitectura y Tecnología de Computadores de la Universidad de Granada

CERTIFICAN

Que la memoria titulada "Specific-Purpose Processing Architectures for Dynamic Artificial Vision Systems" ha sido realizada por D. Francisco Barranco Expósito, bajo nuestra dirección en el Departamento de Arquitectura y Tecnología de Computadores de la Universidad de Granada para optar al grado de Doctor Europeo en Ingeniería Informática y que hasta donde nuestro conocimiento alcanza, en la realización del trabajo, se han respetado los derechos de otros autores a ser citados, cuando se han utilizado sus resultados o publicaciones.

Granada, a de Septiembre de 2012

Los directores

Fdo: Javier A. Díaz Alonso   Fdo: Begoña del Pino Prieto   Fdo: Eduardo Ros Vidal

Agradecimientos

I would like to acknowledge all people that during these years have contributed to this thesis. It is very difficult to enumerate all of then here, hence I am going to thank especially to my family because without its support It would have been impossible. Moreover, I want to mention my thesis supervisors Begoña, Javi, and Edu for their patience, their hard work and dedication, and particularly their support and advice. They gave me the opportunity of developing a very challenging and appealing work. Thanks also to all the ATC department and specially to the CIE and CITIC people. I would like to acknowledge all the DRIVSCO people for their role in my research and the Spanish Ministry of Education for its economic support.

Table of Contents

Abstract

A visual system in dynamic environments copes with an amount of information that becomes unmanageable. Dynamic vision systems consists of several complex elements such as the knowledge database about objects in the scene, their behavior and capabilities, their inter-relationships, or environmental conditions; a list of planned steps to accomplish a task which also includes some feedback for reconsidering it; components to synchronize the multimodal information that system manages; and active vision. This last component is one of the most complex components.

In dynamic environments, static vision systems that merely observe the scene are unfit. Active vision includes the control of gaze (control of the eye movements) and the visual attention; adaptation is included as consequence of the feedback in several components of the dynamic vision systems.

Human eye perceives high-quality information from the region that surrounds the center of gaze (fovea). The active control of gazing is firstly, a task-dependent process and secondly, helps reducing the processing resources selecting a small window of the scene. In this work, we present a system designed to control the vergence for the "iCub" robotic head, by using depth cues. Initially, we give the system the capacity of emulating the neural control mechanism called "fixation", that helps the system to stay still to explore in detail a static target. Secondly, an object moving to variable speed in the 3D space is quite difficult to track. In our work, we also face the solution of this process which is called "smooth pursuit" in the literature.

On the other hand, the biological way of reducing the huge visual bandwidth and then, optimize the available resources is by selecting the most interesting areas and focusing the computation of the system for data that come from them: this process is called visual attention. Our aim is designing a model and an implementation for a visual attention system that selects the most important areas in a scene to further process them there more accurately. This implementation is valuable for systems such as advanced driving assistance, integrated in smart cameras for video-surveillance, industrial inspection applications in uncontrolled scenarios, or even in aid devices for low-vision patients.

This thesis dissertation presents two different approaches to the visual attention model implementation for dynamic environments. Firstly, we implement a strongly bio-inspired system that uses the responses generated by various kinds of cells in an artificial retina. These sparse cues are then used as inputs for a system that computes optical flow. This implementation is complemented by adding an attentional top-down mechanism that selects the most reliable cues. The attentional modulation stream is deployed using color, texture or speed information of the elements in the scene in order to bias the cues of objects of interest depending on the task that is taking place. Or in our case, biasing the computation of optical flow to focus it on our target objects.

We also carry out a second alternative that combines two forms of attention: inherent bottom-up that depends on the contrast of some features of the objects in the scene, and a task-dependent top-down that biases the competition of the most salient locations according to the addressed task requirements. The first part of this alternative consists in designing and implementing a low-level vision system capable of dealing with real-world scenarios in real time, in an embedded device. These strong requirements lead us to focus on implementing our system for an FPGA, a reconfigurable specific purpose device. The implementation is developed in an incremental manner, by firstly setting the basic architecture for the computation of multiscale optical flow. This first architecture is then evolved to include also the computation of disparity, and local energy and orientation, in a single chip. Various alternatives are also tested with the purpose of reducing the resource cost and of improving the estimation precision. Hence, the inclusion of color cues is benchmarked for the implementation of optical flow and disparity, presenting a large amount of results for the estimations with diverse color representations and a thorough study of its impact in the total resource cost. We also present an alternative for the multiscale-with-warping scheme implemented for the optical flow computation that allows a significant reduction of resource cost by dropping out an affordable density of estimations. The selection of these alternative schemes has to be carefully taken, studying the specific application requirements in terms of accuracy, density and resource utilization. After the implementation of this layer of low-level visual processing engines, we design an architecture to generate a visual attention system that coordinates a bottom-up saliency stream using the energy, orientation, motion and a new color opponency engines, and a top-down modulation that uses the disparity and motion ones. The system is tested in the framework of advanced driving assistance systems.

Resumen

Un sistema visual en entornos dinámicos maneja una gran cantidad de información que puede llegar a ser inmanejable. Los sistemas de visión dinámicos están compuestos por diferentes elementos complejos como pueden ser la base de datos de conocimiento sobre los objetos en la escena, su comportamiento y sus capacidades, sus inter-relaciones, o las condiciones ambientales; una lista de pasos planeados para la consecución de la tarea que incluyen lazos de realimentación para reconsiderarla; componentes para sincronizar la información multimodal que el sistema maneja; la visión activa. Éste último componente es uno de los componentes más complejos.

En medios dinámicos, los sistemas de visión estáticos que únicamente observan la escena se revelan inapropiados. Los sistemas de visión activa incluyen el control de la mirada (control de los movimientos de los ojos) y la atención visual; la adaptación es incluida como una consecuencia de la realimentación en varios componentes de los sistemas de visión dinámicos.

El ojo humano percibe información de gran calidad en la región que rodea al centro de la mirada (la fóvea). El control activo de la mirada es inicialmente, un proceso que depende de la tarea que se realiza y en segundo lugar, ayuda a reducir los recursos para el procesamiento seleccionando una ventana pequeña de la escena. En este trabajo, presentamos un sistema diseñado para controlar la vergencia de la cabeza robótica "iCub", utilizando la información de la profundidad. Inicialmente, le damos al sistema la capacidad de emular el mecanismo de control neuronal llamado "fijación", que ayuda al sistema a mantenerse estable para explorar en detalle objetos que están parados. En segundo lugar, un objeto que se mueve a una velocidad variable en el espacio 3D es bastante difícil de seguir. En nuestro trabajo, también nos enfrentamos a solucionar este proceso que es llamado "seguimiento suave" en la literatura.

Por otro lado, la forma biológica de reducir un alto ancho de banda y por tanto, de reducir los recursos que el sistema necesita es seleccionar las áreas más interesantes y centrar la computación del sistema en la información que proviene de esas áreas: este proceso se llama atención visual. Nuestra meta es diseñar un modelo y una implementación para un sistema de atención visual que seleccione las áreas más relevantes en una escena para después procesarlas de forma mucho más precisa. Esta implementación es importante para sistemas tales como lo asistencia avanzada a la conducción, integrados en cámaras inteligentes para video-vigilancia, aplicaciones de inspección industrial en escenarios no controlados, o incluso en dispositivos de ayuda para pacientes de baja visión.

Esta tesis presenta dos aproximaciones para la implementación de un modelo de atención visual para medios dinámicos. En primer lugar, implementamos un modelo fuertemente bio-inspirado que utilizar las respuestas generadas por diversos tipos de células en una retina artificial. Esta información dispersa es entonces usada como entrada para un sistema que calcula flujo óptico. Esta implementación es completada añadiendo un mecanismo de atención visual de arriba a abajo que selecciona la información más fiable. Este flujo de modulación atencional es generado utilizando información de color, textura o de velocidad de los elementos de la escena para modular las respuestas de los objetos de interés dependiendo de la tarea que se está llevando a cabo. O en nuestro caso, modulando la computación del flujo óptico para centrarla en los objetos de interés.

También llevamos a cabo una segunda alternativa que combina dos formas de atención: uno de abajo a arriba que es inherente a los objetos de la escena y que depende del contraste de algunas características de esos objetos con respecto al medio, y un flujo de modulación que funciona de arriba a abajo y que es dependiente de la tarea que se está realizando, de forma que modifica la competición para la elección de las áreas más salientes de acuerdo con los requerimientos de la tarea. La primera parte de esta alternativa consiste en diseñar e implementar un sistema de bajo nivel de visión que sea capaz de trabajar con escenarios reales en tiempo real, en un dispositivo empotrado. Estos fuertes requerimientos nos llevan a la elección de la FPGA para implementar nuestro sistema, un dispositivo de propósito específico reconfigurable. La implementación se lleva a cabo de forma incremental, seleccionando primero la arquitectura básica para el cómputo del flujo óptico multiescala. Esta primera arquitectura es entonces evolucionada para incluir también el cálculo de la disparidad, la energía local y la orientación, en un único chip. Diversas alternativas son testeadas con el propósito de reducir el coste en recursos y de mejorar la precisión de la estimación. Por tanto, la inclusión de la información de color es testeada para la implementación del flujo óptico y de la disparidad, presentado un gran conjunto de resultados para las estimaciones con diferentes espacios de color y un concienzudo estudio de su impacto en el total de recursos requeridos. También presentamos una alternativa para la implementación del flujo óptico eliminando la compensación de movimiento iterativa, lo que permite también una significativa reducción de los recursos, perdiendo una razonable densidad de estimaciones. La selección de estos esquemas alternativos tiene que ser llevada a cabo cuidadosamente, estudiando los requerimientos específicos de la aplicación en términos de precisión, densidad y de utilización de recursos. Después de la implementación de esta capa de motores de procesamiento de bajo nivel de visión, diseñamos una arquitectura final para el sistema de atención visual que coordina el flujo de saliencia de abajo a arriba utilizando la energía, la orientación y el color, junto con el movimiento, y otro de modulación que funciona de arriba a abajo y que usa el flujo óptico y la disparidad para ello. Finalmente, este sistema es testeado en el marco de sistemas avanzados de asistencia a la conducción.

Part I. PhD dissertation

> *"You do not really understand something*
> *unless you can explain it to your grandmother"*
> *[Albert Einstein]*

1 Introduction

One of the most remarkable aspects of the human visual system is its ability to process the huge amount of information that we receive when we open our eyes and its adaptation capability to challenging dynamic environments. Changes in the environments may be due for example to illumination variations, objects that change or move, or a change in the viewpoint. The way in which we are able to efficiently manage this information is very interesting for many different applications. Additionally, the visual perception system translates images into cognitive information, which results in a really complex process. In a conceptual level, it is structured as a set of layers that process the information at different abstraction levels [1]: the low level, with the extraction of visual modalities such as motion, depth, local energy or orientation, etc; the middle level that combines the previous extracted modalities or segments the information; and the high level, the scene understanding level. Furthermore, the system also includes feed-back loops used to modify the image processing at the different levels with respect to the extracted features with a higher confidence in the different stages. According to that, some questions in the nature arise such as why some patterns are difficult to detect, as camouflaged enemies in a battle? Or, why is more salient or prominent a traffic signal than its surroundings? And, why a prey running is quickly detected by its predator? Or from a more abstract level, how is the process to attend to what is substantial to our survival in a very dynamic scenario? How to refine the information of a certain area to fulfill our requirements? How is the connection between these processes and the task that we are addressing? And finally, from an engineering point of view, how can we mimic this process to exploit its advantages?

The surrounding real world is changing hence, biological systems have developed strategies to cope with it. Dealing with dynamic environments is crucial for real-world machine vision systems. They can carry some tasks out wherein the motion of objects, motion of the camera, or illumination changes are involved. The dynamic vision analysis has three different stages [2]. Firstly, the peripheral stage is related with the extraction of information that will be used in the next layers and looks for the areas with the most relevant information. The attentive stage focuses the computation in the previously selected areas and extracts information for instance, for the object recognition or motion. Finally, the cognitive stage applies the 'a priori' knowledge about the objects, the environment conditions, or the relationships between the elements in the scene. This stage analyzes the present state of the elements and explores image understanding to analyze the events that are taking place.

Perception processes are generally active which means that visual perception processes do not consist merely in image acquisition and off-line processing, but include active selection mechanisms of relevant information, modulation of dynamic filters or attention, and active gaze control. Active vision is defined as the problem of applying intelligent control strategies to the data that depend on their current state (R. Bajcsy in 1985, [3]). In this framework, image acquisition and processing are no longer two independent processes. Active vision and thus, gaze control, attention, and recognition, are necessary in several tasks as in manipulation, e.g. for focusing in a hidden region after a change in the fixation; exploration, looking for new areas to be visited; disambiguation, generated by a motion or changes in illumination; adaptation, to select the parameter values for dynamic filters to achieve the most optimized configuration for a given scenario, etc. In a mostly static scenario, all these mechanisms are simulated separately using different models but, in the case of dynamic scenarios, real-time processing is required. In this case, we face the system optimization to reduce the processing time related with the scene dynamics. In this way, it is required to compute the more dynamic characteristics of the scene in an efficient way while the static information can be computed with more complex processes.

In particular, attention is defined as the set of control mechanisms that the visual system uses to perform searches with its constrained resources. It can be seen as an adaptive mechanism to manage limited processing resources on a dynamic scenario. There exist three kinds of mechanisms [3]: selection of the spatial region depending on a response or stimulus, restriction due to the specific task dependency and finally, a suppression process performed by a surround inhibition and the inhibition of return that avoids revisiting locations. These three mechanisms are carried out as a combination of a bottom-up saliency data stream and a top-down task-dependent modulation [3]. Additionally, attention is usually closely related with two widely studied problems in computer vision: recognition and binding. There is no real consensus about attention and recognition inter-relationship, attention is commonly believed to work before recognition but some theories also propose that attention is in a particular way tied to objects or object parts [4] [5]. Finally, binding is defined as the process that links a feature for an object, as e.g. its shape with its location.

The goal of this thesis is taking these ideas about dynamic vision systems, and in particular about active vision and attentional mechanisms and develop limited resource embedded systems for real-world applications. These applications require smart adaptation to the dynamic environment and visual attention for optimizing the use of the available resources. Current technology offers several alternatives for their design and simulation: optimized standard computer, GPU, DSP, custom ASIC, or FPGA-based implementations. FPGAs allow us to implement specific architectures for applications with massively parallel processing, with low power consumption and reduced size and cost. All these properties make these configurable devices appropriate for our requirements.

In addition to this, their easy integration make them quite suitable for industrial platforms such as mobile robots, vehicles, industrial inspection platforms, or smart cameras. Finally, an implementation of this visual system in an embedded device provides a system with real-time performance, a very interesting attribute from a neurobiological point of view. A high-performance system allows the exploration and test of active vision strategies in the framework of perception-action close loops, that are not possible with an off-line system.

Although the purpose of this thesis dissertation cannot be as ambitious as to answer all the questions previously formulated, our main objective is attempting to answer some of them with the implementation of a very complex computational model implemented in an embedded device. Specifically, we have designed an implementation for the saliency computation using the very well-known model of Itti and Koch [6]. This method is widely used in the literature as a model to compare with the performances of new implementations as in [7] [8] [9]. The basic model based on a closed set of image features is enriched with motion estimation. Additionally, we also include a top-down task-dependent modulation that allows to adapt the system parameters to its environment. This bias is generated using diverse knowledge of the scene, including local depth estimations.

In our case, we may address the integration of our hardware system in devices for different applications: in the field of video-surveillance the system may be interesting integrated in real cameras or as an independent system for real-time warning of suspicious event; in the vehicle industry there are lots of works for the use of attention systems in advanced driving assistance as, for example in signal recognition or location; in the field of robotics, such a system can help autonomous navigation of mobile platforms as, e.g. in space robotics vehicles where the transmission of commands is slow or in very dynamic environments wherein the human inspection is not enough for the navigation control; in military applications to detect camouflaged targets in challenging environments, or in medicine, integrated in aid devices for low-vision patients (affected by eye or vision diseases) as e.g. in head mounted displays for improving their independence in tasks such as driving, or coping with job responsibilities or housework. In this thesis dissertation we have evaluated our system in the framework of driving assistance systems.

1.1 Dynamic vision systems

Machine vision systems have developed different strategies for applications that cope with the dynamic real-world. Changes in camera viewpoint, moving or changing objects, different light conditions, or illumination variations make essential the design of specific solutions.

As commented in the introduction, a dynamic vision system is not simply visual attention and gaze control (active vision). It also includes more components that are concerned with (based on [10]): the knowledge base, tasks, models, adaptation, and synchronization. Fig. 1 visualizes the components of the dynamic vision systems.

The most complex element of the dynamic vision systems is active vision, but it will be explained in detail in Section 1.1.1. In Fig. 1 we see an arrow that points to the active vision component and that represent the information of the real-world for the target exploration. This relation represents the relevance of the motion or action in the general scheme, because it generates the visual information, making the system 'active'.

Next, we also find one of the most important components: the knowledge base. This element summarizes all the background knowledge that the system needs to understand for example the objects in the scene and their categories, their interdependencies, their behavioral capabilities for locomotion and vision, or the environment conditions. Furthermore, taking into account the task to be performed, it also needs knowledge about the best estimates of the current and future estates of these objects, their parameters, or their possible actions.

Models conform another component. They mainly cope with methods to translate the real 3D world into the 2D representation that machine vision usually manages. It also includes the models to represent the future

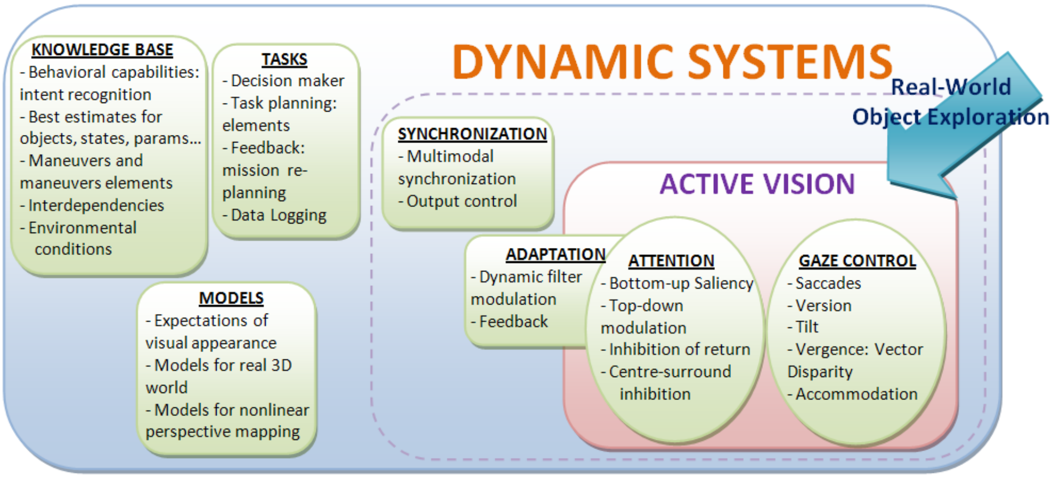

DYNAMIC SYSTEMS
Real-World Object Exploration
KNOWLEDGE BASE
- Behavioral capabilities: intent recognition
- Best estimates for objects, states, params...
- Maneuvers and maneuvers elements
- Interdependencies
- Environmental conditions
TASKS
- Decision maker
- Task planning: elements
- Feedback: mission re-planning
- Data Logging
MODELS
- Expectations of visual appearance
- Models for real 3D world
- Models for nonlinear perspective mapping
SYNCHRONIZATION
- Multimodal synchronization
- Output control
ACTIVE VISION
ADAPTATION
- Dynamic filter modulation
- Feedback
ATTENTION
- Bottom-up Saliency
- Top-down modulation
- Inhibition of return
- Centre-surround inhibition
GAZE CONTROL
- Saccades
- Version
- Tilt
- Vergence: Vector Disparity
- Accommodation

Figure 1: Main components of the Dynamic Vision Systems (inspired in [10]).

expectations about the visual appearance of the objects in the scene, taking into account the knowledge provided by the previous component.

The task component includes the information about the planning. The mission to be accomplished is decomposed in simpler tasks. This component summarizes the knowledge about the way to solve each simple task, the states to be achieved and the re-planning through the feedback, or the relationships of the object after each task.

Till this point, all the explained elements are common to any vision system. Dynamic systems also includes two more elements (taking also into account the active vision components): adaptation and synchronization. Adaptation is needed in active vision and in general, distributed in different components of the dynamic vision systems. Adaptation changes dynamically parameters or methods in the different elements in order to optimize the results and to perform different tasks. The synchronization component monitors the correct functioning of the different elements and minimizes the delays between the information extraction and the task planner. This component also synchronizes the multimodal knowledge and extracted cues.

1.1.1 Active vision

The term ‘active’ is used in opposition to ‘passive’ and links the vision to an action, in other words, it makes clearer the relationship between visual perception and action. An active vision system is able to manage its own resources efficiently in order to perform a task [11]. This kind of systems use passive or active sensors, but the system may act upon the viewpoint,

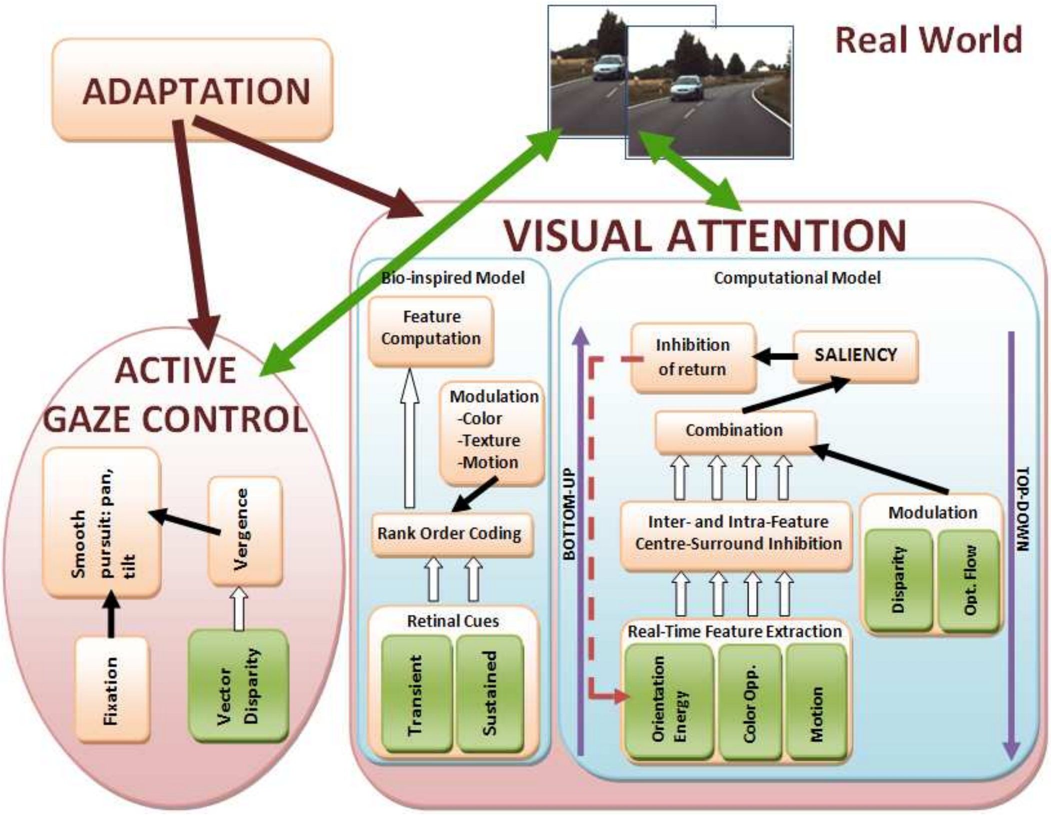


Figure 2: General scheme of the dynamic vision system implemented for this Ph.D. dissertation. Active vision consists in the conjugate work of gaze control and the cognitive attentive process. In this dissertation we explore different systems and implementations for real-time active gaze control and two different models for visual attention.

or the camera parameters. This system is also able to efficiently manage the available resources dedicating them to some relevant areas of the scene or processing those ones that are more important in order to accomplish a specific task. Hence, active vision has to cope with the camera position and properties, or foveation in fully bio-inspired systems. All this is included in the area that we call active gaze control. Secondly, it also has to efficiently manage the resources to select the most relevant information of the scene. This is considered par of the visual attention field. A general scheme of the dynamic vision system is detailed in Fig. 2.

Active gaze control

Gaze control is the process of addressing fixation to different parts of the scene driven by the task that is being performed in a cognitive, perceptual or behavioral level [12]. The information in the small region where is centered the gaze (the area correspondent to the fovea) is acquired with the highest quality. Thus, when we want to perceive the details of an object, we have to move the fovea to it. Actually, we move our eyes around 3 or 4 times each second to address the fovea to different part of the scene. These rapid movements are called 'saccades'.

According with [12], gaze control is crucial because visual perception is active and therefore, the visual systems looks for the most relevant information in the scene according to the task that is being performed. Furthermore, eyes also play a central role in the cognitive attention process and are used as the exploration window for the global scenario.

The gaze system allows us to perform all the previous processing by the conjugation of two different systems [13] [14] [15]: the oculomotor system which moves the eyes in the ocular orbits and the head movement system which moves the eyes in the 3D space. There are six different neural control systems for the gaze control:

- *Saccadic eye movements*: Rapid movements that shift the fovea to an object of interest in the periphery.
- *Vergence*: This mechanism allows to change the fixation for different depth planes.
- *Smooth pursuit*: This system keeps centered on the fovea an object of interest that is smoothly moving.
- *Fixation*: This process helps the eye to stay still in the orbit to focus on a static object.
- *Vestibulo-ocular reflex (VOR)*: This mechanism is used to stabilize the eye during head movements and helps keeping still objects on the retina during these brief head movements.
- *Optokinetic reflex (OKR)*: This mechanism is also used to stabilize the eye during head movements and in this case, keeps images still on the retina during head rotations.

In this thesis dissertation we encompass part of the active gaze control, developing a system that controls three of the mechanisms in the previous list: the fixation, the smooth pursuit and the vergence control.

Visual attention

Perhaps the first plausible definition of attention is, as the Websters's Medical Dictionary stands out, “The ability to focus selectively on a selected stimulus, sustaining that focus and shifting it at will. The ability to concentrate". The Etymology may also help with the definition of attention. It comes from the Latin "attentus", the past participle of "attendere" which means "to heed". After that, several authors have tried to understand it as Malebranche that related attention with conserving evidence of our knowledge or Leibnitz that stated that attention is required for consciously perceiving objects. Furthermore, over the 19th century, the influence of attention has been seen as a reinforcement (Müller 1873, Exner 1894) or an inhibition (Wundt 1902, Exner 1894).

Vision allows us to interact with the dynamic real-world we are involved in with our limited resources (an example of visual attention in a real-world scenario is shown in Fig. 3). In many biological systems, the strategy for inspecting a scene always seems serial. In the case of primates, it may seem contradictory the use of a serial strategy with a massively parallel structure available. The problem is that the huge amount of information transmitted via the optic nerve cannot be processed by the brain with this parallel architecture due to the constrained resources. The only way of overcoming this issue is selecting in a serial manner areas of interest in the scene that have to be attended or processed first and then, shifting to the next location. In fact, as Itti points out [16] we do not see everything around us as is commonly accepted.

According to Rothenstein and Tsotsos classifications [17], our proposals are included into the category of "Computational Models" and within it, into the "Cognitive models" (see [18]) because they are inspired by cognitive concepts.

In the visual attention process, two different mechanisms are acting simultaneously to selectively address attention to objects in the scene [19] [20] [21] [22] [23] [24]: a bottom-up stream and a top-down modulation. In the following list we summarize the fundamental aspects of the attention mechanisms:

- Some stimuli are intrinsically salient in a specific context. This kind of pre-attentive process is driven in a bottom-up way and is very fast (25 - 50 ms per item). This perceptual saliency critically depends on the surrounding context: what seems to be important is the feature contrast with respect to the contextual surround in a task-independent way.

- An efficient strategy for the bottom-up control is building a master 'saliency map' that topographically encodes the stimulus conspicuity. It seems that is coded explicitly in the cortex but neural analogues of the saliency map are being found at multiple locations (in a distributed

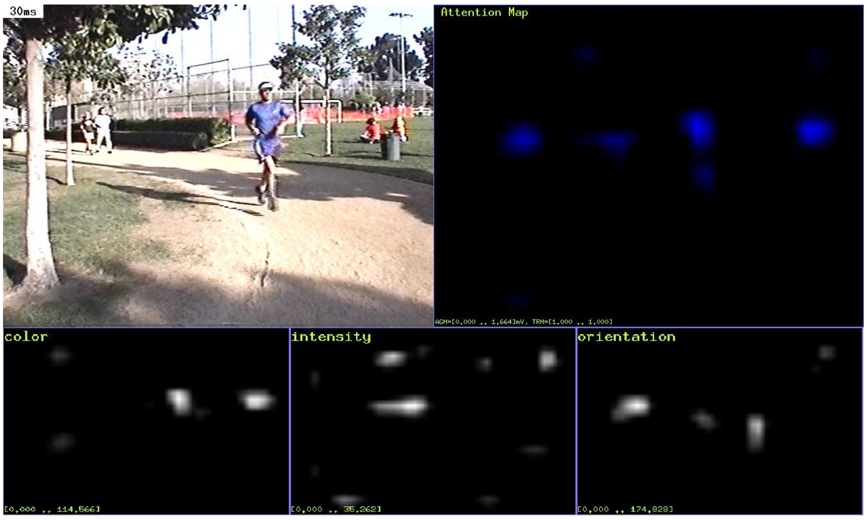


Figure 3: Example of attention application. The figure shows a real-world scenario and a saliency map with the most salient locations (in blue). The second row shows the maps for orientation, energy and color differences from which the saliency map is generated (see c.f. [6]).

manner). The challenge here is how to integrate these many maps into a unitary representation.

- The second form of attention is deliberate or voluntary. It is driven in a top-down manner and the selection criteria change depending on the task. It is more powerful than the first one but also slower (about 200 ms per item). Most of the works model the first bottom-up process but lack modeling the latter due to its complexity and hard generalization.

- The inhibition of return is crucial. This process disables the currently visited location to not to attend it again. Furthermore, inhibition of return (IOR) has been described as a complex, object-based and dynamically adaptive process [25].

- The relationship between attention and eye movement or saccades (coordinate system). This part is usually avoided by the models due to its computational complexity.

As mentioned, visual attention is helpful in a lot of applications such as video-surveillance, industrial inspection or, as shown in Fig. 5, in advanced driving assistance systems or in Fig. 4, in devices for low-level vision patients.

In our case, we initially present Section 5.1 that proposes a bio-inspired model which uses retinal cell responses. In this approach, cues are firstly

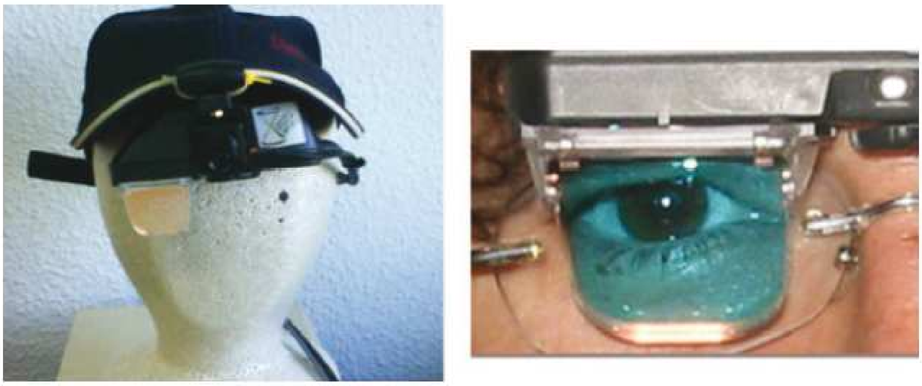


Figure 4: Example of attention application. Figure shows an optoelectronic aid device for low-vision patients. The system includes a camera (scene acquisition) and a wearable processing device that translates the images into useful cues for specific tasks. The attention system may be integrated with this kind of devices to help low-vision patients with tasks such as housework, job controlled environments or even driving scenarios.

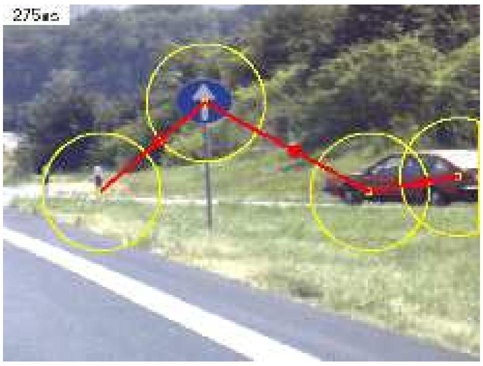


Figure 5: Example of visual attention application: traffic signs detection (c.f. [6]).

ordered using an energy function to pre-select the most relevant ones. Then, the method deploys a simplified top-down attention mechanism that is driven by using color, texture or motion information. The modulation stream suppresses or activates cues according with different task-dependent criteria. Moreover, we also show how the selected cues are used to efficiently estimate an accuracy-improved motion in some areas of interest.

Secondly, we also present an alternative in Section 5.4, a complete cognitive computational model that combines the bottom-up and the top-down forms of attention. This model also includes the inhibition of return mechanism. This more-complex approach is successfully applied in driving scenarios.

Finally, the adaptation and the synchronization elements that are also explored in this thesis, completing the components of our general field of study, the dynamic vision systems.

1.2 State-of-the-art

In the literature we may find a few examples of dynamic vision systems but the works developed by E. Dickmanns deserve special attention, specially [10] [26] [27]. All these real-time approaches have been very useful and powerful in solving problems in driving scenarios. This framework adds more complexity thus, these systems deal with dynamic environments observed from moving platforms.

Focusing on the active vision component, we also find various proposals but most of them handle individually the different elements of active vision more than presenting an integrated contribution. Initially, there are different proposals for the design and implementation of systems for the active control of gazing such as [28] that controls vergence, version and tilt. Or [29] that includes visuo-motor control for robotic applications but with lack of implementation details, and [30] that describes a complete mechanical system for gaze control but limited to the eye movement. Furthermore, in [31] authors propose a more complex system that controls gaze using the vestibulo-ocular reflex and vergence control, emulating also the smooth pursuit mechanism. Finally, we also find works that deals with this problem for autonomous vehicles, designing models for Multifocal Saccadic Vision (EMS-Vision) as in [32]. Another group of works also includes foveation to gaze control, as in [33] [34].

Focused on the visual attention models, we may find plenty of works. Selecting only the most important works of each category we highlight [3] from the Information Theoretic models, [35] [36] from the Cognitive models, [37] in the group of Bayesian models. Most of the works in the literature are included in one of these three sets.

Finally, in the literature there are a few very-interesting approaches that integrates active gaze control and attention into a unique system. In the case of [38], the work includes in the same work active gaze control with foveation and attentional mechanisms based on the idea of maps of interest. It also includes a complete benchmarks using images with one isolated target, with multiple objects, and natural scenes. The main problem of this system is that the attentional method is simply based on edges, and gaussian filters applied to retinal images. Furthermore, it does not include eye and head movement control. In [39] authors propose a very complex system that integrates active gaze control and attention in the framework of grasping applications. The system combines several cues to segment objects and use attention and the saccades information in the recognition task. The work details the model for attention that includes a bottom-up and a top-down stream completed by a reasoning mechanisms which drive the modulation stream. The only problem is that the work does not explain the active gazing mechanism, lacking details about the fixation phase, or the eye and head motion controls.

More detailed specific state-of-the-art sections are included in each Section of this Ph.D. dissertation.

Introducción en Castellano

Uno de los aspectos más importantes del sistema de visión humano es su habilidad para procesar una gran cantidad de información que recibimos cuando abrimos los ojos y su capacidad de adaptación a los medios dinámicos que nos rodean. Los cambios en el entorno pueden ser debidos por ejemplo a variaciones en la iluminación, objetos que cambian o se mueven, o a un cambio en el punto de vista. La forma en que somos capaces de manejar eficientemente esa información es muy interesante para muchas aplicaciones. Además, el sistema de percepción visual traduce imágenes en información cognitiva, lo que resulta en un proceso realmente complejo. A nivel conceptual, se estructura como un conjunto capas que procesan información a diferentes niveles de abstracción [1]: el bajo nivel, con la extracción de las modalidades visuales tales como el movimiento, la profundidad, la energía y orientación locales, etc; el medio nivel, que combina las modalidades extraídas previamente o segmenta la información; y el alto nivel, que incluye la compresión de la escena. Además, el sistema también incluye lazos de realimentación utilizados para modificar el procesamiento de la imagen a diferentes niveles con respecto a las características o modalidades extraídas con una alta fiabilidad en las diferentes etapas. De acuerdo con todo esto, algunas preguntas acerca de la naturaleza emergen tales como ¿por qué algunos patrones son más difíciles de detectar, como los enemigos camuflados en una batalla? O, ¿por qué es más relevante o prominente una señal de tráfico que su entorno? Y, ¿por qué una presa corriendo es rápidamente detectada por un predador? O desde un nivel más abstracto ¿cómo es el proceso de atender a lo que es más importante para nuestra supervivencia en un medio muy dinámico? ¿Cómo refinar la información de una cierta área para cumplir con nuestros requerimientos? ¿Cómo es la conexión entre estos procesos y la tarea que se está llevando a cabo? Y finalmente, desde un punto de vista de la ingeniería, ¿cómo podemos imitar este proceso para explotar sus ventajas?

El mundo real que nos rodea es cambiante y, por tanto, los sistemas biológicos han desarrollado estrategias para lidiar con él. Trabajar con medios dinámicos es crucial para los sistemas de visión artificial que trabajan en el mundo real. Pueden llevar a cabo tareas en las cuales el movimiento de los objetos, el de la cámara o cambios en la iluminación se están produciendo. El análisis de la visión dinámica tiene tres niveles o etapas diferentes [2]. En primer lugar, la etapa periférica se refiere a la extracción de información que será usada en las siguientes capas y busca las áreas más relevantes de la escena. La etapa atencional se centra en la computación en las áreas seleccionadas previamente y extrae información para por ejemplo, el reconocimiento de objetos o el movimiento. Finalmente, la etapa cognitiva usa el conocimiento apriori que posee sobre los objetos, las condiciones ambientales, o las relaciones entre los elementos de la escena. Esta etapa analiza el estado presente de los elementos de la misma y explora su comprensión para analizar los eventos que se están produciendo.

Los procesos perceptivos son generalmente activos, lo que significa que los procesos de percepción visual no consisten únicamente en la adquisición de imágenes y su procesamiento posterior, sino que incluyen mecanismos de selección activos de la información más relevante, modulación de filtros dinámica, atención, y control activo de la mirada. La visión activa es definida como el problema de aplicar las estrategias de control inteligente a los datos que dependen de su estado actual (R. Bajcsy en 1985 [3]). En este marco, la adquisición de la imagen y su procesamiento ya no son independientes. La visión activa y, por tanto, el control de la mirada, la atención y el reconocimiento, son necesarios en diversas tareas como en la manipulación, por ejemplo centrando la búsqueda en las regiones ocultas después de un cambio en la fijación; la exploración, buscando nuevas áreas que visitar; desambiguación, generada por un movimiento o por cambios en la iluminación; adaptación, para seleccionar los valores de los parámetros que define los filtros dinámicos para conseguir la configuración más optimizada para un escenario dado, etc. En un escenario mayormente estático, todos estos mecanismos son simulados de forma separada utilizando diferentes modelos pero, en el caso de los escenarios dinámicos, el procesamiento en tiempo real es necesario. En este caso, necesitamos enfrentarnos a la optimización del sistema para reducir el tiempo de procesamiento. De esta forma, se requiere computar las características más dinámicas de la escena de una forma más eficiente mientras que la información estática puede ser computada por procesos más complejos que tarden más.

En particular, la atención se define como el conjunto de mecanismos de control que el sistema de visión usa para llevar a cabo búsquedas con unos recursos reducidos. Puede ser vista como un mecanismo adaptativo que maneja los recursos de procesamiento limitados que poseemos, en un medio dinámico. Hay tres tipos de mecanismos [3]: la selección de la región espacial dependiendo de una respuesta o un estímulo, la restricción debida a la dependencia con la tarea específica que se va a desarrollar y finalmente, el proceso de supresión llevado a cabo mediante la inhibición del entorno y la inhibición de retorno que impide revisitar las áreas seleccionadas en el primer caso. Estos tres mecanismos son llevados a cabo como una combinación de un flujo de datos de saliencia que funciona de abajo a arriba, junto con uno de modulación que es dependiente de la tarea y que funciona de arriba a abajo [3]. Además, la atención se relaciona normalmente con dos problemas que han sido estudiados ampliamente en la literatura: reconocimiento y conexión ("binding"). No hay un consenso real sobre la relación entre atención y reconocimiento. La atención se cree que funciona antes que el reconocimiento pero algunas teorías también proponen que la atención está de una forma particular atada a los objetos o partes de objetos [4] [5]. Finalmente, la conexión se define como el proceso que une una característica para un objeto como por ejemplo, la forma con su localización.

La meta de esta tesis es tomar todas esas ideas acerca de los sistemas de visión dinámicos, y en particular sobre la visión activa y los mecanismos atencionales y desarrollar sistemas empotrados con recursos limitados para aplicaciones del mundo real. Estas aplicaciones requieren adaptación inteligente al entorno dinámico y atención visual para optimizar el uso de los recursos disponibles. La tecnología actual ofrece diferentes alternativas para su diseño y simulación: implementaciones optimizaras para ordenadores estándar, GPU, DSP, ASIC, o FPGA. Las FPGAs nos permiten implementar arquitecturas específicas para aplicaciones con procesamiento masivamente paralelo, con bajo consumo de potencia y coste y tamaño reducidos. Todas estas propiedades hacen a estos dispositivos reconfigurables los más apropiados de acuerdo con nuestros requerimientos.

Además, su fácil integración las hace bastante apropiadas para plataformas industriales tales como robots móviles, vehículos, plataformas de inspección industrial, o cámaras inteligentes. Finalmente, una implementación del sistema visual en un dispositivo empotrado nos ofrece un sistema de prestaciones en tiempo real, un atributo muy interesante desde el punto de vista neurobiológico. Un sistema de altas prestaciones permite la exploración y el teste de estrategias de visión activa en el marco de ciclos cerrados de acción-percepción, que no son posibles en un sistema que no se esté ejecutando en línea.

Aunque el propósito de esta tesis no puede ser tan ambicioso como para contestar todas las preguntas que se formularon previamente, nuestro principal objetivo es intentar responder algunas de ellas con la implementación de un modelo computacional complejo implementado en un dispositivo empotrado. En concreto, hemos diseñado una implementación para la computación de la saliencia usando el bien conocido modelo de Itti y Koch [6]. Este método es ampliamente usado en la literatura como métrica contra la que comparar las nuevas implementaciones [7] [8] [9]. El modelo básico basado en un conjunto cerrado de modalidades visuales es enriquecido en nuestro caso con la estimación de movimiento. Además, también incluimos una modulación que va de arriba a abajo y que es dependiente de la tarea que se está llevando a cabo, que permite adaptar el sistema a los parámetros de su entorno. Esta modulación es generada usando el conocimiento que se tiene de la escena, incluyendo las estimaciones de la profundidad local de los objetos.

En nuestro caso, podemos llevar a cabo esta integración de nuestro sistema hardware en dispositivos para diferentes aplicaciones: en el campo de la video-vigilancia el sistema puede ser interesante integrado en cámaras reales o como un sistema independiente en tiempo real que advierte sobre las situaciones sospechosas; en la industria del automóvil hay muchos trabajos que usan sistemas de atención en asistencia avanzada para la conducción como, por ejemplo en el reconocimiento de señales o de su localización; en el campo de la robótica, un sistema como el nuestro puede ayudar en la navegación autónoma de plataformas móviles como por ejemplo, en vehículos espaciales robóticos donde la transmisión de los comandos en muy lenta o, en medios muy cambiantes en los que la inspección visual de un operador no es suficiente para el control de la navegación; en aplicaciones militares, para detectar objetivos camuflados en entornos complejos; en medicina, integrado en sistemas de ayuda para pacientes de baja visión (afectados por enfermedades de la vista) como por ejemplo, en dispositivos que se colocan en la cabeza a modo de gafas y que mejoran la independencia del paciente ayudándolo en tareas como la conducción, en tareas diarias de su trabajo o del hogar. En esta tesis hemos evaluado satisfactoriamente el potencial de nuestro sistema en aplicaciones en el campo de la asistencia avanzada a la conducción.

Sistemas dinámicos de visión

Los sistemas de visión artificial han desarrollado diferentes estrategias para aplicaciones que trabajan en medios dinámicos del mundo real. Los cambios en el punto de vista de las cámaras, objetos que se mueven o que cambian, o condiciones de iluminación diferentes hacen esencial que se diseñe soluciones específicas.

Como se ha comentado en la introducción, un sistema de visión dinámico no es simplemente un sistema que aúna atención visual y control de la mirada (visión activa). También incluye más componentes que están relacionados con (para ver más detalles consulta [10]): la base de conocimiento, tareas, modelos, adaptación y sincronización. La Fig. 6 muestra los componentes de un sistema de visión dinámico.

El elemento más complejo en un sistema de visión dinámico es la visión activa y será tratada en detalle en la Sección 1.2. En la Fig. 6 podemos ver una flecha que apunta a la visión activa y que representa la información del mundo real para la exploración que se quiere llevar a cabo. Esta relación representa la relevancia del movimiento o la acción en el esquema general, porque ésta genera la información visual, haciendo al sistema propiamente 'activo'.

A continuación, encontramos uno de los componentes también más importantes: la base de conocimiento. Este elemento resume todo el conocimiento que el sistema necesita para comprender la escena, como por ejemplo los objetos que hay en la misma y sus categorías, sus relaciones, sus comportamien-

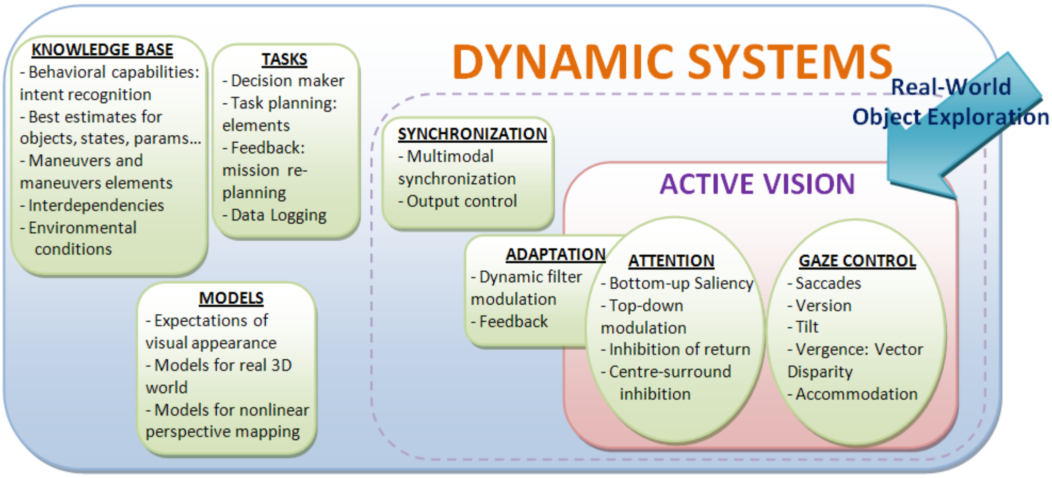

DYNAMIC SYSTEMS
KNOWLEDGE BASE
- Behavioral capabilities: intent recognition
- Best estimates for objects, states, params...
- Maneuvers and maneuvers elements
- Interdependencies
- Environmental conditions
TASKS
- Decision maker
- Task planning: elements
- Feedback: mission re-planning
- Data Logging
MODELS
- Expectations of visual appearance
- Models for real 3D world
- Models for nonlinear perspective mapping
SYNCHRONIZATION
- Multimodal synchronization
- Output control
Real-World Object Exploration
ACTIVE VISION
ADAPTATION
- Dynamic filter modulation
- Feedback
ATTENTION
- Bottom-up Saliency
- Top-down modulation
- Inhibition of return
- Centre-surround inhibition
GAZE CONTROL
- Saccades
- Version
- Tilt
- Vergence: Vector Disparity
- Accommodation

Figure 6: Principales componentes de los sistemas dinámicos de visión (inspirando en [10]).

tos para la locomoción y la visión, o las condiciones del entorno. Además, teniendo en cuenta la tarea que se está llevando a cabo, también se necesita conocimiento sobre las mejores estimaciones para los estados actuales y futuros de los objetos antes mencionados, sus parámetros, o sus posibles acciones futuras a partir del estado presente.

Los modelos conforman otro de los componentes. Éstos principalmente lidian con los métodos que traducen la información real tridimensional a representaciones bidimensionales que las aplicaciones de visión artificial normalmente manejan. Esto también incluye los modelos que representan los estados futuros sobre la apariencia visual de los objetos en la escena, teniendo en cuenta el conocimiento provisto por los mismos previamente.

El componente de las tareas incluye la información sobre planificación. La misión que se tiene que cumplir se descompone en tareas más simples. Este componente resume el conocimiento sobre la forma de resolver cada una de esas tareas simples, los estados que se alcanzan tras ello y la posible re-planificación a través de lazos de realimentación, o las relaciones de los objetos tras cada una de ellas.

Hasta este punto, todos los elementos que se han comentado son comunes con cualquier sistema de visión. Los sistemas dinámicos incluyen también dos componentes específicos más (teniendo en cuenta también el de visión activa): adaptación y sincronización. La adaptación es necesaria en los sistemas de visión activa y en general, distribuidos en diferentes componentes de los sistemas dinámicos de visión. La adaptación cambia dinámicamente los parámetros o los métodos en los diferentes elementos para optimizar los resultados y llevar a cabo diferentes tareas. La sincronización monitoriza el correcto funcionamiento de los diferentes elementos y minimiza los retrasos entre la extracción de la información y el planificador de tareas. Este componente también sincroniza el conocimiento multimodal y las estimaciones extraídas por los diferentes elementos.

Visión activa

El término 'activo' es usado en contraposición a 'pasivo' y une la visión con la acción, en otras palabras, hace más clara la relación entre la percepción visual y la acción. Un sistema de visión activo es capaz de gestionar sus propios recursos eficientemente para llevar a cabo una tarea [11]. Este tipo de sistemas utilizan sensores pasivos y activos, pero el sistema puede actuar sobre el punto de vista o sobre los parámetros de la cámara. Este sistema es también capaz de gestionar eficientemente los recursos disponibles dedicándolos a áreas que se definen como más relevantes o procesando las que son más importantes para cumplir con una tarea específica. Por tanto, el elemento de visión activa puede modificar la posición de la cámara y sus propiedades, o la fóvea en sistemas completamente bio-inspirados. Todo esto se incluye en lo que llamamos control activo de la mirada. En segundo lugar, también tiene que manejar eficientemente los recursos para seleccionar la información más relevante de la escena, lo que se considera parte del campo de la atención visual. Un esquema general de los sistemas de visión dinámicos es detallado en la Fig. 7.

Control activo de la mirada

El control de la mirada es el proceso que permite dirigir la fijación a diferentes partes de la escena conducidos por la tareas que se va a desarrollar a nivel cognitivo, perceptual o de comportamiento [12]. La información dentro de la pequeña región en la que se centra la mirada (la correspondiente a la fóvea) es adquirida con la mayor calidad. Por tanto, cuando queremos percibir en detalle un objeto determinado, tenemos que centrarlo en la fóvea. De hecho, movemos los ojos 3 ó 4 veces cada segundo para poder llevar a cabo este proceso en diferentes partes de la escena. Estos movimientos rápidos son llamados 'sacádicos'.

De acuerdo con [12], el control de la mirada es crucial porque la percepción visual es activa y por tanto, el sistema visual busca la información más relevante en la escena de acuerdo con la tarea que se va a llevar a cabo. Además, los ojos también juegan un papel central en el proceso cognitivo de atención y son usados como una ventana de exploración sobre la escena global.

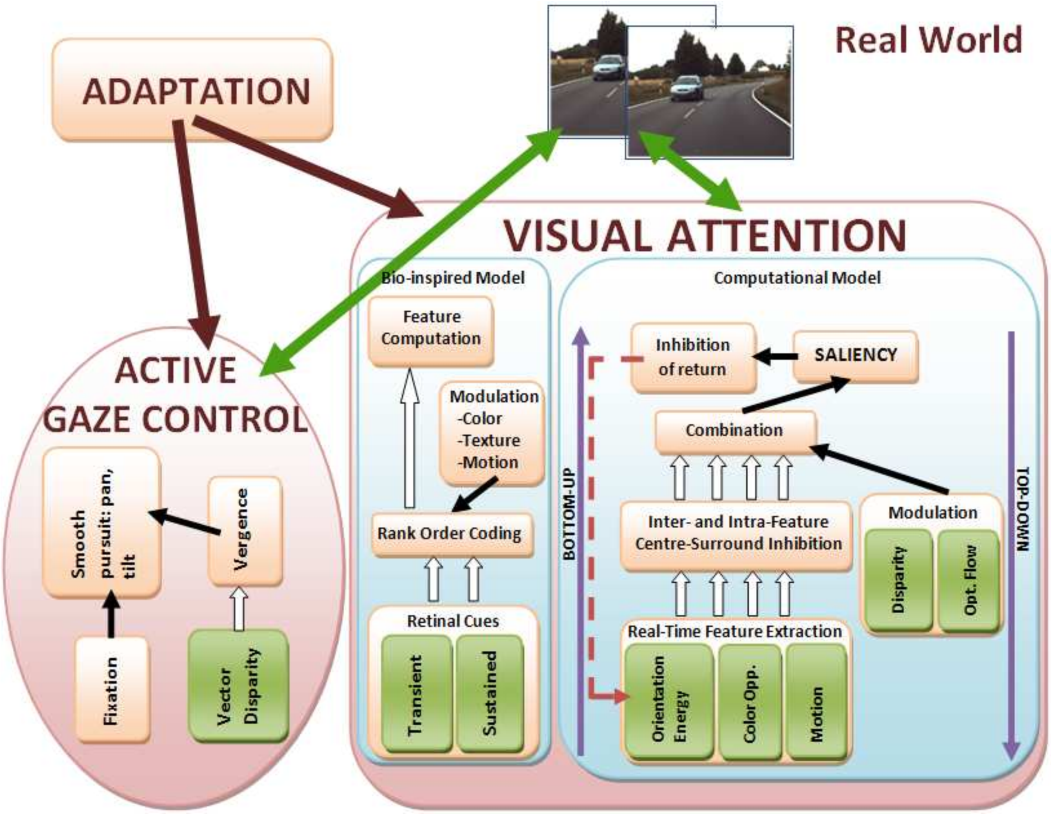


Figure 7: Esquema general del sistema de visión dinámico implementado para esta tesis. La visión activa consiste en conjugar el control de la mirada y el proceso cognitivo atencional. En esta tesis exploramos diferentes sistemas e implementaciones para con control activo de la mirada en tiempo real y dos modelos diferentes para la atención visual.

El sistema de control de la mirada nos permite todo el procesamiento previo para conjugar dos sistemas diferentes [13] [14] [15]: el sistema oculomotor que mueve los ojos dentro de las órbitas oculares y el sistema de movimiento de la cabeza que mueve los ojos en el espacio 3D. Hay seis mecanismos diferentes de control neuronal para el control de la mirada:

- *Movimientos sacádicos*: Movimientos rápidos que mueven la fóvea a un objeto de interés que se encuentra en la periferia.
- *Vergencia*: Este mecanismo permite cambiar la fijación para diferentes planos de profundidad.
- *Seguimiento suave*: Este sistema mantiene centrado en la fóvea un objeto de interés que se mueve suavemente.
- *Fijación*: Este proceso ayuda al ojo a mantenerse estático en la órbita para centra un objeto que es estático.
- *Reflejo vestíbulo-ocular (VOR)*: Este mecanismo es usado para estabilizar el ojo durante los movimientos de la cabeza y ayuda a mantener estáticos los objetos en la retina durante los mismos.
- *Reflejo optokinético (OKR)*: Este mecanismo es también usado para estabilizar el ojo durante los movimientos de la cabeza y este caso, mantiene las imágenes estáticas en la retina cuando se producen rotaciones de la cabeza.

En esta tesis abarcamos parte del control activo de la visión, desarrollando un sistema que controla tres de los mecanismos mencionados anteriormente: la fijación, el seguimiento suave y el control de la vergencia.

Atención visual

Quizás la definición más plausible de atención es, como señala el Diccionario Médico Websters, "La habilidad para centrar selectivamente en un estímulo seleccionado, manteniendo ésta y desplazándola a voluntad. La habilidad para concentrarse en algo". La Etimología puede también ayudar con la definición de la atención. Viene del latín "attentus", el participio de "attendere" que significa "hacer caso" o "prestar atención, atender". Varios autores han intentado entenderlo como en el caso de Malebranche que relacionaba la atención con la conservación de la evidencia en nuestro conocimiento o Leibnitz que estableció que la atención es necesaria para percibir objetos conscientemente. Además, en el siglo XIX, la influencia de la atención se ha visto como un refuerzo (Müller 1873, Exner 1894) o una inhibición (Wundt 1902, Exner 1894).

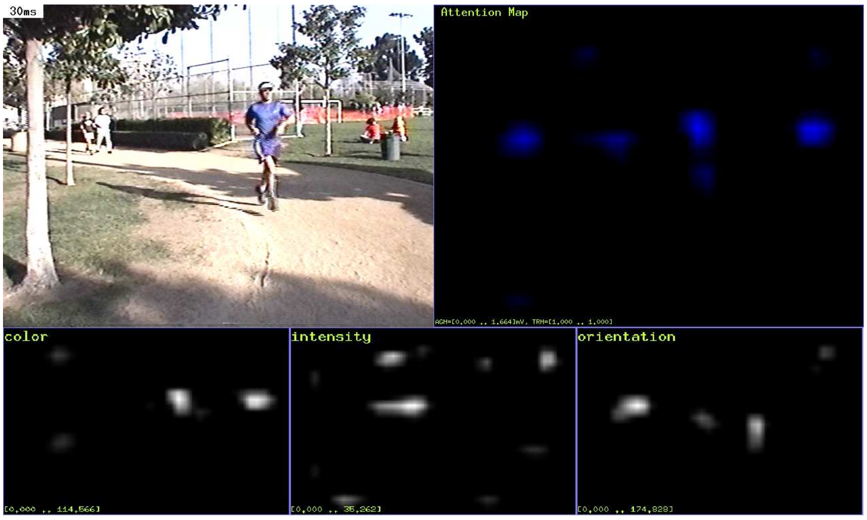


Figure 8: Ejemplo de aplicación de atención visual. La figura muestra un escenario del mundo real y un mapa de saliencia con las localizaciones más relevantes en azul. La segunda fila muestra los mapas de orientación, energía y diferencias de color con los que se ha construido el mapa de saliencia superior (ver [6]).

La visión nos permite interaccionar con el mundo real dinámico que nos rodea con nuestros recursos limitados (un ejemplo de atención visual en un escenario del mundo real es mostrado en la Fig. 8). En muchos sistemas biológicos, la estrategia para inspeccionar una escena siempre parece secuencial. En el caso de los primates, puede ser contradictorio el uso de una estrategia secuencial teniendo en cuenta las estructuras de procesamiento paralelo avanzadas de las que disponemos en el cerebro. El problema es que esa ingente cantidad de información que se transmite por el nervio óptico no puede ser procesada por el cerebro con esta arquitectura paralela debido a la limitación de sus recursos. La única forma de solucionar este problema consiste en seleccionar de manera secuencial las áreas de interés de la escena que tienen que ser atendidas o procesadas primero y entonces, desplazar la atención a la siguiente localización y así sucesivamente. De hecho, como Itti comenta en [16] no vemos todos lo que nos rodea, tal y como es comúnmente aceptado.

De acuerdo con las clasificaciones de Rothenstein y Tsotsos [17], nuestras propuestas son incluidas en la categoría de “Modelos computacionales” y dentro de ésta, en la de “Modelos cognitivos” (ver [18]) porque están inspirados por conceptos cognitivos.

En el proceso de atención visual, dos mecanismos diferentes actúan de forma simultánea para dirigir la atención de forma selectiva a ciertos objetos en la escena [19] [20] [21] [22] [23] [24]: un flujo que funciona de abajo a arriba y una modulación que lo hace de arriba a abajo. En la siguiente lista resumimos los aspectos fundamentales de los mecanismos de atención:

- Algunos estímulos son intrínsecamente salientes para un contexto específico. Este tipo de proceso pre-atencional es dirigido desde abajo hacia arriba y es muy rápido (25 - 50 ms por objeto). Esta saliencia perceptual depende de forma crítica del contexto que rodea a los objetos: lo que parece importante es el contraste de una determinada propiedad con su entrono, y no la respuesta o propiedad en sí, pero sin que haya dependencia con la tarea que se va a desarrollar.
- Una estrategia eficiente para el control de abajo a arriba consiste en construir un mapa centralizado 'mapa de saliencia' que codifica topográficamente los estímulos más relevantes. Parece que es codificado explícitamente en el córtex cerebral pero también se encuentran diferentes analogías neuronales del mapa de saliencia en múltiples localizaciones del cerebro (de forma distribuida). El reto aquí es cómo se integran todos esos mapas en una representación unitaria.
- La segunda forma de atención es deliberada o voluntaria. Es dirigida desde arriba a abajo y los criterios de selección cambian dependiendo de la tarea. Es mucho más poderoso que el anterior pero también más lento (sobre 200 ms por cada objeto). La mayoría de los trabajos modelan el primer proceso de abajo a arriba pero no desarrollan éste último debido a su complejidad y a su difícil generalización.
- La inhibición de retorno es crucial. Este proceso inhibe la localización que acaba de ser visitada para que no sea atendida de nuevo. Además, la inhibición de retorno (IOR) ha sido descrita como un proceso complejo, basado en objetos y dinámicamente adaptativo [25].
- La relación entre la atención y el movimiento de los ojos o sacádicos (sistema coordinado). Esta parte es normalmente evitada por los modelos debido a su alta complejidad computacional.

Como se comentó previamente, la atención visual ayuda en un montón de aplicaciones tales como la video-vigilancia, la inspección industrial o, como se muestra en la Fig. 10, en sistemas de asistencia avanzada a la conducción, o en la Fig. 9, en dispositivos de ayuda para pacientes de baja visión.

En nuestro caso, inicialmente presentamos la Sección 5.1 que propone un modelo bio-inspirado que usa las respuestas de células retinales. En esta aproximación, la información es en primer lugar ordenada utilizando una función de energía que permite ya pre-seleccionar las respuestas más relevantes y fiables. Después, el método genera un mecanismo de atención de

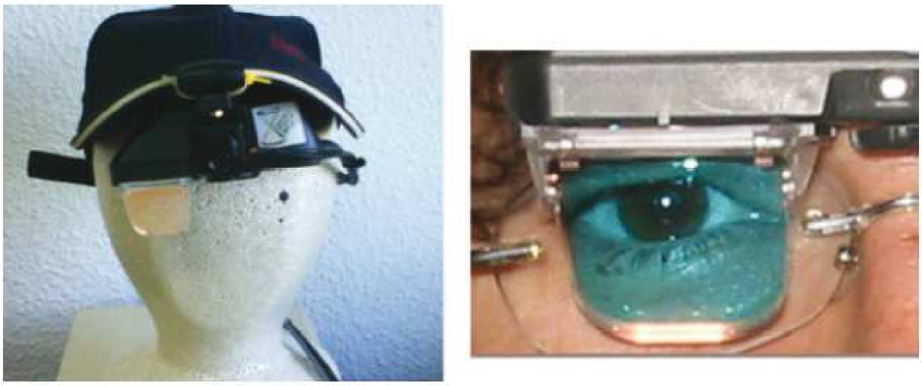


Figure 9: Ejemplo de aplicación de atención visual. La figura muestra un dispositivo de ayuda opto-electrónico para pacientes de baja visión. El sistema incluye una cámara (para la adquisición de las imágenes) y un dispositivo que se coloca sobre la cabeza a modo de gafas y que traduce las imágenes en estimaciones que son útiles para determinadas tareas. El sistema de atención puede ser integrado con este tipo de dispositivos para ayudar a pacientes afectados por baja visión con tareas tales como el trabajo del hogar, actividades laborales en entornos controlados o incluso en la conducción.

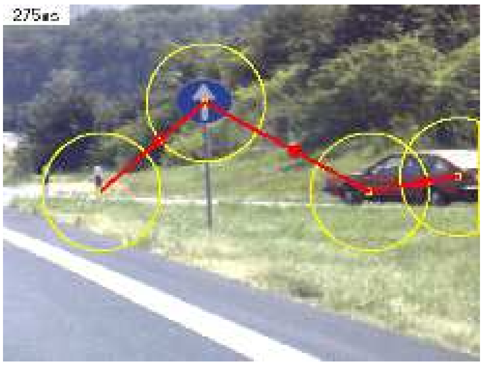


Figure 10: Ejemplo de aplicación de atención visual: detección de señales de tráfico (ver [6]).

arriba a abajo que está dirigido por color, textura o información sobre la velocidad de los objetos. Este flujo de modulación elimina o activa información de acuerdo con el criterio de la tarea que se va a llevar a cabo. Además, también mostramos como la información seleccionada es usada para estimar eficientemente flujo óptico de precisión mejorada en algunas áreas de interés.

En segundo lugar, también presentamos una alternativa a este modelo en la Sección 5.4, un modelo computacional cognitivo complejo que combina un flujo de saliencia de abajo a arriba y una modulación de arriba a abajo. Este modelo también incluye un mecanismo de inhibición de retorno. Este método más complejo se aplica con éxito a escenarios de conducción de vehículos.

Finalmente, los elementos de adaptación y de sincronización que son también explorados en esta tesis, completan los componentes de nuestro campo de estudio general, los sistemas de visión dinámicos.

Estado del arte

En la literatura podemos encontrar muy pocos ejemplos de sistemas de visión dinámicos pero los trabajos desarrollados por E. Dickmanns les prestan especial atención, especialmente [10] [26] [27]. Todas estas aproximaciones en tiempo real has sido muy útiles y valiosas resolviendo problemas en escenarios de conducción. Este marco añade más complejidad si cabe, puesto que estos sistemas tienen que lidiar con entornos dinámicos y cambiantes observados desde plataformas móviles.

Si nos centramos en los componentes de visión activa, también encontramos varias propuestas pero la mayoría de ellas tratan individualmente los diferentes elementos de la visión activa más que presentar una contribución realmente integrada de todos ellos. Inicialmente, hay diferentes propuestas para el diseño y la implementación de sistemas de control activo de la mirada tales como [28] que controla la vergencia, el movimiento sobre el eje vertical y horizontal de los ojos. O [29] que incluye control visuo-motor para aplicaciones robóticas pero sin dar detalles de implementación, y [30] que describe un sistema mecánico completo para el control de la mirada pero limitado al movimiento de los ojos. Además, en [31] los autores proponen un sistema más complejo que controla la mirada usando el reflejo vestíbulo-ocular y la vergencia, y que también emula el mecanismo neuronal del seguimiento suave. Finalmente, también encontramos trabajos que tratan este problema para vehículos autónomos, diseñando modelos para Visión sacádica multifocal (EMS-Vision) como en [32]. Otro grupo de trabajos también incluyen la simulación de la fóvea como en [33] [34].

Centrándonos en los modelos de atención visual, podemos encontrar muchos trabajos. Seleccionando los más importante de cada categoría podemos destacar [3] dentro de los modelos teóricos de información, [35] [36] dentro de los cognitivos, [37] en el grupo de los Bayesianos. La mayoría de los trabajos en la literatura se incluyen en alguno de estos grupos.

Finalmente, en la literatura encontramos muy pocas propuestas interesantes que integren control activo de la mirada y atención en un único sistema. En el caso de [38], el trabajo incluye en el mismo trabajo control activo de la mirada y emulación de la fóvea junto con un mecanismo atencional basado en la idea de mapas de interés. También incluye un complejo conjunto de test usando imágenes de un único objetivo aislado, con múltiples objetivos y con escenas naturales. El principal problema de este sistema es que el método atencional está simplemente basado en bordes y filtros gaussianos que se aplican a imágenes retinales. Además, no incluye control del movimiento de los ojos o de la cabeza. En [39] los autores proponen un sistema muy complejo que integra control activo de la mirada junto con atención en el marco de aplicaciones para coger objetos. El sistema combina diferentes respuestas para segmentar objetos y usa atención y la información de los sacádicos en tareas de reconocimiento. El trabajo detalla el modelo para atención que incluye un flujo de abajo a arriba y otro de modulación de arriba a abajo completados con mecanismos de razonamiento que dirigen la modulación. El inconveniente de este trabajo es que no explica los mecanismos de control de la mirada y faltan detalles sobre la fase de fijación, o el control de los movimientos de los ojos o la cabeza.

Secciones más detalladas de los estados del arte son incluidas en cada sección de esta tesis.

2 Scientific objectives

The development of real-time vision systems allows us to explore dynamic perception schemes which is not possible when images are being processed off-line. The study of the dynamic vision systems in real-time is a fundamental part of this thesis dissertation. The implementation of specific-purpose architectures for real-time vision using devices as FPGAs is becoming feasible and popular thanks to the continuous advance in this kind of technology. Furthermore, the development of embedded vision systems in real-time is a challenging problem with potential perspectives for application in industrial environments such as the automotive industry, robotics, video-surveillance or medicine.

In particular, the objectives we aimed during the elaboration of this Ph.D. dissertation are listed bellow:

- **The implementation of circuits of low-level vision for high-performance embedded systems**. This objective consists in the study and development of the different hardware cores for the estimation of motion, depth and local energy and orientation. We encompass this objectives from different points of view:
  - *From a hardware implementation point of view.* Exploring multiscale gradient-based methods for the hardware implementation of optical flow, disparity and local contrast descriptors. Additionally, it includes benchmarking the accuracy, density and computational power in terms of frame-rate.
  - *From the resource point of view.* The implementation of a multiresolution scheme, as a way to avoid an expensive multiscale-with-warping architecture for the optical flow computation used for fulfilling the previous objective. The purpose is the exploration of alternatives for fusing very efficiently estimates that come from different spatial resolutions, with a lower cost than the multiscale-with-warping strategy. Additionally, a study of the parallelization of this strategy and its suitability for hardware implementation is also required for reaching real-time performances.
  - *From a color-based implementation point of view.* Despite in biological systems there are no evidences of color information in the motion or depth computation, from a pure-engineering approach it is quite worth the addition of color cues to the system to improve its precision. In this case, the objective is the evaluation and benchmarking of the same gradient-based methods for optical flow and disparity but using color cues. Our goal here is comparing the gain due to the addition of the color information

from different color representations, assessing the resource cost increase for real-time embedded systems.

- **The development of a high-performance hardware/software that integrates the previous cores in a single chip**. This objective involves a quite important integration effort, implementing an on-chip architecture for the estimation of motion, depth and local energy and orientation. This objective is aimed as result of our requirement of reducing hardware resources of previous implemented works (see [40] [41] [42] [43]).

- **The exploration and development of embedded active vision architectures for dynamic systems**. Our purpose is the implementation of an architecture that deals with real-world dynamic systems and that includes the active vision aspects:

  - *The development of a system for active gaze control of a robotic platform.* Studying the feasibility of vergence control in a robotic platform with a binocular system, using vector disparity estimation. This system is benchmarked integrated with a real robotic system for the emulation of "fixation" and "smooth pursuit" mechanisms.
  - *The implementation of a bio-inspired system that deploys simple visual attention mechanisms.* The objective is to study the optical flow computation by using sparse signals that an artificial retina generates (as a focal plane processing device). This approach is performed for limited-resource systems inspired in biological attention strategies and using real retinal cues. The attention mechanism is implemented as a top-down modulation system that biases the computation to select the most relevant cues according to different criteria that depends on the task that is being performed or the targets in the scenario.
  - *The implementation of a complete visual attention architecture.* In this case, we require an on-chip adaptable architecture that integrates a saliency bottom-up stream based on motion, local orientation, luminance and color differences (based on [6]) and a top-down task-dependent modulation based on the knowledge of the scene, the depth and also the motion.
  - *The evaluation of applications for the visual attention.* This objective consists in the study of an application for the visual attention system such as video-surveillance, tracking, driving assistance systems, or low-vision aid systems.

3 Project framework

The work presented in this thesis dissertation has been performed mainly in the framework of three research projects: the European project DRIVSCO and the Spanish national projects DINAM-VISION and ARC-VISION. A brief description of each project and the main contributions of this PhD to them are presented in the following subsections.

The ARC-VISION project

The goal of these Spanish national project ARC-VISION (TEC2010-15396) is the on-chip implementation of architectures for specific purpose processing devices for visual perception in real-time.

The main contribution of this PhD to the Spanish national project ARC-VISION has been the development of circuits (defined as cores on FPGA devices) for gradient-based low-level vision extraction (motion, depth, local contrast energy and orientation). Also the implementation of multiscale and multiresolution schemes towards enhancing the working range of the developed motion and depth modules with an affordable increase of the resource cost.

Finally, the development of attention models capable of managing these low-level modalities has been an important contribution to this project.

The DINAM-VISION project

The goal of the Spanish national project DINAM-VISION (DPI2007-61683) is to develop a real-time vision system that is able to dynamically adapt its inherent properties to improve efficiently the information extraction, as for example adapting dynamically the range of the spatio-temporal filters used in low-level vision.

The first stage of this model performs the extraction of the low-level cues: local contrast energy, orientation and phase. A second stage combines the low-level information of the previous stage into multimodal sparse entities. The utilization of massively parallel devices (FPGAs) allows us to obtain real-time processing. The system also enables feed-back loops from the latest stages to the earlier in order to optimize the functionality of the system. With respect to the applications, the system is addressed for driving assistance systems, specially to detect IMOs (independently moving objects) and ego-motion using standard and infrared cameras (see Fig. 11).

As dynamically adaptable systems are required for the implementation of any kind of visual systems capable of dealing with real-world environ-

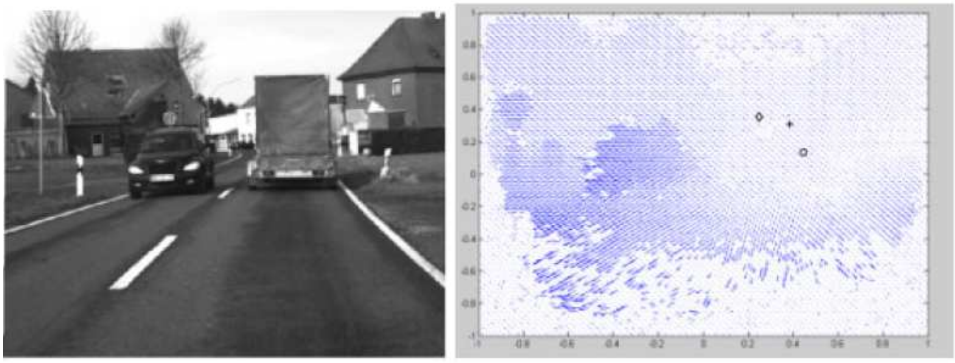


Figure 11: Example of on-chip ego-motion results for driving assistance applications (c.f. [44]).

ments, it is easy to understand the relevance of dynamic systems in fields such as autonomous navigation or generally in robotics (the main application frameworks of this project). The integration of the different low-level modalities in a single chip has been a significant contribution of this PhD in the DINAM-VISION project. The low-level vision modalities were used for different tasks in this project.

The DRIVSCO project

The goal of the European project Drivsco (IST-FP6-16276-2) "Learning to emulate perception-action cycles in a driving scenario" [45], was "to devise, test and implement a strategy of how to combine adaptive learning mechanisms with conventional control, starting with a fully operational human-machine interfaced control system and arriving at a strongly improved, largely autonomous system after learning, that will act in a proactive way using different predictive mechanisms". Thus, its main purpose is learning from the driver behavior and adapting to different patterns for driving habits. For instance, in a potentially dangerous scenario, when detecting an intersection or an obstacle on the road, in difficult weather conditions as heavy raining or during the night, the on-board system in the car generates warnings to the driver that gains time to take an appropriate decision. Fig. 12 shows the prototype system installed in the car.

The objective of our group was the development of an on-chip system for the estimation of optical-flow, depth and local contrast features with complex algorithms based on local phase (for more details, see Fig. 13). The work presented in this thesis is mainly focused on the performed tasks for the on-chip computation of the different visual modalities that are the basis of the visual attention mechanism.

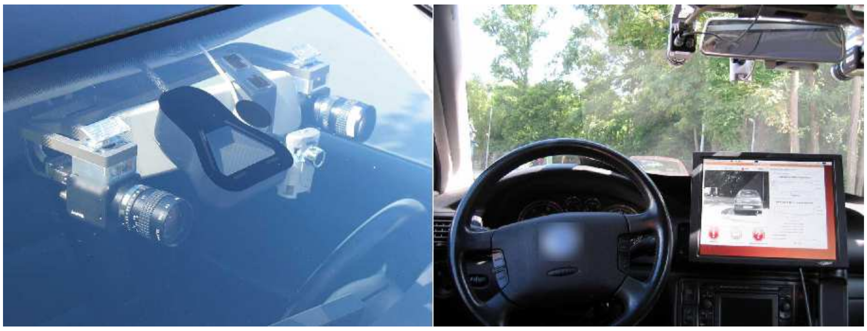


Figure 12: DRIVSCO Project system installed in the car.

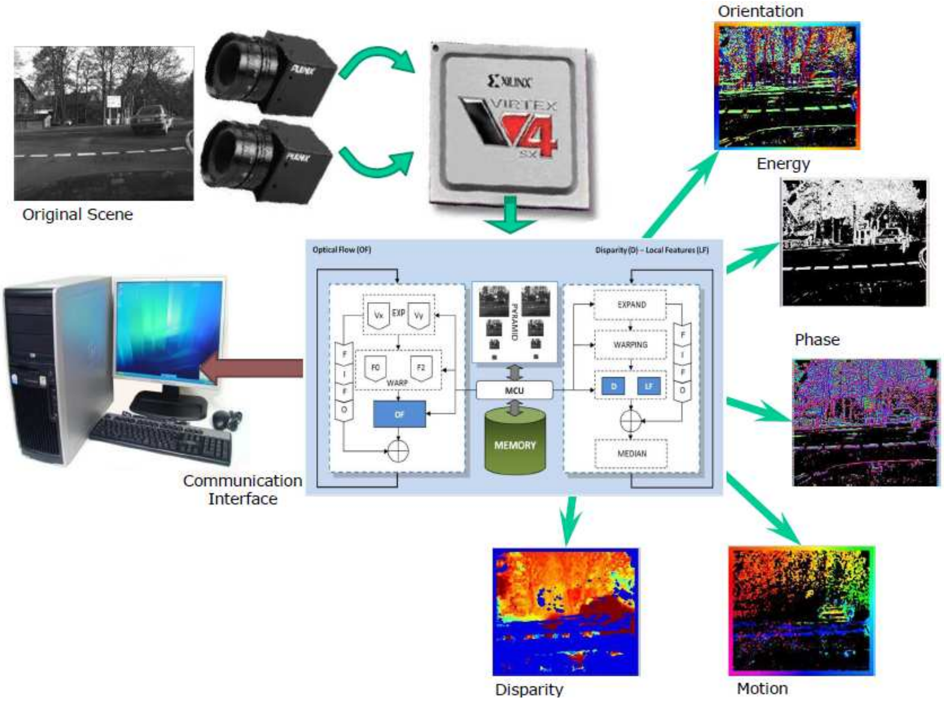


Figure 13: University of Granada final architecture, contribution to the Drivsco Project.

Apart from the dynamic systems, attentional systems also provide some advantages in computing for real-world scenarios. A combination of both may help in quite difficult applications as, for example, advanced driver assistance systems. In this scenario, attention is easily connected with the capability of detecting traffic signs or road surface markings. Hence, attentional systems are a valuable complement for the DRIVSCO system.

4 Methods and tools

The implementation of a visual system, capable of performing certain low- and middle-level perception tasks, is always a very complex problem. In our case, we target real-world application domains, which motivates a strong focus on real-time processing. This requires efficient implementations, optimized image processing toolkits, easy and fast access and manipulation of data stored in matrices. For first model validation stages, MATLAB [46] is a good candidate that satisfies these requirements and moreover, it is a very common tool widely used by the computer vision community.

Hence, once our model has been designed using MATLAB, we validate it in order to reach a good accuracy and sufficient density (if, as in our case, the model provides sparse results). The optimization of the code at this point may provide a fast implementation. For example, the translation into a C-like language and its optimization using parallel processing or data alignment techniques, code reordering, computation in blocks according with the cache memory size, or the use of specific processor directives such as SSE or MMX may be enough for obtaining real-time performances for the computation of visual primitives as in [47].

As mentioned, our final aim is an embedded system that requires low-power consumption and small size (among others). In this case, there are several possible choices such as one that has currently become very popular in computer graphics specially due to the gaming industry: GPUs (graphic processing units). These devices provide a highly-parallel processing structure, with hundreds or thousands of processing cores and a quite good access to fast memory modules together with user-friendly interfaces for programing and libraries of primitives with floating-point operations. There are many works that deal with these devices in computer graphics to implement the extraction of our visual modalities such as in [48] [49] [50] [51] [52]. Their main drawbacks are their difficult integration with for instance, robotic platforms, aircrafts, or smart cameras. It is due to their size and high power consumption, as well as limitations related to safety critical regulations that make impossible to implement, for instance, safety systems on vehicles with GPU-based platforms.

Next, the second step from the MATLAB model is its integration in a more suitable device due to its low-power consumption and cost, small size, and possibility of certification for reliable applications. Our choice to fulfill those requirements are the FPGAs. They also may provide high performances for fulfilling our real-time requirements due to its highly-parallel processing architecture. The main drawback of FPGA implementations is their long learning curve with a very small slope which means a long latency and highly qualified personnel for obtaining the product. The lack of tools for the debugging and validation of complex designs strongly contributes to the referred problem. At the same level than FPGAs, we also find DSPs, that also satisfy the mentioned requirements. They provide also low-power consumption processing but, FPGAs are more appropriate for complex applications that require a very high-performance computing (for more details, see c.f. [53]).

Once the target device has been selected, for the hardware implementation two first steps are required. Firstly, the reorganization of the code to exploit the maximum level of parallelism, it involves for example: block computation, loops unrolling and specially, in our case pipelining. A second step consists in thoroughly studying the bitwidth for the variables in the operations of our model along the datapath, since our work is performed using fixed-point arithmetic and we need to assess the impact of this quantization degradation in the resultant accuracy. We use different tools for these intermediate steps: VHDL for the implementation of interfaces between the hardware cores and the memory management; Handel-C, a C-like programming language facilitate the implementation of algorithmic description of models, for the implementation of the different cores. In both cases, the implementations and the integrations of the complete system is carried out using the Xilinx ISE Design Suite [54] and the DK Design Suite for Handel-C [55].

After the hardware implementation, the resulting core is validated. The validation process includes a first comparison between the hardware results and the ones achieved by the previous quantized software model that emulates hardware. Then, the hardware model is also benchmarked using the well-known ground-truth sequences available at Middlebury site [56]. After that, the hardware model is accepted as a validated implementation that presents a good trade-off between the required accuracy and its resource consumption. Otherwise, the core is refined returning to the steps of the bitwidth study and core programming. This last stage is performed using a set of different applications and tools implemented with Visual Studio .Net [57] and MATLAB for benchmarking, and also Open RTVision [58] a software platform for the final visualization that is detailed in Appendix A.

This last stage is different for the validation of the systems developed in the last Sections. The benchmark of the robotic head is carried out simulating two neural mechanisms that are known to be performed by the gaze control: the fixation and the smooth pursuit. Both of them use a initial phase to select the target using a color and then, we measure the time to achieve the correspondent goal.

In the case of the visual attention, the hardware architecture is also benchmarked using another well-known database of saccades, provided by Itti in [59]. This database consists in different sets of images concerning diverse scenarios, including a driving one. Performances are measured by computing the error as the number of tries to that a system needs to select the most relevant area in the image.

5 Discussion and results

This section presents a brief discussion of the results obtained in this Ph.D. dissertation. Four different subsections group our results: the bio-inspired attention model, the low-level visual processing engine layer architecture on-chip, the system for active gaze control, and the visual attention system on-chip.

5.1 Bio-inspired attention model for motion estimation

From an engineering point of view, many properties of biological systems are worth to study in detail before addressing a design towards a skill, such as vision, that biological systems solve with outstanding efficiency. The efficient way in which they deal with real-world tasks with simple models, their adaptation against unsettled conditions, their capacity of dynamically reducing the available information to the most important cues, the use of their highly-parallel structures to optimize the processing time combined with the use of multiple and distributed sensors, or their auto-configuration and reliability make them very attractive references for current technological challenges.

The main purpose of this contribution is the proposal of a bio-inspired model of visual attention. This model consists in a top-down task-dependent modulation that dynamically biases the computation, i.e. this modulation does not help improving the result of the computation of the whole scene but in some specific areas determined by some target visual feature.

The modulation stream is driven by several scene features: color data that in a visual system comes from the retinal cones; texture cues that have been identified as one of the early steps towards object recognition in human vision (see Julesz [60] [61]), and finally with motion direction or magnitude cues (using in this case a more engineering approach). In our case, the system is used to estimate motion. A general scheme of the complete system is shown in Section 5.1.

In the literature we find diverse works that perform retinal-like processing. For example, in [62] [63] authors propose the use of AER (address-event representation) protocol for an artificial retina and describes the way of computing some two-dimensional filtering. The contributions in [64] [65] [66] consists in proposing a framework for automatically generating retina-like models, including multichannel filtering, neuromorphic encoding and electrode mapping in the framework of design of neuro-prostheses for sensory or motor disabilities. Finally, we even find some works that use retinal cues for motion estimation as in [67]. This method uses four specialized layers to compute the motion speed and direction. However, as far as we know, none of the works deals with optical flow estimation with retinal cues using an adaptive attentional modulation stream to select the most relevant areas in the scene.

Our model and implementation represent a very simple alternative for a dynamic visual system that includes visual attention. The seeds of this system were the works related with spike-based artificial retinas developed by Kwabena Boahen (for more details c.f. [68] [69] [70]). The brain does not execute commands as our programs. Instead of that, it massively activates connections or neural synapses to transmit chemical signals in thousandth of a second. All the efforts in these papers are addressed to understand a small part of this organization and functions for "morphing" it in order to create embedded devices.

In our work, we extend the mentioned efforts by modeling a vision system for optical flow estimation based on retinal cues from several kinds of neurons. In our brain, some areas are known to be connected with motion direction perception specially in the case of the MT area. This has been demonstrated thanks to the work of some neurobiologists as Bill Newsome (see [71] [72]) and the experiments about diseases as the akinotepsia or motion blindness that affect this area of the brain and provokes the incapacity of detecting motion.

This work is inspired in the function that these MT neurons seem to carry out using retinal cues. These retinal cues are basically spatial local contrast and temporal-luminance-change detectors. Our work also delves into this topic by evaluating the importance of different kinds of cues generated by the retinal neurons in order to simulate the behavior of a continuous-time system that with a small amount of responses and in tenths of a millisecond is able to perceive the motion direction and magnitude in a very efficient way.

In comparison with the system detailed in Section 5.4, this approach can be seen as a first and simple biologically-inspired strategy to the attention problem, but specifically focused on the motion computation. As it will be explained in that Section, more cues can be added to the original system in a bottom-up manner, as in the case of local orientation or motion direction and also as a top-down bias. Fig. 14 visualizes the contribution of this Section to the Ph.D. dissertation, in the framework of active vision for dynamic systems. The adaptation component is not highlighted but, as mentioned in Section 1, adaptation is present in a distributed way in every Section of our Ph.D. dissertation.

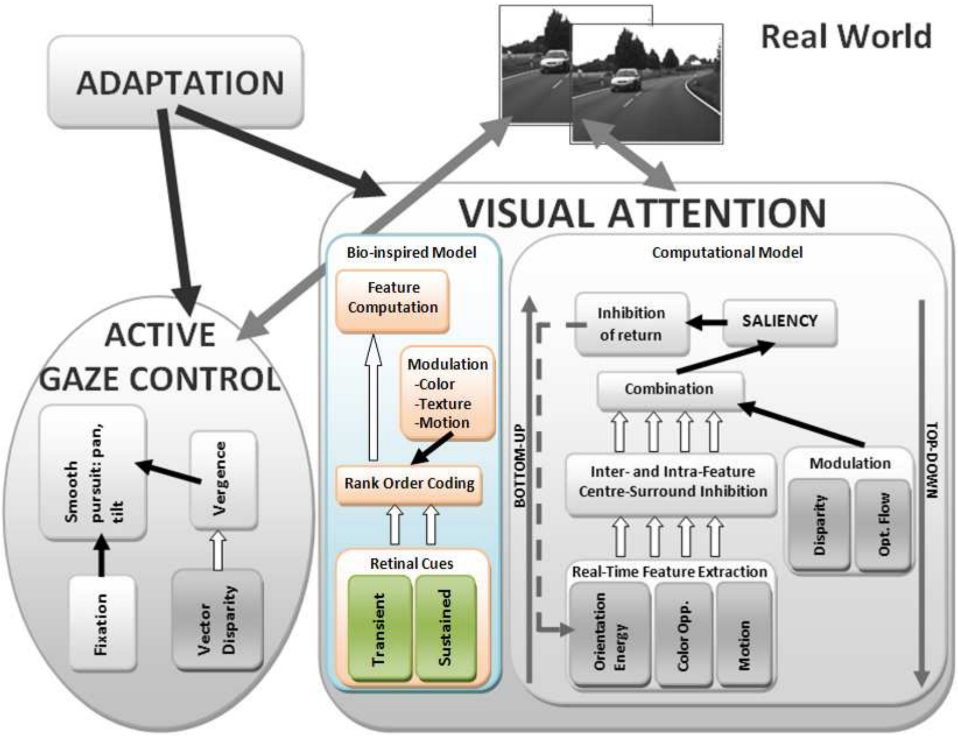


Figure 14: Contribution of this Section to active vision for dynamic system. The contribution is colored to highlight it in the general scheme.

Active visual system for motion estimation based on an artificial retina

This first approach to the implementation of visual modalities is a biologically-plausible system that estimates optical flow using as input the cues provided by an artificial retina [68] [69] [70]. This information consists in four different responses from the correspondent kinds of neurons: two encoding local luminance contrast (ON-OFF and OFF-ON, as sustained on-center-off-surround and viceversa detectors) and two encoding temporal changes (INC and DEC as transient increase or decrease detectors) [68]. In the proposal we evaluate the accuracy of the optical flow computation using a multiscale algorithm based on block-matching.

The active visual attention mechanism is implemented in a two-stage strategy. Firstly, we use a rank-order coding scheme for selecting the most relevant cues using a energy function criteria. Moreover, the work studies different integration strategies for the cues that come from the different kinds of cells in order to select the best choice for a multiresolution motion estimation. At this point, we select the strategy that provides the highest stability and accuracy at the lowest cost. Then, the second stage uses the information from various features such as color, texture or motion speed to generate a modulation stream that biases the a new selection from the remaining cues that come from the previous stage. These twice-filtered cues are selected by a modulation that is driven also taking into account properties from the targets or the task that is being performed. It is worth noticing again that this smart selection is performed with the purpose of reducing the required resources or computational complexity.

The most important aspects of this work are listed below:

- The bio-inspired event-driven model is based on an artificial retina with four different kinds of cells: two static or substained (ON-OFF and OFF-ON) and two transient (INC and DEC). The receptive fields for each kind of cells are emulated using Difference of Gaussians filters according to [68] [73] and the parameters for these filters and for the temporal windows are set in the framework of general scenes and sequences with low spatial resolution realistic movements. The amount of information provided by the artificial retina sums about 12% of the image resolution for each cell which means a drastic reduction from the resource point of view.

- Optical flow computation based on a hierarchical multiscale scheme. The computation is based on a pixelwise region-based matching algorithm [74] using the retinal responses as inputs. The similarity criterion is based on the minimum error (MAE, mean absolute error) between the selected regions, using a searching window with a fixed

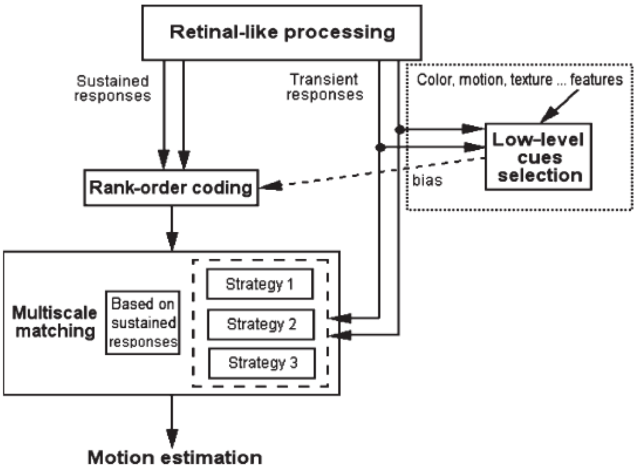


Figure 15: Processing scheme for the artificial retina visual system. The dotted square represents the multimodal attention operator. The dashed square represents the selection process of the strategy for integrating the transient information

size. This kind of algorithms presents problems with illumination, occlusions, and noise. On the other hand, their implementation is simple and equivalent to the energy-based algorithms but more efficient [73]. The basic algorithm is complemented using a multiresolution scheme in order to tune with objects of different spatial resolutions and thus, to improve the accuracy of the computation.

- Rank order coding scheme. Finally, a bio-inspired scheme is also implemented to select the most important estimations. Based on the work by Perrinet and Thorpe [75], neurons that fire the highest spikes provide the more accurate estimations or, they can be seen are the most reliable for the subsequent computation. The criterion for the selection of these neurons consists in a normalized sum of the sustained responses for each region (ON-OFF and OFF-ON responses). After this computation and the ordering of the responses, we select a fixed percentage among the highest values. These responses are the only ones that we will use for the optical flow estimation.

- Integration of transient neurons. One section of the work is devoted to the study of the role that the transient neurons play in the optical flow computation. We design four alternatives for the integration of the transient information in the motion estimation and analyze the computational load of each alternative, studying a combination of 480 different configurations for the parameter set. We also compare the cost, using as function the ratio for the Average Angular Error (AAE), the obtained density (this model is also sparse), and the stability of each alternative. We finally select the alternative that presents the lowest cost and highest stability, with an affordable computation load.

- Multimodal adaptive attentional modulation (motion, color, textures, etc.). Finally, the work presents a final proposal for an adaptive multimodal attention operator that biases the estimation and focuses it on the areas with a significant rate of cues of motion, a specific color or texture.

The selection of the most important features is very relevant for efficient computation thus, in applications such as tracking we can bias the rank order processing according to the transient neuron responses using a small amount of very confident features. All the scheme is designed to be integrated with applications that required a very limited computing resources but without losing the relevant features.

The journal article associated to this part of the dissertation is:

F. Barranco, J. Díaz, E. Ros, B. Pino. Visual system based on artificial retina for motion detection IEEE Trans. on Systems, Man and Cybernetics - Part B:Cybernetics, vol. 39, no. 3, pp. 752-762, 2009. DOI 10.1109/TSMCB.2008.2009067.

5.2 On-chip low-level processing architectures

This Section describes the design and development of the low-level vision processing engines for dynamic environment applications, that are finally integrated in a single architecture. This contribution represents the first processing stage for the architecture presented in Section 5.4 as shown in Fig. 16.

As mentioned the biological vision systems can be improved by the addition of data relative to visual modalities as, in our case, motion and depth (see Section 5.1). In this Section, we encompass from a purely engineering perspective the computation of these data based on methods that provide solutions with the required accuracy but with an affordable resource cost. All the work developed in this Section is then applied in subsequent works. Firstly, we extend this approach to the depth computation (performed in Section 5.2.1) which is also integrated with a vergence control for a robotic platform system (see Section 5.3). Secondly, the motion estimation module (detailed in Section 5.2.1) is integrated in a complete system that

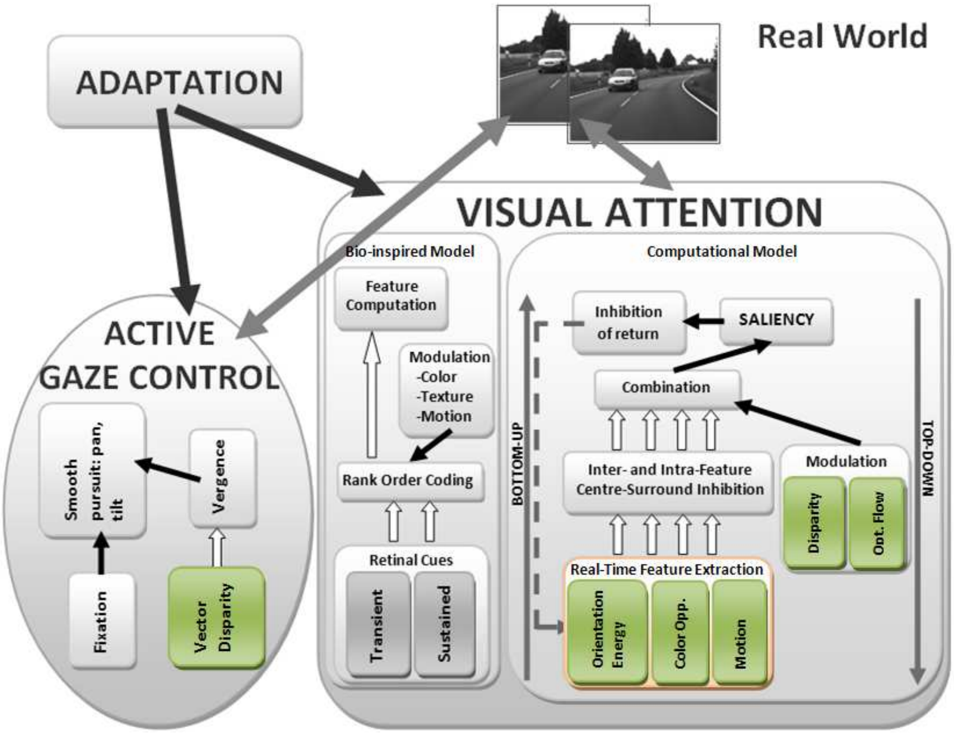


Figure 16: Contribution of this Section to active vision for dynamic system. The contribution is colored to emphasize it in the general scheme.

also provides depth, local energy and orientation (Section 5.2.1). Different engineering-based decisions for the computation of the visual attention system (Section 5.4) are made by assessing the use of color for the optical flow and disparity computation (Section 5.2.2) and the use of biologically-inspired fusion for reducing costs in the motion estimation (Section 5.2.3).

The depth and motion estimations are essential for autonomous navigation systems in a dynamic scenario. Depth is perceived thanks to a binocular vision system. It is defined in terms of disparity, as the difference between the projections of the real image on the right and left visual sensors. Similarly, motion is perceived as a temporal difference of the projection of the real image between consecutive instants. In this way, algorithms for computing disparity and optical flow are very similar, consisting in searches for spatial or temporal correspondences matching respectively. In the case of local orientation and energy, they encode local geometrical information and local contrast present in the scene.

A lot of applications are performed by using these visual primitives: autonomous navigation [76], obstacle avoidance in vehicles [77] [78], advanced driver assistance systems [79], video-surveillance [80], robust tracking [81], object recognition [82], structure from motion [83], etc.

There are plenty of algorithms for motion and depth estimation. In particular, we focus on gradient-based methods for image registration that are classified also as local or global methods. Local methods were developed firstly by Lucas&Kanade [84] [85] [86], assuming that motion or disparity are constant in a local neighborhood. On the other hand, global methods assume the smoothness of the motion or the depth and therefore, the estimate for a pixel depends on the rest of the pixel estimates. These methods were firstly developed by Horn & Schunk [87]. Although a lot of algorithms generate very accurate and dense results, we select a gradient-based method due to its good trade-off between accuracy and resource utilization [79] [88].

For our implementations, we have mainly selected the Barron's proposals [74] [89]based on local methods. In addition to this, we also include an extension to the original work, the multi-scale architecture [90] based on warping operations. This extension allows our algorithm to estimate better for high-range displacements. The details about this architecture are included in Sections 5.2.1 and 5.2.1. It briefly consists in constructing a Laplacian pyramid with the input images (see Adelson's work [91]). Then, it performs a search over a small number of pixels at the coarsest scale resolution. This estimation is used as a seed to search locally at the next finer scale, and so on to the finest one.

In general, the computational effort for estimating these multiscale-with-warping disparity and optical flow schemes is high and even more requiring real-time performances. We implement our hardware architecture in an

FPGA motivated by the potential of the algorithms for exploiting the maximum level of parallelism. About the specific platform for fulfilling our requirements, specially the real-time performing, we used a XircaV4 [92] with a Xilinx Virtex4 XC4vfx100 FPGA. The board provides a PCIe interface for the communication with the computer and four SRAM ZBT memory banks with 8 MB. The PCIe connection enables working with the board as a co-processing platform but, it also may work as a stand-alone platform.

By using fine-pipeline modular design for the implementation, we obtain good performances in terms of maximum frequency, at low power consumption [93], and also a versatile and adaptable design that may be customized for the implementation of multiple new visual modalities. This scheme also provides an easy way of implementing the necessary sharing strategies for our final on-chip system.

Section 5.2.1 describes the process of estimating optical flow, disparity, energy and orientation. The complexity of integrating the extraction of all these visual features in a single chip explains the lack of works involving such volume of visual processing in the literature. Actually, the integration does not only consist in integrating together the different cores as individual entities but, combining on the same chip multiple processing datapaths with limited resources. Several issues such as routing a large amount of logic blocks, memory access, limited number of memory banks and ports, or resource sharing strategies have to be solved when designing this complex architecture. In previous works such as [40] [94] in which we have collaborated, we cope with a similar integration problem but in this case, using phase-based algorithms (being the only ones similar in the literature, as far as we know). They are more accurate and robust against illumination changes but also require more resources.

Despite that a more detailed state-of-the-art and algorithm descriptions are described in the works gathered in this Section, we present here a brief summary of the most relevant contributions in the literature. For the optical flow estimation, we cite in our works various proposals using different technologies: FPGAs, CPUs, GPUs or even, CELL processors. For example, the CPU version of [47] achieves an effective frequency of 90 MHz at a 1280x1026 resolution, with a mono-scale version of the Lucas&Kanade algorithm. In the case of GPUs, [48] obtains a performance of 48.5 fps (frames per second) with a phase-based method. Finally, [95] and [96] with Multichannel-Gradient- and tensor- based approaches also achieve real-time performances in works that are implemented for FPGAs. In the case of the disparity estimation there are also plenty of proposals for diverse technologies and methods. In the case of [97] and [98] [99], the first one is GPU-based and reaches 4.2 fps at VGA resolution (640x480) with a Semi-Global Matching approach that uses mutual information, the remaining works are

FPGA-based solutions that uses region-based matching methods, the more suitable methods to be parallelized which means really high performances (till 550 fps for 800x600 resolution). Their main drawbacks are their low accuracy and robustness ([100] [101]).

As a brief summary of the structure of this Section, we describe in Section 5.2.1 the architecture that integrates the computation of all the low-level visual features. Then, we also describe in Section 5.2.2 the exploration of different alternatives to give more accuracy and density to the estimations of the optical flow and the disparity using color cues. Finally, Section 5.2.3 details a proposal to reduce the amount of resources that the optical flow architecture (the most expensive in terms of resource utilization) requires.

5.2.1 Low-level visual processing engine architecture

As commented, this architecture conforms the first stage of low-level extraction features for the final visual attention architecture of Section 5.4 (see Fig. 16). The work proposes an architecture for computing in real time optical flow, disparity, and local orientation and energy on a single chip. This architecture is proposed as a low-cost version against previous similar architectures [41] [40]. The cost reduction in a first stage is performed by selecting the appropriate algorithm. In our case, this algorithm is gradient-based against the phase-based algorithms of previous implementations [43] [41], that are more expensive in terms of resource costs. These gradient-based methods are very popular in the literature because they provide a very competitive accuracy vs. efficiency trade-off [79] [88].

Hierarchical gradient-based optical flow estimation

The implementation of this architecture has been totally modular and incremental. Hence, the first step was the developing of an architecture for the estimation of multiscale-with-warping optical flow. This FPGA-based architecture implements the Lucas & Kanade [84] [74] gradient-based algorithm in a multiscale scheme [90]. The final architecture reaches 270 fps (frames per second) with a VGA resolution (640x480) for the monoscale version and 32 fps for the same resolution using the multiscale approach. With these results, we fulfill our real-time performance requirements.

The proposal presents a gradient-based architecture that achieves a good trade-off between accuracy and efficiency. The implementation of the monoscale implementation requires among 10% and 15% of the total resources of the FPGA platform. In the case of the multiscale implementation the resources are about 60%. As analyzed in the work, this increase of resources in 45-50% allows the improvement of the accuracy in a factor of 3x.

The design of the architecture is based on fine-grain pipelined datapaths that allow us to achieve high-performance systems and low-power consumption [79]. This approach permits easily adaptation to diverse trade-offs among performances and resource costs and the definition of hardware strategies for sharing resources. This last property provides the opportunity of designing complex systems for integrating different visual modalities on the same chip.

The final frequency of the multiscale system is 44 MHz, constrained by a crucial module in the multiscale implementation, the warping module. This module allows the system to interpolate the inputs with the estimates computed for the previous spatial scale. It requires about 23% of the global resources. In Section 5.2.3 we study how to avoid this large resource consumption by using a fusion of the estimates for the different spatial scales. Another highly relevant module is the median filtering. This module is designed by connecting in cascade two 3-by-3 median filters. This module controls the number of reliable estimates by tuning a threshold parameter. Finally, the study shows the analysis of four different alternatives for the implementation of the motion estimator, two based on fixed-point arithmetic operations and two on floating-point arithmetic, and selects the alternative with the best trade-off between accuracy vs. resource cost.

The journal article associated to this part of the dissertation is:

F. Barranco, M. Tomasi, J. Díaz, M. Vanegas, E. Ros, Parallel architecture for hierarchical optical flow estimation based on FPGA. IEEE Trans. on VLSI, vol. 20, no. 6, pp. 1058-1067, 2012. DOI 10.1109/TVLSI.2011.2145423.

Low-cost architecture for extracting optical flow, disparity, and local energy and orientation

The second step consists in the integration in the same architecture described in 5.2.1, the multi-scale computation of the intensity-based Lucas&Kanade method of disparity and optical flow, and a mono-scale implementation of local energy and orientation (see Fig. 17). The design with superscalar fine-grain pipelines allows a maximum exploitation of the potential parallelism of the FPGA platform, achieving 350 and 270 fps for VGA resolution for disparity and optical flow, or 32 fps for the final on-chip architecture.

As in the previous cases, all the specific cores and the final architecture are benchmarked with the sequences available at the Middlebury database [56]. After the accuracy, stability and density study the final architecture design is also tested. Since this work is presented as a low-cost version for computing optical flow and disparity, we also have compared the resource cost of our architecture with a similar version developed in the same board, and computed for a phase-based algorithms (see [94]). The design uses about 76% of the total available resources against 99% that utilizes the mentioned phase-based approach.

We have also developed a hardware-software platform for the debugging and visualization of the final results. In the case of this platform, the board plays the role of a co-processing board connected to a standard computer. The input images are acquired by a camera connected to the computer and are transferred to the board via the PCIe interface.

The software platform, OpenRT-Vision, an open source platform designed also in the framework of this PhD, was released in 2009 with a GNU LGPL license. The source code, the documentation, the user's manual, the tutorial, and a video-tutorial are available at [58].

Simplicity and efficiency are crucial properties of a user-friendly software platform, especially in a tool for the development of complex hardware systems. This platform constitutes a GUI that is an interface between the acquisition of the inputs and a co-processing board. This tool allows us to execute algorithms for computing different visual modalities, the connection with different boards only by developing the appropriate drivers, debugging the algorithms and comparing with the rest of vision algorithms implemented in any hardware or software platform. More details about this platform can be found in the Appendix A.

The journal article associated to this part of the dissertation is:

F. Barranco, M. Tomasi, J. Díaz, M. Vanegas, E. Ros, Pipelined Architecture for Real-Time Low-Cost Extraction of Visual Primitives based on FPGAs. **Submitted to Journal of Digital Signal Processing**.

5.2.2 Color-based architecture for motion and depth estimation

As commented previously, the architecture presented in Section 5.2.1 conforms the basic stage of the final visual attention architecture for active vision presented in Section 5.4. As mentioned in Section 1 attention uses color in their deployment. Moreover, color is helpful in visual processing because it adds more information to the estimations representing a hypothetical increase in the reliability of the estimations. This may finally mean an increase in density or accuracy of the estimations. On the other hand, this also means an increase in the required resources. In this Section, we

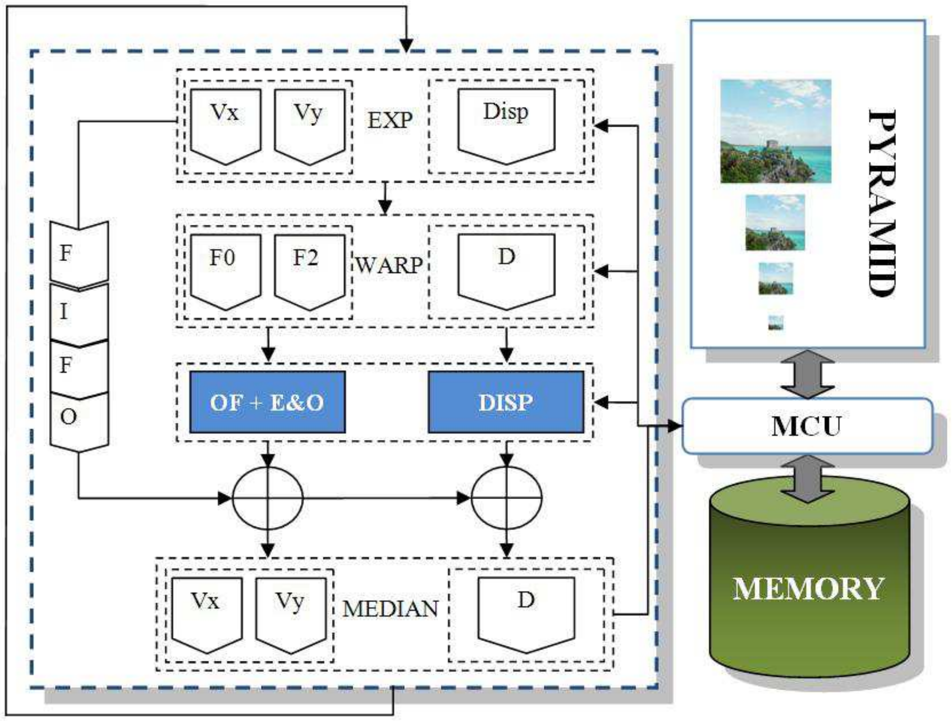


Figure 17: Design of the hardware architecture for the on-chip computation of optical flow, disparity, local energy and orientation. On the right side, we have the pyramid iteration and its communication with memory. On the left side, it can be seen the multi-scale optical flow and local contrast descriptors computation (sampling, warping, merging, median filtering and the L&K optical flow computation) and also the disparity datapath. The communication with memory is performed through the MCU (c.f. [102]).

assess the relevance of color cues in terms of accuracy and density, and also its resource cost impact for the optical flow and disparity computations.

In this case, we have developed two architectures based on color cues for motion and depth estimations similar to the gray-based architecture of Section 5.2.1. Both designs are based on the Lucas&Kanade algorithm and completed with the multiscale extension, which allows a more accurate estimation of large displacements. The final systems achieve a performance of 32 and 36 fps for VGA resolution in the case of the multiscale version. One of the most interesting points of this part of the Thesis dissertation is the study of the different color spaces, analyzing the accuracy and the resource cost trade-off.

We also study a simplification based on Golland's works [103]. She proposes a reduction of the color channels from three to two, without a significant loss in accuracy and density but, in a hardware implementation, this reduction might also lead to a significant reduction in resources. Such a resource cost reduction may represent a positive point for the hardware implementation of the color-based model. Additionally, the reduction is only possible if the color channels may be determined as a ratio relation of the others, which means that they are dependent.

The whole architecture described in Section 5.2.1 is adapted for the color-based implementation of optical flow and disparity. General changes consist in handling more input information, a more complex warping process and a general multiplication of the structures by the number of color channels of the selected representation.

A detailed benchmarking is performed for the color implementations:

- Firstly, for the software color implementations. This stage carries out a complete benchmarking, comparing the different color spaces for both, motion and depth estimation.
- A second step consists in reducing the number of color channels, from three to two. This stage analyzes the best selection of channels for each color space performing the same benchmarking as in the previous case.
- Next, we check that the reduction does not affect substantially the accuracy and density results.
- Finally, the hardware implementations are also tested using this color space. It is worth noticing that the hardware architecture is designed for working with any color space with two color channels.

The main conclusion derived from this detailed study is that the color-based architectures are generally more accurate and provide more density than the intesity-based ones. On the other hand, duplicating the information and the structures required for their computation (we work with color spaces of two color channels) means a final increase in resources about only 1.2-1.3x with respect to the intensity-based architectures.

The final visual attention architecture in Section 5.4 implements motion and depth estimation modules based on intensity-based information, due to the resource limitations. However, the color study allows us to take into account its potential for accuracy improvement and density increase in future applications with strong requirements relate with these properties.

The journal article associated to this part of the dissertation is:

F. Barranco, M. Tomasi, J. Díaz, E. Ros, Hierarchical architecture for motion and depth estimations based on color cues. **Submitted to J. of Real-Time Image Processing**.

5.2.3 Multi-resolution approach for parallel optical flow computation

This contribution presents a hardware-friendly motion estimator suitable for real-time applications in robotics or autonomous navigation, where the precision requirements are not very strong and the possibility to be embedded imposes constraints to the computational complexity or the resource requirements of the model.

This approach is based on the same local gradient-based method described in Section 5.2.1, that presents a good trade-off between its accuracy and its computational cost [79]. The main drawback is the unreliability of its estimates for large-displacements. This problem is overcome by implementing a sequential multiscale-with-warping scheme that allows the model to tune the movement of objects at different spatial resolutions. But now, the main problem of this multiscale-with-warping approach is its resource cost, especially the warping module is the bottle-neck for the potential implemented architecture. This multiscale-with-warping approach for optical flow estimation is the scheme that we perform in the architecture of Section 5.2.1. Both architectures are compared in Fig. 18 and also in the work related to this Section [104].

Our contribution consists in implementing a novel parallel approach for the multiresolution scheme, avoiding the warping module (the general scheme is represented in Fig. 18). The computation is performed by integrating the different scale-by-scale motion estimations with different strategies, with the aim of reducing the computational cost and therefore, the potential hardware resource cost in a GPU, FPGA or ASIC future implementation.

We have developed a strategy for the fusion function of the different spatial resolution estimations. Our purpose is finding the combination of spatial resolution estimations at which the motion is accurately tuned, and consequently the number of spatial scales is crucial. We also carry out a strategy for selecting the more reliable estimations based on the use of non-ill-conditioned structure information. This also motivates the use of structure tensors to weight the areas with more spatio-temporal structure. Our fusion function consists in averaging the estimates weighting them with a spatio-temporal structure tensor that is computed using hardware-friendly operations. The implementation is parameterized in order to weight the spatial or the temporal part making our proposal adaptable. The combination penalizes large movements (they are prone to error) but gives more weight to estimates that are computed for regions with stronger spatio-temporal structure (possibly more reliable).

We have compared different performance and measures for the multiresolution approach against the multiscale-with-warping. In the work, also illustrate the accuracy of our proposal by showing a complete benchmark of the results for different sequences available at the well-known Middlebury database [56]. In average, our computation increases the error about 1.7° and loses 11%, but achieving 30% of resource and 40%-50% of latency reduction in an FPGA implementation, and a speedup of CPU time of 3.5x in a PC architecture with 2 cores.

Resource and latency reductions are important for applications such as object tracking, video-surveillance, robot autonomous navigation, etc. These target applications need parallel and computationally efficient methods. Furthermore, a low latency is strongly required for application with real-time interaction as robot manipulation or collision avoidance. Thus, the parallel nature of this architecture is essential for fulfilling these requirements conversely with the sequential nature of the multiscale-with-warping approach.

The journal article associated to this part of the dissertation is:

F. Barranco, J. Díaz, B. Pino, E. Ros, A multi-resolution approach for massively-parallel hardware-friendly optical flow estimation. **Submitted to Journal of Visual Communication and Image Representation**.

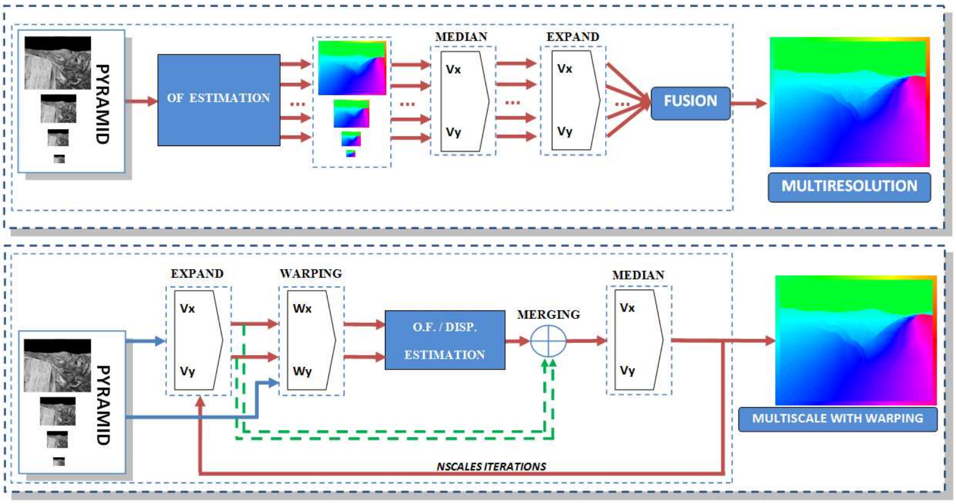


Figure 18: Multiresolution (top) and multiscale-with-warping (bottom) approaches for optical flow estimation. It is worth noticing the difference between the parallel multiresolution that avoids the warping but needs and extra combination stage and the sequential multiscale-with-warping scheme

5.3 Active gaze control system

As explained in the previous Section, we have developed a low-cost real-time architecture that allows us to compute optical flow, disparity, local energy, and local orientation. Since all these visual primitives are provided in real time, we are capable of developing a system to interact with our changing real-world: an active vision system. In this Section we presents our system that actively controls the gaze of a binocular system, integrated in the iCub robotic head [105].

An active vision system is the one that is able to interact with its environment by changing its scene acquisition process more than just passively observing it. It is based on the premise that an observer may understand efficiently its visual environment if the sensors interact with it, for instance moving around it, selecting the visual information or analyzing the visual data to address specific tasks posed by the observer. In an active vision system, the border between scene acquisition and processing vanishes (both stages merge into an active scheme). Different fields and applications are involved in active vision systems as the geometrical modeling and optical flow computation, control theory, filtering, or dynamic modeling for applications such as tracking, control of vision heads, geometric and task planning, etc [106] [107].

Vision is essential in robotics for developing active systems. Complex robots involve sophisticated approaches with consistent world models and high-level interpretation for planning, prediction and object tracking. The interaction with objects, animals or even humans is quite important for these robotic platforms and in such interaction, vision plays the central role by collecting the data or directing attention or gazing on a specific object or human. In these particular cases, depth is crucial thus, it allows the interaction in the real 3D space and represents a substantial cue in recognizing the static objects or obstacles. Additionally, depth information is also essential for controlling binocular vision systems helping to focus on the depth plane of an object of interest ("fixation"). As an example, we suggest an object continuously moving to variable velocities in 3D space. We require a system capable of tracking the object ("smooth pursuit"), and therefore a system able to control the eye movements of our binocular robotic head: vergence (rotation of each eye with respect to the vertical axis to change the disparity), tilt (rotation of the eyes with respect to the horizontal axis), and version (rotation of the eyes with respect to the vertical axis).

Active vision systems are currently integrated in a lot of applications. For example, robots need the capacity of tracking objects in their environment [108] [109] [110], in the case of autonomous guided vehicles they need tracking road markings or other vehicles [111] [112]; in robot arm applications to capture multiple views from a moving camera of a target object [113] or select the optical trajectory for grasping it [114]; lip, hand and gesture tracking and recognition [115] [116]; people tracking for surveillance applications [117]; mapping and 3D reconstruction [118] [119].

This part of the thesis dissertation was developed during a research stay at the Universitá di Genova, Genoa, Italy under the supervision of Prof. S.P. Sabatini. The work was part of the European project iCub [105] (IST-FP6-004370), with the participation of some institutions from Japan and USA, led by the University of Genoa and the Italian Institute of Technology. The goals of this reference project were the creation of a new advanced humanoid robot for embodied cognition (mainly for researching) and to advance in the understanding of cognition by exploiting the robotic platform capabilities. In this case, iCub developed their capabilities progressively as a child, learning about its body, interacting with the real world and even communicating with other fellows or humans. All the developed technology in the project was released with a GPL license for open distribution.

Our work was focused on the use of the robotic head (displayed in Figure 19). The platform is composed by a stereo pair of FireWire cameras, two DSPs (hybrid Motorola 56F807 16-bit), and an inertial sensor connected to a PC (via CAN bus). The head consists of a binocular system with six degrees of freedom: the eyes have two rotational degrees of freedom but with a common tilt; the neck also has three degrees of freedom. The control

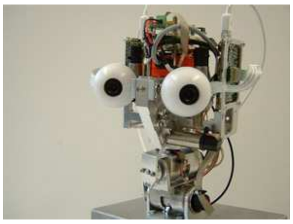


Figure 19: ICub Head robotic platform. The platform consists of two FireWire cameras, two DSPs for neck and eyes separate control, an inertial sensor connected to the PC via CAN bus.

of cameras and neck motors is performed separately by the two DSP units [120] [121].

For implementing an active gaze control for a dynamic environment with the robotic head, we require the continuous change of the viewpoint tracking for example, an object moving in the 3D space. The accomplishment of this task is performed by implementing a control of the tilt and pan angles of the platform. In that way, we may explore the maximum space exploiting the resources and, the vergence angle, that allows us to gaze on different depth planes. Finally, for the vergence control we need a system that computes the depth of the different objects in the scene, in our case, implementing a new hardware core that computes vector disparity. This work is based on the core generated for the computation of horizontal disparity (see Section 5.2.1). For its final multiscale architecture, as the warping operation is now a bidimensional operation (for the vector disparity), we use a similar architecture to the one computed for the the optical flow (see Section 5.2.1).

The contribution of this Section to the general scheme is shown in Fig. 20. Again, adaptation is not highlighted but, as mentioned in Section 1, it is present in the whole work and therefore, also in this Section.

Vergence control for robotic platform based on vector disparity

This work presents an active gaze control system that initially deals with the vergence, tilt and version control problems. As mentioned in Section 1.1.1 gaze control includes six different control mechanisms. In our case, we do not manage the head movement control thus, our system does not take into account the vestibulo-ocular and optokinetic reflexes. However, our work does describe the vergence control, and emulates the “smooth pursuit” and “fixation” mechanisms.

The computation of the depth is usually based on matching image features between two images in a binocular system. Moreover, this matching problem is simplified by using some constraints of the geometry of the stereo

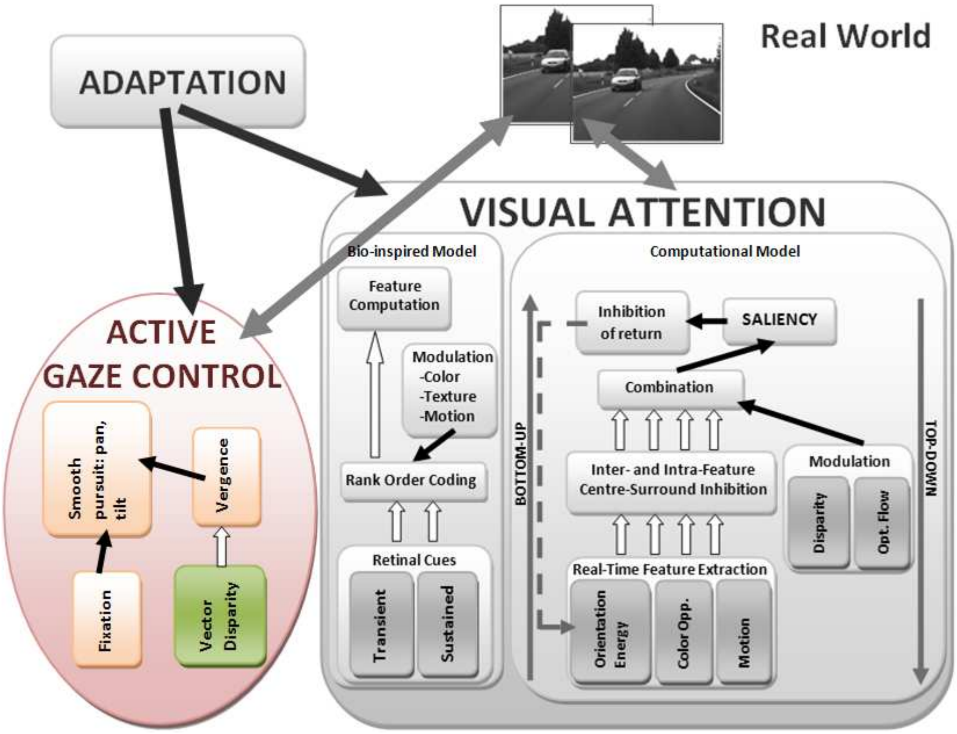


Figure 20: Contribution of this Section to active vision for dynamic system. The contribution is colored to highlight it in the general scheme.

rig: assuming a fixed and known geometry of the cameras, the problem is simplified to a one-dimensional problem (using a previous calibration stage that allows us to know the intrinsic and extrinsic camera parameters). For achieving this objective, the rectification process corrects the lens radial distortions and aligns the image planes with the epipolar lines, which simplifies the disparity estimation, reducing the problem to a one-dimensional matching, only for the horizontal coordinates. The main drawback of this scheme is that the rectification process is required each time that the camera position is modified. Hence, this approach is not suitable for a robotic platform (in our case is the iCub platform [105]) that is able to modify the relative position between the cameras by changing on real-time vergence, tilt or version.

In this Section we describe the design of a multiscale implementation for the vector disparity computation (see a detailed description in [122]). The use of a static vergence cannot allow a correct gazing to a specific location. The solution is the use of a dynamic vergence control of the binocular system to fixate the point. In this case, the most accurate disparity estimates are obtained in the area around the fixation point, thus the image plane is moved to the depth of the fixation point. All these properties provide the possibility of exploring the scene in detail, gazing at different fixation points or areas of interest. The motion control strategy of the robotic platform is based on a separate control of version and vergence inspired in Hering's Law [123] in the same way that is performed in [28].

The defined hardware architecture achieves real-time requirements for VGA resolution with the multiscale scheme (32 fps). In this case, we study two different algorithms for the vector disparity implementation: a gradient-based and a phase-based one. We finally select the gradient-based algorithm for the final implementation due to its stability and its trade-off between accuracy and resource cost (though phase-based algorithms are usually more robust against illumination variations [43]). Additionally, the FPGA-hardware implementation is here justified for its suitability for the integration with active vision systems, and also for the potential exploitation of the parallel resources to obtain high performance.

Finally, the contribution also presents an algorithm for the object fixation and the object smooth pursuit based on color. This algorithm computes a centroid based on the color information of the selected object. Then, it computes the distance to the eye centers and sends a command to the PID controllers to activate the motors towards modifying the version and tilt according to the computed distance. Finally, it acts on the vergence motor computing the vector disparity and moving them until it reaches the zero disparity value. By modifying the vergence of the cameras we translate the image plane to the depth of the object and there, the disparity computation is more accurate. In this point, we act on the vergence to make the disparity zero.

Despite the problems with the physical limitations and the backlash of the camera pan movements in our platform [105], tracking applications provide the possibility of performing a smooth pursuit of the object gazing on different fixation points. In the proposal we present two specific cases: the fixation in a target object and the tracking.

The journal article associated to this part of the dissertation is:

F. Barranco, J. Díaz, A. Gibaldi, S.P. Sabatini, E. Ros, Vector disparity sensor with vergence control for active vision systems. Sensors, vol. 12, no. 2, pp. 1771-1799, 2012, DOI 10.3390/s120201771.

5.4 Visual attention system

Visual perception processes does not consists in image acquisition and offline processing, they also include active selection mechanisms, this is why we call them active vision processes. As mentioned, between these mechanisms we find the active selection of relevant information. As mentioned in Section 1, real-time processing is required for dynamic scenarios, to compute the most dynamic features in an efficient way.

Visual attention explains the huge difference among the amount of information that our visual system receives and the small portion that our brain processes. Authors recognize the multiplicity and diversity of definitions about attention and that we only know marginal details about how it works and what deserves our attention. As Groos wrote in 1896 [124]:" *'What is Attention?' There is not only no generally recognized answer, but the different attempts at a solution even diverge in the most disturbing manners*". One of the most reproduced definitions of attention comes from William James in Principles of Psychology (1890) [125]. He wrote: "*Everyone knows what attention is. It is the taking possession by the mind, in clear and vivid form, of one out of what seem several simultaneously possible objects or trains of thought. Focalization, concentration, of consciousness are of its essence. It implies withdrawal from some things in order to deal effectively with others*".

Attention neurophysiology is still currently an open debate. The brain regions that participate in the deployment of visual attention include most of the early visual processing areas. The visual information enters the primary visual cortex via the lateral geniculate nucleus and then it is processed along two parallel pathways: the dorsal stream that involves the spatial localization (*where pathway*) and directing the attention and gaze towards the more

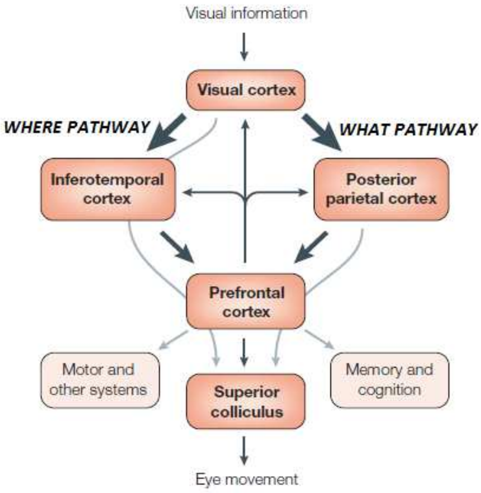


Figure 21: Neural mechanism of the control of visual attention (c.f. [35]). Dorsal and ventral stream are represented in the figure, and the brain regions that participate in the attention deployment.

interesting objects in the scene (right stream in the Fig. 21); the ventral stream (*what pathway*) that involves the identification or recognition of the objects (left stream in the Fig. 21). The dorsal stream controls where to attend next, although object recognition in the ventral stream can bias the next attentional shift through top-down control.

One of the most impressive facts of attention is that with a restricted computational capacity, it accomplishes near real-time performance, smartly reducing the massive amount of data that comes from the visual system (the optic nerve is estimated that generates about 10^7 - 10^8 b/s a bandwidth that exceeds by far what our brain is able to process [35]).

A common error consists in interpreting visual attention and gazing (see Section 1.1.1) as the same process when actually, attention processes the information in a target area that has not to be necessarily the same than the gaze center area. Moreover, attention mechanism provides the selection of particular aspects of a scene driven by a task. The attention mechanism could be justified because of the information reduction but beyond that, the memory, the derived knowledge and the use of this information in the appropriate moment are also part of the attention process [3].

As mentioned in Section 1.1.1, in visual attention two different mechanisms are acting simultaneously to selectively direct attention to objects in the scene [19] [20] [21] [22] [23] [24]:

- A very fast bottom-up task-independent process for some stimuli that are intrinsically salient in a specific context.

- A voluntary top-down task-driven process that is more powerful than the first one but slower.

After that, a competition process between the salient locations is carried out, to select the next area to be attended. In a lot of works in the literature, a scene topographical representation of the relevance of the stimulus, the saliency map, guides where the next most salient location is. Hence, a winner-take-all network implements a distributed maximum detector for that. However, how can we prevent that attention focuses permanently onto the most active location in the saliency map without shifting? The most efficient strategy consists in transiently inhibiting the stimulus at the current visited location. In human psycho-physics this phenomenon received the name of "inhibition of return (IOR)". Computationally, IOR memorizes all the attended locations and allows the selection mechanism to focus on new ones. However, the biological IOR goes beyond that and it is shown to be object-bound, that is, it should track and follow moving objects and compensate for a moving observer as well [19].

To summarize, these two modes of attention operate at the same time and hence, stimuli in the scene have two ways of being attended: willfully or consciously brought into the focus of attention or winning the competition for being the most salient.

Attention may be also discussed in terms of covert and overt attention. The overt attention involves the eye movements while the covert one does not. A lot of saliency studies focus on the track of the eye movement patterns more than discussing the relation with the covert attention, since we are interested in the existent relationship between the eye movements (saccades) and the properties of the visual stimulus itself [19] [126] [16]. Additionally, mostly of the proposed models concern only with the shifts of the focus of attention in the absence of eye movements (covert attention). However, we move our eyes 3-5 times per second to align locations of interest with our foveas. Despite this part of attention is not addressed in our model, the addition of eye movements entails a lot of interesting computational challenges. Of particular interest is the need for compensatory mechanisms to shift the saliency map as eye movements occur. Dominey and Arbib proposed a biologically plausible computational architecture that could perform such dynamic remapping in posterior parietal cortex [19].

As mentioned, this Section presents an alternative to the much simpler and bio-inspired system described in Section 5.1. This alternative is presented as a generic solution that combines the two explained forms of attention. Additionally, this system is generated using the layer of low-level

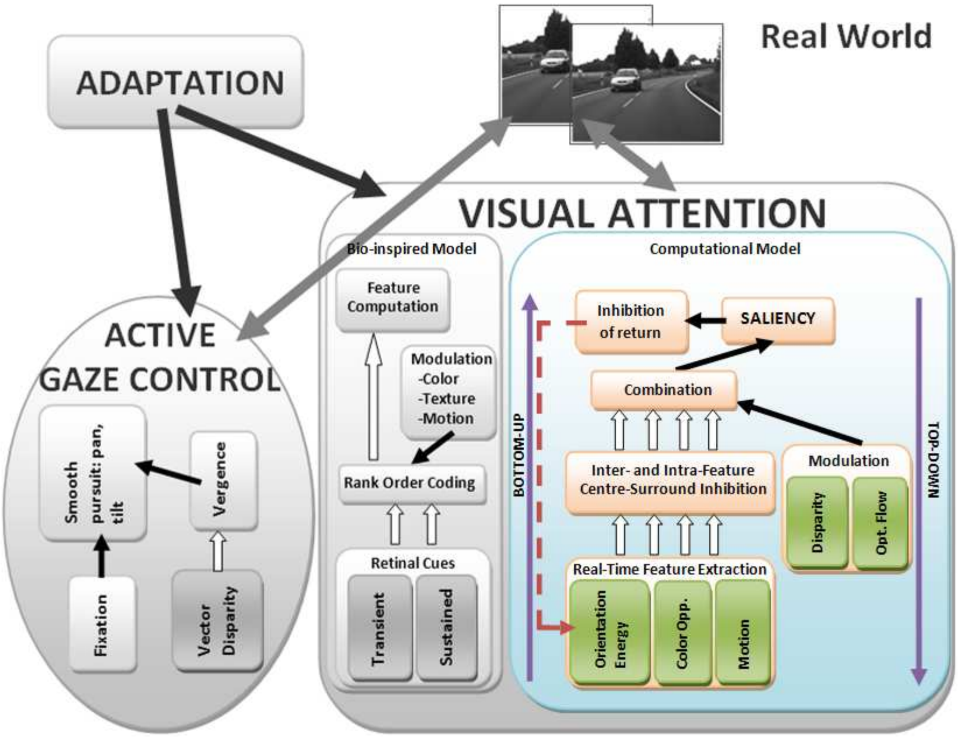


Figure 22: Contribution of this Section to active vision for dynamic system. The contribution is colored to highlight it in the general scheme.

vision processing engines developed and described in the Section 5.2. In our case, this alternative is applied to the specific case of driving scenarios.

Fig. 22 highlights the contribution of this Section to the Ph.D. dissertation, in the framework of active vision for dynamic systems. Again adaptation is required as mentioned in previous Sections.

Bottom-up saliency

Saliency, as the Merriam-Webster dictionary [127] defines it is “standing out conspicuously, prominent, of notable significance”. The saliency concept is focused on what is selected without worrying about how. Salient visual content is conspicuous, attractive or noticeable with respect to its surrounding causing an automatic deployment of the attention. There is an inherent significance of this content with respect to the other contents of the scenario therefore, saliency is explained as a relationship between the significant content and its neighborhood. Hence, it draws the attention automatically and independently of the task that it is being performed.

From the neurological point of view, there are clinical evidences that support the existence of a distributed saliency map. And it is interesting for the computational model to know the place where this map resides in order to understand how it works and the way in which we may mimic it or combine all these information into a centralized one. Even today, the discussion of the location of this distributed saliency map remains controversial: Koch and Ullman (1985) [128] proposed the lateral geniculate nucleus (thalamus), Robinson and Petersen (1992) [129] the pulvinar nucleus (also in the thalamus), Kustov and Robinson (1996) [130] proposed the superior colliculus (more involved in the control of attention) and other authors proposed as well the V1 (Li 2002 [131]), V4 (Mazer and Gallant 2003 [132]) and posterior parietal cortex (Gottlieb 2007 [133]).

Hence, the idea of a centralized saliency map seems controversial taking into account the existence of different brain areas involved in the visual saliency processing. A possible explanation could be that some of the neurons in these areas are specialized in the saliency computation but located at different stages along the sensorimotor processing stream [19]. Most of the latest works are limited to find new features to add to the model but providing poor advances in explaining the way in which the brain is structured or why the process is deployed.

Top-down bias

The top-down modulation or bias of visual activity consists in the response enhancement, the information filtering and also the increase of the response sensitivity, but performed over a specific area or feature [134]. A specific task affects visual perception through top-down modulation. A task is usually related with specific objects, and finding the features that match with this kind of objects can be seen as the inverse of object detection. While the object detection consists in mapping from a set of properties to a symbolic representation, finding these properties for top-down modulation is the inverse of this mapping.

The computational challenge is the integration of bottom-up saliency and top-down modulation, such as to provide coherent control signals for the focus of attention, and in the relationship between attention orientating and scene or object recognition. One of the earliest models that combines object recognition and attention is MORSEL [135], in which attention selection was shown to be necessary for object recognition. These works will be explained in Section **??**.

The effects of attention may be seen in all areas of the visual cortex but, there is evidence that top-down bias signals are generated outside the visual cortex and transmitted there via feedback connections [136]. Current

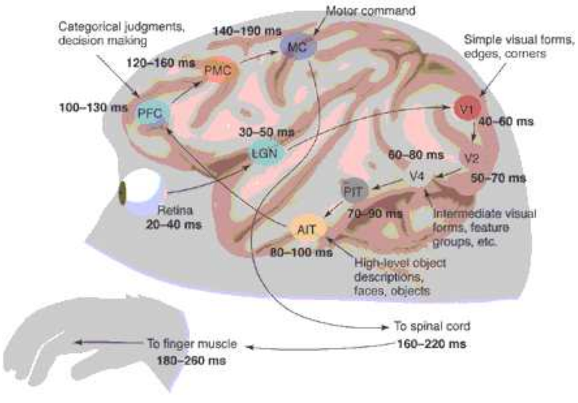


Figure 23: Brain areas in a primate. The main areas related with visual processing are highlighted with an example of the task that is performed there, according with the psycho-physiological evidences (c.f. [151]).

evidences seem to favor selection achieved by a way of competitive interaction in the visual cortex with bias signals from within parietal and frontal cortices.

There exist neurobiological evidences of different kinds of response enhancement. Top-down modulation seems to affect almost every area in the primates visual system (see Fig. 23) as the LGN [137], V1 [138], V2 [138] [139], V4 [139] [140] [141], MT [142] [143], and the LIP [144] [145]. Moreover, it has been demonstrated that this modulation may act to enhance specific scene features as the local contrast and color [146], the local orientation of the edges [141] [147] and the direction of the motion [148]. Another type of modulation refers to an increase of the baseline activity in the absence of visual stimulus. This kind of modulation has been observed in V1 [149], V2 [139], V4 [139] and LIP [145] areas increasing firing rates baseline around 30-40% [136]. Finally, a third kind of modulation happens in scenarios with multiple competitive stimulus, where the interaction between cells in the mentioned areas may also present suppressive interaction [150].

Itti and Koch's model

Our contribution is mainly based in the works developed by Itti and Koch [6] [19] [35], etc. based on Ullman's previous works [128]. These works only model the bottom-up saliency stream thus they are based on the image properties and are task-independent. Related to this bottom-up attention models, many authors indicate that they are not interested in the identification or recognition of the salient object ("what" pathway), but only in its localization ("where" pathway) [126]. In addition to this, Itti has also developed subsequent models for the top-down bias in works as [152].

Authors point out two main principles: the visual information is represented as topographical maps in the early visual processing areas (centralized in this case). Additionally, a center-surround computation for each image feature is performed in order to create these topographical representations, within the same feature and along different spatial scales. Secondly, the information related to a feature is combined in a single "conspicuity map" and then, combined again using local competition with a winner-take-all network in a unique "saliency map". The most salient or prominent location i.e. the location that has to be attended first, is the maximum of this map. Next, the inhibition of return module allows to visit the next most salient location and so on, selecting the next areas in order of decreasing saliency. The complete scheme illustrated with the execution of a driving scenario is shown in Fig. 24.

The first stage of the model extracts the feature from the scene. The feature extraction is carried out in a multi-scale fashion, capturing in this way image contents that tune different spatial resolutions for different feature maps. The implementation of the hierarchical scheme for the generation of the primitives at different spatial scales is based on Burt and Adelson's work [91].

The basic model includes the local energy, four different orientation maps and two channels for color opponencies (red-green and blue-yellow). The number of features is based on the knowledge about the primates and their early visual processing. Wolfe [153] presented a list of all these features that can be use as a good summary of all the features used and their suitability.

Firstly, the local energy or luminance is extracted from the pyramidal representation of the image by using a weighted-average operator over the three color components, as done in [154]. A set of luminance features is produced and it encodes the on-off image intensity contrast that is accepted to exist in the visual system as shown in [155].

In the case of color, the input images are decomposed in its three red (r), green (g), and blue (b) channels and normalized. These maps are widely used in the literature and there are physiological evidences of its presence in the visual processing pathway ([156] [157]).

Finally, the local orientation maps are generated for 0°, 45°, 90°, 135°. The use of four maps that encode the local orientation contrast has also been used in several works ([158] [159]).

The next stage consists in applying the center-surround module. As explained, the visual system is sensitive to the channel local contrasts rather than to their magnitude. The center-surround module is computed as a difference between a coarse and a fine scale. The different across-scale combinations make a total of 6 operations obtaining consequently 6 maps for each feature. Thus far, 7 feature maps are generated (1 for local contrast luminance, 2 for red-green and blue-yellow opponencies and 4 for local orientation contrast) and a total of 42 maps after applying the across-scale center-surround computation.

A subsequent stage carries out the combination of the information of these 42 maps into a single saliency map. The aim of this module is integrating the maps for each feature into a unique conspicuity map representation and then, in a second step, generating a single final saliency map. This final saliency map will give us the topographical representation that we need to find the maximum saliency and hence, to select the location that has to be attended next.

The first main problem is the combination of different primitives with different magnitudes. Additionally, a second issue is related with the number of maps for each feature: in our case, 6 local energy contrast maps, 12 local color-opponency maps and 24 local orientation contrast maps that cannot be summed directly in order to achieve a fair result. As Itti explains in [126] for instance, a salient stimulus in an orientation may keep unnoticed because of the noise produced by a background of strong responses due to luminance contrast. In his work [19], Itti also proposed several methods for this combination of the features to obtain the conspicuity and then, the final saliency map, avoiding the mentioned issues:

- A naive summation. This first method uses a simple summation of the different maps considering a weighted sum for the different features but without specifying these weights or a possible generation.
- A linear combination based on supervised learning. This second method consists in multiplying globally each map by a weighting factor. However, in this case, the weighting factor is computed using a set of images in which targets have been previously selected. The computation is based on a training process that consists in increasing the factor for those maps that respond better to the target and decreasing the factor for those ones with low responses, discarding maps whose factor converge to zero. Regarding to the drawbacks, we are computing only the saliency of the target areas and not of the whole map. In addition to this, a top-down supervision is mandatory in this case and it means losing generality too. However, this approach may be well-suited to integrate concepts of top-down attention modules.
- A content-based global non-linear amplification. This proposal reinforces globally maps with a small number of strong responses and inhibits those ones without meaningful activity contrast responses. The

computation is performed computing global and local maxima for the different maps and areas: a large difference between the global maximum and the average local maxima means that the map has some significance, a small result is identified as homogeneity and therefore the map is suppressed. The main advantage of this design is its suitability for a massively parallel implementation. On the other hand, its main problem is that in the presence of two quite strong stimulus but with a similar response, the algorithm discards the map. Another problem refers to the low robustness against noise (in particular, salt-and-pepper).

- An iterative localized interaction method. This alternative reproduces three interactions observed in the visual processing: there is an inhibitory interaction from the surround to the center of the different areas [160], this inhibition is stronger for the neurons that are tuned to the same feature [161] [162] and finally, this inhibition is stronger at a distance from the center and it is weaker for shorter and longer distances [163]. These three principles are modeled using difference of Gaussians. The implementation consists in convolving iteratively the normalized feature maps using the difference-of-Gaussian filter whose parameters are also provided in [126]. The computation is performed using two separable convolutions, the excitatory and the inhibitory Gaussian, in a process of 10 iterations. In this case, one of the previous scheme problems is overcome: in a map with several equally strong responses the map is discarded while, with a few very strong responses the map is promoted. The main drawback of this scheme is its iterative nature.

The following stage of the algorithm after selecting the alternative for the normalization consists in an across-scale summation to achieve the conspicuity maps: local energy, local color, and local orientation. The justification for the use of these three maps is based on the hypothesis that the competition within the same feature is stronger and that the different features contributes independently to the saliency map [126]. After this summation stage, the last step simply generates a final saliency map.

As mentioned, the maximum peak of the saliency map corresponds with the next location to be attended. The maximum operator is performed using a winner-take-all network [128]. This process selects a location that is visited as the most salient one and, a inhibition of return process that suppresses the stimulus at this location to visit the next most salient locations in an iterative manner, avoiding revisiting.

A criticism about the Itti and Koch model is the lack of top-down bias. This scheme exclusively models the bottom-up saliency stream without using any knowledge about the environment which might be a real problem performing a task in natural scenes. However, this problem is also partially solved in our implementation by using high-level cues as the motion and the depth of the objects and also a feedback to adapt the weights of the different channels for the bottom-up saliency system depending on a specific target application.

Implementations for real-time attentional systems

In this subsection we summarize some of the most important works related with the implementation in real-time of visual attention systems. We summarize different works from 2000 to 2011 based on different platforms as several standard computers, graphical processing units (GPUs), DSPs and hardware processors, analog VLSI chips and FPGAs. Apart from this, attention is a quite open concept and the different approaches are very diverse as will be seen in this section. Hence, the comparison of the results or the computational load or resource consumption is not easy due to the diversity of available architectures and systems in the literature. Although these real-time approaches are attempts to integrate attention concepts within low-level vision models, most of them aim at solving specific problems (for instance robotic vision) rather than emulating human vision concepts, as in the attention models previously indicated.

A first interesting approach was proposed by Stasse et al. in 2000 [164], the idea is to develop a visual system for a humanoid robot, which can intervene in non-specific tasks in real-time. It implements a visual system with log-polar mapping, oriented filters and the FeatureGate [165] visual attention mechanism. Implemented in a cluster of 7 PCs (Pentium II and III at 500 MHz), with a total time of 68 ms for images of 64x64. A second interesting approach was presented by Indiveri in 2001 [166] that fabricates an analog VLSI chip. This chip makes use of a spike-based representation for receiving input signals, transmitting output signals and for shifting the selection of the attended input stimulus over time. It is able to simulate 64 pixels of the network at 500Hz. Park et al. 2003 [167], proposed an active vision system that mimics human-like bottom-up visual attention using saliency map model based on independent component analysis, using a camera, DC motors, PID controllers, and a DSP board. They included edges instead of orientation maps and symmetry and the inhibition-of-return mechanism and the normalization process is carried out with supervised learning using ICA (independent component analysis) over the feature maps. The paper does not give information about performance. Next, Ouerhani et al. 2003 [168], presented a real-time implementation on a SIMD architecture called ProtoEye, a 2D array of mixed analog-digital processing elements. The analog part implements the spatial filtering-based transformations and the normalization and, the digital part does the rest. With 64x64 pixel

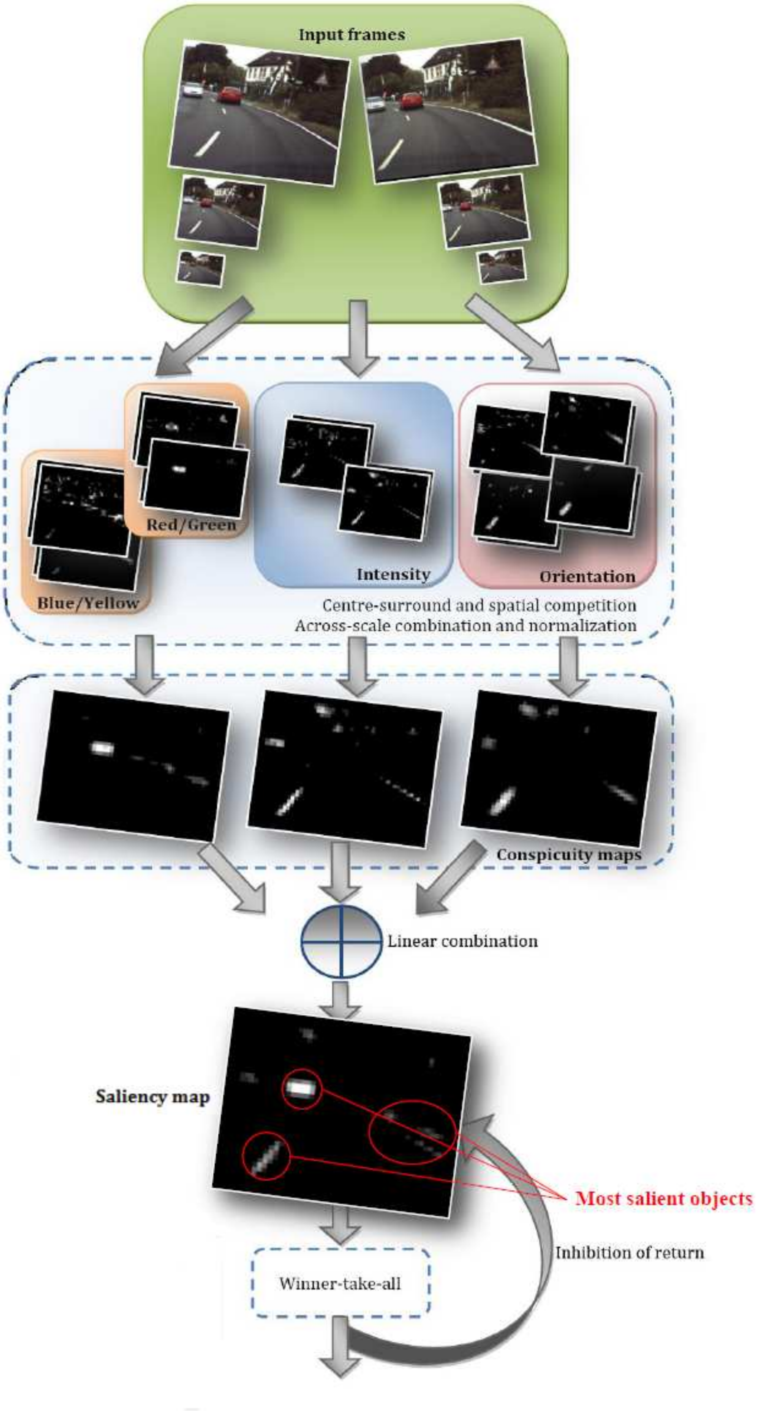


Figure 24: Itti and Koch's model [6] example. The figure shows an example of application of the model with a pair of images of a driving scenario.

images the final system runs at 14 fps. Longhurst et al. in 2006 [169] propose a GPU-based saliency map computation in real time. The visual attention computation is based on Itti and Koch's work [6] and they included some other features as motion, depth and habituation as top-down bias that are directly summed to the conspicuity maps to obtain the saliency map. With a 512x512 image, this implementation achieves about 32 fps using a Nvidia 6600GT. In another work, Chalimbaud and Berry 2007 [170], designed an embedded active vision system implemented in an FPGA. In this work, some examples of fixation and attention are implemented. In the case of attention, it based the method in a difference of frames thus, this algorithm looks for motion in a sequence of images. The implementation is performed in an FPGA (an Altera Stratix EP1S60) for VGA-resolution images but the paper does not give information about the processing time or the frames or maps per second. Frintrop et al. in 2007 [171], apply a method for extracting the feature maps based on integral images and design an attention system called VOCUS (Visual Object detection with a CompUtational attention System). The computation of the maps involves luminance, orientation and color (but not color opponencies). With their most optimized implementation with a 400x300 images the system takes 0.050 s (20 fps) in a standard 2.8 GHz computer. Another GPU-based implementation is proposed by Han and Zhou in 2007 [172], with a Nvidia 6800 GT, authors obtain 122.5 fps for a VGA resolution with a model that also includes PCA (principal component analysis) for the saliency fusion but using only 3 multiscale levels.

More recently, Xu et al. in 2009 [173] presented a work for an implementation of a saliency map model using a multi-GPU platform (up to 4 GeForce 8800 GTX) achieving up to 313 fps at VGA resolution, implementing Itti and Koch's model [6]. Bae et al. 2011 [174], proposed an FPGA based implementation for attention system based on information maximization (AIM). This model is implemented in a Virtex-6 LX240T and achieves around 11.2 fps for VGA resolution. Finally, Kim et al. 2011 [175], used a neuromorphic retina chip as an input and a bottom-up saliency map implemented in an FPGA. The saliency map is modeled using the information of the luminance and the edges. It includes a normalization stage and also, a IOR stage to select more than one location to visit. The silicon retina used 128x128 pixels and computed a frame in 4 ms, using a Xilinx X3CS400.

Visual attention architecture on-chip

This last Section presents our proposal for visual attention architecture. Some part of this work was developed during a research stay in Stanford University, California, USA, at the Department of Bio-engineering and under the supervision of Prof. Kwabena Boahen.

As commented before, our model is based on the Itti and Koch's model [35] for the bottom-up saliency stream but, it also includes a top-down modulation that is based on some features computed previously in Section 5.2.1: the optical flow and the disparity. Compared with the previous presented works, the contribution of this system is that this model is one of the first that combines the top-down and the bottom-up attentional streams. Moreover, it is also embedded on a unique chip, in order to reach real-time performance. The work presents a complete study about the integration of motion, and color, orientation, and energy as visual modalities for the saliency deployment. On the other hand, the work also presents the integration of optical flow (includes motion magnitude and direction) and disparity, as top-down cues to modulate the system response. In addition to this, the work also shows its potential for driving scenarios.

The design of the architecture represents a very complex problem due to its complexity and thus, to the required resources. As it may be seen in the associated work, our resource limitations involve a thorough study of strategies for simplifying the model and sharing the used resources, without losing significant accuracy.

The first stage of our implementation consists in extracting the features that we mentioned in Section 5.4: intensity, orientation, and color opponencies. In our case, we also extend the original model with motion (only motion magnitude, without the direction).

Intensity is computed as a simple average of the color channel values. The orientation maps are generated using Gabor filters for orientations 0°, 45°, 90°, and 135°. These Gabor filters have been also widely used in the generation of some processing engines in several previous works [41] [43] [40]. Gabor filters approximate accurately the receptive visual fields for the orientation-selective neurons in primary visual cortex ([176]).

In the case of color opponencies, instead of using the computation proposed by Itti and Koch [6], we use the computational model of [177] that simplifies the complexity of the model. This computation obtains two different sets of maps for the red-green and the blue-yellow channels.

In the case of the across-scale center-surround modulation, our implementation uses 6 different spatial scales, having then 42 maps. Those maps are combined in a different way than in Itti and Koch's work because we have a different number of spatial scales. Our combination performs the center-surround modulation between scales 5-2, 4-2, 4-1, 3-1, and 3-0.

After the across-scale modulation, the information has to be combined into the conspicuity maps but we require a previous normalization stage. In our case, this normalization uses the iterative localized interaction alternative. As mentioned, its inherent sequential nature is its main drawback.

Nevertheless, this problem may be partially solved by reducing the number of iterations and facing a trade-off between processing time and accuracy. In our implementation, we reduce the number of iterations of the model to only one and give more weight to the DoG convolution trying to simulate the original iterative summation. This part is one of our main contributions towards a hardware-like model.

Finally, we compute the saliency as a linear combination of the normalized conspicuity maps and then, the result is normalized again. This final linear combination is modulated by two high-level cues that represents the top-down bias: the disparity and the optical flow. As commented, the main criticism of the Itti and Koch's model is the lack of a top-down stream. However, this problem is also partially solved in our implementation by using that top-down stream and also a feedback to adapt the weights of the different channels for the bottom-up saliency system depending on a specific target application.

The benchmarking of the system is performed using the Itti's Lab databases [59]. This source provides images of different frameworks but of special interest in our case are the driving-scenario sets. The benchmark consists in computing the error as the number of tries that the system carries out to select the most relevant feature according to the ground-truth. The performance is measured as the average error of these false-positive tries.

However, the previous benchmark only tests the saliency computation. In our case, we present a model that integrates in a single system the saliency and the modulation streams. As an example in a driving scenario, the top-down stream can be modulated to select the vehicles that are moving towards us and moreover, the ones that are less than 10 meters away. This adaptive selection helps a driver to be aware to the vehicles that actually represent a problem for instance, in a road crossing. This small example helps us to understand the relevance that such a powerful mechanism means if integrated in a car, for the traffic security. Fig. 25 visualizes the commented scenario. As seen, in the left image we are using our implementation of the Itti and Koch's model as presented in Fig. 24 without any top-down modulation and, the vehicles, the road markers and signs are salient areas. After applying modulation, the right saliency map clearly highlights the vehicle that is driving towards us, and that is at a medium distance: not too close to be our own car and not too far to be a car which we do not have to care about.

As seen, we test our model for one of the target applications mentioned in Section 5.4: the driving assistance systems. Selecting the cues in a smart way leads us to obtain the response of the system to isolate the information of, for instance, the moving vehicles instead of the static elements.

The journal article associated to this part of the dissertation is:

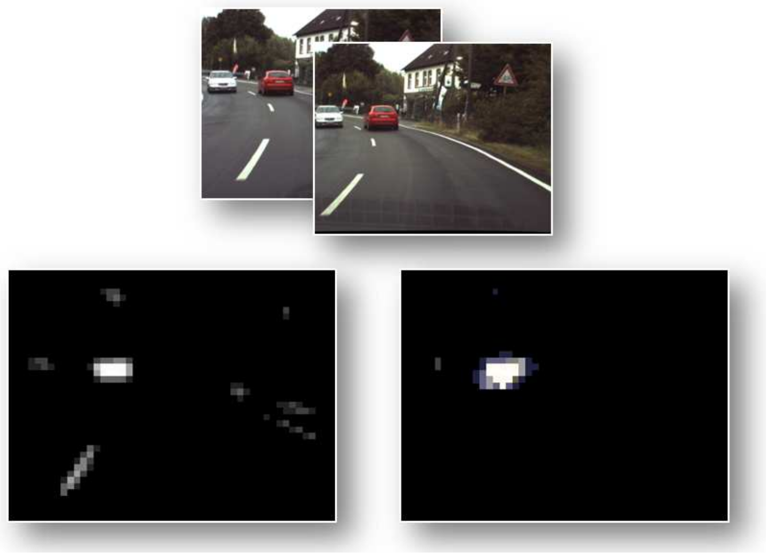


Figure 25: Top-down modulation impact. Saliency map computed using only the bottom-up explained stream (left) or biased by a top-down stream that is deployed using motion and depth cues.

F. Barranco, J. Diaz, B. Pino, E. Ros. “Saliency-based Visual Attention Architecture for FPGA with Top-Down Modulation”. **In submission to IEEE Trans. Systems, Man, and Cybernetics, Part A**.

6 Conclusions

This thesis dissertation explores dynamic vision systems for embedded devices. In the first place, in Section 5.1 we propose a dynamic bio-inspired system, low-cost in terms of resources, that applies attention for the computation of optical flow. This system selects the most relevant cues that come from a retina-like sensor and use additional information to modulate the output of the system allowing a most precise computation for specific areas, determined by the task or the locations of the targets.

On the other hand, we also present the implementation of a complete active vision system for dynamic environments. The work is based on previous approaches, hardware architectures developed for implementing some parts of these systems: local contrast descriptors (energy and orientation), binocular disparity, and optical flow. The implementation of all these visual modalities on a single chip motivated a carefully choice in the selection of the most appropriate algorithms. Despite the current state-of-the-art provides examples of algorithms that generate very accurate and dense results, our choice selecting a gradient-based algorithm is justified by its good trade-off between accuracy and resource utilization. Since the system that integrates the mentioned visual modalities is thought to be a first layer of visual engines for the final system, the moderate use of resources for their computation is quite relevant.

Next, our systems are developed for its use in real-world applications with real-time requirements. The accomplishment of real-time performance is achieved by using devices with potential capabilities for exploiting massive parallel architectures, in our case FPGAs.

The implementation of the hardware architectures for the mentioned visual modalities are performed in an incremental manner. The first one is described in Section 5.2.1 that presents a first architecture for the computation of multiscale optical flow computed using the gradient-based Lucas&Kanade algorithm [84]. It studies the most important aspects of the general hardware architecture that is replicated with some variations in the number of the modules for the subsequent architectures. This Section also details the complete architecture that conforms the basis of the subsequent systems. In this case, for each input image, the system generates results equivalent to eight input images. This sentence illustrates that one of the most important aspect of this last work is the effort dedicated to integrate the computation of a huge amount of information: local energy, local orientation, binocular disparity and optical flow. Section 5.2.2 and Section 5.2.3 present variations of this first design, including the computation using color cues and an algorithm that allows to reduce the final resource utilization.

After generating the system to extract the required visual features, the system developed in Section 5.3 is designed using just the binocular disparity computation implemented in Section 5.2.1, adapted and extended for the computation of vector disparity (due to the limitations of an active system for implementing the rectification of the images). This system is used for an active control of gazing, controlling the vergence, tilt and version of a robotic head. The final system is tested for its use in focusing an object at different depth planes ("fixation"), and also for active tracking of objects moving to variable velocities in the 3D space ("smooth pursuit").

Since the vision engines are implemented with a moderate use of resources, the extension of the system with extra layers or modules for new processing is affordable. Thus the system in Section 5.2.1 is extended, being used as part of an attentional vision system in Section 5.4. The implemented system is based on the algorithm proposed by Itti and Koch [6], that is used as a metric for comparing with other attention implementations. This algorithm is complemented by adding new inherent bottom-up features (the motion) and also with task-dependent top-down modulation that biases the final competition of the most salient locations (with depth and optical flow cues). The system is also adapted by the use of parameters for the weight of the different features that compute the saliency of the locations. The system may be also useful for instance for military applications, industrial inspection, autonomous navigation, video-surveillance, or aid devices for low-vision patients. Finally, in our case this system is tested in the framework of driving assistance systems.

6.1 Future work

In this section, we provide some pointers to future lines of research that stem from the proposals described in this dissertation.

- A new second layer of visual processing engines may be developed and integrated in our system using new strategies for resource sharing or reduction. This new layer of modules would use the estimates generated by the modules developed for Section 5.2.1. Some valuable examples are the detection of IMOs (independently moving objects), trackers based on relevant point descriptors (such as SIFT or SURF), or SLAM methods (simultaneous location and mapping). All these new modules are very interesting for driving scenarios. And in that field, a complete dynamic system that would combine all that information could be modeled and implemented.

- The integration of the active gaze control and the visual attention in a single system. This may help exploring new mechanisms of combination of overt and covert attention for real applications, specially well-suited for robotics.

- The extension of the bottom-up saliency of the system in Section 5.4 with new cues. According to some studies from Wolf and Horowitz [153] [178] [179], new features such as texture, symmetry, or different kind of junctions (see also Julesz [60] [61]) may be added in order to obtain a more accurate saliency deployment. The possibly-improved modeled could be applied for the system of the first point of this list.

- The attention model implemented for Section 5.4 has a potential application specially in the case of low-vision aid devices. Low vision is defined as partial sight that is not fully correctable with surgery, pharmaceuticals, contact lenses or glasses. Low-vision aid devices improve the visual capabilities taking advantage of patients' residual vision, using anti-glare filters, augmented reality devices, telescopes or, in our case, head-mounted electro-optical systems. Applying attention we should be able to highlight in real time some cues that are extracted from the scene and integrate them with the residual vision and simultaneously, reduce the less relevant information. Actually, according with [180] [181], inattentional blindness may reduce the effective detection of important events by 50% in average. Specifically, the top-down modulation might play a very important role because that process could be biased according with the user interest, the task that is being performed, or the device configuration. Furthermore, crowding is a harmful effect that reduces ability of recognizing objects in the peripheral vision when there are other objects near to the target of interest. Attention can help isolating this target by blurring or suppressing all the other objects. Studies with patients involving real-life tasks such as watching TV, office and house work, or even driving could be done.

- As regards the bio-inspired attention mechanism developed in Section 5.1, the development of a more complex attention mechanism based on the retinal cues could also be done. Working with sparse responses makes more difficult to solve the problem but it also reduces its resource requirements. The study of this field might help us to understand the way in which our brain deploys attention and then, to develop new and more efficient strategies to deal with dynamic systems.

6.2 Main contributions

In this section we briefly present the main contributions of this Ph.D. thesis.

- We have shown that an adaptive bio-inspired model can accomplish the vision tasks of computing optical flow using sparse retinal cues and specially, with limited resources.

- This bio-inspired model is tested for the computation of optical flow and we have demonstrated that by smartly selecting the cues we can improve the results accuracy. Moreover, the way in which the different kind of cues are integrated is also crucial for the final precision and density of the system.

- Additionally, an attentional multimodal mechanism based on this bio-inspired retinal model has been developed. This process allows biasing the selection of the cues in a top-down manner according with additional information as color cues, texture or object speed, helping to reduce the required resources of the system.

- We have developed architectures for the computation of optical flow, disparity and local descriptors (energy and orientation) for real-time embedded systems. All those visual modalities have been separately benchmarked.

- Furthermore, a final architecture for the computation of all these visual features has been designed and implemented in a single chip. For that, different strategies to share or reduce the resource requirements have been studied.

- Diverse alternatives for the computation of the previous visual modalities have been addressed. Firstly, we have designed and implemented optical flow and disparity using color information. This alternative provides more accurate and density to the estimations with an affordable cost of resources. We have shown that the adoption of this alternative has to be done considering the trade-off accuracy and density vs. cost.

- Secondly, we have also modeled and implemented a fusion scheme instead of the classical multiscale-with-warping for the optical flow computation. This approach reduces the resource utilization and can be applied to many applications that do not have strong accuracy requirements. This new scheme provides a lower latency which is especially interesting for collision avoidance applications.

- A system for the active control of gaze of a robotic head by using the vector disparity computation has been developed. This system successfully emulates two neural mechanisms, fixation and smooth tracking, by controlling vergence, version, and tilt.

- Finally, a method for the deployment of the bottom-up inherent stream for the implementation of visual attention has been designed. It is based on Itti and Koch's works but also includes a new bottom-up feature, the motion, and a study to reduce its computational complexity.

- We have also designed and implemented a system for the previous model that integrates bottom-up salient cues and top-down task-dependent modulation, using optical flow and disparity estimates. The system has been embedded achieving real-time performance.

- The accuracy of the saliency of this system has been validated by using well-known benchmarks. Our hardware-like model provides good performance compared with the original Itti and Koch's model.

- The top-down modulation has been also evaluated by applying the system to driving scenarios. This stream includes optical flow and disparity cues to bias the selection of the most relevant information of the scene.

Conclusiones en Castellano

Esta tesis explora los sistemas dinámicos de visión para dispositivos empotrados. En primer lugar, en la Sección 5.1 proponemos un sistema bioinspirado dinámico, de bajo coste en términos de recursos, que aplica la atención visual para la computación del flujo óptico. Este sistema selecciona las estimaciones o respuestas más relevantes que vienen de un sensor retinal y utiliza más información adicional para modular la salida del sistema permitiendo una computación mucho más precisa en ciertas áreas, determinadas por la tarea o porque en esas zonas se encuentran los objetivos.

Por otro lado, también presentamos la implementación de un sistema de visión activo completo para entornos dinámicos. El trabajo está basado en aproximaciones previas, arquitecturas hardware desarrolladas para implementar algunas partes de estos sistemas: los descriptores de contraste local (energía y orientación), disparidad binocular y flujo óptico. La implementación de todas esas modalidades visuales en un único chip motivó que se llevara a cabo una elección muy cuidadosa en la selección de los algoritmos más apropiados. Aunque el actual estado del arte ofrece ejemplos de algoritmos que calculan estimaciones muy precisas y con resultados densos, nuestra elección seleccionando un algoritmo basado en gradiente está justificada por el equilibro que éstos mantienen entre precisión y utilización de recursos. Como el sistema que integra las modalidades visuales se piensa que es la primera capa de motores visuales para el sistema final, el uso moderado de recursos para su cálculo es muy relevante.

A continuación, nuestros sistemas son desarrollados para su uso en aplicaciones con requisitos de prestaciones tiempo real. Las prestaciones de tiempo real son conseguidas gracias al uso de dispositivos con capacidades potenciales para la explotación de arquitecturas masivamente paralelas, en nuestro caso, las FPGAs.

La implementación de las arquitecturas hardware para las mencionadas modalidades visuales es llevada a cabo de forma incremental. La primera es descrita en la Sección 5.2.1 y presenta una primera arquitectura para la computación de flujo óptico multiescala utilizando el algoritmo basado en gradiente de Lucas&Kanade [84]. Estudia los aspectos más importantes de la arquitectura hardware general que es replicada con algunas variaciones en los módulos desarrollados para las siguientes arquitecturas realizadas. Esta Sección también detalla una arquitectura completa que conforma la base de los sistemas siguientes. En este caso, para cada imagen de entrada, el sistema genera resultados que son equivalentes a la información de 8 imágenes como las de entrada. Esta frase ilustra que uno de los aspectos más relevantes de este último trabajo es el esfuerzo dedicado a integrar la computación de una gran cantidad de información: la energía y orientación locales, la disparidad binocular y el flujo óptico. La Sección 5.2.2 y la Sección 5.2.3 presentan variaciones de este primer diseño, incluyendo el cómputo usando la información de color y un algoritmo que permite reducir la cantidad final de recursos utilizados.

Tras generar el sistema que extrae las modalidades visuales requeridas, el sistema desarrollado en la Sección 5.3 es diseñado usando sólo la disparidad binocular que se implementó para la Sección 5.2.1, adaptada y extendida para calcular la disparidad vectorial (debido a la limitación de un sistema activo para llevar a cabo la rectificación de las imágenes). Este sistema es usado para el control activo de la mirada, controlando la vergencia, el giro sobre los ejes vertical y horizontal de los ojos de una cabeza robótica. El sistema final es evaluado para su uso centrándose en un objeto que se mueve en diferentes planos de profundidad ("fijación"), y también para seguimiento activo de objetos que se mueven a velocidad variable en el espacio 3D ("seguimiento suave").

Puesto que los motores de visión están implementados con un uso moderado de recursos, se puede abordar la extensión del sistema con capas o módulos extra. Por tanto, el sistema de la Sección 5.2.1 es extendido, siendo usado como parte de un sistema de visión atencional en la Sección 5.4. Este sistema implementado está basado en el algoritmo que propusieron Itti y Koch [6], que es además usado como métrica para comparar los nuevos algoritmos e implementaciones. Este algoritmo es completado añadiendo nuevas características que funciona de abajo a arriba (el movimiento) y también con un flujo de modulación que funciona de arriba abajo y que depende de la tarea que se está llevando a cabo, que sirve para modificar la competición final para la elección de la localización más relevante (con estimaciones de profundidad y flujo óptico). Este sistema también se adapta para el uso de los parámetros para pesar las diferentes modalidades visuales con las que se computa el flujo de la saliencia. Este sistema puede ser útil para por ejemplo, aplicaciones militares, de inspección industrial, de navegación autónoma, de video-vigilancia, o para dispositivos de ayuda para pacientes con problemas de visión. Finalmente, en nuestro caso el sistema es aplicado y testeado con éxito en el marco de sistemas avanzados de asistencia a la conducción.

Trabajo futuro

En esta sección proponemos algunos trazos de posibles futuras líneas de investigación a partir de las propuestas que se han descrito en esta tesis.

- Una segunda capa de motores de procesamiento visual puede ser desarrollada e integrada en nuestro sistema utilizando diferentes estrategias para la reducción o compartición de recursos. Esta nueva capa de módulos usaría las estimaciones generadas por los módulos desarrollados para la Sección 5.2.1. Algunos ejemplos valiosos pueden ser la detección de IMOs (objetos que se mueven de forma independiente), operadores de seguimiento basados en descriptores de puntos relevantes (como SURF y SIFT), o métodos SLAM (de mapeo y localización simultáneos). Todos estos nuevos módulos son muy interesantes para escenarios de conducción. Y en ese campo, un sistema completo y dinámico que combinara toda esa información sería modelado e implementado.

- La integración del sistema de control activo de la mirada y el de atención visual en un único sistema. Esto puede ayudar a explorar nuevos mecanismos para combinar diferentes métodos de atención para aplicaciones reales, especialmente interesantes en el caso de la robótica.

- La extensión del flujo de saliencia de la Sección 5.4 con nuevas características. De acuerdo con algunos estudios de Wolf y Horowitz [153] [178] [179], podrían añadirse nuevas características como la textura, las simetría o diferentes tipos de uniones (ver también Julesz [60] [61]), para obtener un saliencia más precisa. El modelo posiblemente mejorado podría ser aplicado al sistema del primer punto de esta lista.

- El modelo de atención implementado para la Sección 5.4 tiene aplicación potencial especialmente en el caso de dispositivos de ayuda a pacientes con baja visión. La baja visión se define como vista parcial que no puede ser totalmente corregida con cirugía, medicamentos, lentes de contacto o gafas. Los dispositivos de ayuda para baja visión mejoran las capacidades visuales tomando ventaja de la visión residual que aún tienen los pacientes, utilizando filtros anti-reflejos, dispositivos de realidad aumentada, telescopios o, en nuestro caso, dispositivos electro-ópticos que se colocan en la cabeza a modo de gafas. Aplicando la atención podríamos ser capaces de destacar en tiempo real algunas estimaciones que son extraídas de la escena e integrarlos con la visión residual de forma simultánea, reduciendo la información menos relevante de la escena. De hecho, de acuerdo con [180] [181], la ceguera atencional puede reducir la detección efectiva de los eventos más importantes en un 50% en media. Especialmente, la modulación de arriba abajo puede jugar un papel muy importante porque ese proceso podría ser modulado de acuerdo con el interés del usuario, la tarea que está llevando a cabo, o la configuración del dispositivo. Además, otro efecto pernicioso, el “crowding”, reduce la habilidad de reconocer los objetos en la visión periférica cuando hay otros objetos cerca del de interés. La atención puede ayuda aislando el objetivo emborronando o suprimiendo los otros objetos. Los estudios con pacientes conllevan tareas en tiempo real como ver la televisión, trabajo en el hogar o la oficina, o incluso en tareas de conducción.

- En lo que respecta al modelo bio-inspirado con el mecanismo de atención que es desarrollado en la Sección 5.1, el desarrollo de un mecanismo de atención más complejo basado en respuestas retinales puede también ser llevado a cabo. Trabajar con información poco densa dificulta resolver un problema pero también ayuda a reducir los recursos necesarios. El estudio de este campo puede ayudarnos a entender la forma en que el cerebro calcula la atención y entonces, desarrollar nuevas y más eficientes estrategias para trabajar en entornos dinámicos.

Principales contribuciones

En esta sección resumimos las principales contribuciones de esta tesis

- Hemos demostrado que un modelo adaptativo bio-inspirado puede llevar a cabo con éxito tareas de visión como el cómputo de flujo óptico con información poco densa a partir de respuestas retinales y especialmente, con recursos limitados.
- Este modelo bio-inspirado es testeado para el cálculo de flujo óptico y hemos demostrado que seleccionando de forma inteligente las respuestas podemos mejorar los resultados de precisión. Además, la forma en que los diferentes tipos de respuestas son integrados es también crucial para la precisión y densidad finales del sistema.
- Además, se ha desarrollado un mecanismo atencional multimodal basado en el modelo retinal bio-inspirado. Este proceso nos permite modular la selección de las respuestas trabajando de arriba a abajo de acuerdo con cierta información adicional como puede ser el color, la textura o información de velocidad de los objetos. Todo esto ayuda a reducir los recursos necesarios para el sistema.
- Hemos desarrolado diferentes arquitecturas para el cómputo de flujo óptico, disparidad y descriptores locales (energía y orientación) para dispositivos empotrados en tiempo real. Todas esas modalidades visuales han sido además testeadas individualmente.
- Además, se ha diseñado e implementado en un único chip una arquitectura final para el cómputo de todas ellas. Para ello, se han estudiado diferentes estrategias para compartir y reducir los requisitos de recursos necesarios.
- Se han llevado a cabo diferentes alternativas para el cómputo de las modalidades visuales anteriores. En primer lugar, hemos diseñado e implementado el flujo óptico y la disparidad usando información de

color. Esta alternativa nos ofrece estimaciones más precisas y densas con un coste razonable dependiendo de la tarea que se quiere llevar a cabo. Hemos mostrado también que la adopción de esta alternativa tiene que ser considerada cuidadosamente en función de la precisión, la densidad y el coste.

- En segundo lugar, también hemos modelado e implementado un esquema de función en lugar del modelo clásico de refinamiento escala a escala para el cómputo del flujo óptico. Esta aproximación reduce la utilización de recursos y puede ser aplicada a muchas aplicaciones que no tienen requerimientos de precisión muy fuertes. Este nuevo esquema nos da baja latencia lo que es especialmente interesante para aplicaciones para navegación evitando obstáculos.

- También se ha desarrollado un sistema para el control active de la mirada de una cabeza robótica. Este sistema emula con éxito dos mecanismos neuronales, la fijación y el seguimiento suave, controlando la vergencia y el giro sobre el eje vertical y horizontal de los ojos.

- Finalmente, se ha diseñado un método para la generación del flujo de saliencia de un modelo de atención visual. Está basado en los trabajos de Itti y Koch pero incluye nuevas características como el movimiento, y un estudio para reducir la complejidad computacional del mismo.

- Hemos diseñado e implementado un sistema para el modelo anterior que integra ese flujo de saliencia junto con un flujo de modulación que depende de la tarea que se lleva a cabo, usando el flujo óptico y la disparidad. Este sistema se ha empotrado consiguiendo también prestaciones en tiempo real.

- La precisión de la saliencia del sistema ha sido validada usando test bien conocidos en el campo. Nuestro modelo hardware nos da unas prestaciones muy buenas en comparación con el original.

- El flujo de modulación ha sido también evaluado aplicado a sistemas para escenarios de conducción. Este flujo incluye el flujo óptico y la disparidad para modificar la selección de la información más relevante de la escena.

Part II. Publications

"The world is moving so fast these days that the man who says it can't be done is generally interrupted by someone doing it."
[Elbert Hubbard]

1 Bio-inspired attention model for motion estimation

1.1 Active visual system for motion estimation with an artificial retina

The journal paper associated to this part of the dissertation is:

- F. Barranco, J. Díaz, E. Ros, B. Pino. Visual system based on artificial retina for motion detection IEEE Trans. on Systems, Man and Cybernetics - Part B:Cybernetics, vol. 39, no. 3, pp.752-762, 2009. DOI 10.1109/TSMCB.2008.2009067.
  - Status: **Published**.
  - Impact Factor (JCR 2010): 2.699
  - Subject Category:
    - ∗ Automation and Control Systems. Ranking 3 / 60.
    - ∗ Computer Science, Artificial Intelligence. Ranking 13 / 108.
    - ∗ Computer Science, Cybernetics. Ranking 3 / 19.

Visual System Based on Artificial Retina for Motion Detection

Francisco Barranco, Javier Díaz, Eduardo Ros, and Begoña del Pino

***Abstract*—We present a bioinspired model for detecting spatiotemporal features based on artificial retina response models. Event-driven processing is implemented using four kinds of cells encoding image contrast and temporal information. We have evaluated how the accuracy of motion processing depends on local contrast by using a multiscale and rank-order coding scheme to select the most important cues from retinal inputs. We have also developed some alternatives by integrating temporal feature results and obtained a new improved bioinspired matching algorithm with high stability, low error and low cost. Finally, we define a dynamic and versatile multimodal attention operator with which the system is driven to focus on different target features such as motion, colors, and textures.**

***Index Terms*—Artificial retina, bioinspired vision, block matching, motion processing, multiscale motion estimation, rank-order coding, retinomorphic chip.**

I. Introduction

VISION IS one of the most useful and efficient sensory systems developed throughout evolution. Together with the other senses, its purpose is to provide animals with information about the world so that they can operate efficiently within a changing environment to achieve their ends and help ensure their survival. The visual system in humans is quite complex and structured in multiple processing layers that deal with different aspects of the visual input [1]. Motion processing is a key function for the survival of most living beings, and so, their visual systems have specific areas dedicated to this task [2]. Neurophysiological data [3] suggest that primary visual areas are modeled using spatiotemporal receptive filters [4]–[6] to compute motion.

Artificial processing architectures designed for tasks that biological systems solve with impressive efficiency can benefit considerably from mimicking computing strategies evolved in nature over millions of years. We have developed a visual model for motion estimation that integrates different bioinspired concepts. Simoncelli and Heeger modeled how the cortical areas (V1 and MT cells) can extract the structure of motion through local competitive neural computation [6]–[8]. We have developed this model following an engineering strategy. We integrate the multiresolution scheme carried out by the brain cells using a multiscale computation scheme [9], [10], as described by Willert [11], and use rank-order coding [12] as a natural way to choose the relevant information (salient maps according to a specific cost function). The processing scheme presented in this paper is based on Boahen's retinomorphic system, which translates visual stimuli using a population of cells that mimics retinal functions [13]–[16].

The first aim of this paper was to design and implement a bioinspired model based on artificial retinas for motion estimation using multiscale and rank-order coding computation. The system uses restricted-density saliency maps and is consequently of great interest for applications with strict bandwidth constraints.

By selecting responses in an intelligent way, we significantly improve the accuracy of the region-based matching model. As will be demonstrated in Section II, neurons that fire trains of spikes with the highest energy are the most reliable ones for region-based matching. We also implement new strategies for the matching algorithm integrating temporal information. To choose the best strategy, we define a cost function by comparing different ones in the search for that with the lowest average cost and the highest stability over different scheme parameters.

Finally, we define a versatile multimodal attention operator that focuses the matching algorithm on different features such as motion, colors, textures, etc., by preselecting the input responses for the model using rank-order coding biasing. The system scheme processing is shown in Fig. 1.

II. Bioinspired Motion Computation Based on Neural Population

Our novel development is an event-driven processing scheme based on the artificial-retina model described in Boahen's work [13], [14], [17]. These events are spikes fired by the encoding neurons when they tune different spatiotemporal features.

The retinomorphic front end uses four different kinds of neurons: sustained neurons (center-surround ON_OFF and OFF_ON units) and transient neurons (temporal INCREASING and DECREASING units). They can be seen as four output cells firing specific spikes in response to concrete stimuli. These spikes represent the input data for our model.

One of the main outcomes of this paper is the study of which kind of "retinal modality" (cell type) or group is more suitable for accurate motion estimation.

Manuscript received February 19, 2008; revised July 19, 2008. This work was supported in part by the VI EU research framework funds through European Projects DRIVSCO (IST-016276-2) and DINAM-VISION (DPI2007-61683) and Junta de Andalucía Project P06-TIC-02007. This paper was recommended by Associate Editor S. Sarkar.

The authors are with the Department of Computer Architecture and Technology, University of Granada, 18071 Granada, Spain (e-mail: fbarranco@atc.ugr.es; jdiaz@atc.ugr.es; eduardo@atc.ugr.es; bego@atc.ugr.es).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCB.2008.2009067

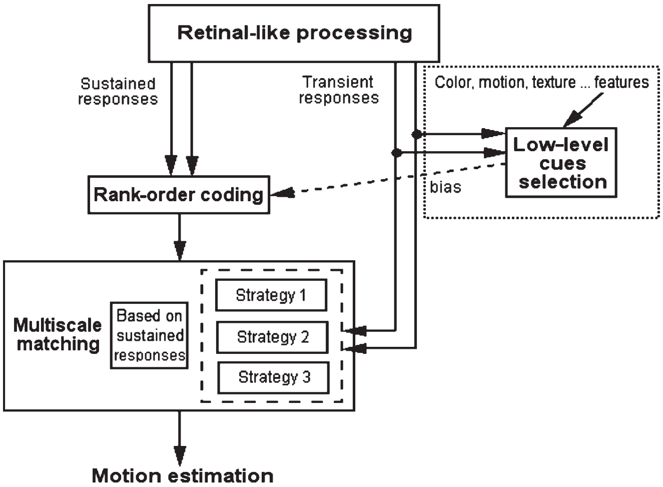


Fig. 1. System processing scheme. The dotted square is the part implemented using the attention operator described in Section IV. The cue-selection unit biases the rank order of the sustained responses according to other feature responses (e.g., transient responses, colors, textures, etc.). On the other hand, the dashed square is the unit for the different matching strategies that also integrate transient responses in order to incorporate temporal information into the matching computation.

A. Neuromorphic Chip Emulation: Sustained and Transient Neurons

We mimic Boahen's retinomorphic chip [13], [14], [17] using the following models.

1) Sustained neurons are modeled using difference of Gaussians (DoGs) with different inner and outer ratios or standard deviations to model a center-surround response model. Sustained ON–OFF neurons tune local regions with more intensity than surrounding regions; in this way, they fire a spike train with an instantaneous rate which depends on a Gaussian response. In a digital image, sustained ON–OFF neurons respond when a pixel has more intensity than the pixels in its neighborhood. Sustained OFF–ON neurons tune regions with less intensity than surrounding neighborhoods. In this case, they fire a spike train with an instantaneous rate which depends on a Gaussian response. In a digital image, sustained OFF–ON neurons respond when a pixel has less intensity than the pixels in its neighborhood. The calculus of all the sustained responses needs the definition of the different receptive fields, which are characterized by their size. The receptive field sizes for sustained neurons set the spatial features obtained or tuned (typically, we use the values of 7 and 11 pixels for the spatial center-surround Gaussian filters in our experiments).
2) Transient neurons model temporal changes with different filters. INCREASING transient neurons tune local regions where light intensity increases. DECREASING transient neurons tune local regions where intensity decreases. For transient neurons, we do not have spatial receptive fields but just temporal filters. To calculate transient responses, we use the previous and following frames, taking into account different weights for each of them. After evaluating different alternatives, we choose the one that uses the two previous, the current, and the two following frames with weights of -1, -2, 0, 2, 1, respectively.

TABLE I
PARAMETER CONFIGURATION USED FOR THE EXPERIMENTAL RESULTS. THE NUMBER OF RESPONSES IS NOT RESTRICTED BUT SET TO THE MAXIMUM NUMBER

Parameter	Value
Inner deviation (filter size)	1 (7 pixels)
Outer deviation (filter size)	1.5 (11 pixels)
Sustained threshold	5
Transient threshold	10
Temporal filter configuration	{-1, -2, 0, 1, 2}
No. of responses threshold	Maximum

In the image processing computations, errors are generated by noise, temporal aliasing, model limitations, and so on. Therefore, we need a threshold to reduce the impact of these errors. In our experiments, we used a threshold for the sustained responses with a value of 5 and another one for the transient responses with a value of 10. To produce a stable level of activation (rate of active cells), we also define a dynamic threshold. In this way we define the minimum number of responses our model needs and the other parameters are tuned to achieve it.

Only the cells with a stimulus suitably tuned to their receptive field fire a spike, and therefore, they produce a saliency map with a restricted density. We calculated the response density for different sequences and features to evaluate the activity rate (in percent) over the total number of pixels in the sequences. In standard images, using the parameters in Table I, the activity rate is around 11%–12% for each kind of response. An example is shown in Fig. 2.

Thus, our model is of potential interest for a wide variety of applications with strict bandwidth constraints. Nevertheless, activity depends strongly on the standard deviations of the DoGs used and also on the characteristics of the image.

B. Multiscale Motion Model for Spike Matching

Our motion-processing system computes a region-based matching method, as described in [19], in which we define the motion estimation as $v = (d_x = d, d_y)$ for the neighborhood of a specific cell as the best fit between the current image region and the previous one. In this way, we find the best matching region using a distance measure, the sum of squared difference (SSD) value, or the minimum mean square error (mse). In our experiments, to design the block-matching model, we implemented a full search using an exploration window of 14 pixels and a block size of 8 pixels. The block-matching example pseudocode is detailed in Appendix I-A.

The first processing stage computes the four neuron-cell responses emulating retinal processing. The second stage computes motion displacement by matching the responses produced by the sustained and transient neurons throughout every frame taken at different instants by mimicking the MT area of the cerebral cortex [6]. Bioinspired motion-estimation models are usually based on energy models due to their affinity to a neural-like description, but they are rather complex and require much

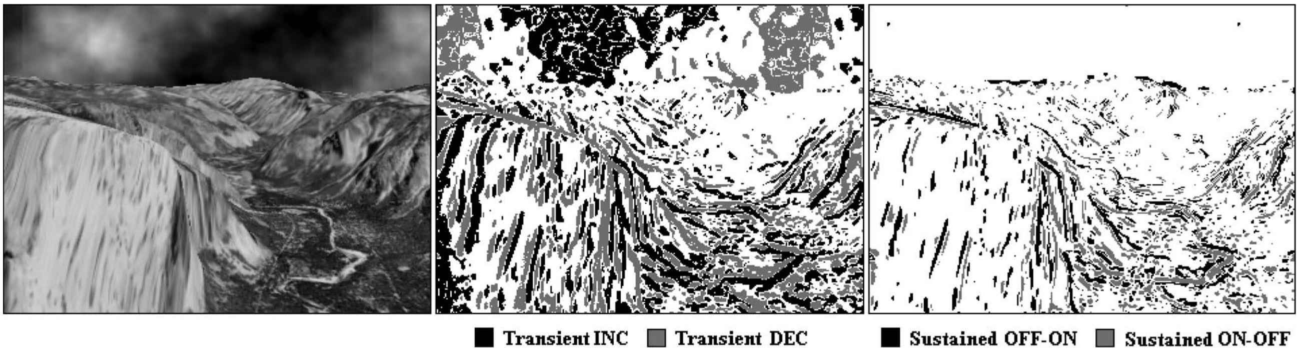


Fig. 2. Retina responses for a frame from the Yosemite sequence [18]. Activity varies from 11% to 12% for each type of neuron. (Left) Original image from the Yosemite sequence. (Center) Transient responses. (Right) Sustained responses.

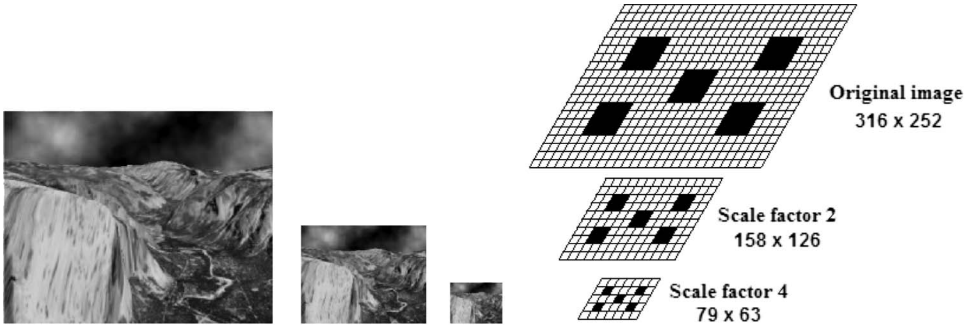


Fig. 3. (Right) Multiscale images for the Yosemite sequence. (Left) Generic scheme for multiscale with three scales and factor two, as in our tests.

more computational resources. As shown by Simoncelli [7], matching methods are equivalent to energy models but are much more efficient in terms of computational load.

For the matching process, we used a standard block-matching method, as described in [19] and [20], but relied on the neuron responses as input. We evaluated multiple matching cost functions combining the sustained and transient neuron responses. In this section, we focus on the use of sustained neurons for the matching process. In Section III, we will discuss other matching alternatives, including sustained and/or transient neurons.

We introduce here a new bioinspired element, multiscale processing [9], [10]. We use in our work three scales, one of them being the original sequence and the other two being obtained by subsampling with a factor of two. An illustrative example is shown in Fig. 3.

We apply our sustained neuron model to three well-known benchmark sequences to extract the responses (the Yosemite, the translation tree, and the diverging tree sequence). Different scales will tune better spatial features of different sizes.

At the smallest scale, the system computes the motion-estimation field, and at the next scale, this estimation is an input parameter for the new motion estimation, oversampled by a factor of two. The block matching in the next scale guides its own motion estimation using the previous motion estimation for fixing a search window of 3 pixels.

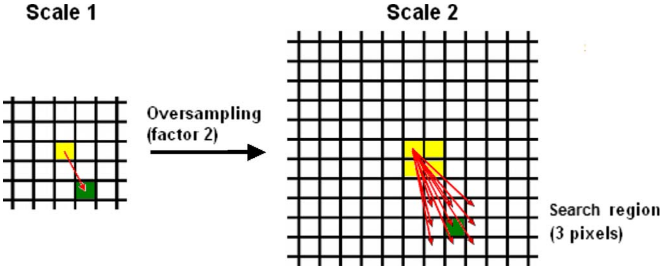


Fig. 4. Oversampling of a neuron motion estimation used for guiding the motion-estimation search at the next scale with a search window of 3 pixels, as in (1).

For instance, if the previous motion estimation is $v = (i, j)$ for the neuron n, as input parameter for the next scale, we will use $v = (2 \cdot i, 2 \cdot j)$ and oversample it by a factor of two (Fig. 4). This motion estimation guides the new motion-estimation computation by setting up a redefined search window for the algorithm and by exploring the region defined by (1)

$$E_R = \{v = (2 \cdot i + \alpha, 2 \cdot j + \beta) : \alpha, \beta \in \{-1, 0, 1\}\} \qquad (1)$$

$$E_R = \{v = (2 \cdot i + \alpha, 2 \cdot j + \beta) : \alpha, \beta \in \{-2, -1, 0, 1, 2\}\}. \qquad (2)$$

The system fixes the best motion estimation for the new scale, and we follow the same process for the last scale, the original image, using a search window of 5 pixels instead of 3 pixels, as defined in (2).

The pseudocode for the multiscale approach is given in Appendix I-B. The minimum mse motion estimation updating uses a new threshold, which is set to 1.0 in our implementation.

One of the first results obtained from the last paragraphs is that if a sequence for the smallest scale contained a large object with a high contrast between it and the background, we would obtain the motion estimation for this region, but

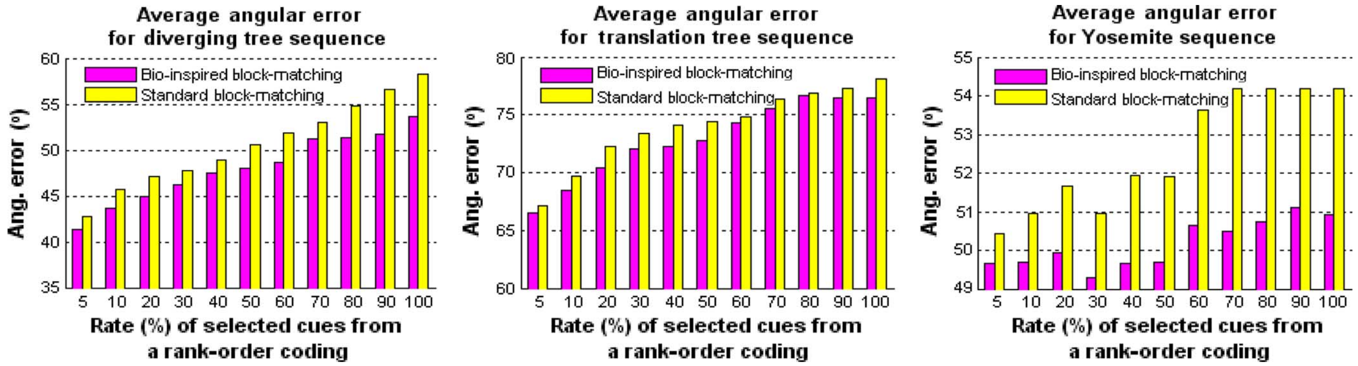


Fig. 5. Average angular error for test sequences. The rate of selected cues from rank-order coding is the percentage of responses chosen from sustained responses because of their local energy measure value (local structure support). This represents the optical flow estimation density.

if we had a small object with a low contrast, we would not detect it. This object will be invisible at the coarse scale due to the low-pass filter operations. As the motion-estimation field guides the following scale fields, we lose the smallest objects for the next scales and cannot retrieve them (this is caused by spatiotemporal aliasing).

To solve this problem, we could compute all the estimations at each scale, but the computational complexity would be very high, and there are other ways of solving these problems, as we propose in Section IV, where we explain the techniques used to reduce this computational complexity.

C. Rank-Order Coding: Contrast Versus Accuracy Tradeoff

Focusing on the use of sustained neurons, our first goal is to demonstrate that those neurons with larger firing responses produce more accurate motion estimations.

First of all, rank-order coding [12] consists of selecting only the most important responses or cues, sorting all the data or responses according to a measure or a criterion (cost function), the local energy measure for instance. The energy measure used for rank-order coding, in this case, is just the normalized sum of responses from every ON–OFF and OFF–ON sustained neuron (which, in a neural-like computing scheme, can be done by a collecting neuron). Therefore, we sort the responses by the energy values and select the highest ones according to a predefined rate threshold (in percent). The set of selected cues from a rank-order coding is the percentage of most important or most reliable responses that we are going to use in the block-matching algorithm according to the local light contrast. This is a concrete setup, but we also define this selection procedure in a different way, implementing a multimodal operator, as described in Section IV.

We use a set of well-known sequences to test our model: the Yosemite, the translation tree, and the diverging tree sequence [18]. The results are shown in Fig. 5. To facilitate the comparison of accuracy, the proposed motion-estimation model is compared with a standard block-matching method (which directly matches image luminance instead of neuron responses), as shown in Fig. 5.

In Fig. 5, there are three different plots, each of which has two different groups of columns. The first represents a bioinspired block matching, as can be seen in the legend, in which we use rank-order coding to select the most important responses and apply an adapted block-matching algorithm. The second column shows the results of a standard block-matching algorithm based on sustained neuron responses.

The metric used is the average angular error, which is the average of the angle difference between the computed motion vectors and the ground-truth ones, as defined in the following:

$$AAE = mean\left(\cos^{-1}(\hat{e} \cdot \hat{g})\right) \tag{3}$$

where AAE is the average angular error, $\hat{e}$ is the normalized estimated motion vector, and $\hat{g}$ is the normalized ground-truth vector.

An analysis of the average angular error clearly supports the bioinspired approach. As can be seen in every test, the average angular error increases concomitantly with an increase in the number of responses provided to the block-matching stage. At the top left, diverging tree results follow a stable increasing curve, while the percentage of responses taken (or rate (in percent) of selected cues from rank-order coding) is growing. Therefore, as we increase the number of cell responses provided to the block-matching stage, the system produces higher error. Furthermore, the same progression can be seen in all the tests. We also have to take into account the error difference between the two types of block-matching algorithms, and we see that the bioinspired algorithm based on cell responses always leads to the best results.

In conclusion, our working hypothesis is supported by the results: Neurons that provide the best results, i.e., neurons that fire spikes with higher energy due to their tuning regions with specific spatial and contrast features are the most reliable ones for estimating motion with block matching. Therefore, we use rank-order coding to choose the image areas with higher confidence. This is of critical importance for the subsequent processing layers.

III. Transient Neuron Integration Into the Motion Model

The transient neuron responses provide us with a way to incorporate temporal information into the matching process described in Section II. This alternative consists of integrating the increasing and decreasing (INC and DEC) transient response neurons with the cues given by the sustained neurons for our sparse block-matching algorithm.

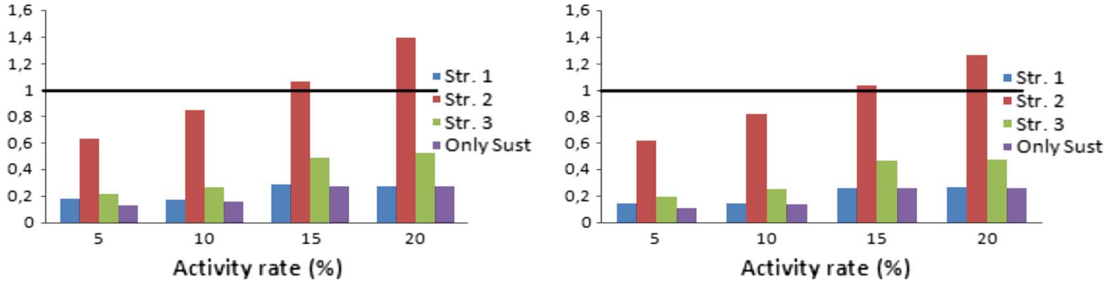


Fig. 6. CPU relative time graphics. The solid line represents the CPU time for the standard block-matching algorithm. This is compared with the CPU time consumed with each of the three strategies presented and the approach that relies only upon the sustained responses, using the Yosemite sequence. On the left, the exploration region used is of 3 × 3, and on the right, it is of 5 × 5. CPU time depends a great deal upon the ratio between the percentage of retinal responses and the resolution, i.e., upon the activity rate (several values of activity rate are explored along the x-axis).

We have implemented the following three different strategies.

1) Strategy 1 preselects dynamic local areas, i.e., local activation of transient response cells preselects areas of interest for motion estimation. The INC and DEC responses are used for the localization (definition) of regions where there is motion. In this case, we use local transient information only for finding these areas, thus providing sustained cell responses from these areas alone to the block-matching algorithm (the pseudocode is detailed in Appendix I-C). This alternative is based on a block-matching scheme that always focuses on areas where there is movement. If it is applied to a static background, there is no estimation provided. The scheme only generates motion estimation if temporal transitions exist in the processed scenario. This alternative significantly reduces the computations and helps to focus attention directly on the moving objects.

2) Strategy 2 is based on the idea of using all the retinal responses directly. In this case, the way to find the best matching region is the SSD, or the search of the region with which the current response neighborhood has the minimum mse, as described in Section II. Using transient and sustained responses, this strategy computes a new error measure to guide the exploration of the block-matching process. The new error measure, inspired by Simoncelli's work [6], [7], is defined in (4). It is not only a normalization; we define a new normalized mse to find the best matching in the region

$$E_n = \frac{S_n}{\sum_{n \in R} S_n + K} + \frac{T_n}{\sum_{n \in R} T_n + K} \tag{4}$$

where E_n is the error value for neuron response n, S_n is the mse for sustained response, T_n is the mse for transient response, R is the region or neighborhood where n belongs, and K is a constant. In addition, we use E_n to choose the right motion estimation in the matching algorithm search. The pseudocode is shown in Appendix I-D.

This strategy is based on energy models and represents a preselection of areas with higher energy for sustained and transient responses, thus normalizing their error.

3) Strategy 3 uses transient information only in specific situations. The algorithm uses the error measure defined in (4) in the exploration of the best region matching only when the decision is ambiguous, i.e., when the choice of a region is uncertain because two or more regions have a similar mse. The algorithm uses a threshold to decide whether to use mse or the new error measure (the threshold is set to 1.0). The fully detailed algorithm is shown in Appendix I-E.

The relative CPU times for the different strategies are shown in Fig. 6. The standard block-matching CPU time is defined by the line (1 in relative time) in the graphic. The graphic on the left depicts the relative time with an exploration region of 3 × 3, and on the right, it is of 5 × 5. The results support our hypothesis that our model is more efficient with lower activity rates than the standard block-matching algorithm. Strategies 1 and 3 achieve the best results, reducing CPU time by around 75% and 60%, respectively, in the worst case. On the other hand, strategy 2 needs more computational resources than the standard block matching, even with low activity rates.

In addition to this, we also studied the computational load (L) for the different strategies. These are detailed in (5)–(9), where n is the resolution in pixels for each frame; f_{sust} is the activity rate for sustained cells; f_{trans} is the activity rate for transient cells; $f_{\mathrm{matching\ ambiguity}}$ is the likelihood that the minimum mse block search finds more than a single block; and A, B, and C are the computational loads for the different processing tasks of each pixel or response (note that $A < B, C$). The activity rates in benchmark sequences are tuned to be around 10%. The operations included in A, B, and C are indicated in Appendix I

$$L_{\mathrm{standard}} = n(A) \tag{5}$$

$$L_{\mathrm{sust}} = f_{\mathrm{sust}}(n)(A) \tag{6}$$

$$L_{str1} = \left(f_{\mathrm{tran}}(n) \cap f_{\mathrm{sust}}(n)\right)(A) \tag{7}$$

$$L_{str2} = \left(f_{\mathrm{tran}}(n) \cup f_{\mathrm{sust}}(n)\right)(B) \tag{8}$$

$$L_{str3} = f_{\mathrm{sust}}(n) \times \left(C + F_{\mathrm{matching\ ambiguity}}\left(\frac{n}{block_{\mathrm{size}}} B\right)\right). \tag{9}$$

Finally, we used a cost function to compare the different bioinspired block-matching strategies. The error estimation was calculated by the angular error using the three benchmarking

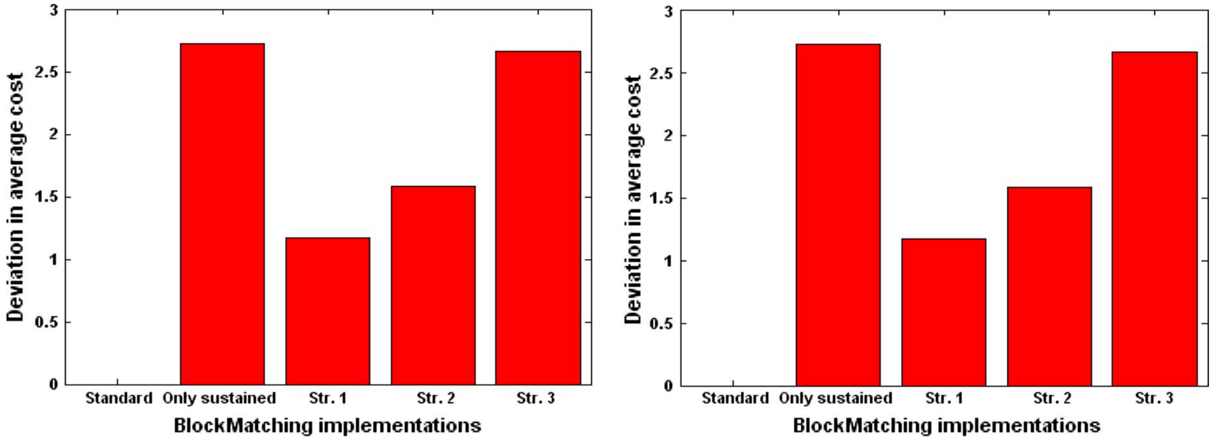


Fig. 7. (Left) Total average cost function when different scheme parameters are used for the various matching alternatives. The purpose is not to achieve an optimum solution but rather to evaluate the configuration and the role of each kind of response. (Right) Stability model with respect to parameter tuning, using the configurations shown in Table II.

TABLE II
COMBINATORY SCANNING OF PARAMETERS. THIS INVOLVED ANALYZING 480 CONFIGURATIONS FOR EACH STRATEGY AND SEQUENCE

Parameter	Values
Inner and Outer deviations (DOG)	1,2; 1,3; 1,5; 1,7; 3,5
Sustained threshold	5, 10, 20
Transient threshold	10, 15, 25
Temporal filter configuration	{-1,1}, {-1,0,1},{-1, -2, 0, 1, 2}
Exploration window	6,10,14
Block size	8, 16

sequences (comparing the obtained results with the known real ground-truth motion fields).

Angular error is not, however, a definitive evaluation estimation. It is important to emphasize that our strategies are sparse block-matching algorithms, and therefore, we need a cost function that selects the strategy with the smallest error and also with a higher number of responses (higher density).

The cost function (10) is defined as the ratio between the average angular error and the density of neurons that are active

$$Cost = \frac{Ang.Error}{Density}. \tag{10}$$

A comparison of the accuracy of the different matching strategies is shown in Fig. 7, in which five columns represent the standard block matching, the block matching using only sustained responses, and the other three strategies, which function according to the responses of the transient sensitive cells. We explored the space of the matching method parameters (modifying DoG sizes, search areas, block sizes, etc.) in order to determine the strategy with the highest accuracy and stability as far as parameter tuning was concerned. The optimization of the parameters involved a combinatory scanning, with the object being not to obtain an optimum solution but rather to evaluate the configurations and alternatives and a way of estimating the role of the different kinds of neuron. The multiparameter combinatory search scans inner and outer standard deviations for sustained responses, temporal filters for transient responses, and block-matching parameters (Table II).

Strategies 1 and 2 turned out to be best, not just on comparing average costs but also on analyzing their stability as far as different scheme parameters were concerned (Fig. 7). Strategies 3 and that consisting of just using sustained cells led to higher average cost and greater instability. On the other hand, high computational requirements were needed for strategy 2. After analyzing the results, we chose the first strategy as being a good tradeoff between accuracy, stability, and computational cost.

IV. ATTENTION MODELS

As demonstrated in Section II, it is possible to use multiscale and rank-order coding for the implementation of a system which is able to select a low percentage of sustained neuron responses to calculate optical flow fields very accurately. In that model, we did not use the transient neuron responses, but the information involved might be useful for attention models.

We define a versatile multimodal attention operator which is able to focus on different features such as motion, colors, textures, and so on. We apply rank-order coding by weighting the responses in the image that tune with a new feature, such as temporal events (transient cues), a specific color, or even a texture, and focus the system computing resources on them, even dynamically. The operator is defined in (11) and (12)

$$C_{\text{sel}} = R(\alpha \cdot S_F) \tag{11}$$

where C_{sel} is the result of the operator, i.e., the list of selected cues from rank-order coding, R is the operator, S_F is the sustained energy value for the frame F, and α is the multimodal factor defined in (11)

$$\alpha = 1 + \frac{\omega_T \cdot T_F}{\sum_{n\in F} T_n + k} + \frac{\omega_C \cdot C_F}{\sum_{n\in F} + k} + \frac{\omega_{Tx} \cdot Tx_F}{\sum_{n\in F} Tx_n + k} + \cdots \tag{12}$$

where T_F, C_F, and Tx_F are the transient, color, and texture energy values of the F frame, respectively, and ω_T, ω_C, and ω_{Tx} are the weights for each feature. The α factor is normalized by the sum of the values of the neuron responses (n) for each feature, and K is a constant (with a value of 1.1) used to avoid a null denominator. The multimodal operator (R) extracts the fixed rate of cues (in percent) from the sorted list. The

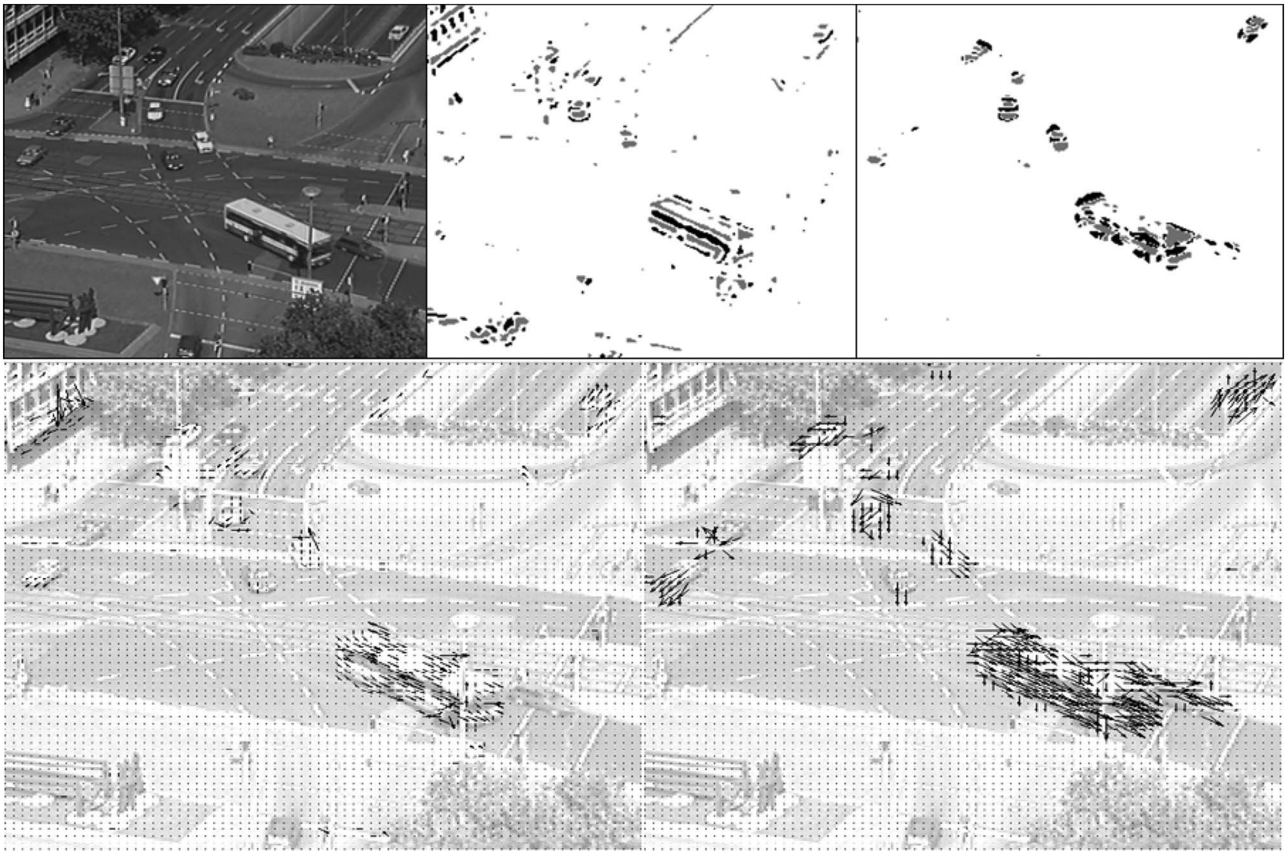


Fig. 8. Comparison between the block-matching algorithm and the block-matching algorithm using the attention operator focusing on motion over traffic sequences [21]. (From left to right) first row—original image, sustained responses and rank-order sustained responses; second row—flow field for the first algorithm and flow field for the second one overlapping the original image.

criterion for sorting depends on the sustained energy and on the multimodal factor α.

Rank-order coding can be used to reduce the number of computations, but if not carefully used, it produces undesirable effects. For instance, if a low-contrast object is moving in a high-contrast static environment, rank-order coding based on local energy rejects the moving object. This is so if the rank order only uses information from sustained neurons. Furthermore, it is important to point out that it is possible to use the multiscale approach, focusing the attention operator on each scale. For example, if we have a small object and a large object in motion, with our motion detection model, even with multiscale, it may be impossible to extract motion for the small object because the receptive fields do not tune its spatial features accurately because of its size. If we use only sustained neuron responses, we cannot improve the estimation, but if we use transient neuron responses, it is possible to extract the small object's motion by weighting its temporal features. In this way, an attention model based on motion will focus on the largest object's motion when they are both moving, but once the largest one becomes static, if the smallest one continues, its motion becomes salient.

An attention module can be implemented by defining a new energy measure, as shown in (11) and (12), where we weight the new features according to the application of our system. For example, if we need to locate objects in motion irrespective of their size, the last example would be a good solution. Nevertheless, if we need to extract the motion or to track a red object or one with a specific texture, we modify the attention operator to set higher weight upon responses that tune red objects or the specific target texture and raise them in the rank-order coding list to the highest status. As shown in Fig. 8, the attention operator focuses on motion, and the algorithm extracts more information for objects that are moving, although the contrast between them and the background is not particularly significant. Furthermore, the number of responses is similar for the algorithms; around 3% of the number of pixels and the estimations in the block-matching algorithm based only on sustained responses are the sparsest. It is assumed that the estimations for a block-matching algorithm that uses rank-order coding are the best because, where there is no motion, it estimates null velocity, and the system focuses on objects in motion, while the basic algorithm estimates different erroneous velocities in these cases. It can be seen in Fig. 8 how the system's response to moving objects is enhanced and the sparse erroneous responses to static objects in the basic model are neatly cleaned up (see the top row in Fig. 8).

An example where the attention operator is focused on the motion and on the red color is shown in Fig. 9, where a red car can be seen driving south. If we apply the standard algorithm, as the contrast with the background is not very significant, the sustained cells do not fire a high response. Therefore, the algorithm does not provide a motion estimation for the red car. On the other hand, if we use the attention operator focusing

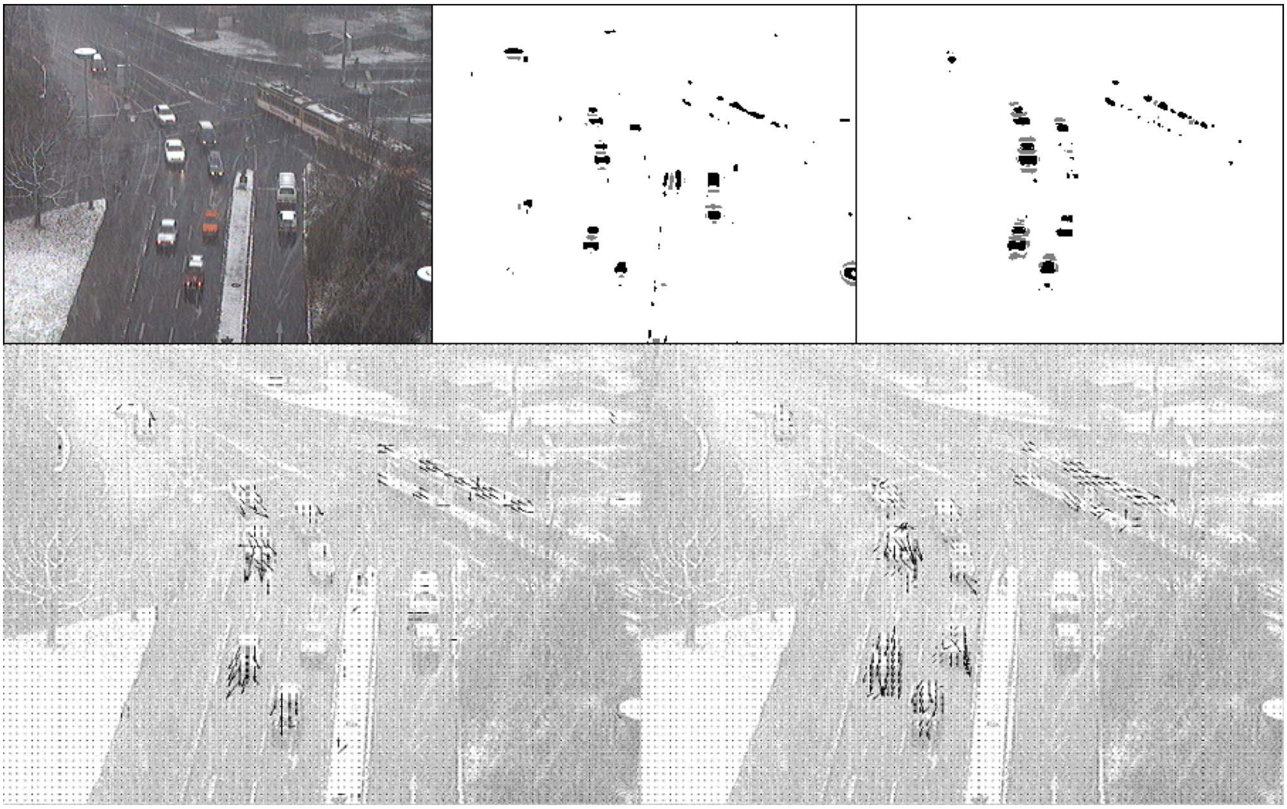


Fig. 9. Comparison between the block-matching algorithm and the block-matching algorithm using the attention operator focusing on color and motion over traffic sequences [21]. (From left to right) first row—original image, sustained responses, and rank-order sustained responses; second row—flow field for the first algorithm and flow field for the second one overlapping the original image. In this latter case, the red car (inside the dotted circle) attracts a larger number of estimations by the attention module.

on the red color, we reinforce the red car cues which emerge at the top of the rank-order coding list. In this way, we obtain the motion estimation for the red car. Furthermore, the operator is also focused on the motion, as shown in Fig. 8. Fig. 10 shows the same example, now focusing on the orange color. In Fig. 9, the focus was on the red car, but now, in Fig. 10, the orange trams provide the focus. The block-matching algorithm provides a sparse motion estimation for the tram in the upper left-hand corner, but it does not give any estimation for the other tram. Using the attention operator focused on the orange color, we achieve more responses for the two trams, and the algorithm provides a slightly denser motion estimation for both of them than does the former algorithm.

The difference between the two alternatives in each case is significant, showing more responses for the focus objects (red car in Fig. 9 and orange trams in Fig. 10) and estimating the motion.

Moreover, we can also define the multimodal attention operator dynamically. We are currently exploring schemes in which we first fix the operator to extract only objects in motion, and then, we modify the operator to extract objects of some colors and possibly objects that match a specific texture.

V. Conclusion

We have designed and implemented a bioinspired model based on artificial retinas for the detection of spatiotemporal features.

We have demonstrated that by choosing responses in an intelligent way, we have been able to improve the accuracy of our model significantly. We have shown that regions with higher local contrast lead to more accurate estimations. Furthermore, the average angular error decreases to around 32%, and the average improvement in accuracy is around 16% when preselecting the proper responses (before computing motion by a block-matching model). We have also implemented a new motion-estimation bioinspired model based on the standard block-matching algorithm integrating concepts such as multiscale and rank-order coding. The selection of the responses produces low-density saliency maps (about 11%–12% of the number of pixels in standard images for each kind of neuron), which is of interest for applications with strict bandwidth constraints.

We have designed a more stable improved block-matching algorithm which also has a lower cost by integrating transient neuron responses. We have implemented different strategies based on energy models and defined a cost function. We then selected the most stable and lowest cost strategy which preselects dynamic local areas for the matching processing. The cost-function tests show similar errors to standard block-matching algorithms and low deviations.

Finally, we define a versatile multimodal attention operator to extract other potential features by modifying the rank-order coding computation.

Our models allow features to be selected according to attention processes by using rank-order coding, and thus, we can

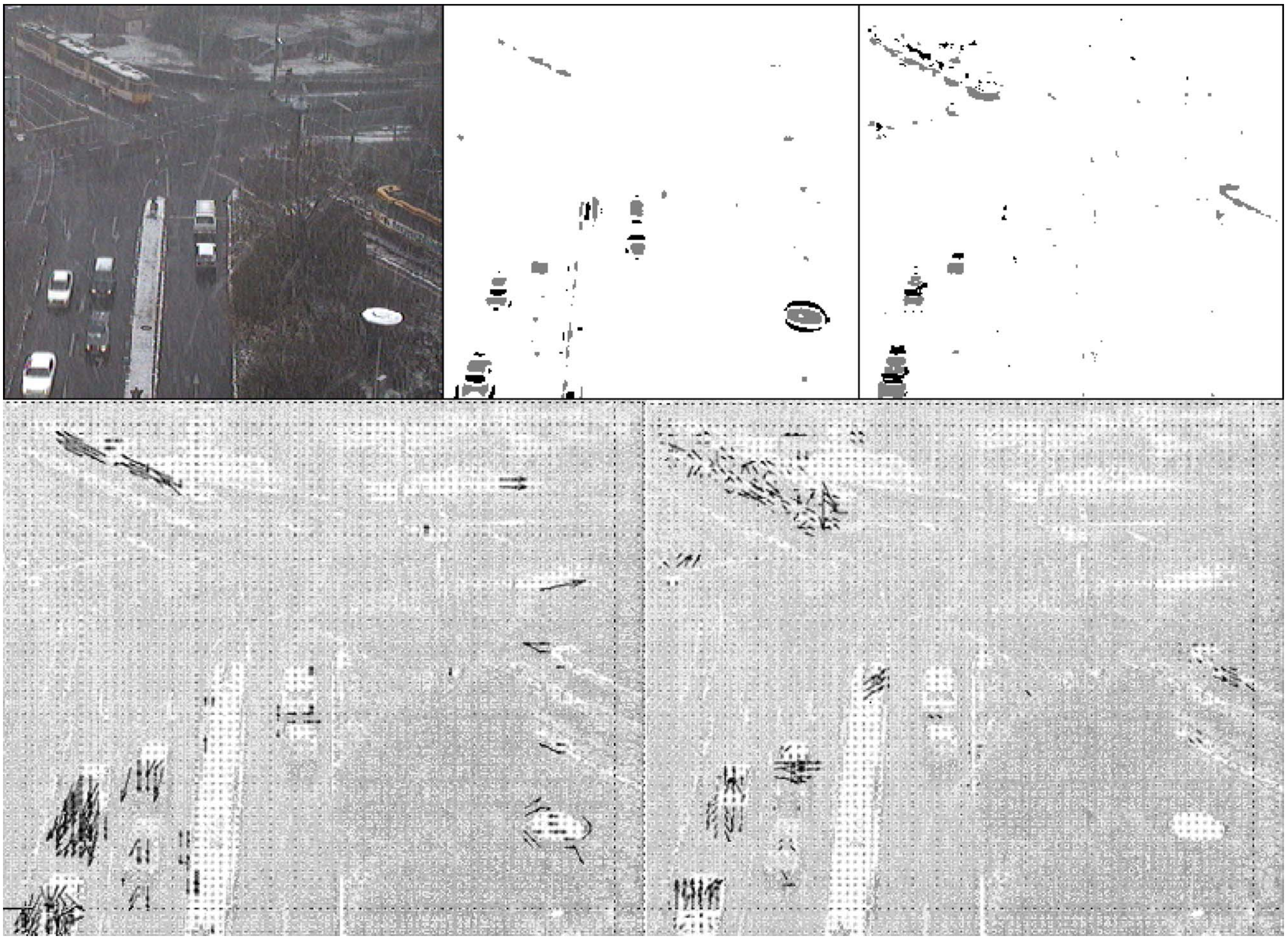


Fig. 10. Comparison between the block-matching algorithm and the block-matching algorithm using the attention operator focusing on color and motion in traffic sequences [21]. (From left to right) first row—original image, sustained responses, and rank-order sustained responses using motion and orange color to drive the attention operator; second row—flow field for the first algorithm and flow field for the second one overlapping the original image. In this latter case, the orange trams (inside the dotted ellipses) provide the focus of attention.

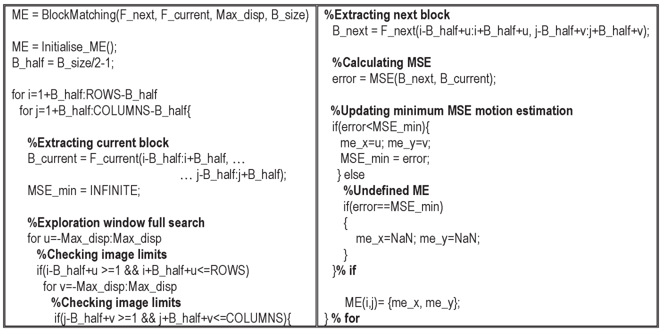
choose the image features that lead to more accurate motion estimations. This is highly relevant for efficient computation because only a few pixels of the image need to be processed. In tasks such as object tracking, for instance, we bias the rank order of the sustained neurons according to the transient neuron information by using a small number of high-confidence features.

This computing scheme is designed to be used with applications involving embedded processors, such as video surveillance [22]. For these devices, computing resources are very limited, but using the proposed method, we can produce motion estimations without losing the relevant features. We also plan to optimize the parameter search and integrate our model into a more general framework to study different vision schemes or even into real-time processing technologies.

APPENDIX I

A. Block-Matching Pseudocode

The input parameters are as follows: F_next is the next frame in the sequence, F_current is the frame for which we calculate the motion estimation (ME), Max_disp defines a square exploration window of (Max_disp · 2) × (Max_disp · 2) size, and B_size is the block size.

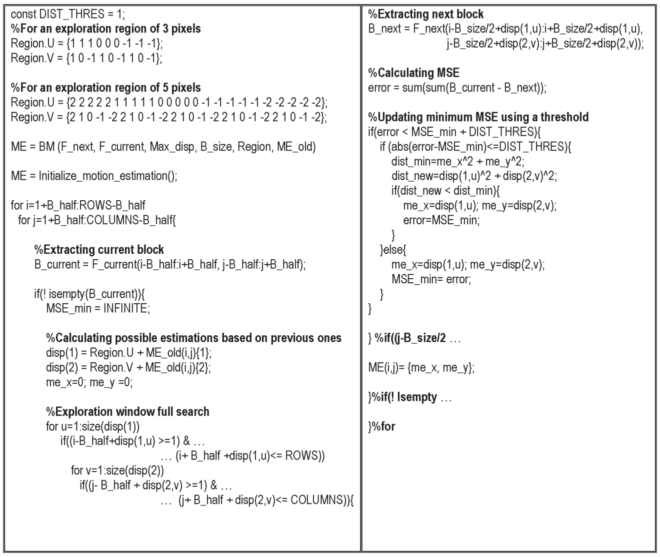


```
ME = BlockMatching(F_next, F_current, Max_disp, B_size)

ME = Initialise_ME();
B_half = B_size/2-1;

for i=1+B_half:ROWS-B_half
  for j=1+B_half:COLUMNS-B_half{

    %Extracting current block
    B_current = F_current(i-B_half:i+B_half, …
                                        … j-B_half:j+B_half);
    MSE_min = INFINITE;

    %Exploration window full search
    for u=-Max_disp:Max_disp
      %Checking image limits
      if(i-B_half+u >=1 && i+B_half+u<=ROWS)
        for v=-Max_disp:Max_disp
          %Checking image limits
           if(j-B_half+v >=1 && j+B_half+v<=COLUMNS){
%Extracting next block
  B_next = F_next(i-B_half+u:i+B_half+u, j-B_half+v:j+B_half+v);

  %Calculating MSE
  error = MSE(B_next, B_current);

 %Updating minimum MSE motion estimation
  if(error<MSE_min){
    me_x=u; me_y=v;
    MSE_min = error;
   } else
     %Undefined ME
     if(error==MSE_min)
     {
        me_x=NaN; me_y=NaN;
     }
  }% if

      ME(i,j)= {me_x, me_y};
} % for
```

B. Block-Matching Multiscale Pseudocode

The input parameters are as follows: F_next is the next frame in the sequence, F_current is the frame for which we calculate the motion estimation (ME), Max_disp defines a squared exploration window of (Max_disp · 2) × (Max_disp · 2) size, B_size is the block size, Region is a structure that defines the search region for the new motion estimation based on the previous estimation, and ME_old is the previous motion estimation oversampled.

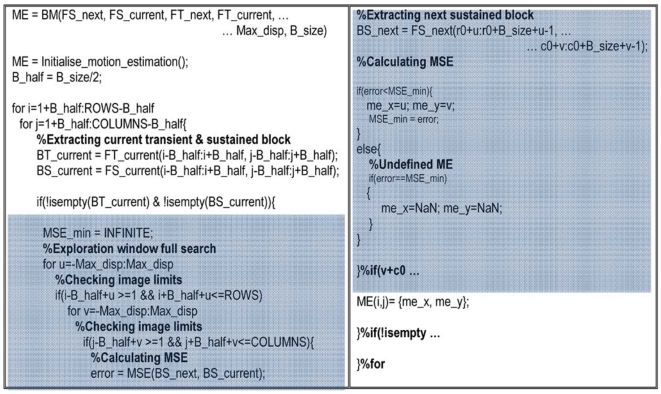
The minimum mse motion estimation is updated in two cases: The new mse is significantly lower than the previous one (the mse of the new ME is DIST_THRES lower than the previous stored mse); the new mse and the Euclidean distance to estimation $\{0, 0\}$ are lower than the previous ones. The method is based on the distance because the lowest estimations are the most reliable ones in a multiscale approach, since the firing frequency is high.



```
const DIST_THRES = 1;
%For an exploration region of 3 pixels
Region.U = {1 1 1 0 0 0 -1 -1 -1};
Region.V = {1 0 -1 1 0 -1 1 0 -1};

%For an exploration region of 5 pixels
Region.U = {2 2 2 2 2 1 1 1 1 1 0 0 0 0 0 -1 -1 -1 -1 -1 -2 -2 -2 -2 -2};
Region.V = {2 1 0 -1 -2 2 1 0 -1 -2 2 1 0 -1 -2 2 1 0 -1 -2 2 1 0 -1 -2};

ME = BM (F_next, F_current, Max_disp, B_size, Region, ME_old)

ME = Initialize_motion_estimation();

for i=1+B_half:ROWS-B_half
  for j=1+B_half:COLUMNS-B_half{

      %Extracting current block
      B_current = F_current(i-B_half:i+B_half, j-B_half:j+B_half);

      if(! isempty(B_current)){
         MSE_min = INFINITE;

         %Calculating possible estimations based on previous ones
         disp(1) = Region.U + ME_old(i,j){1};
         disp(2) = Region.V + ME_old(i,j){2};
         me_x=0; me_y =0;

         %Exploration window full search
         for u=1:size(disp(1))
             if((i-B_half+disp(1,u) >=1) & …
                                         … (i+ B_half +disp(1,u)<= ROWS))
               for v=1:size(disp(2))
                  if((j- B_half + disp(2,v) >=1) & …
                                         …  (j+ B_half + disp(2,v)<= COLUMNS)){

%Extracting next block
B_next = F_next(i-B_size/2+disp(1,u):i+B_size/2+disp(1,u),
                     j-B_size/2+disp(2,v):j+B_size/2+disp(2,v));

%Calculating MSE
error = sum(sum(B_current - B_next));

%Updating minimum MSE using a threshold
if(error < MSE_min + DIST_THRES){
   if (abs(error-MSE_min)<=DIST_THRES){
      dist_min=me_x^2 + me_y^2;
      dist_new=disp(1,u)^2 + disp(2,v)^2;
      if(dist_new < dist_min){
         me_x=disp(1,u); me_y=disp(2,v);
         error=MSE_min;
      }
   }else{
      me_x=disp(1,u); me_y=disp(2,v);
      MSE_min= error;
   }
}

} %if((j-B_size/2 …

ME(i,j)= {me_x, me_y};

}%if(! Isempty …

}%for
```

C. Block-Matching Strategy-1 Pseudocode

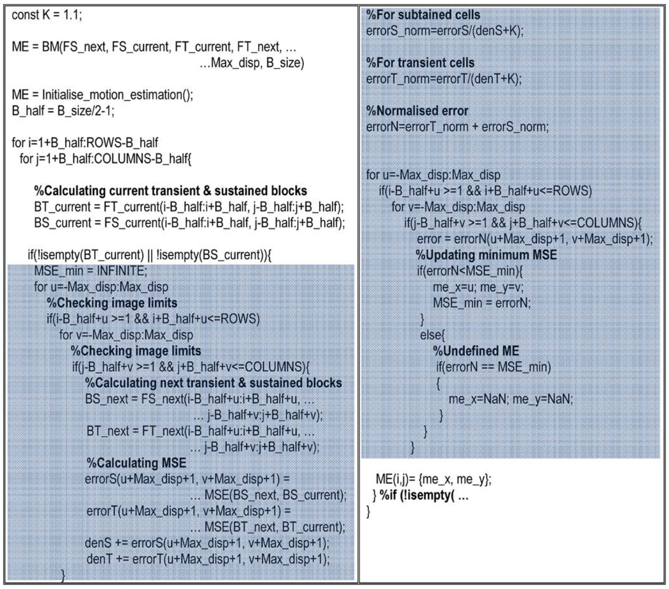
The input parameters are as follows: FS_next is the next frame in the sequence of sustained responses, FS_current is the sustained response frame for which we calculate the motion estimation (ME), FT_next is the next frame in the sequence of transient responses, FT_current is the transient response frame for which we calculate the motion estimation (ME), Max_disp defines a squared exploration window of (Max_disp · 2) × (Max_disp · 2) size, and B_size is the block size. The highlighted code represents the computational load A in (4)–(6).



```
ME = BM(FS_next, FS_current, FT_next, FT_current, ...
                                         ... Max_disp, B_size)

ME = Initialise_motion_estimation();
B_half = B_size/2;

for i=1+B_half:ROWS-B_half
  for j=1+B_half:COLUMNS-B_half{
      %Extracting current transient & sustained block
      BT_current = FT_current(i-B_half:i+B_half, j-B_half:j+B_half);
      BS_current = FS_current(i-B_half:i+B_half, j-B_half:j+B_half);

      if(!isempty(BT_current) & !isempty(BS_current)){

       MSE_min = INFINITE;
       %Exploration window full search
       for u=-Max_disp:Max_disp
          %Checking image limits
          if(i-B_half+u >=1 && i+B_half+u<=ROWS)
             for v=-Max_disp:Max_disp
               %Checking image limits
                 if(j-B_half+v >=1 && j+B_half+v<=COLUMNS){
                   %Calculating MSE
                   error = MSE(BS_next, BS_current);

%Extracting next sustained block
BS_next = FS_next(r0+u:r0+B_size+u-1, ...
                              ... c0+v:c0+B_size+v-1);
%Calculating MSE

if(error<MSE_min){
  me_x=u; me_y=v;
   MSE_min = error;
}
else{
   %Undefined ME
   if(error==MSE_min)
  {
      me_x=NaN; me_y=NaN;
   }
}

}%if(v+c0 ...
ME(i,j)= {me_x, me_y};

}%if(!isempty ...

}%for
```

D. Block-Matching Strategy-2 Pseudocode

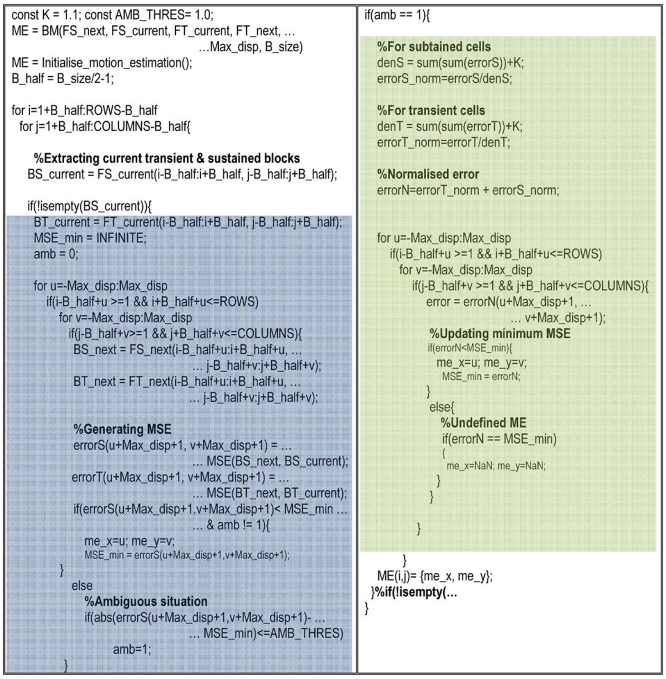
The input parameters are as follows: FS_next is the next frame in the sequence of sustained responses, FS_current is the sustained response frame for which we calculate the motion estimation (ME), FT_next is the next frame in the sequence of transient responses, FT_current is the transient response frame for which we calculate the motion estimation (ME), Max_disp defines a squared exploration window of (Max_disp · 2) × (Max_disp · 2) size, and B_size is the block size. The highlighted code represents the computational load B in (7) and (8).



```
const K = 1.1;

ME = BM(FS_next, FS_current, FT_current, FT_next, ...
                                        ...Max_disp, B_size)

ME = Initialise_motion_estimation();
B_half = B_size/2-1;

for i=1+B_half:ROWS-B_half
  for j=1+B_half:COLUMNS-B_half{

     %Calculating current transient & sustained blocks
     BT_current = FT_current(i-B_half:i+B_half, j-B_half:j+B_half);
     BS_current = FS_current(i-B_half:i+B_half, j-B_half:j+B_half);

    if(!isempty(BT_current) || !isempty(BS_current)){
     MSE_min = INFINITE;
     for u=-Max_disp:Max_disp
        %Checking image limits
        if(i-B_half+u >=1 && i+B_half+u<=ROWS)
           for v=-Max_disp:Max_disp
              %Checking image limits
              if(j-B_half+v >=1 && j+B_half+v<=COLUMNS){
                 %Calculating next transient & sustained blocks
                 BS_next = FS_next(i-B_half+u:i+B_half+u, ...
                                        ... j-B_half+v:j+B_half+v);
                 BT_next = FT_next(i-B_half+u:i+B_half+u, ...
                                        ... j-B_half+v:j+B_half+v);
                 %Calculating MSE
                 errorS(u+Max_disp+1, v+Max_disp+1) =
                                        ... MSE(BS_next, BS_current);
                 errorT(u+Max_disp+1, v+Max_disp+1) =
                                        ... MSE(BT_next, BT_current);
                 denS += errorS(u+Max_disp+1, v+Max_disp+1);
                 denT += errorT(u+Max_disp+1, v+Max_disp+1);
           }

%For subtained cells
errorS_norm=errorS/(denS+K);

%For transient cells
errorT_norm=errorT/(denT+K);

%Normalised error
errorN=errorT_norm + errorS_norm;


for u=-Max_disp:Max_disp
   if(i-B_half+u >=1 && i+B_half+u<=ROWS)
      for v=-Max_disp:Max_disp
         if(j-B_half+v >=1 && j+B_half+v<=COLUMNS){
            error = errorN(u+Max_disp+1, v+Max_disp+1);
            %Updating mínimum MSE
            if(errorN<MSE_min){
               me_x=u; me_y=v;
               MSE_min = errorN;
            }
            else{
               %Undefined ME
                if(errorN == MSE_min)
                {
                   me_x=NaN; me_y=NaN;
                }
            }
         }

   ME(i,j)= {me_x, me_y};
  } %if (!isempty( ...
}
```

E. Block-Matching Strategy-3 Pseudocode

The input parameters are as follows: FS_next is the next frame in the sequence of sustained responses, FS_current is the sustained response frame for which we calculate the motion estimation (ME), FT_next is the next frame in the sequence of transient responses, FT_current is the transient response frame for which we calculate the motion estimation (ME), Max_disp defines a squared exploration window of (Max_disp · 2) × (Max_disp · 2) size, and B_size is the block size. The highlighted code in the left column represents the computational load B in (7) and (8), and in the right column, it represents the computational load C in (8).



```
const K = 1.1; const AMB_THRES= 1.0;
ME = BM(FS_next, FS_current, FT_current, FT_next, ...
                                      ...Max_disp, B_size)
ME = Initialise_motion_estimation();
B_half = B_size/2-1;

for i=1+B_half:ROWS-B_half
  for j=1+B_half:COLUMNS-B_half{

     %Extracting current transient & sustained blocks
    BS_current = FS_current(i-B_half:i+B_half, j-B_half:j+B_half);

    if(!isempty(BS_current)){
     BT_current = FT_current(i-B_half:i+B_half, j-B_half:j+B_half);
     MSE_min = INFINITE;
     amb = 0;

     for u=-Max_disp:Max_disp
        if(i-B_half+u >=1 && i+B_half+u<=ROWS)
           for v=-Max_disp:Max_disp
             if(j-B_half+v>=1 && j+B_half+v<=COLUMNS){
              BS_next = FS_next(i-B_half+u:i+B_half+u, ...
                                      ... j-B_half+v:j+B_half+v);
              BT_next = FT_next(i-B_half+u:i+B_half+u, ...
                                      ... j-B_half+v:j+B_half+v);

              %Generating MSE
              errorS(u+Max_disp+1, v+Max_disp+1) = ...
                                      ... MSE(BS_next, BS_current);
             errorT(u+Max_disp+1, v+Max_disp+1) = ...
                                      ... MSE(BT_next, BT_current);
              if(errorS(u+Max_disp+1,v+Max_disp+1)< MSE_min ...
                                      ... & amb != 1){
                me_x=u; me_y=v;
                MSE_min = errorS(u+Max_disp+1,v+Max_disp+1);
           }
             else
                %Ambiguous situation
                if(abs(errorS(u+Max_disp+1,v+Max_disp+1)- ...
                                      ... MSE_min)<=AMB_THRES)
                      amb=1;
           }

if(amb == 1){

   %For subtained cells
   denS = sum(sum(errorS))+K;
   errorS_norm=errorS/denS;

   %For transient cells
   denT = sum(sum(errorT))+K;
   errorT_norm=errorT/denT;

   %Normalised error
   errorN=errorT_norm + errorS_norm;


   for u=-Max_disp:Max_disp
      if(i-B_half+u >=1 && i+B_half+u<=ROWS)
        for v=-Max_disp:Max_disp
           if(j-B_half+v >=1 && j+B_half+v<=COLUMNS){
              error = errorN(u+Max_disp+1, ...
                                     ... v+Max_disp+1);
              %Updating mínimum MSE
              if(errorN<MSE_min){
                me_x=u; me_y=v;
                 MSE_min = errorN;
              }
              else{
                 %Undefined ME
                 if(errorN == MSE_min)
                 {
                  me_x=NaN; me_y=NaN;
                 }
              }

           }

       }
   ME(i,j)= {me_x, me_y};
 }%if(!isempty(...
}
```

ACKNOWLEDGMENT

The authors would like to thank A. L. Tate for improving the language of this paper.

REFERENCES

[1] J. T. Martin, *An Introduction to the Visual System*. Cambridge, U.K.: Cambridge Univ. Press, 1996.

[2] K. Nakayama, "Biological image motion processing: A review," *Vis. Res.*, vol. 25, no. 5, pp. 625–660, Nov. 1985.

[3] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interactions and functional architecture in the cat's visual cortex," *J. Physiol.*, vol. 160, pp. 106–154, 1962.

[4] E. H. Adelson and J. R. Bergen, "The extraction of spatio-temporal energy in human and machine vision," in *Proc. IEEE Workshop Motion: Representation Anal.*, 1986, pp. 151–155.

[5] D. J. Heeger, "Model for the extraction of image flow," *J. Opt. Soc. Amer. A, Opt. Image Sci.*, vol. 4, no. 8, pp. 1455–1471, Aug. 1987.

[6] E. P. Simoncelli and D. J. Heeger, "A model of neuronal responses in visual area MT," *Vis. Res.*, vol. 38, no. 5, pp. 743–761, Mar. 1998.

[7] E. P. Simoncelli, "Distributed analysis and representation of visual motion," Ph.D. dissertation, MIT, Cambridge, MA, 1993.

[8] S. Mota, E. Ros, J. Díaz, E. M. Ortigosa, and A. Prieto, "Motion-driven segmentation by competitive neural processing," *Neural Process. Lett.*, vol. 22, no. 2, pp. 125–147, Oct. 2005.

[9] E. H. Adelson and P. J. Burt, "The Laplacian pyramid as a compact image code," *IEEE Trans. Commun.*, vol. COM-31, no. 4, pp. 532–540, Apr. 1983.

[10] E. H. Adelson, C. H. Anderson, J. R. Bergen, P. J. Burt, and J. M. Ogden, "Pyramid methods in image processing," *RCA Eng.*, vol. 29, no. 6, pp. 33–41, Nov./Dec. 1984.

[11] V. Willert, J. Eggert, J. Adamy, and E. Körner, "Non-Gaussian velocity distributions integrated over space, time, and scales," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 36, no. 3, pp. 482–493, Jun. 2006.

[12] L. Perrinet, M. Samuelides, and S. Thorpe, "Coding static natural images using spiking event times: Do neurons cooperate?"*IEEE Trans. Neural Netw.*, vol. 15, no. 5, pp. 1164–1175, Sep. 2004.

[13] K. Boahen, "A retinomorphic chip with parallel pathways: Encoding ON, OFF, INCREASING, and DECREASING visual signals," *Analog Integr. Circuits Signal Process.*, vol. 30, no. 2, pp. 121–135, 2002.

[14] K. A. Zaghloul and K. A. Boahen, "Optic nerve signals in a neuromorphic chip I: Outer and inner retina models," *IEEE Trans. Biomed. Eng.*, vol. 51, no. 4, pp. 657–666, Apr. 2004.

[15] E. Culurciello, R. Etienne-Cummings, and K. Boahen, "A biomorphic digital image sensor," *IEEE J. Solid-State Circuits*, vol. 38, no. 2, pp. 281–294, Feb. 2003.

[16] S. Granados, R. Rodríguez, E. Ros, and J. Díaz, "Visual processing platform based on artificial retinas," in *Proc. IWANN*, 2007, pp. 506–513.

[17] K. A. Zaghloul and K. Boahen, "Optic nerve signals in a neuromorphic chip II: Testing and results," *IEEE Trans. Biomed. Eng.*, vol. 51, no. 4, pp. 667–675, Apr. 2004.

[18] *Optical Flow Synthetic Sequences With Ground-Truth Information*. [Online]. Available: ftp://ftp.vislist.com/SHAREWARE/CODE/OPTICAL-FLOW

[19] J. L. Barron, D. J. Fleet, and S. Beauchemin, "Performance of optical flow techniques," *Int. J. Comput. Vis.*, vol. 12, no. 1, pp. 43–77, Feb. 1994.

[20] E. Trucco, T. Tommasini, and V. Roberto, "Near-recursive optical flow from weighted image differences," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 35, no. 1, pp. 124–129, Feb. 2005.

[21] H. Nagel, *Institut für Algorithmen und Kognitive Systeme: Ettlinger-Tor*. [Online]. Available: ftp://ftp.ira.uka.de/pub/vid-text/image_sequences/dt/sequence.mpg

[22] L. Snidaro, R. Niu, G. L. Foresti, and P. K. Varshney, "Quality-based fusion of multiple video sensors for video surveillance," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 37, no. 4, pp. 1044–1051, Aug. 2007.

Francisco Barranco received the M.S. degree in computer science from the University of Granada, Granada, Spain, in 2007.

He is with the Department of Computer Architecture and Technology, University of Granada. He is currently participating in an EU project related to adaptive learning mechanisms and conventional control. His main research interests include image processing architectures and embedded systems based on reconfigurable devices, real-time machine vision, general-purpose graphical programming devices, biologically processing schemes, and spiking neurons.

Javier Díaz received the M.S. degree in electronics engineering and the Ph.D. degree in electronics from the University of Granada, Granada, Spain, in 2002 and 2006, respectively.

He is currently an Assistant Professor with the Department of Computer Architecture and Technology, University of Granada. His main research interests include cognitive vision systems, high-performance image processing architectures, and embedded systems based on reconfigurable devices. He is also interested in spiking neurons, biomedical devices, and robotics.

Eduardo Ros received the Ph.D. degree from the University of Granada, Granada, Spain, in 1997.

He is currently an Associate Professor with the Department of Computer Architecture and Technology, University of Granada, where he is currently a Researcher for two European projects related to bioinspired processing schemes. His research interests include simulation of biologically plausible processing schemes, hardware implementation of digital circuits for real-time processing in embedded systems, and high-performance computer vision.

Begoña del Pino received the Ph.D. degree in computer science from the University of Granada, Granada, Spain, in 1999.

She is currently an Associate Professor with the Department of Architecture and Computer Technology, University of Granada. She has participated in different EU projects in the fields of visual rehabilitation, bioinspired processing schemes based on spiking neural networks, and real-time computer vision. Her main research interests include implementation of systems for visual rehabilitation, bioinspired processing systems, and reconfigurable hardware codesign for computer vision on-chip.

2 On-chip low-level processing architectures

2.1 Hierarchical gradient-based optical flow estimation

The journal paper associated to this part of the dissertation is:

- F. Barranco, M. Tomasi, J. Díaz, M. Vanegas, E. Ros, Parallel architecture for hierarchical optical flow estimation based on FPGA. IEEE Trans. on VLSI, vol. 20, no. 6, pp. 1058-1067, 2012. DOI 10.1109/TVLSI.2011.2145423.
  - Status: **Published**.
  - Impact Factor (JCR 2010): 0.907
  - Subject Category:
    - Computer Science, Hardware and architecture. Ranking 27 / 48.
    - Engineering, electrical and electronic. Ranking 135 / 247.

Parallel Architecture for Hierarchical Optical Flow Estimation Based on FPGA

Francisco Barranco, Matteo Tomasi, Javier Diaz, Mauricio Vanegas, and Eduardo Ros

Abstract—**The proposed work presents a highly parallel architecture for motion estimation. Our system implements the well-known Lucas and Kanade algorithm with the multi-scale extension for the computation of large motion estimations in a dedicated device [field-programmable gate array (FPGA)]. Our system achieves 270 frames per second for a 640×480 resolution in the best case of the mono-scale implementation and 32 frames per second for the multi-scale one, fulfilling the requirements for a real-time system. We describe the system architecture, address the evaluation of the accuracy with well-known benchmark sequences (including a comparative study), and show the main hardware resources used.**

Index Terms—**Field-programmable gate arrays (FPGAs), machine vision, real time systems, reconfigurable architectures.**

I. Introduction

THE optical flow computation is the challenging task of estimating bidimensional motion fields projected by the scene on the image plane of a camera from a sequence of captured frames. For the computation of the optical flow, we assume the constancy of the intensity for each pixel in the image at one instant for the successive frame, which is fulfilled with a good approximation for the object with smooth and small displacements.

Optical flow is a low-level vision feature widely used for many applications, as for instance, for the computation of middle-level applications such as motion in depth [1], structure from motion [2], independently moving objects (IMOs), [3], or heading [4]. Furthermore, its potential applications encompass video stabilization [5], object tracking [6], video denoising and restoration [7], segmentation [8], or active vision [9], [10]. All of them are useful in a wide range of fields such as autonomous robot navigation [11], video surveillance [12], or driving assistance systems [13], [14].

In our work, we implement the Lucas and Kanade (L&K) algorithm [15], particularly the model described in [16], [17]. We include an extension to the original algorithm, a hierarchical architecture, capable of working with an extended motion range much larger than the standard mono-scale approaches (with a typical range of a few pixels [13]). This hierarchical architecture has been previously extensively detailed [18] and many works deal with the multi-scale implementation [19]–[22]. It basically consists in the construction of a pyramid of images with the image resolution being reduced one octave at each pyramid level. Then, the motion estimation stage performs the search over a small number of pixels at the coarsest scale. The obtained estimation is used as a seed to search locally at the next finer scale and so on to the finest one. This process is extensively detailed on Section II-B and significantly allows us to increase the motion range of the algorithm, which is critical for many real-world applications.

Nowadays, optical flow estimation is the main topic of a large number of works in the literature and its real-time computation is one of the main related challenges. We can find works in the literature addressing this topic using dedicated hardware architectures [23], graphic processing units (GPUs) as accelerators [24], and even optimized software implementations [25], or clusters of processors [26]. Finally, in previous works [13], [27], we can also find field-programmable gate array (FPGA)-based solutions.

In this paper, we have selected FPGA devices because local optical flow algorithms are good candidates due to their potential for a high-performance massive parallelization. The results are obtained using a fine-pipeline based architecture designed in a modular way. Our objective is to achieve a throughput of one pixel per clock cycle along the whole processing scheme. The correct exploitation of these high performance characteristics allows us to achieve, in the case of the mono-scale approach, up to 270 fps (frames per second) for an image resolution of 640 × 480 thanks to the presented specific-purpose architecture. In the case of the multi-scale implementation, the frame rate reaches almost 32 fps for the same image resolution with a motion working range 30 times larger than the mono-scale approach. The error results are shown in Section III.

This paper is structured as follows. In Section II, we summarize the L&K algorithm and the multi-scale extension. Section III details the hardware implementation and presents the results comparing the proposed alternatives and the hardware resource utilization. Section IV shows the architecture of the final system and its benchmarking. Finally, Section V presents the conclusions of the paper.

Manuscript received August 05, 2010; revised February 02, 2011 and March 30, 2011; accepted April 13, 2011. Date of publication May 23, 2011; date of current version May 05, 2012. This work was supported in part by the Spanish Project DINAM-VISION (DPI2007-61683), by the Spanish Grant (AP2007-00275), and by the regional projects (P06-TIC02007, TIC-3873).

F. Barranco, M. Tomasi, J. Diaz, and E. Ros are with the Department of Computer Architecture and Technology, CITIC, ETSIIT, University of Granada, E18071 Granada, Spain (e-mail: fbarranco@atc.ugr.es; mtomasi@atc.ugr.es; jdiaz@atc.ugr.es; eduardo@atc.ugr.es).

M. Vanegas is with the Department of Computer Architecture and Technology, CITIC, ETSIIT, University of Granada, E18071 Granada, Spain, and also with the PSPC Group, Department of Biophysical and Electronic Engineering (DIBE), University of Genoa, I-16145 Genoa, Italy (e-mail: mvanegas@atc.ugr.es).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2011.2145423

II. L&K Algorithm for Optical Flow Computation

The optical flow estimation consists in the computation of the motion field that represents the pixel displacements between successive frames, which is defined for each pixel as a velocity vector (u, v) as in (1). We compute the optical flow estimation assuming the brightness constancy, i.e., the optical flow constraint (OFC) (1). In (1), we assume that the intensity of the pixel in the (x, y) position in the image at time t, and the pixel $(x{+}u, y{+}v)$ in the image of time $t{+}1$ does not change. As commented before, this is only a valid assumption for slow-moving objects between frames (it depends on the distance from the object to the camera, on the 3-D object velocity, and on the camera frame-rate)

$$I(x, y, t) = I(x + u, y + v, t + 1). \tag{1}$$

This equation is solved by linearization using the first-order Taylor expansion and this leads to the subsequent linear one (2)

$$I_x u + I_y v + I_t = 0 \tag{2}$$

where subscripts represent the partial derivatives in each direction (x, y, and t). Due to the fact that from (2), we cannot determine a unique solution, we need another constraint to find the solution of the optical flow estimation (only velocity vectors which are normal to the image structure could be computed with this equation). There are different approaches to solve this problem. Many local methods assume that the flow is constant in the same local neighborhood. Thus, we compute the motion estimation for a pixel focusing only on its neighborhood and ignoring the rest and it is supported because close pixels are very likely to correspond to the same objects and, therefore, similar velocity vector values can be expected. This approach was firstly proposed by L&K [15]. In addition, other local approaches impose that not only luminance values remain constant but also image derivatives (see for instance second or higher order gradient methods [16]). A different approach consists in solving an optimization problem minimizing a global function depending on the OFC (1) and other constraints. This is the approach originally proposed by Horn & Schunk [28].

The L&K approach is one of the most accurate, computationally efficient, and widely used methods for the optical flow computation. It is a local method that belongs to the gradient based ones, because its computation is based on the image derivatives [16], [17] and because it solves the previous equation by assuming that the flow is constant in the same local neighborhood. This is the model which we have selected for our implementation. To solve the previous OFC (2), we estimate the optical flow as the value which minimizes the energy function building an over-constrained equation system as in (3)

$$E(u, v) = \frac{1}{2} \sum_{i \in \Omega} \left(W_i^2 (I_x u + I_y v + It)^2 \right) \tag{3}$$

where W_i stands for the weight matrix of the pixels in neighborhood ω. And then, for the resolution of the system of equations, we use a least squares-fitting procedure as shown in (3)

$$(u, v) = \left(A^T W^2 A\right)^{-1} A^T W^2 b. \tag{4}$$

From (4), we solve the equation, obtaining a 2-by-2 linear system defined by (6) and (5)

$$(A^T W^2 b) = \begin{bmatrix} \sum\limits_{i \in \Omega} W_i^2 I_{xi} I_{ti} \\ \sum\limits_{i \in \Omega} W_i^2 I_{yi} I_{ti} \end{bmatrix} \tag{5}$$

$$(A^T W^2 A) = \begin{bmatrix} \sum\limits_{i \in \Omega} W_i^2 I_{xi}^2 & \sum\limits_{i \in \Omega} W_i^2 I_{xi} I_{yi} \\ \sum_{i \in \Omega} W_i^2 I_{xi} I_{yi} & \sum\limits_{i \in \Omega} W_i^2 I_{yi}^2 \end{bmatrix}. \tag{6}$$

Barron computes the confidence of the estimation using the minimum of the eigenvalues of Matrix (5). We compute the reliable values with the determinant of (5) to simplify the hardware implementation (in this case, it corresponds to the eigenvalues product) and the loss of accuracy is not significant [13], [29].

The previous analysis requires that only a small displacement is presented in the scene in order for the first order Taylor expansion approximation to be valid. Therefore, the main disadvantage of the L&K algorithm is the accuracy for the estimations of large displacements, which is not possible with the previous schemes (in fact, there are some few exceptions [13] but they are not practical for real-world scenarios). For the resolution of the problem, we can adopt one of the following strategies: 1) increasing the frame rate of the sequence to decrease the motion range (it depends on the capture device or the available sequences and is not always possible) or 2) implementing a hierarchical approach. The second approach consists in a multi-scale implementation that computes the optical flow velocity components for each different resolution input simulating the use of filters of different sizes for a better tuning of the different range displacements.

A. *Multi-Scale Implementation*

In this paper, the problem of the limited motion working range with the L&K algorithm is solved with a coarse-to-fine multi-scale implementation based on the hierarchical model proposed by [18].

The first stage is the image pyramid computation of the input frames, using a variable number of scales which mainly depends on the size of the sequence and on the range of the displacements. Then, the computation is performed in the coarse-to-fine scheme where motion computed at coarse scales is used as a seed that is refined at the next pyramid levels as described in the next paragraphs.

After the pyramid image construction, the next stage is the L&K motion estimation. A pair of estimations is obtained for the x and y velocity components (u, v) for the current scale. At this point of the algorithm, the scale is the coarsest one.

The next step is the over-sampling of the velocity estimation to the resolution of the subsequent scale and which is also multiplied by 2 (the sub-sampling factor in order to increase one octave).

Then, the warping process is performed. This is a complex operation in terms of hardware and it consumes a significant amount of computational resources. It consists in warping the input frames with the estimations calculated in the previous steps to move each pixel to the previously estimated position. In such a way, we address a "motion compensation", reduce the

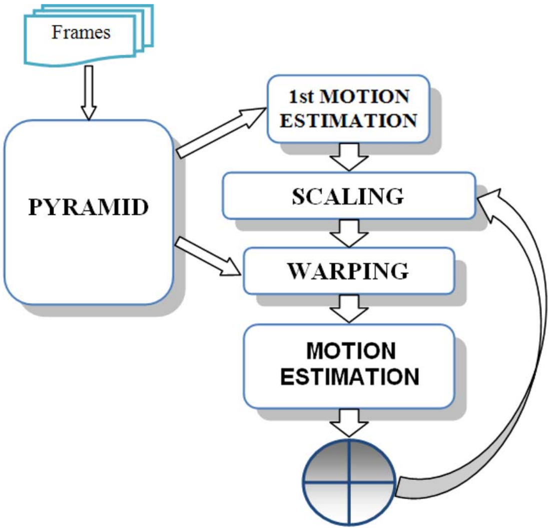


Fig. 1. Scheme of the multi-scale optical flow algorithm. Main stages: over-sampling from the previous motion estimation, warping with the new frames, new estimation computation, and merging between the two computed estimations. This is carried out by iterating from the coarsest to the finest scale.

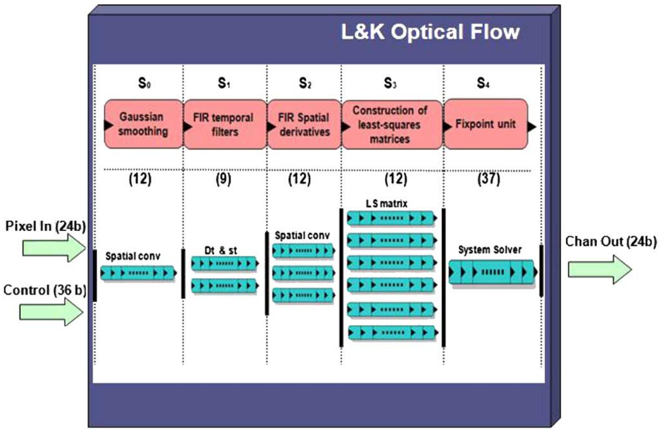


Fig. 2. Scheme of the pipelined stages for the optical flow core. It indicates the computation stages (from 0 to 4): the Gaussian smoothing to reduce aliasing effects; the computation of the spatio-temporal smoothing (St) and temporal derivative (Dt); partial derivatives (x, y, and t); the construction of the matrix for the least-squares resolution and the system solver. The numbers in parenthesis at each stage indicate the number of micropipelined stages and the parallel datapaths show the superscalar architecture level for fulfilling the high performance requirements.

range of the highest displacements, and keep local motion in the filter tuning range. The number of frames and the order of each one decide the warping of their pixels. In our case, we use three frames and thus, the pixels of the first and the last frames are warped (with different signs) to reduce the range of displacements with respect to the second (central) frame. Furthermore, for this sub-pixel warping, we implement a 2-by-2 bilinear interpolation to produce smoothed images, which is required for the accurate computation of the image derivatives.

The new warped frames are the input frames for the optical flow computation in the next stage, achieving a new velocity estimation (u, v).

The next stage performs the merging of the first optical flow estimation and the last one.

At this point, the algorithm iterates from the over-sampling of the velocity estimation to the merging stage. The number of iterations is defined by the number of scales set with the input parameters. The algorithm is described by the flow diagram in Fig. 1.

III. Hardware Implementation

The hardware implementation is performed in an FPGA, in our case, a Xilinx Virtex4 XC4vfx100 chip. The board, a Xirca V4 [30], provides a PCI express interface and four 8 MB SRAM ZBT memory banks and is able to work as a co-processing or standalone platform.

The hardware implementation has been performed using two different abstraction levels. The RTL language VHDL has been used for the memory controller unit (MCU), the off-chip memory, and the PCI interfaces. We used the high level Handel-C language for the implementation of the optical flow core and the multi-scale extension. This language is much more suitable for the algorithmic descriptions with a low degradation in terms of performance or resource utilization [31].

A. *L&K Optical Flow Core*

The implementation of the L&K core is based on a previous approach described in [13], [32]. The most significant changes are the migration to the new platform, the connection with the new memory interface with the MCU, and the multi-scale implementation with warping. The multi-scale extension is a critical difference which leads to an expansion of the system working range (in terms of motion computation) at the cost of significant computing resources. This multi-scale extension is usually avoided in hardware approaches due to its architectural complexity in the framework of a massively parallel datapath. Furthermore, in this paper, instead of presenting a single implementation, we describe different alternatives at the design stage that lead to different performances versus hardware cost tradeoffs.

The input of the core consists of two channels, as is shown in Fig. 2: 1) *Pixel In* is the channel of the input frame pixels (3 frames × 8 bits/frame) and 2) *Control* is the channel for the parameters of the core (the number of scales, the image resolution, and the confidence thresholds). As an output, the system computes the optical flow estimation, i.e., the component velocities for the x and the y directions (12 bits per component).

The core computation is divided in five stages.

- S_0: The stage consists in the filtering of the input frames. The filtering uses a five tap Gaussian smoothing kernel. It reduces the aliasing effects and increases the accuracy of image derivatives.
- S_1: At this point, the algorithm computes the temporal derivative (Dt) and the spatio-temporal smoothing (St) of the three frames. This computation is carried out using a derivative and a three tap smoothing filter.
- S_2: It computes the spatial derivatives from the latter results: the It partial derivative is obtained by applying a Gaussian filtering to the temporal derivative; the I_x and I_y partial derivatives are achieved by differentiating the St

term with a derivative filter in the respective direction following the Simoncelli complementary derivative kernels approach [33].

- S_3: This stage computes the coefficients of the linear system of (6) and (5). The weights W_i for the neighborhood are set by the 5-by-5 separable kernel used in [13], [17], [29]: $W = [1\ 4\ 6\ 4\ 1]/16$. It computes the $W_i^2 I_{ti}^2$ coefficient used as a threshold for the temporal noise too. This term is very useful for handling real-world sequences by helping to reduce the noise effects.
- S_4: The last stage calculates the resolution of the 2-by-2 system and uses the determinant of the resultant matrix to threshold the less confident results.

For this system, we use 199 parallel processing units: stage S_0 has three paths (one for each frame) for the Gaussian filtering, S_1 has two paths (for the temporal derivative and the temporal smoothing), S_2 has three paths (one for each derivative I_x, I_y, I_t), S_3 has 6 paths (one for each coefficient of (6) and (5)) and finally, S_4 has only one path. The number in brackets in each stage represents the micropipelined stages for each of them.

Our scheme implements four alternatives for the system solver. Two alternatives use the fix-point arithmetic and the other two, the floating-point arithmetic. They are listed as follows.

- *Low resources and fixed-point arithmetic (LRF)*: Implementation using fixed-point arithmetic with the lowest hardware utilization for the solver of the equation system (using as divider an IP core from the Xilinx CoreGenerator [34]).
- *High resources and fixed-point arithmetic (HRF)*: Implementation using a fixed-point arithmetic with the highest hardware utilization for the solver of the equation system (also using the IP divider)
- *Floating-point arithmetic (FLO)*: Using floating-point arithmetic and our customized and highly pipelined division circuit. This alternative is the most similar one to the alternative developed in [13].
- *Floating-point arithmetic and use of a core for the division (FCD)*: Implementation with a floating-point arithmetic and use of an IP core from the Xilinx CoreGenerator as float divider.

Table I shows the bit-width representation for the listed alternatives. In the previous work [13], the author carried out an extensive batch of simulations for the bit-width decision. Moreover, that work focused on a target implementation constrained by the hardware resources, the memory interface, and the bit-width of the embedded resources in that board. Now, our representations include more bits for the most constrained stages to improve the accuracy and the density. This is important because in the framework of multi-scale approach, the accuracy of the estimations at the coarsest scales needs to be high, because they drive the estimations of the finest ones. It means that we need a good accuracy at the coarsest scales in order to keep a high accuracy along the multi-scale approach as images are warped according to the velocity results of the coarser scales.

TABLE I
BIT-WIDTH REPRESENTATION FOR THE PROPOSED ALTERNATIVES DEPENDING ON THE STAGE[a]

	S_0	S_1	S_2	S_3	S_4
LRF	[9 0]	[9 0]	[8 1]	[14 4]	[29 8]
HRF	[9 0]	[9 1]	[9 1]	[16 4]	[33 8]
FLO	[9 0]	[9 0]	[8 1]	[14 4]	[11 7][b]
FCD	[9 0]	[9 0]	[8 1]	[14 4]	[11 7][b]

[a] Integer representation: [integer part fractional part]; floating-point representation [mantissa exponent].
[b] Stage with floating point representation.

TABLE II
HARDWARE RESOURCE UTILIZATION FOR THE PRESENTED ALTERNATIVES USING A XILINX VIRTEX-4 FX100 FPGA

	4 input LUTs	Slice Flip-Flops	Slices	Freq (MHz)
LRF	5039 (5%)	6622 (7%)	4224 (10%)	83
HRF	5399 (6%)	6851 (8%)	4803 (11%)	67
FLO	8865 (10%)	4715 (5%)	6551 (15%)	76
FCD	8203 (9%)	4838 (5%)	5916 (14%)	73

TABLE III
ACHIEVED FRAME RATE FOR THE MONO-SCALE ARCHITECTURE AT THE 640 × 480 RESOLUTION

	LRF	HRF	FLO	FCD
Frame rate (fps)	270	218	247	238

In Table I, the pair of values represents the number of bits of the integer and the fractional parts respectively. In the case of the use of the floating-point arithmetic (only for S_4), they represent the number of bits for the mantissa and the exponent. The *HRF* alternative presents a different bit-width configuration: this is the alternative with the highest resource use and fixed-point. The bit-width is incremented in S_1 and S_2 to exploit the full precision of the convolution operations without losing accuracy for the use of fixed-point arithmetic. In S_3, this effect is even more considerable reaching a word size of 20 bits to build the coefficients of the linear system and finally, the bit-width is incremented to 42 bits in the last stage, which performs the division, which is the stage with the highest loss of precision.

The resources used for the proposed alternatives are shown in Table II. All the results are obtained using the synthesis Xilinx ISE tools [34]. The choice with the lowest resource cost is the first one, which uses only 10% and has also the highest clock frequency (83 MHz). It is important to remark that the last alternative (which uses a floating divider core) saves resources with respect to the previous one but the frequency is 73 MHz. Although we have to add an interface to communicate the new divider core with the rest of the system and to parse the input and output parameters for the new format, the resources are lower than in the case of our pipelined division circuit (*FLO* choice).

Table III shows the achieved frame rate using the new proposed architectures (mono-scale), reaching the fastest one (*LRF*) about 270 fps.

On the other hand, the analysis of accuracy is also very significant for the optical flow estimation. Using the "*Marble*" sequence benchmark (available at [35]), we compute the average angular error (AAE, described in [17]), the standard deviation (SAE of the AAE), and the density (percentage of valid values).

TABLE IV
COMPARISON OF THE SOFTWARE AND HARDWARE ERROR MEASURES FOR THE "*MARBLE*" SEQUENCE FOR THE MONO-SCALE IMPLEMENTATION

	AAE (°)	SAE (°)	Dens (%)
Software	29.14	24.31	80.90
LRF	22.84	21.47	59.71
HRF	22.84	21.48	59.70
FLO	25.98	24.77	68.17
FCD	26.03	24.86	68.15

The five rows of Table IV represent the results for the software model and for the four alternatives explained in this section.

We obtain the best results with the floating-point approaches because of the higher density but with 3° more of AAE than the fixed-point one. It is important to remark that using the standard "*Marble*" sequence as benchmark prevents us to take full advantage of the high processing power (in terms of frames per second) to improve accuracy. This is due to its smaller inter-frame displacements or slower movements at high frame rates when the sequence is captured with high frame-rate cameras. Therefore, it should be noted that the errors of these mono-scale alternatives can be expected to be very low when using appropriate high frame rate cameras (better fitting the processing power of the mono-scale optical flow engine).

It is well-known in computer vision that the selection of the best alternative is executed depending on the target application, as a good trade-off between the required accuracy and the constraints about the maximum frequency and the resource utilization. In our case, we will present two systems for the multi-scale computation using the LRF and the FLO alternatives, because they have yielded up as the less expensive ones in terms of hardware resources with an affordable (very low) loss of accuracy.

B. Multi-Scale Architecture

The components of the complete system are (see Fig. 3): the L&K optical flow core, the scaling module, the warping module, the merging module, and the median filter circuits, and they all work in parallel. For the smallest scale (the first iteration of the algorithm), the merging circuit is omitted and the optical flow block works with the pyramid output.

All the other blocks in the processing stages interact with memory through the MCU that multiplexes in time the huge amount of data which they have to read/store [36].

The data flow is displayed in Fig. 3 and can be summarized as follows.

- The Scaling circuit reads old partial optical flow values and scales them with a bilinear interpolation (the new values are multiplied by 2 to adapt the optical flow values to the next scale).
- The Warping circuit reads the pyramid images and displaces them using the expanded motion.
- The Median filtering stage removes the outliers homogenizing the partial and the final results. This stage consists in a median bidimensional filter and can be a cascade of them. This filter also contributes by incrementing the density of the final results removing the non-confident values and filling the holes with the filter results (in our case, the filter computes on a 3 × 3 neighborhood).

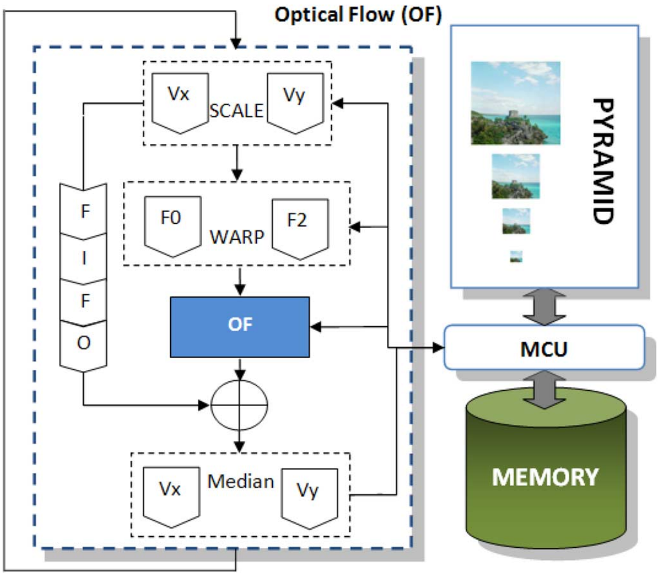


Fig. 3. Hardware system architecture. On the right side, we have the pyramid iteration and its communication with memory. On the left side, the multi-scale optical flow computation (scaling, warping, merging, median filtering, and the L&K optical flow computation) can be seen. The communication with memory is performed through the MCU.

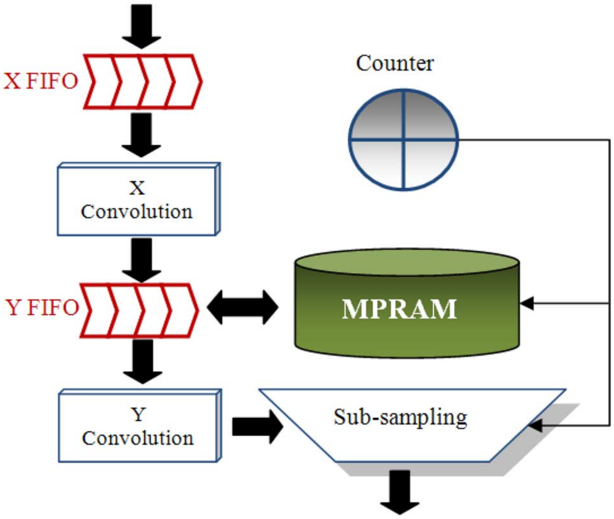


Fig. 4. Architecture for the image sub-sampling required for the pyramid building. The 2-D convolution is split into two different 1-D convolutions to take advantage of the inherent massive parallelism of the device.

- The Merging stage allows the sum of the previous optical flow estimation and the current one in order to compute the final estimation.

The interaction with memory is a very critical problem and needs a dedicated circuit and a specific memory mapping. The parallel access to the RAM blocks is allowed using the multiple available banks and a sequential operation strategy.

- Pyramid: A pyramid is built by a smoothing and sub-sampling circuit (see Fig. 4). Each pyramid scale is obtained sequentially (mainly, due to the limitations of the sequential access to the external memory). Input and output images are directly read/stored into an external RAM memory. The main operations at this step are the 2-D convolution with the five tap Gaussian filter (smoothing) and the sub-sampling. The kernel is a 5-by-5 matrix decomposed in two arrays $K = [1\ 4\ 6\ 4\ 1]/16$, as suggested in [37].

The convolution of the input image is divided in the x and y operations to take advantage of the FPGA massive parallelism. Five different image lines are stored in an embedded multi-port BlockRAM used like a FIFO for the convolution.

After the convolution computation, we send a pixel to the output (the external SRAM) every two clock cycles: one pixel is discarded (sub-sampling).

- Warping: It consists in a bilinear interpolation of the input images with the increment values that we have stored from the optical flow in the previous scale in a LUT.

  The computation of each warped pixel requires the reading of the pair $(\Delta x, \Delta y)$ from their correspondent matrices as well as the pixel P. The integer part of the pair $(\Delta x, \Delta y)$ is used for retrieving from memory the four pixels of the original image. Then, the warped pixel is calculated with the fractional part performing a weighted bilinear interpolation.

  The warping process needs to execute four memory accesses per clock cycle to calculate one warped pixel to achieve the maximum throughput. This is one of the reasons for the choice of a specific MCU to manage data with different Abstract Access Ports (AAP) [36].

  The warping architecture uses a reading AAP of the MCU for accessing the original image. Two reading AAPs are used by the two warping blocks: one for the first and one for the third frame in the temporal sequence. The MCU provides a 36-bit bus allowing to access four pixels per memory read. The X-matrix and Y-matrix which store the new locations of the pixels are provided from the oversampling circuit through two blocking FIFOs.

  The warping requires a neighborhood of four pixels and the number of data available per memory access is limited to four in a same line. Thus, one access brings two pixels of the same line in the best case. Nevertheless, it is impossible to access the four-pixel window in a single memory access. In the worst case, we access four different memory words (all four consecutive memory accesses). Therefore, performance is constrained to up to 4 pixels every 10 memory accesses.

- Merging: This module computes the addition of the previous optical flow estimation and the current one. The result is stored for the next iteration. The non-valid values are propagated from the coarsest scales to the finest ones. These non-confident values are obtained at each scale applying the threshold mentioned before as the eigenvalues product (5). At the last scale, the finest one, we make the logical "*and*" operation between its non-valid values and the propagated ones for the final estimation. The propagation for the other scales is implemented using an "*or*" logical operation; this difference in the computation is performed to weight more the non-valid values of the finest scale because they are the more exact ones in terms of non-confidence. The main problem at this module is the synchronization between the current and the stored results.

TABLE V
ACHIEVED FRAME RATE RELATED TO THE INDICATED SPATIAL RESOLUTION

Resolution	512 x 512	640 x 480	800 x 600	1024 x 1024
Frame rate (fps)	37.39	31.91	20.42	9.34

TABLE VI
COMPARISON OF THE SOFTWARE AND HARDWARE ERROR (USING *LRF* VERSION) FOR THE "*Marble*" SEQUENCE USING A DIFFERENT NUMBER OF SCALES

	Software results			Hardware results		
No. scales	AAE (°)	SAE (°)	Dens. (%)	AAE (°)	SAE (°)	Dens. (%)
1	29.14	24.31	80.90	23.08	22.23	71.51
2	13.39	15.38	79.80	11.12	14.41	70.69
3	9.64	12.26	79.63	9.73	13.03	70.68
4	9.62	13.08	79.49	9.48	12.41	70.58
5	9.16	11.46	79.48	9.99	13.07	70.56

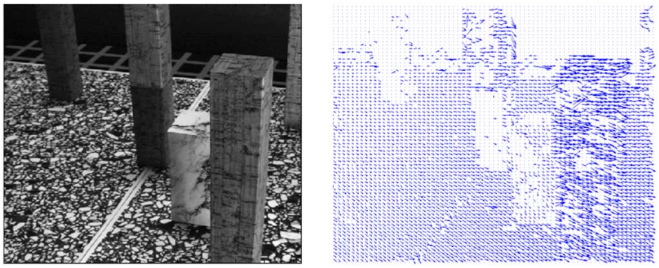


Fig. 5. Captures of the hardware generated qualitative results for the "*Marble*" sequence. The left image shows the central and the right frame of the motion estimation. The arrows have the direction and magnitude of the computed estimation.

- Median Filtering: The stage filters the output of the L&K computation using a 3-by-3 median filter module parameterized by a threshold. The threshold controls the computation of the filter depending on the number of unreliable values. This stage can be enabled or disabled by an input parameter too. Furthermore, the module can be a cascade of two 3-by-3 median filters. Our data (see Tables V and VI) are extracted from a system with a single median filter at this stage.

C. Multi-Scale System Performances and Resource Utilization

In this subsection, we show the results for the implementation of the system including the frame rate and the quantitative and qualitative results. In addition, we also present here the tables with the resource information utilization. For this implementation, we use the *LRF* core (see Section III-A for details).

As proposed in the introduction, our objective is the implementation of a high-performance system, which means a system with a high frame rate to work in real time. In Table V, we can see the information about the frame rate related to the corresponding resolution. With an image resolution of 512 × 512 pixels, the processing of the system reaches more than 37 fps, which is significantly higher than the commonly accepted frame rate for real-time performances (25 fps). Fig. 5 shows some qualitative results for the "*Marble*" sequence.

The hardware results (see Table VI) for this sequence are, in the best case (using 4 scales), very similar in AAE, about 9.5°, to

TABLE VII
PERFORMANCE COMPARISON WITH PREVIOUS WORKS (SORTED BY PUBLICATION DATE) AND ERROR MEASURE FOR "*Yosemite*" SEQUENCE

Implementation	Algorithm Family	Max. Image Resolution	Frame rate (fps)	Throughput (Mpixels/s)	Architecture	AAE (°)	Dens. (%)	Cloud handling
Our mono-scale[c] core	L&K	640x480	270	82.9	Xilinx V4 (83 MHz)	5.97	59.88	Cloudless
Our multi-scale work	L&K	640x480	31.91	9.8	Xilinx V4 (44 MHz)	4.55	58.50	Cloudless
Tomasi [40] (2010)	Multiscale Phase-based	640x480	31.5	9.6	Xilinx V4 (45 MHz)	7.91	92.01	Cloudless
Botella [41] (2010)	Multi-channel Gradient	128x96	16	0.2	Xilinx V2	5.5	100	UNK[d]
Mahalingam [42] (2010)	L&K (mono-scalar)	640x480	30	9.2	Xilinx V2 Pro (55 MHz)	6.37	38.6	UNK
Anguita [25] (2009)	L&K (mono-scalar)	1280x1026	68.5	90.0	Core2 Quad Q9550 (2830 MHz)	3.79	71.8	Cloudless
Gwosdek [43] (2009)	Variational	316x252	210	16.5	Cell Processor (PS3)	5.73	UNK	With clouds
Pauwels [24] (2008)	Phase-based	640x512	48.5	15.9	NVIDIA GeForce 8800 GTX	2.09	63	Cloudless
Diaz [13] (2008)	L&K (mono-scalar)	800x600	170	81.6	Xilinx V2 (82 MHz)	7.86	57.2	Cloudless
Chase [44] (2008)	Tensor-based (mono-scalar)	640x480	64	19.7	Xilinx V2 (187MHz)	12.9	UNK	With clouds
Chase [44] (2008)	Tensor-based (mono-scalar)	640x480	150	46.1	NVIDIA GeForce 8800 GTX	12.9	UNK	With clouds
Wei [45] (2007)	Tensor-based	640x480	64	19.7	Xilinx V2 (100 MHz)	12.7	UNK	Cloudless
Bruhn [46] (2006)	Variational	160x120	63	1.2	Intel Pentium4 (3.06 GHz)	5.77	100	Cloudless
Niitsuma [47] (2005)	Region-based	640x480	30	9.2	Xilinx V2	UNK	UNK	UNK

[c] This implementation only refers to the best mono-scale optical flow core (*LRF* core).
[d] *UNK* means Unknown (data is not provided by the authors).

the software model (which has 9.6°) losing only 9% of density with a very similar standard deviation too.

For the well-known "*Yosemite*" sequence (ignoring the clouds), the hardware AAE results with the multi-scale approach reach 4.55° with a density of 58.49%, (using the *LRF* hardware version), while the software version achieves 3.72° with 63.63% of density (in this case, the AAE increment is of 0.83°). This comparison is carried out using similar representations for hardware and software implementations (using the same regularization filter size, the same median thresholds, without a final thresholding or similar sizes for image boundaries).

In Table VII, we compare our implementation performances with previous works in different architectures and with different algorithms for the optical flow estimation. The advantages of the mono-scale version are obvious; it achieves the best results in terms of computing power (frames per second for the given resolutions) compared to some state of the art works. The multi-scale version does not achieve the same performances, but the accuracy in this case is significantly better as explained previously. Moreover, this table also includes the error measures (AAE and density) for all the alternatives (when provided by the authors). Table VII shows that our implementation achieves a good position in the rank for the Yosemite sequence taking into account that this sequence does not take full advantage of multi-scale performance improvements, because shift ranges are very small ($[-4.28, 3.25]$ for u and $[-4.19, 5.19]$ for v). There are also several L&K software implementations (real-time driven) that present lower errors than ours as the one of Marzat *et al.* [38], with an AAE of 2.34° (density is not provided). There are other approaches, as the OpenCV implementation [19], [39], which achieves 6.10° for a fully dense motion field. In both these latter cases, they are L&K hierarchical and iterative implementations (and the error values are obtained using the Yosemite sequence without clouds). In any case, in the framework of on-chip implementations, it is important to remark that the accuracy of our approach is one of the highest of those described in the literature, which validates our design.

Table VIII shows the information about the hardware resource utilization. Table VII presents the percentages and the total amount of used elements: the total four input LUTs, the Slice Flip Flops, and the Slices, used DSPs, Block RAMs and, finally, the maximum frequency (MHz) which is reached is shown.

The listed options include the complete systems with the implemented cores using either the fixed-point or the floating-point arithmetic; it also includes the interface with the resources of the selected board and the MCU, the implemented modules for the multi-scale computation, the over-sampling module, the warping module (which includes two warping modules, one for each direction, and the interface with the MCU), and the merging module. It is important to remark that the systems use between 61% and 64% of the available resources of a Virtex4 FX100. Therefore, in the framework of real applications, it is possible to add some more cores to the multi-scale architecture to build in new on-chip engines capable of computing other vision features. This would lead to a real-time on-chip engine

TABLE VIII
HARDWARE RESOURCE UTILIZATION FOR THE PRESENTED COMPLETE ARCHITECTURE USING A VIRTEX-4 FX100 FPGA

	4 input LUTs (out of 84352)	Slice Flip-Flops (out of 84352)	Slices (out of 42716)	DSP (160)	Block RAM (378)	Freq. (MHz)
L&K LRF sys	31796 (37%)	24694 (29%)	26036 (61%)	62 (38%)	112 (29%)	44
L&K FLO sys	35753 (42%)	22586 (26%)	27359 (64%)	44 (27%)	107 (28%)	41
L&K LRF core	4589 (5%)	6622 (5%)	4128 (9%)	30 (18%)	48 (12%)	83
L&K FLO core	8160 (9%)	4715 (5%)	6457 (15%)	12 (7%)	48 (12%)	76
Board Interface	4774 (5%)	5195 (6%)	5388 (12%)	0	36 (9%)	112
Over-sampling	413 (1%)	270 (1%)	367 (1%)	0	1 (1%)	86
Warping + Int.	9943 (11%)	9097 (10%)	9894 (23%)	32 (20%)	43 (11%)	51
Merging	364 (1%)	244 (1%)	235 (1%)	0	4 (1%)	107

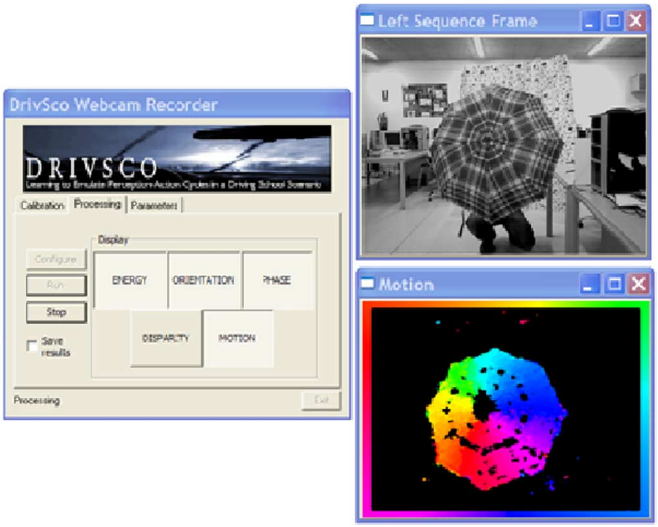


Fig. 6. Capture of the software environment in a real-time execution of the optical flow. The input is an umbrella rotating counter clockwise with a color-coding describing the direction of the motion according to the frame of the picture.

for the computation of low-level vision features which are significantly powerful for systems in the middle and high-level.

IV. Testing Platform

In our work, the FPGA has been used as a coprocessing board and it is connected to a computer. In this system, images are written by the computer into the available memory banks at the board and the communication is carried out using the PCI express interface. We also implemented a hardware-software protocol with a double-buffer architecture for the communication between the board and the computer.

The use of a simple and efficient user interface is a crucial tool for the development of complex systems. In our case, the hardware implementation was tested using a software environment which has the role of an interface between the capture of the images from the real world using different kinds of cameras and the coprocessing board. The developed software allows us to execute the implemented algorithms with the FPGA board for their display and the storing of the obtained results. It also allows the debugging of the hardware algorithms and the comparison with the rest of alternatives and vision algorithms or implementations (hardware or software).

The software platform is freeware (open source) and was released in 2009 by the GNU LGPL license, thus the source files, the documentation, the user's manual, the tutorial, and the video-tutorial are available at [48]. In Fig. 6, we can see a screenshot of this software platform.

V. Conclusion

In this work, we have designed and implemented an embedded system for the optical flow estimation. Among the numerous alternatives, we selected a multi-scale implementation of the well-known L&K method due to its good tradeoff between accuracy and efficiency. However, we have also described the mono-scale L&K approach as a valid alternative when using high frame-rate cameras.

One of the main objectives which we achieved is the real-time optical flow computation capable of dealing with large displacements. The system achieves 270 fps for the mono-scale approach and 32 fps for the multi-scale one for VGA resolution taking full advantage of the massive parallelism of the FPGA, also obtaining good accuracy rates. Moreover, the hardware resource utilization is a key point too. In our case, for the proposed system, the mono-scale implementation corresponds to 10%–15% of the available resources in a Xilinx Virtex4 XC4vfx100 device while the multi-scale takes about 60%. For our system, this increase in resources (roughly 45%) allows us to improve the accuracy results in about 3× while maintaining the density results but leading to almost a 10× decrease of the frame rate (see Table VI). The resource increment is mainly due to the warping stage of the hierarchical implementation, which roughly consumes 23% of the total resources. Furthermore, the maximum working frequency is decreased from 83 MHz for the mono-scale core to 44 MHz for the complete system, being constrained by the warping module whose frequency is 51 MHz. This is a key point to understand that our system bottleneck is the warping computation (in terms of cost and performances). Moreover, as shown in Table VII, our AAE of 4.55 degrees with almost 60% of density is one of the best implementations of the listed ones and actually, the best of the hardware implementations.

The fine pipelined architecture benefits the high system performances and the low power consumption [13] (crucial for industrial applications). Rather than a specific system design, the described implementation shall be seen as a versatile approach that can be easily adapted to different performance versus hardware resource tradeoffs, depending on the target application requirements. The adopted design strategy allows an easy sharing of hardware resources (using more or less pipelined stages and superscalar units). This makes the system definition easy to reuse in different application domains.

The characteristics which have been listed before make the system suitable for the implementation of more complex systems by adding new computation engines of visual features as depth, orientation, and phase. Moreover, we can even add new layers in order to compute IMOs detection, heading, or structure from motion, as indicated in the introduction section.

As future works, we will address the inclusion of additional on-chip image features in order to be able to develop generic and more complex image applications.

References

[1] S. P. Sabatini, F. Solari, and G. M. Bisio, "Spatiotemporal neuromorphic operators for the detection of motion-in-depth," in *Proc. 2nd ICSC Symp. Neural Comput.*, 2000, pp. 874–880.

[2] P. C. Merrell and D. Lee, "Structure from motion using optical flow probability distributions," in *Intelligent Computing: Theory and Applications III*, K. L. Priddy, Ed. Orlando, FL: SPIE, 2005, vol. 5803, pp. 39–48.

[3] K. Pauwels, N. Kruger, M. Lappe, F. Worgotter, and M. M. V. Hulle, "A cortical architecture on parallel hardware for motion processing in real-time," *J. Vision*, vol. 10, no. 18, pp. 1–21, 2010.

[4] K. Pauwels and M. M. V. Hulle, "Optimal instantaneous rigid motion estimation insensitive to local minima," *Comput. Vision Image Understanding*, vol. 104, no. 1, pp. 77–86, 2006.

[5] Y. Matsushita, E. Ofek, W. Ge, X. Tang, and H.-Y. Shum, "Full-frame video stabilization with motion inpainting," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 28, no. 7, pp. 1150–1163, Jul. 2006.

[6] T. Brox, B. Rosenhahn, D. Cremers, and H. Seidel, "High accuracy optical flow serves 3-D pose tracking: Exploiting contour and flow based constraints," in *Proc. Euro. Conf. Comput. Vision (ECCV)*, 2006, pp. 98–111.

[7] A. Kokaram, "On missing data treatment for degraded video and film archives: A survey and a new bayesian approach," *IEEE Trans. Image Process.*, vol. 13, no. 3, pp. 397–415, 2004.

[8] J. M. L. K. Nakayama, "Optical velocity patterns, velocity-sensitive neurons and space perception," *Perception*, vol. 3, pp. 63–80, 1974.

[9] J. Aloimonos, I. Weiss, and A. Bandyopadhyay, "Active vision," *Int. J. Comput. Vision*, vol. 1, pp. 333–356, 1988.

[10] F. Barranco, J. Diaz, E. Ros, and B. del Pino, "Visual system based on artificial retina for motion detection," *IEEE Trans. Syst., Man, Cybern.*, vol. 39, no. 3, pp. 752–762, Jun. 2009.

[11] A. Giachetti, M. Campani, and V. Torre, "The use of optical flow for the autonomous navigation," in *Proceedings of the 3rd European Conference on Computer Vision*, ser. (ECCV). New York: Springer-Verlag, 1994, vol. 1, pp. 146–151.

[12] A. Wali and A. M. Alimi, "Event detection from video surveillance data based on optical flow histogram and high-level feature extraction," in *Proc. 20th Int. Workshop Database Expert Syst. Appl.*, 2009, pp. 221–225.

[13] J. Diaz, E. Ros, R. Agis, and J. Bernier, "Superpipelined high-performance optical-flow computation architecture," *Comput. Vision Image Understand.*, vol. 112, no. 3, pp. 262–273, 2008.

[14] H. Frenz, M. Lappe, M. Kolesnik, and T. Bhrmann, "Estimation of travel distance from visual motion in virtual environments," *ACM Trans. Appl. Perception*, vol. 4, pp. 419–436, 2007.

[15] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. 7th Int. Joint Conf. Artif. Intell.—Vol. 2 (IJCAI)*, 1981, pp. 674–679.

[16] S. S. Beauchemin and J. L. Barron, "The computation of optical flow," *ACM Comput. Surveys*, vol. 27, no. 3, pp. 433–466, 1995.

[17] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, "Performance of optical flow techniques," *Int. J. Comput. Vision*, vol. 12, pp. 43–77, 1994.

[18] J. Bergen, P. Anandan, K. Hanna, and R. Hingorani, "Hierarchical model-based motion estimation," in *Computer Vision ECCV'92*, ser. Lecture Notes in Computer Science, G. Sandini, Ed. New York: Springer-Verlag, 1992, vol. 588, pp. 237–252.

[19] J. Y. Bouguet, "Pyramidal implementation of the Lucas-Kanade feature tracker: Description of the algorithm," OpenCV Document, Intel Microprocessor Research Labs, 2000.

[20] S. Baker and I. Matthews, "Equivalence and efficiency of image alignment algorithms," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recog. (CVPR)*, 2001, vol. 1, pp. 1090–1097.

[21] F. Dellaert and R. T. Collins, "Fast image-based tracking by selective pixel integration," in *Proc. ICCV Workshop Frame Rate Process.*, 1999, pp. 1–22.

[22] G. Hager and P. Belhumeur, "Efficient region tracking with parametric models of geometry and illumination," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 10, pp. 1025–1039, Oct. 1998.

[23] Y. Murachi, Y. Fukuyama, R. Yamamoto, J. Miyakoshi, H. Kawaguchi, H. Ishihara, M. Miyama, Y. Matsuda, and M. Yoshimoto, "A vga 30-fps realtime optical-flow processor core for moving picture recognition," *IEICE Trans. Electron.*, vol. 91, pp. 457–464, 2008.

[24] K. Pauwels and M. M. V. Hulle, "Realtime phase-based optical flow on the GPU," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recog. Workshops (CVPRW)*, 2008, pp. 1–8.

[25] M. Anguita, J. Diaz, E. Ros, and F. J. Fernandez-Baldomero, "Optimization strategies for high-performance computing of optical-flow in general-purpose processors," *IEEE Trans. Circuits Syst. for Video Technol.*, vol. 19, no. 10, pp. 1475–1488, Oct. 2009.

[26] T. Kohlberger, C. Schnorr, A. Bruhn, and J. Weickert, "Domain decomposition for variational optical-flow computation," *IEEE Trans. Image Process.*, vol. 14, no. 8, pp. 1125–1137, Aug. 2005.

[27] J. Diaz, E. Ros, F. Pelayo, E. M. Ortigosa, and S. Mota, "FPGA-based real-time optical-flow system," *IEEE Trans. Circuits Syst. for Video Technol.*, vol. 16, no. 2, pp. 274–279, Feb. 2006.

[28] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artif. Intell.*, vol. 17, pp. 185–203, 1981.

[29] J. W. Brandt, "Improved accuracy in gradient-based optical flow estimation," *Int. J. Comput. Vision*, vol. 25, pp. 5–22, 1997.

[30] Sevensols, Granada, Spain, "Seven solutions," 2011. [Online]. Available: http://www.sevensols.com/

[31] E. Ortigosa, A. Canas, E. Ros, P. Ortigosa, S. Mota, and J. Diaz, "Hardware description of multi-layer perceptrons with different abstraction levels," *Microprocess. Microsyst.*, vol. 30, no. 7, pp. 435–444, 2006.

[32] J. Diaz, "Multimodal bio-inspired vision system. High performance motion and stereo processing architecture," Ph.D. dissertation, Dept. Comput. Arch. Technol., Univ. Granada, Granada, Spain, 2006.

[33] E. P. Simoncelli, "Design of multi-dimensional derivative filters," in *Proc. IEEE Int. Conf. Image Process.*, 1994, pp. 790–794.

[34] Xilinx, San Jose, CA, "FPGA and CPLD solutions from Xilinx, Inc.," 2011. [Online]. Available: http://www.xilinx.com/

[35] M. Otte and H.-H. Nagel, "Optical flow estimation: Advances and comparisons," in *Proceedings of the Third European Conference on Computer Vision*, ser. ECCV'94. New York: Springer-Verlag, 1994, pp. 51–60.

[36] M. Vanegas, M. Tomasi, J. Diaz, and E. Ros, "Multi-port abstraction layer for FPGA intensive memory exploitation applications," *J. Syst. Arch.*, vol. 56, no. 9, pp. 442–451, 2010.

[37] P. J. Burt, Edward, and E. H. Adelson, "The laplacian pyramid as a compact image code," *IEEE Trans. Commun.*, vol. 31, no. 4, pp. 532–540, Apr. 1983.

[38] D. A. Marzat J. and Y. Dumortier, "Real-time dense and accurate parallel optical flow using cuda," in *Proc. 17th Int. Conf. Central Euro. Comput. Graphics, Visualization, Comput. Vision (WSCG)*, 2009, pp. 105–111.

[39] M. Heindlmaier, L. Yu, and K. Diepold, "The impact of nonlinear filtering and confidence information on optical flow estimation in a lucas &#38; kanade framework," in *Proc. 16th IEEE Int. Conf. Image Process.*, 2009, pp. 1573–1576.

[40] M. Tomasi, M. Vanegas, F. Barranco, J. Diaz, and E. Ros, "High-performance optical-flow architecture based on a multiscale, multi-orientation phase-based model," *IEEE Trans. Circuits Syst. for Video Technol.*, vol. 20, no. 12, pp. 1797–1807, Dec. 2010.

[41] G. Botella, A. Garcia, M. Rodriguez-Alvarez, E. Ros, U. Meyer-Baese, and M. C. Molina, "Robust bioinspired architecture for optical-flow computation," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 18, no. 4, pp. 616–629, Apr. 2010.

[42] V. Mahalingam, K. Bhattacharya, N. Ranganathan, H. Chakravarthula, R. Murphy, and K. Pratt, "A VLSI architecture and algorithm for lucas-kanade-based optical flow computation," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 18, no. 1, pp. 29–38, Jan. 2010.

[43] P. Gwosdek, A. Bruhn, and J. Weickert, "Variational optic flow on the sony playstation 3," *J. Real-Time Image Process.*, vol. 5, pp. 163–177, 2010.

[44] J. Chase, B. Nelson, J. Bodily, Z. Wei, and D.-J. Lee, “Real-time optical flow calculations on FPGA and GPU architectures: A comparison study,” in *Proc. 16th Int. Symp. Field-Program. Custom Comput. Mach. (FCCM)*, 2008, pp. 173–182.

[45] M. Martineau, Z. Wei, D.-J. Lee, and M. Martineau, “A fast and accurate tensor-based optical flow algorithm implemented in fpga,” in *Proc. IEEE Workshop Appl. Comput. Vision (WACV)*, 2007, pp. 18–18.

[46] A. Bruhn, J. Weickert, T. Kohlberger, and C. Schnrr, “A multigrid platform for real-time motion computation with discontinuity-preserving variational methods,” *Int. J. Comput. Vision*, vol. 70, pp. 257–277, 2006.

[47] H. Niitsuma and T. Maruyama, “High speed computation of the optical flow,” in *Proc. Image Anal. Process. (ICIAP)*, 2005, vol. 3617, pp. 287–295.

[48] “Open rt-Vision Project,” 2011. [Online]. Available: http://code.google.com/p/open-rtvision/

Francisco Barranco received the B.S. degree in computer science and the M.Sc. degree in computer and network engineering from the University of Granada, Granada, Spain, in 2007 and 2008, respectively.

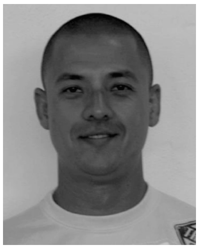
He is with the Department of Architecture and Computer Technology, University of Granada. His main research interests deal with image processing architectures and embedded systems based on reconfigurable devices, real-time machine vision, general purpose graphical programming devices, biologically processing schemes, spiking neurons. He is currently participating in an EU Project related with adaptive learning mechanisms and conventional control.

Matteo Tomasi received the Bachelor’s degree in electronic engineering from the University of Cagliari, Italy, in 2006, and the M.Sc. degree in computer engineering from University of Granada, Granada, Spain, in 2007, where he is currently pursuing the Ph.D. degree from the Department of Computer Architecture and Technology.

He was involved with the European Project DRIVSCO. His main research interests include HDL and hardware design, real-time systems, reconfigurable systems, computer vision and image processing.

Javier Diaz received the M.S. degree in electronics engineering and the Ph.D. degree in electronics from the University of Granada, Granada, Spain, in 2002 and 2006, respectively.

Currently, he is an Assistant Professor with the Department of Computer Architecture and Technology, University of Granada. His main research interests include cognitive vision systems, high performance image processing architectures, and embedded systems based on reconfigurable devices. He is also interested in spiking neurons, biomedical devices, and robotics.



Mauricio Vanegas received the Bachelor’s degree in electronic engineering from the University Pontificia Bolivariana, Medellin, Colombia, in 2001, and the Ph.D. degree from the University of Granada, Granada, Spain, in 2010.

He was a Lecturer with the Electrical and Electronic faculty of the University Pontificia Bolivariana from 2002 until 2006. He is currently with the Department of Biophysical and Electronic Engineering (DIBE), University of Genoa, Genoa, Italy. He is interested in embedded systems, HDL and hardware specification, real-time systems, and reconfigurable systems.

Eduardo Ros received the Ph.D. degree from the University of Granada, Granada, Spain, in 1997.

He is currently an Associate Professor with the Department of Computer Architecture and Technology, University of Granada, where he is currently the responsible researcher at the University of Granada of two European projects related with bio-inspired processing schemes and real-time image processing. His research interests include hardware implementation of digital circuits for real time processing in embedded systems and high performance computer vision.

2.2 Low-cost architecture for extracting optical flow, disparity, energy, and orientation

The journal article associated to this part of the dissertation is:

- F. Barranco, M. Tomasi, J. Díaz, M. Vanegas, E. Ros, Pipelined Architecture for Real-Time Low-Cost Extraction of Visual Primitives based on FPGAs. **Submitted to Journal of Digital Signal Processing**.
    - Status: **Submitted**.
    - Impact Factor (JCR 2010): 1.22
    - Subject Category:
        - ∗ Engineering, Electrical and Electronic. Ranking 102 / 247.

Pipelined Architecture for Real-Time Cost-Optimized Extraction of Visual Primitives based on FPGAs

F. Barranco[a], M. Tomasi[a,b], J. Díaz[a], M. Vanegas[a,c], E. Ros[a]

[a]*Dept. Computer Architecture and Technology, ETSIIT, CITIC, University of Granada, C/P. Daniel Saucedo Aranda, s/n, E-18071, Granada, Spain*
[b]*Schepens Eye Research Institute, Department of Ophthalmology, Harvard Medical School, 20 Staniford St, 02114, Boston, MA, United States*
[c]*PSPC Group, Department of Biophysical and Electronic Engineering (DIBE), University of Genoa, Via Opera Pia 11A, I-16145, Genoa, Italy*

Abstract

This paper presents an architecture for the extraction of visual primitives on chip: energy, orientation, disparity, and optical flow. This cost-optimized architecture processes in real time high-resolution images for real-life applications. In fact, we present a versatile architecture that may be customized for different performance requirements depending on the target application. In this case, dedicated hardware and its potential on-chip implementation on FPGA devices become an efficient solution. We have developed a multi-scale approach for the computation of the gradient-based primitives. Gradient-based methods are very popular in the literature because they provide a very competitive accuracy vs. efficiency trade-off. The hardware implementation of the system is performed using superscalar fine-grain pipelines to exploit the maximum degree of parallelism provided by the FPGA. The system reaches 350 and 270 VGA frames per second (fps) for the disparity and optical flow computations respectively in their mono-scale version and up to 32 fps for the multi-scale scheme extracting all the described features in parallel. In this work we also analyze the performance in accuracy and hardware resources of the proposed implementation.

Keywords:
Image processing, Hardware implementation, Machine vision, Real-time systems, Reconfigurable architectures

1. Introduction

The perception of motion and depth is essential for an autonomous system which is moving in a dynamic environment. Binocular vision is the mechanism we use to perceive depth. It is defined as the disparity or difference between the projections of the real image on the right and left visual sensors. Analogously, motion is perceived as a temporal difference of these projections between consecutive instants (or image frames in case of using cameras as sensors). Therefore, algorithms for disparity and optical flow computation should be respectively seen as searching models for spatial or temporal feature correspondence matching. A lot of applications require the integration of these visual primitives: structure from motion [1], robust real-time tracking [2], obstacle detection [3] [4], autonomous navigation [5], active vision [6], video-surveillance [7], advanced driver assistance systems [8] or even, biomedicine [9]. Furthermore, the computation of local descriptors as the ones implemented for the presented architecture, are also the first stages towards the long term goal of scene understanding. Common local descriptors are orientation and energy, they encode the geometric information and the local contrast respectively. These features are widely used in the literature for applications such as texture analysis [10], pattern recognition [11], and object recognition [12] [13]. In addition, they are the base for complex descriptors [14] [15].

In this work, the image registration is based on a gradient method. Registration techniques can be classified as local or global methods. Local methods were firstly developed in 1981 by Lucas & Kanade [16] [17]. This first work [17] is directly connected with the disparity computation (registration). These methods assume that motion or disparity are constant in a local neighborhood. Thus, the computation of an estimation for a pixel is done focusing only on its neighborhood. The constraint for global method approaches is the smoothness of motion or disparity fields which means that the estimation for one pixel depends on the whole image pixels. They are based on Horn & Schunk's works [18].

The computational load for the estimation of disparity and optical flow is very high. On the other hand, their real-time computation is essential for many real-world applications. To achieve this high performance, we have different alternatives: dedicated hardware architectures [19], graphics processing units (GPUs) as accelerators [20] [21] and even optimized software implementations [22] or clusters of processors [23]. Finally, in previous works [21] [8] [24] we also find FPGA-based solutions.

In our work we select the Lucas & Kanade (L&K) algorithm and the implementation based on Barron's works [25] [26]. We include an extension to the original algorithm, a hierarchical architecture based on [27], capable of working with an extended motion range much larger than standard mono-scale approaches (with a typical range of few pixels [8]). It briefly consists on the construction of a pyramid with the input images. Then, it performs the search over a small number of pixels at the coarsest scale. The obtained estimation is used as a seed to search locally at the next finer scale and so on to the finest one.

We implement the proposed architecture on FPGA because the chosen vision processing algorithms are good candidates due to their potential for a high-performance parallelization. The results are obtained using a fine-pipeline based architecture designed in a modular way. Our objective is to achieve a throughput of one pixel per clock cycle for the whole processing. The correct exploitation of these high-performance characteristics allows us to achieve, in the case of the mono-scale versions up to 350 and 270 fps (frames per second) for disparity and optical flow respectively, for an image resolution of 640x480. In the case of the multi-scale implementation, the frame rate of the complete system reaches almost 32 fps for the same image resolution in an accurate way and with a motion range 30 times larger (also larger than previously described approaches [8] [24]). The accuracy, stability and density of our results are analyzed in Section IV.

The lack of works involving such volume of visual processing is due to the complexity of the problem. Note that including together on the same chip multiple processing datapaths with limited resources is a much more complex problem than just integrating together different cores on the chip independently instantiated. Routing of large number of logic blocks, system resource sharing, memory access scheduling with a reduced number of external memory banks are just examples of issues that need to be addressed when developing this computing engine. However, we have published another work [28] which deals with a similar visual processing. We based our design in the architecture described in that work, and specially in the case of the optical flow is also based in [8], but there are some differences. In the contribution presented here, our development uses gradient-based algorithms and the previous work a phase-based approach. A phase-based approach is more robust against illumination variations which leads to more accurate results for real-world sequences. On the other hand, gradient-based developments presents a good trade-off between the efficiency and the resources utilization (they are computationally lighter than the phase-based ones). Methods for orientation and energy computation are also different, again toward reducing resource costs. In addition to this, we also show the performance evaluation of the energy and orientation visual modalities, and a complete updated state-of-the-art comparison. Our contribution describes the main differences between both works and especially assesses the resource cost analysis, which is an important issue when implementing complex processing architectures on a single chip.

Our purpose in this work is to evolve our system to the following level, developing a real-time visual processing engine as a first layer for dynamic environment applications. In this case, the current state of the art is insufficient to satisfy our requirements since the phase-based system entails high resource utilization for its real-time computation. Once the optimization of the resource consumption and the sharing strategies are exhausted, the solution of the problem consists in a design with more expensive chips or boards (to increase the hardware resources) or even the implementation in a multi-chip scheme. In our case, the adoption of a new model, based on image gradient, is revealed as the best alternative to obtain a considerable resource reduction to satisfy our aims. This is the reason we present our system as a cost-optimized version of the low-level vision processing system on-chip.

This paper is structured as follows: in Section II, we summarize the algorithms for feature extraction and the multi-scale extension. Section III details the hardware implementation and presents the results comparing the proposed alternatives and the hardware resource utilization. Section IV shows the architecture of the final system and its benchmarking. Finally, Section V briefly presents the conclusions of the paper.

2. Description of the gradient-based image analysis algorithms

In this section we describe the algorithms we use for the extraction of the different primitives. The local contrast descriptors (energy and orientation) were extracted using the spatial gradient of the image: for the energy we use the module and for the orientation the direction.

Optical flow and disparity estimations are computed using the well-known Lucas&Kanade algorithm [16] [17]. In both cases we also add the multi-scale extension in order to increase the working range of the computed estimations and properly tune the spatio-temporal frequency of the objects in the scene. All these modules are gradient-based algorithms in comparison with the modules developed in [28] that were based on phase.

2.1. *Local contrast descriptors: Energy and Orientation*

For the computation of these two local characteristics of the image we need the local spatial derivatives I_x and I_y (we need the local gradient) which are previously smoothed. They are also used for the computation of the optical flow, using the kernels proposed in [29]. Beginning with these two derivatives, we obtain the Energy (E) and Orientation (O) using expressions (1) and (2) respectively, for each pixel (x, y)

$$E(x,y) = \sqrt{I_x^2 + I_y^2} \tag{1}$$

$$O(x,y) = arctan\left(\frac{I_y}{I_x}\right) \tag{2}$$

2.2. *Optical Flow*

The computation of the optical flow computation assumes the constancy of the brightness of the pixel through the time. In our case we choose the local Lucas&Kanade (L&K) [16] [17] method which assumes that flow is constant for each pixel in its neighborhood. It is currently one of the most used due to its accuracy vs. computational complexity trade-off [30] [31].

The assumption of the constancy of the image intensity is known as Optical Flow Constraint (OFC) and can be expressed as in (3),

$$I(x,y,t) = I(x+u, y+v, t+1). \tag{3}$$

where I stands for the Image intensity for the pixel at the (x, y) location at time t. By this way, the optical flow is defined as a velocity vector (u, v). Applying the first-order Taylor expansion to linearize the expression we obtain (4)

$$I_x u + I_y v + I_t = 0. \tag{4}$$

where I_x, I_y and I_t stand for the partial derivatives. From (4) we cannot determine a unique solution, we need another constraint: local methods assume that flow is constant in a local neighborhood. We estimate the optical flow by minimizing the energy function as in (5) building an over-constrained equation system.

$$E(u,v) = \frac{1}{2}\sum_{i\in\Omega}(W_i^2(I_x u + I_y v + I_t)^2) \tag{5}$$

In (5) W_i stands for the a matrix that weights more the values in the center than the ones in the surround, of the pixels in neighborhood Ω. And then, for the resolution of the system, we use least squares-fitting procedure obtaining(6).

$$(u,v) = (A^T W^2 A)^{-1} A^T W^2 b \tag{6}$$

In (6), A and b stand for the coefficient matrices and the independent terms respectively. From (6) we solve obtaining a 2-by-2 linear system defined by (8)

$$(A^T W^2 A) = \begin{bmatrix} \sum_{i\in\Omega} W_i^2 I_{xi}^2 & \sum_{i\in\Omega} W_i^2 I_{xi} I_{yi} \\ \sum_{i\in\Omega} W_i^2 I_{xi} I_{yi} & \sum_{i\in\Omega} W_i^2 I_{yi}^2 \end{bmatrix} \tag{7}$$

$$(A^T W^2 b) = \begin{bmatrix} \sum_{i\in\Omega} W_i^2 I_{xi} I_{ti} \\ \sum_{i\in\Omega} W_i^2 I_{yi} I_{ti} \end{bmatrix} \tag{8}$$

As quality metrics for our estimations we use the AAE and SAE (Average and Standard deviation of the Angular Error) proposed by Barron in [25]. The AAE is computed as in (9), where $\vec{g}$ is the ground-truth vector velocity and $\vec{e}$ is our estimation for the pixel i. The SAE is computed as the standard deviation of the angular error.

We also compute the density of the results (our estimations are sparse due to a confidence thresholding). The most important difference is that Barron computes the confidence using the minimum eigenvalue of (8) while we use its determinant (the product of eigenvalues of (8)). This alternative does not reduce significantly the precision as shown in [32] [8] but simplifies the implementation reducing the computational complexity.

$$AAE = \frac{\sum_{i=1}^{N} arccos(\vec{g}_i \cdot \vec{e}_i)}{N} \tag{9}$$

The main drawback of the L&K algorithm is the reduced motion range for the estimation of the image displacements, typically a few pixels but it depends on the spatial frequency (size of objects or texture patterns) of the image objects [8]. We can adopt one of the following strategies to solve this problem: increasing the frame rate of the sequence to decrease the motion range (it depends on the capture device or the available sequences and is not always possible) and implementing an hierarchical extension [8][33][34][35]. The second approach consists in a multi-scale implementation that computes the optical flow velocity components for each different spatial resolution input simulating the use of filters of different sizes (by previously scaling the input image) for a better tuning of the different range displacements. This hierarchical approach allows the utilization of conventional cameras with standard video-rates even with the presence of large motion objects.

2.3. Disparity

The disparity, in a binocular system as the one we propose, is defined for a pixel as the difference in the x coordinate between right and left images. It is expressed in (10)

$$I^R(x) = I^L(x + \delta(x)) \tag{10}$$

where I^R and I^L are respectively the intensity of right and left images and $\delta(x)$ is the disparity. The techniques to compute the disparity can be grouped again into local and global methods. Local methods are centered in the neighborhood surrounding a pixel to compute the disparity. Global methods take into account the complete image. In our case, we use the Lucas&Kanade [17] algorithm as in the optical flow. This technique estimates small local disparities assuming the intensity or brightness constancy of a pixel between left and right images. In order to increase the working range of the model we use again the multi-scale extension.

The resolution of the system in this case is similar to the resolution of the previous presented section. In this case, from (10) we apply the Taylor expansion to obtain (11)

$$I(x + \delta) \approx I(x) + \delta I'(x) \tag{11}$$

and we solve (10) minimizing the error with respect to the disparity, obtaining (12)

$$\delta = \frac{\sum_{i\in\Omega} W_i^2 L_{xi}(R_i - L_i)}{\sum_{i\in\Omega} W_i^2 L_{xi}^2} \tag{12}$$

where we have simplified the mathematical notation and added the weighting matrix W_i, as in the optical flow case. L_{xi} stands for the partial derivative in x of the left image and R_i and L_i are the values of left and right images for the pixel i in the neighborhood Ω.

As error metrics we use the MAE and SAE (Mean Absolute Error and Standard deviation of the Absolute Error).In (13) we define the Absolute Error for a pixel, where e_i and d_i are the estimated and the real disparity for the pixel i, and the SAE is computed as the standard deviatio of the Absolute Error. We also include the density (as in the case of motion) and the RMS defined in (14)

$$MAE = \frac{\sum_{i=1}^{N} |e_i - d_i|}{N} \tag{13}$$

$$RMS = \sqrt{\frac{\sum_{i=1}^{N} (\bar{x_i} - x_i)^2}{N}} \tag{14}$$

2.4. Multi-scale implementation

The hierarchical approach is necessary to increase the dynamic range of our complete system. The estimated values for the disparity and the optical flow with the multi-scale extension with five scales are up to 30 times higher than the computed with the mono-scale versions. This approach is based on Bergen's work [27]. The main modules of our multi-scale extension are:

- Gaussian pyramid computation: For the input frames (3 in the case of the optical flow and one additional received from the other sensor for the disparity computation). Their computation depends on the number of scales we use for the implementation. This value also depends on the resolution of the image and the displacement ranges we want to cover. Furthermore, we perform an initial filtering to reduce the aliasing effect produced by the sampling process. This development is based on [36].

- Visual primitive estimation: This stage computes the low-level visual primitives. The computation of the energy and orientation is performed in a mono-scale way but they are computed for each scale (however, in the testing section we present only the last scale results).

- Scaling: This stage expands the current scale results to the resolution of the subsequent finer scale (the sampling factor is 2)

- Warping: This is the most computationally complex stage of the multi-scale implementation. It consists in the warping of the current images with the computed estimation (disparity or motion) to reduce the displacements with subpixel accuracy. The warping is 1D for the disparity computation (only in x direction) but it is 2D for the motion estimation (x and y directions). These warped frames will be the input frames for the primitive computation in the next scale. The design of a hierarchical scheme is essential for the disparity implementation due to the disparity ranges. Furthermore, we also need the undistortion and rectification stage for the disparity computation, this stage compensates the lens distortion effects and transform the epipolar geometry by aligning the image planes (matching the epipolar lines up).

$$I_L^s(x) = Warp(I_L^{s-1}(x) - 2\Theta(\delta^{s-1}(x))). \qquad (15)$$

$$I^s(x,y) = Warp(I^{s-1}(x,y) - 2\Theta(v^{s-1}(x,y))(1-f)). (16)$$

The warping computation is shown in (15) and (16) for disparity and optical flow. In both cases, Θ is the upsampling operator for $\delta(x)$ (disparity) and $v(x,y)$ (optical flow field) and $Warp$ stands for the warping operator and it entails an interpolation (bilinear in the case of optical flow). In (16) f is the frame number, in our optical flow implementation we use 0 to 2 (the temporal window is of 3 frames).

- Merging: This stage combines the results of the previous scale estimations with the current one to achieve the total value of the visual primitive.

Fig.1 illustrates the multi-scale approach. Firstly, it computes the pyramid for the input images and computes the first estimation. Then, it starts the loop: the first operation is the expansion or upsampling of the last estimation results, the following stage is the warping using the obtained upsampled estimation and the frames for the next finer scale; the next step is the new estimation which receives as input the warped frames computed in the previous step and finally, the sum of the new estimation and the previous partial one to achieve a final estimation. This loop is iterated depending on the number of scales.

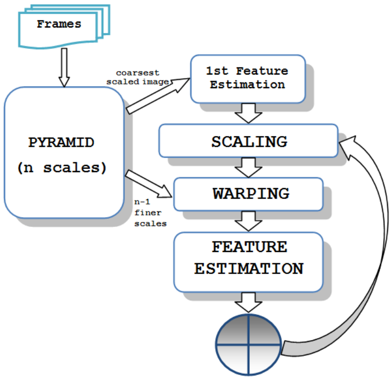


Figure 1: Scheme of the multi-scale extension for the extraction of visual primitives. The stages from the scaling to the merging are iterated from the coarsest to the finest scale (as many times as number of scales used).

3. Hardware Implementation

As mentioned before, the implementation is performed using an FPGA device, in our case a Xilinx Virtex4 XC4vfx100 chip. The board, a Xirca V4 [37], provides a PCIe interface and four SRAM ZBT memory banks of 8 MB. This platform is able to work as co-processing or stand-alone platform.

The hardware implementation has been performed using two abstraction levels: the primitive estimations and the multi-scale modules are implemented using the Handel-C language because of its suitability for algorithmic descriptions (without significantly degrading the performance or increasing the resource utilization [38]) taking into account the complexity of the architecture and the design time; the communication with memory, interfaces between modules, and the memory control unit (MCU, for details see [39]) are developed in VHDL, because they are critical architecture elements that require an optimal implementation.

The development of our architecture is performed implementing fine-grain pipelined datapaths. This strategy benefits the system performance and the low power consumption [8] taking advantage of the highly massive level of parallelism inherent to the regular and local vision algorithms. Rather than a specific system design, our architecture is a versatile and adaptable design for different target application requirements.

Our implementation is based on fixed-point arithmetic. The evaluation of the accuracy degradation with respect to the software floating point version, consists in measuring the final accuracy of a version based on fixed-point arithmetic. We test different bitwidths for the variables used at each functional stage searching for the shortest bitwidth achieving a minimum of accuracy degradation. Once we have the more efficient bitwidth, we set it for the current stage and advance to the next stage. We do this evaluation for each operation along the datapath.

3.1. *Local Contrast Descriptors: Energy and Orientation*

For the computation of these two primitives we need the derivatives of the current input frame. They are computed for all the scales independently and sequentially (the presented results in Fig. 7 belong to the finest grain image scale).

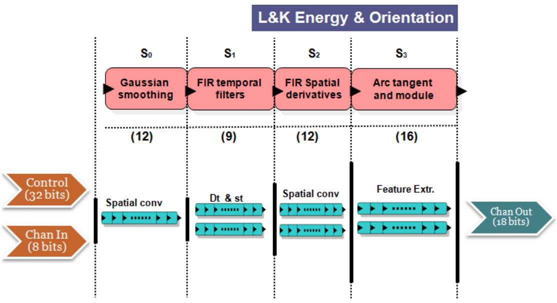


Figure 2: Scheme of the pipelined stages for the local contrast descriptors core. It describes the computation stages (from S_0 to S_2) indicating the pipelined stages (in brackets) and the number of parallel datapaths for each one of them.

The stages for the computation of these local contrast descriptors are listed below and they are illustrated in Fig. 2:

- S_0: this stage performs the Gaussian filter with the kernel $K = [\ 1\ 2\ 1]/4$ to reduce the aliasing.
- S_1: it computes the partial spatial derivatives I_{xi}, I_{yi} of (4).
- S_2: it calculates the orientation and the energy using the module and the direction of the gradient according to (1) and (2).

In this core we employ 68 parallel processing units. This term refers to each one of the complete processor that computes one of the parallel paths in the superscalar architecture. Thus, it is computed taking into account the number of micro-stages that has each complex stage, and the number of parallel paths that are computed at the same time (as it is micropipelined-based design, each stage is computed at the same time by a different parallel processing unit). In this case, as S_0 has a single path (12 parallel pipelined stages), and S_1 and S_2 are structured in 2 paths (one path for each local contrast descriptor, with a total number of 28 pipelined stages). As shown in Fig. 2 the inputs have 8 bits (integers) and the output 18 bits, 9 for each primitive (8 bits of integer part and 1 bit for the fractional part).

These primitives are used for all kinds of applications. For instance, the orientation is very useful in pattern recognition tasks, 3D reconstruction or tracking [14] [12] [13] [15]. As our scheme is based on the gradient, the module used for the computation of these primitives can be integrated in any other primitive core to save resources.

3.2. Lucas&Kanade optical flow core

The implementation of the L&K core is based in previous works [40] [8]. The most significant changes are the migration to the new platform, the connection with the new memory interface and through the MCU (Memory Control Unit) that multiplexes in time the huge amount of memory accesses [39], the multi-scale implementation with warping and finally, the integration with the new cores for the disparity and the implementation of the local contrast descriptors.

Our implementation uses only 3 frames for the optical flow estimation due to the hardware cost that involves the warping stage. This operation is the most expensive in terms of hardware resources as explained in the next section.

The input of the new core is composed by two channels, as shown in Fig. 3: *Pixel In* is the channel of the input frames (3 frames with 8 bits/pixel) and *Control*. *Chan Out* is the optical flow estimation, the component velocities for x and y directions (12 bits per component).

The core computation is divided into 5 stages:

- S_0: The stage consists in filtering the input frames. The filtering uses a Gaussian smoothing kernel of 3 taps, $K = [1\ 2\ 1]/4$. It reduces the aliasing effects and increases the accuracy of the image derivatives.
- S_1: This stage computes the temporal derivative (D_t) and the spatio-temporal smoothing (S_t) of the three frames. This computation is carried out using a derivative and a smoothing filter of 3 taps, the kernels for the derivative filters are based on the Simoncelli's work [29].
- S_2: It computes the spatial derivatives from the latter results: I_t is a partial derivative obtained by applying a Gaussian filtering to the temporal derivative; I_x and I_y are the partial derivatives achieved by differentiating the S_t term with a derivative filter in the respective direction following the Simoncelli complementary derivative kernels approach [29].
- S_3: This stage computes the coefficients of the linear system defined in (8). The weights W_i for the neighborhood are set by the 5-by-5 separable kernel used in [25] [32] [8]: $W = [1\ 4\ 6\ 4\ 1]/16$. It computes the $W_i^2 I_{ti}^2$ coefficient used as a threshold for the temporal noise too.
- S_4: The last stage calculates the resolution of the 2-by-2 system and uses the determinant of the resultant matrix to threshold the less confident re-

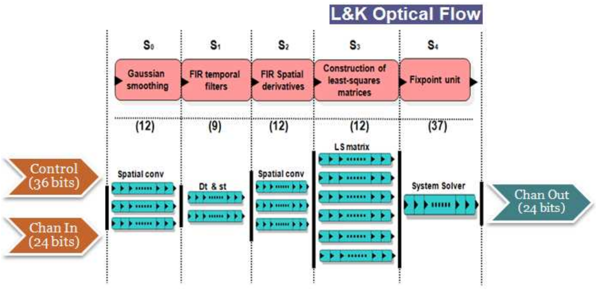


Figure 3: Scheme of the pipeline stages for the optical flow core. It describes the computation stages (from S_0 to S_5) and shows the pipeline stages and the number of datapaths for each one of them.

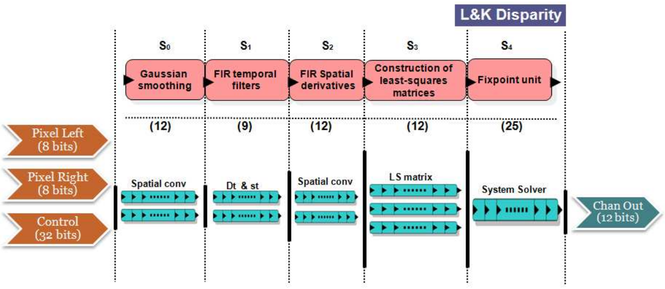


Figure 4: Scheme of the pipeline stages for the disparity core. It describes the computation stages (from S_0 to S_5) and shows the pipeline stages and the number of datapaths for each one of them.

sults. The system also uses a divider from the Xilinx CoreGenerator [37].

For this system we use 199 parallel processing units: the stage S_0 has 3 paths (one for each frame) for the Gaussian filtering, S_1 has 2 paths (for the temporal derivative and the temporal smoothing), S_2 has 3 paths (one for each derivative I_x, I_y, I_t), S_3 has 6 paths (one for each coefficient of (8)) and finally S_4 has only one path.

3.3. Lucas&Kanade disparity core

The implementation of the disparity is also based on the L&K algorithm extended with the hierarchical multi-scale scheme increasing the working range.

Three channels are the inputs of our core, as shown in Fig. 4: *Pixel In Left* and *Pixel In Right* are the input frames (8 bits/pixel for each frame) and *Control* is the channel for the parameters of the core (the number of scales, the image resolution and the confidence thresholds). The output (*Chan Out*) is the disparity estimation (12 bits: 4 bits for the fractional part, 7 for the integer part and the most significant bit for the sign). The core computation is divided into 5 functional stages:

- S_0: The stage consists in filtering left and right frames. The filtering uses a Gaussian smoothing kernel of 3 taps, $K = [1\ 2\ 1]/4$. It reduces the aliasing effects.
- S_1: It computes the left-right image difference and the smoothed frames.
- S_2: It computes the spatial derivatives: the partial derivatives I_{xi}, I_{yi} of (4).
- S_3: This stage computes the coefficients of the linear system of (8). The weights W_i for the neighborhood are set by the 5-by-5 separable kernel used in [25] [32] [8]: $W = [1\ 4\ 6\ 4\ 1]/16$. It computes the $W_i^2 I_{ti}^2$ coefficient used as a threshold for the temporal noise too.
- S_4: The last stage calculates the resolution of the 2-by-2 system and uses the determinant of the resultant matrix to threshold the less confident results. The implementation of this stage is performed with a fixed-point arithmetic (using as divider an IP core from the Xilinx CoreGenerator [37]).

For this system we use 127 parallel processing units, as shown in Fig. 4: S_0 and S_1 has two paths correspondent to each input channel (and S_1 has a double path for the temporal derivatives and the smoothing computation, 42 superscalar parallel pipeline stages), S_2 has two paths (one for each partial derivative, 24 pipeline stages), S_3 has three paths (one for each coefficient of (8), 36 pipeline stages) and finally, S_4 has a single path (with 25 pipeline stages).

3.4. Multi-scale extension

The hardware implementation of the multi-scale extension with warping is usually avoided in this kind of architectures due to its complexity and the resource cost. This extension allows to cope with large motion without requiring any special cameras. As mentioned in the Introduction, this part of the work is based on a previous work [28].

The data flow of our scheme is displayed in Fig.1 and summarized as follows:

- The Pyramid module consists in a smoothing and a sampling. The main computation is done by a 2D convolution with a Gaussian filter of 5 taps with a kernel decomposed in two arrays $K = [1\ 4\ 6\ 4\ 1]/16$ as suggested in [36]. The convolution of the input image is decomposed in the *x* and *y* operations to keep as small as possible the resource utilization of the system.

- The Scaling circuit reads the old partial estimation and scales it to the next finer scale (the upsampling factor is 2).
- The Warping circuit reads the correspondent frames for the current scale (produced by the pyramid) and shifts them using the scaled estimation primitive. Actually, it consists in a bilinear interpolation of the input images with the values of the previously estimated primitive. The case of the warping for the optical flow is the most complex in terms of cost. The bilinear interpolation used by the warping requires 4 access per pixel (the central pixel, the right one, the one below and the bottom-right corner one, see (16)). The disparity warping needs only two accesses per clock cycle for the computation of one warped pixel and, furthermore, it has a single warping block (see (15)).
- The Merging stage performs the sum of the previous estimations and the current ones to compute the final results. The unreliability marks are propagated. The main problem at this module is the synchronization between current and stored results.
- The Median filter regularizes the partial/final results and stores them. This stage is a cascade of two median filters with a size of 3 by 3. This module filters the partial estimations of the primitive and the final estimation. This regularization also increases the density of the maps.

The rest of blocks in the processing stages interact with memory through multiplexed memory accesses [39].

4. Hardware performance and testing

This section summarizes the analysis of the developed system in terms of accuracy, stability and resource utilization. We firstly present a state-of-the-art comparison of optical flow and disparity computation, with different technologies and algorithms. Then, we present a complete benchmark for both computations including the analysis of the accuracy and the qualitative results. Finally we also present the summary of resource utilization.

As mentioned before, this development is related to previous works [41] [28]. In fact, our architecture is presented as a cost-optimized version of the one presented in [28]. They perform the visual processing using phase-based algorithms, which are more accurate and robust against illumination changes. Our work is developed using gradient-based algorithms which are efficient concerning resources and computational load. Note that although the high-level block description of the different modules is similar (the same features are extracted), the goal of reducing resource utilization and power has motivated to completely re-design the different modules. We also present a comparison of the two systems.

4.1. Performance analysis

As proposed in the Introduction section our objective is the implementation of a high performance system which means, a high frame rate to work in real time. In Table 1, our system with an image resolution of 640 x 480 pixels, reaches up to 32 fps, which is significantly higher than the commonly accepted frame rate for real-time performance (25 fps).

Table 1 and Table 2 show performance comparisons of our work and previous ones. We compare our developments with the last works in the literature. The multi-scale values are empirically obtained. The mono-scale approaches would obtain about 270 and 350 fps for optical flow and disparity respectively (for VGA resolution).

For the optical flow, Table 1 lists the performance of works using different technologies: FPGAs, CPUs, GPUs or even, CELL processors. Selecting the works with the same technology (FPGA), our results are comparable to the approach described in [28]. They are better than the rest of the cases considering that the other works present mono-scale versions. We also highlight the CPU version [22] with an effective frequency of 90 MHz at a 1280x1026 resolution but, with a mono-scale version of the Lucas&Kanade algorithm. This table also shows the AAE and density values for the different implementations for the “Yosemite” sequence. Our work achieves a good position in the proposed ranking, being the most accurate among the FPGA implementations. We have also added the power consumption (w), that in our case is estimated by using the Xilinx XPower tool [37]. To estimate the power consumption, especially the dynamic device power consumption, we set the toggle rate to 20%, estimating for our design as for the worst-case taking into account a logic-intensive design.

Table 2 shows the performance comparison for the disparity computation. The works are all mono-scale developments except the one described in [28]. We observe important differences in the case of the mono-

	Resolution	Frame rate (fps)	MPPS	AAE (°)	Dens (%)	Power (w)	Resources (slices)	Architecture	Algorithm
Our work work	640x480	31.91	9.80	4.55	58.50	3.4	32442	Xilinx V4 (44 MHz)	LK
Tomasi (2012) [28]	640x480	31.5	9.68	7.91	92.01	7.2	42072	Xilinx V4 (45 MHz)	Phase-based multiscale
Botella (2010) [42]	128x96	16	0.20	5.5	100	NP	≈ 19000	Xilinx V2	Multi-channel Gradient
Anguita (2009) [22]	1280x1026	68.5	89.96	3.79	71.8	NP	NP	Quad2 Core (2830 MHz)	LK
Gwosdek (2009) [43]	316x252	210	16.72	5.73	NP	NP	NP	Cell Processor (PS3)	Variational
Pauwels (2008) [20]	640x512	48.5	15.89	2.09	63	NP	NP	GeForce 8800 GTX	Phase-based
Diaz (2008) [8]	800x600	170	81.6	7.86	57.2	NP	≈ 9200	Xilinx V2 (82 MHz)	LK
Chase (2008) [44]	640x480	64	19.66	12.9	NP	NP	NP	Xilinx V2 (187 MHz)	Tensor-based
Chase (2008) [44]	640x480	150	46.08	12.9	NP	NP	≈ 250	GeForce 8800 GTX	Tensor-based
Wei (2007) [45] [45]	640x480	64	19.66	12.7	NP	2	10288	Xilinx V2 (100 MHz)	Tensor-based
Bruhn (2006) [46]	160x120	63	1.21	5.77	100	NP	NP	Intel P. IV (3.06 GHz)	Variational

Table 1: Optical flow performance comparison with previous works (sorted by date of publication). The AAE and Density values are computed for the “Yosemite” sequence.

	Resolution	Frame rate (fps)	PDS (x 10^6)	Power (w)	Resources (slices)	Architecture	Algorithm
Our multi-scale work	640x480	31.91	1254	3.4	32442	Xilinx V4 (44 MHz)	LK multi
Tomasi (2012) [28]	512x512	28	939	7.2	42072	Xilinx V4 (42 MHz)	Phase-based
Villalpando (2011) [47]	1024x768	12	132	NP	49924	Xilinx V4 (66 MHz)	Color SAD
Chen (2011) [48]	320x240	30	148	7	26138	Xilinx V4 (60 MHz)	Color SAD
Calderon (2010) [49]	288x352	142	2534	NP	≈ 1800	Xilinx V2 Pro (174.2 MHz)	BSAD
Hadjitheofanous (2010) [50]	320x240	75	184	NP	12560	Xilinx V2 Pro	SAD
Georgoulas (2009) [51]	800x600	550	21120	NP	15442	Stratix IV (511 MHz)	SAD
Ernst (2009) [52]	640x480	4.2	165	NP	NP	GeForce 8800	SGM
Ambrosch (2009) [53]	450x375	600	10125	NP	22400	Stratix II (110MHz)	SAD
Gibson (2008) [54]	450x375	6	65	NP	NP	G80 NVIDIA	SGM
Diaz (2006) [24]	1280x960	52	1885	NP	≈ 19200	Xilinx V2 (65 MHz)	Phase-based

Table 2: Disparity performance comparison with previous works (sorted by date of publication).

scale versions and similar performance for the multi-scale one. The algorithms based on SAD (Sum of Absolute Differences) achieve good performance, here in terms of frame rate, but with the drawbacks of their lack of accuracy [55] and robustness [56]. The table also shows the PDS (Points x Disparity measure per Seconds) for the different works proposed and it is important from this point of view to remark some works [51] and [53] with the highest PDS values. We achieve in our case 1254 millions PDS for our multi-scale system.

4.2. Accuracy measurement

As explained, the multi-scale extension of the basic algorithms allows us to compute wide-range estimations. The advantages of the mono-scale version are obvious, it achieves the best frame rate results compared with the recent works. The multi-scale version does not achieve the same performance in terms of frame rates, but the accuracy in this case is significantly better. Table 3 shows the precision analysis for the Marble sequence [57] (also called in this work $b6of$). In this table we observe the evolution of the error (Average Angular Error, Standard Deviation of the Angular Error and Density) along the scales. We have also included the weighted RMS for a weighted Angular Error: $weightedAAE = AAE/Density$. This error takes into account the disparity to fairly compare results with different densities. This value is useless for comparing results changing the algorithm or the benchmark, consequently is not included in the rest of the tables. The first row shows the accuracy of the mono-scale version for software and hardware versions. The improvement for the error measure comparing the mono and the multi-scale versions is a factor of 3 (3x) maintaining the same density. Furthermore, the comparison between hardware and software versions shows a slight loss of density (almost 9%) in the hardware version, due to the use of fixed-point arithmetic, without considerable loss of accuracy (0.83° less). This fact is clearer focusing on the weighted RMS error (wRMS). The increase in the SAE and the AAE since the fourth and fifth scales is due to the image resolution and the maximum number of scales for the computation. In this case, with images of 512 by 512, using 5 scales means that the coarsest scale resolution is 16 by 16. The use of a maximum of 4 or 5 scales is justified by the use of filters of 5 by 5, and taking into account the size of the objects in the sequence and the issues estimating in the image borders. A low-accurate estimation in the first scale (the coarsest) represents a strong problem in the final estimation if the refinment process cannot reduce its impact.

	Software results				Hardware results			
No. of scales	AAE	SAE	Dens.	wRMS	AAE	SAE	Dens.	wRMS
1 (mono-scale)	29.14	24.31	80.90	43.46	23.08	22.23	71.51	39.19
2	13.39	15.38	79.80	22.76	11.12	14.41	70.69	21.33
3	9.64	12.26	79.63	17.22	9.73	13.03	70.68	18.95
4	9.62	13.08	79.49	17.82	9.48	12.41	70.58	18.28
5	9.16	11.46	79.48	16.25	9.99	13.07	70.56	19.26

Table 3: Comparison of the software and hardware versions for the "*Marble*" sequence [57](also called in this paper $b6of$) using different number of scales. We analyze AAE, SAE and Density.

Fig. 5 and Fig. 6 show the benchmark results for the disparity and optical flow computations. The benchmark sequences are available at the Middlebury website [58], they also provide the ground-truth and comparatives for all them.

Fig. 5 shows the disparity results for the "*Tsukuba*", "*Sawtooth*", "*Venus*", "*Teddy*" and "*Cones*" sequences (marked as $b1d$ to $b5d$ respectively). We show (from left to right) the original left image, the real disparity and the hardware results. The values in black in the hardware column are the unreliable values (NaN). Fig. 6 shows the optical flow results for the "*Yosemite*", "*Diverging tree*", "*Translation tree*", "*Rubberwhale*", "*Hydrangea*" and "*Otte Marble*" (indicated as $b1of$ to $b6of$ respectively). From left to right, we show one original frame of the sequence (the central one), the ground-truth values and the hardware estimation (showing with arrows the direction and module of the flow). These two figures show the qualitative results of the estimations. Table 4 and Table 5 show the values of the error measures for the benchmark sequences with the refined hardware implementation. In both cases, we are discarding estimates in the borders of the images due to the lack of information, considering them non-confident estimates. For sequences with a resolution lower than 320x240, we discard 10 rows or columns for each border. For higher resolutions we discard 15 rows or columns. Fig. 7 shows the local contrast descriptor results for the circle image (the left image). We show the local energy (center) and the local orientation (right). Table 6 shows the local feature performance using the MAE and SAE for energy and the AAE and SAE for the orientation. We test the average error evaluating 500 randomly selected points.

The subsequent figures (Fig 8 and Fig 9) illustrate the analysis of the error of the sequences of Fig 5 and Fig 6. For the optical flow we analyze the AAE (Average Angular Error), the SAE (Standard Deviation of the Angular Error) and Density. For the disparity we use the MAE (Mean Absolute Error), SAE (Standard Deviation of the Absolute Error) and Density. The SAE in

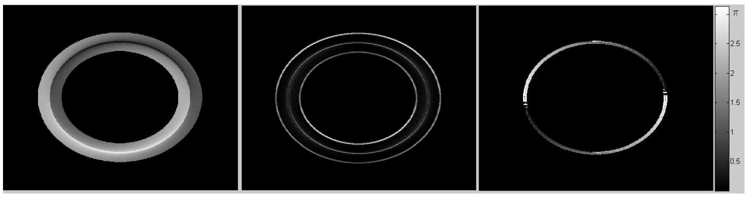


Figure 7: Local contrast results. From left to right: original circle image, energy and local orientation [0,π].

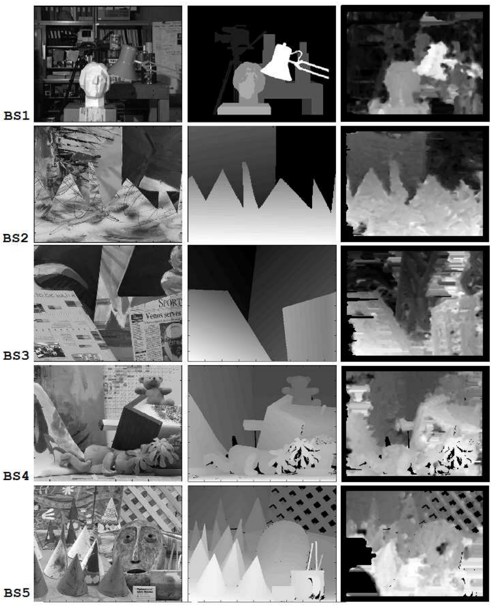


Figure 5: Disparity benchmark results. From left to right: original image, real disparity, hardware disparity results.

Sequence		MAE (pix)	SAE (pix)	Dens (%)	RMS
*b*1*d*	Tsukuba	0.88	1.32	99.60	1.58
*b*2*d*	Sawtooth	1.10	1.89	82.51	2.19
*b*3*d*	Venus	1.22	1.50	81.20	1.94
*b*4*d*	Teddy	2.94	4.25	83.71	5.16
*b*5*d*	Cones	3.43	5.28	71.49	6.30

Table 4: Disparity performance measures for the benchmark sequences, including the MAE, SAE, Density and RMS.

Sequence		AAE (°)	SAE (°)	Dens (%)
*b*1*of*	Yosemite	4.42	4.12	71.00
*b*2*of*	Diverging tree	4.83	4.25	93.78
*b*3*of*	Translation tree	2.24	2.08	88.18
*b*4*of*	Rubber whale	15.31	23.18	33.26
*b*5*of*	Hydrangea	15.13	17.89	70.08
*b*6*of*	Otte Marble	7.76	10.24	79.80

Table 5: Optical flow performance measures for the benchmark sequences, including the AAE, SAE and Density.

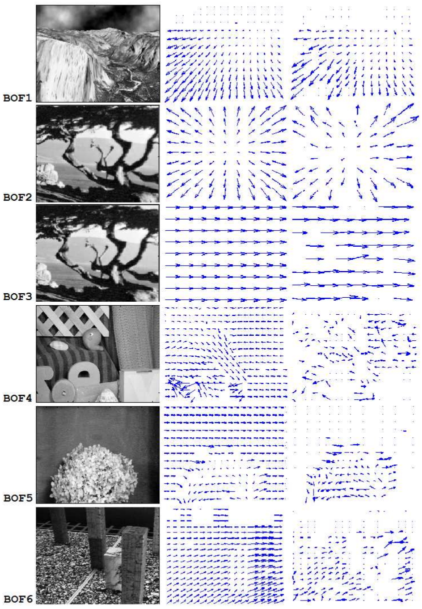


Figure 6: Optical flow benchmark results. From left to right: original image, optical flow ground-truth, hardware optical flow results.

Energy		Orientation	
MAE (pixel)	SAE (pixel)	AAE (°)	SAE (°)
0.12	1.19	4.15	13.49

Table 6: Local contrast features performance measures for the benchmark circle image.

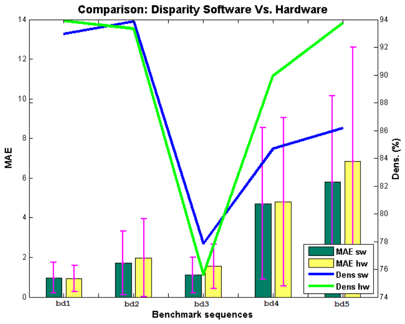


Figure 8: Disparity hardware Vs. software comparison (MAE, SAE, density).

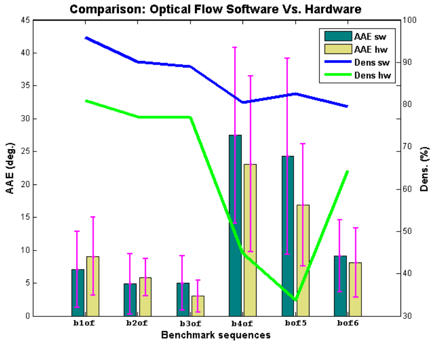


Figure 9: Optical flow hardware Vs. software comparison (AAE, SAE, density).

both cases is shown using error lines over the AAE and MAE bars. Our implementation is sparse, however, the obtained density is higher than other similar or mono-scale approaches. All these error metrics are defined in Section 2.

Fig. 8 shows the disparity error comparison between the hardware and the software versions for the sequences displayed previously in Fig. 5 ($b1d$ to $b5d$). As it can be seen, the MAE of hardware is always slightly higher than the software but with the same stability and with similar densities. The decrease in density in $b3d$ is due to the structureless sequence, and for $b4d$ and $b5d$ to the occlusions. For the "*Tsukuba*" images ($b1d$) the MAE is of 0.93 pixels with 93.89% of density and SAE of 1.37 pixels.

Fig. 9 shows the optical flow error for the hardware vs. software comparison the sequences displayed in Fig. 6 ($b1of$ to $b6of$). The differences between hardware and software versions are not relevant except for the $b4of$ and $b5of$ sequences (but mainly due to the modification of density). On the other hand, the AAE of the hardware versions of both cases is about 4° and 7° less than the software versions respectively (at lower density). The loss of density in $b5of$ is due to the lack of texture of the sequence and for $b4of$, due to the complexity (particularly the occlusions) of the real scene and the range of the movements. For the "*Yosemite*" sequence ($b1of$) the AAE is of 9.08°. with 80.94% of density and SAE of 12.4°.

4.3. Hardware resource utilization

In computer vision, the selection of the best alternative is done depending on the target application, as a good trade-off between the required accuracy and the constraints about the maximum frequency and the resource utilization. Table 7 shows the resource utilization of the implemented system. This table presents the information about FPGA resource utilization: total number of 4 input LUTs, Slice Flip Flops, Slices, DSPs, Block RAMs used and finally, the maximum frequency (MHz) allowed in each module (the speed grade of our FPGA is -10). In addition to this, as mentioned in the Introduction, one of the specially interesting aspects of this Section is also the comparison in resources between our gradient-based and the phase-based ([28]) approaches.

The listed options include: the multi-scale optical flow system and the optical flow core, the multi-scale disparity system and the disparity core, and finally the complete system. We also present the resources for the most important parts of the multi-scale extension: the interface with the resources of the selected board and the MCU, the over-sampling module, the warping module and the merging module. In the case of the complete system, the optical flow core also includes the computation of the local contrast descriptors (energy and orientation). As shown, warping module has the highest resource utilization of the multi-scale implementation modules (it is also the module with the highest computational load) with 23% of the total available resources.

For the evaluation of the resource utilization we also show Fig. 10. This figure shows three main sets of columns: the percentage of total slices used, of internal DSPs and of internal Block RAMs. For each set, we have six columns grouped in pairs. From left to right, they present the utilization of resources for the disparity core, the optical flow core and the complete system, comparing the resource utilization of [28] and our resource utilization (first and second column respectively for each measure). In fact, considering the total slices, for the disparity core the saving is about 80%, 75% in the case of the optical flow and about 25% for the complete system. This last saving is less considerable because the basic multi-scale architecture of both

	4 input LUTs	Slice Flip-Flops	Slices	DSP	Block RAM	Freq
	(out of 84352)	(out of 84352)	(out of 42716)	(160)	(378)	(MHz)
OF system	31796 (37%)	24694 (29%)	26036 (61%)	62 (38%)	112 (29%)	44
OF core	4589 (5%)	6622 (5%)	4128 (9%)	30 (18%)	48 (12%)	83
Disp system	21287 (25%)	16989 (20%)	18773 (44%)	6 (3%)	92 (24%)	49
Disp core	2358 (2%)	3120 (3%)	2129 (5%)	3 (1%)	28 (7%)	108
Complete system	**41613(49%)**	**32706 (38%)**	**32442 (76%)**	**49 (30%)**	**162 (43%)**	**44**
Board Interface	4774 (5%)	5195 (6%)	5388 (12%)	0	36 (9%)	112
Scaling	413 (1%)	270 (1%)	367 (1%)	0	1 (1%)	86
Warping + Int.	9943 (11%)	9097 (10%)	9894 (23%)	32 (20%)	43 (11%)	51
Merging	364 (1%)	244 (1%)	235 (1%)	0	4 (1%)	107

Table 7: Hardware resource utilization for the presented complete architecture using a Virtex-4 FX100 FPGA.

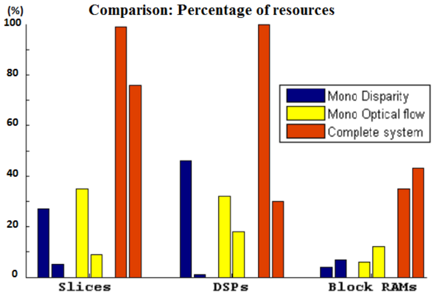


Figure 10: Resource utilization comparison between the gradient-based (here described) and the phase-based ([28]) implementation for the disparity core, the optical flow core and the complete system. The first column at each group represents the resources for the phase-based implementation and the second column for the gradient-based one.

systems is very similar. Moreover, the total architecture uses about 40% of the total resources.

On the other hand, we use a bigger amount of Block RAMs for our computation but much less DSPs because our filter stages are not as computationally heavy as the Gabor filters used in the case of [28].

Finally, we also present Table 8 that validates our hypotheses. We compute a figure of merit (FOM) to compare the phase-based and the gradient-based approaches for the estimation of optical flow and disparity. This FOM is computed as: $(accuracy \times stability \times throughput)/resources$. Accuracy is computed as the inverse of the error (AAE or MAE for optical flow and disparity respectively) weighted by the density, as in: $1/(Error/Density)$; similarly, stability is estimated as the inverse of the standard deviation (SAE in both cases). Using this merit metrics, as shown in the table, the gradient-based FOM is approximately 4x better (higher) than the phase-based approach.

5. Conclusions

In this work we have designed and implemented an embedded system for a complete low-level vision processing engine. We implement optical flow, disparity, local energy, and orientation estimations. Among the numerous alternatives, we selected a multi-scale implementation of the well-known L&K method due to its good trade-off between accuracy and efficiency. In fact, we have implemented three systems: one for the multi-scale optical flow estimation, one for the multi-scale disparity estimation and finally another one for the complete low-level vision engine.

This work was proposed as a real-time low-level vision platform. As explained, our approach reaches about 32 fps for the multi-scale estimation of optical flow and disparity, also including the computation of local energy and orientation. The optical flow core individually achieves up to 270 fps and the disparity core about 350 fps for a 640x480 resolution. This performance fulfill the real-time requirements. Moreover, the accuracy and stability results are similar to the resolutions implemented with the same technology in the literature. This performance are obtained using fine-grain pipeline implementation strategy. It means the massive parallelization through the pipelining of the complete implementation using about 400 processing units (taking into account only the computation of the cores). It also means a low power consumption [8], which is essential for example in the embedded systems industrial field. The last advantage of our design strategy is the customization of the system considering the requirements and the target application, adopting choices as the modification of the number of pipeline stages, the superscalar units or even, the inclusion of new visual primitive estimation cores.

The presented system adopts a multi-scale strategy which is seldom implemented in FPGAs (only [28] is previously described). This expands significantly the working range of the system. The multi-scale computation is usually avoided in FPGA implementations due to the complexity of the inter-scale warping operation. Besides its complex design described in Section 3.4, it requires high resource utilization as indicated in Table 7.

	Accuracy x Stability	Accuracy x Stability	Throughput	Resources	FOM
	Disp. (pix^{-2})	O. Flow (deg^{-2})	($Mpix/s$)	(slices)	($(deg^{-2}pix^{-2}Mpix/s)/slices$)
Phase-based	0.8416	0.0122	46.4	42072	$1.13x10^{-5}$
Gradient-based	0.8574	0.0390	44	32442	$4.53x10^{-5}$

Table 8: Figure-of-merit computation for phase-based and gradient-based estimations of optical flow and disparity. The resources are estimated using the same Virtex4 XC4vfx100 chip.

Nevertheless, its accuracy improvement has been evaluated in this paper including an explicit comparison with mono-scale engines using standard benchmark images and sequences.

As analyzed, a hardware system is also characterized by the resource utilization. In our case, for the estimation of the visual primitives we use about 15% of the total resources of a Xilinx Virtex4 XC4vfx100 chip. Comparing with a similar work based on phase models [28], our gradient-based implementation achieves a resource saving of 80% for the disparity core and 75% for the optical flow core. The total system, including the multi-scale implementation and the interface with the MCU, saves about 25%. The total amount of resources that the system uses is 32442 slices, 169 BRAMs, 49 DSPs, and it achieves a maximum clock frequency of 44 MHz. This saving allows us to provide the system as a cost-optimized (though still multiscale) vision processing engine for dynamic environments.

The presented scheme and architecture represent a very powerful tool for the computation of low-level visual primitives required for the visual middle-level computation. Moreover, the development of the customizable architecture allows us the adaptation of the system, a very interesting feature for a computation system if we want to close the action-perception cycle in active vision systems.

Acknowledgments.

This work has been supported by the EU grant TOMSY (FP7-270436), the Spanish projects DINAM-VISION (DPI2007-61683), ARC-VISION (TEC2010-15396), the Spanish Grant (AP2007-00275) and the regional projects MULTIVISION (TIC-3873) and ITREBA (TIC-5060).

References

[1] P. C. Merrell, D. Lee, Structure from motion using optical flow probability distributions, in: K. L. Priddy (Ed.), Intelligent Computing: Theory and Applications III, volume 5803, SPIE, 2005, pp. 39 – 48.

[2] S. Rougeaux, Y. Kuniyoshi, Velocity and disparity cues for robust Real-Time binocular tracking, in: Computer Vision and Pattern Recognition, IEEE Computer Society Conference on, IEEE Computer Society, 1997.

[3] W. Zhang, Q. J. Wu, H. bing Yin, Moving vehicles detection based on adaptive motion histogram, Digital Signal Processing 20 (2010) 793 – 805.

[4] R. Marzotto, P. Zoratti, D. Bagni, A. Colombari, V. Murino, A real-time versatile roadway path extraction and tracking on an fpga platform, Computer Vision and Image Understanding 114 (2010) 1164 – 1179.

[5] A. Giachetti, M. Campani, V. Torre, The use of optical flow for the autonomous navigation, in: Proceedings of the third European conference on Computer vision, volume 1 of *ECCV '94*, Springer-Verlag New York, Inc., 1994, pp. 146 – 151.

[6] J. Aloimonos, I. Weiss, A. Bandyopadhyay, Active vision, Int Journal of Computer Vision 1 (1988) 333 – 356.

[7] A. Wali, A. M. Alimi, Event detection from video surveillance data based on optical flow histogram and high-level feature extraction, in: Database and Expert Systems Application, 20th International Workshop on, pp. 221 – 225.

[8] J. Diaz, E. Ros, R. Agis, J. Bernier, Superpipelined high-performance optical-flow computation architecture, Computer Vision and Image Understanding 112 (2008) 262 – 273.

[9] M. Cvikl, A. Zemva, Fpga-oriented hw/sw implementation of ecg beat detection and classification algorithm, Digital Signal Processing 20 (2010) 238 – 248.

[10] J. Bigun, G. Granlund, J. Wiklund, Multidimensional orientation estimation with applications to texture analysis and optical flow, Pattern Analysis and Machine Intelligence, IEEE Transactions on 13 (1991) 775 – 790.

[11] M.-K. Hu, Visual pattern recognition by moment invariants, Information Theory, IRE Transactions on 8 (1962) 179 – 187.

[12] M. Brown, D. Lowe, Unsupervised 3d object recognition and reconstruction in unordered datasets, in: 3-D Digital Imaging and Modeling, 2005. 3DIM 2005. Fifth International Conference on, pp. 56 – 63.

[13] D. G. Lowe, Object recognition from local scale-invariant features, Computer Vision, IEEE International Conference on 2 (1999) 1150.

[14] H. Bay, A. Ess, T. Tuytelaars, L. V. Gool, Speeded-up robust features (surf), Computer Vision and Image Understanding 110 (2008) 346 – 359.

[15] D. G. Lowe, Distinctive image features from scale-invariant keypoints, Int. J. Comput. Vision 60 (2004) 91 – 110.

[16] S. Baker, I. Matthews, Lucas-Kanade 20 Years On: A Unifying Framework: Part 1, Technical Report, Robotics Institute, 2002.

[17] B. D. Lucas, T. Kanade, An iterative image registration technique with an application to stereo vision (1981) 674 – 679.

[18] B. K. P. Horn, B. G. Schunck, Determining optical flow, Artificial Intelligence 17 (1981) 185 – 203.

[19] L. Chen, Y. Jia, A parallel reconfigurable architecture for Real-Time stereo vision, in: Embedded Software and Systems, 2009. ICESS '09. International Conference on, pp. 32 – 39.

[20] K. Pauwels, M. M. V. Hulle, Realtime phase-based optical flow on the GPU, in: Computer Vision and Pattern Recognition Workshops, CVPRW '08. IEEE Computer Society Conference on, pp. 1 – 8.

[21] K. Pauwels, M. Tomasi, J. Diaz Alonso, E. Ros, M. Van Hulle, A comparison of fpga and gpu for real-time phase-based optical

flow, stereo, and local image features, Computers, IEEE Transactions on In Press (2011).

[22] M. Anguita, J. Diaz, E. Ros, F. J. Fernandez-Baldomero, Optimization strategies for High-Performance computing of Optical-Flow in General-Purpose processors, Circuits and Systems for Video Technology, IEEE Transactions on 19 (2009) 1475 – 1488.

[23] T. Kohlberger, C. Schnorr, A. Bruhn, J. Weickert, Domain decomposition for variational optical-flow computation, Image Processing, IEEE Transactions on 14 (2005) 1125 – 1137.

[24] J. Diaz, E. Ros, F. Pelayo, E. M. Ortigosa, S. Mota, FPGA-based real-time optical-flow system, Circuits and Systems for Video Technology, IEEE Transactions on 16 (2006) 274 – 279.

[25] J. L. Barron, D. J. Fleet, S. S. Beauchemin, Performance of optical flow techniques, Int Journal of Computer Vision 12 (1994) 43 – 77.

[26] S. S. Beauchemin, J. L. Barron, The computation of optical flow, ACM Computing Surveys 27 (1995) 433 – 466.

[27] J. Bergen, P. Anandan, K. Hanna, R. Hingorani, Hierarchical model-based motion estimation, in: G. Sandini (Ed.), Computer Vision ECCV'92, volume 588 of *Lecture Notes in Computer Science*, pp. 237 – 252.

[28] M. Tomasi, M. Vanegas, F. Barranco, J. Daz, E. Ros, Massive parallel-hardware architecture for multiscale stereo, optical flow and image-structure computation, Circuits and Systems for Video Technology, IEEE Transactions on 22 (2012) 282 – 294.

[29] E. P. Simoncelli, Design of multi-dimensional derivative filters, in: Image Processing, 1994. Proceedings. ICIP-94., IEEE International Conference, volume 1, pp. 790 – 794.

[30] H. Liu, T.-H. Hong, M. Herman, R. Chellappa, Accuracy vs. efficiency trade-offs in optical flow algorithms, in: Computer Vision ECCV 96, volume 1065 of *Lecture Notes in Computer Science*, Springer Berlin, Heidelberg, 1996, pp. 174 – 183.

[31] B. McCane, K. Novins, D. Crannitch, B. Galvin, On benchmarking optical flow, Computer Vision and Image Understanding 84 (2001) 126 – 143.

[32] J. W. Brandt, Improved accuracy in Gradient-Based optical flow estimation, Int. J. Comput. Vision 25 (1997) 5 – 22.

[33] S. Jin, D. Kim, D. D. Nguyen, J. W. Jeon, Pipelined hardware architecture for high-speed optical flow estimation using fpga, in: Field-Programmable Custom Computing Machines (FCCM), 2010 18th IEEE Annual International Symposium on, pp. 33 – 36.

[34] S. Lim, A. El Gamal, Optical flow estimation using high frame rate sequences, in: Image Processing, 2001. Proceedings. 2001 International Conference on, volume 2, pp. 925 – 928.

[35] M. Tomasi, F. Barranco, M. Vanegas, J. Diaz, E. Ros, Fine grain pipeline architecture for high performance phase-based optical flow computation, Journal of Systems Architecture 56 (2010) 577 – 587.

[36] P. J. Burt, Edward, E. H. Adelson, The laplacian pyramid as a compact image code, IEEE Transactions on Communications 31 (1983) 532 – 540.

[37] Xilinx, FPGA and CPLD solutions from Xilinx, Inc., 2010.

[38] E. Ortigosa, A. Canas, E. Ros, P. Ortigosa, S. Mota, J. Diaz, Hardware description of multi-layer perceptrons with different abstraction levels, Microprocessors and Microsystems 30 (2006) 435 – 444.

[39] M. Vanegas, M. Tomasi, J. Diaz, E. Ros, Multi-port abstraction layer for FPGA intensive memory exploitation applications, Journal of Systems Architecture 56 (2010) 442 – 451.

[40] J. Diaz, Multimodal bio-inspired vision system. High performance motion and stereo processing architecture, Ph.D. thesis, Universidad de Granada, 2006.

[41] M. Tomasi, M. Vanegas, F. Barranco, J. Diaz, E. Ros, High-Performance Optical-Flow architecture based on a multiscale, Multi-Orientation Phase-Based model, Circuits and Systems for Video Technology, IEEE Transactions on 20 (2010) 1797 – 1807.

[42] G. Botella, A. Garcia, M. Rodriguez-Alvarez, E. Ros, U. Meyer-Baese, M. C. Molina, Robust bioinspired architecture for optical-flow computation, IEEE Trans. Very Large Scale Integr. Syst. 18 (2010) 616 – 629.

[43] P. Gwosdek, A. Bruhn, J. Weickert, Variational optic flow on the sony playstation 3, Journal of Real-Time Image Processing 5 (2010) 163 – 177.

[44] J. Chase, B. Nelson, J. Bodily, Z. Wei, D.-J. Lee, Real-time optical flow calculations on fpga and gpu architectures: A comparison study, in: Field-Programmable Custom Computing Machines. FCCM '08. 16th International Symposium on, pp. 173 – 182.

[45] M. Martineau, Z. Wei, D.-J. Lee, M. Martineau, A fast and accurate tensor-based optical flow algorithm implemented in fpga, in: Applications of Computer Vision, 2007. WACV '07. IEEE Workshop on, pp. 18 – 18.

[46] A. Bruhn, J. Weickert, T. Kohlberger, C. Schnrr, A multigrid platform for Real-Time motion computation with Discontinuity-Preserving variational methods, Int. J. Comput. Vision 70 (2006) 257 – 277.

[47] C. Villalpando, A. Morfopolous, L. Matthies, S. Goldberg, Fpga implementation of stereo disparity with high throughput for mobility applications, in: Aerospace Conference, 2011 IEEE.

[48] L. Chen, Y. Jia, M. Li, An fpga-based rgbd imager, Machine Vision and Applications (2011) 1 – 13.

[49] H. Calderon, J. Ortiz, J. Fontaine, High parallel disparity map computing on FPGA, in: Mechatronics and Embedded Systems and Applications (MESA), 2010 IEEE/ASME International Conference on, pp. 307 – 312.

[50] S. Hadjitheophanous, C. Ttofis, A. Georghiades, T. Theocharides, Towards hardware stereoscopic 3D reconstruction a real-time FPGA computation of the disparity map, in: Design, Automation & Test in Europe Conference & Exhibition (DATE), 2010, pp. 1743 – 1748.

[51] C. Georgoulas, I. Andreadis, A real-time occlusion aware hardware structure for disparity map computation, in: Image Analysis and Processing ICIAP 2009, volume 5716, Springer Berlin, 2009, pp. 721 – 730.

[52] I. Ernst, H. Hirschmüller, Mutual information based semi-global stereo matching on the gpu, in: Proceedings of the 4th International Symposium on Advances in Visual Computing, Springer-Verlag, 2008, pp. 228 – 239.

[53] K. Ambrosch, M. Humenberger, W. Kubinger, A. Steininger, Sad-based stereo matching using fpgas, in: Embedded Computer Vision, Advances in Pattern Recognition, Springer London, 2009, pp. 121 – 138.

[54] J. Gibson, O. Marques, Stereo depth with a unified architecture gpu, in: Computer Vision and Pattern Recognition Workshops, 2008. CVPRW '08. IEEE Computer Society Conference on, pp. 1 – 6.

[55] D. Scharstein, R. Szeliski, A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms, International Journal of Computer Vision 47 (2002) 7 – 42.

[56] J. Banks, P. Corke, Quantitative evaluation of matching methods and validity measures for stereo vision, The International Journal of Robotics Research 20 (2001) 512 – 532.

[57] M. Otte, H.-H. Nagel, Optical flow estimation: advances and comparisons, in: Proceedings of the third European conference on Computer vision, ECCV '94, Springer-Verlag New York, Inc., 1994, pp. 51 – 60.

[58] M. C. Vision, Middlebury computer vision, 2010.

2.3 Color-based architecture for motion and depth estimation

The journal article associated to this part of the dissertation is:

- F. Barranco, M. Tomasi, J. Díaz, E. Ros, Hierarchical architecture for motion and depth estimations based on color cues. **Submitted to J. of Real-Time Image Processing**.
    - Status: **Submitted**.
    - Impact Factor (JCR 2009): 0.962
    - Subject Category:
        - ∗ Computer Science, Artificial Intelligence. Ranking 72 / 108.
        - ∗ Engineering, electrical and electronic. Ranking 126 / 247.
        - ∗ Image Science and Photographic Technology. Ranking 8 / 19.

Journal of Real-Time Image Processing manuscript No.
(will be inserted by the editor)

Francisco Barranco · Matteo Tomasi · Mauricio Vanegas · Javier Diaz · Sara Granados · Eduardo Ros

Hierarchical Architecture for Motion and Depth Estimations based on Color Cues

Received: date / Revised: date

Abstract This work presents an FPGA implementation of a highly parallel architecture for the motion and disparity estimations of color images. Our system implements the well-known Lucas & Kanade algorithm with multi-scale extension for the computation of large displacements using color cues. We empirically fulfill the real-time requirements computing up to 32 and 36 frames per second for optical flow and disparity respectively, with a 640x480 resolution. In this paper, we present our design technique based on fine pipelines, our architecture, and benchmarks of the different color-based alternatives analyzing the accuracy and resources utilization trade-off. We finally include some qualitative results and the resource utilization for our platform, concluding that we have obtained a system that manages a good trade-off between the increase in resources and the improvement in precision and the density of our results compared with other approaches described in the literature.

1 Introduction

Depth and motion estimation are very well known image registration problems. This problem consists in finding the image correspondence of the object presented in two or more images acquired at different spatial positions, time instants, or scenarios.

Depth is perceived as the disparity between the projections of the real image on the binocular system. There are three steps involved in this processing: the selection of the target in one image of the obtained pair, the search of the correspondent area in the other image, and the disparity computation between them. The second one is the most difficult step because of the matching ambiguity. Motion computation estimates a two-dimensional motion field of the scene at different instants and therefore, requires similar operations using several images from the same camera. Binocular disparity performs a one-dimensional search problem thanks to the epipolar geometry constraint and its main issues are related with the use of different views of the same scene (or different sensors that add artifacts such as different illumination, sensitivity, affine deformations, etc). In the case of optical flow, it is a two-dimensional search problem across different time instants but if the frame-rate is high enough, small displacements could be assumed. In both cases, large spatial displacements, occlusions, and illumination mismatching are the most frequent encountered problems.

Motion and depth estimation has been extensively studied in the literature but nowadays, their computation in an efficient and accurate way is still an open issue. Intensity-based estimation is one of the most used approaches for their computation by assuming the constancy of the intensity. This approach is similar to the computation performed at the visual cortex of superior mammals as cats or primates. The drawback is that the intensity-based approach has a strong ambiguity that could be partially solved up by using color information that also adds more stability to the results. There are plenty of works in the literature concerning this topic. With respect to the color-based optical flow, some authors worked initially with multi-spectral images (Markandey and Flinchbaugh (1990)); Ohta (1989) proposed a computation with the color cues similar to the computation that uses gradient-based information; Golland and Bruckstein (1997) proposed the color conservation assumption instead of the intensity conservation; Andrews and Lovell (2003) implemented some well-known gradient-based algorithms using color cues; and Barron and Klette (2002) evaluated their performances. For the color disparity computation, the first works considered only the correspondence-matching methods as in J.R. Jordan III

F. Barranco, M. Tomasi, J. Diaz, S. Granados, E. Ros
Dep. of Comp. Architecture and Technology, CITIC, Univ. of Granada, Daniel Saucedo sn, E18071, Granada, Spain
Tel.: +34958241775
E-mail: {fbarranco, mtomasi, jdiaz, sgranados, eduardo}@atc.ugr.es

M. Vanegas
Dep. of Biophysical and Electronic Engineering, Univ. of Genoa, Via Opera Pia 11A, I16145, Genoa, Italy
E-mail: mauricio.vanegas@unige.it

(1992) and Koschan et al (1996), or the edge-based color information (J.R. Jordan III (1991)). Some works explained that color information improves the signal to noise ratio about 25%-30% (Mühlmann et al (2002)). In fact, there are multiple applications for the color-based optical flow: motion estimation for underwater images (Negahdaripour and Madjidi (2003)), UAVs detection (Ortiz and Neogi (2006)), or image segmentation improvement (Denman et al (2009)). Besides, for the color-based disparity, some applications are real-time vision system for mobile robots (Matsumoto et al (1997)), surveillance (Krotosky and Trivedi (2008)), or people tracking (Muñoz Salinas et al (2007)).

Summarizing our ideas, we have implemented an integrated system for the optical flow and disparity computation based on gradient information, particularly the Lucas & Kanade (L&K) method described in Beauchemin and Barron (1995) and Lucas and Kanade (1981). We improve this model by adding color cues and extending it to a multi-scale implementation to increase the dynamic range of the estimations. Secondly, integrating that computation in autonomous navigation systems is one of our requirements (which also requires real-time performance). For all these reasons, we propose FPGA devices as the best candidate to exploit the high parallelism of the vision algorithms to achieve our high performance and a technology that enables the implementation of the color visual processing on-chip. We reach up to 36 fps for the disparity and 32 fps for the optical flow computation measured empirically with a real system. The theoretical rate according to the maximum working frequency of our designs would be 160 and 143 fps respectively, but from the system point of view, the maximum peak bandwidth of the port used to communicate the FPGA with the computer (PCIe in our case) has also to be taken into account.

This paper is structured as follows: Section 2 summarizes the L&K algorithms for optical flow and disparity and the multi-scale extension; Section 3 is focused on the benchmarking using the color information and its comparison with intensity-based, mono-scale, and multi-scale versions; Section 4 details the hardware implementation separately for each computation and for the multi-scale extension; Section 5 presents the system performance and the hardware resource utilization; finally, Section 6 presents the conclusions of the paper.

2 Color-Based Algorithms for Depth and Motion Estimation

Optical flow and disparity are computed using local and global methods. Local methods are centered in the surrounding neighborhood of a pixel to estimate its displacement value. Global methods take into account the complete image by using a diffusion process that globally propagates the local information across the whole image.

Our implementation uses local gradient-based techniques. They estimate binocular disparity by assuming the constancy of the color of each pixel between a pair of images (left and right frames) and the optical flow assuming the color constancy of each pixel for a sequence of frames (3 frames in our case). We avoid global methods due to their complexity for on-chip implementation (the resource cost is rather high), while local methods are selected due to their simplicity and efficiency (see Diaz et al (2008) and Liu et al (1996)). Local methods estimate very accurately for small differences or displacements. To improve the accuracy for high disparity or motion ranges, we implement the multiscale-with-warping extension.

2.1 Gradient-Color-based Optical Flow

The optical flow estimates the motion at each pixel with time, i.e. it consists in the estimation of the two-dimensional velocity as the projection of the real tridimensional velocity of the points onto the image plane. In our case, as mentioned before, we use the local gradient-based techniques for its computation.

For intensity-based implementations, the computation of the L&K optical flow is based on the brightness constancy assumption (also called *Optical Flow Constraint, OFC*). After applying Taylor expansion, in (1) optical flow is defined as the vector (u, v), claiming the constancy of intensity I along time.

$$\frac{\delta I}{\delta x}u + \frac{\delta I}{\delta y}v + \frac{\delta I}{\delta t} = 0 \tag{1}$$

Extending the OFC from Intensity to color channel values, we replicate it for each channel where F_c is the value of any channel of the color representation (2):

$$\frac{\delta F_c}{\delta x}u + \frac{\delta F_c}{\delta y}v + \frac{\delta F_c}{\delta t} = 0 \tag{2}$$

There are several works which tackle with the color optical flow computation. Golland and Bruckstein (1997) estimate the optical flow from a color representation with just two channels as in $R = \{F_1, F_2\}$, where R stands for the color representation, and F_1 and F_2 are the values for the color channels. These authors assume the color conservation assumption rather than the intensity one.

Moreover, we may save resources by taking just two color channels from the three-channel representations without losing significant information according with Golland and Bruckstein (1997), but the way in which we select those two channels is quite important. For instance, in the normalized RGB representation, we can select any two channels because they are represented by ratios and not intensity values. It can be explained due to the dependency between channels, which ensures that adding more than two channels just adds more redundant information to the system. On the other hand, there are many other different representations in which the information about the color is presented in two channels and the third one only represents the intensity (e.g. in the HSV representation, we take Hue and Saturation channels discarding the Value channel). Intensity channels are usually discarded because they are more prone to illumination problems, and consequently, to violating the OFC. In this case, the equations for our model simplified by using just two channels of any representation from (2) are presented in (3).

$$\frac{\delta F_1}{\delta x}u + \frac{\delta F_1}{\delta y}v + \frac{\delta F_1}{\delta t} = 0; \frac{\delta F_2}{\delta x}u + \frac{\delta F_2}{\delta y}v + \frac{\delta F_2}{\delta t} = 0 \quad (3)$$

It is worth noticing a difference with the monochrome approach: in (3), we have two equations for the u and v unknowns; consequently, it is a well-posed system. However, because of image noise and model hypothesis violations, it is not a good idea to follow strictly this approach. As an alternative, we integrate the information on larger image areas and follow a least-squares fitting technique as used on the original monochrome approach. This has the advantage of increasing the method robustness and improving the accuracy. We minimize (4) using 5x5 neighborhoods, applying standard least-square methods (see (5)), and solving it as is shown in (**??**) and (7). In the equations, $F_{c\xi i}$ stands for the F_c derivative with respect to ξ (x, y or t) at pixel i; W_i stands for the weights for the neighborhood Ω (weighting higher values close to the center). Cs is the set of the color channels c of our color representation R.

Finally, $A = [\nabla F_c(x_1, y_1), ..., \nabla F_c(x_n, y_n)]^T$, and $b = -(F_{ct}(x_1, y_1), ..., F_{ct}(x_n, y_n))^T$, for n points with $(x_i, y_i) \in \Omega$.

$$\sum_{i \in \Omega} W^2 (F_{cx}u + F_{cy}v + F_{ct})^2 \quad (4)$$

$$A^T W^2 A (u, v)^T = A^T W^2 b \quad (5)$$

$$\begin{bmatrix} \sum\limits_{c \in Cs} \sum\limits_{i \in \Omega} W_i^2 F_{cxi}^2 & \sum\limits_{c \in Cs} \sum\limits_{i \in \Omega} W_i^2 F_{cxi} F_{cyi} \\ \sum\limits_{c \in Cs} \sum\limits_{i \in \Omega} W_i^2 F_{cxi} F_{cyi} & \sum\limits_{c \in Cs} \sum\limits_{i \in \Omega} W_i^2 F_{cyi}^2 \end{bmatrix} (u, v)^T \quad (6)$$

$$= \begin{bmatrix} \sum\limits_{c \in Cs} \sum\limits_{i \in \Omega} W_i^2 F_{cxi} F_{cti} \\ \sum\limits_{c \in Cs} \sum\limits_{i \in \Omega} W_i^2 F_{cyi} F_{cti} \end{bmatrix} \quad (7)$$

2.2 Gradient-Color-based Disparity

Binocular disparity is defined as the difference between the left and right images and refers the depth perception. The same local gradient-based algorithm (Lucas & Kanade algorithm) as in the optical flow (Lucas and Kanade (1981)) was used for disparity. The assumption here is the color constancy of a pixel between left and right images. Applying the same assumption than in the optical flow, color disparity is defined as (8):

$$R_c(x) = L_c(x + \delta(x)) \quad (8)$$

where R_c and L_c represent a color channel of the right and left images respectively. The strategy for the resolution is the same tas the one performed in the case of the optical flow. From (8), we apply the Taylor expansion to obtain (9)

$$L_c(x + \delta) \approx L_c(x) + \delta L'_c(x) \quad (9)$$

and we solve (8) minimizing the error with respect to the disparity, achieving (10)

$$\delta = \frac{\sum_{c \in Cs} \sum_{i \in \Omega} W_i^2 L_{cxi}(R_{ci} - L_{ci})}{\sum_{c \in Cs} \sum_{i \in \Omega} W_i^2 L_{cxi}^2} \quad (10)$$

where we have added weights W_i which are the same as the ones in the optical flow computation. L_{cxi} stands for the partial derivative in x of the color channel c of the left image and R_{ci} and L_{ci} are the color value of channel c of the left and right image for pixel i in neighborhood Ω.

2.3 Hierarchical Extension: Multi-scale Implementation

Local algorithms for the estimation of optical flow and disparity work with small ranges because we use only the first term of the Taylor expansion as in (9) and because the derivative operator is approximated. The implementation of a multi-scale extension for the basic algorithms gives us the possibility of working with a range extended up to 30x (using 5 scales) with respect to the mono-scale versions, obtaining more accurate results for higher motion ranges. This approach is based on Bergen et al (1992) and also detailed in Tomasi et al (2010b), and it is shown in Fig. 1. A modular implementation of this extension is performed for the adaptation of the system to the estimation of disparity and motion:

- Gaussian pyramid computation: It computes the pyramid of the input images, initially performing a Gaussian smoothing filtering to reduce the aliasing effect due to the sampling process. In our case, this module computes a Laplacian pyramid (based on Burt et al (1983)) for each input frame (3 in the case of optical flow and two for the disparity) and for each color channel (we finally use 2 channels as shown in Section 3). The levels of the pyramid depend on the number of scales (typically, we use 5 scales for VGA resolution), with a down-sampling factor of 2.
- Motion or Disparity estimation: This stage computes the mono-scale motion or disparity. Our approach is performed in a coarse-to-fine way, which means that the first time, we estimate motion and disparity for the coarsest scale images of the precomputed pyramid and then, we will compute them for each scale using the refined results from the previous one.

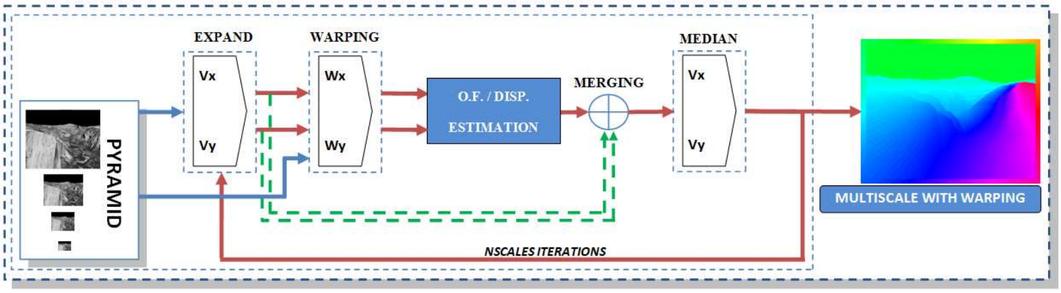


Fig. 1 Multiscale-with-warping algorithm scheme. The pyramid is the first module that is performed before the iteration through the different scales. The first iteration does not need the warping computation.

– Scaling: This stage up-samples the current scale results to the resolution of the subsequent finer scale (the upsampling factor is 2). This module is computed twice for motion (horizontal and vertical velocity estimations) and just once for disparity.
– Warping: This module is the most expensive one in terms of computational complexity. The warping stage varies depending on the visual primitive estimation. For optical flow, this stage consists in a two-dimensional interpolation (bilinear) keeping the central frame and warping the previous and the next frames with the precomputed estimation to compensate the motion presented in this image using previous motion estimation (from coarser scales). For disparity, it entails a one-dimensional interpolation due to the previous stage of undistortion and rectification that corrects the lens radial distortions and aligns the image planes with the epipolar lines, simplifying the disparity estimation only to its computation for the x axis. The undistortion and rectification values are computed once and off-line because they only depend on the camera set-up.

$$I_w^s(x,y) = Warp(I^s(x,y), \Theta(V^{s-1}(x,y))) \tag{11}$$

$$L_w^s(x) = Warp(L_w^s(x), \Theta(\delta^{s-1}(x))) \tag{12}$$

(11) and (12) show the warping computation expressions for optical flow and disparity, respectively. Θ stands for the scaling operator for $\delta(x)$ (disparity) and $v(x,y)$ (motion), and $Warp$ is the warping operator. We require the scaling operator because the estimation came from the previous coarser scale of a smaller resolution. In addition to this image resolution resize, Θ involves a multiplication of the estimation by 2 (the sampling factor among the consecutive scales to properly scale flow or disparity estimates).
– Merging: The partial estimations are collected in this module to compute the total estimation scale by scale. The refined motion or disparity of the current scale is added with the previous (and coarser) estimation.

The computation of the algorithm is simple; we firstly compute the pyramid for the input images and estimate the coarsest results (the estimation for the coarsest-scale images). Then, we iterate the following steps depending on the number of scales: we upsample the last estimation results, then we perform the warping using the previous upsampled estimation multiplied by 2 and generate the motion or disparity compensated frames for the next finer scale computation; after iterating throughout all the scales we collect and merge all the estimations. A scheme of the process is presented in Section 4.

3 Benchmarking the Color-based Implemented Designs

This section evaluates the results for the color-based optical flow and disparity implementations. The analysis includes a detailed accuracy and stability study. This represents one of the significant contributions of this paper, because though color has been used previously in different optical flow and disparity works (Golland and Bruckstein (1997), Koschan et al (1996), and Andrews and Lovell (2003)), a detailed analysis using standard benchmarks has seldom been performed.

For the optical flow, we benchmark with the following color sequences available at the Middlebury Database (Middlebury (2011)) that also provides the ground-truth: $'Grove2'$, $'Grove3'$, $'Hydrangea'$, $'RubberWhale'$, $'Urban2'$ and $'Urban3'$ (represented as $bof1$ to $bof6$ respectively). We also use the available set of images provided by the Middlebury Database for the disparity: $'Tsukuba'$, $'Sawtooth'$, $'Venus'$, $'Teddy'$ and $'Cones'$ (represented as $bd1$ to $bd5$). Our proposal evaluates the intensity- and color-based implementations and the mono-scale and multiscale version of each of them. We test different color representations to select the best alternative: RGB, normalized RGB, HSV, and YUV. A key point here is evaluating the choice of using 2 or 3 color channels for the different color spaces. The stability study is performed with a benchmark with the previous sequences injecting noise (additive, multiplicative, and salt & pepper). This is also an important decision parameter because the target application deals with real-world applications and therefore, needs to be robust against image artifacts.

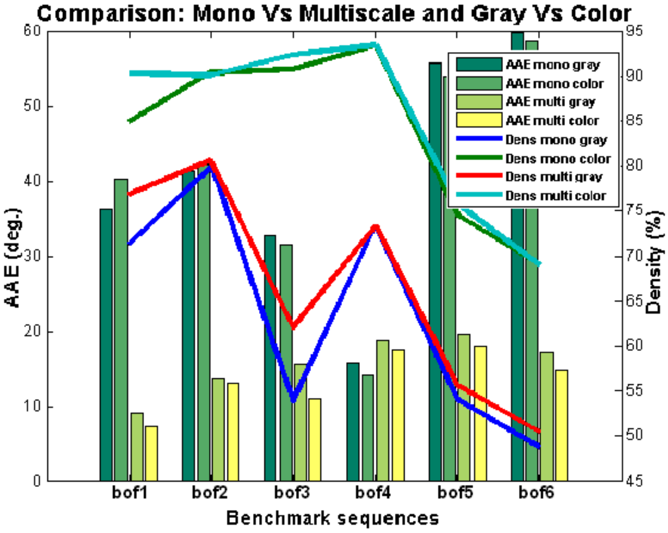


Fig. 2 Precision comparison of the optical flow estimations for the multi-scalar, mono-scalar, and intensity and color-based versions (using the RGB color space).

3.1 Optical Flow Evaluation

The evaluation of the optical flow accuracy uses the AAE (Average Angular Error as defined in Barron et al (1994)) and the density for the color and monochrome versions. We also present the density because, as mentioned in Section 2, our algorithms achieve sparse results due to the use of confidence measures that discard bad estimations.

Accuracy and density comparison. Fig. 2 shows the comparison among multi- and mono-scale versions and the intensity- or gray- and color-based alternatives (using the RGB color space) explained on next sections. As presented, the accuracy improvement for the multi-scale version is about 5x in the best cases (for $bof1$ and $bof5$) using 5 scales, with respect to the monoscale approach. With respect to the color, the main difference is the density increase: the maximum increase is 30.33% and the average is 18.68%. Generally, this high increase in density is also accompanied by an increase in accuracy: 4.5 degrees less in the best case and 1.96 degrees on average. The use of the information from three color channels entails an increase in the confidence due to the information redundancy, adding more robustness to our estimation. The same threshold has been chosen to properly compare the monochrome and color versions. Note that the accuracy and density of the methods are significantly affected by the confidence threshold and consequently, a trade-off value has been selected.

Using only 2-color channels. We have compared the multi- and mono-scale versions and shown the improvements with the color-based implementation for the optical flow. In Fig. 3, we find the AAE and density results for the intensity- and color-based alternatives including RGB, normalized RGB, HSV, and YUV color spaces (first column of each group). The best 3-component results are presented for the RGB and the HSV color spaces.

Another key aspect is the use of representations with 2 (instead of 3) color channels to save resources. Fig. 3 illustrates the comparison between the color representations and their 2-channel versions. As Golland and Bruckstein (1997) explained, applying the reflectivity model to color optical flow, we may use 2 color components without losing accuracy and saving resources. The values of these two color channels should be two independent properties of the color spectrum, representing color as color properties, not as brightness. Golland and Bruckstein (1997) proposed the use of any two components of the normalized RGB color space; we select R and B channels because the results for this alternative are the best ones. For the HSV representation, we select H and S (V represents the luminance channel), and for YUV, the best alternative is Y and V. RGB representation is not suitable for adopting these 2-color channel implementations since it is not based on ratios.

Analyzing the different panels separately, the normalized RGB result figure (first row) presents slight differences in density (only for $bof5$ and $bof6$) while the AAE is maintained in the same level for both alternatives (2 and 3 color channels). In this way, we justify the use of only the R and B components of this color representation. For the HSV result figure (second row), differences in density are more considerable for $bof1$ and $bof2$ with a loss of 30% of density; with respect to the accuracy, the two color component alternative achieves a slight increase (though less than 2 degrees). Finally, the YUV result figure (third row) shows the comparison for the YUV representation; density values are similar in both cases (YUV and YV) and the difference in the AAE measure is substantial for $bof3$ and $bof6$. In the first case, the AAE is lower for the three color component alternative but for the two color channel approach, it sometimes degrades the accuracy significantly (as it is the case in $bof3$ at the same density rate) while it achieves a better accuracy (at a slight density reduction cost) in $bof6$. This difference could be attributed to the impact of the bad results in AAE obtained by this representation for sequences $bof2$, $bof3$, and $bof6$. In the comparison, we see that there are no substantial differences in accuracy and density for using 2 or 3 color components in the first two cases (except for the density in the second case). Since we cannot extract very clear conclusions, we add another analysis: to demonstrate the possibility of using the 2 channel versions we inject noise to the sequences to study a case more similar to the real-world sequences and also to test the stability of the different versions.

Stability analysis (2-color channels). The stability study shows the behavior of the representation in the presence of noise (similar to the effect presented in real-world sequences) for optical flow computation. Fig. 4 shows these

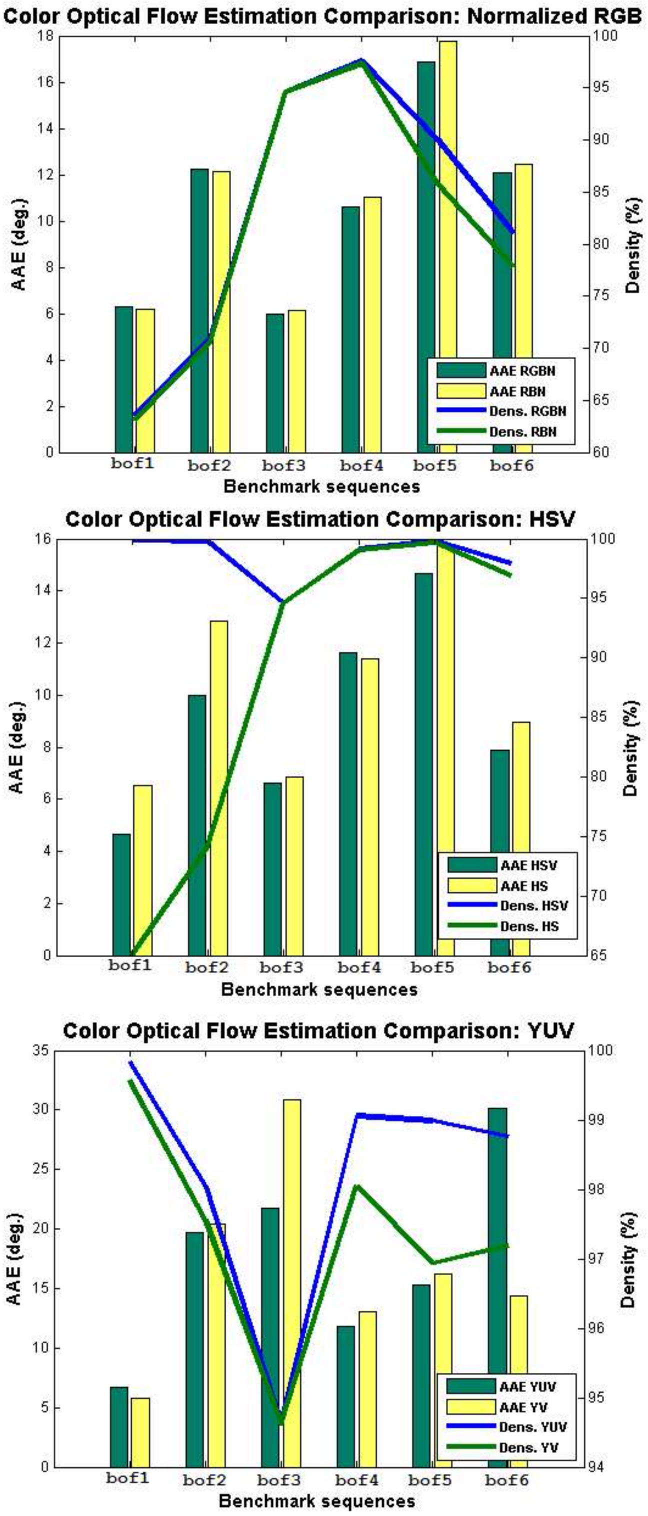


Fig. 3 Multi-scale optical flow estimation with 2 and 3 color components. From top to bottom: normalized RGBN vs. normalized RB, HSV vs. HS, and YUV vs. YV.

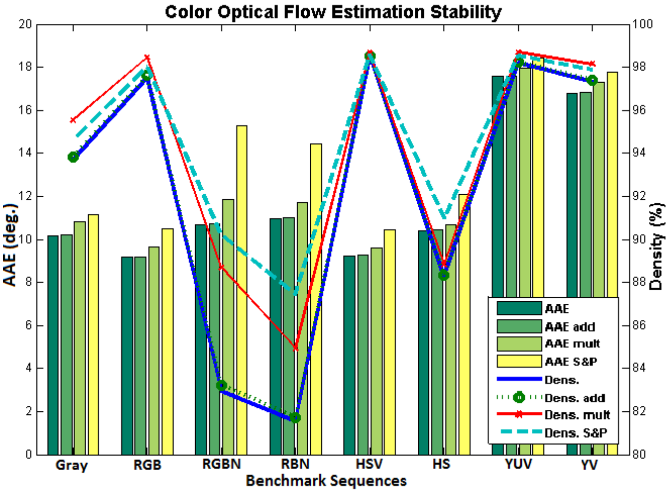


Fig. 4 Stability of the average multi-scale color optical flow estimations injecting noise: additive, multiplicative, and salt & pepper (average values of all benchmark sequences).

results of all the versions (intensity- and color-based) comparing the average of all the synthetic sequences and the average of the same sequences with a different kind of injected noise: additive with a probability distribution of $N(0.1, 0)$, multiplicative with $N(\alpha * I, 0)$ where $\alpha = 0.01$, and salt & pepper with a uniform probability distribution with salt likelihood of 0.0005 and pepper likelihood of 0.001. For the average, we use all the benchmark sequences ($bof1$ to $bof6$) and apply a median filtering for post-processing.

The optical flow stability analysis reveals some crucial conclusions: the use of 2 or 3 color components for the normalized RGB or the HSV has not a considerable loss of accuracy or density, and the stability is stronger for these two representations; the best 3-color-component representation, taking into account the density, error, and stability is the normalized RGB and then, the best 2-color-component representation is the HS representation (the accuracy loss with respect to HSV is not substantial, about 7%). Another conclusion regarding the impact of the noise is that the salt & pepper one produces more instability. This impact is not so significant for the intensity-based implementation but this is because in the color-based versions, the injection is performed in each channel separately, what is translated into a strong negative effect in the final color sequence. Nevertheless, this is not a critical issue because this noise may be easily removed in a real-world scenario by applying median filtering. In Section 5, we will compare accuracy and density of monochrome and color final system.

3.2 Disparity Evaluation

In the case of disparity, we present the MAE (Mean Average Error) and density for the accuracy study. We also show the comparison in terms of these error measures

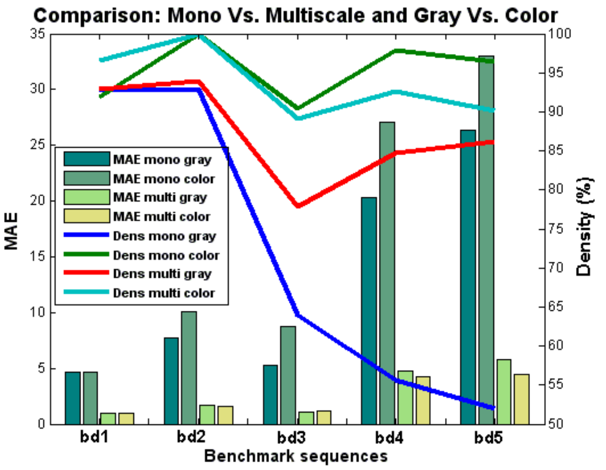


Fig. 5 Precision comparison of the disparity estimations for the multi-scalar, mono-scalar and intensity and color-based versions (using RGB color space).

between the mono- and multi-scale, and the color and monochrome alternatives. Finally, we also test the stability of the algorithm by injecting different types of noise and benchmarking, as in the case of optical flow.

Accuracy and density comparison. Fig. 5 shows the differences for the mono-scale and multi-scale implementations and also compares the intensity and color versions using RGB representation for the disparity computation. Multi-scale extension means a decrease in the error of about 85% for *bd*5 (the best case), while the density increase reaches 40%. It is important to remark that for the two first sequences, the values are very similar, which is due to the disparity ranges ($'Tsukuba'$ [5, 14], $'Sawtooth'$ [3.875, 17.875], and $'Cones'$ [0, 54]). The differences between the color and the intensity-based versions are also more significant for the last sequences than for the first ones, but this could be attributable to the number and size of planar regions.

Using only 2-color channels. Fig. 6 illustrates that the use of two-channel representations instead of three is possible without degrading performances and it allows us to save computing resources. Applying the same principles and hypothesis as in the case of the optical flow, it is possible to use just the appropriate channels of the color representations to compute the disparity to minimize the resource utilization. The choices for the color representations are the same as in the previous optical flow analysis. Describing each panel in detail, the one which compares the HSV results shows the best behavior comparing the two and three-components versions with just slight differences in the error (the biggest difference of 0.8 in MAE (pixels) is reached for *bd*2). With respect to the density, there are no differences between both alternatives. The general trend of the graphics that compare the normalized RGB and the YUV results is a very small accuracy difference (not bigger than 0.5 in MAE) and light drops in the density with a not very significant peak in the case of the normalized RGB representation for the sequence *bd*4 (with a difference of 6%). This means that for the disparity computation, the use of two color component representation instead of three to optimize the resource utilization is suitable.

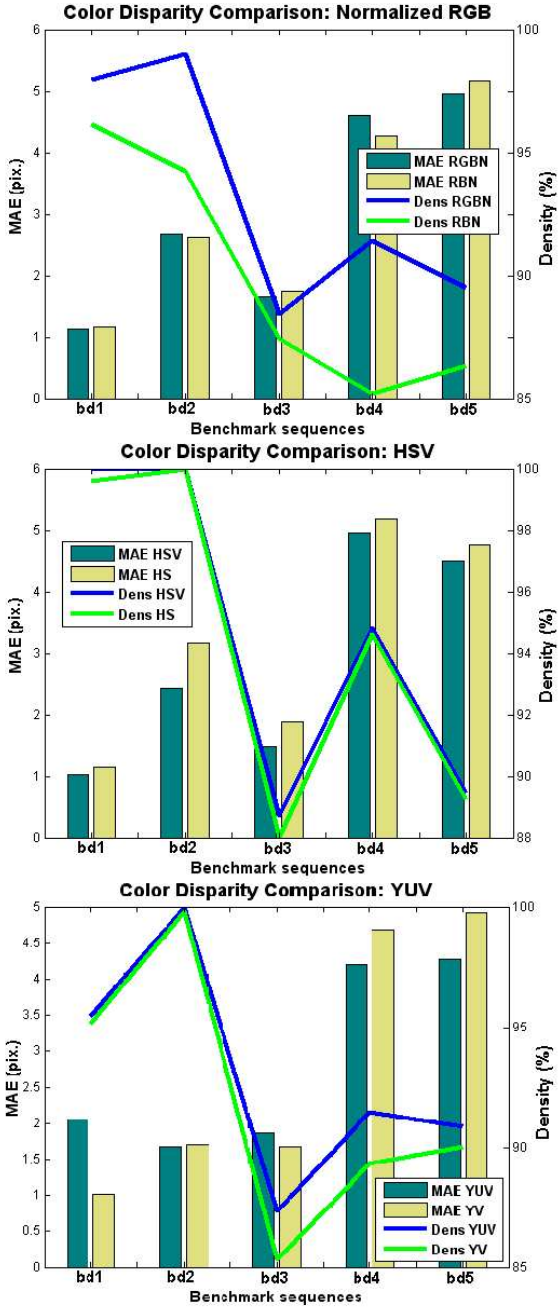


Fig. 6 Multi-scale disparity estimation with 2 and 3 color components. From top to bottom: normalized RGBN vs. normalized RB, HSV vs. HS, and YUV vs. YV.

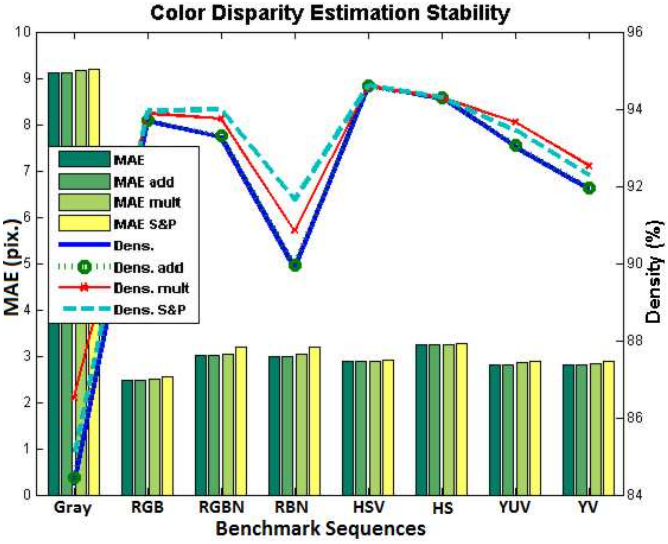


Fig. 7 Stability of the multi-scale color disparity analysis for the different versions injecting noise: additive, multiplicative, and salt & pepper (average values of all the benchmark sequences).

Stability analysis (2-color channels). Finally, Fig. 7 presents the average results for the stability for the different color representations benchmarking in the same way as the optical flow (the injected noise is characterized by the same parameter setting as in the optical flow experiments). From this study, we conclude as follows: the most stable results are the HSV and HS ones, with essentially the same density values and similar precisions comparing 2 and 3 color channels implementations; the results corresponding to the normalized RGB representation have the worst stability behavior, but this is not very considerable in general terms; RGB (with 3 color channels) and YV and HS representations (for 2 color channels) provided the best performances in terms of density and accuracy as seen before. Salt & pepper noise causes the worst accuracy results but, in the case of the disparity, the impact of the noise injection it is not as significant as in the case of optical flow. A preliminary analysis shows that a possible reason is the use of the MAE metric for the disparity accuracy. All these results lead us to use the two color channel representation to obtain a good trade-off between accuracy and density vs. resource utilization.

In Section 5, we will discuss the comparison in accuracy and density of monochrome and color final systems.

4 Hardware Implementation

As explained, our main goal is to obtain real-time performance for our implementation with reasonable accuracy and density. Between the possible hardware alternatives, we select the FPGA, in our case, a XircaV4 board (provided by SevenSolutions (2011)) with a Xilinx Virtex4 XC4vfx100 chip. This board can be used as a standalone platform or as a co-processing board connecting it to the computer via the PCIe interface.

The implementation is based on the fine-pipelined design strategy. Our purpose is to exploit the maximum level of parallelization inherent to the algorithm using the advantages of our hardware platform, achieving a throughput of one pixel per clock cycle with a frequency that allows us to fulfill the real-time requirements. It also allows us to obtain a low power consumption (Diaz et al (2008)).

The components dedicated to the communication protocols, the interfaces, and the Memory Controller Unit (MCU, based on Vanegas et al (2010)) are implemented using VHDL. The most complex algorithmic components of the complete system, in our case, the optical flow estimation, the disparity, and the hierarchical architecture for the multi-scale design, are implemented using the high-level Handel-C language. It avoids a substantial loss of performance in comparison with a fully VHDL solution, as evaluated in Ortigosa et al (2006).

In the next subsections, we will describe the resource utilization of the optical flow (subsection 4.1) and disparity (subsection 4.2) monoscale computing cores. This will allow us to focus on the effect of the inclusion of the color representation in the proposed architecture. Section 4.3 will present the multiscale model extension for these implementations (disparity and optical flow) showing the final system resource utilization.

4.1 Color Lucas&Kanade Optical Flow Core

The implementation of the color L&K core is based on a previous approach described in Diaz et al (2008) and Barranco et al (2011b). They apply the same strategy for the intensity-based implementation of the optical flow estimation. In a previous work (Barranco et al (2011a)), we have also studied the optical flow implementation with color in a very superficial way. After justifying the use of color to improve the density and accuracy of our system, we have included in the present work an extensive study of the different color spaces. Moreover, we have also added a stability analysis and a final comparison between the hardware monochrome and color systems. Finally, our development achieves an improvement in the accuracy maintaining a good frame rate for our real-time requirements.

The implementation of the single-scale optical flow core takes the 6 input pixels (2 for each color channel using 3 frames), with a bitwidth of 8. It also receives a control word with input parameters as the number of scales, the resolution of the input image, the thresholds for the confidence measures, or the median filtering. The output of our core is a 24-bit word (packed in 32-bit words due to the memory alignment): 12 bits per component velocity.

The first stage of the computation filters the input frames using Gaussian smoothing kernels ($K = [1\ 2\ 1]/4$) which allows us to reduce the aliasing effect due to the scaling operation that is performed previously in the hierarchical extension. Then, the next step computes the temporal derivative (using the 3 frames) and the spatio-temporal ones, using the previous results, obtaining then the partial derivatives I_{xi}, I_{yi}, I_{ti} of (1). The temporal and spatial derivative kernels are based on Simoncelli (1994). Both, the derivative and the smoothing kernels are based on separable filters, which means that the convolution operation can be applied as two consecutive mono-dimensional row and column-wise operations, which allows us to maintain a throughput of 1 pixel per clock cycle. The implementation of these convolution operation involves the buffering of rows of the image, in our case, in the available embedded Block RAMs memories in our FPGA. The number of stored rows depends on the size of the filter, 2 rows in the case of a 5-by-5 filter (spatial derivative filter) and 1 row for the 3-by-3 filters (smoothing and temporal derivative filters).

The last stage computes the coefficients of the linear system proposed in (**??**) and (7). Weights W_i for neighborhood Ω are set by the 5-by-5 separable kernel used in Barron et al (1994) and Diaz et al (2008): $W = [1\ 4\ 6\ 4\ 1]/16$. Finally, the last step consists in solving the 2-by-2 resultant system. The resolution of the system involves the most expensive operations, two IP division cores (one of each component velocity) with signed 24-bit operands that were developed using Xilinx Core Generator tool.

Local gradient-based algorithms are sparse. This means that we use a threshold to ensure the confidence of the estimates for each pixel. In the case of optical flow, we use the determinant of (**??**) as in Brandt (1997) and Diaz et al (2008). We set to NaN (we use a reserved word for these special values) the discarded values using this threshold.

As mentioned, the implementation of the core is based on fine-grain pipelines, designing a superscalar architecture with a variable number of pipelines that depends on each stage intrinsic parallelism. Each pipeline has a length of 82 stages with multiple parallel paths implemented to keep the total system throughput. The final architecture has a total of 361 micro-stages.

All the operations in both cores, disparity and optical flow, are implemented using fixed-point arithmetic. This decision reduces significantly the hardware resource utilization, nevertheless, it also degrades the accuracy of the solution. Hence, the operand bitwidth at each stage is selected as a trade-off between the accuracy at each stage and the impact for the complete datapath, and the final resource consumption. This method is detailed in Diaz et al (2008), where the author carried out an extensive batch of simulations for the bit-width decision. This implementation was constrained by the hardware resources, the memory interface, and the bit-width of the embedded resources in its board. In our work, we extend the bit-width of the operations for the most constrained stages to improve accuracy and density. It is important to take into account that the accuracy of the estimations at the coarsest scales is significant for the multi-scale implementation because they drive the estimations of the finest ones. This means that we need a good accuracy at the coarsest scales in order to keep high accuracy along the multi-scale approach as images are warped according to the velocity results of the coarsest scales.

Resource utilization. Table 1 shows the used resources and the reached frame rate for each alternative. The synthesis software is from Xilinx (2011). In the best case, we reach a frequency of 83 MHz. It means a theoretical (not taking into account board-PC communication bandwidth constraints) performance of 270 fps (648x480 VGA resolution) and an occupation of 10% for the mono-scale approach.

We evaluate now how the color approach implementation impacts on resource utilization. Using 3 or 2 color channel cores does not mean an increase of 3x or 2x of the resource utilization with respect to the monochrome version (intensity based). In fact, the optimization process allows us in the worst case (using 3 channels) an increase with a factor of 2.2x. This fact is fundamental because we have previously shown the possibility of using only 2 channels for color. It helps us saving 30% core resource utilization (reducing the total available resource utilization in 6%) without a significant loss of accuracy and keeping frequency. It also means that we only need two warping modules and Laplacian pyramids instead of three for the whole system implementation. As the resource use is small for the different cores, the use of more color channels does not affect the final delay and hence, the frequency remains approximately constant.

4.2 Color Lucas&Kanade Disparity Core

The color disparity development is based on the L&K algorithm, in a similar way to the previous approach for the optical flow. As explained in the first section, its implementation is performed using the same algorithm but, in this case, using right and left images as inputs.

Three channels are the inputs of our core: 16-bit inputs for left and right color images (8 bits per each color channel) and a control word with parameters for the core (number of scales, image resolution and confidence thresholds). The disparity estimation bitwidth is 12 bits.

The first stage consists in a smoothing as in the case of optical flow and the computation of the spatial derivative computation between the left and right images. Then, it implements the partial spatial derivative and the local difference between left and right images, obtaining L_{xi} and $R_i - L_i$ that are going to be used in (10). We use the same kernels for the spatial derivative, smoothing and weighting matrix (W_i) as in the optical flow core.

Table 1 Hardware resource utilization and frame rate (resolution of 640 x 480) for the presented mono-scale alternatives for optical flow and disparity design using a Xilinx Virtex-4 FX100 FPGA. The notation of 2 Ch and 3 Ch are used to differentiate between the 2-color-channel and 3-color-channel implementations.

	4 input LUTs (out of 84352)	Slice Flip Flops (out of 84352)	Slices (out of 42716)	DSP (160)	Block RAMs (376)	Freq (MHz)	Frame Rate (fps)
O.F. Intensity	5039 (5%)	6622 (7%)	4224 (10%)	30 (18%)	48 (12%)	83	270
O.F. 2 Ch	7939 (9%)	9068 (10%)	6562 (15%)	35 (21%)	92 (24%)	83	270
O.F. 3 Ch	10837 (12%)	11519 (12%)	9274 (21%)	40 (25%)	136 (36%)	83	270
Disp. Intensity	2358 (2%)	3120 (3%)	2129 (5%)	3 (1%)	28 (7%)	109	355
Disp. 2 Ch	4534 (5%)	4660 (5%)	3916 (9%)	6 (3%)	56 (14%)	109	355
Disp. 3 Ch	6435 (7%)	6212 (7%)	5367 (12%)	9 (5%)	84 (22%)	109	355

The resolution of Equation (10) is performed by using a single division core of 16-bits. Furthermore, as in the previous case, to ensure the confidence of the estimates, we use a threshold, based on the difference between the right and left image spatial derivatives, setting non-confident values to NaN values.

Finally, and in this case, each pipeline has a length of 70 stages, with a total of 265 micro-stages.

Resource utilization. Table 1 shows the used resources and the reached frame rate for each alternative. The synthesis software is Xilinx ISE from Xilinx (2011). In the best case, we reach a frequency of 83 MHz. It means about 270 fps (640x480 VGA resolution) and an occupation of 10% for the mono-scale approach. Using 3 or 2 channel cores does not represent an increase of 3x and 2x in resources. The increase is similar to the one reached for the optical flow implementation.

4.3 Multi-scale Architecture Extension

Our multi-scale architecture is based on warping, an approach usually avoided due to its architectural complexity and its critical resource costs. In this section, we describe the main components of the multi-scale architecture and the algorithm which are based on Tomasi et al (2012). The multiscale architecture is described in detail in Tomasi et al (2010b).

The interactions with memory are performed by the MCU that multiplexes in time the data that have to be read/stored (Vanegas et al (2010)). We also have parallel access to memory using the multiple available banks (4 DDR memory banks). The MCU provides an easy interface by showing multiple independent ports, providing the abstraction of a multi-port memory and facilitating the implementation of the algorithm. Finally, the main modules are:

- Pyramid: It is built by a smoothing and a sub-sampling circuit. Each pyramid scale is obtained sequentially by a set of cascade decimation filters and images are directly read/stored into an external RAM memory. This module is replicated for each color channel and for each frame (three for the optical flow and two for disparity). This storage is carried out by using the multiple banks and the MCU, to ensure the simultaneous access to the frame data (3 in the case of optical flow and 2 for disparity). The final subsampling is performed by discarding one of every 2 pixels in the smoothed output.
- Upsampling: This module performs the upsampling of the estimation for the previous scale to the spatial resolution for the current iteration, and multiplies them by 2 (upsampling factor is 2). The computation consists in a bilinear interpolation, but we need circular buffers to store one image row, to reduce memory accesses. The main problem in this stage is the synchronization between this circuit and the rest of the processing (it produces 4 pixels for each input data). This problem is solved by buffering the data in embedded BlockRAMs as FIFOs whose length is selected to match the proper latency.
- Warping: This stage is the most complex one of the multi-scale implementation. As mentioned before, in the case of optical flow, it consists in a bilinear interpolation of the input images with the shift values that we obtained from the optical flow estimation in the previous scale. The computation of each warped pixel requires the reading of the pair (Δx, Δy), that is, the values of the shift and pixel P. With this pair (Δx, Δy), we retrieve from memory four pixels of the original image. This module is replicated for each color channel. In the case of disparity, the warping computation is reduced to a linear interpolation for which we need only the two values of the nearest pixels. In this case, we have only Δx for pixel P.
- Merging: This module computes the addition of the previous feature estimation and the current one, storing the result for the next one. Non-confident values are propagated from the coarsest to the finest scales. The main problem at this module is the synchronization between the current and the stored results, performed by using FIFOs (embedded BlockRAMs).
- Median Filtering: We filter the output of the feature computation using two 3-by-3 median filters in cascade, for the scale-by-scale estimations as seen in Fig. 1. This regularizes the final values and eliminates

non-confident values in plain areas, also increasing the density of the final results.

Next, we summarize the main differences between the multiscale monochrome and the color architectures. The pyramid computation is replicated depending on the number of color channels, replicating also channels to read/write from/to memory. The warping stage is also replicated for each color channel, involving a high cost in terms of resource utilization. The merging module remains for both monochrome and color approaches as well as the median filtering post-processing. As the pyramid is computed sequentially before the disparity or optical flow computation, the accesses to/from memory are not a crucial problem. However, in the case of the warping, these accesses to/from memory become a bottleneck and restrict our final performance.

The final system resource utilization, performance, and accuracy evaluation are addressed on Section 5.

5 Final System: Resource Utilization, Performance, and Accuracy

This section analyzes the complete multi-scale on-chip systems for the disparity and optical flow estimation. It begins with a brief comparison of our systems with the state-of-the-art works. The second part is dedicated to the final performance of the implemented systems, the precision analysis, the resource utilization, and some results in real-world scenarios.

5.1 State-of-the-art Comparison

In this section, we include a brief comparison of our work and several implementations in the literature. As presented in the Introduction, our main requirement is the implementation of a high-performance system. In Table 2, our system achieves 36 fps and 32 fps (for disparity and optical flow respectively and measured directly with the used board) for an image resolution of 640x480 (VGA), which fulfills our real-time goal. In both cases, we compare our mono and multiscale implementations with the latest works in the literature, including monochrome and color-based implementations for different architectures: FPGAs, standard PCs, GPUs, etc.

For the disparity, most of the works are monoscale approaches and the FPGA-based ones are mostly SAD algorithms. The table also shows the PDS (Points x Disparity measure per Seconds). Our performance is in the middle of the PDS ranking.

For the optical flow, the table lists the performance of the motion estimation. Among the FPGA-based implementations, we highlight the one by Tomasi et al (2010a), with a monoscale implementation that works at 80 fps, or the one by Botella et al (2010), that achieves 16 fps (but at a smaller image resolution). Finally, we also highlight the CPU version from Anguita et al (2009) with an effective frequency of 90 MHz at 1280x1026 resolution but, with a mono-scale version of the L&K algorithm.

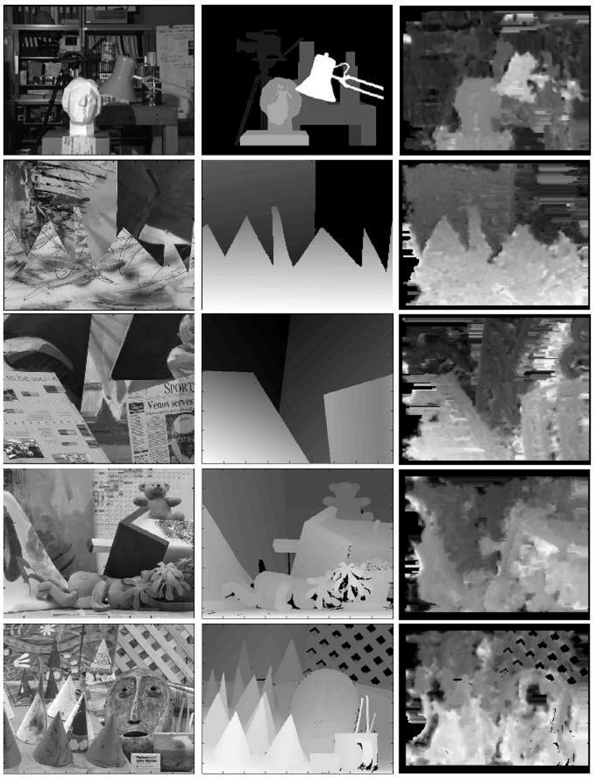


Fig. 8 Disparity benchmark results for the YUV representation (using Y and V channels). From left to right: original image, ground-truth disparity, hardware disparity results. Bright colors represent closer objects and dark colors farther objects. Black represents unreliable data.

5.2 Results Discussion

We summarize here the final performances of the implemented systems, the resource utilization, the accuracy analysis and some qualitative results. The final systems are implemented using 2-color-channels architectures and the results are provided using: YUV representation for disparity and HSV for optical flow.

Resource utilization and performances. Table 3 presents the resource utilization, including the intensity and color systems and the detailed data of the most important modules of the multi-scale implementation.

The resource utilization increase is 1.34x and 1.20x for the color optical flow and disparity implementations with respect to the intensity designs respectively (using 2 color channels). Detailing this information, the warping module is the most expensive component, using about 23% of the total resources (including the interface

Table 2 Optical flow and disparity performance comparison with previous works sorted by date of publication (SAD is Sum of Absolute Differences; SGM is Semi-Global Matching).

OPTICAL FLOW	Resolution	Frame rate (fps)	MPPs	Architecture	Algorithm
Our mono-scale core	640x480	270	82.9	Xilinx V4 (83 MHz)	Color LK
Our multi-scale work	640x480	32	9.8	Xilinx V4 (44 MHz)	Color LK
Botella et al (2010)	128x96	16	0.20	Xilinx V2	McG
Tomasi et al (2010a)	640x480	81	24.88	Xilinx V4 (50 MHz)	Phase-based mono
Gwosdek et al (2010)	316x252	210	16.72	Cell Processor, PS3	Variational
Claus et al (2009)	640x480	45	13.8	Xilinx V2 (100 MHz)	Color Census
Claus et al (2009)	640x480	25	7.68	Core 2Duo (1.86 GHz)	Color Census
Lei and Yang (2009)	640x480	0.004	0.0012	AMD 2.2GHz	Color Discrete Optimization
Anguita et al (2009)	1280x1026	68.5	71.8	Quad Core 2 (2830 MHz)	LK
DISPARITY	Resolution	Frame rate (fps)	PDS (10^6)	Architecture	Algorithm
Our mono-scale core	640x480	355	218	Xilinx V4 (109 MHz)	Color LK
Our multi-scale core	640x480	36	704	Xilinx V4 (49 MHz)	Color LK
Villalpando et al (2011)	1024x768	12	132	Xilinx V4 (66 MHz)	Color SAD
Chen et al (2011)	320x240	30	148	Xilinx V4 (60 MHz)	Color SAD
Calderon et al (2010)	288x352	142	2534	Xilinx V2 Pro (174.2 MHz)	SAD
Tomasi et al (2012)	512x512	28	939	Xilinx V4 (42 MHz)	Phase-based
Georgoulas and Andreadis (2009)	800x600	550	21120	Stratix IV (511 MHz)	Color SAD
Gibson and Marques (2008)	450x375	6	65	G80 NVIDIA	SGM
Woodfill et al (2004)	512x480	200	2555	Tyzx ASIC	Color Census

Table 3 Resource utilization for the multi-scale implementation. In the case of the color-based cores or architecture, we only use two color components

	4 input LUTs (out of 84352)	Slice Flip Flops (out of 84352)	Slices (out of 42716)	DSP (160)	Block RAMs (376)	Freq (MHz)
Intensity-based O.F.	31796 (37%)	24694 (29%)	26036 (61%)	62 (38%)	112 (29%)	44
Intensity-based Disp	21287 (25%)	16989 (20%)	18773 (44%)	6 (3%)	92 (24%)	49
Color-based O.F.	44239 (52%)	37916 (44%)	34630 (82%)	98 (61%)	220 (58%)	44
Color-based Disp.	28517 (33%)	20818 (24%)	22423 (53%)	11 (6%)	143 (38%)	49
Board Interface	4774 (5%)	5195 (6%)	5388 (12%)	0	36 (9%)	112
Scaling	413 (1%)	270 (1%)	367 (1%)	0	1 (1%)	86
Warping 2D + Int.	9943 (11%)	9097 (10%)	9894 (23%)	32 (20%)	43 (11%)	51
Warping 1D	510 (1%)	324 (1%)	436 (1%)	2 (1%)	5 (1%)	55
Merging	364 (1%)	244 (1%)	235 (1%)	0	4 (1%)	107
Regularization	3904 (4%)	2540 (3%)	2343 (5%)	59 (36%)	8 (2%)	78

Table 4 Comparison of hardware multi-scale disparity estimations for the monochrome and the color-based approaches (software and hardware versions). The color version is performed using Y and V channels of the YUV representation

	bd1		bd2		bd3		bd4		bd5	
	MAE	Dens.	MAE	Dens.	MAE	Dens.	MAE	Dens.	MAE	Dens.
HW Monochrome	0.94	93.89	1.98	93.34	1.55	75.56	6.22	81.83	6.94	84.62
HW Color-based	2.74	98.51	1.31	90.29	1.74	87.35	5.34	83.45	6.97	80.99
SW Color-based	2.06	95.47	1.67	99.99	1.86	87.33	4.21	91.49	4.27	90.84

with the MCU), this justifies that a lot of works avoid its implementation and therefore most of them so far in the literature deal only with mono-scale approaches or matching-based techniques. Furthermore, it is worth noticing that this module has to be also duplicated, one for each color channel. The multi-scale architecture, including the MCU and its interface, sums almost 20% and 45% for the disparity and optical flow intensity-based alternatives. For the color implementations, these percentages increase approximately to 23% and 56%. The difference is due to the warping module: for the color design this module has to be duplicated as well as the operations with memory which implies a high cost in the case of the optical flow. The one-dimensional warping used for the stereo is implemented using on-chip memories working as caches, since it requires a much simpler architecture. Focusing on the performance, the final color-based systems achieve frequencies of 44 MHz fulfilling the real-time requirements, while the intensity-based ones obtain only about 5 MHz more.

Our system uses 82% and 53% of the resources for the optical flow and the stereo computation respectively.

Table 5 Comparison of hardware multi-scale optical flow estimations for the monochrome and the color-based approaches (software and hardware versions). The color version is performed using the H and S channels of the HSV representation

	bof1		bof2		bof3		bof4		bof5		bof6	
	AAE	Dens.	AAE	Dens.	AAE	Dens.	AAE	Dens.	AAE	Dens.	AAE	Dens.
HW Monochrome	6.21	64.44	11.88	72.09	16.88	33.80	23.11	44.81	18.44	49.61	19.95	45.32
HW Color-based	7.22	61.53	12.7	70.47	17.82	73.79	19.72	66.55	18.10	94.25	11.04	81.98
SW Color-based	4.64	99.85	9.99	99.75	6.61	94.61	11.61	99.09	14.65	99.80	7.86	97.88

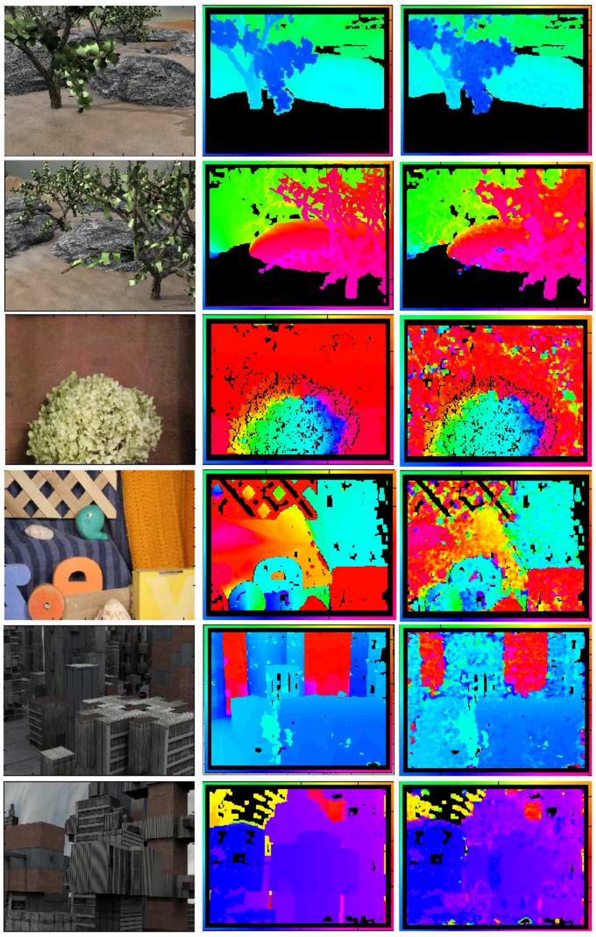


Fig. 9 Optical flow benchmark results using H and S channels (from HSV). From left to right: original image, ground-truth, hardware optical flow result (the color represents motion direction according to the image frame). Black areas represent NaN data for the ground-truth or the hardware results.

Therefore, it is possible to add more cores to the multi-scale architecture based on color cues.

Accuracy analysis. Firstly, we present the qualitative results of the benchmark computed for the color disparity and optical flow hardware architectures in Figs. 8 and 9. For disparity, the most important errors are in areas close to the image borders, or for certain sequences due to the disparity ranges. Although multiscale implementation provides more accuracy, gradient-based implementations are not well-suited for high range disparity computation and are significantly affected by illumination changes and in general, occluded or textureless regions.

As explained before, our purpose is the implementation of a generic architecture to compute disparity and optical flow using two color channels of independent information. In both cases, we perform a median filter stage to regularize the estimations and a post-processing stage to fill the holes (NaN values). This row-wise processing consists in substituting the NaN values for the average pixel in a neighborhood of 5 pixels.

In this case, these two channels of independent information can be modified to implement different alternatives (not only for color cues). Moreover, Table 4 and Table 5 show the error measures for both architectures. In those tables, we see the density drop and the error increase due to the use of fixed-point arithmetic of our hardware implementations. Furthermore, in the case of the optical flow, we achieve a similar AAE to the software version (with a difference of 2 or 3 degrees) except for $bof3$ and $bof4$. In these sequences, the precision is not enough and we lose a large percentage of the results (the density is 73% and 66.5%). The implementations for each case are performed using the best alternative color representation (for the 2-color-channel versions) analyzed in Section 3: for the disparity (Fig. 8 and Table 4), we use the Y and V channels of YUV color space and for the optical flow (Fig. 9 and Table 5), H and S of HSV.

Finally, we also compare the monochrome and color-based results obtained with the hardware systems. As seen in Tables 4 and 5, the improvement is more significant in the case of the optical flow although it is also noticeable in the case of the disparity. In the case of optical flow for the first two sequences (*'bof1'* and *'bof2'* i.e. *'Grove2'* and *'Grove3'*), we obtain almost the same density and a slightly worse accuracy. This might be motivated by the color of this particular set of artificial images that have shown a different behavior also in the stability analysis. In the case of the disparity, the differences between color-based and monochrome approaches are not substantial (except for the first sequence); this fact may be attributable to the size and number of planar regions as commented before. For *'bd1'*, the error increase might be determined by some artifact in the YUV representation, that is also present in Fig. 6.

Real-world results. As the car industry is one of our most important fields of application, Fig. 10 shows the results for two real driving sequences. In real scenarios, we find

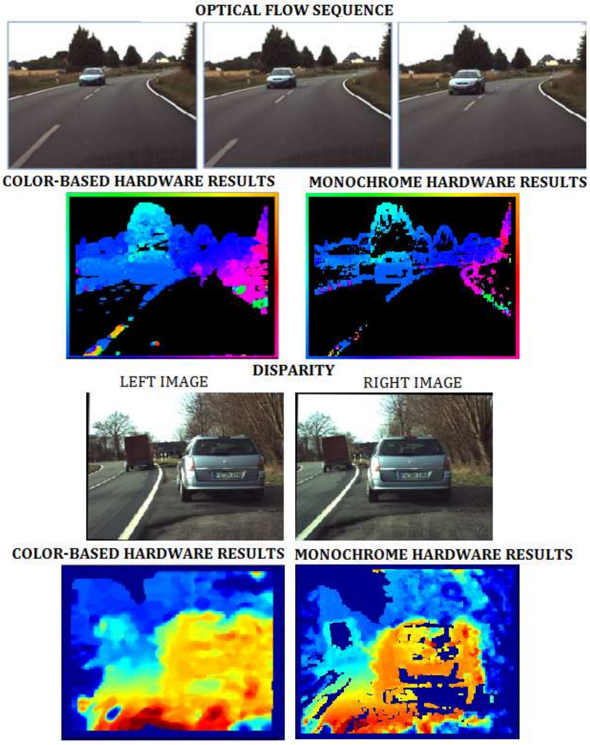

OPTICAL FLOW SEQUENCE
COLOR-BASED HARDWARE RESULTS
MONOCHROME HARDWARE RESULTS
DISPARITY
LEFT IMAGE
RIGHT IMAGE
COLOR-BASED HARDWARE RESULTS
MONOCHROME HARDWARE RESULTS

Fig. 10 The first two rows show the original frames and the optical flow estimation for both monochrome and color-based approaches in a real driving scenario (the direction of the motion is encoded by a color, according to the image frame); the following rows present the left and right images and the disparity results (warm colors represent the closest objects and cool colors the farthest ones; NaN values are in dark blue).

important issues as the noise of the capture devices or illumination changes. In the first row, images are captured by a camera in a car that is moving forward and there is another car approaching. In the second case, we show a pair of images (left and right) where we identify a car immediately in front of us and a truck a bit farther away. We show the results using the color-based and the monochrome approaches. It is worth noticing the significant increase in density that we have explained in previous subsections.

6 Conclusions

In this work, we have designed and implemented an embedded system for the color-based optical flow and disparity computation. We selected a multi-scale implementation of the L&K method taking advantage of its accuracy and efficiency.

Our objective is the implementation of a generic architecture for multi-scale algorithms that have to deal with color cues. Nevertheless, we also analyze the implementation of such algorithms using 2 or 3 color channels. As shown, the 2-channel implementation allows us to save hardware resources without a significant loss of accuracy. This last statement is correct with the appropriate color representation, in our case, HSV for the optical flow and YUV for the disparity using only 2 color channels (i.e. HS and YV respectively) or RGB using the 3 color channels. We also show that the increase in resources derived from the use of color cues leads to a considerable improvement in accuracy and density with respect to the intensity-based version.

One of our objectives was the development of a design which was able to perform the computation in real time and exploit the color information provided by the image sensor and which was usually wasted. The FPGA development and the highly parallel architecture implemented allow us to fulfill these real-time requirements. In fact, the presented systems achieve 270 and 355 fps for the mono-scale versions and 32 and 36 fps for the multi-scale approaches for the optical flow and disparity computations respectively (VGA resolution). These data are obtained empirically (direct measures with the prototyping board) using a software platform developed as an interface between a computer and our FPGA board. The documentation and the source code of this project (”Open rt-Vision project”) are available at OpenRT-Vision (2011). It was released in 2009 with GNU LGPL license.

In our case, for the proposed system, the mono-scale implementation corresponds to 15% and 9% of the available resources for the optical flow and the disparity designs and the multi-scale approaches make use approximately of 82% and 53% of a Xilinx Virtex4 XC4vfx100 chip. In conclusion, we obtain an implementation with a reasonable accuracy, neglecting the expected precision problems for a fixed-point implementation (which are detailed in other works as in Diaz et al (2008)) with an affordable increase in the resource utilization (duplicating the information we have to deal with, the resource utilization is increased in a 1.2-1.3x factor with respect to the intensity-based implementation).

Our design strategy allows an easy sharing of hardware resources (varying the number of pipelined stages and superscalar units). It makes the system suitable for the implementation of new vision algorithms only with the development of the new cores and their respective interfaces. Moreover, the fine pipelined architecture benefits from a high performance and low power consumption (Diaz et al (2008)), crucial for industrial applications.

Finally, the implemented architecture could be the low-level layer for future implementations of complex vision algorithms such as segmentation, 3D reconstruction, or attention. Furthermore, the integration of the color-based cores with other color-based visual primitives in a single chip represents interesting future work.

Acknowledgements This work has been supported by Spanish Grant (AP2007-00275), EU Project TOMSY (FP7-270436), Spanish project ARC-VISION (TEC2010-15396), and regional projects MULTIVISION (TIC-3873), and ITREBA (TIC-5060).

References

Andrews RJ, Lovell BC (2003) Color optical flow. In Proc Workshop on Digital Image Computing 1:135 – 139

Anguita M, Diaz J, Ros E, Fernandez-Baldomero FJ (2009) Optimization strategies for High-Performance computing of Optical-Flow in General-Purpose processors. CSVT, IEEE T 19(10):1475 – 1488

Barranco F, Tomasi M, Diaz J, Ros E (2011a) Hierarchical optical flow estimation architecture using color cues. In: Reconfigurable Computing: Architectures, Tools and Applications, LNCS, vol 6578, pp 269 – 274

Barranco F, Tomasi M, Diaz J, Vanegas M, Ros E (2011b) Parallel architecture for hierarchical optical flow estimation based on FPGA. VLSI Systems, IEEE T DOI 10.1109/TVLSI.2011.2145423

Barron J, Klette R (2002) Quantitative color optical flow. In Int Conf on Pattern Recognitioin 4:251 – 255

Barron JL, Fleet DJ, Beauchemin SS (1994) Performance of optical flow techniques. Int J of Comp Vision 12:43 – 77

Beauchemin SS, Barron JL (1995) The computation of optical flow. ACM Computing Surveys 27(3):433 – 466

Bergen J, Anandan P, Hanna K, Hingorani R (1992) Hierarchical model-based motion estimation. In: ECCV'92, LNCS, vol 588, pp 237 – 252

Botella G, Garcia A, Rodriguez-Alvarez M, Ros E, Meyer-Baese U, Molina MC (2010) Robust bioinspired architecture for optical-flow computation. IEEE Trans Very Large Scale Integr Syst 18:616 – 629

Brandt JW (1997) Improved accuracy in Gradient-Based optical flow estimation. Int J Comput Vision 25:5 – 22

Burt PJ, Edward, Adelson EH (1983) The Laplacian pyramid as a compact image code. IEEE Transactions on Communications 31:532 – 540

Calderon H, Ortiz J, Fontaine J (2010) High parallel disparity map computing on FPGA. In: (MESA), 2010 IEEE, pp 307 – 312

Chen L, Jia Y, Li M (2011) An fpga-based rgbd imager. Machine Vision and Applications pp 1 – 13

Claus C, Laika A, Jia L, Stechele W (2009) High performance fpga based optical flow calculation using the census transformation. In: Intelligent Vehicles Symposium, 2009 IEEE, pp 1185 – 1190

Denman S, Fookes C, Sridharan S (2009) Improved simultaneous computation of motion detection and optical flow for object tracking. In: Digital Image Computing: Techniques and Applications, pp 175 – 182

Diaz J, Ros E, Agis R, Bernier J (2008) Superpipelined high-performance optical-flow computation architecture. Computer Vision and Image Understanding 112(3):262 – 273

Georgoulas C, Andreadis I (2009) A real-time occlusion aware hardware structure for disparity map computation. In: Image Analysis and Processing ICIAP 2009, Springer Berlin, vol 5716, pp 721 – 730

Gibson J, Marques O (2008) Stereo depth with a unified architecture gpu. In: CVPRW '08, IEEE Conf. on

Golland P, Bruckstein AM (1997) Motion from color. Computer Vision and Image Understanding 68(3):346 – 362

Gwosdek P, Bruhn A, Weickert J (2010) Variational optic flow on the sony playstation 3. Journal of Real-Time Image Processing 5:163 – 177

JR Jordan III AB (1991) Using chromatic information in edge-based stereo correspondence. CVGIP: Image Understanding 54:98 – 118

JR Jordan III AB (1992) Using chromatic information in dense stereo correspondence. Pattern Recognition 25(4):367 – 383

Koschan A, Rodehorst V, Spiller K (1996) Color stereo vision using hierarchical block matching and active color illumination. In: Conf. Pattern Recog., vol 1, pp 835 – 839

Krotosky S, Trivedi M (2008) Person surveillance using visual and infrared imagery. Circuits and Systems for Video Technology, IEEE Transactions on 18(8):1096 – 1105

Lei C, Yang YH (2009) Optical flow estimation on coarse-to-fine region-trees using discrete optimization. In: Comp. Vision, 2009 IEEE Int. Conf., pp 1562 – 1569

Liu H, Hong TH, Herman M, Chellappa R (1996) Accuracy vs. efficiency trade-offs in optical flow algorithms. In: ECCV 96, LNCS, vol 1065, pp 174 – 183

Lucas BD, Kanade T (1981) An iterative image registration technique with an application to stereo vision. Conf on Artificial intelligence 2:674 – 679

Markandey V, Flinchbaugh B (1990) Multispectral constraints for optical flow computation. In: Proc. Int. Conf. on Comp. Vision, pp 38 – 41

Matsumoto Y, Shibata T, Sakai K, Inaba M, Inoue H (1997) Real-time color stereo vision system for a mobile robot based on field multiplexing. In: Robotics and Automation, IEEE International Conference on, vol 3, pp 1934 – 1939

Middlebury (2011) Middlebury computer vision. http://vision.middlebury.edu/

Mühlmann K, Maier D, Hesser J, Männer R (2002) Calculating dense disparity maps from color stereo images, an efficient implementation. Int J Comp Vision 47:79 – 88

Negahdaripour S, Madjidi H (2003) Robust optical flow estimation using underwater color images. In: Proceedings on Oceans, vol 4, pp 2309 – 2316

Ohta N (1989) Optical flow detection by color images. pp 801 – 805

OpenRT-Vision (2011) Open rt-vision project. http://code.google.com/p/open-rtvision/

Ortigosa E, Canas A, Ros E, Ortigosa P, Mota S, Diaz J (2006) Hardware description of multi-layer perceptrons with different abstraction levels. Microprocessors and Microsystems 30(7):435 – 444

Ortiz AE, Neogi N (2006) Color optic flow: A computer vision approach for object detection on UAVs. In: IEEE Digital Avionics Systems Conference, pp 1 – 12

Muñoz Salinas R, Aguirre E, García-Silvente M (2007) People detection and tracking using stereo vision and color. Image Vision Computation 25:995 – 1007

SevenSolutions (2011) http://www.sevensols.com/

Simoncelli EP (1994) Design of multi-dimensional derivative filters. In: Image Processing, 1994. Proceedings. ICIP-94., IEEE International Conference, vol 1, pp 790 – 794

Tomasi M, Barranco F, Vanegas M, Diaz J, Ros E (2010a) Fine grain pipeline architecture for high performance phase-based optical flow computation. Journal of Systems Architecture 56(11):577 – 587

Tomasi M, Vanegas M, Barranco F, Diaz J, Ros E (2010b) High-Performance Optical-Flow architecture based on a multiscale, Multi-Orientation Phase-Based model. CSVT, IEEET on 20(12):1797 – 1807

Tomasi M, Vanegas M, Barranco F, Diaz J, Ros E (2012) Massive parallel-hardware architecture for multiscale stereo, optical flow and image-structure computation. CSVT, IEEE T 22(2):282 – 294

Vanegas M, Tomasi M, Diaz J, Ros E (2010) Multi-port abstraction layer for FPGA intensive memory exploitation applications. J of Systems Architecture 56(9):442 – 451

Villalpando C, Morfopolous A, Matthies L, Goldberg S (2011) Fpga implementation of stereo disparity with high throughput for mobility applications. In: Aerospace Conference, 2011 IEEE

Woodfill J, Gordon G, Buck R (2004) Tyzx deepsea high speed stereo vision system. In: Computer Vision and Pattern Recognition, Conf. on

Xilinx (2011) FPGA and CPLD solutions from xilinx, inc. http://www.xilinx.com/

2.4 Multi-resolution approach for parallel optical flow computation

The journal article associated to this part of the dissertation is:

- F. Barranco, J. Díaz, B. Pino, E. Ros, A multi-resolution approach for massively-parallel hardware-friendly optical flow estimation. **Submitted to Journal of Visual Communication and Image Representation**.
  - Status: **Submitted**.
  - Impact Factor (JCR 2010): 1.101
  - Subject Category:
    - Computer Science, Information Systems. Ranking 66 / 128.
    - Computer Science, Software engineering. Ranking 44 / 99.

A multi-resolution approach for massively-parallel hardware-friendly optical flow estimation

F. Barranco[a], J. Díaz[a], B. Pino[a], E. Ros[a]

[a]*Dept. Computer Architecture and Technology, ETSIIT, CITIC, University of Granada, C/P. Daniel Saucedo Aranda, s/n, E-18071, Granada, Spain*

Abstract

This paper presents a novel hardware-friendly motion estimation for real-time applications such as robotics or autonomous navigation. Our approach is based on the well-known Lucas&Kanade local algorithm, whose main problem is the unreliability of its estimations for large-range displacements. This disadvantage is solved in the literature by adding the sequential multiscale-with-warping extension, although it dramatically increases the computational cost. Our choice is the implementation of a multiresolution scheme that avoids the warping computation and allows the estimation of large-range motion. This alternative allows the parallel computation of the scale-by-scale motion estimation which makes the whole computation lighter and significantly reduces the processing time compared with the multiscale-with-warping approach. Furthermore, this last fact also means reducing the hardware resource cost for its potential implementation in digital hardware devices such as GPUs, ASICs, or FPGAs. In the discussion, we analyze the speedup of the multiresolution approach compared to the multiscale-with-warping scheme. For an FPGA implementation, we obtain a reduction of latency between 40% and 50% and a resource reduction of 30%. The final solution copes with large-range motion estimations with a simplified architecture very well-suited for customized digital hardware datapath implementations as well as current multicore architectures.

Keywords:
Image motion analysis, Optical flow, FPGA, Architectures for embedded systems, Real-time systems

1. Introduction

Optical flow estimates the bidimensional velocity of each pixel as the projection of the real tridimensional velocity of the points on the object surfaces of the scene onto the image plane. Motion estimation is extensively used in applications such as image segmentation [1][2], autonomous robot navigation [3], video surveillance [4][5], or driving assistance systems [6][7].

We implement the Lucas & Kanade (L&K) algorithm [8], particularly the model described in [9][10]. Due to its local computation, this algorithm efficiently estimates velocity for small ranges, but not for large inter-frame displacements (the typical range is a few pixels [7]). A lot of works overcome this problem by applying a multiscale extension with motion compensation or scale-by-scale warping, also called coarse-to-fine approach with warping, to improve the accuracy and the motion range of the system (inspired in [11]). From now on in this paper, we will simply refer to the multiscale-with-warping approach as multiscale. Its main drawback is its computational cost, which means a high resource cost in a dedicated device as we will shown later.

It is important to remark that the monoscale approach, correctly parameterized, is perfectly suitable for many applications that do not demand quite accurate results but real-time computation. For instance, on autonomous navigation, the collision avoidance may be addressed on controlled scenarios with large obstacles and agents traveling at moderated speeds. The problem is that parameters for real scenarios are not a priori known and therefore, monoscale methods fail in many situations. As explained, multiscale approaches solve this problem thanks to the motion warping operation throughout spatial scales (a feasible solution but with a significantly higher computational cost). Furthermore, as the process is sequentially performed scale by scale, using the previous computed results as the input to the subsequent scale computation, its parallelization is rather hard and the system has an important latency due to that intrinsically sequential nature.

We propose a novel approach to avoid the high-hardware cost of the multiscale-with-warping approach which takes advantage of the good behavior of monoscale approaches. These monoscale approaches estimate accurately the object motion when the appropriate spatial scale is tuned (inspired in previous works such as [12][13]).This new method computes several estimations for different filter sizes (scales) in the same way as it is performed for the multiscale method. Then, the key step is the *parallel* combination of all these single-scale estimations to obtain a final optical flow result (in contrast with the *sequential* multiscale-with-warping approach). As will be shown, this approach requires a much simpler architecture and is able to tune optical flow estimations for different object sizes and motion ranges. We propose a fusion function based on a weighted average of the several estimations and also a hardware-friendly version that is the alternative that we will use.

In a second part, in order to fulfill the real-time requirements, we need a device which is capable of exploiting the maximum parallelism level. Nowadays, optical flow estimation is the main challenge of a large number of works that use dedicated hardware architectures [14], graphic processing units (GPUs) [15][16], and even optimized software implementations [17] or clusters of processors [18]. Previous works have proposed FPGA-based solutions as in [19][20][21] as well. In our discussion, we initially propose a multicore architecture (available in any standard PC) and analyze the scalability of the solution. Secondly, we present the resource cost of an implementation in an FPGA and compare it with the multiscale solution, confirming a substantial resource saving.

This paper is structured as follows: in Section 2, we summarize the L&K algorithm, the multiscale-with-warping extension, the multiresolution method, and the combination functions; Section 3 analyzes the accuracy benchmarking of the proposed alternatives and the comparison with the multiscale-with-warping method; and Section 4 finally presents the conclusions.

2. Lucas & Kanade algorithm for optical flow computation

The gradient-based L&K method [9] is one of the most accurate and efficient approaches in terms of computational complexity (see [22][7]). Our implementation is based on [23][24].

We estimate the optical flow assuming the brightness constancy, i.e. the Optical Flow Constraint (OFC) (1): we assume that the intensity of the pixel in (x, y) at time t is the same as its intensity in $(x + u, y + v)$ at $t + 1$.

$$I(x, y, t) = I(x + u, y + v, t + 1) \tag{1}$$

The OFC expression (1) is a nonlinear equation which can be solved by a first order Taylor expansion leading us to (2)

$$I_x u + I_y v + I_t = 0 \tag{2}$$

where I_x, I_y, and I_t are the partial derivatives of the intensity. As (2) does not have a unique solution, we need an additional assumption to find it. There are several approaches to solve it: some methods assume a constant flow in the same local neighborhood and compute the optical flow of a pixel by focusing only on its neighborhood; some other local methods require the constancy of the image derivatives too; finally, global methods assume the global smoothness of the optical flow, which means that the optical flow for one pixel depends on the values of the rest. The Lucas & Kanade algorithm is a widely used local gradient-based approach based on the image derivatives. Its advantages are its simplicity, efficiency, and potential parallelization, which makes it very suitable for hardware implementations [24][22].

We estimate the velocity (u, v) for each pixel $\mathbf{x}$ as a weighted least-squares solution of (2). The velocity is computed by minimizing (3)

$$\sum_{\mathbf{x}\in\Omega}(W_i^2(\mathbf{x})(I_x(\mathbf{x})u + I_y(\mathbf{x})v + It(\mathbf{x}))^2) \tag{3}$$

with $W_i^2(\mathbf{x})$ being a weighting matrix of pixels in Ω neighborhood, that gives more weight to the values of pixels closer to the center than to those in the surround. And then, solving (3) as shown in (4), with $A = [\nabla I(\mathbf{x}_1), ..., \nabla I(\mathbf{x}_n)]^T$, and $b = -(I_t(\mathbf{x}_1), ..., F_t(\mathbf{x}_n))^T$ for $\mathbf{x}_1, ..., \mathbf{x}_n \in \Omega$.

$$(u, v)^T = (A^T W^2 A)^{-1} A^T W^2 b \tag{4}$$

$$(A^T W^2 A) = \begin{bmatrix} \sum_{\mathbf{x}\in\Omega} W^2(\mathbf{x}) I_x^2(\mathbf{x}) & \sum_{\mathbf{x}\in\Omega} W^2(\mathbf{x}) I_x(\mathbf{x}) I_y(\mathbf{x}) \\ \sum_{\mathbf{x}\in\Omega} W^2(\mathbf{x}) I_x(\mathbf{x}) I_y(\mathbf{x}) & \sum_{\mathbf{x}\in\Omega} W^2(\mathbf{x}) I_y^2(\mathbf{x}) \end{bmatrix} \tag{5}$$

$$(A^T W^2 b) = \begin{bmatrix} \sum_{\mathbf{x}\in\Omega} W^2(\mathbf{x}) I_x(\mathbf{x}) I_t(\mathbf{x}) \\ \sum_{\mathbf{x}\in\Omega} W^2(\mathbf{x}) I_y(\mathbf{x}) I_t(\mathbf{x}) \end{bmatrix} \tag{6}$$

We compute the reliable estimations by rejecting those for which the determinant of (5) is lower than a threshold (a parameter in our model), while Barron [24] used the minimum of the eigenvalues of (5). In this way, we save hardware resources without losing significant accuracy (see [25][7]). Finally, in order to increase the accuracy of the optical flow estimation, we implement the hierarchical extension.

2.1. Hierarchical extension: Multiscale-with-warping implementation

The hierarchical extension [11] improves the dynamic range of the optical flow estimator, e.g. using 5 scales we increase the dynamic range up to 30x compared to the monoscale version.

The main modules for this hierarchical extension are:

- Gaussian pyramid computation [26]: It computes the Gaussian pyramid of the input frames (we use 3 frames). The pyramid levels depend on the number of scales and a previous filtering stage to reduce the aliasing effect produced by the subsampling process is also included.
- Expansion: This stage expands the current scale optical flow estimation to the resolution of the subsequent scale size (the sampling factor is 2).
- Optical flow estimation: This module computes the single-scale optical flow estimation.
- Warping: This stage computes the image warping with the computed optical flow estimation at the current scale by shifting images with these displacements. The warped images are the input of the subsequent optical flow estimation after being scaled to the next scale size in a coarse-to-fine grain scheme. The warping computation is shown in (7). The upsampling operator with factor 2 is Θ and $Warp$ stands for the warping operator which performs a bilinear interpolation.

$$I_w^s(\mathbf{x}) = Warp(I^s(\mathbf{x}), \Theta(V^{s-1}(\mathbf{x})) \tag{7}$$

- Merging: This stage adds the previous partial estimations and the new one for each iteration.

It is strictly mandatory to perform all these steps in a sequential way to properly extract the flow values of each pixel at each scale. The first step is the computation of the Gaussian pyramid and then, the iteration of the following stages as many times as scales: the expansion stage of the last estimation to the next scale size, the warping of the current estimation, and the correspondent frames computed by the pyramid module, the computation of the new estimation that receives as input the warped frames and finally, the sum of the new estimation and the previous one.

As previously commented, the warping operation is the most expensive one in terms of computational complexity. This explains that a lot of works in the literature avoid this operation in the multiscale computation [27][28]. In terms of hardware resources, it also means a high cost. In [19] and [29], it is shown that the warping stage (including the interface with memory) consumes up to 23% of the total hardware resources of a Virtex4 XC4VFX100 FPGA chip, while the module that computes the single-scale optical flow consumes 35%.

2.2. *Multiresolution optical flow estimation*

In this paper, we propose a novel approach for the estimation of optical flow: the multiresolution method. In contrast with the sequential scheme of the previous multiscale-with-warping model, the parallel nature of this new method makes it suitable for exploiting the maximum performance in current multicore architectures or in hardware reconfigurable devices. In the last case, it also means a fundamental saving of hardware resources.

In Fig. 1, we present a scheme for the two approaches: multiscale-with-warping and multiresolution. As shown, the multiscale-with-warping process is inherently sequential and includes the warping stage. On the other hand, the multiresolution scheme is naturally parallelizable, avoids the warping operation, and also includes the fusion stage that combines the estimations for the different spatial resolutions. The implementation of this stage is the main issue under study in Section 2.3.

The fundamental difference between both approaches is that the multiresolution method avoids the warping stage which is very expensive in terms of computational load. This stage iteratively compensates the input image for each scale with the optical flow estimation computed for the previous scale (after being upsampled to the current scale resolution). This compensation reduces the total movement range in the input images adapting it to the size of the optical flow core filters. For each scale, the algorithm computes a partial optical flow estimation for the compensated images and not for the original input images. Consequently, the total optical flow estimation is computed as the summation of all these partial estimations in the merge stage (see Section 2.1). Actually, this warping stage is replicated depending on the number of frames. In our case, we use 3 frames and thus, we perform the image warping for the previous and for the future frame (the current frame is not warped). Moreover, the warping stage also represents a bottleneck in terms of memory accesses which is crucial in the case of an on-chip hardware implementation.

Note that although we call our method multiresolution, this term is also used for methods that use of different sized filters. Our approach uses a multiscale pyramid to achieve high computational efficiency. As stated in [26], enlarging filter sizes or reducing images accordingly keeping filter sizes identical are equivalent operations, although the second choice is computationally much more efficient.

We do not intend to prove that our multiresolution method is better in terms of accuracy performance, but rather to show that it is able to smartly combine the estimates for different spatial scales and to efficiently compute a final estimation. In that way, this method is accurate enough for a lot of applications without very strong precision requirements such as tracking or obstacle avoidance, as for instance [30] [31]. The main goal of the application is the detection of objects that are moving in the scene and their global motion on unconstrained environments (without object size or velocity range limitations) [32] [33] [34]. Indeed, accurate motion estimations are not required because we mainly use blob-based processing in this kind of applications. In this case, our method is perfectly suited, very competitive, and accurate. A last additional advantage of our model parallelization is the substantial global system latency reduction (since the iterative mechanism required for the multiscale-with-warping approach is not required any more). Latency is fundamental, for instance, in collision scenarios, where reducing latency increases the reaction time for potential collisions (it estimates the time-to-contact faster). This provides larger response times for real-world applications such as collision avoidance or driver warning mechanisms. Note that for each frame, latency typically increases about 40ms for these applications.

The algorithm in this case consists of the following stages:

- Gaussian pyramid: Same as in the previous method.
- Optical flow estimation: We estimate the motion for all the images computed in the last step. In such a way, we simulate the optical flow estimation with different filter sizes. It allows us to tune filters for matching different spatial frequencies. It is fully parallel, all the estimations are computed at the same time. It is worth noticing that for objects whose motion range does not match the filter range for that spatial scale, estimations would be unreliable, but this will be detected by reliability estimates (to know more about these issues, cf. [7]).
- Expansion: In this case, we scale in parallel the estimations computed in the previous step to the original image size before the fusion.
- Fusion: This final stage combines all the estimations to obtain a final optical flow result. In this work, we will focus on this point by benchmarking the different possibilities and analyzing their performances.

As shown, this implementation does not need the warping and merging stages, but we perform the combination step.

2.3. *Fusion functions*

In this subsection, we address the development of the fusion function for the scale-by-scale motion estimations. The objec-

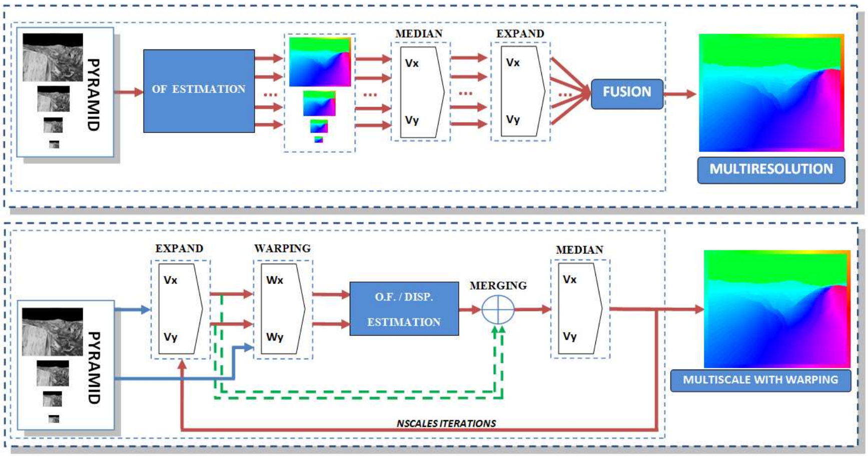


Figure 1: Multiresolution (top) and multiscale-with-warping (bottom) approaches for optical flow estimation. Note the difference between the *parallel* (multiresolution) and *sequential* (multiscale-with-warping) streams, the avoidance of the warping stage, and the extra combination stage.

tive is to overcome the main problems of the monoscale estimation at each spatial scale: displacement out of scale range and/or lack of spatial structure on this scale at the neighborhood of the target pixel. Thus, we use techniques that give the lowest weights to high velocities and structureless areas at different scales. The fusion of different spatial resolutions partially solves the problem of low-range estimations as in the case of multiscale with warping. However, it requires to be able to find the scale at which the motion is accurately sampled (we need to measure a small motion but which is large enough to provide a good SNR, especially for fixed-point arithmetic implementations). The number of estimations (or number of different spatial resolutions) is crucial, since more scales mean a higher likelihood tuning a certain displacement. Errors in high-range estimations and the borders of images strongly affect the final error measurement. The thresholds for the motion vector norm and the avoidance of the image borders allow us to control this impact. Finally, the confidence of the estimations is based on the use of the product of the eigenvalues of (11). In other words, it is based on the use of non-ill-conditioned structure information as shown in [25][7]. This fact motivates the use of structure tensors to give more weight to the areas with a high spatial (or spatio-temporal) structure.

Next, we propose a function for the fusion of estimates but that is presented incrementally. The first step presents a basic version that uses spatial tensors. The final function is presented as a second step, that adds the use of spatio-temporal tensors computed in a hardware-friendly way, and the mentioned thresholds for high velocities and areas with a strong 3D structure. As will be presented in Section 3, the final version achieves the best results in terms of accuracy.

The general fusion function is represented in (8):

$$O_f(\mathbf{x}) = \frac{\sum_{s=1}^{nscales} \Psi(W(s)V(\mathbf{x}, s))}{\sum_{s=1}^{nscales} \Psi(W(s))} \tag{8}$$

where $W(s)$ is the weighting function, $V(\mathbf{x}, s)$ is the velocity estimation computed for each pixel at scale s, Ψ is the scaling operator, and $O_f(\mathbf{x})$ is the final optical flow estimation. The scaling operator is used to scale all the estimations to the same resolution and also, to multiply all of them by the appropriate weight of $\mathbf{n}^{s-1}$ (this $\mathbf{n}$ factor means that a velocity at scale s is perceived as an absolute speed on the next scale equal to its value multiplied by the image resolution scaling factor). The use of a weighted normalized sum has been used for a long time in the literature as seen in [35][36][37]. Moreover, it is a very simple and efficient combination of different estimations.

The proposed versions for these weighting functions are listed below:

- The first function (W_1) consists in a Gaussian average of the estimations for each scale. The Gaussian gives more weight to the coarsest-scale estimations than to the finest ones because the risk of motion out of filter range is less likely in the case of the first ones. The Gaussian is normalized before its application to the scale-by-scale estimations and the standard deviation (σ) is 4.66.

  $$K(s) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-s^2}{2\sigma^2}} \tag{9}$$

  This weighting function that penalizes high velocities and gives more weight to the coarsest-scales is supported by [24][38].

  The second part of the equation consists in multiplying the previous result by an exponential factor of the 2D structure tensor as in (10).

  $$W_1(\mathbf{x}, s) = K(s) e^{\sqrt{\theta_{2D}(\mathbf{x},s)}} \tag{10}$$

  $T_{2D}(\mathbf{x}, s)$ is the 2D structure tensor for scale s defined as shown in (11). $\lambda_1(\mathbf{x}, s)$ and $\lambda_2(\mathbf{x}, s)$ are the $T_{2D}(\mathbf{x}, s)$ eigenvalues and $\theta_{2D}(\mathbf{x}, s)$ is the product of the eigenvalues of (11) as shown in (12). The structure tensors are defined as in (11) using the products of the partial derivatives.

  The use of the product of the eigenvalues of (11) is based on the use of non-ill-conditioned structure information as shown in [25][7]. Using the product of the eigenvalues is equivalent to the use of the determinant of the structure tensor.

$$T_{2D}(\mathbf{x}, s) = \left[\begin{array}{cc} I_x^2(\mathbf{x}, s) & I_x(\mathbf{x}, s)I_y(\mathbf{x}, s) \\ I_x(\mathbf{x}, s)I_y(\mathbf{x}, s) & I_y^2(\mathbf{x}, s) \end{array} \right] \quad (11)$$

$$\theta_{2D}(\mathbf{x}, s) = \lambda_1(\mathbf{x}, s)\lambda_2(\mathbf{x}, s) \quad (12)$$

The tensor structure is computed as products of image derivatives of the original image (after a Gaussian smoothing filter application) computed in a local neighborhood.

In (10), the goal of multiplying the product by the exponentiation of $\theta_2 D(\mathbf{x}, s)$ is to reward the estimations for the local areas with a richer structure, this is motivated by the low reliability of the flow estimation in structureless areas as shown in the literature [39] [40].

- In the final version (W_2), the weighting function consists in the average of the estimations multiplied by an approximation to the 3D tensor (it represents the extension of the 2D tensor to the temporal component). Spatio-temporal structure tensors have been widely used in the literature and, especially, applied to motion estimation [41][42]. Its computation is similar but now, the new tensor is computed as shown in (14). Confidence strongly depends on the spatial or, in this case, spatio-temporal structure which motivates the use of this tensor as in the case of the 2D tensor.

  The 3D tensor is approximated with (14) because the product of the eigenvalues of (12) or in fact, the determinant of (11), extended with the temporal component, is a hardware-unfriendly computation (it requires 6 sum and 12 product operations that are very expensive in terms of hardware resources).

  Furthermore, $||V(\mathbf{x}, s)||$ stands for the motion norm, τ_{θ_s} is the threshold for the new structure tensor, and τ_{V_s} is the threshold for the velocity module at scale s. As the areas with very strong 3D structures (spatio-temporal) and large velocities that represent large and very fast movements are prone to the highest errors, we penalize these regions. At the same time, thresholding operations guarantee a minimum of structure and velocity at the target scale.

  This new $\theta_{sim}(\mathbf{x}, s)$ tensor approximates the behavior of the 3-D structure tensor; it is computed as a square root of a weighted sum of the product of partial derivatives $I_x^2(\mathbf{x}, s)$, $I_y^2(\mathbf{x}, s)$ and $I_t^2(\mathbf{x}, s)$, using the spatial and temporal information without computing the eigenvalues. This approach is novel and, as we will see in the next section, presents the best trade-off between accuracy-density and computational cost.

$$W_2(\mathbf{x}, s) = \begin{cases} K(s)e^{\sqrt{\theta_{sim}(\mathbf{x},s)}} & \theta_{sim}(\mathbf{x}, s) > \tau_{\theta_s} \quad \&\& \\ & (||V(\mathbf{x}, s)|| > \tau_{V_s}) \\ 0 & otherwise \end{cases} \quad (13)$$

$$\theta_{sim}(\mathbf{x}, s) = \sqrt{\mu_1(I_x^2(\mathbf{x}, s) + I_y^2(\mathbf{x}, s)) + \mu_2 I_t^2(\mathbf{x}, s)} \quad (14)$$

  In (14), μ_i are the weights for spatial and temporal components terms in order to parameterize this simulated tensor and to penalize or to reward the spatial $(I_x^2(\mathbf{x}, s), I_y^2(\mathbf{x}, s))$ or temporal $(I_t^2(\mathbf{x}, s))$ component terms. In our case, we weight them with 2/3 and 1/3 respectively, since we want to keep a balance between the spatial and the temporal structures. As in the last case, the objective of the function is to penalize large velocities that represent large movements, but it gives more weight to the spatio-temporal structure with the new tensor. In this way, we simplify the computation without using the velocity module. Moreover, the thresholding operations ensure a minimum of structure and velocity.

  Finally, taking into account the resource utilization, exponentials, divisions, or the computation of the eigenvalues entail the use of a huge amount of resources. In our alternative, all these operations are avoided by the use of a linear combination and a simple square root, saving a significant amount of resources.

In addition to these combination methods of single-scale estimations for the optical flow, we have added some improvements to the previously described algorithm: the implementation of the pyramid with a sub-sampling factor of $\sqrt{2}$ instead of 2, this means a finer granularity for the filter sizes along the scales to cover the same range of displacements, and the use of median filters as post-processing operations to homogenize the individual estimations of each scale.

3. Results and discussion

In this Section, we present and discuss the results of our proposal. The first step is showing the comparison of our work with respect to the state of the art. As shown in the first subsection, we achieve good results compared to the listed works. However, as mentioned before, our aim is not to obtain a very accurate implementation but an implementation that presents a good trade-off between accuracy and efficiency. The second part is focused on showing that the multiresolution method is not a mere selection of the estimation for the proper scale according to the filter sizes that better tune the sequence. In fact, we show in this Section that, in general, the fusion is always more accurate than the best single-scale estimation.

Next, we also compare the multiresolution and the multiscale-with-warping approaches to show that our method achieves very accurate results using a combination of single-scale estimations, but with its corresponding loss of density (the key point will be to limit this density drop keeping a high accuracy; otherwise, having a high density is a trivial task but increases the error of the motion fields). Finally, in the last point of this subsection, we reinforce it applying our implementation to a real-world sequence. We segment parts of this sequence and test the accuracy for the whole frame and the segmented regions, showing that there are no important differences between the multiscale-with-warping and the multiresolution. On the other hand, there are substantial differences with respect to the single-scale estimation.

	AAE (Dens)	Architecture	Algorithm
Our multires. approach	6.64° (47.70%)	Core 2 Duo (2600 MHz)	LK multires.
Our multisc. approach	7.23° (89.34%)	Core 2 Duo (2600 MHz)	LK multisc.
Botella (2010) [44]	5.5° (100%)	Xilinx V2	McG
Mahalingam (2010) [45]	6.37° (38.6%)	Xilinx V2 Pro (45 MHz)	LK mono
Tomasi (2010) [29]	7.91° (92.01%)	Xilinx V4 (45 MHz)	Phase-based multi
Tomasi (2010) [46]	11.8° (63%)	Xilinx V4 (50 MHz)	Phase-based mono
Tomasi (2010) [19]	4.69° (82.81%)	Xilinx V4 (45 MHz)	Phase-based multi
Anguita (2009) [17]	3.79° (71.8%)	Quad Core 2 Q9550 (2830 MHz)	L&K
Gwosdek (2009) [47]	5.73° (NP)	Cell Processor, PS3	Variational
Pauwels (2008) [15]	2.09° (63%)	NVIDIA GeForce 8800 GTX	Phase-based
Diaz (2008) [7]	7.86° (57.2%)	Xilinx V2 (82 MHz)	L&K
Bruhn (2006) [48]	5.77° (100%)	Intel Pentium 4 (3.06GHz)	Variational
Correia (2002) [49]	10.44° (40.9%)	MV200 [50]	L&K

Table 1: Accuracy comparison with previous works (sorted by publication date) with the error for the *Yosemite* ($b1$) sequence.

The last subsection evaluates the speedup of our implementation with respect to the multiscale-with-warping and includes a brief summary of the hardware resources of its potential hardware implementation.

The benchmark sequences used in our analysis are available at the well-known Middlebury database [43] that also provides their ground-truth: *Yosemite*, *Otte-Marble*, *Diverging tree*, *Translation tree*, *Grove2*, *Grove3*, *Hydrangea*, *Rubber-Whale* (we call them $b1$ to $b8$ in this work). We benchmark the sequences using the AAE (Average Angular Error, in degrees) the error metric proposed in [24]. As our implementations are sparse, we do not achieve estimations for each pixel; due to this fact, we also show their density.

3.1. State of the art

In this first subsection, we summarize a brief comparison with the state-of-the-art works for the optical flow comparison. Table 1 shows the accuracy for several implementations in the literature (including hardware-based designs) for the *Yosemite* sequence ($b1$). In this rank, our work achieves a good position. Our approach is outperformed by, for instance, the variational methods but achieves a competitive precision compared to the Lucas & Kanade based implementations except for [17]. That work achieves the best results because it is using a very large temporal filter which means that may obtain very good results in temporally-continuous sequences as the Yosemite fly-through. On the other hand, its performances might be negatively affected if using with any other sequences with different properties.

Note that some implementations are embedded in a dedicated hardware, which may restrict their accuracy due to constraints in computing precision (the use of fixed-point arithmetics with constrained bit-width). It is important to remark that our objective is to obtain a hardware-friendly implementation for the estimation of optical flow with a constrained precision rather than obtaining the best implementation in terms of accuracy. The combination function for our multiresolution implementation in this table is the one that obtains the best results in the following subsection (W_2).

We also show Table 2, where we present the results for all our benchmark sequences. As it can be seen, most of the works only provide results for the *Yosemite* sequence. The comparison between our multiscale-with-warping and multiresolution methods is carried out in Section 3.4.

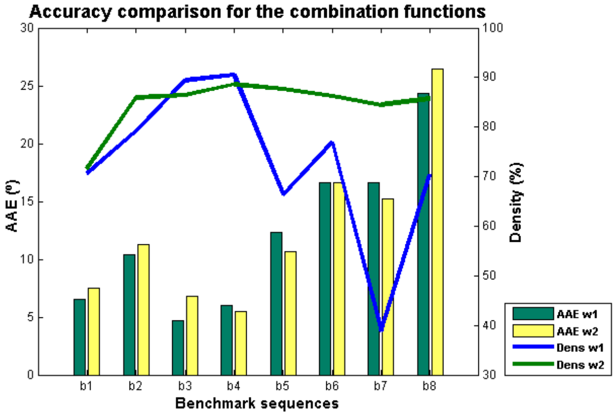


Figure 2: Accuracy analysis for the optical flow implementation for the fusion functions (W_1 and W_2). Columns show the AAE (bars, referred to left axis) and density (lines, referred to right axis) for the fusion functions.

3.2. Benchmarking of the fusion function

This Section benchmarks the alternatives explained in the previous Section for the fusion function, comparing the two incremental versions: the basic function and the final version.

As seen in Fig. 2, W_1 and W_2 achieve similar results in terms of accuracy. Concerning density, the W_2 alternative obtains the best results; in the best case (for sequence $b7$), the improvement in density is higher than 40%, with an error increase of less than 2 degrees. As seen, the selected final version W_2 achieves a significant improvement taking into account density, and its selection is also motivated by its hardware feasibility for the computation of its simplified tensor.

3.3. Multiresolution implementation results

Once the fusion function (W_2) is developed and benchmarked, this Section firstly presents some qualitative results for our multiresolution method (Fig. 3). As the fusion stage performs a combination of the single-scale estimations for each scale, in the second part of this subsection, we show that our combination is better than just selecting the best single-scale estimation (in fact, it should be at least as good as the best single-scale estimation). Actually, our combination tunes the best spatial scale according to the object sizes and speeds taking advantage of all these spatial scales.

Fig. 3 illustrates the results for different benchmark sequences (*Yosemite* ($b1$), *Otte Marble* ($b2$), *diverging tree* ($b3$), *translation tree* ($b4$), and *Hydrangea* ($b7$)). It shows the qualitative results (right) for these sequences, including the ground-truth (central) and the original central frame (left). The multiresolution results have been post-processed using a median filter to homogenize the results at each direction independently. We have also performed a row-wise technique filling the holes with an average estimation (taking the pixel itself, the two previous ones, and the two subsequent ones).

As mentioned, Fig. 4 shows the accuracy for each single-scale estimation compared with the best alternative for the combination stage (we selected the W_2 approach). As displayed, the

	Yosemite	Otte Marble	Diverging Tree	Translation Tree	Grove2	Grove3	Hydrangea	Rubber Whale
Our multires. approach	6.64° (47.7%)	7.91° (60.16%)	4.56° (43.33%)	4.81° (54.73%)	8.36° (41.49%)	16.39° (31.67%)	15.00° (19.45%)	13.73° (45.95%)
Our multisc. approach	7.13° (74.87%)	16.39° (99.57%)	7.18° (99.92%)	4.55° (100%)	7.69° (99.30%)	13.70° (99.36%)	11.76° (94.05%)	18.27° (98.98%)
Botella (2010) [44]	5.5° (100%)	NP	NP	NP	NP	NP	NP	NP
Mahalingam (2010) [45]	6.37° (38.6%)	NP	5.51° (49.1%)	2.94° (41.0%)	NP	NP	NP	NP
Tomasi (2010) [29]	7.91° (92.01%)	NP	NP	NP	NP	NP	NP	NP
Tomasi (2010) [46]	11.8° (63%)	NP	6.63° (98%)	NP	NP	NP	NP	NP
Tomasi (2010) [19]	4.69° (82.81%)	NP	NP	NP	NP	12.08° (92.62%)	NP	11.2° (79.09%)
Anguita (2009) [17]	3.79° (71.8%)	NP	3.4° (76.2%)	0.85° (74.5%)	13.0° (54.1%)	NP	NP	24.0° (39.7%)
Gwosdek (2009) [47]	5.73° (NP%)	NP	NP	NP	NP	NP	NP	NP
Pauwels (2008) [15]	2.09° (63%)	NP	NP	NP	NP	NP	NP	NP
Diaz (2008) [7]	7.86° (57.2%)	NP	NP	NP	NP	NP	NP	NP
Bruhn (2006) [48]	5.77° (100%)	NP	NP	NP	NP	NP	NP	NP
Correia (2002) [49]	10.44° (40.9%)	NP	7.64° (51.8%)	7.07° (41.3%)	NP	NP	NP	NP

Table 2: Accuracy comparison with previous works (sorted by publication date) with the error and density for all the benchmark sequences ($b1$ to $b8$).

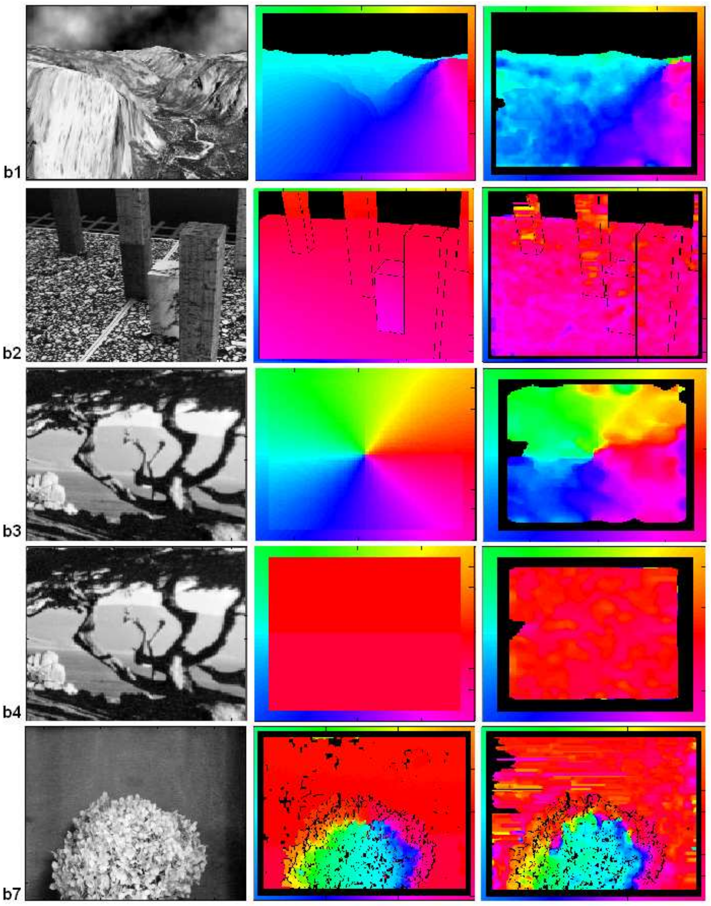


Figure 3: Multiresolution optical flow results for benchmark sequences ($b1$ to $b4$ and $b7$). The left column shows the original central frame of each sequence, the central image represents the optical flow ground-truth with a color-coding describing the direction of the motion according to the frame of the picture and the right image represents the multiresolution optical flow estimations with the same color-coding.

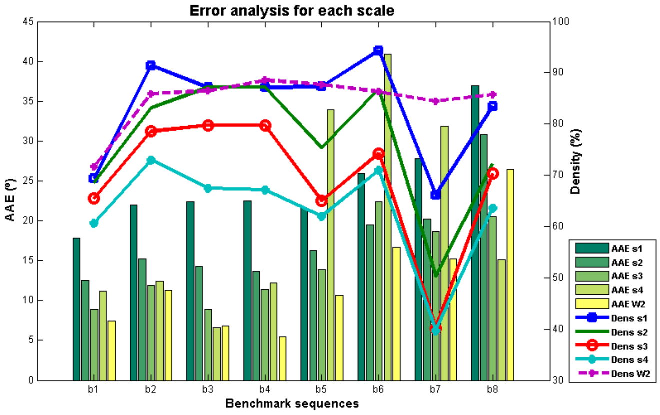


Figure 4: Accuracy analysis for the optical flow implementation for each scale. Columns 1 to 4 show the AAE (referred to the left axis) for scales 1 to 4 and column 5 shows the AAE for the multiresolution implementation with the W_2 alternative. Lines show the density (referred to the right axis).

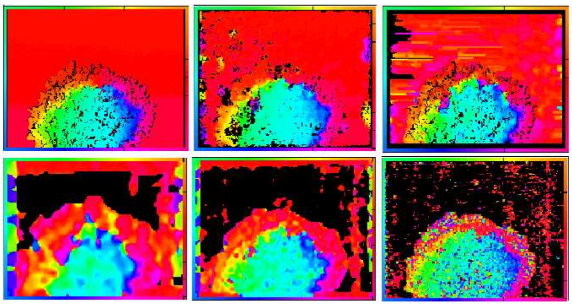


Figure 5: Comparison of optical flow results for the Hydrangea sequence ($b7$). The first row illustrates (from left to right): ground-truth, multiscale-with-warping approach result, and multiresolution result. The second row shows the monoscale estimations for the 1^{st}, 2^{nd}, and 4^{th} scale. All the estimations have a color-coding frame describing the direction of the motion.

column distribution shows the presence of sets of pixels tuning the filter sizes (spatial and temporal). The fifth column of each subset shows the AAE and the density for the best alternative that combines the estimations for each scale. In this way, the results for the combination function are better for $b1$, $b2$, and from $b4$ to $b7$ than any individual result for a specific spatial resolution scale, improving in the best case 40% in AAE (for $b4$) with respect to the best single-scale estimation and losing only 5% of density. In the case of $b8$, the result is slightly worse than for scales $s4$ and $s3$. The main reason is that we give more weight to the values of the coarsest scales and in this case, the best tuning is achieved with high-speed ranges (which means that the coarsest scales estimate worse than the finest ones). Finally, the management of the unreliable values (see Section 2) is essential to achieve an improvement in the computation of efficient values and to reduce the sparseness of our final results. Regarding density, the W_2 version gives us always about 85% of valid values, except for the first sequence. For this figure, we have computed AAE and density using the same threshold for the all the versions and sequences.

Fig. 5 shows the comparison of the different estimations using the multiscale-with-warping, the monoscale, and the multiresolution approaches and besides, the estimations for scales 1, 2, and 4. As seen, the best qualitative results are provided by the multiscale approach, but the comparison with the multiresolution results reveals a moderate difference which is affordable in many real-world applications. The comparison with the monoscale and the results for scales 2 and 4 show that they are insufficient for the computation of accurate optical flow estimations caused by the filter sizes and the lack of refinement processes.

3.4. Multiresolution Vs. multiscale-with-warping benchmarking

After developing our multiresolution method, we show in this subsection a comparison of the results obtained for the multiscale-with-warping and the multiresolution methods. In this subsection, we reinforce the idea of using our implementation for real-world applications that need a limited accuracy allowing a moderated loss of density but with a constrained resource cost.

The comparison between the multiscale-with-warping and the multiresolution approaches is displayed in Fig. 6. As shown, the multiresolution approach achieves similar results in terms of error compared with the multiscale-with-warping, except for the $b8$ sequence. This error is obtained with a reduction in the density of about 10% to 15%. In this case, in order to make a fair comparison, we use the same implementation of the Lucas&Kanade core, using the same thresholds and a unique post-processing median filter to regularize the results. All these results reinforce the suitability of our algorithm for real-world applications that do not have strong requirements in terms of accuracy but that do have them in terms of resources as for example in [30] [31]. Both works deal with driving assistance systems for overtaking, integrated in real systems with limited resources. Moreover, in both cases their blob-based methods successfully solve the problem. Our work may represent a good candidate for such a kind of application since it achieves even better accuracy performance.

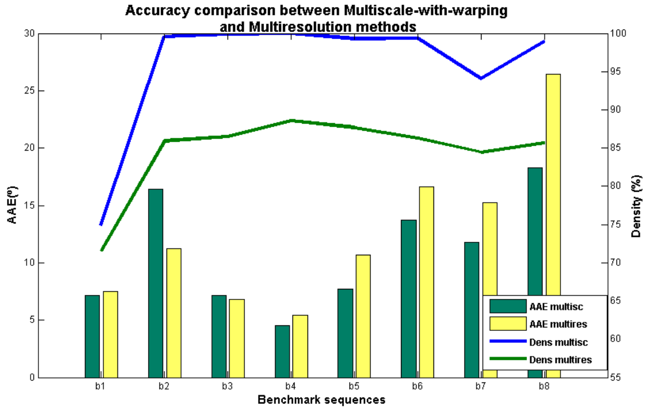


Figure 6: Comparison between our multiscale-with-warping and multiresolution approaches. Columns 1 and 2 for each graphic show the AAE (see left axis) for the multiresolution and the multiscale methods using the W_2 alternative with the optimized implementation. Lines show the density (see right axis). We present different results according to the energy threshold and the border sizes (indicated).

Additionally, we also show that our algorithm leads to a significant reduction in resources, as shown in Section 3.5.

Our proposed implementation is presented as a low-cost computation of the optical flow for applications with not very demanding accuracy requirements as explained in Section 2. In Fig. 8, we apply our algorithm and the multiscale-with-warping approach to a driving scene sequence [51]. The result comparison between the multiscale and the multiresolution approaches shows not very important qualitative differences, and in comparison with the computations for the scale by scale (from scale 1 to 5), we appreciate a considerable improvement. Fig. 7 presents the AAE results for the same sequence. We have segmented 2 regions which include two cars, a small car that is far from our point of view (region A), and a big car very close to us (region B). Once we have segmented these regions, we compute the AAE for these regions and for the complete frame. Multiscale and multiresolution AAE results are the best ones and the difference is unsubstantial in the case of the complete frame (about 1°). Focusing on region A (the small car moving quickly from left to right), the best match is obtained for scale 4 (a finer scale tunes better with an object of this size). In the case of a different spatial resolution object (the car included in region B), we achieve similar results for scales 2, 3, and 4 (coarser scales for a bigger object). The AAE decrease for the complete frame is about 25% comparing it with the best tuning using the monoscale approach (2^{nd} scale).

3.5. Towards an efficient implementation

Our new proposal for optical flow estimation exploits the maximum level of parallelization in order to implement it in embedded software, DSPs, or multicore architectures. In this section, we present the speedup of our implementation with respect to the multiscale-with-warping approach and we finally

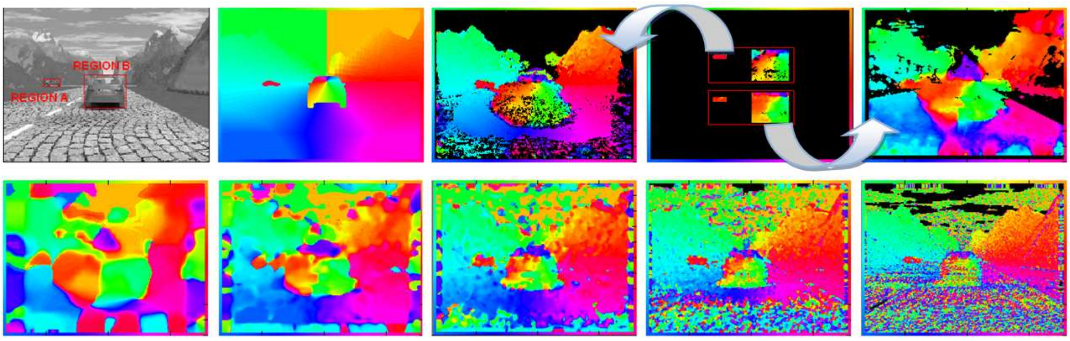


Figure 8: Multiresolution and multiscale-with-warping approach results using a car sequence [51] compared with the scale-by-scale computation. First row, from left to right: Original central frame (segmenting Region A for the furthest car and B for the nearest one), ground-truth, multiscale results, comparison of the segmented objects (Region A and Region B) for both, multiresolution and multiscale-with-warping approaches and finally, the multiresolution results in the fifth column. Second row: results for the scale-by-scale computation (scales 1, 2, 3, 4, and 5).

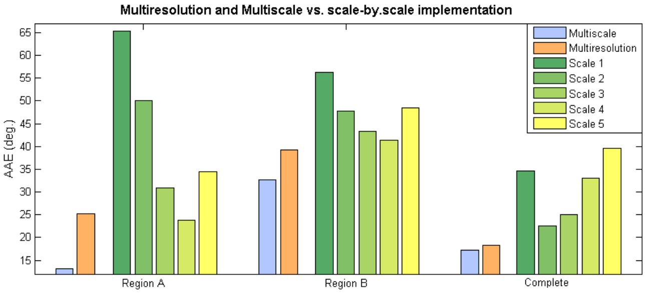


Figure 7: Error analysis of the multiresolution, multiscale-with-warping, and scale-by-scale methods for a car sequence [51]. The first two rows represent the multiscale and multiresolution results and the following 5, the results for the scale-by-scale computation (scales from 1 to 5). We have represented the results for Region A, Region B (indicated in Fig. 7), and the complete frame.

present a brief summary of its resource utilization using an FPGA as target computing device.

Fig. 9 shows the CPU time speedup in an Intel Pentium Core 2 Duo at 2.66 GHz. Both approaches have been implemented in C (with common modules for the Gaussian pyramid computation, optical flow estimation, and expansion) and using warping (for the multiscale approach) and the combination module (for the multiresolution one). The obtained value is the result of dividing the CPU time measured for the multiresolution approach by the CPU time for the multiscale one for each benchmark sequence (speedup). In the worst case, the gain is about 40% and in the best case, the CPU time saving is almost 55% with respect to the multiscale implementation. Moreover, we remark that in this case, we do not take advantage of the inherent parallelism of our approaches. A multi-threaded implementation of our method on a two-core machine allows us a computation time reduction of 30% more (by simply using MPI); therefore, this is not only a simplification of the computational resources but also a new possibility to more efficiently parallelize the algorithm.

A fully-parallel implementation represents a substantial improvement for an FPGA-based architecture in comparison with an inherently sequential one (as the multiscale-with-warping architecture). The hardware implementation of the proposed algorithm is out of the aims of this work; nevertheless, we estimate the hardware resource that would be obtained in comparison with the multiscale approach, considering the hardware consumption of the Lucas & Kanade implementation from a previous work [21].

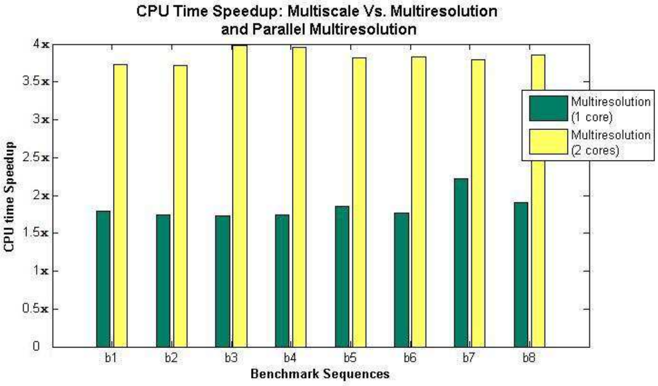


Figure 9: Comparison between multiresolution and multiscale-with-warping optical flow estimations, in terms of speedup CPU time. Time is measured in relative terms assuming the time for the multiscale computation to be 1 unit. We compare the implementation with a single core and a parallel implementation using a multicore architecture (2 cores in this case).

As seen in Table 3, the total system can save more than 30% of the resources that we use for the multiscale-with-warping approach (in our case, about 12% of our FPGA resources). This estimation was computed considering that we do not need the warping computation and the median filtering cascade (we use only one tap of median filtering). Finally, the final architecture is much simpler than the multiscale one. Furthermore, we also have to take into account that the merging module is not a simple sum; we have implemented a version of this function and the utilization is about 4%. The rest of resource utilization data have been extracted from a previous work in which the multiscale approach was implemented on a dedicated hardware [21] and from the implementation of the combination function.

Module	Hardware Resource: 4 Input LUTs (out of 84352)
L&K estimation core	4589 (5%)
Board Interface	4774 (5%)
Warping + Interface	9943 (11%)
Up-sampling	413 (1%)
Merging	364 (1%)
Median filtering	1675 (2%)
Combination function	3515 (4%)
Multiscale system	31796 (37%)
Estimated Multiresolution system	≈ 21500 (25%)

Table 3: Estimation of the resource utilization for our implementation for a Xilinx Virtex4 XC4VFX100.

4. Conclusions

In this work, we have designed and implemented a novel approach for the optical flow estimation. There are plenty of alternatives for the implementation, but there are very few choices for an efficient hardware implementation. Our approach aims to maintain a good trade-off between accuracy and resource utilization and also searches for an implementation with inherent parallel characteristics to be exploited with hardware accelerators. In artificial vision, the real-time computation of the vision algorithms in a real-life environment is essential for their use in applications such as tracking, scene understanding, video-surveillance, robot navigation, etc. Its implementation needs computationally efficient methods with parallelizable characteristics to exploit its maximum potential. In addition, reduced latency is essential for a wide range of applications that require interaction with the world in real-time (for instance robot manipulation or collision avoidance in driving situations). The proposed method allows a substantial reduction not only of the system computing time but also, a reduction of the system latency, both thanks to the inherent parallel architecture. This represents an important improvement compared with the sequential nature of the multiscale-with-warping method.

For the single-scale optical flow estimation, we selected the well-known L&K method due to its good accuracy vs. cost trade-off [22]. In order to improve accuracy, most of the works in the literature use the multiscale-with-warping approach but, as a hardware friendly alternative, we propose our multiresolution method. The multiscale method is shown to be the best alternative in terms of accuracy, since it refines the single-scale estimations, but not in terms of computational complexity (being also inherently sequential). Our proposal, the multiresolution method, combines different estimations varying the filter stage to compute the final optical flow. We propose a function for the fusion stage of our new method and benchmark it. This function achieves the best results in terms of accuracy and density, and for its selection we also considered its hardware implementation feasibility (which represents an essential improvement due to the simplicity of the implementation of our approach for a hardware device).

In our implementation, we also benchmark our approach in comparison with the multiscale-with-warping algorithm to evaluate accuracy. Firstly, we compare our implementation with some of the last implementations for different architectures, concluding that our implementation is very competitive, obtaining similar precision results to the other implementations (even better for the hardware-based architectures). In the comparison with the multiscale-with-warping method, we achieve an increase in the average AAE for all the benchmarked sequences of only 1.66°, with an average drop in density of 11.12%. On the other hand, we achieve a CPU time speed-up larger than 3.5x using a standard-PC architecture with 2 cores, or a reduction of 30% in resources for a hardware FPGA implementation, with a latency reduction of 40-50%. This reduction is due to the elimination of the warping circuit, some of the regularization stages, and to the simplification of the final architecture. In addition, due to the reduction in the processing time, we may compute the motion at a higher rate (improving in this way the optical flow accuracy due to a higher temporal sampling rate) or share some hardware logic to reduce the required resources even more.

As future work, due to the resource reduction, we will explore complementary on-chip image features that can be integrated in a single-chip architecture for more complex image applications. As explained, we will also explore the application of this method to collision avoidance applications.

Acknowledgments.

This work has been supported by the National Grant (AP2007-00275), projects ARC-VISION (TEC2010-15396), MULTIVISION (TIC-3873), and ITREBA (TIC-5060), and the EU grant TOMSY (FP7-270436).

References

[1] K. Nakayama, J. M. Loomis, Optical velocity patterns, velocity-sensitive neurons and space perception, Perception 3 (1974) 63 – 80.

[2] Y. R. Huang, C. M. Kuo, F. C. Huang, Block-based motion field segmentation for video coding, Journal of Visual Communication and Image Representation 16 (2005) 668 – 687.

[3] A. Giachetti, M. Campani, V. Torre, The use of optical flow for the autonomous navigation, in: Proceedings of the third European conference on Computer vision, volume 1 of *ECCV '94*, Springer-Verlag New York, Inc., 1994, pp. 146 – 151.

[4] X. Ji, Z. Wei, Y. Feng, Effective vehicle detection technique for traffic surveillance systems, Journal of Visual Communication and Image Representation 17 (2006) 647 – 658.

[5] D. Y. Chen, P. C. Huang, Motion-based unusual event detection in human crowds, Journal of Visual Communication and Image Representation 22 (2011) 178 – 186.

[6] H. Frenz, M. Lappe, M. Kolesnik, T. Buehrmann, Estimation of travel distance from visual motion in virtual environments, ACM Trans. Applied Perception 4 (2007) 419 – 436.

[7] J. Diaz, E. Ros, R. Agis, J. Bernier, Superpipelined high-performance optical-flow computation architecture, Computer Vision and Image Understanding 112 (2008) 262 – 273.

[8] T. Brox, From pixels to regions: partial differential equations in image analysis, Ph.D. thesis, Saarland University, 2005.

[9] B. D. Lucas, T. Kanade, An iterative image registration technique with an application to stereo vision (1981) 674 – 679.

[10] B. Horn, B. Schunck, Determining optical flow, Artificial Intelligence 17 (1981) 185 – 203.

[11] J. Bergen, P. Anandan, K. Hanna, R. Hingorani, Hierarchical model-based motion estimation, in: G. Sandini (Ed.), Computer Vision ECCV'92, volume 588 of *Lecture Notes in Computer Science*, pp. 237 – 252.

[12] A. Darabiha, J. Rose, J. Maclean, Video-rate stereo depth measurement on programmable hardware, in: Computer Vision and Pattern Recogni-

tion, 2003. Proceedings. 2003 IEEE Computer Society Conference on, volume 1, pp. 203 – 210.

[13] T. Gautama, M. Van Hulle, A phase-based approach to the estimation of the optical flow field using spatial filtering, Neural Networks, IEEE Transactions on 13 (2002) 1127 – 1136.

[14] Y. Murachi, Y. Fukuyama, R. Yamamoto, J. Miyakoshi, A vga 30-fps realtime optical-flow processor core for moving picture recognition, IEICE Trans. on Electronics E91 (2008) 457 – 464.

[15] K. Pauwels, M. M. V. Hulle, Realtime phase-based optical flow on the GPU, in: Computer Vision and Pattern Recognition Workshops, CVPRW '08. IEEE Computer Society Conference on, pp. 1 – 8.

[16] K. Pauwels, M. Tomasi, J. Diaz Alonso, E. Ros, M. Van Hulle, A comparison of FPGA and GPU for real-time phase-based optical flow, stereo, and local image features, Computers, IEEE Transactions on In Press (2011).

[17] M. Anguita, J. Diaz, E. Ros, F. J. Fernandez-Baldomero, Optimization strategies for High-Performance computing of Optical-Flow in General-Purpose processors, Circuits and Systems for Video Technology, IEEE Transactions on 19 (2009) 1475 – 1488.

[18] T. Kohlberger, C. Schnorr, A. Bruhn, J. Weickert, Domain decomposition for variational optical-flow computation, Image Processing, IEEE Transactions on 14 (2005) 1125 – 1137.

[19] M. Tomasi, M. Vanegas, F. Barranco, J. Diaz, E. Ros, High-Performance Optical-Flow architecture based on a multiscale, Multi-Orientation Phase-Based model, Circuits and Systems for Video Technology, IEEE Transactions on 20 (2010) 1797 – 1807.

[20] J. Diaz, E. Ros, F. Pelayo, E. M. Ortigosa, S. Mota, FPGA-based real-time optical-flow system, Circuits and Systems for Video Technology, IEEE Transactions on 16 (2006) 274 – 279.

[21] F. Barranco, M. Tomasi, J. Diaz, M. Vanegas, E. Ros, Parallel architecture for hierarchical optical flow estimation based on FPGA, Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 6 (2012) 1058 – 1067.

[22] H. Liu, T.-H. Hong, M. Herman, R. Chellappa, Accuracy vs. efficiency trade-offs in optical flow algorithms, in: Computer Vision ECCV 96, volume 1065 of *Lecture Notes in Computer Science*, Springer Berlin, Heidelberg, 1996, pp. 174 – 183.

[23] S. S. Beauchemin, J. L. Barron, The computation of optical flow, ACM Computing Surveys 27 (1995) 433 – 466.

[24] J. L. Barron, D. J. Fleet, S. S. Beauchemin, Performance of optical flow techniques, Int Journal of Computer Vision 12 (1994) 43 – 77.

[25] J. W. Brandt, Improved accuracy in Gradient-Based optical flow estimation, Int. J. Comput. Vision 25 (1997) 5 – 22.

[26] P. J. Burt, E. H. Adelson, The Laplacian pyramid as a compact image code, IEEE Transactions on Communications 31 (1983) 532 – 540.

[27] C. Cassisa, S. Simoens, V. Prinet, Two-frame optical flow formulation in an unwarping multiresolution scheme, in: Proceedings of the 14th Iberoamerican Conference on Pattern Recognition: Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications, CIARP '09, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 790 – 797.

[28] F. Steinbruecker, T. Pock, D. Cremers, Large displacement optical flow computation without warping, in: IEEE International Conference on Computer Vision (ICCV), Kyoto, Japan.

[29] M. Tomasi, M. Vanegas, F. Barranco, J. Diaz, E. Ros, A novel architecture for a massively parallel low level vision processing engine on chip, in: Industrial Electronics (ISIE), 2010 IEEE International Symposium on, pp. 3033 – 3039.

[30] P. Guzmán, J. Díaz, J. Ralli, R. Agís, E. Ros, Low-cost sensor to detect overtaking based on optical flow, Machine Vision and Applications 1 (????) 1–13.

[31] J. Diaz Alonso, E. Ros Vidal, A. Rotter, M. Muhlenberg, Lane-Change Decision Aid System Based on Motion-Driven Vehicle Tracking, Vehicular Technology, IEEE Transactions on 57 (2008) 2736 – 2746.

[32] H. Qian, X. Wu, Y. Ou, Y. Xu, Hybrid algorithm for segmentation and tracking in surveillance, in: Robotics and Biomimetics, 2008. ROBIO 2008. IEEE International Conference on, pp. 395 – 400.

[33] Y. Benabbas, N. Ihaddadene, T. Yahiaoui, T. Urruty, C. Djeraba, Spatio-temporal optical flow analysis for people counting, in: Advanced Video and Signal Based Surveillance (AVSS), 2010 Seventh IEEE International Conference on, pp. 212 –217.

[34] M. Kristan, S. Kovacic, A. Leonardis, J. Pers, A two-stage dynamic model for visual tracking, Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on 40 (2010) 1505 –1520.

[35] I. Laptev, T. Lindeberg, On space-time interest points, in: International Conference on Computer Vision, pp. 432 – 439.

[36] J. Alison, Noble, Finding corners, Image and Vision Computing 6 (1988) 121 – 128.

[37] Ralph, Hartley, A Gaussian-weighted multiresolution edge detector, Computer Vision, Graphics, and Image Processing 30 (1985) 70 – 83.

[38] J. Weber, J. Malik, Robust computation of optical flow in a multi-scale differential framework, International Journal of Computer Vision 14 (1995) 67–81.

[39] A. Motten, L. Claesen, Low-cost real-time stereo vision hardware with binary confidence metric and disparity refinement, in: Multimedia Technology (ICMT), 2011 International Conference on, pp. 3559 – 3562.

[40] M. Ye, R. Haralick, L. Shapiro, Estimating optical flow using a global matching formulation and graduated optimization, in: International Conference on Image Processing, volume 2, pp. 289 – 292.

[41] J. Wright, R. Pless, Analysis of persistent motion patterns using the 3d structure tensor, in: Motion and Video Computing, 2005. WACV/MOTIONS '05 Volume 2. IEEE Workshop on, volume 2, pp. 14 –19.

[42] H. Liu, R. Chellappa, A. Rosenfeld, Accurate dense optical flow estimation using adaptive structure tensors and a parametric model, Image Processing, IEEE Transactions on 12 (2003) 1170 – 1180.

[43] Middlebury computer vision, http://vision.middlebury.edu/, 2010.

[44] G. Botella, A. Garcia, M. Rodriguez-Alvarez, E. Ros, U. Meyer-Baese, M. C. Molina, Robust bioinspired architecture for optical-flow computation, IEEE Trans. Very Large Scale Integr. Syst. 18 (2010) 616 – 629.

[45] V. Mahalingam, K. Bhattacharya, N. Ranganathan, H. Chakravarthula, R. Murphy, K. Pratt, A VLSI architecture and algorithm for LucasKanade-Based optical flow computation, Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 18 (2010) 29 – 38.

[46] M. Tomasi, F. Barranco, M. Vanegas, J. Diaz, E. Ros, Fine grain pipeline architecture for high performance phase-based optical flow computation, Journal of Systems Architecture 56 (2010) 577 – 587.

[47] P. Gwosdek, A. Bruhn, J. Weickert, Variational optic flow on the Sony PlayStation 3, Journal of Real-Time Image Processing 5 (2010) 163 – 177.

[48] A. Bruhn, J. Weickert, T. Kohlberger, C. Schnrr, A multigrid platform for Real-Time motion computation with Discontinuity-Preserving variational methods, Int. J. Comput. Vision 70 (2006) 257 – 277.

[49] M. Correia, A. Campilho, Real-time implementation of an optical flow algorithm, in: 16th International Conference on Pattern Recognition, volume 4, pp. 247 – 250.

[50] J. R. Vallino, Datacube MV200 and ImageFlow User"s Guide, Technical Report, Rochester, NY, USA, 1995.

[51] T. Vaudrey, C. Rabe, R. Klette, J. Milburn, Differences between stereo and motion behaviour on synthetic and real-world stereo sequences, in: Image and Vision Computing New Zealand, 2008. IVCNZ 2008. 23rd International Conference, pp. 1 –6.

3 Active gaze control system

3.1 Vergence control for robotic platform based on vector disparity

The journal paper associated to this part of the dissertation is:

- F. Barranco, J. Díaz, A. Gibaldi, S.P. Sabatini, E. Ros, Vector disparity sensor with vergence control for active vision systems. Sensors, vol. 12, no. 2, pp. 1771-1799, 2012, DOI 10.3390/s120201771.
    - Status: **Published**.
    - Impact Factor (JCR 2010): 1.774
    - Subject Category:
        - Chemistry, Analytical. Ranking 38 / 73.
        - Electrochemistry. Ranking 16 / 26.
        - Instruments and Instrumentation. Ranking 14 / 61.

Sensors **2012**, *12*, 1771-1799; doi:10.3390/s120201771

OPEN ACCESS

sensors

ISSN 1424-8220

www.mdpi.com/journal/sensors

Article

Vector Disparity Sensor with Vergence Control for Active Vision Systems

Francisco Barranco [1,⋆], Javier Diaz [1], Agostino Gibaldi [2], Silvio P. Sabatini [2] and Eduardo Ros [1]

[1] Department of Computer Architecture and Technology, CITIC, ETSIIT, University of Granada, C/Daniel Saucedo Aranda s/n, E18071, Granada, Spain; E-Mails: jdiaz@atc.ugr.es (J.D.); eduardo@atc.ugr.es (E.R.)

[2] PSPC Group, Department of Biophysical and Electronic Engineering (DIBE), University of Genoa, Via Opera Pia 11A, I-16145, Genoa, Italy; E-Mails: agostino.gibaldi@unige.it (A.G.); silvio.sabatini@unige.it (S.P.S.)

⋆ Author to whom correspondence should be addressed; E-Mail: fbarranco@atc.ugr.es; Tel.: +34-95-824-1775; Fax: +34-95-824-8993.

Received: 28 December 2011; in revised form: 18 January 2012 / Accepted: 7 February 2012 / Published: 9 February 2012

Abstract: This paper presents an architecture for computing vector disparity for active vision systems as used on robotics applications. The control of the vergence angle of a binocular system allows us to efficiently explore dynamic environments, but requires a generalization of the disparity computation with respect to a static camera setup, where the disparity is strictly 1-D after the image rectification. The interaction between vision and motor control allows us to develop an active sensor that achieves high accuracy of the disparity computation around the fixation point, and fast reaction time for the vergence control. In this contribution, we address the development of a real-time architecture for vector disparity computation using an FPGA device. We implement the disparity unit and the control module for vergence, version, and tilt to determine the fixation point. In addition, two on-chip different alternatives for the vector disparity engines are discussed based on the luminance (gradient-based) and phase information of the binocular images. The multiscale versions of these engines are able to estimate the vector disparity up to 32 fps on VGA resolution images with very good accuracy as shown using benchmark sequences with known ground-truth. The performances in terms of frame-rate, resource utilization, and accuracy of the presented approaches are discussed. On the basis of these results, our study indicates that the gradient-based approach leads to the best trade-off choice for the integration with the active vision system.

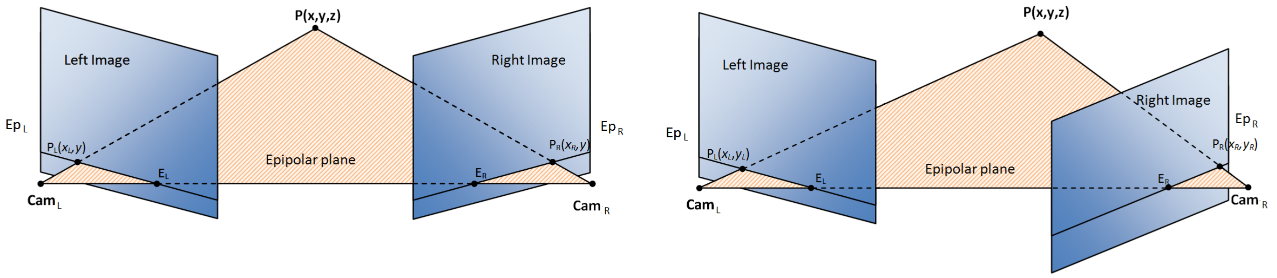
Keywords: field programmable gate arrays; active vision; real time systems

1. Introduction

Depth perception is essential for an autonomous system that is moving in a dynamic environment. It is applied in multiple applications such as autonomous navigation [1], obstacle detection and avoidance [2], 3-D reconstruction [3], tracking [4,5], grasping [6,7], *etc.* For example, depth computation is used for obtaining 3D information of real-world scenarios with real-time implementations such as [8,9]. Many computer vision algorithms have been proposed for extracting depth information from multiple-view images [10,11].

An alternative for depth estimation is based on the correspondences of image features of a binocular system. Disparity computation models basically consist of a matching problem, finding correspondences between features or areas in both left and right images and taking into account some constraints based on the geometry of the stereo rig that may help to simplify the search process (due to the epipolar geometry constraint [10]). This model assumes that the geometry of the cameras is fixed and known (e.g., using a previous calibration stage that allows us to extract the intrinsic and extrinsic binocular cameras parameters [10]). Based on this assumption, in most of the literature contributions, the disparity is managed as a mono-dimensional matching problem of correspondences. They usually assume that images are calibrated or include a pre-processing stage for undistortion and rectification. This process consists of correcting the lens radial distortions and aligning the image planes with the epipolar lines, which simplifies the disparity estimation, reducing the problem to a matching only for the horizontal coordinates (see Figure 1). Nevertheless, the main problem is that the rectification process is required each time the camera configuration is modified. Therefore, this technique is not suitable for any active system that is able to modify the vergence of the cameras, which means that we need a different approach.

Figure 1. Epipolar geometry for a pair of cameras. **Left**: P corresponds to (x_L, y) and (x_R, y) coordinates for the left and right image planes respectively. **Right**: The same P point corresponds to different x and y coordinates on each image plane. Ep_L and Ep_R are the epipolar lines, Cam_L and Cam_R, the camera optical centers, and E_R and E_L stand for the epipoles.



In our case, we select a system that adaptively controls the vergence of its binocular camera setup. The use of a static vergence angle cannot yield to a correct gazing position to a specific point. However, a solution for this problem is to use a vergence control that allows the modification of the relative position of the cameras in order to fixate a point. This bio-inspired solution achieves the best disparity results around the fixation point because it moves the image plane at the depth of this point and computes a very small range of disparities around it. In general, indeed, the highest accuracy of the disparity calculation kernel is obtained with the lowest disparity ranges.

On the other hand, our active binocular sensor acts depending on the target, which means that its fixation point is determined by three parameters: version, tilt, and vergence angles (see Section 2). There are some previous works that support the use of the disparity for guiding vergence eye movements in active vision systems [12–14]. The listed advantages allow us to explore the scene in detail, by changing gazing and fixation at different targets or areas of interest. In this work, we present the implementation of a system where vision and motion control act in a collaborative manner for solving the presented issue. With respect to the motion control strategy, our work is based on a separate control of version and vergence inspired by the Hering's law [15]. Hering proposed that the movement of one eye is coordinated with the other with a movement of equal amplitude and velocity, but it could be in opposite directions. This coordinated movements are specified in terms of version and vergence components. However, eyes only approximate this law, because saccadic movements may include additional vergence components [16]. More details are provided in the following subsection.

In the literature, we find different works that deal with the vector disparity estimation problem [17–19]. In this work, we propose a comparison between a gradient-based [20,21] and a phase-based [22–24] approach for the disparity computation. Phase-based approaches are more accurate and robust against variations in the illumination. On the other hand, gradient-based approaches may be implemented at a reduced cost, saving resources in comparison with the first one and presenting a good accuracy.

FPGA is selected as the platform for our system because of the requirements of real-time performances and the suitability of the integration with the active vision system due to the reduced chip size and limited power consumption compared with other approaches as the one based on GPUs. For vision processing algorithms with a high-computational complexity, we need to exploit the maximum parallelism at different levels and in our case, this objective is matched using a fine-pipeline based architecture in an FPGA. In the case of the single-scale version of our disparity computation, we achieve up to 267 and 118 fps (frames per second) for the gradient- and the phase-based algorithms, respectively. For the multi-scale implementations, the reached frame rate is almost 32 fps. In both cases, the image resolution is VGA (640 × 480), although higher resolutions are possible at the cost of reducing the frame rate.

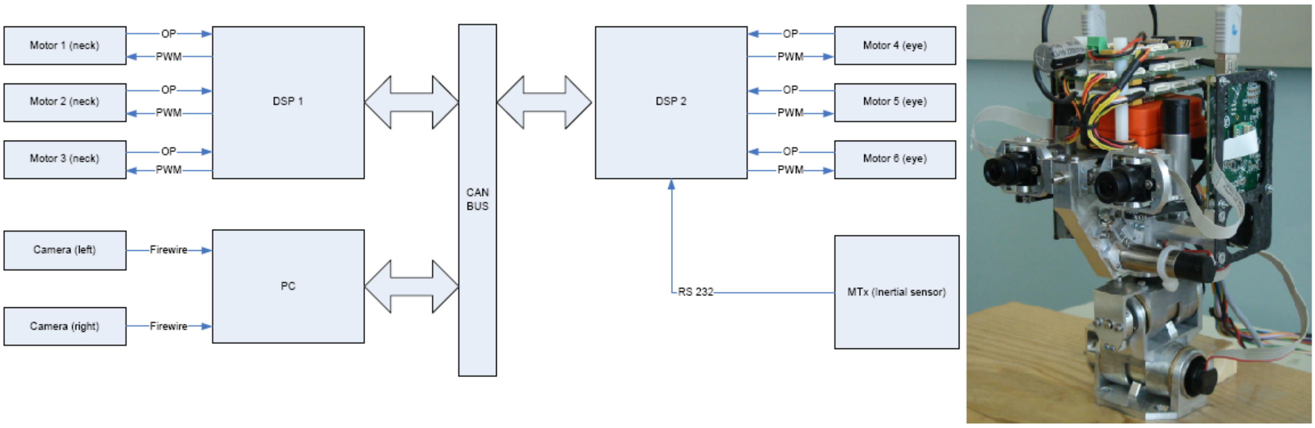
This paper is structured as follows: in Section 2, we present the vergence, tilt, and version control for our sensor; in Section 3, we describe the mathematical formulation of the used approaches for the disparity computation (gradient-based and phase-based); Section 4 analyzes the hardware implementations for both gradient-based and phase-based approaches detailing their mono- and multi-scale versions. This section also presents the performance analysis (accuracy and frame rate) and the resource utilization. Section 5 shows the motion control strategy for our binocular system and the integration of the system with the disparity computation. Finally, Section 6 presents the conclusions and the future work.

2. Version, Tilt, and Vergence Control

In this section, we present a system that manages the vergence, version, and tilt control with a stereo pair of cameras to fixate a target object. This is possible thanks to the utilization of a vector disparity model and a camera setup endowed with a shared motor for the tilt control, and two motors for the independent control of the pan angle of each camera. The separate controls allow us to move the gaze toward the point of interest and to control the vergence angle to fixate the target.

The experiments described in Section 5 were performed using the iCub head platform designed by the RobotCub Consortium [25]. Figure 2 displays the real RobotCub head and the architecture scheme. This platform is composed by a stereo pair of FireWire cameras, two DSPs, and an inertial sensor connected to a PC. The communication between the elements and the PC is carried out through a CAN bus. The DSP controllers use Motorola 56F807 16-bit hybrid processors. The head consists of a binocular system with six degrees of freedom: the eyes have two rotational degrees of freedom but with a common tilt; the neck also has three degrees of freedom. The control of cameras and neck motors is performed separately by two DSP units. The main problem with such a platform is the sensitivity of the motors to the low velocities, or different friction for the motors which might cause oscillations in the fixation task. Finally, slower responses of the system yield more predictable results. More information about the mechanics is provided in [26,27]. Finally, cameras provide pairs of 15 fps with a 1,024 × 768 resolution, although we are able to compute up to 32 fps of 640 × 480 pixels of resolution, we crop and subsample the resolution to 320 × 240 for our experiments because it provides good results for the vergence control algorithm. Our disparity computation system is based on an adaptable architecture that allows us to set the image resolution using input parameters. In this way, it is possible to adopt the most appropriate resolution for the target application. A smaller resolution for the images in our experiments allows the tracking of the objects in a very controlled scenario, and helps us to avoid any possible problem that may be caused by the rapid movements of the head.

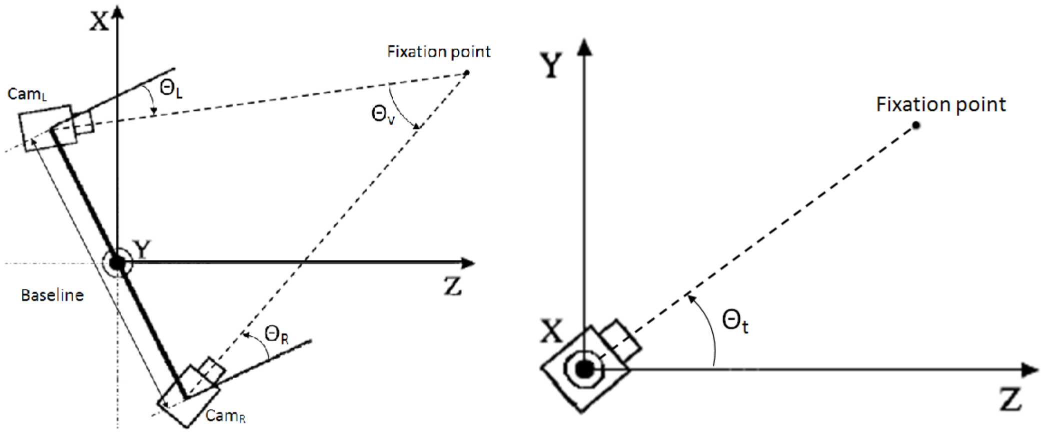
Figure 2. Hardware architecture of the iCub head. On the left, the architecture with the two DSPs, the PC connected to the CAN bus, and the inertial sensor are shown. An image of the head is shown on the right.



The voluntary fixation of the visual system is directly related with the fovea, the central area of the retinas, that provides high-resolution visual stimuli. In fact, in human vision, fixation movements have the task to register the target into the foveae, in order to maximize the perceptible details in the area of interest [28]. Version is defined as the rotation of the eyes about the vertical axis to maintain a constant disparity. Meanwhile, tilt is the rotation of each eye with respect to the horizontal axis. Finally, vergence is the rotation of each eye about the vertical axis to change the disparity. These parameterization of the system allows to define a unique fixation point with three angles [29]. In Figure 3, we observe the top and side views of our camera setup. In the figure, Θ_v defines the vergence angle, Θ_L and Θ_R stand for the pan angles (version) and finally, Θ_t is the tilt angle (displayed on the right). As we mentioned before, tilt angles are common for both right and left cameras. Due to the correlation between the angles, if the vergence angle is defined, we only need either Θ_L or Θ_R to define version.

Version might be determined monocularly using a master eye (camera in our case) for the gazing to the target object and the slave camera performs a vergence movement to fixate the point. In this case, we only have two separate controls. However, vergence has to be defined in terms of binocular disparity.

Figure 3. Version, vergence, and tilt angles for the stereo vision system. Top and side views of the camera configuration showing the version, vergence, and tilt angles for each camera.



The selected joint control method combines version, vergence, and tilt as independent parallel movements. The performed control can be expressed by the following set of equations in Equation (1).

$$\begin{aligned} \Theta_{version} &= K_1(\mathbf{x}_L + \mathbf{x}_R) \\ \Theta_{tilt} &= K_2(\mathbf{y}_L + \mathbf{y}_R) \\ \Theta_{vergence} &= K_3 f(\mathbf{d}) \end{aligned} \tag{1}$$

where $(\mathbf{x}_L, \mathbf{y}_L)$ and $(\mathbf{x}_R, \mathbf{y}_R)$ are the coordinates of the target for the left and right images regarding the current fixation and $f(\mathbf{d})$ is a function of the disparity for the fixation point for almost similar vertical disparities that only avoids nullifying the vergence angle for the zero disparity and gives the disparity in other case. Finally, K_1, K_2 and K_3 are tuning gains. The area close to the fixation point is defined as "zero disparity" region. Vergence control allows us to fixate the gazed target by searching for this region.

We assume the computation of real-time disparity for the implementation of our control model. A fast vergence control can be computed using the information of the disparity of the target when the gazing is changing continuously, and our system is trying to reduce it to "zero" at this point.

There are also some different alternatives such as performing the fixation with several fixed setups for the cameras, constraining them to a set of more likely depths for the fixation point. This approach seems to save resources, since it avoids the computation of the real vector disparity (the undistortion and rectification stage has to be done, but the implementation is easy since the setups are fixed and known "a priori"). As our approach is also applied to autonomous mobile vehicles, vibrations may affect the setup, therefore a periodic evaluation of the rectification would be needed. From this perspective, the strategy of fixed setups is unsuccessful, since the rectification steps keep being mandatory. Another alternative may consist of using several fixed cameras with different vergence configurations, completely avoiding the rectification stage. This approach is similar to the previous one but it entails a high economic cost, which makes it unsuitable for a lot of applications.

3. Comparison of Vector Disparity Algorithms

The binocular disparity is usually defined as the difference in the x coordinates between the right and left images of a binocular vision system. This definition is useful because, even if in real-world applications the images are uncalibrated, binocular disparity systems usually include a preprocessing rectification stage that compensates for the difference in the y coordinate. This operation is related to the geometry of the cameras, and needs to be recomputed each time that the relative position of the cameras is modified. If the rectification is not applied (for instance, because the relative position of the cameras is continuously changing as in active vision systems), the disparity becomes a 2D problem, as we will discuss in the next subsections.

Disparity estimation techniques can be grouped into local and global methods. Local methods are centered in the surrounding neighborhood of a pixel to estimate its disparity. Global methods take into account the complete image. In our case we implement two different alternatives: a gradient-based technique, the well-known local algorithm of Lucas and Kanade [20,21] and a phase-based one detailed in [23] (also a local algorithm). The first technique estimates small local disparities assuming the intensity or brightness constancy of a pixel between left and right images, while the second one computes the disparity using the phase information for different orientations, in a contrast-independent way. In order to increase the working range for disparity detection, we use the multi-scale extension of the algorithms. Both implementations have been developed from the horizontal-disparity implementations (1D) but conveniently extended for the vector disparity (2D).

In order to validate our approach, we compare both the methods in terms of their different advantages and drawbacks. While the gradient-based method provides a good trade-off between efficiency and resource utilization, phase-based methods are very robust against variations in illumination and shadows, which makes them specially appropriate for real-world applications. In this paper, we avoid the use of global algorithms because they are not suitable for the on-chip implementation, at least with an affordable resource cost. This also explains why most of the current embedded implementations of disparity computation engines are based on local approaches [28,30–34].

3.1. Gradient-Based Lucas–Kanade Disparity Model

The disparity is defined as a 2-D problem as shown in Equation (2):

$$I^{right}(x,y) = I^{left}(x + d_x, y + d_y) \tag{2}$$

where I^{right} and I^{left} are the intensity of right and left images respectively and d_x and d_y are the disparity components. The solution of the system in this case is similar to the case of the optical flow in [35,36]. Applying the Taylor expansion in Equation (2), we obtain Equation (3)

$$d_x I_x^{left} + d_y I_y^{left} + I^{left}(x, y) - I^{right}(x, y) = 0 \tag{3}$$

where I_x^{left} and I_y^{left} are the partial derivatives of the left image. With this ill-posed system, additional assumptions need to be considered. The Lucas–Kanade algorithm supposes that pixels in the same neighborhood correspond to the same object and therefore, have a similar disparity. And then, applying a least-square fitting for solving Equation (4) procedure, we obtain the system defined by Equations (5) and (6).

$$(d_x, d_y) = (A^T W^2 A)^{-1} A^T W^2 b \tag{4}$$

$$(A^T W^2 b) = \begin{bmatrix} \sum\limits_{i\in\Omega} W_i^2 I_{xi}^{left}(I_i^{left} - I_i^{right}) \\ \sum\limits_{i\in\Omega} W_i^2 I_{yi}^{left}(I_i^{left} - I_i^{right}) \end{bmatrix} \tag{5}$$

$$(A^T W^2 A) = \begin{bmatrix} \sum\limits_{i\in\Omega} W_i^2 (I_{xi}^{left})^2 & \sum\limits_{i\in\Omega} W_i^2 I_{xi}^{left} I_{yi}^{left} \\ \sum\limits_{i\in\Omega} W_i^2 I_{xi}^{left} I_{yi}^{left} & \sum\limits_{i\in\Omega} W_i^2 (I_{yi}^{left})^2 \end{bmatrix} \tag{6}$$

where W_i stands for the weighting matrix of the pixels in the neighborhood Ω. For the optical flow, Barron [20] computes the confidence of the computed estimation thresholding with the minimum of the eigenvalues of Matrix (6). We simplify it by using the determinant of this Matrix without a significant loss of accuracy, as shown in [36,37]. Finally, this algorithm also provides very computationally efficient solutions with a competitive accuracy as shown in [38,39].

3.2. Phase-Based Disparity Algorithm

Sabatini *et al.* proposed in [23] a multi-channel interaction algorithm to combine the phase information from multiple spatial scales (the multi-scale approach will be detailed in Section 3) and multiple orientations. The computation is performed combining multiple Gabor filter responses tuned at 8 orientations. Using the same formulation for Gabor filtering of [40], the vector disparity can be computed from the phase difference applying an intersection of constraint along different orientations, as in [41], assuming that points on an equi-phase contour satisfy $\phi(x, t) = c$, with c a constant. Differentiation with respect to time yields Equation (7):

$$\nabla\phi \cdot d + \psi = 0 \tag{7}$$

where $\nabla\phi = (\frac{\delta\phi}{\delta x}, \frac{\delta\phi}{\delta y})^T$ is the spatial phase gradient, $d = (d_x, d_y)^T$ is the vector disparity, and ψ is the phase difference between left and right images. The phase difference is computed without any explicit phase computation using the formulation proposed in [24]. In a linear model, the spatial gradient can be substituted by the radial frequency vector $(w_0 cos\theta_q, w_0 sin\theta_q)$ [23] where q indicates one of the eight orientations of the Gabor filter bank that we use. Next, Equation (7) can be rewritten as Equation (8)

$$w_0(cos\theta_q, sin\theta_q) \cdot d = -\psi_q \tag{8}$$

where · denotes scalar product. From this point, we can extract the component disparity and finally compute the disparity estimation solving the over-determined system defined in Equation (9)

$$d_x(\mathbf{x})w_0cos\theta_q + d_y(\mathbf{x})w_0sin\theta_q + \psi_q(\mathbf{x}) = 0 \tag{9}$$

3.3. Multi-Scale Generalization

As mentioned, our development is based on the multi-scalar generalization or coarse-to-fine scheme, inspired on Bergen's work [42], that increases the working range 30 times with respect to the mono-scalar implementation (using 5 scales). Moreover, it is a mandatory operation to achieve fully-operative systems on real-world scenarios. Our architecture is based on warping images, an approach that is usually avoided in the real-time embedded system literature because of its high resource costs (although it is more cost-effective than global methods).

The multi-scalar version implementation is simple. Firstly, we compute the pyramid for the input images (left and right) and they are stored (see [43]). In such a way, we have a bank of images sampled at different spatial resolutions (depending on the number of scales). The second step consists of iterating in a loop as many times as scales: for each iteration, we upsample the previous disparity estimation; then, we warp the previous upsampled results and the frames for the following finer scale; next, the new disparity estimation is computed using the previous results as input for the core; the final stage collects the new estimation and the partial previous, stored results to combine them in a new partial estimation for the next iteration. The first iteration only consists of the initialization, computes the estimation using as input the images computed in the pyramid for the first spatial resolution scale, and continues to the next iteration.

4. Hardware Implementation

The selected device for our hardware implementation is an FPGA. As mentioned in the introduction, the FPGA is a good candidate as a platform that allows us to exploit the maximum level of parallelism to achieve an embedded system able to work in real-time. Furthermore, the second key point is the possibility of integration with the motor control for our binocular system. The selected FPGA is a Xilinx Virtex4 chip (XC4vfx100). The board is a Xirca V4 [44] with a PCIe interface and four SRAM ZBT memory banks of 8 MB each one. This platform can work as a stand-alone platform or a co-processing board. This means that the platform can be used separately working alone or connected with a PC which facilitates the hardware implementation debugging and the result display.

Our super-scalar architecture was developed using fine-pipelined datapaths. The implementation of this ILP (Instruction Level Parallelism) provides high performances and low power consumption [36]. Our architecture is displayed as an adaptable design for different applications and requirements as is shown in some previous works [30,35,36,45]. The use of fixed-point arithmetic entails a high resource saving with an acceptable loss of accuracy (provided that the bit-width at different operations is carefully chosen).

Our design validation consists of evaluating the accuracy of this degradation with respect to a software floating-point version. We split the complete processing engine into different stages and test different bit-widths for the variables at each stage. The key point is to find the minimum bit-width that leads to a minimum loss of accuracy. Once our objective is reached, we continue with the following stage sequentially along the datapath. This process has been successfully adopted in other contributions as [30,35,36] and represents a good working methodology to allow the fixed-point implementation of floating point algorithms using digital hardware.

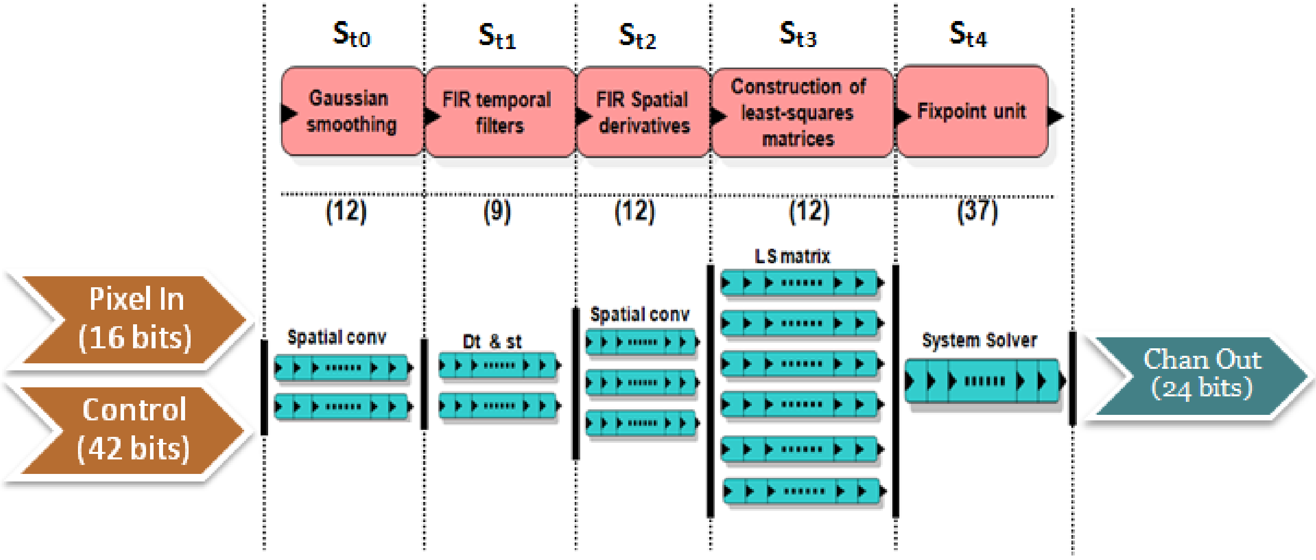
The development of the implementation was performed using two hardware description languages for two different abstraction levels. Firstly, the implementation of the modules that perform the communication protocols and interfaces and the Memory Controller Unit (MCU [46]) are implemented in VHDL. The disparity computation and the multi-scale architecture was developed using Handel-C because this C-like language is better suited for algorithmic descriptions, without significantly degrading performances or increasing resources [47].

The hardware implementation of the multiscale extension is described in detail [35,40,45]. This multi-scale-with-warping architecture is usually avoided in the literature because of the high computational costs that imply in hardware implementations high resource costs. On the other hand, the multi-scale architecture that we implemented allows us to increase the working range 30 times with respect to the mono-scale implementation.

4.1. Lucas–Kanade Vector Disparity Core

The implementation of this core is based on previous approaches [35,36,48]. In the cited works, the authors implement optical flow gradient-based algorithms, for both mono- and multi-scale versions. In this work, the most important difference is the computation of the vector disparity or bidimensional disparity for uncalibrated images instead of the 1-D disparity for rectified and undistorted images.

Figure 4. Scheme of the pipelined stages for the Lucas–Kanade vector disparity core. It describes the computation stages (from St_0 to St_4) indicating the pipelined stages (in brackets) and the number of parallel datapaths for each one of them.



In Figure 4, *Pixel In* denotes the input to the disparity core (in our case, 2 frames for the left and right images with a bitwidth of 8); *Control* represents the control word, with the parameters for the number of scales, confidence thresholds, and the input resolution. The output bitwidth is 24 bits: 12 bits for the vertical and the horizontal component of the disparity. The core is implemented in a segmented pipeline design with 5 stages:

- St_0: In this stage, the filtering of the inputs reducing the aliasing effects is performed. It convolves them with Gaussian filters whose kernels of 3 taps are $K = [1\ 2\ 1]/4$.
- St_1: It computes the left-right difference and applies a smoothing filter to the inputs.
- St_2: In this stage, we compute the spatial derivatives to the results of the previous stage: I_x, I_y, and filter again the results (including the left-right difference).
- St_3: This stage performs the calculation of the coefficients for the linear systems of 5 and 6. The weights W are set to a 5 × 5 separable kernel defined $W = [\ 1\ 4\ 6\ 4\ 1]/16$ as in [36,37].
- St_4: The final stage computes the solution of the system using the previous results. It uses as confidence measure the determinant of this matrix.

For the computation of this core, we use 187 parallel processing units: stage St_0 has 2 paths (2 frames) for the Gaussian convolution, St_1 has 2 paths (for the left-right difference and the smoothing), St_2 has 3 paths (2 for the derivatives I_x and I_y and 1 for the left-right difference), St_3 has 6 paths (one for each coefficient in Equations (5) and (6)) and St_5, has only one path for the system resolution. The number in brackets in the figure denotes the micropipelined stages for each of them.

4.2. Phase-Based Vector Disparity Core

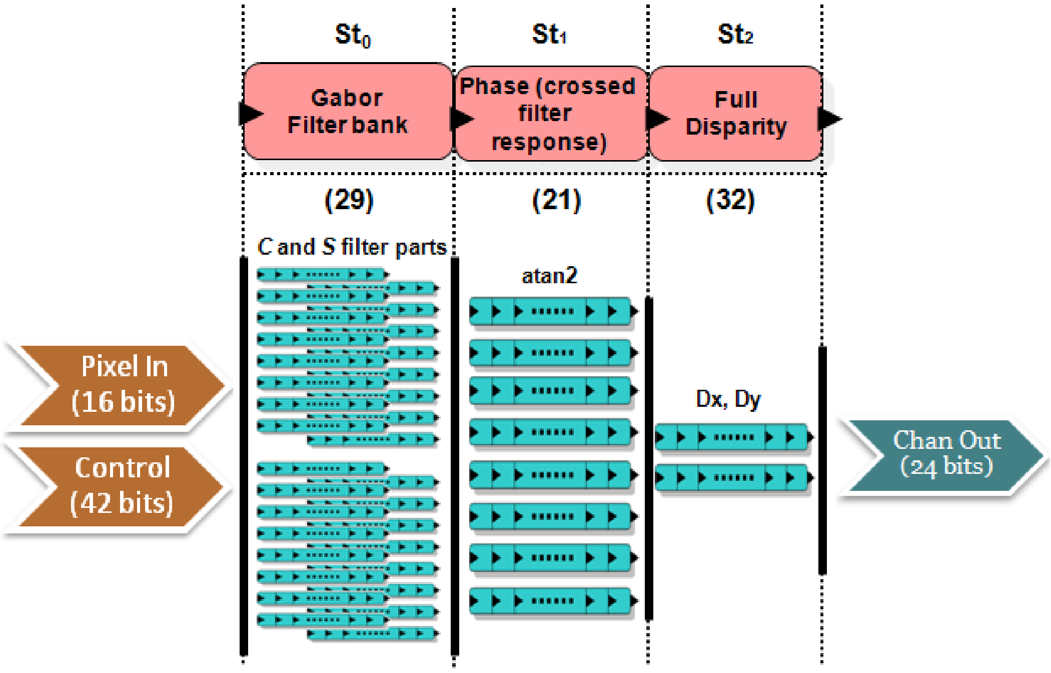
The design of this core is also based on a previous approach [30]. The main difference with respect to our case is that previous approaches implement a core for the computation of disparity for calibrated images, as in the previous core (as in [30,34]).

In Figure 5, we show a scheme of the design. Input parameters are the same as in the case of the previous core: *Pixel In* denotes the input data (2 frames for left and right images with 8 bits); *Control* for the control word (42 bits). The output bitwidth is 24 bits: 12 bits for each disparity component. In this case, the core is implemented in a segmented pipeline with 3 stages:

- St_0: This stage computes the odd and even filtering quadrature components for the image pair.
- St_1: It computes the disparity for each orientation (we use 8 orientations).
- St_2: The provided component disparities need to be combined to compute the final full disparity. They conform an equation system solved in this stage applying least-squares (see Equation (9)).

For the computation of this core, we use 1,160 parallel processing units: stage St_0 has 32 paths (2 frames) for the computation of the odd and even filter components (with 8 orientations, we have 16 different filters), St_1 has 8 paths for the *atan2* operations and finally, St_2 has 2 paths, one for each disparity component. The number in brackets in the figure denotes the micropipelined stages for each of them. We have used two IP cores from Xilinx Core Generator platform to compute complex arithmetic operations such as arctangent (St_1) and a pipelined division (St_2).

Figure 5. Scheme of the pipelined stages for the phase-based vector disparity core. It describes the computation stages (from St_0 to St_2) indicating the pipelined stages (in brackets) and the number of parallel datapaths for each one of them.

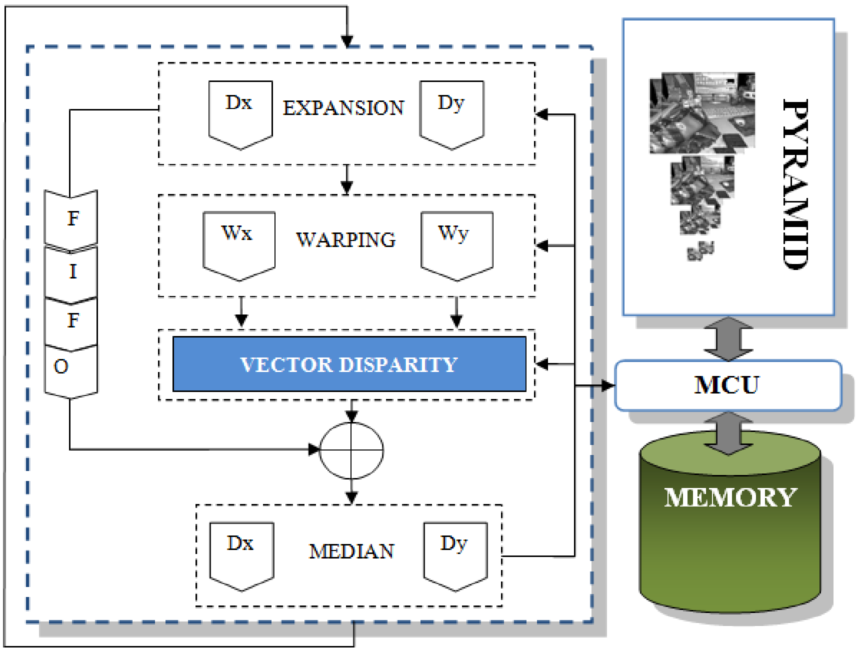


4.3. Multi-Scale Architecture

The implementation of the multi-scale architecture allows us the expansion of the working range of our disparity estimation more than 30× (using 5 scales) compared to the case of mono-scale versions. The design of this multi-scale architecture is the same as in previous works [30,45]. This approach is inspired by Bergen's work [42].

The multi-scalar version implementation is simple and it is displayed in Figure 6. Firstly, we compute the pyramid for the input images (left and right) and store them. In such a way, we have a bank of images sampled at the different spatial resolutions (depending on the number of scales). The second step consists of iterating in a loop as many times as scales: for each iteration, we upsample the previous estimation of the disparity; the next stage consists of the computation of the warping using the previous up-sampled results and the frames for the following finer scale; the warping operation compensates (removes) the apparent movement of scene elements due to the different points of view, allowing us the computation of disparity estimates in a valid range for that scale; the new disparity estimation is computed using the previous results as input for the core; the final stage collects both the new estimation and the partial previous stored results, combining them in a new partial estimation for the next iteration. The first iteration only consists of the initialization and computes the estimation using as input the images computed in the pyramid for the first spatial resolution scale and goes to the next iteration.

Figure 6. Hardware system architecture. Right side: the pyramid and its communication with memory. Left side: multi-scale computation (scaling, warping, merging, median filtering and vector disparity computation).



As a brief summary of this architecture, the design is divided into the following modules:

- Gaussian pyramid: This module implements the computation of the Gaussian pyramid of the left and right images for the disparity computation inspired in [43]. The number of levels of this pyramid depends on the number of scales with a downsampling factor of 2. It is built by a smoothing and a subsampling circuit. The main operations at this step are the 2D convolution with Gaussian filters of 5 taps (smoothing) and the sub-sampling. The kernel is a 5-by-5 matrix decomposed in two arrays $K = [\ 1\ \ 4\ \ 6\ \ 4\ \ 1]/16$. Thanks to the use of separable filters, the convolution is performed in a parallel way for the x and y operations. Five image lines are stored in an embedded multi-port BlockRAM used like a FIFO for the convolution. Then, we send to the output (the external SRAM) a pixel every two clock cycles: one pixel is discarded (sub-sampling). This sequential part of the processing is performed once and the results are stored/read to/from the RAM memory banks.
- Vector Disparity: This module has been explained in detail in previous subsections. Depending on the case, this module implements the vector disparity estimation based on the Lucas–Kanade or the phase-based algorithm.
- Expansion: This module up-samples the current scale results to the resolution of the following finer scale (the up-sampling factor is also 2). In contrast with the mono-dimensional disparity, this module is duplicated for the vector disparity. We use one module for each component of the vector disparity that needs to be up-sampled to the next scale resolution.
- Warping: In the literature, this computationally expensive stage is usually avoided because of its high computational cost. In our case, it consists of a bilinear interpolation between left and right image (keeping the left frame and warping the right one with the estimation of the disparity

computed for the previous scale). In Equation (10), the warping computation for the vector disparity computation is shown. Θ denotes the bidimensional scaling operator for $\mathbf{d}(x, y)$ (vector disparity) at the scale s with a factor of 2 and I^R stands for the right image.

$$I_R^s = Warp(I_R^{s-1}(x, y), \Theta(d^{s-1}(x, y))) \tag{10}$$

The warping operation for the vector disparity consists of a bilinear interpolation of the input images with the shift values which we have stored from the computed disparity in the previous scale. The computation of each warped pixel requires the reading of the pair $(\Delta x,\ \Delta y)$ and the correspondent pixel P. Each pair $(\Delta x,\ \Delta y)$ is used for retrieving from memory the four pixels of the original image for each P pixel. Then, the warped pixel is calculated performing a weighted bilinear interpolation with the obtained values. The warping process needs to perform four memory accesses per clock cycle to calculate one warped pixel to achieve the maximum throughput. This is one of the reasons for the choice of a specific MCU to manage data with different Abstract Access Ports (AAP) [46]. The warping architecture uses a reading AAP of the MCU for accessing the original image. The MCU provides a 36-bit bus allowing the access to four pixels per memory read. The X and Y matrices are provided from the expansion circuit through two blocking FIFOs. Warping requires a neighborhood of 4 pixels and the number of data available per memory access is limited to four in a same line. Thus, one access brings 2 pixels of the same line in the best case. Nevertheless, it is impossible to access the 4-pixel window in a single memory access. In the worst case, we access 4 different memory words (every 4 consecutive memory accesses). Therefore, the performance is constrained up to 4 pixels every 10 memory accesses.

- Merging: This module computes the addition of the previous feature estimation and the current one. The result is stored for the next iteration. The non-valid values are propagated from the coarsest scales to the finest ones. At the last scale, the finest one, we make the logical AND operation between its non-valid values and the propagated ones for the final estimation (the non-valid values obtained at the finest scale are the most reliable). The main problem of this module is the synchronization between the current and the stored results.
- Homogenization: This stage filters the results for each iteration with two 3 × 3 median filters in cascade. This filtering removes non-reliable values and homogenizes the results.

More details about the hardware implementation of this architecture can be found in [35,40,45].

5. Discussion of Results

As explained in the Introduction, with an active sensor that changes the vergence, the rectification process is required each time that the camera configuration is modified. This fact makes the static rectification approach unsuitable for this kind of systems. Moreover, an adaptive vergence system allows gazing on a target or fixation point and it may obtain the optimal disparity estimations around the image plane that is set using the depth of that fixation point. This section analyzes the performances of the vector disparity computation and the comparison in resource costs between our system and the same system with the static rectification unit.

We present the comparison of the developed systems in terms of resource utilization, accuracy and density of the results. Firstly, we list the performances of our work and some state-of-the-art publications with different implementations and technologies. Next, we benchmark our work (both approaches) with the set of images of Middlebury [49]. All this dataset is addressed to work with horizontal disparity algorithms; therefore, it is only shown to illustrate the comparison of our 2D disparity performances with the 1D ones. As mentioned before, our approach is suitable for working with active vision systems. We also test our implementations with our own benchmark images especially plotted to test the vector disparity accuracy and some images from an on-line set of benchmarks available at [50]. The benchmarking is performed for the hardware and software approaches to compare also the accuracy degradation due to the fixed-point arithmetic adopted in the hardware implementation. The last part of the section is dedicated to present a summary of the resource utilization.

In the case of the mono-dimensional benchmarks, we compute the MAE (Mean Absolute Error) and the Density. We also compute the PoBP (percentage of bad pixels, *i.e.*, the percentage of pixels whose MAE is greater than 1, see [40]). For the vector disparity estimation, we compute the AAE (Average Angular Error), and the Density. Furthermore, we also compute the PoGP (percentage of good pixels, or the percentage of pixels whose AAE is less than 5, see again [40]).

5.1. State-of-the-Art Comparison

We mentioned in the Introduction that our aim is the development of a high-performance system that works in real time (in our case, it means a frame rate of at least 25 fps with VGA resolution). In Table 1, we find that our implementation reaches up to 32 fps with a resolution of 640 × 480. This result fulfills our requirements.

Table 1. Disparity performance comparison (works sorted by date of publication). For vector disparity implementations, two PDS values are given: the first considers only 1-D displacement performance and the second takes into account that 2-D matching of the vector methods have a search region that is the squared of the 1-D ones.

	Resolution	**Frame rate (fps)**	**PDS ($\times10^6$)**	**Architecture**	**Algorithm**
Our phase-based mono-scale core	640 × 480	118	218/1,304	Xilinx V4 (36 MHz)	2D Phase-based
Our Lucas–Kanade mono-scale core	640 × 480	267	492/1,968	Xilinx V4 (82 MHz)	2D Lucas–Kanade
Our phase-based multi-scale system	640 × 480	32	1,887/7,122	Xilinx V4 (42 MHz)	2D Phase-based
Our Lucas–Kanade multi-scale system	640 × 480	32	1,132/4,528	Xilinx V4 (41 MHz)	2D Lucas–Kanade
Tomasi (2010) [30]	512 × 512	28	939	Xilinx V4 (42 MHz)	1D Phase-based
Chang (2010) [51]	352 × 288	42	273	UMC 90nm Cell	1D Semi-Census
Hadjitheofanous (2010) [31]	320 × 240	75	184	Xilinx V2 Pro	1D SAD

Table 1. *Cont.*

	Resolution	**Frame rate (fps)**	**PDS ($\times 10^6$)**	**Architecture**	**Algorithm**
Jin (2010) [32]	640 × 480	230	4,522	Xilinx V5 (93.1 MHz)	1D Census Transform
Calderon (2010) [33]	288 × 352	142	2,534	Xilinx V2 Pro (174.2 MHz)	1D BSAD
Chessa (2009) [17]	256 × 256	7	59/236	QuadCore Processor	2D Energy-based Pop. coding
Georgoulas (2009) [28]	800 × 600	550	21,120	Stratix IV (511 MHz)	1D SAD
Ernst (2009) [52]	640 × 480	4.2	165	GeForce 8800	1D SGM
Han (2009) [53]	320 × 240	144	707	ASIC (150MHz)	1D SAD
Gibson (2008) [54]	450 × 375	6	65	G80 NVIDIA	1D SGM
Diaz (2006) [34]	1,280 × 960	52	1,885	Xilinx V2 (65 MHz)	1D Phase-based
Gong (2005) [55]	384 × 288	16	30–60	ATI Radeon x800	1D GORDP

Table 1 shows a performance comparison between our four developments (mono- and multi-scale approaches, phase-based, and Lucas–Kanade) with the last works in the literature. Due to the lack of works for computing vector disparity, we summarize in the table the most important ones that compute horizontal disparity in the last years. Besides, the mono-scalar version achieves a frame rate calculated using the maximum working frequency. Finally, in the multi-scalar versions, the frame rate is empirically measured using the proposed architecture and taking into account the PCIe bandwidth restrictions for the communication with a PC.

Most of the listed works in Table 1 are mono-scalar developments except [30] and [17]. The performances depend on the algorithm, the architecture, and the optimization level. Algorithms with a lower computational cost as SAD-based implementations may obtain better speed performances but the accuracy results are also rather low. Furthermore, there are substantial differences between mono- and multi-scalar versions. On the other hand, we merge horizontal and vector disparity algorithms in the same table to easily compare the main performances, but this comparison is not completely fair, e.g., in the case of the PDS. The PDS (Points × Disparity measures per Second) is a metric that measures the number of operations performed by our algorithm per time taking into account the disparity range. With respect to this last measure, Georgoulas *et al.* [28] achieves impressive results with a PDS of 21120×10^6. Our PDS for the best multi-scalar version is 7122×10^6. The computation of the PDS depends on the disparity range and, for vector disparity algorithms, this range is squared compared with the horizontal disparity range. In the column, we show firstly the PDS using the simple range to check the magnitude differences with the 1D algorithms and the second values are the PDS for the 2D implementation.

To estimate the disparity at different spatial resolution levels, the multi-scalar architecture is revealed as essential. The Laplacian pyramidal approach for the multi-scale architecture [43] is the way to implement this multi-resolution analysis through a fine-to-coarse strategy. The objective is to estimate disparities larger than the filter support. The factor depends on the number of scales (determined by the target application). In the case of the mono-scalar versions, the main advantage is obviously a computation with a high frame rate. Moreover, multi-scalar versions obtain significant precision improvements due to its wide-range estimations.

5.2. Benchmarking for Horizontal Disparity

In this subsection, we briefly analyze the performances of our vector disparity algorithms with the well-known benchmark images of Middelbury [49], whose vertical disparity components are always zero. As vector disparity implementations are rare in the literature, we include this section for future comparisons with horizontal disparity algorithms. Actually, we do not expect better results than the proper 1D algorithms, any variation in the vertical disparity (that should be zero) adds error to the resolution of the 2D equation system. Moreover, the search space for the solution is developed from 1D to 2D (see Equations (3) and (9) respectively) which makes the solutions more unreliable.

Figure 7 and Table 2 show the performance analysis for the horizontal disparity computation. In the literature, besides the example which is being used in this paper, there are not many image benchmarks for vector disparity. Most of the works which we find in the literature test their algorithms and implementations with the classical horizontal disparity benchmark provided by Middlebury [49]. Firstly, we also show the efficiency and density of our approaches with some images of this benchmark, in particular: "*Tsukuba*", "*Sawtooth*", "*Venus*", "*Teddy*", and "*Cones*" (respectively called $bm1$ to $bm5$). We distinguish between the Lukas–Kanade and the phase-based version and between the hardware and the software implementations. In the figure, we notice that the error is similar for the different versions except in the case of $bm4$ and $bm5$ ("*Teddy*" and "*Cones*" cases). The increment in these cases of the errors for the phase-based hardware version is substantial and also entails an important loss of density. The Lucas–Kanade hardware version achieves even better results than the software version due to the use of homogenization stages. The table shows the best results for the Lucas–Kanade hardware version but, except for the last two cases, all the results are very similar and the degradation of the precision is not quite significant.

The generalized loss of density is very significant and may be attributed to the loss of precision of both designs. But it is even worst in the case of the phase-based implementation constrained for the fixed-point arithmetic, especially in the complex operations such as divisions and arctangent modules and in computations for the Gabor filtering stage. Moreover, the use of the fixed-point arithmetic also affects the computation of the confidence measure that performs the thresholding operation. Finally, it is also worth regarding that hardware implementations in Figure 7 are generally more accurate than their correspondent software implementations (more accurate and reliable estimations entail discarding more estimations).

Figure 7. Horizontal disparity comparison: Lucas–Kanade *vs.* Phase-based and Hardware *vs.* Software versions (MAE and density).

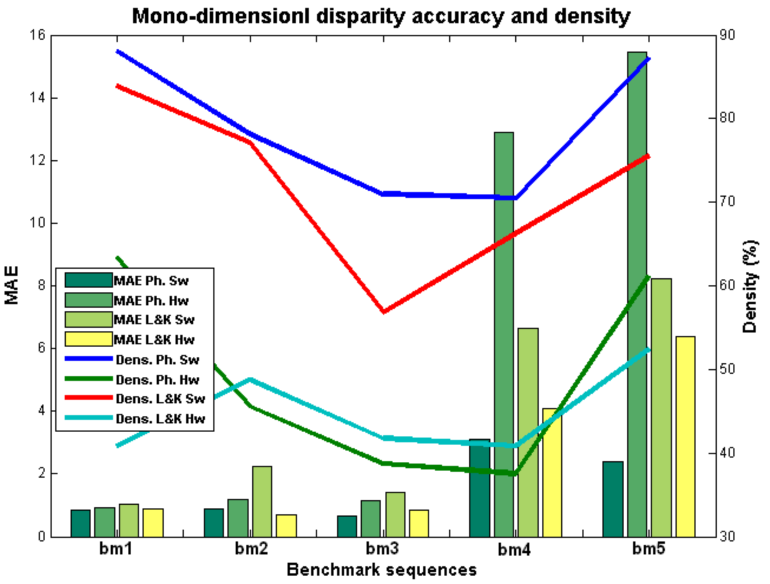


Table 2. Horizontal disparity performances: PoBP(%) percentage of pixels where MAE >1 and the density between parentheses.

	Phase-based		Lucas-Kanade	
	SW	**HW**	**SW**	**HW**
Tsukuba	16.65 (88.05%)	13.76 (63.42%)	21.77 (83.89%)	9.00 (40.89%)
Sawtooth	10.82 (78.11%)	10.58 (45.56%)	27.66 (77.10%)	5.90 (48.79%)
Venus	8.37 (70.99%)	7.84 (38.69%)	18.07 (56.83%)	7.57 (41.78%)
Teddy	25.73 (70.46%)	27.06 (37.55%)	40.91 (66.20%)	25.25 (40.85%)
Cones	27.18 (87.20%)	48.32 (61.10%)	58.06 (75.52%)	40.65 (52.45%)

5.3. Benchmarking for Vector Disparity

In this last subsection, we perform the benchmarking using the convenient set of images for the analysis for the performances of the vector disparity implementations. Figure 8 and Table 3 show the vector disparity results for the "*plane 0H 0V*", "*plane 15H 15V*", "*desktop00*", and "*desktop09*" sequences (marked as $bv1$ to $bv4$). The first two sequences (*bv1* and *bv2*) were generated using an initial ground-truth, whereas $bv3$ and $bv4$ are active views of 3D acquired natural scenes generated by a virtual reality system [50] and available at [56]. In this case, differences between the software and hardware versions are more important. On the other hand, the differences between the phase-based and Lucas–Kanade have been shortened, although the Lucas–Kanade hardware implementation seems slightly better. This fact can be attributed, as in the previous section, to the loss of accuracy that affects the phase-based model due to the bit-widths and fixed-point arithmetic limitations. Table 3 also supports it, showing a PoGP that model is about 25% higher in the Lucas–Kanade than in the phase-based model.

Figure 8. Vector disparity comparison: Lucas–Kanade *vs.* Phase-based and Hardware *vs.* Software versions (AAE and density).

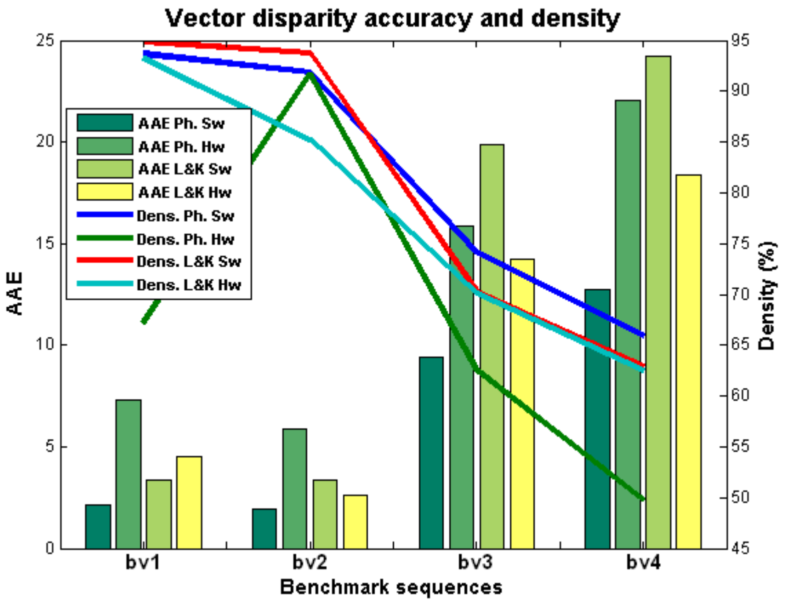


Table 3. Vector disparity performances: PoGP(%), defined as the percentage of pixels where AAE < 5 deg. and the density between parentheses.

	Phase-based		Lucas–Kanade	
	SW	**HW**	**SW**	**HW**
plane 0H 0V	70.04 (93.66%)	93.77 (91.84%)	67.57 (74.17%)	68.67 (65.88%)
plane 15H 15V	69.99 (67.15%)	73.27 (91.73%)	67.82 (62.55%)	68.82 (49.76%)
desktop00	82.30 (94.84%)	80.48 (93.76%)	79.61 (70.29%)	80.68 (62.97%)
desktop09	84.24 (93.19%)	86.74 (85.26%)	82.92 (70.16%)	82.99 (62.57%)

In general, in the case of this benchmark, the density is quite similar in contrast with the previous subsection. The computation is not affected as dramatically as in the previous case because now we are using appropriately the confidence measure (implemented for the 2D model not for the 1D). For instance (except for the first sequence $bv1$), the maximum difference between hardware and software results in density is about 15%, with similar tendencies for all the sequences.

Finally, Figures 9 and 10 show the disparity results for the software and hardware versions for the sequences available at [56] and the ones that we generated. In the figures, we display the original images, the ground-truth, and the results for the software versions in the case of the first figure and the hardware results for the second one. All the estimations have a color-coding frame describing the direction of the estimated vector disparity. The values in black in the hardware column are the unreliable values (NaN). As it can be seen, the phase-based results present better results in the case of the software but, for the hardware implementation, the bit-width constraining degrades the final estimation in an appreciable way. On the other hand, the phase-based implementation presents some advantages for its use in real-world applications such as a better behavior against variations in the illumination [57,58]. This last advantage is illustrated in the figures, especially for "*desktop00*" and "*desktop09*". For a more detailed study of the robustness of phase-based implementations against illumination changes and affine transformations between the stereo pair images, cf. [57,59,60].

Figure 9. Software benchmark results for vector disparity. From left to right: original image, ground-truth, software results for phase-based, and Lucas–Kanade algorithms. The frame codes the vector disparity with a color.

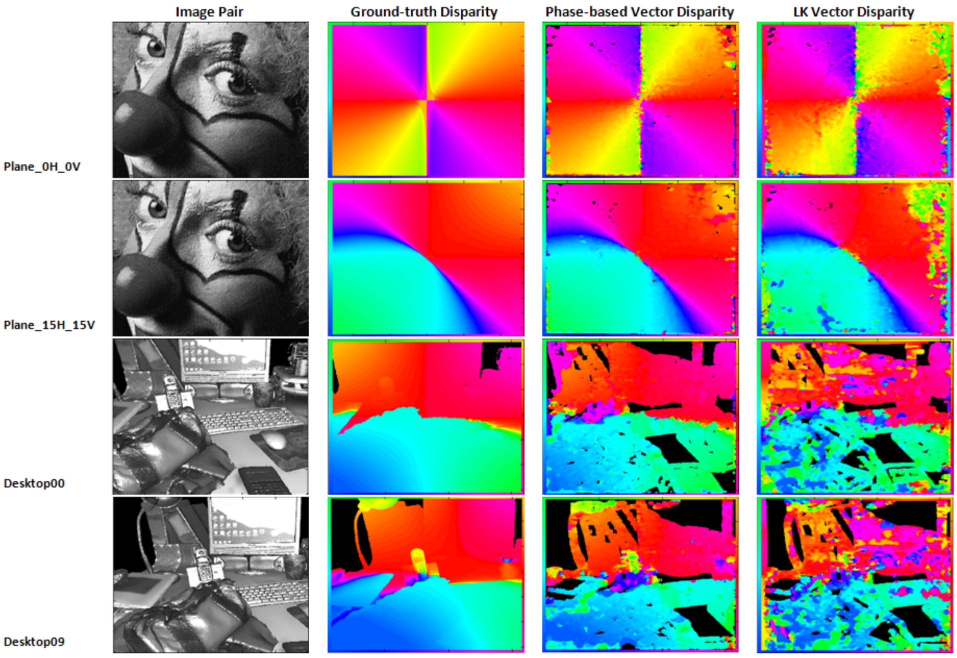
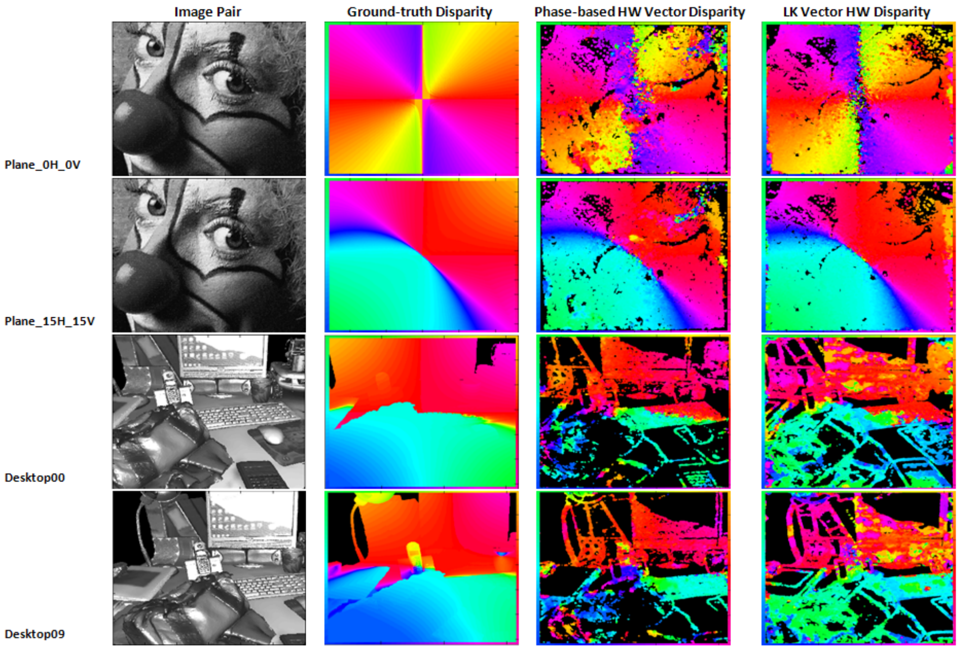


Figure 10. Hardware benchmark results for vector disparity. From left to right: original image, ground-truth, hardware results for phase-based, and Lucas–Kanade algorithms. The frame codes the vector disparity with a color.



5.4. Hardware Resource Utilization

In computer vision, the selection of the best alternative is performed depending on the target application, searching for a good trade-off between the required accuracy (see previous subsections) and constraints about the maximum frequency and the resource utilization. Table 4 shows the resource utilization and the maximum working frequency of the implemented system. This table presents the information about the board resource utilization: total number of 4 input LUTs, Slice Flip Flops, Slices, DSPs, Block RAMs used, and finally, the maximum frequency (MHz).

Table 4. Hardware resource utilization for the presented complete architecture using a Virtex-4 FX100 FPGA (XC4vfx100).

	4 input LUTs (out of 84,352)	**Slice Flip-Flops (out of 84,352)**	**Slices (out of 42,716)**	**DSP (160)**	**Block RAM (378)**	**Freq (MHz)**
1D Phase-based system + Rectification	47,109 (56%)	27,221 (32%)	32,678 (76%)	159 (99%)	88 (23%)	44
1D Lucas–Kanade system + Rectification	55,152 (65%)	35,360 (41%)	38,560 (90%)	154 (96%)	100 (26%)	42
2D Phase-based Disp. system	55,445 (65%)	40,597 (48%)	37,383 (88%)	107 (66%)	126 (33%)	42
2D Lucas–Kanade Disp. system	33,039 (39%)	28,123 (33%)	27,749 (65%)	50 (31%)	148 (39%)	41

The listed options include: the multi-scale vector disparity systems and the horizontal disparity systems with the rectification stage, for both approaches. More details about the resource cost of the different unit components of the multi-scale generalization can be found at [30,35,45].

For the horizontal disparity systems, the Lucas–Kanade version uses about 9% more 4-Input LUTs, this is due to the difference in Block RAMs and DSPs and especially, to the optimization level. Tools for hardware implementations apply optimization techniques based on heuristics that cannot be completely controlled using the software parameters. A totally fair comparison can also be done with a more intensive use of resources.

On the other hand, focusing on the vector disparity computations, while the phase-based approach increases the resource utilization about 10%, the Lucas–Kanade approach achieves a remarkable reduction of about 25%. As the multi-scale architecture is the same for both approaches, the increment in resource costs for the phase-based approach is due to the 2D core. While the rectification unit is the bottle-neck for the Lucas–Kanade architecture, it is the proper 2D core for the phase-based scheme.

All these results support the idea of implementing the final system for motion control of our binocular system with the Lucas–Kanade approach taking into account also the low degradation of accuracy obtained in the previous subsection and the resource saving of more than 25% with respect to the phase-based approach.

6. Control of Vergence for a Binocular System

As described in the Introduction section, our final objective is the implementation of a system that, using the vector disparity (implemented according to the previous sections), manages the vergence, version, and tilt control.

Motor control is distributed using Digital Signal Processing (DSPs) boards, in this case, Freescale DSP-56F807, 80 MHz, fixed point 16 bits which perform a fast low-level control loop in real time. A CAN-bus line allows the communication between the boards and a remote PC. Motors are directly controlled by standard PID controllers. This DSP, due to its memory and computation limitations, implements simple operations such as pre-filtering, signal acquisition, and PID position control (using absolute position encoders). More details about the mechanics and the architecture of the ICub and RobotCub can be found in [26,27].

The algorithm that we use for the fixation consists of selecting a point of our object of interest and computing a centroid based on the color of this point, using a threshold for the RGB components (a range of [−5,5] for each component). Once we have the centroid computed, we firstly compute the distance between this point and the current position of the eye centers and send it via the CAN bus to the DSP-based control cards. The communication protocol allows four working modes: sending relative angles, absolute angles, relative pixel positions, or normalized relative position (see the documentation at [25]); as explained, we use the third alternative. The PID low-level control loop acts on the motors to modify version and tilt according to these coordinates of the color-based centroid and in this way, we set our eye centers in the object of interest. The last step is acting on the vergence motors and computing the vector disparity until it reaches a value near zero (we use a threshold of 5). Changing the vergence of the cameras means translating the image plane to the depth of the target. Once the object is fixated, we will obtain the optimal performances around this object (zero disparity plane).

Tracking adds the possibility of performing a smooth pursuit of the object gazing on several fixation points. It is worth noticing that, due to the huge backlash on the eyes pan movement [25] and the limits in the physical properties of the motors, the pursuit is not as smooth as it may be expected and there are some fluctuations shown in the second experiment.

One of the problems that might appear in this scenario is the motion blur between the left and right camera images. In our algorithm, we firstly have a master camera that moves to the color centroid (where the object of interest is) and fixates to this position, followed by the second camera. As explained, the tilt is common to the cameras, while the version is achieved to move the cameras towards the same orientation. Then, the second camera (the slave) achieves the vergence movement to fixate the object, according to the position of the first eye.

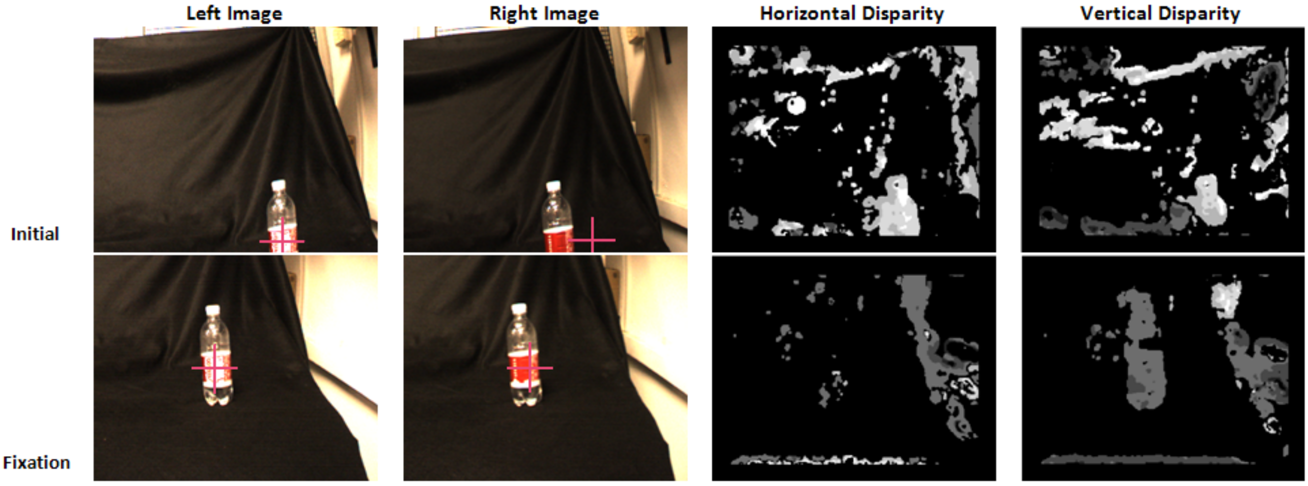
The blur may happen during this second step, since the disparity is now being computed. However, we partially avoid this situation by using the disparity after each movement when the rig setup of the cameras is steady and not constant. In addition, the multiscale process also provides robustness to this issue due to the coarse-to-fine process and the limited disparity range possible at each scale.

In this final section, we present the results for two real experiments: the fixation in a target object, in our case it is a red plastic bottle, and the fixation and tracking of a blue LED light.

6.1. Experiment 1: Red Plastic Bottle Fixation

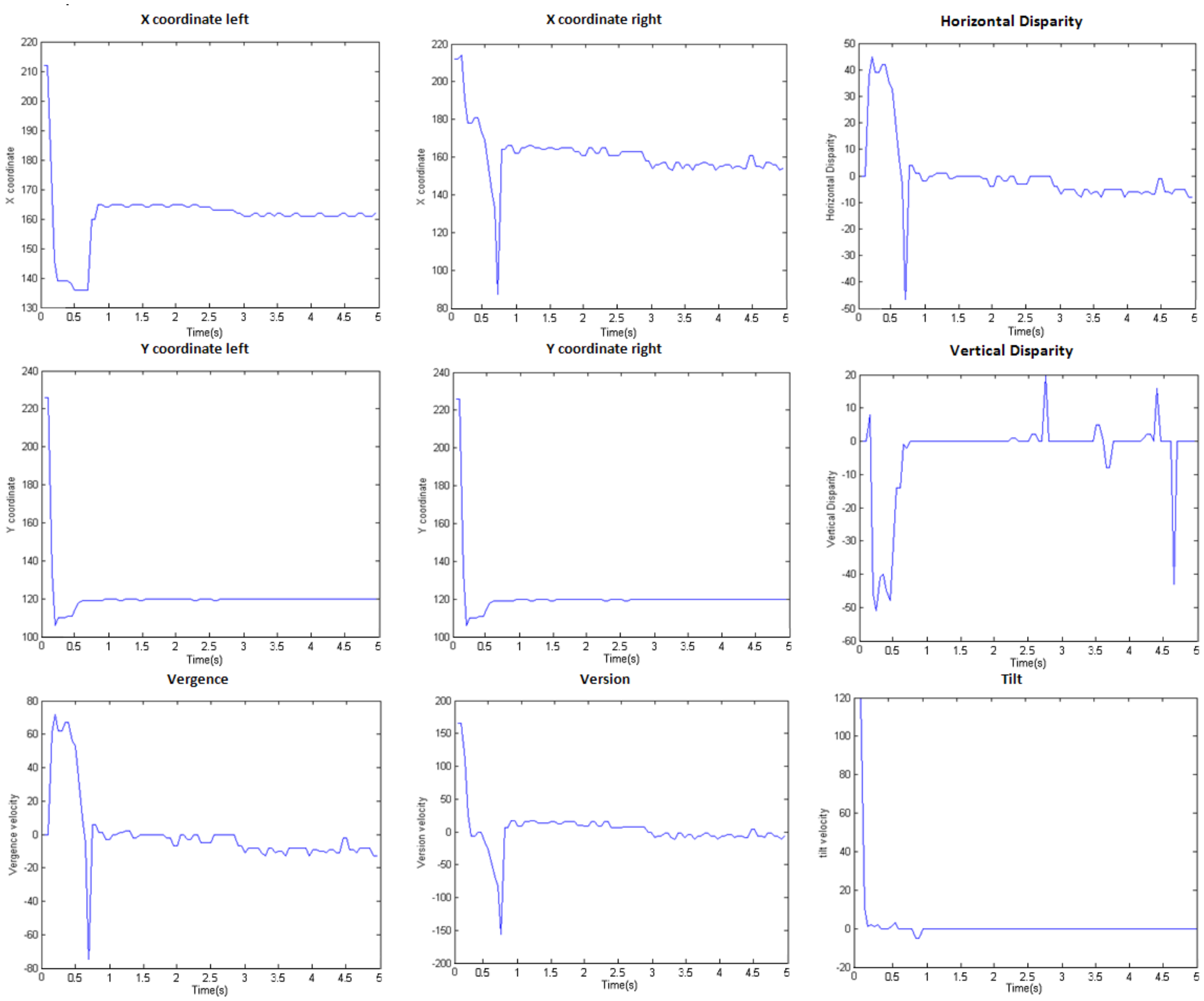
The first experiment consists of making the system fixate at a red plastic bottle. Figure 11 displays the initial position of the system (left and right images of the cameras). The red cross shows the fixation point in each case. The processing begins with the selection of the fixation point and then, the system performs the fixation task with the static object. In the figure, we observe the initial horizontal disparity at the fixation point; after fixating the point, the horizontal disparity tends to zero (we use a threshold to avoid variations that might affect the stability of the system). The vertical disparity is also almost zero, and in our case, with a common tilt for both cameras, it is not possible to reduce it.

Figure 11. Initial position and fixation for the red plastic bottle example. Disparity components are displayed.



The evolution of the X and Y coordinates of the fixation point for both cameras and the component disparities are plotted in Figure 12. As we see, with an image resolution of 320 × 240, the fixation is finally performed in a very accurate and fast way. After 0.8 s, all the evolutions of the different parameters are stabilized (X and Y coordinates at 160 and 120 values respectively; component disparities around zero). The third row shows the evolutions for the vergence, version, and tilt. We plot the velocities for the different parameters that are the inputs of the DSP that controls the different motors of our system. After 0.8 s, all of them are also stable. Similar results are obtained for fixation in several works using other architectures: in [61], the fixation time is about 0.5–0.6 s and in [62], about 1.5 s. Our approach is shown as a very competitive alternative according to the fixation time and it has been also presented as one the most accurate ones.

Figure 12. Red plastic bottle fixation example: X and Y coordinates for left and right images, horizontal and vertical component disparities and the evolution of version, vergence, and tilt velocities along the time (image resolution is 320 × 240).

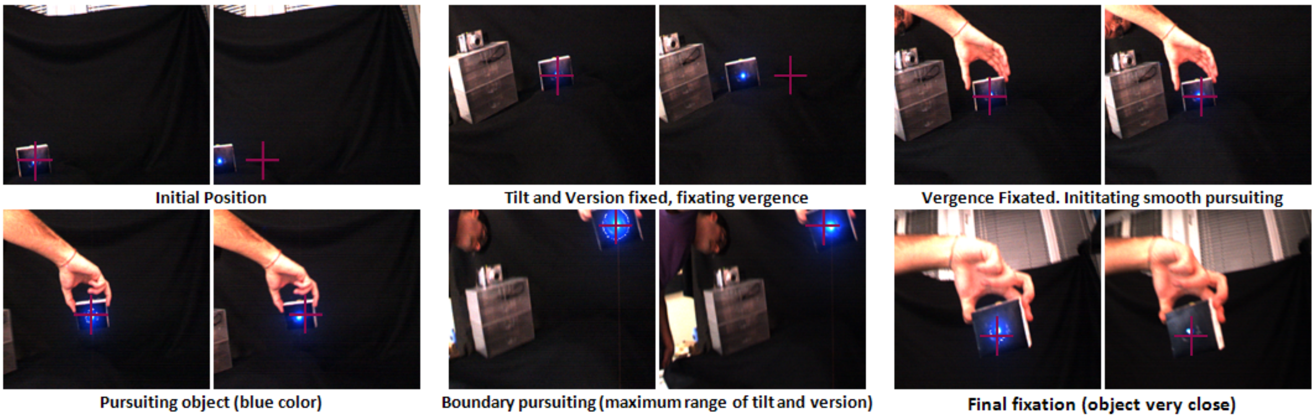


6.2. Experiment 2: Smooth Pursuit

In this experiment, we show the smooth pursuit example with a blue object. Figure 13 shows the experiment performed for the pursuit of a blue object. The sequence shows the initial fixation and some subsequent captures of the cameras of our system tracking the object even at the limits of its physical properties. Once the object is detected, the system always performs the fixation again.

It is worth noticing that the fixation point is not exactly the same point of the object in both cases, on the left and right camera images, because, as we mentioned, we use a threshold for the zero disparity region. In this way, our system is more stable and smoother trajectories are ensured.

Figure 13. Blue LED smooth pursuit. At each different phase, we show the left and right images of our system, beginning with the fixation and then, with the pursuit of the target object.



Initial Position

Tilt and Version fixed, fixating vergence

Vergence Fixated. Inititating smooth pursuiting

Pursuiting object (blue color)

Boundary pursuiting (maximum range of tilt and version)

Final fixation (object very close)

7. Conclusions

The main contribution of this work is the implementation of a sensor for an active vision system with dynamic vergence control to explore dynamic environments using a scheme different to the common static camera configurations. The main problem of an active system is that the rectification process is required each time that the camera configuration is modified. Therefore, the use of a static rectification preprocessing stage technique is unsuitable for any active system that actively modifies the vergence angle of its camera setups. Our solution consists of using vector disparity estimations to control vergence. We have developed a SoC (system-on-a-chip) that integrates vector disparity estimation with vergence control on the same chip. This allows an embedded implementation with a low consumption at a single device that can be integrated on the i-Cub head (or any vision system including low level processing stages such as active stereo vision).

The first part of this work is dedicated to the design and implementation of an embedded system for the estimation of vector disparity. We have developed two different alternatives: a gradient-based one and a phase-based one. As mentioned, there are multiple examples for the horizontal disparity, while in the case of the vector disparity, as far as we know, the developments of this phase-based estimation in the literature are very seldom (see [13,17]), and their hardware implementation is even rarer. We have also designed a multi-scale generalization to increase the number of disparity levels 30× (using 5 spatial scales).

The requirements of a real-time system have been successfully fulfilled, since we are able to reach a working frequency of 32 fps with a VGA resolution (640 × 480) and, resorting on a multi-scale architecture, we are able to cope with large disparities. In the best case, the mono-scalar version of the system may achieve up to 267 fps for the same resolution, which shows the maximum level of parallelism provided by an FPGA (in our case, a Xilinx Virtex4 XC4vfx100 device). We have also analyzed the accuracy and density of our designs showing competitive results. By comparing different techniques with proper benchmark sequences, the Lucas-Kanade algorithm (including the homogenization stage) is revealed as the best choice, showing optimal efficacy *vs.* efficiency trade-off for vector disparity computation.

Hence, we have also compared the hardware and software versions of the algorithms; this comparison shows a low degradation for our hardware implementation which is affordable taking into account the use of fixed-point arithmetic instead of floating-point one. We have also compared the phase-based and the gradient-based algorithms and summarize the resource utilization. With the gradient-based algorithm, the resource cost is about 23% less than in the case of the phase-based one comparing the multi-scalar versions and 37% for the mono-scalar version. In terms of maximum working frequency, the gradient-based system is 2.3 times faster than the phase-based one.

The last section in this work deals with the implementation of the control model for a binocular camera system inspired by the Hering's Law. The control manages the vergence, version, and tilt angles of the system to modify the fixation point in order to focus on a target object using the real-time vector disparity computation. The easy integration with such a system and the low power consumption of the system [36] support the employment of the FPGA as well.

In the paper, we have also presented a fixation application and a tracking trajectory example for a real-world scenario. The implementation is tackled using the real-time vector disparity estimation, computed by the gradient-based algorithm. It consists of moving the cameras towards the position of an object of interest and afterwards, in moving the fixation point using the computed vector disparity to take advantage of the optimal computation that can be performed at the zero disparity plane. Once the proper fixation is ensured, we achieved a second experiment for a smooth pursuit movement towards an object of interest. As shown, we achieve fixating at the point of interest in approximately 0.8 s (it involves the gazing towards the object and the vergence control).

Finally, as future works, we will address the integration of the system on a robotic mobile platform for the implementation of real algorithms for autonomous navigation and scene mapping.

Acknowledgment

This work has been supported by the National Grant (AP2007-00275), the projects ARC-VISION (TEC2010-15396), ITREBA (TIC-5060), MULTIVISION (TIC-3873), and the EU project TOMSY (FP7-270436).

References

1. Bertozzi, M.; Broggi, A. GOLD: A parallel real-time stereo vision system for generic obstacle and lane detection. *IEEE Trans. Image Process.* **1998**, *7*, 62–81.
2. El Ansari, M.; Mousset, S.; Bensrhair, A. A new stereo matching approach for real-time road obstacle detection for situations with deteriorated visibility. In *Proceedings of the 2008 IEEE Intelligent Vehicles Symposium*, Eindhoven, The Netherlands, 4–6 June 2008; pp. 355–360.
3. Oisel, L.; Memin, E.; Morin, L.; Galpin, F. One-dimensional dense disparity estimation for three-dimensional reconstruction. *IEEE Trans. Image Process.* **2003**, *12*, 1107–1119.
4. Lu, Z.; Shi, B. Subpixel resolution binocular visual tracking using analog VLSI vision sensors. *IEEE Trans. Circuits Syst. II Analog Digit. Signal Process.* **2000**, *47*, 1468–1475.
5. Musleh, B.; Garca, F.; Otamendi, J.; Armingol, J.M.; de la Escalera, A. Identifying and tracking pedestrians based on sensor fusion and motion stability predictions. *Sensors* **2010**, *10*, 8028–8053.

6. Coelho, J.; Piater, J.; Grupen, R. Developing haptic and visual perceptual categories for reaching and grasping with a humanoid robot. *Robot. Auton. Syst.* **2001**, *37*, 195–218.
7. Song, W.; Minami, M.; Yu, F.; Zhang, Y.; Yanou, A. 3-D hand amp; Eye-vergence approaching visual servoing with lyapunouv-stable pose tracking. In *Proceedings of the 2011 IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, 9–13 May 2011; pp. 5210–5217.
8. Diaz, J.; Ros, E.; Carrillo, R.; Prieto, A. Real-time system for high-image resolution disparity estimation. *IEEE Trans. Image Process.* **2007**, *16*, 280–285.
9. Murphy, C.; Lindquist, D.; Rynning, A.; Cecil, T.; Leavitt, S.; Chang, M. Low-cost stereo vision on an FPGA. In *Proceedings of the 15th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM '07)*, Napa, CA, USA, 23–25 April 2007; pp. 333–334.
10. Trucco, E.; Verri, A. *Introductory Techniques for 3-D Computer Vision*; Prentice Hall PTR: Upper Saddle River, NJ, USA, 1998.
11. Faugeras, O.; Luong, Q.T.; Papadopoulou, T. *The Geometry of Multiple Images: The Laws that Govern the Formation of Images of A Scene and Some of Their Applications*; MIT Press: Cambridge, MA, USA, 2001.
12. Samarawickrama, J.G.; Sabatini, S.P. Version and vergence control of a stereo camera head by fitting the movement into the hering's law. In *Proceedings of the 4th Canadian Conference on Computer and Robot Vision*, Montreal, QC, Canada, 28–30 May 2007; pp. 363–370.
13. Theimer, W.M.; Mallot, H.A. *Vergence Guided Depth Reconstruction Using a Phase Method*; *Neuro-Nimes '93*; EC2 Publishing: Nanterre, France, 1993; pp. 299–308.
14. Gibaldi, A.; Canessa, A.; Chessa, A.; Sabatini, S.P.; Solari, F. A neuromorphic control module for real-time vergence eye movements on the iCub robot head. In *Proceedings of the 10th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, Nashville, TN, USA, 6–8 December 2011; pp. 543–550.
15. Semmlow, J.L.; Yuan, W.; Alvarez, T.L. Evidence for separate control of slow version and vergence eye movements: Support for Hering's law. *Vis. Res.* **1998**, *38*, 1145–1152.
16. Enrights, J. Changes in vergence mediated by saccades. *J. Physiol.* **1983**, *350*, 9–31.
17. Chessa, M.; Sabatini, S.P.; Solari, F. A fast joint bioinspired algorithm for optic flow and two-dimensional disparity estimation. In *Proceedings of the 7th International Conference on Computer Vision Systems: Computer Vision Systems (ICVS '09)*, Liege, Belgium, October 2009; pp. 184–193.
18. Xu, Y.; Zhou, J.; Zhai, G. 2D phase-based matching in uncalibrated images. In *Proceedings of the IEEE Workshop on Signal Processing Systems Design and Implementation*, Athens, Greece, 2–4 November 2005; pp. 325–330.
19. Nalpantidis, L.; Amanatiadis, A.; Sirakoulis, G.; Gasteratos, A. Efficient hierarchical matching algorithm for processing uncalibrated stereo vision images and its hardware architecture. *IET Image Process.* **2011**, *5*, 481–492.
20. Beauchemin, S.S.; Barron, J.L. The computation of optical flow. *ACM Comput. Surv.* **1995**, *27*, 433–466.

21. Lucas, B.D.; Kanade, T. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI '81)*, Vancouver, BC, Canada, August 1981; pp. 674–679.
22. Fleet, D.J.; Jepson, A.D.; Jenkin, M.R. Phase-based disparity measurement. *CVGIP Image Underst.* **1991**, *53*, 198–210.
23. Sabatini, S.P.; Gastaldi, G.; Solari, F.; Pauwels, K.; Hulle, M.M.V.; Diaz, J.; Ros, E.; Pugeault, N.; Krger, N. A compact harmonic code for early vision based on anisotropic frequency channels. *Comput. Vis. Image Underst.* **2010**, *114*, 681–699.
24. Solari, F.; Sabatini, S.P.; Bisio, G.M. Fast technique for phase-based disparity estimation with no explicit calculation of phase. *Electron. Lett.* **2001**, *37*, 1382–1383.
25. iCub Project. The EU iCub Project: An Open Source Cognitive Humanoid Robotic Platform. Available online: http://www.icub.org/ (accessed on 28 December 2011).
26. Beira, R.; Lopes, M.; Praga, M.; Santos-Victor, J.; Bernardino, A.; Metta, G.; Becchi, F.; Saltaren, R. Design of the robot-cub (iCub) head. In *Proceedings of the 2006 IEEE International Conference on Robotics and Automation (ICRA '06)*, Orlando, CA, USA, May 2006; pp. 94–100.
27. Ivaldi, S. From Humans to Humanoids: A Study on Optimal Motor Control for the iCub. Ph.D. Thesis, University of Genoa, Genoa, Italy, 2011.
28. Georgoulas, C.; Andreadis, I. A real-time occlusion aware hardware structure for disparity map computation. In *Proceedings of the Image Analysis and Processing (ICIAP '09)*, Vietri sul mare, Italy, September 2009; Volume 5716, pp. 721–730.
29. Hansard, M.; Horaud, R. Cyclopean geometry of binocular vision. *J. Opt. Soc. Am.* **2008**, *25*, 2357–2369.
30. Tomasi, M.; Vanegas, M.; Barranco, F.; Diaz, J.; Ros, E. A novel architecture for a massively parallel low level vision processing engine on chip. In *Proceedings of the 2010 IEEE International Symposium on Industrial Electronics (ISIE '10)*, Bari, Italy, 5–7 July 2010; pp. 3033–3039.
31. Hadjitheophanous, S.; Ttofis, C.; Georghiades, A.; Theocharides, T. Towards hardware stereoscopic 3D reconstruction a real-time FPGA computation of the disparity map. In *Proceedings of the 2010 IEEE International Symposium on Design, Automation & Test in Europe Conference & Exhibition (DATE '10)*, Dresden, Germany, 8–12 March 2010; pp. 1743–1748.
32. Jin, S.; Cho, J.; Pham, X.D.; Lee, K.M.; Park, S.K.; Kim, M.; Jeon, J.W. FPGA design and implementation of a real-time stereo vision system. *IEEE Trans. Circuits Syst. Video Technol.* **2010**, *20*, 15–26.
33. Calderon, H.; Ortiz, J.; Fontaine, J. High parallel disparity map computing on FPGA. In *Proceedings of the 2010 IEEE/ASME International Conference on Mechatronics and Embedded Systems and Applications (MESA '10)*, Suzhou, China, 8–10 July 2010; pp. 307–312.
34. Diaz, J.; Ros, E.; Pelayo, F.; Ortigosa, E.M.; Mota, S. FPGA-based real-time optical-flow system. *IEEE Trans. Circuits Syst. Video Technol.* **2006**, *16*, 274–279.
35. Barranco, F.; Tomasi, M.; Diaz, J.; Vanegas, M.; Ros, E. Parallel architecture for hierarchical optical flow estimation based on FPGA. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2012**, doi: 10.1109/TVLSI.2011.2145423.

36. Diaz, J.; Ros, E.; Agis, R.; Bernier, J. Superpipelined high-performance optical-flow computation architecture. *Comput. Vis. Image Underst.* **2008**, *112*, 262–273.
37. Brandt, J.W. Improved accuracy in Gradient-based optical flow estimation. *Int. J. Comput. Vis.* **1997**, *25*, 5–22.
38. Barron, J.L.; Fleet, D.J.; Beauchemin, S.S. Performance of optical flow techniques. *Int. J. Comput. Vis.* **1994**, *12*, 43–77.
39. Liu, H.; Hong, T.H.; Herman, M.; Chellappa, R. Accuracy *vs.* efficiency trade-offs in optical flow algorithms. In *Computer Vision ECCV 96*; Springer: Berlin, Germany, 1996; Volume 1065, pp. 174–183.
40. Pauwels, K.; Tomasi, M.; Diaz, J.; Ros, E.; Hulle, M.M.V. A comparison of FPGA and GPU for real-time phase-based optical flow, stereo, and local image features. *IEEE Trans. Comput.* **2011**, doi: 10.1109/TC.2011.120.
41. Gautama, T.; van Hulle, M. A phase-based approach to the estimation of the optical flow field using spatial filtering. *IEEE Trans. Neural Netw.* **2002**, *13*, 1127–1136.
42. Bergen, J.; Anandan, P.; Hanna, K.; Hingorani, R. Hierarchical model-based motion estimation. In *Computer Vision ECCV'92*; Sandini, G., Ed.; Springer: Berlin, Germany, 1992; Volume 588, pp. 237–252.
43. Burt, P.J.; Adelson, E.H. The laplacian pyramid as a compact image code. *IEEE Trans. Commun.* **1983**, *31*, 532–540.
44. Sevensols. Seven Solutions. Available online: http://www.sevensols.com/ (accessed on 28 December 2011).
45. Tomasi, M.; Vanegas, M.; Barranco, F.; Diaz, J.; Ros, E. High-performance optical-flow architecture based on a multiscale, multi-orientation phase-based model. *IEEE Trans. Circuits Syst. Video Technol.* **2010**, *20*, 1797–1807.
46. Vanegas, M.; Tomasi, M.; Diaz, J.; Ros, E. Multi-port abstraction layer for FPGA intensive memory exploitation applications. *J. Syst. Archit.* **2010**, *56*, 442–451.
47. Ortigosa, E.; Canas, A.; Ros, E.; Ortigosa, P.; Mota, S.; Diaz, J. Hardware description of multi-layer perceptrons with different abstraction levels. *Microprocess. Microsyst.* **2006**, *30*, 435–444.
48. Tomasi, M.; Barranco, F.; Vanegas, M.; Diaz, J.; Ros, E. Fine grain pipeline architecture for high performance phase-based optical flow computation. *J. Syst. Archit.* **2010**, *56*, 577–587.
49. Vision, M.C. Middlebury Computer Vision. Available online: http://vision.middlebury.edu/ (accessed on 28 December 2011).
50. Chessa, M.; Solari, F.; Sabatini, S.P. Virtual reality to simulate visual tasks for robotic systems. In *Virtual Reality*; Kim, J.J., Ed.; InTech: New York, NY, USA, 2010; pp. 71–92.
51. Chang, N.C.; Tsai, T.H.; Hsu, B.H.; Chen, Y.C.; Chang, T.S. Algorithm and architecture of disparity estimation with mini-census adaptive support weight. *IEEE Trans. Circuits Syst. Video Technol.* **2010**, *20*, 792–805.
52. Ernst, I.; Hirschmüller, H. Mutual information based semi-global stereo matching on the GPU. In *Proceedings of the 4th International Symposium on Advances in Visual Computing*, Las Vegas, NV, USA, December 2008; pp. 228–239.

53. Han, S.K.; Woo, S.; Jeong, M.H.; You, B.J. Improved-quality real-time stereo vision processor. In *Proceedings of the 22nd International Conference on VLSI Design*, New Delhi, India, 5–9 January 2009; pp. 287–292.
54. Gibson, J.; Marques, O. Stereo depth with a unified architecture GPU. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW '08)*, Anchorage, AK, USA, 23–28 June 2008; pp. 1–6.
55. Gong, M.; Yang, Y.H. Fast unambiguous stereo matching using reliability-based dynamic programming. *IEEE Trans. Pattern Anal. Mach. Intell.* **2005**, *27*, 998–1003.
56. Virtual Reality Tool for Active Vision. Available online: http://www.pspc.dibe.unige.it/Research/vr.html (accessed on 28 December 2011).
57. Carneiro, G.; Jepson, A. Multi-scale phase-based local features. In *Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Madison, WI, USA, 18–20 June 2003; pp. 736–743.
58. Oppenheim, A.; Lim, J. The importance of phase in signals. *Proc. IEEE* **1981**, *69*, 529–541.
59. Tomasi, M.; Vanegas, M.; Barranco, F.; Diaz, J.; Ros, E. Real-time architecture for a robust multi-scale stereo engine on FPGA. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2011**, doi: 10.1109/TVLSI.2011.2172007.
60. Fleet, D.; Jepson, A. Stability of phase information. *IEEE Trans. Pattern Anal. Mach. Intell.* **1993**, *15*, 1253–1268.
61. Bernardino, A.; Santos-Victor, J. Binocular tracking: Integrating perception and control. *IEEE Trans. Robot. Autom.* **1999**, *15*, 1080–1094.
62. Sapienza, M.; Hansard, M.; Horaud, R. *Real-Time 3D Reconstruction and Fixation with an Active Binocular Head*; Technical Report; INRIA: Le Chesnay Cedex, France, 2011.

4 Visual attention system

4.1 Visual attention architecture on-chip

The journal paper associated to this part of the dissertation **is still in process**:

- F. Barranco, Saliency-based Visual Attention Architecture for FPGA with Top-Down Modulation. **It will be submitted IEEE Trans. Systems, Man, and Cybernetics, Part A**.
    - Status: **In submission**.
    - Impact Factor (JCR 2009): 2.093
    - Subject Category:
        - ∗ Computer Science, Cybernetics. Ranking 5 / 19.
        - ∗ Computer Science, Theory and Methods. Ranking 15 / 97.

Saliency-based Visual Attention Architecture for FPGA with Top-Down Modulation

Francisco Barranco, Javier Diaz, Begoña Pino, and Eduardo Ros

***Abstract*—This paper designs a model and presents an architecture for the computation of visual attention for FPGA based on the combination of a bottom-up saliency and a top-down task-dependent modulation streams. The bottom-up stream is deployed including local energy, red-green and blue-yellow color opponencies, and different local orientation maps. Moreover, we also include motion in this bottom-up stream. We detail a thorough study of the strategies to simplify and parallelize the model in order to reduce the hardware utilization without a significant loss of accuracy. One of the most novel parts of this work is that the final saliency is modulated by two high-level features: optical flow and disparity. We also include some feedback masks to adapt the weights of the features that are part of the bottom-up stream, depending on the specific target application. In this work we also present a benchmark of the system and a comparison with other works by using well-known databases. Some results are also presented for the saliency computation. Finally, an example is also presented to visualize the role of the modulation stream in the field of driving scenarios.**

***Index Terms*—Field programmable gate arrays; Machine vision; Real time systems; Reconfigurable architectures; Visual attention; Saliency.**

I. Introduction

VISUAL attention is a crucial process of the human visual systems that allows visual searching by processing the huge amount of information that we receive. The bandwidth estimation of the information that comes down through the optic nerve is about 10^7 - 10^8 bit/s, quantity that widely exceeds our processing capacity [1]. The perception mechanisms are in general active i.e. they does not simply consist in an acquisition process but in active selection mechanisms of the relevant information, adaptable tuning filters and optimized strategies for efficiently control dynamic ranges, and attention mechanisms.

Visual attention deploys control optimized mechanisms for performing efficient searches in dynamic scenarios. There are three mechanisms related with attention: selection of spatial regions due to the stimulus activity, task-dependent restriction and suppresion or surround inhibition and inhibition of return, process that allows visiting next relevant locations after having attended the current one [2]. These three processes are developed as a combination of two information streams, a bottom-up inherent saliency map and a top-down task-dependent modulations [2] [1].

From the neurophysiologic point of view, some aspects of the visual attention are still controversial, for instance the brain regions that in some way participate in attention deployment. However, there is a consensus about that the visual information enters the primary visual cortex via the lateral geniculate nucleus and then, is processed along two parallel pathways: the dorsal stream that involves the spatial localization (‘where’ pathway) and directing the attention and gaze towards the more interesting objects in the scene and the ventral stream (‘what’ pathway) that involves the identification or recognition of the objects [1]. Attention deals only with the ’where’ pathway, focusing on the most relevant areas in the scene, selected due to its inherent relevance or to the target task dependency.

The first ideas about attention were formulated by Deutsch [3] and Norman and Norman [4] in the 1960’s, in the field of object recognition, determining the relevance of the selected regions in function of the image content. Treisman and Gelade proposed in 1980 their Feature Integration Theory, basis of a lot of subsequent models. Beginning with the results of real experiments for visual searching, the authors proposed a topographical feature ’master’ map for the representation of the image contents that summarizes the visual stimuli activity for the scenario, without explaining a combination function for obtaining this map. Previously, in 1976 Grossberg [5] presented an attention cell model that integrates bottom-up and top-down activation and top-down modulation. Koch and Ullman [6] presented for the first time a centralized “saliency map” mainly based on the Treisman and Gelade’s work [7] in 1985 . They proposed also a function for computing this saliency map, a winner-take-all network for the selection of the most relevant location and a inhibition or return mechanism. In 1998, Itti and Koch [8] complemented the Koch and Ullman’s work [6] considering a bottom-up model with spatial competition for saliency modeled after surround inhibition. Iteratively, a feature map receives additional inputs from the convolution with a difference-of-Gaussian filter and discards the locations with the lowest activities, simulating a winner-take-all network. A final combination of the different feature activity maps is performed to achieve the final saliency map. In general, this approach is the metric against which all the attention models are compared due to its accuracy predicting human gazing [9]. Due to this argument and to its potential parallel implementation, this model is the one selected for our implementation. After that, Navalpakkam in 2005 developed a model [10] for the integration of the task-dependent modulation in Itti and Koch’s scheme [8].

On the other hand, some authors designed models based on different hypothesis as Fukushima [11], that proposed a cell

F. Barranco, J. Diaz, B. Pino, and E. Ros are with the Department of Computer Architecture and Technology, CITIC, ETSIIT, University of Granada, C/Daniel Saucedo Aranda s/n, E18071, Granada, Spain. E-mails: {fbarranco, jdiaz, bego, eduardo}@atc.ugr.es

Manuscript received September XX, 2012; revised month day, year.

model in which the attention shifts causing the relaxation or inhibiting the attended cell, allowing the rest re-gaining their response to be subsequently attended. Wolfe [12] proposed a model there saliency is computed as the likelihood of a target will be at a given location, using an activation map that is biased also by a top-down modulation. Tsotsos [13] designed the Selective Tuning model, that uses a combination of a feedforward bottom-up extraction based on features and a feedback selective tuning of the extraction mechanisms, in a hierarchical way. Desimone and Duncan [14] also developed their own model, the Biased Competition model, that introduces the concept of a distributed modulatory scheme across the feature maps to explain the saliency map. Finally, Oliva and Torralba [15] predict the image locations to be fixated based on a Bayesian framework, combining bottom-up and top-down modulation.

Our work deals with the challenge of developing a visual attention system with real-time performances. A real-time attention process allows us to explore active vision strategies that permit adaptation to its inherent features. In addition to this, some implementation and testing strategies are not possible with off-line systems, as for example a camera in a mobile platform where the direction of the movement is determined by the continuous information extraction. The selected strategy to fulfill our real-time requirements is the hardware implementation of the system in an FPGA. The selected device presents more advantages as their low power consumption, a requirement that industrial applications demand, and its easy integration in robotic platforms. Some works in the literature have attempted similar objectives as in [16] which implemented a visual system with log-polar mapping and oriented filters in a cluster of Pentium II and III computers at 500 MHz, obtaining poor performances results. Or in the case of [17], applies its method based on integral images to extract the visual attention from luminance, color and orientation information, that obtains 400-by-300 images at 20 fps with a 2.8 GHz computer. Moreover, there are almost-real-time implementations [18] in a customized SIMD platform that combines analog and digitial circuitry achieving 14 fps for 64-by-64 image resolution, or GPU solutions as in [19] that proposed a saliency map computation based on Itti and Koch's model [8] obtaining 32 fps for 512-by-512 images with an NVIDIA 6600GT, or as in another proposal [20] in a NVIDIA 6800GT that computes 122.5 fps for 640-by-480 images (VGA), but using PCA for the selection of the features to compute the saliency fusion. In the case of FPGAs, a contribution [21] proposed an attention system based on information maximization using a Virtex-6 LX240T achieving 11.2 fps for VGA resolution. Another work [22], uses a neuromorphic retina chip and a bottom-up saliency map implemented in an FPGA, a Xilinx X3CS400. The saliency, based on the edge information that comes from the retina and that includes normalization and an inhibition-of-return stages, is computed for 128-by-128 images at 250 fps.

Finally, visual attention is applied in different target fields as, for example: predicting eye movements ("overt attention", [23] [24]), robotics navigation [25], video-compression [26], medical applications [27] or even in driving assistance systems or military applications [28] [29].

This paper is structured as follows: in Section II we present the mathematical formulation of the saliency-based attention system including the top-down modulation stream; Section III describes the hardware implementations for both approaches detailing the mono- and multi-scaled versions of both of them. This section also presents the performance analysis (accuracy and frame rate) and the resource utilization. Section IV presents and discusses the benchmark results for the saliency and the final system modulated with optical flow and disparity applied to driving scenarios. Finally, Section V presents the conclusions.

II. Saliency-based attention system with top-down bias

Our proposal is based on the works developed by Itti and Koch [8] [1] that model the bottom-up saliency, extended with the addition of optical flow and disparity information. In the case of the optical flow, the integration is performed in a bottom-up manner, using just the optical flow magnitude (since here we call it motion in this work). Finally, the complete optical flow (magnitude and direction of the motion) and disparity cues are integrated as a top-down modulation stream.

The bottom-up stream determines the most salient locations by selecting the areas with the highest contrast with the objects in its surroundings, for each feature and between different spatial resolutions. The features used in the mentioned basic model are local energy, orientation and color opponencies. In our case, we also add to all these features the motion of the objects in the scene. Finally, the disparity and the optical flow are integrated as a modulation stream for the final estimation, providing depth cues of the objects in the scene.

A. Bottom-up attention model (based on Itti and Koch's [8])

The hypothesis of the Itti and Koch's model is that the visual information is represented as a centralized topographical map in the early visual processing areas, created by a center-surround computation across different spatial scale resolutions and several features extracted of the scene. After that, the information belonging to each feature is collected in a "conspicuity map" and then, combined using spatial competition in the same way than a winner-take-all network in a unique "saliency map". The most salient location of these map, i.e. the location with the highest activity will be the first to be attended and then, an inhibition-of-return mechanism is carried out in order to suppress the activity of the attended location and to allow visiting the next most salient one.

The complete scheme illustrated with the execution of a driving scenario is shown in Fig. 1.

In detail, the first step of the model extracts features from the scenario in a hierarchical way, to allow the subsequent generation of the conspicuity maps using the information of several spatial resolutions. This hierarchical extraction is based on [30]. The basic initial model included three different features: the luminance, the local orientation and the color. Different subsequent works has added more features to this set as the symmetry, the motion direction, junctions, edges or

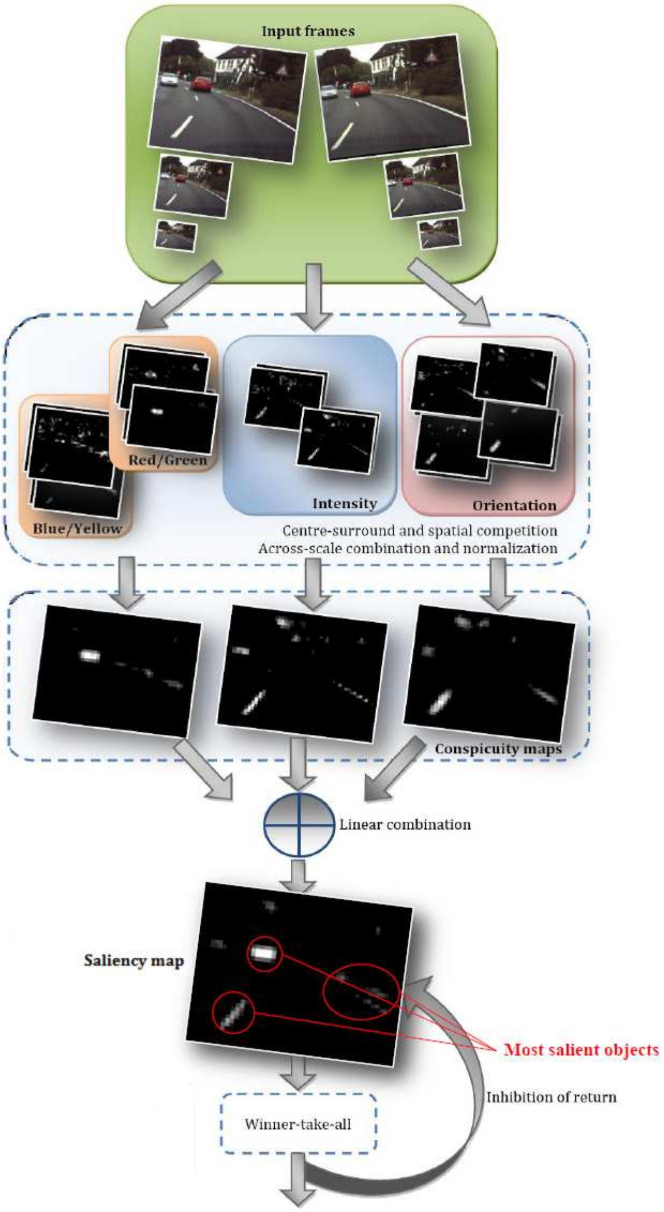


Fig. 1. Itti and Koch's model [8] example. The figure shows an example of application of the model with a pair of images of a driving scenario

even, skin and face detectors. A complete summary of the feasibility of features that may collaborate in the attention deployment can be found at [31].

A previous stage consists in sub-sampling the input image I into a Gaussian pyramid of factor two in the same way that [30], with 9 scales in the original model. Assuming an RGB-color input image, our first purpose consists in extracting the color opponencies as defined in [32], obtaining the red-green and the blue-yellow features as is shown in (1). The local orientation maps are generated using oriented Gabor filters for the orientations $0°$, $45°$, $90°$ and $135°$ as are defined in (2). These filters approximate accurately the receptive visual fields for the orientation-selective neurons in the visual cortex. In the case of the luminance, taking into account that we have computed the Gabor filter responses, we generate it as square root of the average of the energy response for each orientation O_θ that is computed as the amplitude of the complex response (see 4)).

$$\begin{aligned} M_{RG} &= \frac{r-g}{max(r,g,b)} \\ M_{BY} &= \frac{b-min(r,g)}{max(r,g,b)} \end{aligned} \tag{1}$$

$$\begin{aligned} G_\theta(\mathbf{x}) &= e^{\frac{x^2+y^2}{2\delta^2}} e^{jw_0(xcos\theta+ysin\theta)} \\ Q_\theta(\mathbf{x}) &= (I * G_\theta)(\mathbf{x}) = \rho_\theta(\mathbf{x}) e^{j\phi_\theta(\mathbf{x})} \\ &= C_\theta(\mathbf{x}) + jS_\theta(\mathbf{x}) \end{aligned} \tag{2}$$

$$M_\theta = C_\theta^2(\mathbf{x}) + S_\theta^2(\mathbf{x}) \tag{3}$$

$$M_I = \sqrt{\frac{\sum_N M_\theta}{N}} \tag{4}$$

Gabor filters $G_\theta(\mathbf{x})$ are defined in 2 for a specific orientation θ and pixel $\mathbf{x} = (x,y)^T$. Moreover, w_0 is the peak frequency and σ the standard deviation. The second line $*$ stands for the convolution of the Gabor filter with the input image (intensity). As the response of the filter is complex, ρ_θ and ϕ_θ are the amplitude and the phase components, and C_θ and S_θ the real and imaginary responses of the quadrature filter pair. For more details about the implementation of this filter bank, see [33].

The next stage consists in simulating the role of the center-surround receptive fields. It is performed by an across-scale subtraction $\ominus$ between a center c and the surround s scale levels for two feature maps M_f as in (5).

$$\overline{M}_f = N(|M_f(c) \ominus M_f(s)|) \tag{5}$$

In this case, the N represents a normalizer operator that simulates local competition. Between the proposals in [34] for implementing this operator, we have selected the iterative localized interaction. This method simulates three interactions observed in the visual processing: the inhibitory interaction from the surround to the center [35], this inhibition is stronger for the neurons of the same feature and finally, it is also stronger at a specific range distance from the center [36] and [37]. This interactions may be simulated using difference of Gaussians, convolving iteratively the feature maps with the same parameters as in [38]. The computation in the original model is performed by convolving two separable convolutions, the excitatory and inhibitory Gaussian, iteratively 10 times. In fact, the iterative process is the main problem for a potential parallel implementation.

In the original work [8], the across scale subtraction is done for 6 combinations of the resolution levels and is summed into a single feature map F_f as in (6), finally achieving 42 maps.

$$\overline{F}_f = N\left(\oplus_{c=2}^{4} \oplus_{s=c+3}^{c+4} \overline{M}_f\right) \tag{6}$$

Next, it computed the "conspicuity maps" for each feature map C_F, as in (7).

$$\begin{aligned}
\overline{C}_I &= \overline{F}_I \\
\overline{C}_C &= N\left(\sum_{C\in\{RG,BY\}} \overline{F}_C\right) \\
\overline{C}_O &= N\left(\sum_{O\in\{0,45,90,135\}} \overline{F}_O\right)
\end{aligned} \tag{7}$$

Finally, the proposed method generates the final saliency map S as in (8)

$$S = \sum_{F\in\{I,C,O\}} \overline{C}_F \tag{8}$$

After the generation of the saliency map, its maximum peak corresponds with the next location to be attended. The maximum operator is performed using a winner-take-all network [6] [39]. This process selects a location that is visited as the most salient one and, an inhibition of return process suppresses the reinforcement at this location to visit the next most salient location in an iterative manner.

A criticism about the presented model is the lack of top-down bias. This scheme models exclusively the bottom-up saliency stream without using any knowledge about the environment which might be a real problem performing a task in natural scenes. However, this problem is also partially solved in our implementation by using high-level cues as the motion and the depth of the objects and also a feedback to adapt the weights of the different channels for the bottom-up saliency system depending on a specific target application.

B. Optical Flow and Disparity Integration

As mentioned, the depth and motion estimations for the scenario are also valuable for the computation of the visual attention. Depth is computed as a disparity or difference of the projections of the real image in a binocular system. After selecting an object of interest, the computation performs a search of the correspondence between left and right images (complex due to the ambiguity). In the case of the optical flow, it estimates a two-dimensional motion field of the scene at different time instants, hence estimating the direction of the motion and its magnitude. In both cases, specific tuning methods are required in order to improve the accuracy of the estimations, as the multi-scale computation.

There exist local and global methods for optical flow and disparity estimations. Local methods estimates the displacement of a pixel by centering in its surrounding neighborhood. On the other hand, global methods use a diffusion process that globally propagates the local information across the whole image. We estimate both optical flow and disparity with the well-known Lucas&Kanade algorithms [40] [41], based on the local energy gradient. This algorithm provides a good trade-off between the result accuracy and its computational cost. Due to its local computation, this algorithm efficiently estimates velocities for a few pixels [42]. Resolving the problem for larger displacements usually involves the application of multiscale or coarse-to-fine methods (inspired in [43]), which consequently provides more accurate results with a notable increase in the computational cost.

We have extensively explored the hardware implementation of these algorithms in FPGA in previous works. For example, in the case of optical flow, we have designed architectures for monoscale versions [42], for mono and multiscale phase-based alternatives [44] [45] [46], or multiscale gradient-based [47]. In the case of disparity, we have also addressed implementations for mono and multiscale gradient-based methods [48] [49] and phase-based versions [46] [50]. More details of the implementations of the hardware algorithms used in this paper can be found specifically in [47].

The hardware implementation of optical flow and disparity in FPGA allows their integration with the saliency-based attentional system. In addition to this, their real-time performances also maintain the fulfillment of our requirements.

The way in which we integrate optical flow and disparity into our attention final architecture is different for each one. In the case of the motion (optical flow magnitude), it is integrated in a bottom-up manner. Initially, the addition of motion in this manner is easily understandable, due to the fact that in a dynamic environment as the real-world, any contrast in movement deserves instinctively our attention. Moreover, this choice is supported by various works as in [51] [52] [53] and, specially by the Horowitz and Wolfe neurophysiological studies that also consider motion as a bottom-up saliency feature [12] [31] [54]. Following these contributions, motion is included in our case by just using the magnitude of the estimated optical flow, as a unique map. In such a way, the equations in (7) are completed with the computation of the "motion conspicuity map" C_M in (9)

$$\overline{C}_M = \overline{F}_O \tag{9}$$

where F_O is the sum of feature maps after the across-scale subtraction for the different spatial resolutions. After that, final saliency computation in (8) is rewritten as in (10)

$$S = \sum_{F\in\{I,C,O,M\}} \overline{C}_F \tag{10}$$

On the other hand, as mentioned before, the disparity is not included as a bottom-up feature. This decision is also supported by [31] that considers stereoscopic depth as just a "probable attribute" in attention deployment. Nevertheless, it is clear that this attribute may help in various applications as robotics, navigation or driving assistance. In all these applications the distance between the subject and the obstacles or general objects in the scene is crucial for any task to be addressed. In this case, disparity is used as a bias or modulation top-down stream that influences the determines the next most salient location in the last step, after generating the saliency map computation of (10).

In addition to this, regardless optical flow is not used for the bottom-up saliency, it is actually computed for extracting the motion magnitude and it may be useful for the top-down modulation stream. Hence, optical flow may also be meaningful in selecting the most salient locations depending

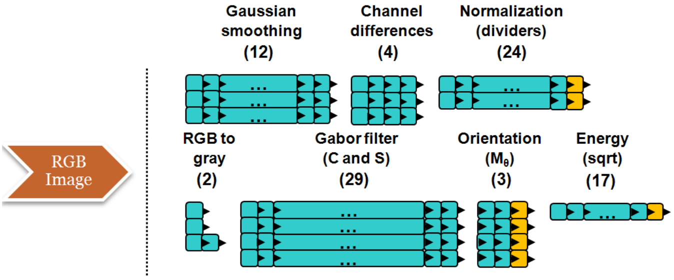


Fig. 2. Hardware scheme of the micro-pipelined datapaths for the energy, orientation, and color opponency computations. The number in parenthesis are the number of stages of each datapath. A final stage is required to synchronize the extraction of the different features (we use embedded FIFO memory blocks)

on the application. Finally, we also include some weights for the combination of the feature maps.

Summarizing, the final saliency map computation is addressed as in (11), where the ω_F are the top-down weights for the feature maps

$$S = \sum_{F \in \{I,C,O,M\}} \omega_F \overline{C}_F \qquad (11)$$

And the selection of the next most salient location P is carried out using a WTA strategy (winner-take-all) implemented as in (12), where D is the disparity map, OF is the optical flow and f and g are the selection functions for these visual modalities respectively. In both cases, it is easier to focus the attention of the system in the objects that are not further than several meters and those that are moving towards us simultaneously.

$$P = WTA\left(S, f\left(D\right), g\left(OF\right)\right) \qquad (12)$$

III. Hardware Implementation

As mentioned, the FPGA is the most appropriate platform selected for our implementation. This device allows us to take advantage of a massively parallel architecture for achieving real-time performances in an embedded system. Several additional advantages refer to the its easy integration with robotic platforms, certification suitability, low power consumption, and small size. In our case, our FPGA is a Xilinx Virtex4 chip (XC4vfx100), and the board is a XircaV4 manufactured by SevenSols [55] with PCIe interface and four SRAM ZBT memory banks. This platform offers the possibility of working separately as a stand-alone system or as a co-processing board connected with a computer.

Our architecture is developed using fine-pipelined datapaths which allows an adaptable design with low-power consumption and high performances [42]. In our case, the arithmetic is fixed point, which provides solutions with affordable resource cost and acceptable loss of accuracy after carefully choosing the bit-width of the involved operations.

The hardware implementation of our architecture is carried out using two HDL (hardware description languages): the implementation of the modules of communication protocols and memory interfaces are implemented in VHDL (see [56]); the multi-scale architecture and the saliency, optical flow and disparity modules are developed using Handel-C, a C-like language that permits more easily the algorithmic descriptions (without any significant increase in resources or degradation in performances [57]).

A. Hardware Saliency implementation

As mentioned, the saliency generation is based on [8]. The first step of this implementation is the extraction of the local primitives, in our case: energy, orientation and color opponencies. In the case of color, the inputs of the system are the R, G, and B components, so it only computes the RG and BY color opponencies according to (1). To achieve them, it is required the use of two divider cores, one of each color opponency. These cores are generated using the Xilinx CoreGenerator tool [58]. For the orientation and the energy we use a bank of Gabor oriented-filters, in our case with N=4 different orientations. The Gabor filters are computed with 11 taps for the orientations 0°, $45°$, $90°$, and $135°$. In addition to this, we also require a root square core for the computation of the energy. This core is also generated with the Xilinx CoreGenerator tool. To synchronize the outputs of the diverse extracted features we also use embedded FIFO memories (Block-RAMs). Fig. 2 illustrates a simple scheme of the hardware computation for energy, orientation and color opponency. The number in parentheses stands for the number of micro-pipelined taps for each stage.

As mentioned, the feature extraction is performed at different spatial scales. This computation is carried out by iteratively sub-sampling the input image I with a factor of 2. The number of iterations depends on the number of levels of the pyramid, in our case it is 6. The first part of this circuit consists in a smoothing filter, by convolving with a two-dimensional separable Gaussian filter of 5 taps whose kernel is $K = [1\ 4\ 6\ 4\ 1]/16$ as suggested in [30]. This pre-filtering helps reducing the aliasing effects generated by the subsequent sampling process. As the sub-sampling factor is 2, we discard one of each 2 smoothed pixels. Similar modules to this pyramid computation are used in previous works as in [45] [47] [50].

Once the 42 maps have been computed (6 spatial resolutions x 7 feature maps), the next stage consists in resizing all these maps to the resolution of the final saliency map. We set up this resolution to the $(original resolution)/(2^3)$ (the resolution of the third scale). In the case of scales 6^{th}, 5^{th}, and 4^{th} we require 3, 2, and 1 modules for the reduction to the selected spatial resolution; for scales 2^{nd} and 1^{st} we respectively require 1 and 2 additional modules for expansion (all of them with a factor 2). The reduction circuit is based on the same basis than the modules for the Gaussian pyramid. They consist in the convolution with a smoothing filter (the same than in the case of the pyramid) and after this filtering, the selection of 1 of each 2 filtered pixels. Connecting in cascade 2 and 3 filters, we achieve the reductions of factor 2^2 and 2^3 that we need for scales 4^{th}, 5^{th}, and 6^{th} respectively. The expansion circuit computes separately the values for the columns and the rows. For columns, for each cycle it sends two pixels, the original one and the computed as the interpolation between it and the next one. For rows, it stores a row (using embedded FIFOs) and then writes a row and the next one as the interpolation between the correspondent value of the stored and the current one. In a similar way to the reduction computation, connecting in cascade two of these circuits provides a 2^2 expansion.

After resizing the maps to the selected final resolution, the next stage is the across-scale center-surround inhibition and normalization. For each feature, we firstly performs the across-scale center-surround inhibition as the subtraction of the maps according with the following sequence {6-3, 6-2, 5-2, 4-1, 4-1}.

After this step, we should normalize the resultant 35 maps as a previous stage to the combination of the results for each feature to compute the conspicuity maps. As mentioned, for the normalization we have selected the iterative localized interaction alternative. It basically consists in iteratively convolving the feature maps with a separable difference of Gaussians kernel. The computation is performed by convolving by the excitatory and inhibitory separable kernels and then, by subtracting the partial results. The next subsection is dedicated to this normalization operator.

After the normalization, the conspicuity maps are computed by the summation of the different maps for each feature. After this summation, a new normalization is required. The final combination of the conspicuity maps leads to the saliency map. The most salient location is selected using a WTA mechanisms for this map (using a maximum operator). After the selection, the inhibition of return is performed over the selected location by using a disc whose diameter is computed as the 10% of the size of the minimun dimension of the image (height or width) and then, a new location can be selected.

1) Normalization operator: In the original model, the Difference of Gaussian is emulated with a excitatory kernel which is generated using a variance of $\sigma_{ex} = 2\%$, and the inhibitory kernel with a $\sigma_{inh} = 25\%$ of the image width. However, this means very large two-dimensional filters which leads to a high resource utilization and to store more rows for the convolution computation. In our case, we use filters of 15 and 9 taps for the inhibitory and excitatory filters respectively. The original process is shown in (13)

$$M \longleftarrow |M + M * DoG - C_{inh}|_{\geq} 0 \tag{13}$$

where M is the map, $*$ depicts the convolution, C_{inh} is a constant inhibitory term and $||_{\geq} 0$ means that we discard the negative values. But in our case, to reduce the number of iterations in the hardware model (in order to exploit the maximum parallelism) we use (14)

$$M \longleftarrow \left| M + \sqrt{(M * DoG)} - C_{inh} \right|_{\geq} 0 \tag{14}$$

As seen in the example (see Fig. 3), using the square of the convolution by the difference of Gaussian, we can achieve a similar result and save a large amount of resources. As values are normalized and lower than 1, we use the square root in the notation instead of the power of two.

In addition to this, the hardware implementation of this operator presents also a high complexity. As commented previously, the filtering is carried out by using separable kernels. For the separable computation of the convolution, we require storing the rows that we are using for computing the current pixel to calculate in parallel the result for each pixel. In such a case, as the excitatory kernel uses 15 taps and the inhibitory 9 tamps, we need to store at the same time 22 rows. On the other hand, the storage of the image lines is performed in embedded memory FIFOs.

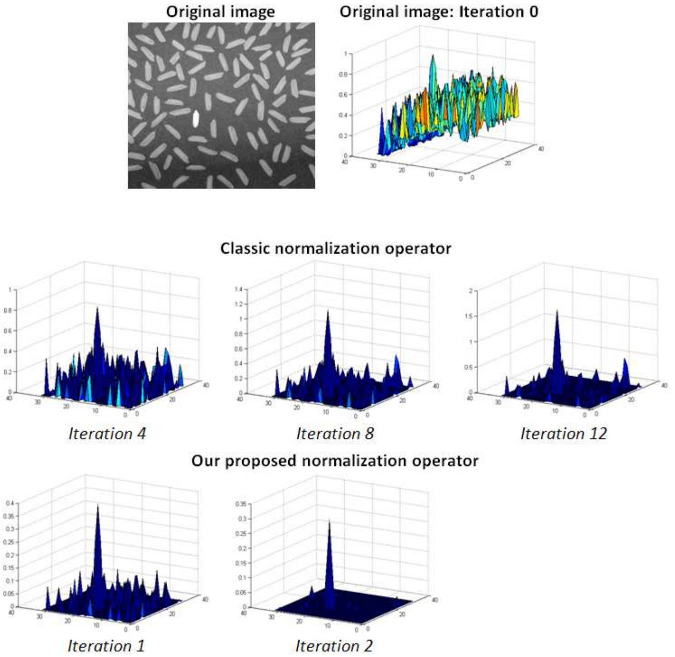


Fig. 3. Normalization operator proposal. The first row shows the original image, the second row the iterations 4^{th}, 8^{th}, and 12^{th} using the classical operator in [34]. Finally, the third row shows two iterations of the modified normalization operator. In the hardware model we only iterate once to save hardware resources.

In the worst case, applying the data proposed in [34] and instantiating a BlockRAM memory per each row, considering a VGA input image (640x480) we would need:

- Excitatory kernel: The separable kernel size would be 361 $(3 * \sigma_{ex} + 1)$. We would need 360 BlockRAMs for the 360 rows we need to store, for each data:
  *5 maps * 360 rows/map = 1800 BRAMs*
  And then, the occupation of each BRAM is:
  *7 features/data * 9 bits/feature * 60 data/row = 3780 bits/row*
- Inhibitory kernel: The separable kernel size would be 29. And the, we would need to store 28 rows per each data:
  *5 maps * 28 rows/map = 140 BRAMs*
  With the same occupation as in the previous case.

In fact, optimizing the memory occupation to a theoretical maximum, we would need 1840 BRAMs x 3780 bits/BRAM = 6793 Kb.Furthermore, we also require a second normalization stage before computing the conspicuity maps, the previous step for the combination to obtain the final saliency map. In this case, we have only 3 maps and a single feature for each map:

- Excitatory kernel: We would need 360 BlockRAMs for the 360 rows we need to store, for each data:
  *3 maps * 360 rows/map = 1080 BRAMs*
  And then, the occupation of each BRAM is:

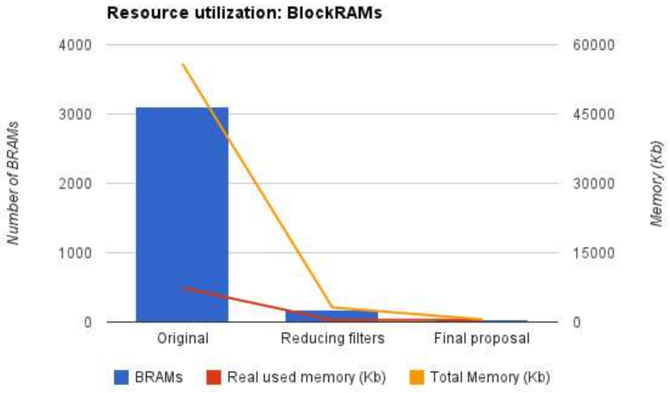


Fig. 4. BlockRAM utilization. The graphic depicts the number of BlockRAMs (left axis) and memory usage for three alternatives (right axis). The memory usage is shown taking into account the real memory that is written and the memory that is used since the instantiation of a BlockRAMs does not allow to use the memory that is not written for additional data.

*1 features/data * 9 bits/feature * 60 data/row = 540 bits/row*

- Inhibitory kernel: We would need to store 28 rows per data:
  *3 maps * 28 rows/map = 84 BRAMs*
  With the same occupation as in the previous case.

Thus, we would need 3104 BRAMs, and at least in the best case up to 7406 Kb.

The first step to reduce the embedded memory utilization is the use of smaller kernels. In our case, we use kernels with 15 and 9 taps. This means that we require storing 22 rows per each data. With the same computation than in the previous case, instantiating a BlockRAM per each row that we need to store, we would need 176 BRAMs reducing approximately the required BRAMs in a 90%. With this scheme, the embedded memory that we are really using is about 441 Kb (we only write about 3.69 Kb of each BRAM for the first normalization and 0.53 Kb for the second one).

The previous data does consider the BlockRAMs that is used in the other modules. For example, the initial architecture with the communication with the board, the memory interface and the pyramid computation uses 74 BlockRAMs. Moreover, the expansion and reduction circuits also requires BRAMs (8 BlockRAMs each one of them). Hence, we need a different approach to save more resources. As commented, of each BlockRAM we are exploiting only about 3.69 Kb when their size is 18 Kb (in the case of our FPGA). The new scheme consists in storing 4 complete rows per each BRAM in the first case, and the 22 rows in the same BlockRAM for the second normalization. With this final implementation we only need 31 BlockRAMs although the real embedded memory usage is about 458 Kb. A summary of the BlockRAMs and memory used for the three mentioned cases are shown in Fig. 4.

B. Complete System Architecture

As mentioned, the saliency model is completed with a top-down modulation stream that is deployed using depth and optical flow cues. For their computation, we use two cores that estimate disparity and optical flow respectively. The computation of these modules are based on gradient-based algorithms and their implementation are detailed in [47] [48].

In this section we briefly summarize the architecture for the computation of both cores. Both cores are based on the Lucas & Kanade algorithm (for more details see [40] [47]), that assumes the constancy of the image intensity between the consecutive frames for the optical flow or, between left and right images for the disparity (in a stereo system). As the system is ill-posed with just one assumption, it also assumes that the optical flow or the disparity is very similar in small neighborhoods. The implementation uses 3 consecutive frames for the optical flow and 2 (right and left images) for the disparity computation.

After a first filtering of the input frames with a Gaussian smoothing kernel to reduce the aliasing effects, the approaches apply a new filtering stage to compute the derivatives (the spatial derivatives in both cases and the temporal one in the case of the optical flow). The kernels used in both cases are inspired in [59]. Once all the partial derivatives have been computed, it generates the Hessian matrix and solves the system.

Due to its local computation, this algorithm efficiently estimates velocity for small ranges, but not for large interframe displacements or disparities (the typical range is a few pixels [42]). The multiscale extension consists in compensating motion or disparity scale-by-scale (warping), and it is also called coarse-to-fine approach with warping. This extension improves substantially the accuracy and the motion or disparity range of the system (inspired in [43]). Therefore, taking advantage of the pyramidal computation performed for the saliency deployment, we also include a multiscale extension for the optical flow and disparity computation. In both cases, the implementation is based on the same architecture than in [47]. We also includes a homogenization post-processing stage in the architecture to refine the estimates by using a median filtering operator. Fig. 6 visualizes on the right side the described multiscale extension.

C. Hardware Resource Utilization

In computer vision, it is crucial a good trade-off between the required accuracy and the constraints about the maximum frequency and the resource utilization, particularly whether we are going to embed an architecture into a specific purpose device as in our case. Table I shows the resource utilization of the implemented system. This table presents the information about the board resource utilization: total number of 4 input LUTs, Slice Flip Flops, Slices, DSPs, Block RAMs used and finally, the maximum frequency (MHz) allowed in each module.

The listed options include: the multi-scale optical flow and disparity system, as well as the optical flow core and the disparity core individually; the feature extraction and the across-scale center-surround subtraction (called Incl. across-scale subtraction), and finally the complete saliency system. We also present the information for the interface with the resources of our board and the memory control unit.

Table I summarizes the resource demand for the different modules of our system. As seen, the implementation of the

	4 input LUTs	Slice Flip-Flops	Slices	DSP	Block RAM	Freq
	(out of 84352)	(out of 84352)	(out of 42716)	(160)	(378)	(MHz)
Board and memory interfaces	4774 (5%)	5195 (6%)	5388 (12%)	0	36 (9%)	112
Incl. across-scale subtraction	43535 (51%)	21034 (24 %)	30047 (71%)	8 (5%)	97 (25%)	50
Saliency system	66708 (79%)	31684 (37%)	39176 (92%)	11 (6%)	126 (33%)	50
O.F. core	4589 (5%)	6622 (5%)	4128 (9%)	30 (18%)	48 (12%)	83
Disp. core	2358 (2%)	3120 (3%)	2129 (5%)	3 (1%)	28 (7%)	108
OF and disp. System	41613(49%)	32706 (38%)	32442 (76%)	49 (30%)	162 (43%)	44

TABLE I
Hardware resource utilization for the presented complete architecture using a Virtex-4 FX100 FPGA.

multsicale-with-warping disparity and optical flow system already utilizes 76% of the available resources. Moreover, the saliency system uses the 92% of the resources. Summing the embedded BRAMs for both systems we require 288 of these units (although we are not taking into account 40 BRAMs that are demanded by the interface with the board and the memory). This justifies the high effort addressed in Section 3 to reduce the number of utilized BRAMs. However, the optimization may be more powerful depending on the final architecture and the resources that are required for the architecture that in our case, is the same in both cases.

Additionally, the frequency of the final system reaches about 44 MHz (the maximum clock frequency of the optical flow and disparity computation), obtaining a rate of up to 30 fps (fulfilling our real-time requirements). Furthermore, as commented previously, one of the main important advantages of FPGA-based implementations is their low-power consumption (a valuable point for its integration with real platforms). In our case, the total power consumption is about 2.64 w. The value has been estimated using the Xilinx tool XPower Analyzer [58], using a profile of a logic-intensive design that set up a high dynamic power (using for the worst case a toggle rate of 20%).

IV. Benchmarking

The benchmarking of this part of the system is split into two different parts. Firstly, the saliency benchmarking is performed using a well-known database available at [60], and compared with the performances of the Itti and Koch's model [8] that can be found in [34]. We also show some examples of saliency maps for some images of the database.

The second part of this benchmark section is dedicated to show the combination of the bottom-up saliency stream with top-down cues (optical flow and disparity) and its potential for traffic applications. In this part we also includes motion as a feature that is part of the bottom-up stream.

A. Saliency-based benchmarking

We use three databases of natural images: the first one are images that contain a red can, the second database consists of images with a car emergency triangle and, the third one is composed by images of German roads that contains traffic signs (59, 32 and 45 images respectively). For all the images, the database provides binary masks for the object of interest. According with [34], the target is considered to be detected if the focus of attention (FOA) intersects it. The radius of the FOA for 640x480 images (red can and car triangle databases) is 80, and for the 512x384 images (traffic signs) is of 64. In our case, if the object is not detected after 10 trails, the image is discarded.

Table II presents the benchmark results for different proposals. Firstly, we evaluate the original work based on Itti and Koch's model [8]. The second proposal is based on the previous work but including the modifications that we have explained in the previous section for the normalization operator: using different variances for the excitatory and inhibitory kernels and just two iterations of (14) instead of the 10 iterations of the original (13). The third group of rows are generated considering the model that we have proposed (except that we do not use the motion because the database is of images, not sequences). The method includes a the computation of the energy and the orientation with different algorithms than in the case of [8], the normalization operator uses only one iteration of (14), as explained, the variances are also lower than in the case of the first algorithm, we use only 6 scales (instead of the 9 of the original model) and the sequence of maps that is subtracted in the across-scale center-surround inhibition is also different. Finally, the Hardware proposal includes all the listed changes and the with the use of a fixed-point arithmetic which means a constrained precision, the use of hardware cores for the implementation of dividers or square root cores, and the implementation of a hardware-like normalization operator as mentioned before.

For each method, we are presenting the average number of false detection before the most salient location is selected (AVG), the standard deviation of this error (STD), and the number of images that we are using for the computation (Valids). We use this last value because we discard the images if we do not find the most salient location for 10 tries.

As seen in the results, our proposal or its hardware version maintain a good accuracy in terms of number of false positives compare with the original version of the algorithm. Furthermore, the second group of results ("Itti-Koch model with new normalization operator") also shows that the modification of the variances of the Difference-of-Gaussian kernels, and the use of the normalization as in (14) does not have a significant impact for the final saliency computation in terms of accuracy but, it allows us to reduce the computational complexity

	Itti-Koch model [8]			Itti-Koch model with new normalization operator			Our proposal			Our HW proposal		
	AVG	STD	Valids	AVG	STD	Valids	AVG	STD	Valids	AVG	STD	Valids
Red can	1.24	1.42	59	0.92	1.81	58	1.07	1.73	56	1.24	2.1	55
Triangle	1.42	1.67	32	1.26	2.16	31	1.27	2.3	30	1.17	1.69	29
Autobahn	0.52	1.05	45	0.49	1.38	45	0.43	1.27	44	0.34	1.08	44

TABLE II
BENCHMARKING OF THE BOTTOM-UP MODELS USING THE IMAGE WELL-KNOWN DATABASE AVAILABLE AT [60].

of the model and therefore, the final hardware resources of our architecture. A detailed discussion about the hardware resources is presented in the next Section.

Fig. 5 shows some saliency results for several images from the benchmark of [60]. The figure visualizes the original image, the saliency map, and the conspicuity maps for energy, orientation, and color. The red can, the triangle and the traffic sign are the most relevant locations in each case, and the saliency maps select them as the most salient (red pixels). The most significant feature in the three cases is color but, orientation also adds important information. The intensity is hard particularly in the case of the triangle, due to the sun and the light artifact.

B. Full Architecture Examples

As mentioned in Section 2, motion (only the optical flow magnitude) is also used in the final architecture as part of the bottom-up saliency deployment. As commented, the features that are part of the saliency computation are inherent to the targets and, they are salient due to the contrast between them and their surrounding areas. Additionally, as mentioned in Section 2, there exist also evidences that support the use of motion as one of the features that take part into the saliency deployment [31]. In Fig. 6 we show a model for the visual attention system that includes this modification with respect to the original algorithm.

In addition to this, the final architecture also includes the a top-down modulation stream that is task-dependent. In this case, the full system is tested for driving scenarios. For the computation of this top-down bias we use disparity and optical flow cues, visual features that are very valuable and widely used in our specific field. The combination of the top-down cues allows us to select the object of interest that, in our case are further than a few meters and, that have a specific optical flow pattern: cars that are approaching to us or the ones that have our own movement direction and a relatively high speed. As we are applying our computation to driving scenarios, these patterns make sense to maintain a driver aware of the cars in the road. As seen in the example, in the case of the system with only the bottom-up stream, we select different elements as the road markers, the traffic signs, or the cars. After applying the top-down modulation, cars are selected as the most relevant objects of interest.

V. CONCLUSIONS

In this paper we present a model for the deployment of visual attention based on Itti and Koch's works [8] [6] that model only the bottom-up saliency stream, for an FPGA device. Our method, as explained in this work, includes motion as a feature that is part of the computation of the saliency in a bottom-up way. As commented, we also use different methods for the computation of the local orientation, the local energy, and the color opponencies. Furthermore, we also model a different combination for the across-scale computation, reducing the number of maps in order to reduce the total required resources. Finally, we also propose a new method for the DoG normalization operator that reduces the iterative processing of the original approach in order to allow a better exploitation of the parallel massively resources that are available in our device. It also reduces the resources by using smaller kernels for the inhibitory and excitatory kernels. A specific study of this resource reduction is also addressed.

We have also modeled a system for the previous approach that integrates that bottom-up saliency stream with a top-down task-dependent modulation that is also embedded in the mentioned device. The modulation biases the results allowing the selection of a different region of interest influenced by the task that is being performed. In our case, we compute optical flow and disparity to generate this modulation stream. The application of the system to the driving scenarios justify the selection of these visual modalities.

Additionally, we have benchmarked the saliency stream using a well-known test database [60] and have shown that our proposals (the implementation of the model and the hardware model results) are good in terms of accuracy, compared with the original implementation of the model. We have also benchmarked the DoG normalization operator, because this part of the algorithm is relevant for the saliency implementation. We have shown that our optimized-cost implementation has no significant impact for the accuracy of the system.

The computation in an FPGA is particularly relevant to allow us to achieve real-time performance. The implementation of the complete architecture utilizes more than 90% of the available resources. For this system, the maximum clock frequency is about 44 MHz. This respresents up to 30 fps with a VGA resolution. The multiscale-with-warping computation of the optical flow and the disparity represent in this case the system bottle-neck. Furthermore, as we commented, one of the advantages of the FPGA implementation consists in its integration in a robotic or industrial platform, mainly due to the low-power consumption. In our case, the total power consumption is 2.64 w.

Finally, we have evaluated the complete architecture for a specific driving scenario, showing that the top-down modulation may be useful. Optical flow and disparity cues allow us

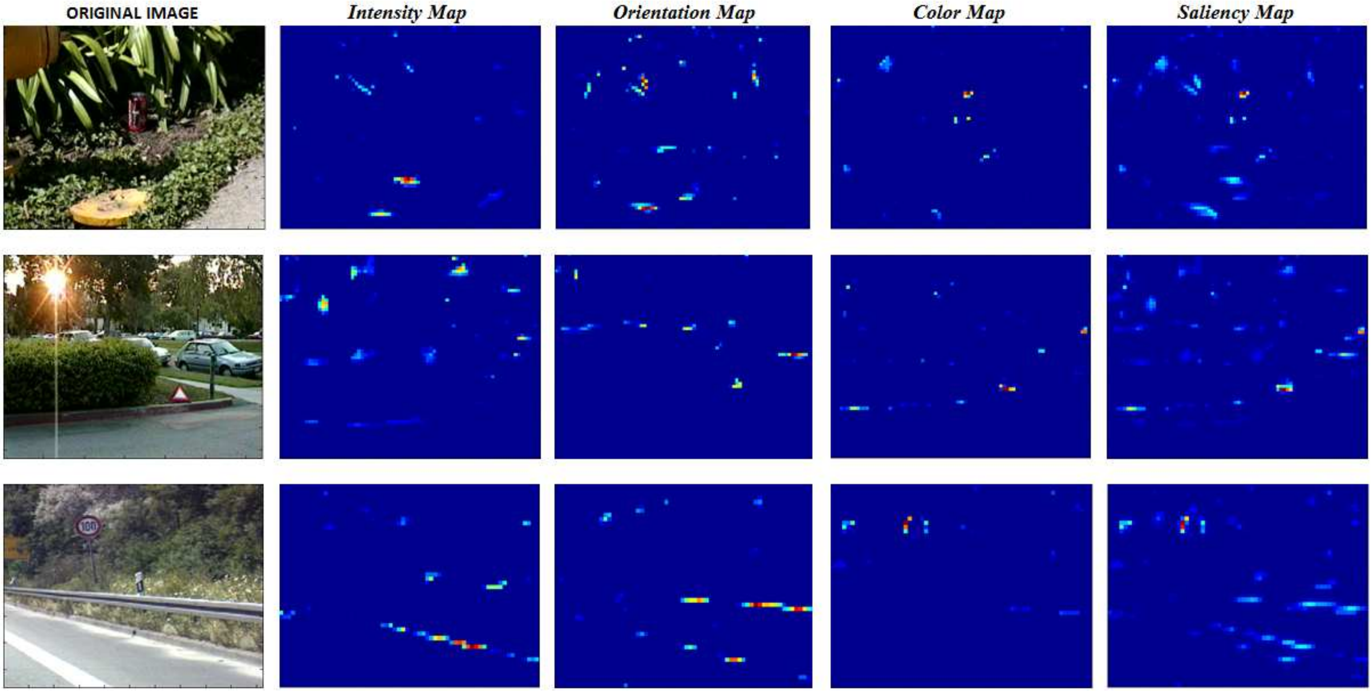


Fig. 5. Intensity, orientation, and color conspicuity maps, and saliency map results for some benchmark images from the database [60]. From top to bottom, images are from 'red can', 'triangle' and 'autobahn' series.

to select different targets of interest, in our case selecting the cars in the scene that are driving on the road.

ACKNOWLEDGMENT

This work has been supported by the National Grant (AP2007-00275), the projects ARC-VISION (TEC2010-15396) and MULTIVISION (TIC-3873) and the EU project TOMSY (FP7-270436).

REFERENCES

[1] L. Itti and C. Koch, "Computational modelling of visual attention," *Nature Review Neuroscience*, vol. 2, no. 3, pp. 194 – 203, Mar. 2001.

[2] J. K. Tsotsos, L. Itti, and G. Rees, "A brief and selective history of attention," in *Neurobiology of Attention*, L. Itti, G. Rees, and J. K. Tsotsos, Eds. San Diego, CA: Elsevier, Jan 2005, pp. xxiii – xxxii.

[3] J. A. Deutsch and D. Deutsch, "Attention: Some theoretical considerations." *Psychological Review*, vol. 70, no. 1, pp. 51 – 60, 1963.

[4] D. A. Norman, "Toward a theory of memory and attention," *Psychological Review*, vol. 75, no. 6, pp. 522 – 536, 1968.

[5] S. Grossberg, "Adaptive pattern classification and universal recording: II. Feedback, expectation, olfaction, illusions," *Biological Cybernetics*, vol. 23, pp. 187–202, 1976.

[6] C. Koch and S. Ullman, "Shifts in selective visual attention: towards the underlying neural circuitry." *Human neurobiology*, vol. 4, no. 4, pp. 219 – 227, 1985.

[7] A. M. Treisman and G. Gelade, "A feature-integration theory of attention," *Cognitive Psychology*, vol. 12, no. 1, pp. 97 – 136, Jan. 1980.

[8] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 11, pp. 1254 – 1259, Nov. 1998.

[9] N. Bruce, "Saliency, attention and visual search: an information theoretic approach," Ph.D. dissertation, 2008.

[10] V. Navalpakkam and L. Itti, "Modeling the influence of task on attention," *Vision Research*, vol. 45, no. 2, pp. 205–231, Jan 2005.

[11] K. Fukushima, "A neural network model for selective attention in visual pattern recognition," *Biological Cybernetics*, vol. 55, no. 1, pp. 5 – 15, 1986.

[12] J. M. Wolfe, K. R. Cave, and S. L. Franzel, "Guided search: an alternative to the feature integration model for visual search." *Journal of experimental psychology. Human perception and performance*, vol. 15, no. 3, pp. 419 – 433, Aug. 1989.

[13] J. K. Tsotsos, S. M. Culhane, W. Y. K. Winky, Y. Lai, N. Davis, and F. Nuflo, "Modeling visual attention via selective tuning," *Artificial Intelligence*, vol. 78, no. 1-2, pp. 507 – 545, Oct. 1995.

[14] R. Desimone and J. Duncan, "Neural Mechanisms of Selective Visual Attention," *Annual Review of Neuroscience*, vol. 18, no. 1, pp. 193 – 222, 1995.

[15] A. Torralba, A. Oliva, M. S. Castelhano, and J. M. Henderson, "Contextual guidance of eye movements and attention in real-world scenes: the role of global features in object search." *Psychological review*, vol. 113, no. 4, pp. 766 – 786, Oct. 2006.

[16] O. Stasse, Y. Kuniyoshi, and G. Cheng, "Development of a biologically inspired real-time visual attention system," in *Proceedings of the First IEEE International Workshop on Biologically Motivated Computer Vision*, ser. BMVC '00. London, UK: Springer-Verlag, 2000, pp. 150 – 159.

[17] S. Frintrop, M. Klodt, and E. Rome, "A real-time visual attention system using integral images," *Science*, vol. 11, no. 1, pp. 191 – 193, 2007.

[18] N. Ouerhani, H. Hügli, P.-Y. Burgi, and P.-F. Ruedi, "A real time implementation of the saliency-based model of visual attention on a simd architecture," in *Proceedings of the 24th DAGM Symposium on Pattern Recognition*. London, UK, UK: Springer-Verlag, 2002, pp. 282 – 289.

[19] P. Longhurst, K. Debattista, and A. Chalmers, "A gpu based saliency map for high-fidelity selective rendering," in *Proceedings of the 4th international conference on Computer graphics, virtual reality, visualisation and interaction in Africa*, ser. AFRIGRAPH '06. New York, NY, USA: ACM, 2006, pp. 21 – 29.

[20] B. Han and B. Zhou, "High speed visual saliency computation on gpu." in *ICIP (1)*. IEEE, 2007, pp. 361–364.

[21] S. Bae, Y. C. P. Cho, S. Park, K. M. Irick, Y. Jin, and V. Narayanan, "An fpga implementation of information theoretic visual-saliency system and its optimization," in *Proceedings of the 2011 IEEE 19th Annual International Symposium on Field-Programmable Custom Computing Machines*, ser. FCCM '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 41 – 48.

[22] B. Kim, H. Okuno, T. Yagi, and M. Lee, "Implementation of visual attention system using artificial retina chip and bottom-up saliency map model." in *ICONIP (3)*, ser. Lecture Notes in Computer Science, B.-L.

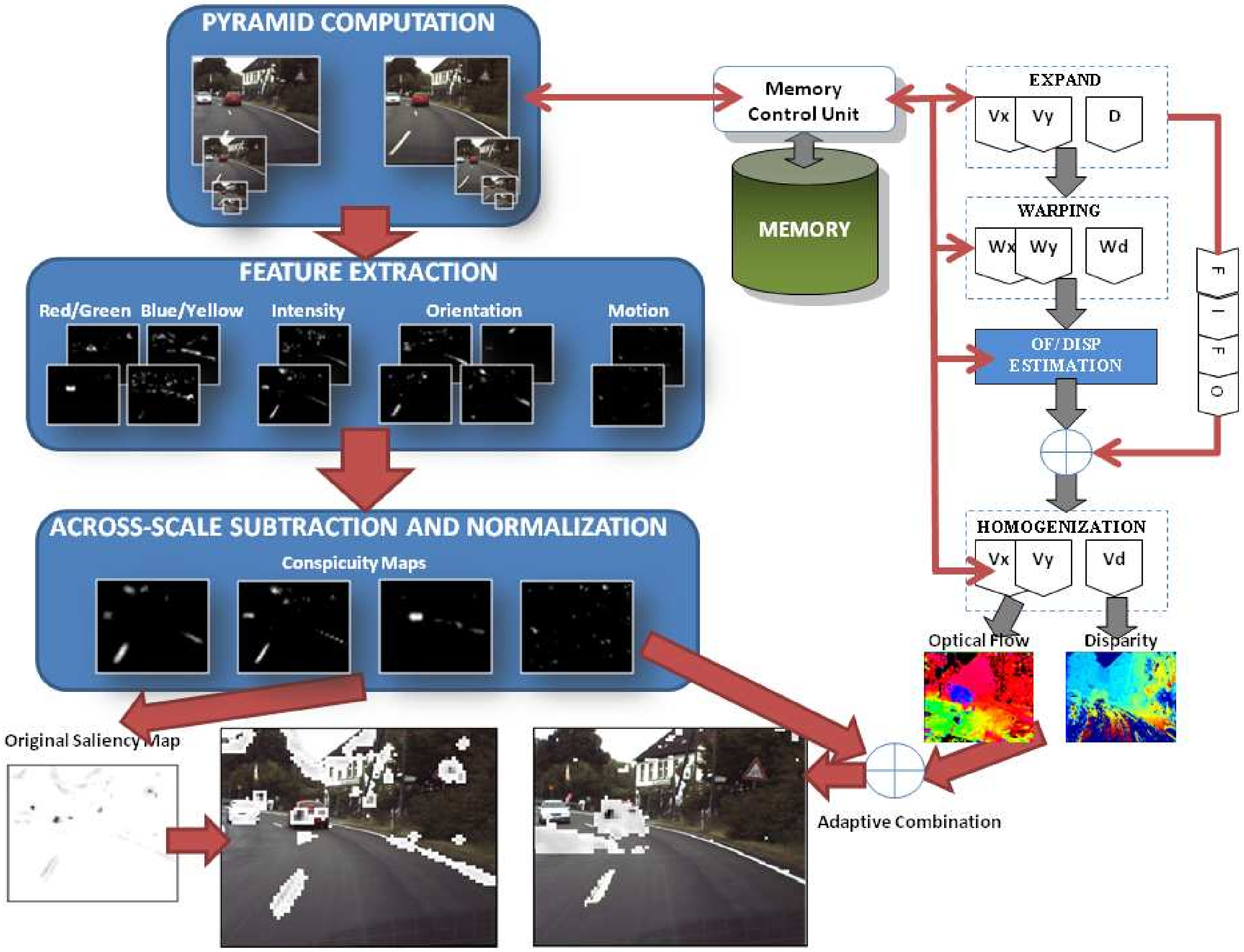


Fig. 6. Final architecture of the visual attention system. In this case, the saliency computation includes the motion and we also add the estimates of the multiscale computation of optical flow and disparity. Those cues conform the top-down modulation stream that combined with the saliency improves the detection of the objects of interest (in this driving scenario, the cars driving on the road).

Lu, L. Zhang, and J. T. Kwok, Eds., vol. 7064. Springer, 2011, pp. 416 – 423.

[23] D. Parkhurst, K. Law, and E. Niebur, “Modeling the role of salience in the allocation of overt visual attention.” *Vision Res*, vol. 42, no. 1, pp. 107 – 123, Jan. 2002.

[24] G. Underwood, T. Foulsham, E. Van Loon, and L. Humphreys, “Eye movements during scene inspection: A test of the saliency map hypothesis,” *European Journal of Cognitive Psychology*, 2006.

[25] *Attentional Landmark Selection for Visual SLAM*, 2006.

[26] A. J. Maeder, J. Diederich, and E. Niebur, “Limiting human perception for image sequences,” B. E. Rogowitz and J. P. Allebach, Eds., vol. 2657, no. 1. SPIE, 1996, pp. 330 – 337.

[27] B.-W. Hong and M. Brady, “A Topographic Representation for Mammogram Segmentation,” 2003, pp. 730 – 737.

[28] P. Santana, M. Guedes, L. Correia, and J. Barata, “A saliency-based solution for robust off-road obstacle detection,” in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, 2010, pp. 3096 – 3101.

[29] L. Itti and C. Koch, “A saliency-based search mechanism for overt and covert shifts of visual attention,” *Vision Research*, vol. 40, no. 10-12, pp. 1489–1506, May 2000.

[30] P. J. Burt, Edward, and E. H. Adelson, “The laplacian pyramid as a compact image code,” *IEEE Transactions on Communications*, vol. 31, pp. 532 – 540, 1983.

[31] J. M. Wolfe and T. S. Horowitz, “Opinion: What attributes guide the deployment of visual attention and how do they do it?” *Nature Reviews Neuroscience*, vol. 5, no. 6, pp. 495 – 501, 2004.

[32] D. Walther and C. Koch, “Modeling attention to salient proto-objects,” *Neural Networks*, vol. 19, no. 9, pp. 1395 – 1407, Nov. 2006.

[33] S. P. Sabatini, G. Gastaldi, F. Solari, K. Pauwels, M. M. V. Hulle, J. Diaz, E. Ros, N. Pugeault, and N. Krger, “A compact harmonic code for early vision based on anisotropic frequency channels,” *Computer Vision and Image Understanding*, vol. 114, no. 6, pp. 681 – 699, 2010.

[34] L. Itti and C. Koch, “Feature combination strategies for saliency-based visual attention systems,” *Journal of Electronic Imaging*, vol. 10, no. 1, pp. 161–169, Jan 2001.

[35] M. W. Cannon and S. C. Fullenkamp, “Spatial interactions in apparent contrast: inhibitory effects among grating patterns of different spatial frequencies, spatial positions and orientations.” *Vision research*, vol. 31, no. 11, pp. 1985 – 1998, 1991.

[36] C. D. Gilbert and T. N. Wiesel, “Columnar specificity of intrinsic horizontal and corticocortical connections in cat visual cortex,” *The Journal of Neuroscience*, vol. 9, no. 7, pp. 2432 – 2442, Jul. 1989.

[37] B. Zenger and D. Sagi, “Isolating excitatory and inhibitory nonlinear spatial interactions involved in contrast detection.” *Vision Res*, vol. 36, no. 16, pp. 2497 – 2513, Aug. 1996.

[38] L. Itti, “Models of bottom-up and top-down visual attention,” Ph.D. dissertation, Jan 2000.

[39] C. D. Salzman and W. T. Newsome, “Neural mechanisms for forming a perceptual decision.” *Science*, vol. 264, no. 5156, pp. 231 – 237, 1994.

[40] B. D. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” pp. 674 – 679, 1981.

[41] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, “Performance of optical

flow techniques," *Int Journal of Computer Vision*, vol. 12, pp. 43 – 77, 1994.

[42] J. Diaz, E. Ros, R. Agis, and J. Bernier, "Superpipelined high-performance optical-flow computation architecture," *Computer Vision and Image Understanding*, vol. 112, no. 3, pp. 262 – 273, 2008.

[43] J. Bergen, P. Anandan, K. Hanna, and R. Hingorani, "Hierarchical model-based motion estimation," in *Computer Vision ECCV'92*, ser. Lecture Notes in Computer Science, G. Sandini, Ed., vol. 588, 1992, pp. 237 – 252.

[44] M. Tomasi, F. Barranco, M. Vanegas, J. Diaz, and E. Ros, "Fine grain pipeline architecture for high performance phase-based optical flow computation," *Journal of Systems Architecture*, vol. 56, no. 11, pp. 577 – 587, 2010.

[45] M. Tomasi, M. Vanegas, F. Barranco, J. Diaz, and E. Ros, "High-Performance Optical-Flow architecture based on a multiscale, Multi-Orientation Phase-Based model," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 20, no. 12, pp. 1797 – 1807, 2010.

[46] ——, "A novel architecture for a massively parallel low level vision processing engine on chip," in *Industrial Electronics (ISIE), 2010 IEEE International Symposium on*, 2010, pp. 3033 – 3039.

[47] F. Barranco, M. Tomasi, J. Diaz, M. Vanegas, and E. Ros, "Parallel architecture for hierarchical optical flow estimation based on fpga," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. PP, no. 99, pp. 1 – 10, 2011.

[48] F. Barranco, J. Diaz, A. Gibaldi, S. P. Sabatini, and E. Ros, "Vector disparity sensor with vergence control for active vision systems," *Sensors*, vol. 12, no. 2, pp. 1771 – 1799, 2012.

[49] J. Diaz, E. Ros, R. Carrillo, and A. Prieto, "Real-time system for high-image resolution disparity estimation," *Image Processing, IEEE Transactions on*, vol. 16, no. 1, pp. 280 – 285, 2007.

[50] M. Tomasi, M. Vanegas, F. Barranco, J. Diaz, and E. Ros, "Real-time architecture for a robust multi-scale stereo engine on fpga," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. PP, 2011.

[51] R. Rosenholtz, "Search asymmetries? what search asymmetries?" *Attention, Perception, & Psychophysics*, vol. 63, pp. 476 – 489, 2001, 10.3758/BF03194414.

[52] M. Dick, S. Ullman, and D. Sagi, "Parallel and serial processes in motion detection." *Science*, vol. 237, no. 4813, pp. 400 – 402, 1987.

[53] P. McLeod, J. Driver, and C. J, "Visual search for conjunctions of movement and form is parallel," *Nature*, vol. 332, pp. 154 – 155, 1988.

[54] L. Itti, N. Dhavale, and F. Pighin, "Realistic avatar eye and head animation using a neurobiological model of visual attention," in *Proc. SPIE 48th Annual International Symposium on Optical Science and Technology*, B. Bosacchi, D. B. Fogel, and J. C. Bezdek, Eds., vol. 5200. Bellingham, WA: SPIE Press, Aug 2003, pp. 64–78.

[55] Sevensols, "Seven solutions." [Online]. Available: http://www.sevensols.com/

[56] M. Vanegas, M. Tomasi, J. Diaz, and E. Ros, "Multi-port abstraction layer for FPGA intensive memory exploitation applications," *Journal of Systems Architecture*, vol. 56, no. 9, pp. 442 – 451, 2010.

[57] E. Ortigosa, A. Canas, E. Ros, P. Ortigosa, S. Mota, and J. Diaz, "Hardware description of multi-layer perceptrons with different abstraction levels," *Microprocessors and Microsystems*, vol. 30, no. 7, pp. 435 – 444, 2006.

[58] I. Xilinx, "FPGA and CPLD solutions from xilinx, inc." 2010. [Online]. Available: http://www.xilinx.com/

[59] E. P. Simoncelli, "Design of multi-dimensional derivative filters," in *Image Processing, 1994. Proceedings. ICIP-94., IEEE International Conference*, vol. 1, 1994, pp. 790 – 794.

[60] L. Itti, "ilab image databases." [Online]. Available: http://ilab.usc.edu/imgdbs/

Francisco Barranco received his BS in Computer Science from the University of Granada in 2007 and his MSc in Computer and Network Engineering in 2008. He works at the Department of Architecture and Computer Technology at the same university. His main research interests deal with image processing architectures and embedded systems based on reconfigurable devices, real-time machine vision, general purpose graphical programming devices, biologically processing schemes, spiking neurons. He is currently participating in an EU project related with adaptive learning mechanisms and conventional control.

Javier Diaz received his MS in Electronics Engineering in 2002 and a Ph.D. in Electronics in 2006 both from the University of Granada. Currently, he is assistant professor at the Department of Computer Architecture and Technology at the same university. His main research interests are cognitive vision systems, high performance image processing architectures and embedded systems based on reconfigurable devices. He is also interested in spiking neurons, biomedical devices and robotics.

Begoa Pino received her Ph.D. degree in Computer Science from the University of Granada in 1999. She is an Associate Professor at the Department of Architecture and Computer Technology at the same University. Her main research interest currently lies on implementation of systems for visual rehabilitation, bio-inspired processing systems, and reconfigurable hardware codesign for computer vision on-chip. She has participated in different EU projects in the fields of visual rehabilitation, bio-inspired processing schemes based on spiking neural networks and real-time computer vision.

Eduardo Ros received the Ph.D. degree in 1997 from the University of Granada. He is currently Associate Professor at the Department of Computer Architecture and Technology at the same University. He is currently the responsible researcher at the University of Granada of two European projects related with bio-inspired processing schemes and real-time image processing. His research interests include hardware implementation of digital circuits for real time processing in embedded systems and high performance computer vision.

5 Complete list of publications

In this section, we present a complete list with all the publications related with this Ph.D. dissertation, including the references of the presented proposals.

List of publications from international journals with impact factor

- F. Barranco, J.Díaz, E. Ros, B. Pino, “Visual system based on artificial retina for motion detection”, IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernetics, vol. 39, no. 3, pp. 752-762, 2009.

- M. Tomasi, M. Vanegas, F. Barranco, J. Díaz and E. Ros, “Massive parallel-hardware architecture for multiscale stereo, optical flow and image-structure computation”, IEEE Trans. Circuits and Systems Video Technology, vol. 20, no. 2, pp. 282-294, 2012.

- M. Tomasi, M. Vanegas, F. Barranco, J. Díaz and E. Ros, “High-performance optical flow architecture based on a multi-scale, multi-orientation phase-based model”, IEEE Trans. on Circuits and Systems for Video Technology, vol. 20, no. 12, pp. 1797- 1807, 2010.

- M. Tomasi, F. Barranco, M. Vanegas, J. Diaz, and E. Ros, “Fine grain pipeline architecture for high performance phase - based optical flow computation”, Journal of System Architecture, vol. 56, no. 11, pp. 577 - 587, 2010.

- M. Tomasi, M. Vanegas, F. Barranco, J. Diaz, and E. Ros, “Real-time architecture for a robust multiscale stereo engine”, IEEE Trans. on VLSI (In Press).

- F. Barranco, J. Diaz, A. Gibaldi, S.P. Sabatini, E. Ros, “Vector disparity sensor with vergence control for active vision systems”, Sensors, vol. 12, no. 2, pp. 1771-1799, 2012.

- F. Barranco, M. Tomasi, J. Díaz, M. Vanegas, E. Ros, “Parallel architecture for hierarchical optical flow estimation based on FPGA”, IEEE Trans. on VLSI, vol. 20, no. 6, pp. 1058-1067, 2012.

- F. Barranco, M. Tomasi, J. Díaz, M. Vanegas, E. Ros, “Pipelined Architecture for Real-Time Low-Cost Extraction of Visual Primitives based on FPGAs”, submitted to Digital Signal Processing (in peer review).

- F. Barranco, M. Tomasi, J. Díaz, E. Ros, "Hierarchical Architecture for Motion and Depth Estimations based on Color Cues", submitted to J. of Real-Time Image Processing (in peer review).
- F. Barranco, J. Díaz, B. Pino, E. Ros, "A multi-resolution approach for massively-parallel hardware-friendly optical flow estimation", minor-revision on J. of Visual Comm. And Image Representation (in peer review).
- M. Tomasi, F. Barranco, J. Ralli, J.M. Gomez-Lopez, J. Díaz, E. Ros, "Hardware-friendly regularization for coarse-to-fine image registration with median filtering", submitted to J. of Real-Time Image Processing (in peer review).

List of publications from international conferences

- M. Tomasi, M. Vanegas, F. Barranco, J. Díaz, E. Ros, "A novel architecture for a massively parallel low level vision processing engine on chip", ISIE2010. IEEE International Symposium on Industrial Electronics, Bari (Italy) (4-7 July 2010).
- F. Barranco, M. Tomasi, J. Diaz, E. Ros, "Hierarchical Optical Flow Estimation Architecture using Color Cues", 7th International Symposium on Applied Reconfigurable Computing (ARC11), 23-25 March 2011. Belfast (Ireland), LNCS 201, vol. 6578/2011, 269-274.

List of publications from national conferences

- S. Granados, F. Barranco, J. Díaz, S. Mota, E. Ros, "Condensación de primitivas visuales de bajo nivel para aplicaciones atencionales", Sept. 2010, Congreso Español de Informática (CEDI 2010), X JCRA. Valencia (Spain), Pp.199-206.
- F. Barranco, M. Tomasi, Vanegas, S. Granados, J. Díaz, "Arquitectura para la extracción de primitivas visuales de bajo nivel en un mismo chip en tiempo real", Sept. 2010, Congreso Español de Informática (CEDI 2010), X JCRA. Valencia (Spain), Pp.207-214.
- M. Tomasi, M. Vanegas, F. Barranco, J. Díaz, E. Ros, "Arquitectura multiescala de cálculo de flujo óptico basado en la fase", in: IX Jornadas de Computación Reconfigurable y Aplicaciones. JCRA2009, 2009, pp. 295 - 304.
- F. Barranco, M. Tomasi, Vanegas, S. Granados, J. Díaz, "Entorno software para visualización y configuración de procesamiento de imá-

genes en tiempo real con plataformas reconfigurables", Sept. 2009, IX JCRA. Alcalá de Henares (Spain), pp.327-336.

Appendix A

Open-RT Vision platform

Introduction

Open-RT Vision is an open source tool developed using .Net Framework with Visual Studio [57]. The application works as an interface between a camera stereo rig and a co-processing board connected to the computer using the PCI/PCIe bus. Beyond a simple interface, this software interface is used also to pre and post-process the data that these two components are transmitting. A detailed description can be found at [58] but a brief is presented in this section. An example of the complete system running with a road sequence is displayed in Fig. 1.

Software platform

Machine vision applications help us to understand the dynamic environment where we live. There are a lot of target applications such as robotics, industrial inspection, monitoring and surveillance systems, medicine, driving assistance systems, ... This GUI was developed in the framework of the Drivsco Project, in which our work consists in the development of an embedded system integrating the algorithms for the extraction of several low-level visual primitives: energy, orientation, phase and also, disparity and optical flow estimation. However, the number and kind of the extracted modalities may be increased.

The most important property of this software platform is that it must achieve real-time performance (i.e. more than 20 frames per second). In addition to this, a simple and efficient interface is quite important for the development of complex systems, specially in our case in artificial vision applications.

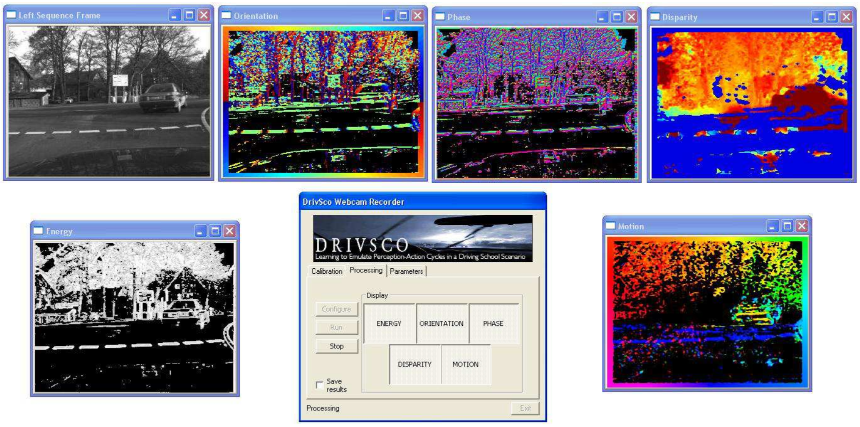
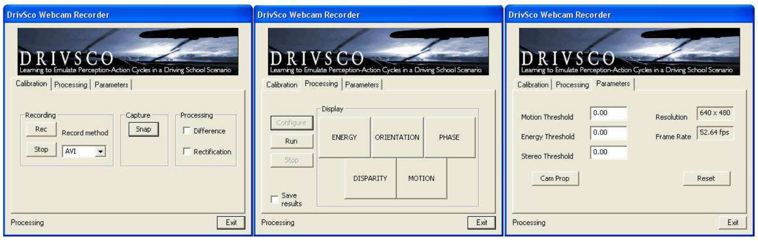


Figure 1: Example of the complete system working in real-time computing a road scene.

The software platform provides the possibility of configuring the system and displaying the results connecting a standard computer with a co-processing board (in our case, we use XircaV4 and RC2000). As we mentioned in the introduction, the work also involves the development of interface libraries to establish the communication with the co-processing boards, making the software interface flexible and generic: developing a new specific interface library we may use any co-processing board.

In the following list, we briefly explain the functions that the Open-RT Vision environment performs (see also Fig. 2):

- Image difference, useful for calibration and disparity debugging.
- Undistortion and Rectification using LUTs for the disparity computation. It also can be done in hardware.
- Debayerization (required for the configuration with the scientific cameras).
- Snapshots.
- Input recording and storage in different standard video and image formats.
- Result post-processing and storage.
- Configuration of the camera settings.
- Configuration for the different processing parameters (such as input/output resolution, recording file formats, frame rate, colormaps ...)



Record method: AVI

Motion Threshold: 0.00

Energy Threshold: 0.00

Stereo Threshold: 0.00

Resolution: 640 x 480

Frame Rate: 52.64 fps

Figure 2: Setting dialogs of the Open-RT Vision platform showing its functionality.

- Display of the different results

As seen in the list, software obtains and post-processes the results, to be displayed and stored. Just the visualization in real-time of the huge amount of data that is generated from the board is a big challenge. Hence, we use different libraries and software packages for the optimization of the data management. Moreover, the first stage of this optimization consists in the code optimization, minimizing the read and write operations to disk and the cache-memory faults, unrolling loops and using threads for the recording computation.

- OpenCV (Open Source Computer Vision [182]) is a software library for vision applications and it is optimized for the data processing in this field. We used it for the display of the results and the optimization of the filter post- and pre-processing. It also improves the throughput in the results storage on the hard disk such as image or video files.

- The IPP library package (Intel Integrated Performance Primitives [183]) provides low-level routines for high performance applications. It improves the operations on well-aligned data arrays. The library provides also optimized routines for matrix arithmetic operations such as matrix divisions, multiplications, convolutions ... All the routines are optimized for their use under Intel architectures.

- OpenMP (Open Multi-Processing [184]) provides support for multiplatform parallel programming with shared memory. We use it for complex loops taking advantage of the multicore architectures. In this case, the scheduling (static or dynamic) is an important key for maximizing the performances.

The optimization is performed using the different packages but, the main contribution is obtained by using OpenMP. An example of the performances obtained by the implementation with the different packages in Fig. 3 and Table A.1.

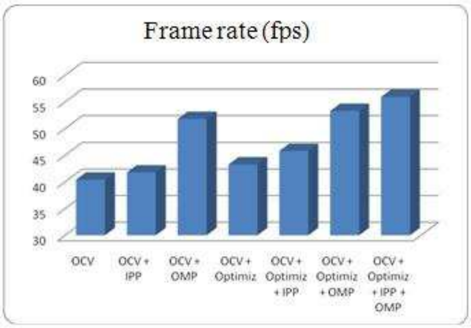


Figure 3: Graphic shows the frame rate obtained for the computation of a 320x240 image, extracting all the visual primitives (energy, orientation, phase, disparity and optical flow).

Processing	Frame Rate (fps for 512x512)
Local Features	40.77
Disparity	35.04
Optical Flow	30.65
Complete Processing	23.21

Table A.1: Obtained frame rate for the extraction of all the image features with a 512x512 image.

Finally, the inception of this work is the result of the need to join the different tools used for the development of image processing algorithms and achieve high-performance requirements for the post-processing and the visualization of the obtained results computed by the co-processing board. With this application we integrate the different tools in a platform which also integrates more functionalities. More requirements for our platform are a user-friendly interface, usability, scalability or a platform optimized for image processing.

In the project work homepage (see [58]) can be found some videos with real-time processing. At the “Downloads” section it can be found tutorials, a video-tutorial, examples and the complete documented code. The software documentation can be generated using *doxygen* [185].

Architecture

The complete system includes two cameras (high-performance scientific or webcams), connected with a computer (via the Camera Link interface or the USB respectively). In the case of the scientific cameras, we need a specific image acquisition device, a frame grabber. The application runs in the computer, acquiring the images from the cameras and sending this processed information to the co-processing platform using the PCIe (XircaV4) or the PCI (RC2000) interfaces. Once, the hardware computation is completed, the application reads the results from the board memory and post-processes the results for an appropriate visualization or storage.

- A stereo camera rig. We have two different setups: A pair of USB webcams or a pair of high-performance scientific cameras with Camera Link connection. We study also other interface connections such as FireWire or Ethernet.
- With the scientific cameras, we use a frame grabber.
- The presented software is running in a computer with PCI/PCIe interfaces for the board connections.
- The co-processing board. We developed the libraries for:
  - 1. The XircaV4 platform, developed by Seven Solutions [92]. It includes four SSRAM (ZBT) memory banks of 2 Mwords of 36 bits, 2 512 MB DDR memory modules, and a 32 MB flash memory module. Furthermore, it has a 10/100/1000Mbits Ethernet connection, a JTAG configuration interface and the PCIe 1x interface. The FPGA is a Virtex4 XCVFX100 - 10FFG1152.
  - 2. The RC2000 platform, developed by Celoxica (now Mentor Graphics [186]). It includes 6 ZBT memory banks of 1 Mword of 36 bits, with a 16 MB flash memory. In this case, the connection is PCI. Finally, the FPGA is a Xilinx Virtex-II XCV6000-6.

An scheme of the architecture platform is presented in Fig. 4

The hardware co-design allows us to take advantage from the hardware processing such as the high parallelism, the low size and the low power consumption. But it also provides us the possibility of implementing complex algorithms easily and rapidly or to use the software as an interface with the visualization devices, or even, to validate and verify the hardware computation. With the presented tool we also developed two interface libraries for connecting the software application and the co-processing device, which are also documented.

Conclusions and future work

The system fulfills the most important requirements of the work involving co-processing boards:

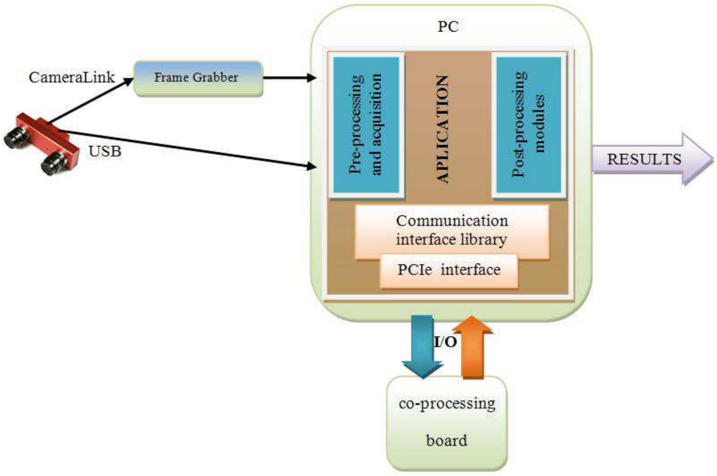


Figure 4: Scheme of the complete system, including the image acquisition, a generic co-processing board, the application including the communication libraries via the PCIe interface, and the pre- and post- processing modules.

- Capacity for the final verification/debugging of final results and hence, the respective algorithms.
- Friendly interfaces for the system configuration.
- Display of a huge volume of data in real-time.

Now Open-RT Vision has been released as Open Source facilitated by the OSL “Oficina de Software Libre” through the advice of J.J. Merelo of the University of Granada. This means that any other development effort can be done by any other member or the research community. This software was released with a GNU Lesser GPL.

Currently, we are improving the project working on new functionalities for the software platform such as the automatic camera calibration, the connection with new interfaces (FireWire and Ethernet) or the improvement of the communication by the use of sockets.

Bibliography

[1] C. Weems, “Architectural requirements of image understanding with respect to parallel processing,” *Proceedings of the IEEE*, vol. 79, no. 4, pp. 537 – 547, 1991.

[2] A. P. Tirumalai, B. G. Schunck, and R. C. Jain, “Evidential reasoning for building environment maps,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 25, no. 1, pp. 10–20, 1995.

[3] J. K. Tsotsos, L. Itti, and G. Rees, “A brief and selective history of attention,” in *Neurobiology of Attention* (L. Itti, G. Rees, and J. K. Tsotsos, eds.), pp. xxiii – xxxii, San Diego, CA: Elsevier, Jan 2005.

[4] J. Duncan, “Selective attention and the organization of visual information.,” *Journal of Experimental Psychology. General*, vol. 113, no. 4, pp. 501 – 517, 1984.

[5] P. R. Roelfsema, V. A. F. Lamme, and H. Spekreijse, “Object-based attention in the primary visual cortex of the macaque monkey,” *Nature*, vol. 395, no. 6700, pp. 376 – 381, 1998.

[6] L. Itti, C. Koch, and E. Niebur, “A model of saliency-based visual attention for rapid scene analysis,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, pp. 1254 – 1259, Nov. 1998.

[7] R.-J. Lin, W.-S. Lin, and Y.-W. Huang, “Computational visual attention model capable of exploring similarity,” in *Computational Intelligence for Multimedia, Signal and Vision Processing (CIMSIVP), 2011 IEEE Symposium on*, pp. 7 –11, april 2011.

[8] R. Peters and L. Itti, “Beyond bottom-up: Incorporating task-dependent influences into a computational model of spatial attention,” in *Computer Vision and Pattern Recognition, 2007. CVPR ’07. IEEE Conference on*, pp. 1 –8, 2007.

[9] O. Le Meur, P. Le Callet, D. Barba, and D. Thoreau, “A coherent computational approach to model bottom-up visual attention,” *Pattern*

Analysis and Machine Intelligence, IEEE Transactions on, vol. 28, pp. 802 –817, may 2006.

[10] E. D. Dickmanns, *Dynamic Vision for Perception and Control of Motion.* Springer, 2007.

[11] V. Cantoni, S. Levialdi, and V. Roberto, *Artificial vision: image description, recognition and communication.* Signal processing and its applications, Academic, 1997.

[12] J. Henderson, "Human gaze control during real-world scene perception," *Trends in Cognitive Sciences*, vol. 7, no. 11, pp. 498 – 504, 2003.

[13] E. R. Kandel, J. H. Schwartz, and T. M. Jessell, *Principles of Neural Science.* McGraw-Hill Medical, 5th ed., 2013.

[14] F. Du, J. M. Brady, and D. W. Murray, "Gaze control for a two-eyed robot head," in *Proc 2nd British Machine Vision Conference, Glasgow*, pp. 193 – 201, Springer-Verlag, 1991.

[15] D. J. Coombs and C. M. Brown, "Cooperative gaze holding in binocular vision," *IEEE Control Systems Magazine*, vol. 11, no. 4, pp. 24 – 33, 1991.

[16] L. Itti and C. Koch, "A saliency-based search mechanism for overt and covert shifts of visual attention," *Vision Research*, vol. 40, pp. 1489–1506, May 2000.

[17] A. Rothenstein, J. K. Tsotsos, "Computational models of visual attention," *Scholarpedia*, vol. 6, no. 1, p. 6201, 2011.

[18] A. Borji and L. Itti, "State-of-the-art in visual attention modeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. PrePrints, 2012.

[19] L. Itti and C. Koch, "Feature combination strategies for saliency-based visual attention systems," *Journal of Electronic Imaging*, vol. 10, pp. 161–169, Jan 2001.

[20] W. James, *The Principles of Psychology, Vol. 1.* Dover Publications, 1950.

[21] A. M. Treisman and G. Gelade, "A feature-integration theory of attention," *Cognitive Psychology*, vol. 12, pp. 97 – 136, Jan. 1980.

[22] J. Braun and D. Sagi, "Vision outside the focus of attention.," *Perception And Psychophysics*, vol. 48, no. 1, pp. 45 – 58, 1990.

[23] J. R. Bergen and B. Julesz, “Parallel versus serial processing in rapid pattern discrimination,” *Nature*, vol. 303, pp. 696 – 698, 1983.

[24] A. Treisman, “Features and objects: the fourteenth Bartlett memorial lecture.,” *The Quarterly journal of experimental psychology. A, Human experimental psychology*, vol. 40, no. 2, pp. 201 – 237, 1988.

[25] D. Walther, *Interactions of Visual Attention and Object Recognition: Computational Modeling, Algorithms, and Psychophysics*. PhD thesis, 2010.

[26] E. D. Dickmanns and T. Christians, “Relative 3d-state estimation for autonomous visual guidance of road vehicles,” *Robotics and Autonomous Systems*, vol. 7, no. 2 - 3, pp. 113 – 123, 1991.

[27] E. Dickmanns, B. Mysliwetz, and T. Christians, “An integrated spatio-temporal approach to automatic visual guidance of autonomous vehicles,” *SMC*, vol. 20, no. 6, pp. 1273 – 1284, 1990.

[28] J. G. Samarawickrama and S. P. Sabatini, “Version and vergence control of a stereo camera head by fitting the movement into the hering’s law,” in *Proceedings of the Fourth Canadian Conference on Computer and Robot Vision*, pp. 363 – 370, 2007.

[29] A. Koene, J. Moren, V. Trifa, and G. Cheng, “Gaze shift reflex in a humanoid active vision system,” in *Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007), Applied Computer Science Group*, 2007.

[30] J. P. Brooker, P. M. Sharkey, J. P. Wann, and A. M. Plooy, “A helmet mounted display system with active gaze control for visual telepresence,” *Mechatronics*, vol. 9, no. 7, pp. 703 – 716, 1999.

[31] S. Viollet and N. Franceschini, “A high speed gaze control system based on the vestibulo-ocular reflex,” *Robotics and Autonomous Systems*, vol. 50, no. 4, pp. 147 – 161, 2005.

[32] M. Pellkofer and E. Dickmanns, “Ems-vision: gaze control in autonomous vehicles,” in *Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE*, pp. 296 – 301, 2000.

[33] A. Bernardino and J. Santos-Victor, “A binocular stereo algorithm for log-polar foveated systems.,” in *Biologically Motivated Computer Vision*, vol. 2525 of *Lecture Notes in Computer Science*, pp. 127 – 136, Springer, 2002.

[34] B. Scassellati, “A binocular, foveated active vision system,” tech. rep., Technical report, MIT Artificial Intelligence Lab. In submission, 1998.

[35] L. Itti and C. Koch, “Computational modelling of visual attention,” *Nature Review Neuroscience*, vol. 2, pp. 194 – 203, Mar. 2001.

[36] E. Niebur and C. Koch, “A model for the neuronal implementation of selective visual attention based on temporal correlation among neurons,” *Journal of Computational Neuroscience*, vol. 1, pp. 141 – 158, June 1994.

[37] A. Torralba, A. Oliva, M. S. Castelhano, and J. M. Henderson, “Contextual guidance of eye movements and attention in real-world scenes: the role of global features in object search.,” *Psychological review*, vol. 113, pp. 766 – 786, Oct. 2006.

[38] H. Yamamoto, M. D. Levine, and Y. Yeshurun, “An active foveated vision system: attentional mechanisms and scan path convergence measures,” *Computer Vision and Image Understanding*, vol. 63, pp. 50 – 65, Jan. 1996.

[39] B. Rasolzadeh, M. Björkman, K. Huebner, and D. Kragic, “An active vision system for detecting, fixating and manipulating objects in the real world,” *Int. J. Rob. Res.*, vol. 29, no. 2-3, pp. 133 – 154, 2010.

[40] M. Tomasi, M. Vanegas, F. Barranco, J. Diaz, and E. Ros, “High-Performance Optical-Flow architecture based on a multiscale, Multi-Orientation Phase-Based model,” *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 20, no. 12, pp. 1797 – 1807, 2010.

[41] M. Tomasi, M. Vanegas, F. Barranco, J. Diaz, and E. Ros, “A novel architecture for a massively parallel low level vision processing engine on chip,” in *Industrial Electronics (ISIE), 2010 IEEE International Symposium on*, pp. 3033 – 3039, 2010.

[42] M. Tomasi, F. Barranco, M. Vanegas, J. Diaz, and E. Ros, “Fine grain pipeline architecture for high performance phase-based optical flow computation,” *Journal of Systems Architecture*, vol. 56, no. 11, pp. 577 – 587, 2010.

[43] M. Tomasi, M. Vanegas, F. Barranco, J. Diaz, and E. Ros, “Real-time architecture for a robust multi-scale stereo engine on fpga,” *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. PP, 2011.

[44] M. Vanegas, L. Rubio, M. Tomasi, J. Diaz, and E. Ros, “On-chip ego-motion estimation based on optical flow,” in *Reconfigurable Computing: Architectures, Tools and Applications* (A. Koch, R. Krishnamurthy, J. McAllister, R. Woods, and T. El-Ghazawi, eds.), vol. 6578

of *Lecture Notes in Computer Science*, pp. 206 – 217, Springer Berlin / Heidelberg, 2011.

[45] Drivsco, "Drivsco project: Learning to emulate perceptio-action cycles in a driving school scenario.," 2012.

[46] Matlab, "Matlab." http://www.mathworks.com/.

[47] M. Anguita, J. Diaz, E. Ros, and F. J. Fernandez-Baldomero, "Optimization strategies for High-Performance computing of Optical-Flow in General-Purpose processors," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 19, no. 10, pp. 1475 – 1488, 2009.

[48] K. Pauwels and M. M. V. Hulle, "Realtime phase-based optical flow on the GPU," in *Computer Vision and Pattern Recognition Workshops, CVPRW '08. IEEE Computer Society Conference on*, pp. 1 – 8, 2008.

[49] K. Pauwels, M. Tomasi, J. Diaz Alonso, E. Ros, and M. Van Hulle, "A comparison of fpga and gpu for real-time phase-based optical flow, stereo, and local image features," *Computers, IEEE Transactions on*, vol. PP, 2011.

[50] C. Banz, H. Blume, and P. Pirsch, "Real-time semi-global matching disparity estimation on the gpu," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, vol. PP, pp. 514 –521, nov 2011.

[51] J. Chase, B. Nelson, J. Bodily, Z. Wei, and D.-J. Lee, "Real-time optical flow calculations on fpga and gpu architectures: A comparison study," in *Field-Programmable Custom Computing Machines, 2008. FCCM '08. 16th International Symposium on*, pp. 173 –182, april 2008.

[52] J.-S. Kim, M. Hwangbo, and T. Kanade, "Parallel algorithms to a parallel hardware: Designing vision algorithms for a gpu," in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pp. 862 – 869, 2009.

[53] A. Zarandy, C. Rekeczky, and P. Foldesy, "Analysis of 2d operators on topographic and non-topographic processor architectures," in *Cellular Neural Networks and Their Applications, 2008. CNNA 2008. 11th International Workshop on*, pp. 57 – 62, 2008.

[54] I. Xilinx, "FPGA and CPLD solutions from xilinx, inc.," 2012.

[55] MentorGraphics, "Handel-c synthesis methodology." http://www.mentor.com/products/fpga/handel-c/.

[56] M. C. Vision, "Middlebury computer vision," 2012.

[57] Microsoft, "Visual studio." http://www.microsoft.com/visualstudio/en-us/products/2010-editions.

[58] O. rt Vision project, "Open rt-vision project." http://code.google.com/p/open-rtvision/, 2012.

[59] L. Itti, "ilab image database," 2012.

[60] B. Julesz, "Visual pattern discrimination," *IEEE Transactions on Information Theory*, vol. 8, no. 2, pp. 84 – 92, 1962.

[61] B. Julesz, "Textons, the elements of texture perception, and their interactions.," *Nature*, vol. 290, no. 5802, pp. 91 – 97, 1981.

[62] A. Linares-Barranco, G. Jimenez-Moreno, A. Civit-Ballcels, and B. Linares-Barranco, "On synthetic aer generation," in *Circuits and Systems, 2004. ISCAS '04. Proceedings of the 2004 International Symposium on*, vol. 5, pp. 784 – 787, 2004.

[63] T. Serrano-Gotarredona, A. Andreou, and B. Linares-Barranco, "Aer image filtering architecture for vision-processing systems," *Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on*, vol. 46, no. 9, pp. 1064 – 1071, 1999.

[64] A. Martinez, L. Reyneri, F. Pelayo, S. Romero, C. Morillas, and B. Pino, "Automatic generation of bio-inspired retina-like processing hardware," vol. 3512, pp. 527 – 533, 2005.

[65] C. A. Morillas, S. F. Romero, A. Martinez, F. J. Pelayo, E. Ros, and E. Fernandez, "A design framework to model retinas," *Biosystems*, vol. 87, no. 2-3, pp. 156 – 163, 2007.

[66] F. Pelayo, A. Martinez, S. Romero, C. Morillas, E. Ros, and E. Fernandez, "Cortical visual neuro-prosthesis for the blind: retina-like software/hardware preprocessor," in *Neural Engineering, 2003. Conference Proceedings. First International IEEE EMBS Conference on*, pp. 150 – 153, 2003.

[67] M. Kawaguchi, T. Jimbo, M. Umeno, and N. Ishii, "Multi-layered analog electronic circuits for motion detection using biomedical vision and brain model," *Int. J. Know.-Based Intell. Eng. Syst.*, vol. 8, no. 1, pp. 1 – 7, 2004.

[68] K. Boahen, "A retinomorphic chip with parallel pathways: Encoding increasing, on, decreasing, and off visual signals," *Analog Integr. Circuits Signal Process.*, vol. 30, pp. 121 – 135, January 2002.

[69] K. A. Zaghloul and K. Boahen, "Optic nerve signals in a neuromorphic chip II: Testing and results.," *IEEE Trans Biomed Eng*, vol. 51, pp. 667 – 675, Apr. 2004.

[70] K. A. Zaghloul and K. Boahen, "Optic nerve signals in a neuromorphic chip i: Outer and inner retina models," *IEEE Trans Biomed Eng*, vol. 51, no. 4, pp. 657 – 666, 2004.

[71] W. T. Newsome and E. B. Pare, "A selective impairment of motion perception following lesions of the middle temporal visual area (MT)," *J. Neurosci.*, vol. 8, no. 6, pp. 2201 – 2211, 1988.

[72] C. D. Salzman, C. M. Murasugi, K. H. Britten, and W. T. Newsome, "Microstimulation in visual area MT: effects on direction discrimination performance.," *J Neurosci*, vol. 12, no. 6, pp. 2331 – 2355, 1992.

[73] E. P. Simoncelli and D. J. Heeger, "A model of neuronal responses in visual area MT," *Vision Res.*, vol. 38, pp. 743 – 761, Mar. 1998.

[74] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, "Performance of optical flow techniques," *Int Journal of Computer Vision*, vol. 12, pp. 43 – 77, 1994.

[75] L. Perrinet, M. Samuelides, and S. J. Thorpe, "Coding static natural images using spiking event times: do neurons cooperate?," *IEEE Transactions on Neural Networks*, vol. 15, no. 5, pp. 1164 – 1175, 2004.

[76] A. Giachetti, M. Campani, and V. Torre, "The use of optical flow for the autonomous navigation," in *Proceedings of the third European conference on Computer vision*, vol. 1 of *ECCV '94*, pp. 146 – 151, Springer-Verlag New York, Inc., 1994.

[77] R. Marzotto, P. Zoratti, D. Bagni, A. Colombari, and V. Murino, "A real-time versatile roadway path extraction and tracking on an fpga platform," *Computer Vision and Image Understanding*, vol. 114, no. 11, pp. 1164 – 1179, 2010.

[78] S. Nedevschi, R. Danescu, D. Frentiu, T. Marita, F. Oniga, C. Pocol, R. Schmidt, and T. Graf, "High accuracy stereo vision system for far distance obstacle detection," in *Intelligent Vehicles Symposium, IEEE*, pp. 292 – 297, 2004.

[79] J. Diaz, E. Ros, R. Agis, and J. Bernier, "Superpipelined high-performance optical-flow computation architecture," *Computer Vision and Image Understanding*, vol. 112, no. 3, pp. 262 – 273, 2008.

[80] A. Wali and A. M. Alimi, “Event detection from video surveillance data based on optical flow histogram and high-level feature extraction,” in *Database and Expert Systems Application, 20th International Workshop on*, pp. 221 – 225, 2009.

[81] S. Rougeaux and Y. Kuniyoshi, “Velocity and disparity cues for robust Real-Time binocular tracking,” in *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, IEEE Computer Society, 1997.

[82] M.-K. Hu, “Visual pattern recognition by moment invariants,” *Information Theory, IRE Transactions on*, vol. 8, no. 2, pp. 179 – 187, 1962.

[83] P. C. Merrell and D. Lee, “Structure from motion using optical flow probability distributions,” in *Intelligent Computing: Theory and Applications III* (K. L. Priddy, ed.), vol. 5803, pp. 39 – 48, SPIE, 2005.

[84] B. D. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” pp. 674 – 679, 1981.

[85] S. Baker and I. Matthews, “Lucas-kanade 20 years on: A unifying framework: Part 1,” tech. rep., Robotics Institute, 2002.

[86] V. Mahalingam, K. Bhattacharya, N. Ranganathan, H. Chakravarthula, R. Murphy, and K. Pratt, “A VLSI architecture and algorithm for Lucas-Kanade-Based optical flow computation,” *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 18, no. 1, pp. 29 – 38, 2010.

[87] B. K. P. Horn and B. G. Schunck, “Determining optical flow,” *Artificial Intelligence*, vol. 17, pp. 185 – 203, 1981.

[88] H. Liu, T.-H. Hong, M. Herman, and R. Chellappa, “Accuracy vs. efficiency trade-offs in optical flow algorithms,” in *Computer Vision ECCV 96*, vol. 1065 of *Lecture Notes in Computer Science*, pp. 174 – 183, Springer Berlin, Heidelberg, 1996.

[89] S. S. Beauchemin and J. L. Barron, “The computation of optical flow,” *ACM Computing Surveys*, vol. 27, no. 3, pp. 433 – 466, 1995.

[90] J. Bergen, P. Anandan, K. Hanna, and R. Hingorani, “Hierarchical model-based motion estimation,” in *Computer Vision ECCV’92* (G. Sandini, ed.), vol. 588 of *Lecture Notes in Computer Science*, pp. 237 – 252, 1992.

[91] P. J. Burt, Edward, and E. H. Adelson, “The laplacian pyramid as a compact image code,” *IEEE Transactions on Communications*, vol. 31, pp. 532 – 540, 1983.

[92] Sevensols, “Seven solutions.”

[93] J. Bigun, G. Granlund, and J. Wiklund, “Multidimensional orientation estimation with applications to texture analysis and optical flow,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 13, no. 8, pp. 775 – 790, 1991.

[94] M. Tomasi, M. Vanegas, F. Barranco, J. Daz, and E. Ros, “Massive parallel-hardware architecture for multiscale stereo, optical flow and image-structure computation,” *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, no. 2, pp. 282 – 294, 2012.

[95] G. Botella, A. Garcia, M. Rodriguez-Alvarez, E. Ros, U. Meyer-Baese, and M. C. Molina, “Robust bioinspired architecture for optical-flow computation,” *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 18, pp. 616 – 629, 2010.

[96] M. Martineau, Z. Wei, D.-J. Lee, and M. Martineau, “A fast and accurate tensor-based optical flow algorithm implemented in fpga,” in *Applications of Computer Vision, 2007. WACV ’07. IEEE Workshop on*, 2007.

[97] I. Ernst and H. Hirschmüller, “Mutual information based semi-global stereo matching on the gpu,” in *Proceedings of the 4th International Symposium on Advances in Visual Computing*, pp. 228 – 239, Springer-Verlag, 2008.

[98] C. Georgoulas and I. Andreadis, “A real-time occlusion aware hardware structure for disparity map computation,” in *Image Analysis and Processing ICIAP 2009*, vol. 5716, pp. 721 – 730, Springer Berlin, 2009.

[99] H. Calderon, J. Ortiz, and J. Fontaine, “High parallel disparity map computing on FPGA,” in *Mechatronics and Embedded Systems and Applications (MESA), 2010 IEEE/ASME International Conference on*, pp. 307 – 312, 2010.

[100] D. Scharstein and R. Szeliski, “A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms,” *International Journal of Computer Vision*, vol. 47, no. 1, pp. 7 – 42, 2002.

[101] J. Banks and P. Corke, “Quantitative evaluation of matching methods and validity measures for stereo vision,” *The International Journal of Robotics Research*, vol. 20, no. 7, pp. 512 – 532, 2001.

[102] M. Vanegas, M. Tomasi, J. Diaz, and E. Ros, “Multi-port abstraction layer for FPGA intensive memory exploitation applications,” *Journal of Systems Architecture*, vol. 56, no. 9, pp. 442 – 451, 2010.

[103] P. Golland and A. M. Bruckstein, "Motion from color," *Computer Vision and Image Understanding*, vol. 68, no. 3, pp. 346–362, 1997.

[104] F. Barranco, M. Tomasi, J. Diaz, and E. Ros, "A multi-resolution approach for massively-parallel hardware-friendly optical flow estimation," *In Press, Journal of Visual Communication and Image Representation*, 2012.

[105] iCub Project, "The eu icub project: an open source cognitive humanoid robotic platform, http://www.icub.org/."

[106] A. Blake and A. Yuille, *Active vision.* Artificial intelligence, MIT Press, 1992.

[107] E. Cuevas, E. Jiménez, D. Navarro, and R. Rojas, *Intelligent active vision systems for robots.* Cuvillier, 2007.

[108] K. Daniilidis, C. Krauss, M. Hansen, and G. Sommer, "Real-time tracking of moving objects with an active camera," *Real-Time Imaging*, vol. 4, pp. 3 – 20, Feb. 1998.

[109] D. Murray and A. Basu, "Motion tracking with an active camera," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 16, no. 5, pp. 449 – 459, 1994.

[110] N. Papanikolopoulos, P. Khosla, and T. Kanade, "Visual tracking of a moving target by a camera mounted on a robot: a combination of control and vision," *Robotics and Automation, IEEE Transactions on*, vol. 9, no. 1, pp. 14 – 35, 1993.

[111] D. Koller, J. Weber, and J. Malik, "Robust multiple car tracking with occlusion reasoning," in *Computer Vision - ECCV '94* (J.-O. Eklundh, ed.), vol. 800 of *Lecture Notes in Computer Science*, pp. 189 – 196, Springer Berlin / Heidelberg, 1994.

[112] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik, "A real-time computer vision system for vehicle tracking and traffic surveillance," *Transportation Research Part C: Emerging Technologies*, vol. 6, no. 4, pp. 271 – 288, 1998.

[113] R. Collins, O. Amidi, and T. Kanade, "An active camera system for acquiring multi-view video," in *Image Processing. 2002. Proceedings. 2002 International Conference on*, vol. 1, pp. 520 – 527, 2002.

[114] A. Namiki, Y. Nakabo, I. Ishii, and M. Ishikawa, "High speed grasping using visual and force feedback," in *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, vol. 4, pp. 3195 – 3200, 1999.

[115] A. Blake and M. Isard, "3d position, attitude and shape input using video tracking of hands and lips," in *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, SIGGRAPH '94, pp. 185 – 192, 1994.

[116] T. Darrell, B. Moghaddam, and A. Pentland, "Active face tracking and pose estimation in an interactive room," in *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR '96, 1996 IEEE Computer Society Conference on*, pp. 67 –72, 1996.

[117] G. Foresti, C. Micheloni, L. Snidaro, P. Remagnino, and T. Ellis, "Active video-based surveillance system: the low-level image and video processing techniques needed for implementation," *Signal Processing Magazine, IEEE*, vol. 22, no. 2, pp. 25 – 37, 2005.

[118] C. Harris, "Active vision," ch. Tracking with rigid models, pp. 59 – 73, Cambridge, MA, USA: MIT Press, 1993.

[119] A. Blake and M. Isard, *Active Contours: The Application of Techniques from Graphics, Vision, Control Theory and Statistics to Visual Tracking of Shapes in Motion.* Springer-Verlag New York, Inc., 1998.

[120] R. Beira, M. Lopes, M. Praga, J. Santos-Victor, A. Bernardino, G. Metta, F. Becchi, and R. Saltaren, "Design of the robot-cub (icub) head," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pp. 94 –100, 2006.

[121] S. Ivaldi, *From humans to humanoids: a study on optimal motor control for the iCub.* PhD thesis, University of Genoa, 2011.

[122] F. Barranco, J. Diaz, A. Gibaldi, S. P. Sabatini, and E. Ros, "Vector disparity sensor with vergence control for active vision systems," *Sensors*, vol. 12, no. 2, pp. 1771 – 1799, 2012.

[123] J. L. Semmlow, W. Yuan, and T. L. Alvarez, "Evidence for separate control of slow version and vergence eye movements: support for hering's law," *Vision Research*, vol. 38, no. 8, pp. 1145 – 1152, 1998.

[124] K. Groos, "Die spiele der thiere," 1896.

[125] K. Groos, "Principles of psychology," 1890.

[126] L. Itti, *Models of Bottom-Up and Top-Down Visual Attention.* PhD thesis, Jan 2000.

[127] "Merriam webster dictionary (2012 web edition)," 2012.

[128] C. Koch and S. Ullman, “Shifts in selective visual attention: towards the underlying neural circuitry.,” *Human neurobiology*, vol. 4, no. 4, pp. 219 – 227, 1985.

[129] D. L. Robinson and S. E. Petersen, “The pulvinar and visual salience.,” *Trends in neurosciences*, vol. 15, pp. 127 – 132, Apr. 1992.

[130] A. A. Kustov and D. L. Robinson, “Shared neural control of attentional shifts and eye movements.,” *Nature*, vol. 384, pp. 74 – 77, Nov. 1996.

[131] Z. Li, “A saliency map in primary visual cortex,” *Opinion TRENDS in Cognitive Sciences*, vol. 6, pp. 9 – 16, Jan. 2002.

[132] J. A. Mazer and J. L. Gallant, “Goal-related activity in V4 during free viewing visual search. Evidence for a ventral stream visual salience map.,” *Neuron*, vol. 40, pp. 1241 – 1250, Dec. 2003.

[133] J. Gottlieb, “From Thought to Action: The Parietal Cortex as a Bridge between Perception, Action, and Cognition,” *Neuron*, vol. 53, pp. 9 – 16, Jan. 2007.

[134] R. Desimone and J. Duncan, “Neural Mechanisms of Selective Visual Attention,” *Annual Review of Neuroscience*, vol. 18, no. 1, pp. 193 – 222, 1995.

[135] M. C. Mozer, *The perception of multiple objects: a connectionist approach.* PhD thesis, Jun 1991.

[136] N. Bruce, *Saliency, attention and visual search: an information theoretic approach.* PhD thesis, 2008.

[137] D. H. O’Connor, M. M. Fukui, M. A. Pinsk, and S. Kastner, “Attention modulates responses in the human lateral geniculate nucleus,” *Nature Neuroscience*, vol. 5, pp. 1203 – 1209, Oct. 2002.

[138] B. C. Motter, “Focal attention produces spatially selective processing in visual cortical areas V1, V2, and V4 in the presence of competing stimuli.,” *Journal of neurophysiology*, vol. 70, pp. 909 – 919, Sept. 1993.

[139] S. J. Luck, L. Chelazzi, S. A. Hillyard, and R. Desimone, “Neural mechanisms of spatial selective attention in areas V1, V2, and V4 of macaque visual cortex.,” *Journal of neurophysiology*, vol. 77, pp. 24 – 42, Jan. 1997.

[140] C. E. Connor, J. L. Gallant, D. C. Preddie, and D. C. Van Essen, “Responses in area V4 depend on the spatial relationship between stimulus and attention.,” *J Neurophysiol*, vol. 75, pp. 1306 – 1308, Mar. 1996.

[141] P. E. Haenny and P. H. Schiller, “State dependent activity in monkey visual cortex. I. Single cell activity in V1 and V4 on visual tasks.,” *Experimental brain research. Experimentelle Hirnforschung. Expérimentation cérébrale*, vol. 69, no. 2, pp. 225 – 244, 1988.

[142] S. Treue and J. C. Trujillo, “Feature-based attention influences motion processing gain in macaque visual cortex,” *Nature*, vol. 399, pp. 575 – 579, June 1999.

[143] S. Treue and J. H. Maunsell, “Attentional modulation of visual motion processing in cortical areas MT and MST.,” *Nature*, vol. 382, pp. 539 – 541, Aug. 1996.

[144] M. Bushnell, M. Goldberg, and D. Robinson, “Behavioral enhancement of visual responses in monkey cerebral cortex. i. modulation in posterior parietal cortex related to selective visual attention.,” *J Neurophysiol*, vol. 46, no. 4, pp. 755–72, 1981.

[145] C. L. Colby, J. R. Duhamel, and M. E. Goldberg, “Visual, presaccadic, and cognitive activation of single neurons in monkey lateral intraparietal area.,” *J Neurophysiol*, vol. 76, pp. 2841 – 2852, Nov. 1996.

[146] B. C. Motter, “Neural correlates of attentive selection for color or luminance in extrastriate area V4,” *Journal of Neuroscience*, vol. 14, no. 4, pp. 2178 – 2189, 1994.

[147] J. H. Maunsell, G. Sclar, T. A. Nealey, and D. D. Depriest, “Extraretinal representations in area V4 in the macaque monkey.,” *Visual Neuroscience*, vol. 7, pp. 561 – 573, Dec. 1991.

[148] V. P. Ferrera, T. A. Nealey, and J. H. Maunsell, “Responses in macaque visual area V4 following inactivation of the parvocellular and magnocellular LGN pathways.,” *The Journal of neuroscience : the official journal of the Society for Neuroscience*, vol. 14, pp. 2080 – 2088, Apr. 1994.

[149] S. Kastner, M. A. Pinsk, P. De Weerd, R. Desimone, and L. G. Ungerleider, “Increased Activity in Human Visual Cortex during Directed Attention in the Absence of Visual Stimulation,” *Neuron*, vol. 22, pp. 751 – 761, Apr. 1999.

[150] S. Kastner, P. De Weerd, R. Desimone, and L. G. Ungerleider, “Mechanisms of Directed Attention in the Human Extrastriate Cortex as Revealed by Functional MRI,” *Science*, vol. 282, pp. 108 – 111, Oct. 1998.

[151] S. J. Thorpe and M. Fabre-Thorpe, “Seeking Categories in the Brain,” *Science*, vol. 291, no. 5502, pp. 260 – 263, 2001.

[152] V. Navalpakkam and L. Itti, "Modeling the influence of task on attention," *Vision Research*, vol. 45, pp. 205–231, Jan 2005.

[153] J. M. Wolfe, K. R. Cave, and S. L. Franzel, "Guided search: an alternative to the feature integration model for visual search.," *Journal of experimental psychology. Human perception and performance*, vol. 15, pp. 419 – 433, Aug. 1989.

[154] J. Foley, A. van Dam, S. Feiner, and J. Hughes, *Computer Graphics: Principles and Practice, second edition.* Addison-Wesley Professional, 1990.

[155] A. Leventhal, *The Neural basis of visual function.* Vision and visual dysfunction, CRC Press, 1991.

[156] A. Lüschow and H. C. Nothdurft, "Pop-out of orientation but no pop-out of motion at isoluminance," *Vision Research*, vol. 33, pp. 91 – 104, 1993.

[157] S. Engel, X. Zhang, and B. Wandell, "Colour tuning in human visual cortex measured with functional magnetic resonance imaging.," *Nature*, vol. 388, pp. 68 – 71, July 1997.

[158] R. L. De Valois, D. G. Albrecht, and L. G. Thorell, "Spatial frequency selectivity of cells in macaque visual cortex.," *Vision Res*, vol. 22, no. 5, pp. 545 – 559, 1982.

[159] R. B. Tootell, S. L. Hamilton, M. S. Silverman, and E. Switkes, "Functional anatomy of macaque striate cortex. i. ocular dominance, binocular interactions, and baseline conditions.," *Journal of Neuroscience*, vol. 8, no. 5, pp. 1500 – 1530, 1988.

[160] M. W. Cannon and S. C. Fullenkamp, "Spatial interactions in apparent contrast: inhibitory effects among grating patterns of different spatial frequencies, spatial positions and orientations.," *Vision research*, vol. 31, no. 11, pp. 1985 – 1998, 1991.

[161] C. D. Gilbert and T. N. Wiesel, "Columnar specificity of intrinsic horizontal and corticocortical connections in cat visual cortex," *The Journal of Neuroscience*, vol. 9, pp. 2432 – 2442, July 1989.

[162] R. Malach, Y. Amir, M. Harel, and A. Grinvald, "Relationship between intrinsic connections and functional architecture revealed by optical imaging and *in vivo* targeted biocytin injections in the primate striate cortex," *Proceedings of the National Academy of Sciences, USA*, vol. 90, pp. 10469 – 10473, 1993.

[163] B. Zenger and D. Sagi, “Isolating excitatory and inhibitory nonlinear spatial interactions involved in contrast detection.,” *Vision Res*, vol. 36, pp. 2497 – 2513, Aug. 1996.

[164] O. Stasse, Y. Kuniyoshi, and G. Cheng, “Development of a biologically inspired real-time visual attention system,” in *Proceedings of the First IEEE International Workshop on Biologically Motivated Computer Vision*, BMVC ’00, (London, UK), pp. 150 – 159, Springer-Verlag, 2000.

[165] K. R. Cave, “The FeatureGate model of visual selection.,” *Psychological research*, vol. 62, no. 2-3, pp. 182 – 194, 1999.

[166] G. Indiveri, “A neuromorphic vlsi device for implementing 2-d selective attention systems.,” *IEEE Transactions on Neural Networks*, vol. 12, no. 6, pp. 1455 – 1463, 2001.

[167] S.-J. Park, S.-W. Ban, J.-K. Shin, and M. Lee, “Implementation of visual attention system using bottom-up saliency map model,” in *Proceedings of the 2003 joint international conference on Artificial neural networks and neural information processing*, ICANN/ICONIP’03, (Berlin, Heidelberg), pp. 678 – 685, Springer-Verlag, 2003.

[168] N. Ouerhani, H. Hügli, P.-Y. Burgi, and P.-F. Ruedi, “A real time implementation of the saliency-based model of visual attention on a simd architecture,” in *Proceedings of the 24th DAGM Symposium on Pattern Recognition*, (London, UK, UK), pp. 282 – 289, Springer-Verlag, 2002.

[169] P. Longhurst, K. Debattista, and A. Chalmers, “A gpu based saliency map for high-fidelity selective rendering,” in *Proceedings of the 4th international conference on Computer graphics, virtual reality, visualisation and interaction in Africa*, AFRIGRAPH ’06, (New York, NY, USA), pp. 21 – 29, ACM, 2006.

[170] P. Chalimbaud and F. Berry, “Embedded active vision system based on an fpga architecture,” *EURASIP J. Embedded Syst.*, vol. 2007, pp. 1 – 12, January 2007.

[171] S. Frintrop, M. Klodt, and E. Rome, “A real-time visual attention system using integral images,” *Science*, vol. 11, no. 1, pp. 191 – 193, 2007.

[172] B. Han and B. Zhou, “High speed visual saliency computation on gpu.,” in *ICIP (1)*, pp. 361–364, IEEE, 2007.

[173] T. Xu, T. Pototschnig, K. Kühnlenz, and M. Buss, “A high-speed multi-gpu implementation of bottom-up attention using cuda,” in *Proceedings of the 2009 IEEE international conference on Robotics and*

Automation, ICRA'09, (Piscataway, NJ, USA), pp. 1120 – 1126, IEEE Press, 2009.

[174] S. Bae, Y. C. P. Cho, S. Park, K. M. Irick, Y. Jin, and V. Narayanan, "An fpga implementation of information theoretic visual-saliency system and its optimization," in *Proceedings of the 2011 IEEE 19th Annual International Symposium on Field-Programmable Custom Computing Machines*, FCCM '11, (Washington, DC, USA), pp. 41 – 48, IEEE Computer Society, 2011.

[175] B. Kim, H. Okuno, T. Yagi, and M. Lee, "Implementation of visual attention system using artificial retina chip and bottom-up saliency map model.," in *ICONIP (3)* (B.-L. Lu, L. Zhang, and J. T. Kwok, eds.), vol. 7064 of *Lecture Notes in Computer Science*, pp. 416 – 423, Springer, 2011.

[176] H. Greenspan, S. Belongie, R. Goodman, P. Perona, S. Rakshit, and C. H. Anderson, "Overcomplete steerable pyramid filters and rotation invariance," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition CVPR94*, pp. 222 – 228, 1994.

[177] D. Walther and C. Koch, "Modeling attention to salient proto-objects," *Neural Networks*, vol. 19, pp. 1395 – 1407, Nov. 2006.

[178] J. M. Wolfe and T. S. Horowitz, "Opinion: What attributes guide the deployment of visual attention and how do they do it?," *Nature Reviews Neuroscience*, vol. 5, no. 6, pp. 495 – 501, 2004.

[179] J. M. Wolfe, M. L. H. Võ, K. K. Evans, and M. R. Greene, "Visual search in scenes involves selective and nonselective pathways," *Trends in Cognitive Sciences*, vol. 15, no. 2, pp. 77 – 84, 2011.

[180] H. L. Apfelbaum, C. Gambacorta, R. L. Woods, and E. Peli, "Inattentional blindness with the same scene at different scales," *Ophthalmic physiological optics the journal of the British College of Ophthalmic Opticians Optometrists*, vol. 30, no. 2, pp. 124 – 131, 2010.

[181] H. L. Apfelbaum, D. H. Apfelbaum, R. L. Woods, and E. Peli, "Inattentional blindness and augmented-vision displays: effects of cartoon-like filtering and attended scene," *Ophthalmic physiological optics the journal of the British College of Ophthalmic Opticians Optometrists*, vol. 28, no. 3, pp. 204 – 217, 2008.

[182] W. Garage, "Opencv wiki." http://opencv.willowgarage.com/wiki/.

[183] Intel, "Intel integrated performance primitives (intel ipp) 7.0." http://software.intel.com/en-us/articles/intel-ipp/.

[184] OpenMP, “The openmp api specification for parallel programming.” http://openmp.org/wp/.

[185] D. V. Heesch, “Doxygen manual.” http://www.stack.nl/ dimitri/doxygen/.

[186] Celoxica, “Accelerated trading solutions.” http://www.celoxica.com/.