Universidad de Granada

DEPARTAMENTO DE ARQUITECTURA Y TECNOLOGÍA DE COMPUTADORES

Sistema de visión bio-inspirado multi-modal. Arquitectura de procesamiento de movimiento y visión estéreo de altas prestaciones

Multimodal bio-inspired vision system.

High performance motion and stereo processing architecture.

TESIS DOCTORAL

Javier Diaz Alonso

Granada, 2006

Editor: Editorial de la Universidad de Granada
Autor: Antonio Javier Díaz Alonso
D.L.: Gr. 1366 - 2006
ISBN: 978-84-338-4032-5

Dr. Eduardo Ros Vidal, Profesor Titular de Universidad, y Alberto Prieto Espinosa, Catedrático de Universidad, ambos del Departamento de Arquitectura y Tecnología de Computadores de la Universidad de Granada,

CERTIFICAN:

Que la memoria titulada **“Sistema de visión bio-inspirado multi-modal. Arquitectura de procesamiento de movimiento y visión estéreo de altas prestaciones”**, ha sido realizada por D. Javier Díaz Alonso bajo nuestra dirección en el Departamento de Arquitectura y Tecnología de Computadores de la Universidad de Granada para optar al grado de Doctor y a la mención especial de Doctor Europeo por la Universidad de Granada.

Granada, a 16 de Mayo de 2006

Fdo.: Dr. Eduardo Ros Vidal

Director de la Tesis

Fdo.: Alberto Prieto Espinosa

Director de la Tesis

Universidad de Granada

DEPARTAMENTO DE ARQUITECTURA Y TECNOLOGÍA DE COMPUTADORES

Sistema de visión bio-inspirado multi-modal. Arquitectura de procesamiento de movimiento y visión estéreo de altas prestaciones

Multimodal bio-inspired vision system.

High performance motion and stereo processing architecture.

Memoria presentada por

Javier Díaz Alonso

Para optar al grado de

DOCTOR POR LA UNIVERSIDAD DE GRANADA

Fdo.: Javier Díaz Alonso

Agradecimientos

Llegados a este punto perdonadme el formalismo. Muchos son los sucesos y muchas las personas que me han ayudado. Disculpad las seguro injustas omisiones.

Quiero empezar agradeciendo a toda mi familia su apoyo y cariño. Aun a riesgo de ser injusto, permitidme dar nombres:

A mi mujer, Beatriz, mi complemento. Por enseñarme a parar, que la vida la forman todo un conjunto de cosas.

A mi Madre, mi aliento. Por saber enseñarme a ver los escalones sin necesidad de caerme.

A mis hermanas, mi orgullo. Por iluminarme con su alegría.

A mis tíos Nani e Ignacio y mis primos, Nayara y Alvaro, mi refugio, Por abrirnos sus puertas, por su generosidad desmedida y por su cariño.

Tampoco quiero olvidar a mis amigos, especialmente a Juan Pablo, Juanito, David y Rodo. En su amistad se encuentra mi riqueza. Ellos siempre han sabido disculpar mis ausencias y alentarme en mis empresas. Gracias a todos por saber estar a mi lado.

Por supuesto existe mucha gente que desde este lado, el de la ciencia, me ha ayudado de múltiples maneras. Quiero agradecer a todos los miembros del Departamento de Arquitectura de computadores de la Universidad de Granada su consejo y ejemplo en numerosas ocasiones. Sin menoscabo de nadie, me gustaría destacar a:

Eduardo Ros. Gracias por permitirme ver las cosas desde ese otro tan infrecuente punto de vista, ese que no se enseña. Gracias por no escatimar esfuerzo para apostar por las personas.

Francisco Pelayo. Cuando alguien disfruta de su trabajo, lo hace bien. Y esa capacidad para disfrutar se transmite. Gracias por escuchar mis preguntas. Gracias por, hace ya cerca de cuatro años, abrirme la puerta.

Alberto Prieto. Es difícil crear buen clima de trabajo, difícil conjugar distintas inquietudes y difícil entender que la heteregenoidad en un grupo no merma sino enriquece. Gracias por ver que el total es más que la suma de sus partes.

Por supuesto no olvido a mis compañeros/as del "Torreón de las siete llaves", Sonia, Eva, Samuel, Antonio, Rodrigo, Richard, Christian, y Rafa, mis compañeros de viaje. Gracias por vuestro consejo, paciencia y nobleza.

Finalmente y no menos importante, Mancia Anguita, Begoña del Pino, Manuel Rodríguez, Francisco Illeras y Encarnación Redondo. Gracias por vuestros consejos y ayuda.

Acknowledgment

I have been very fortunate to be involved in the DRIVSCO and ECOVISION working groups, which are filled with unusual and notable assortment of characters and scientific skills. Thanks to all of them for answer all my questions and tolerate my frequent mistakes. I have learnt a lot from them but, if I would have to keep just one thing, I would like to remark that I learn from them the curiosity spirit, curiosity about all the fields, scientific or not. Thanks to all of you.

I would like to thank to the reader for his interest and please, let me take the opportunity to apologize for mistakes of this document especially in the English version where my skills are very limited. Thanks again to all of you.

"Tell a man that there are 300 billion stars in the universe, and he'll believe you.... Tell him that a bench has wet paint upon it and he'll have to touch it to be sure."

Raimond Verwei (also credited to Albert Einstein)

“A clever person solves a problem. A wise person avoids it. “

Albert Einstein

“La inteligencia anula al destino. Mientras un hombre piensa, es libre. “

Ramon y Cajal

Resumen

Esta tesis doctoral trata sobre el diseño de nuevas arquitecturas hardware para la visión por computador en tiempo real. Está claramente estructurada en dos partes. La primera parte es una revisión de los modelos existentes para la estimación de primitivas visuales. Tres primitivas visuales han sido objeto del estudio: características visuales locales (fase, energía y orientación), estimación de movimiento y estimación de disparidad para sistemas binoculares. En esta parte de la memoria se concluye qué aproximaciones presentan un buen compromiso entre consumo de recursos del hardware y precisión del modelo, teniendo como objetivo la optimización de estos parámetros

La segunda parte de la memoria trata acerca del diseño de estos modelos usando caminos de datos específicos. En ella adoptamos técnicas de diseño innovadoras basadas en un uso intensivo del paralelismo masivo disponible en dispositivos basados en hardware reconfigurable. Hemos realizado un análisis detallado de la profundidad de bits requerida en distintas etapas de computación para optimizar el consumo de recursos del sistema. El uso extensivo de la segmentación de cauce de grano fino así como la utilización de múltiples unidades escalares permite alcanzar un rendimiento final de una estimación por ciclo de reloj del sistema. Esto representa una alta potencia computacional que no ha sido alcanzada con anterioridad para modelos de visión de bajo nivel.

Los resultados muestran que estas técnicas pueden ser utilizadas satisfactoriamente en el diseño de circuitos de alto rendimiento para visión por computador. Fase, energía, orientación, movimiento y disparidad son procesadas usando un único chip FPGA como elemento de cómputo. Esto abre la puerta para nuevos retos en el campo de visión por computador en tiempo real gracias al uso de sistemas de procesamiento de alto rendimiento.

Abstract

This Phd-dissertation focuses on the design process of new hardware architectures for real-time computer vision models. This work is clearly structured in two parts. The first one is a review of models which analyzes the different alternatives for computing the early vision features. Three main visual primitives are addressed: local image features (energy, phase and orientation), motion estimation and disparity for binocular systems. This part highlights the main models suitable of hardware implementation, taking into account that our goal is to achieve a good accuracy vs. performance/cost trade-off.

The second part of this dissertation describes the implementation of these models into specific datapaths. We adopt innovative design techniques based on the intensive utilization of the inherent parallelism available on devices based on reconfigurable hardware. We perform a detailed bit-width analysis to effectively adjust the required hardware resources. We exploit the fine-grain pipelining and superscalar units capabilities of such devices to develop computing circuits that achieve a throughput of one feature estimation per clock cycle. This represents an outstanding performance for early computer vision models.

The results show that this design strategy is very efficient for the hardware implementation of high performance computer vision circuits. Phase, energy, orientation, motion and disparity estimation are available using just a single FPGA device as processing element, which open the doors to new challenges in the field of real-time computer vision based on high performance smart processing systems.

Astratto

Questa tesi di dottorato si occupa del disegno di nuove architetture hardware per la visione in tempo reale attraverso un calcolatore. La sua struttura puó essere chiaramente divisa in due parti.

La prima parte consiste in una rivisitazione dei modelli giá esistenti per la stima delle primitive visuali. In particolare si sono studiate tre di queste primitive: caratteristiche visuali locali (fase, energia e orientazione), stima di moto e stima della disparitá per sistemi binoculari. In questa parte del trattato si giunge alla conclusione di quali approssimazioni presentano un buon compromesso tra l'utilizzo di risorse hardware e la precisione del modello, tenendo sempre come obiettivo l'ottimizzazione di questi parametri.

La seconda parte invece si occupa del disegno di questi modelli tramite l'utilizzo di specifici datapaths. In essa si adottano tecniche di disegno innovative, basate sull'uso intensivo del parallelismo massivo, disponibile in dispositivi caratterizzati da hardware riconfigurabile. Per ottimizzare il consumo di risorse del sistema abbiamo realizzato un'analisi dettagliata della profonditá di bit richiesta, attraverso diverse tappe di calcolo. L'uso intensivo della segmentazione tramite pipeline a grana fine e l'uso di unitá scalari multiple permette di ottenere un risultato finale con la precisone di un ciclo di clock. Questo rappresenta di fatto un'alta potenza di calcolo che non é mai stata raggiunta prima d'ora in modelli per la visione di basso livello. I risultati mostrano che queste tecniche possono essere usate in maniera soddisfacente anche nel disegno di circuiti di alto rendimento per la visione attraverso un calcolatore. Fase, energia, orientazione, movimento e disparitá sono processati utilizzando come elemento di calcolo una singola FPGA.

Possiamo dire infine che tutte queste importanti innovazioni aprono le porte a nuovi sviluppi nel campo della visione in tempo reale attraverso il calcolatore, grazie all'uso di sistemi di processamento ad alto rendimento.

General Index

Figure Index

Table index

Capítulo 1
Introducción

En este capítulo de introducción explicamos de forma breve la motivación y los objetivos del trabajo. También se describen las distintas contribuciones de esta tesis, en el marco de un sistema de visión por computador constituido por diversos componentes que han sido desarrollados con circuitos específicos. Este trabajo se ha realizado en el marco de dos proyectos Europeos; y dentro de este capítulo indicamos cómo los objetivos de este trabajo se relacionan con los objetivos más globales de los proyectos de investigación en los que se enmarca. Finalmente esquematizamos los contenidos de la presente memoria.

1.1. Motivación

Tras varias décadas de investigación en el campo de la visión por computador aún estamos muy lejos de entender cómo funciona el sistema visual humano. Muchos grupos de investigación se han centrado en la simulación de procesamientos corticales de áreas cerebrales concretas (estimación de movimiento, estéreo, color, etc.). Este tipo de simulaciones consumen mucho tiempo y han forzado a muchos investigadores a desarrollar modelos my simplificados.

La principal aportación de este trabajo es la implementación de arquitecturas eficientes de procesamiento de visión en tiempo real y viene motivada por los siguientes aspectos:

a. *Procesamiento en tiempo real para experimentos con agentes que incluyan visión.* En la actualidad es claro que simulaciones sencillas *off-line* no son suficientes para entender la forma en que distintas tareas de procesamiento se realizan concurrentemente en el cortex visual. Es más, existe una hipótesis fuerte de trabajo llamada "embodiment concept" (difícil de traducir) que establece que cualquier simulación realista de procesamientos bio-inspirados debe ser probada en el marco de alguna tarea concreta también biológicamente plausible. El grado de consecución de esta tarea se puede utilizar para validar las distintas partes que contribuyen al éxito del sistema diseñado. Este concepto se basa en la hipótesis que establece que la biología ha desarrollado sistemas de unas prestaciones impresionantes por medio del proceso evolutivo natural en el que los sistemas que perduran son los que optimizan ciertas tareas que mejoran la supervivencia individual y la perpetuación de la especie.

b. *Visión activa.* El proceso de percepción es activo. Combina capacidades sensori-motoras de forma integradora. No sólo el sentido del tacto requiere una planificación de movimientos y exploración, también la visión se entiende como un proceso activo en el que primitivas volutivas dirigen ciertos mecanismos (como fijación, seguimiento suave para estabilización, etc.) que aumentan la precisión del sistema. Además se cree que la atención constituye un mecanismo eficiente para alcanzar altas prestaciones con recursos limitados. Pero este tipo de procesos activos pueden ser estudiados sólo en el marco de ciclos cerrados de percepción-acción. Todo ello requiere procesamiento en tiempo real y representa un motivación muy fuerte para el desarrollo de arquitecturas de procesamiento visual de altas prestaciones.

c. *Aprender construyendo.* Desde un punto de vista de un ingeniero una metodología habitual de trabajo es construir sistemas para llegar a conocer a fondo cómo funcionan. En el marco de sistemas de visión por computador, como ingenieros, tratamos de construir arquitecturas de procesamiento basadas en los sistemas biológicos. Esta aproximación es muy interesante ya que nos fuerza a enfrentarnos a los mismos objetivos parciales y limitaciones en cuanto a recursos de procesamiento reducidos. De acuerdo con el paradigma de la *Ingeniería Neuromórfica* adoptamos una actitud oportunista tratando de emular de la biología esquemas de procesamiento que parecen eficientes pero evitamos adoptar otras características que están más relacionadas con los tejidos nerviosos en los que se basan. Además, al no estar limitados por algunas características de las neuronas naturales (como potencia, conductividad, velocidad de conmutación y conexiones punto a punto) podemos aprovechar ciertas características relevantes de la tecnología electrónica en la que vamos a implementar nuestros circuitos como son los altos anchos de banda, la rápida conmutación de estado, etc.

d. *Sistemas de visión eficientes para aplicaciones reales.* El procesamiento en tiempo real al que se refiere esta memoria (fase, magnitud, orientación, movimiento y estéreo) es muy interesante para una gran variedad de aplicaciones. Por lo tanto, la implementación de arquitecturas de procesamiento de altas prestaciones para estas tareas tiene un interés en sí mismo para su aplicación directa a problemas reales.

1.2. Estructura de una máquina visual

La percepción visual es un proceso complejo que transforma señales en conocimiento. Aunque no existe un acuerdo general sobre la estructura de este sistema, a nivel conceptual podemos dividir la visión en distintas etapas que manejan información de distintos niveles de abstracción (mostrado esquemáticamente en la Figura 1.1):

1. **Visión pre-cognitiva.** (Visión de bajo nivel).

1.1. *Muestreo espacio-temporal básico.* Esta etapa está compuesta por filtros espacio-temporales. Los sistemas biológicos utilizan células cuyos campos receptivos proyectan en las retinas. Los modelos visuales artificiales utilizan un serie de filtros espacio-temporales básicos de distinto tamaño espacial y características temporales (correspondientes a derivadas espacio-temporales de distinto orden) que tratan de emular el comportamiento de esas estructuras neuronales y campos receptivos.

1.2. *Visión estéreo.* Combinación de respuestas de filtros de dos retinas para extraer estimaciones de profundidad.

1.3. *Procesamiento de movimiento.* Combinación de respuestas de filtros de la misma retina para obtener estimaciones de movimiento.

1.4. *Procesamiento de color.* Combinación de distintos filtros monocromos para codificar de forma fiable y eficiente los colores naturales.

1.5. *Extracción de estructura.* Integración de salidas de diferentes filtros espaciales para resaltar áreas de la imagen que codifican una cantidad significativa de estructura de la imagen.

2. **Visión cognitiva media.** Mecanismos de integración que permiten combinar eficientemente y de forma constructiva diferentes modalidades visuales (movimiento, estéreo, orientación, etc.). En esta etapa diferentes canales de información se integran para formar entidades multimodales. Para extraer sólo información fiable y descartar características erróneas se utilizan distintos mecanismos (integración co-planar, criterios de co-linealidad, etc.).

   Los mecanismos de segmentación que integran entidades multimodales correspondientes a distintos eventos del entorno, por ejemplo *efectos de causa común* como *IMOs* (Objetos Independientes en Movimiento), movimiento propio (egomovimiento), etc. Estos mecanismos de segmentación llevan a una escena en la que diferentes elementos se identifican como candidatos a objetos o efectos de una causa común (como puede ser egomovimiento) y se relacionan con caracterizaciones multimodales que pueden ser estáticas o dirigidas por ciertas limitaciones estructurales de modelos como es el caso del movimiento de sólidos rígidos.

3. **Visión de alto nivel.** Esta es una etapa de muy alto nivel en donde tiene lugar la interpretación de la escena por medio de sub-tareas más específicas como son el

reconocimiento de objetos, predicción de efectos, comparación con escenas/situaciones ya percibidas con anterioridad, etc.

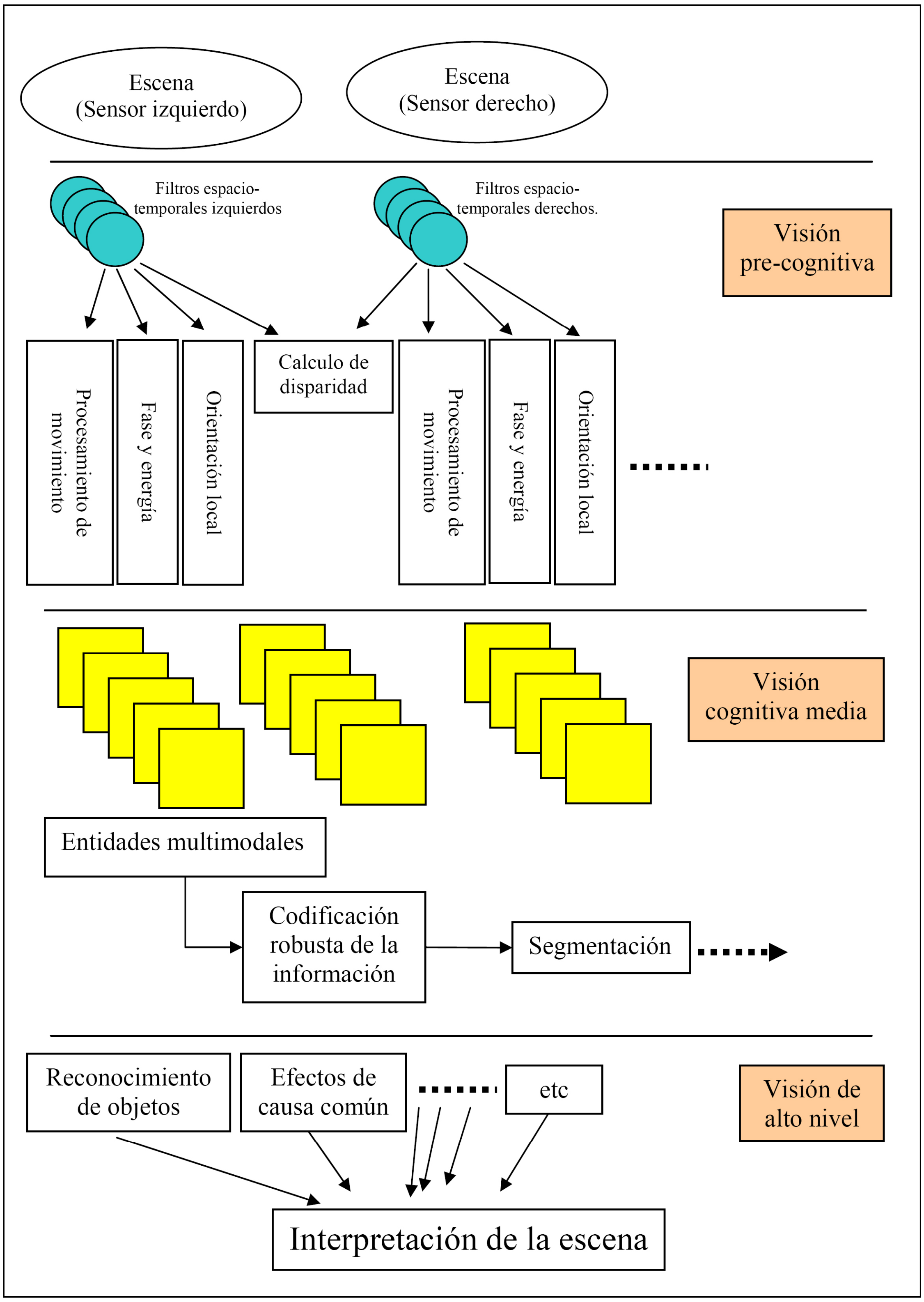

Escena (Sensor izquierdo)
Escena (Sensor derecho)
Filtros espacio-temporales izquierdos
Filtros espacio-temporales derechos.
Visión pre-cognitiva
Procesamiento de movimiento
Fase y energía
Orientación local
Calculo de disparidad
Procesamiento de movimiento
Fase y energía
Orientación local
Visión cognitiva media
Entidades multimodales
Codificación robusta de la información
Segmentación
Reconocimiento de objetos
Efectos de causa común
etc
Visión de alto nivel
Interpretación de la escena

Figura 1.1. Esquema de un sistema de visión con distintos niveles de abstracción.

Es importante resaltar que el procesamiento de las etapas pre-cognitivas (extracción de características) es inherentemente denso, es decir, requiere del procesamiento de cada píxel de la imagen, mientras que etapas de más alto nivel sólo computan entidades discretas (entidades multimodales, objetos, efectos de causa común como el egomovimiento, etc.). Los sistemas de visión por computador actuales no son capaces de realizar este procesamiento denso de las distintas modalidades en tiempo real.

Uno de los principales objetivos de este trabajo de tesis es la implementación de una arquitectura de procesamiento visual de altas prestaciones capaz de computar en tiempo real primitivas de bajo nivel. Esto tiene un gran interés porque las etapas de más alto nivel de sistemas de visión complejos (que son inherentemente discretos) se ajustan mejor a plataformas basadas en un solo procesador potente mientras que las tareas de visión pre-cognitiva se adaptan mejor a un paralelismo de grano fino. Esto se ilustrará en los Capítulos del 5 al 7 en los que diseñamos arquitecturas con gran paralelismo basadas en un flujo de datos regular para procesamientos de bajo nivel. Más concretamente, en este trabajo nos centramos en la implementación de arquitecturas de altas prestaciones para movimiento, estéreo y análisis de estructura local (orientación, fase y energía).

1.3. De modelos biológicos a sistemas hardware en tiempo real

El desarrollo de arquitecturas de procesamiento diseñadas para tareas que los sistemas biológicos resuelven con impresionante facilidad puede realizarse tratando de emular a estos últimos para de este modo aprovechar estrategias de computación evolucionadas por la naturaleza durante millones de años. Pero la adaptación de estas técnicas no es directa ya que los principios físicos en los que se basan los tejidos del sistema nervioso son muy diferentes de los que utiliza la tecnología electrónica. Además, las *"tecnologías"* biológica y electrónica tienen diferentes limitaciones que superan con estrategias también diferentes.

Sin embargo, una "actitud oportunista" que adopte los principios clave en los que se basan las impresionantes prestaciones de los sistemas biológicos y utilice técnicas propias de la tecnología electrónica para adaptar esas primitivas bio-inspiradas es de gran interés. Esta metodología oportunista debería dar lugar a soluciones específicas para tareas individuales y además ayudar a identificar y caracterizar los principios funcionales en los que se basan las altas prestaciones de los sistemas biológicos. Por ejemplo, los sistemas biológicos utilizan de forma generalizada computación masivamente paralela para aprovechar al máximo los lentos procesos electro-químicos en los que se basan las transmisiones sinápticas. Por otro lado, aunque la tecnología electrónica permite dispositivos más rápidos (más de tres órdenes de magnitud en cuanto a capacidad de conmutación), la conectividad posible en la tecnología del silicio actual, frenta a la biología, está limitada a patrones 2-D por lo que el paralelismo masivo real es imposible a nivel de dispositivos electrónicos.

Para adoptar esquemas de procesamiento inspirados en la biología utilizamos técnicas como multiplexado temporal. Además desarrollamos elementos de computación rápida que abstraen los principios funcionales en los que se basa el sistema que emulamos. De esta forma, por ejemplo en estéreo, podemos computar la disparidad entre dos imágenes varias veces (con distintas escalas y desplazamientos espaciales) para obtener múltiples estimaciones de disparidad que en un sistema biológico se extraerían en poblaciones de neuronas diferentes. Tras esto, integramos todas estas estimaciones de forma constructiva para conseguir el mejor rendimiento.

En esta memoria se ilustran varios ejemplos de esta metodología. Hemos desarrollado un sistema para procesar características locales de las imágenes, flujo óptico y estéreo, de forma que es capaz de extraer todas ellas con alta resolución temporal y espacial. Esto permite el estudio de esquemas de integración en el ámbito de tareas de procesamiento en tiempo real. Por ejemplo, esta computación eficiente permite incluso el estudio de códigos basados en codificación en poblaciones de neuronas que representen un conjunto de estimaciones obtenidas en intervalos de tiempo sucesivos.

Convencionalmente, el paralelismo de la mayoría de circuitos diseñados con dispositivos de tipo FPGA está restringido por un ancho de banda limitado. Este factor es especialmente crítico en los accesos a memoria que representan un importante cuello de botella. Por ello hemos diseñado circuitos específicos para la gestión eficiente de los recursos de memoria dentro y fuera del chip. Además como el sistema desarrollado en este trabajo se ha implementado en un solo dispositivo (*sistema en un chip*, SoC) el acceso a estos valiosos recursos secuenciales se ha diseñado de forma eficiente.

Las arquitecturas desarrolladas son escalables; como sobran recursos de computación en el dispositivo pueden replicarse unidades funcionales si se precisace un mayor paralelismo para aumentar las prestaciones del sistema.

1.4. Marco de este trabajo

Este trabajo se ha desarrollado en el marco de dos proyectos Europeos:

- ECOVISION: Artificial Vision Systems based on early cognitive cortical processing (IST-2001-32114), (01-01-2002 hasta 30-12-2004), [ECO06].
- DRIVSCO: Learning to emulate perception action cycles in a driving school scenario (016276-2) (01-02-2006 hasta 31-01-2009) , [DRI06].

En ambos proyectos uno de los principales objetivos es la implementación de sistemas de visión por computador en tiempo real debido a su interés para potenciales aplicaciones en diversos campos. Es más, en ampos proyectos participa un miembro industrial relacionado con la industria automovilística que ha definido aplicaciones específicas en las cuales este tipo de sistemas son de gran interés. De hecho, se ha realizado un esfuerzo importante durante este periodo de tesis en la validación de esta tecnología (visión en tiempo real en FPGAs) para tareas concretas en escenarios reales (por ejemplo el seguimiento fiable de vehículos durante maniobras de adelantamiento). DRIVSCO es un proyecto actualmente en curso que tiene como objetivo la implementación y validación de estrategias de aprendizaje basadas en información visual para el desarrollo de sistemas de ayuda a la conducción en condiciones nocturnas (véase por ejemplo el Capítulo 8). Esta tarea requiere movimiento y estéreo procesado en tiempo real utilizando hardware específico de altas prestaciones como el presentado en esta memoria. Los circuitos desarrollados en este trabajo se utilizan actualmente en el consorcio DRIVSCO para el estudio de mecanismos de integración de información en tareas visuales de nivel medio y alto.

1.5. Compromiso entre investigación y desarrollo en una tesis científica

El trabajo descrito en este documento presenta los resultados de cuatro años dedicados a diferentes tareas de investigación en arquitecturas eficientes de procesamiento de imágenes. La línea divisoria entre investigación y desarrollo tecnológico no está claramente trazada. De hecho, cualquier acción investigadora requiere de unas tareas de desarrollo para realizar los experimentos y extraer resultados. Más concretamente en nuestro caso, el diseño del sistema y la evaluación de su rendimiento requiere un considerable trabajo de desarrollo.

La tesis tiene objetivos de investigación bien definidos (representados en la Figura 1.2)

- Viabilidad del diseño de circuitos de visión para cálculo de primitivas de la imagen (rasgos locales, movimiento y estéreo) en tiempo real.
- Evaluación de la precisión frente a la eficiencia y prestaciones de diferentes métodos de procesamiento. Hemos utilizado estudios previos que analizaban la precisión de las diferentes alternativas para estimación de estéreo o movimiento. Por otra parte, para la extracción de los rasgos locales de la imagen (fase, orientación y energía) hemos llevado a cabo estudios detallados para ser capaces de elegir un enfoque apropiado del tipo de filtrado espacial. Por lo que sabemos, no existen hasta ahora en la literatura estudios comparativos sobre las distintas alternativas existentes para su cómputo.
- Implementación de caminos de datos de altas prestaciones para diferentes primitivas visuales:
    - Exploración de estrategias de diseño de circuitos para modelos de flujo de datos regular.
    - Implementación de segmentación de cauce de grano fino y caminos de datos superescalares.
    - Evaluación de la realización de los circuitos propuestos.

El cumplimiento de estos objetivos ha requerido una alta carga de trabajo de desarrollo, pero también nos hemos enfrentado a multitud de retos durante la realización de esta tesis. Por ejemplo, en lugar de proponer nuevos algoritmos para primitivas de visión, hemos implementado circuitos específicos para procesarlas eficientemente. Pero la translación de los algoritmos a su implementación hardware en la tecnología adecuada requiere la evaluación del modelo propuesto, su simplificación y la medida de la degradación del diseño debido a la existencia de recursos limitados (aritmética de punto fijo, y profundidad de bits muy restringida, etc...). De hecho, la implementación misma puede considerarse como un nuevo modelo cuyas prestaciones y precisión deben ser evaluadas convenientemente comparándolas con otros enfoques descritos en la literatura. La metodología de trabajo adoptada ha forzado a usar secuencias o imágenes de test (tanto sintéticos como reales) para la evaluación de las prestaciones y de la precisión de los sistemas diseñados.

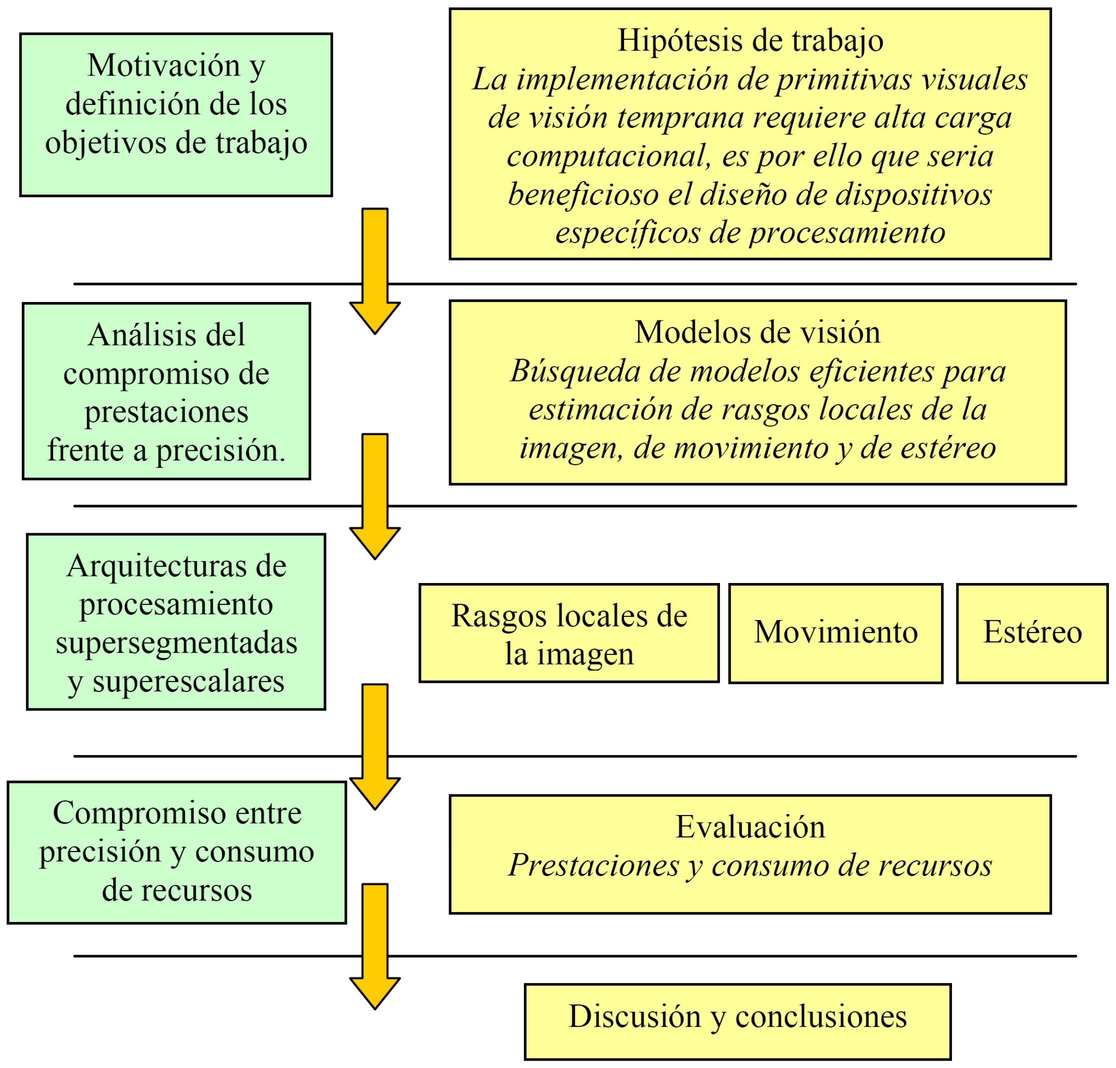

Motivación y definición de los objetivos de trabajo
Hipótesis de trabajo
La implementación de primitivas visuales de visión temprana requiere alta carga computacional, es por ello que seria beneficioso el diseño de dispositivos específicos de procesamiento
Análisis del compromiso de prestaciones frente a precisión.
Modelos de visión
Búsqueda de modelos eficientes para estimación de rasgos locales de la imagen, de movimiento y de estéreo
Arquitecturas de procesamiento supersegmentadas y superescalares
Rasgos locales de la imagen
Movimiento
Estéreo
Compromiso entre precisión y consumo de recursos
Evaluación
Prestaciones y consumo de recursos
Discusión y conclusiones

Figura 1.2. Esquema del trabajo desarrollado en la tesis.

En suma, el campo del diseño de sistemas de computación de altas prestaciones de computación requiere un gran trabajo de desarrollo no sólo para el diseño específico de la arquitectura propuesta, sino también para el estudio de las prestaciones obtenidas con secuencias o imágenes de evaluación. No obstante, hay cuestiones específicas que representan el objetivo principal de este trabajo:

a. Los circuitos de visión desarrollados deben considerarse como modelos que se comportan de forma similar a las versiones de software, pero que debido a las limitaciones de recursos, implican unos compromisos entre potencia de cálculo y precisión totalmente diferentes a las respectivas versiones software. Por tanto, los nuevos modelos necesitan ser evaluados y comparados con los demás métodos tal y como hemos hecho en el presente trabajo.
b. En nuestros circuitos computacionales hemos hecho uso amplio de la técnica de segmentación de cauce de grano fino para el diseño de diferentes modelos. Esta estrategia, aunque conocida, ha sido explotada de una manera innovadora, aunque es rara vez adoptada por otros autores. Más aún, mostramos que esta estrategia es óptima para un uso eficiente del paralelismo inherente a los dispositivos FPGA, permitiendo diseñar arquitecturas de gran potencia de cálculo.
c. Estos dos tópicos han facilitado la amplia publicación de resultados del trabajo de investigación, como se detalla en el capitulo de conclusiones.

1.6. Estrategia de definición de circuitos

La principal contribución de este trabajo puede considerarse el diseño y la evaluación de las prestaciones de diferentes sistemas de cómputo de primitivas visuales. Pero el diseño de un sistema complejo puede hacerse a diferentes niveles de abstracción y con diferentes herramientas de definición. Dada la alta complejidad de los modelos, hemos utilizado Handel-C [CEL06c] como lenguaje de descripción de hardware, porque permite la definición de arquitecturas de computación a diferentes niveles de abstracción. Además, la comparación con otros lenguajes más comunes, tales como VHDL o Verilog, muestra que el aumento del consumo de recursos es moderado pese a ser descrito con mayores niveles de abstracción [ORT06b]. Esto nos ha permitido definir los circuitos a nivel de transferencia de registros (RTL), pero con las abstracciones pre-definidas en un lenguaje de descripción de mayor nivel de abstracción como es Handel-C (por ejemplo, genera la abstracción de una máquina de estados que facilita el diseño de los diferentes esquemas de procesamiento serie y paralelo). Más aún, el lenguaje de descripción elegido (similar al C estándar) ha facilitado en mucho el diseño de los modelos de visión que son descritos usualmente mediante descripciones algorítmicas y no RTL.

Para la implementación de arquitecturas de procesamiento específicas, hemos utilizado dispositivos de hardware reconfigurable (FPGA). No hemos necesitado utilizar su capacidad de reconfiguración dinámica para optimizar la potencia de computación en tiempo de ejecución, pero esta tecnología ha facilitado enormemente la definición y evaluación de diferentes implementaciones. Siguiendo la ley de Moore, este tipo de circuitos integrados mantiene el incremento en términos del número de recursos que están disponibles en un solo chip. Más aún, estos dispositivos en la actualidad incluyen circuitos muy optimizados (como memorias y multiplicadores embebidos, interfaces entrada/salida de gran ancho de banda para comunicaciones, etc.). Todo ello facilita e incrementa su interés en un amplio rango de aplicaciones.

Hemos optado en este trabajo, en lugar de utilizar todas las ventajas de los recursos disponibles en una arquitectura dada (como un procesador de uso general), por definir arquitecturas de uso específico para tareas específicas, y hemos demostrado que sobrepasan claramente las prestaciones de lo procesadores de uso general. Este es un resultado destacado, que no puede obtenerse sin el uso intensivo del paralelismo de que disponen los dispositivos FPGA. De hecho, al principio del trabajo no estaba claro si los sistemas diseñados superarían a las arquitecturas de uso general (como los procesadores convencionales) ya que estos tienen frecuencias de reloj casi dos órdenes de magnitud mayores que nuestros circuitos. Además, el diseño de circuitos digitales en lógica reconfigurable diseñada como dispositivos de uso general, permite aprovechar de todas las ventajas del avance continúo de la tecnología de integración de circuitos digitales, en lugar de estar sujetos a una arquitectura de computación concreta y con el tiempo obsoleta. Todos los circuitos presentados en este trabajo podrán recompilarse (con mínimas modificaciones) a los dispositivos FPGA futuros, o chips de bajo coste para orientarse a diferentes campos de aplicación.

1.7 Metodologías y herramientas de diseño de hardware

El proceso de diseño hardware de un modelo de visión de computador en alto nivel, requiere la superación de varios retos. El modelo ha de ser adecuadamente modificado para ser adaptable al hardware. La representación aritmética debe revisarse (la representación en punto flotante utilizada en los modelos de software no encaja bien en los dispositivos embebidos) y el sistema resultante debería describirse adecuadamente para conseguir un buen compromiso entre tiempo dedicado al diseño frente a las prestaciones obtenidas del mismo. Después de todo el proceso, el sistema ha de ser evaluado para determinar la precisión final, que puede ser significativamente diferente de los modelos de software originales. Esto es evaluado específicamente para cada circuito desarrollado en los Capítulos 5, 6 y 7.

1.7.1. Motivación y herramientas para análisis de la profundidad de bits por palabra de datos

Los sistemas embebidos de altas prestaciones utilizan recursos computacionales específicos para cada etapa del proceso en un cauce de datos segmentado. Debido a ello, es crucial utilizar circuitos computacionales de bajo coste tanto como sea posible. La aritmética de punto flotante demanda grandes recursos de hardware, y por eso, los diseñadores usualmente tratan de utilizar aritmética en punto fijo. El análisis de la degradación del sistema (cuando se comparan las representaciones en punto flotante y punto fijo con determinada precisión) requiere dividir el algoritmo en múltiples subetapas con profundidades de bits limitadas y controladas. Como se comenta en [MAL06], las estrategias para llevar esto a cabo pueden ser caracterizadas, a grandes rasgos, en dos grupos. El primero es una aproximación analítica utilizada por los desarrolladores de algoritmos, que analizan los efectos de la longitud finita de palabra debido a la aritmética de punto fijo [CHA95], [GRA98]. El otro análisis se basa en las técnicas de simulación del número de bits verdaderos (*bit-true*), usadas por los diseñadores de hardware [KED98].

Existen trabajos en la literatura reciente de técnicas de compilación automática para convertir representaciones en punto flotante a punto fijo [SYN06a],[SYN06b]. El compilador BITWISE [STE00] determina la precisión de todas las entradas, las señales intermedias y de salida, en un diseño de hardware a partir de un programa descrito en C. El compilador MATCH [NAY01] desarrolla técnicas de análisis de error y precisión para programas en MATLAB. Synopsys tiene una herramienta comercial llamada *Cocentric Fixed-Point Designer* [SYN06b], que convierte punto flotante a punto fijo en el marco de un entorno de programación basado en C. Sin embargo, el código generado no es sintetizable. Constantinides [CON03] ha desarrollado una herramienta de evaluación para afrontar diseños lineales y no lineales. Chang y colaboradores [CHA02] han desarrollado una herramienta llamada PRECIS para analizar la precisión con MATLAB. También para MATLAB, en [BAN03] se presenta un algoritmo para convertir automáticamente de punto flotante a punto fijo utilizando el compilador AccelFPGA. Su propuesta necesita una precisión por defecto en constantes y variables, especificada por el usuario, que el compilador es incapaz de inferir. En el 2004, Roy y otros [ROY04] propusieron algoritmos de automatización para convertir programas MATLAB en punto flotante a programas en punto fijo en MATLAB, usando perfiles de entrada. Este sistema permite optimizar el área utilizada y las prestaciones del sistema frente al error de cuantización.

Como se deduce de la discusión previa, la selección del número de bits adecuado es un campo de investigación muy activo. Sin embargo, aunque la aritmética en punto fijo normalmente se ajusta muy bien a las características de los diseños basados en electrónica digital, algunos autores centran su trabajo en reducir área mediante el diseño de circuitos de punto flotante personalizados. Por ejemplo, basándose en la idea de que en un circuito algunos nodos son más sensibles que otros al proceso de cuantización de bits, [GAF02] utiliza técnicas de minimización diferencial para encontrar esos nodos y diseñar circuitos flotantes personalizados en cada nodo acorde a su sensibilidad. A medio camino entre punto flotante y punto fijo tenemos la aritmética dual. Tal y como se describe en [CHU04], esta aproximación presenta un rango dinámico mayor que la aritmética en punto fijo manteniendo un consumo de recursos limitado.

Debemos considerar que, además del compromiso entre consumo de recursos y precisión, el análisis del diseño de sistemas con profundidades de bits limitadas tiene otros efectos. Tal y como es mostrado en [CON03], la profundidad de bits tiene un efecto decisivo en el consumo de potencia del sistema. Los bits menos significativos tienden a conmutar su estado con mucha frecuencia y ello provoca un gran consumo de potencia en los circuitos digitales. Una utilización de registros de datos con número de bits demasiado alto puede no reportar beneficio en términos de precisión (esto depende de las especificaciones) y a la vez ser el origen de un gran consumo de recursos y potencia, lo que es especialmente importante para la migración de FPGAs a circuitos VLSI.

Desgraciadamente, aunque el número de contribuciones que pretenden solventar este problema es considerablemente grande, las mencionadas aportaciones sólo solventan parcialmente el problema. La mayoría de los métodos propuestos han sido diseñados para el estudio de circuitos concretos como filtros FIR o circuitos tipo unidades aritmético-lógicas (ALU). Sistemas completos usan etapas lineales y no lineales (como funciones trigonométricas) que requieren un análisis detallado de los sistemas debido a las múltiples posibilidades de solución. Además, estas herramientas podrían usar el cálculo de probabilidades para analizar el rango dinámico efectivo de los datos o incluir consideraciones de diseño tales como consumo de potencia o recursos requeridos que incrementarían su utilidad. El diseño completo de los sistemas requeriría que, además de las anteriores capacidades, la herramienta fuese capaz de dividir los algoritmos en subetapas más sencillas, analizar el tipo de aritmética óptima para cada una de ellas y el número de bits para representar los datos. Las herramientas presentadas, aunque en esa línea, están aún lejos de conseguir estas funcionalidades.

Nosotros hemos desarrollado una herramienta semi-automática basada en librerías de MATLAB para conversión de sistemas software con datos en punto flotante a aritméticas mas adecuadas para circuitos digitales. Nuestro sistema requiere una especificación detallada de las subetapas del sistema y definir los rangos y tipos de representación de las distintas variables. A partir de esta entrada, la herramienta realiza un extensivo análisis de las diferentes alternativas de diseño y genera las tablas de ruido de cuantización basándonos en la comparación con los resultados software en doble precisión. Podemos usar la *relación entre la energía de la señal y el ruido de cuantización* (SQNR) como hemos hecho en el Capítulo 5 pero, la herramienta es flexible y permite otras medidas de error. Por ejemplo podemos usar bancos de pruebas de imágenes con propiedades conocidas y usar el error de la medida como referencia (siempre comparando con los resultados del sistema software). Por ejemplo la estimación del error de la orientación o el error angular para el flujo óptico. Ello permite incluir los errores de cuantización y la precisión del modelo en el proceso de especificación del sistema.

Hemos llamado a nuestro software: *MCode for DSP Analyzer* (analizador de código MATLAB para procesamiento digital de las señales). Consiste en una serie de biblioteca de funciones y archivos de ejecución de órdenes que permiten el uso de aritméticas en punto fijo y en punto flotante personalizadas. Además, hemos incluido funciones para evaluar la degradación de los modelos debido a la cuantización de bits usando la SQNR, el error absoluto medio, el error relativo medio, etc.., teniendo especial cuidado en funciones periódicas como las trigonométricas en las que se necesita funciones de error especiales.

El MCode permite, de manera iterativa, explorar la sensibilidad al proceso de cuantización de las diferentes subetapas así como evaluar la necesidad de representaciones en punto flotante o en punto fijo personalizadas. Como mostramos en el Capítulo 6, si el rango dinámico de las variables es muy alto, una representación usando aritmética en punto fijo puede llegar a consumir más recursos que una usando flotantes por lo que ambas alternativas son tenidas en cuenta por nuestra herramienta.

Una de las diferencias principales de nuestro método es que el proceso de estimación de precisión admite la inclusión de umbrales de confianza en la estimación de los resultados. El cómputo de la información existente en las imágenes se basa en modelos de visión por computador cuyos resultados son aproximados y por tanto un valor exacto (como si de una calculadora se tratase) no es posible. Es por ello que las propias estimaciones tienen cierto margen de error y, aunque el ruido de cuantización debe siempre mantenerse por debajo de estos limites, no tiene sentido que la precisión aritmética del sistema supere los límites del modelo mismo. Si tenemos esto en cuenta, el uso de medidas de error como la SQNR puede incluir de manera sencilla las estimaciones de fiabilidad en los errores del modelo, pesando de esta manera los errores de cuantización acorde a la fiabilidad del modelo en esa medida. Esta técnica permite optimizar las longitudes de bits en las etapas del sistema a la vez que minimizar el consumo de recursos al eliminar lógica superflua. Esta modificación es una importante característica diferenciadora con otras herramientas. Gracias a que en el campo de aplicación (la visión por computador) las estimaciones de las medidas incluyen umbrales de confianza, nosotros hemos podido utilizar esta información en el proceso de optimización de bits, permitiendo el diseño de arquitecturas muy eficientes.

Como principal limitación de esta herramienta hemos de mencionar que el proceso a realizar no es suficientemente automático y es necesario un buen conocimiento de aquella. Además, la versión actual realiza una búsqueda completa en el espacio de las soluciones de trabajo lo cual produce una alta carga computacional. Ello implica por parte usuario la restricción sobre el número de bits de los datos de algunas etapas para reducir la dimensionalidad del problema. Como trabajo futuro trataremos de automatizar el proceso de diseño, incluyendo técnicas de búsqueda de soluciones basadas en técnicas de inteligencia artificial que permitan una exploración eficiente del espacio de las posibles soluciones. Además, pretendemos incluir en las funciones de coste información sobre el consumo de recursos de las diferentes representaciones, consumo de memoria, de potencia, etc, de manera que el problema pueda ser formulado como un problema de optimización multi-modal.

1.7.2. Requisitos y especificaciones de los sistemas hardware

En nuestro proceso de diseño hemos intentado el uso de aritmética en punto fijo ya que nuestros dispositivos de procesamiento son FPGAs y como hemos comentado en la sección anterior, esta aritmética consume menos recursos. Para conseguir diseños óptimos, debemos analizar la viabilidad de cada modelo de visión, así como la potencia y los requisitos de área de la misma. Puesto que el número de variables a manejar es muy alto, herramientas como el *MCode for DSP analyzer* simplifican este proceso ya que proporcionan la información de la sensibilidad de las diferentes etapas de los modelos frente el proceso de cuantización, facilitan las profundidades de bits, métodos de escalado, redondeo y tipos de aritmética que mejor se ajustan a nuestras especificaciones.

El estilo de codificación que usemos para describir nuestros circuitos hardware debe ser parametrizable para permitir la exploración de las diferentes alternativas de diseño (tipo de representación y profundidad de bits de los elementos del camino de datos). Puesto que las arquitecturas utilizadas se basan en descripciones algorítmicas de alto nivel, no son adecuadas codificaciones basadas en descripciones RTL usando por ejemplo VHDL o Verilog. Aplicaciones como *Catapult C Synthesis* de Mentor [MEN06] admiten la especificación de hardware con este nivel de abstracción pero difícilmente permiten el control de bajo nivel que en ocasiones necesitamos. Para sistemas de procesamiento de señales (DSPs) existen herramientas basadas en conexionado de bloques como *System Generator for DSP* de Xilinx [XIL06a], *PixelStreams* de Celoxica [CEL06c], el *DSP Builder* de Altera [ALT06] o el *Codesimulink* [COD06] desarrollado en el Politecnico de Turín. Estas herramientas se basan en esquemas de bloques como los utilizados por Simulink que ayudan a simplificar el proceso de diseño del hardware, generando código VHDL o Verilog que después es transladado a la tecnología con las herramientas propietarias de los diferentes fabricantes.

Los objetivos de nuestro sistema vienen propuestos por los proyectos ECOVISION [ECO06] y DRIVSCO [DRI06], donde se requiere alta potencia de cómputo (como referencia más de 25 cuadros/s para 2 cámaras y resoluciones de 1000x1000 píxeles). Es por ello que descripciones de nuestros sistemas con alto grado de abstracción difícilmente alcanzan los objetivos de diseño. Por otra parte, el diseño a bajo nivel, por ejemplo RTL, consume un tiempo muy considerable (aunque permite el diseño eficiente de los circuitos) que no podemos permitir en el entorno de los mencionados proyectos. Además, el estudio de las diferentes alternativas de diseño requiere repetir la síntesis del sistema modificando los distintos parámetros en numerosas ocasiones, lo que consume un gran tiempo de cómputo. Es por ello que nuestro trabajo resultan muy atractivas herramientas de síntesis en las que puedan programarse ejecuciones por lotes (serie) para el análisis del espacio de soluciones son muy atractivas.

Basándonos en estos requisitos hemos elegido el sintetizador DK Design Suite de la compañía Celoxica [CEL06b]. El lenguaje de especificación utilizado es el Handel-C [CEL06c]. Este lenguaje es una solución intermedia que permite descripciones con un grado relativamente alto de abstracción pero que, caso de requerirse, permite definir etapas de nuestro sistema a nivel RTL. Además, el motor de síntesis proporciona buenos resultados como es mostrado en [ORT06b] y la herramienta integra simulación funcional de alto nivel muy útil para visualizar los resultados del procesamiento de imágenes. La herramienta genera salida en código *Edif* a partir del cual las herramientas del fabricante generan el fichero de programación de la FPGA; es decir, hacen la traslación de la descripción del sistema a la tecnología.

Nuestras especificaciones de sistema requieren un grado alto de paralelismo que se adapte al sistema descrito y además el uso de arquitecturas fuertemente segmentadas para cumplir nuestros objetivos. Para ello debemos ser capaces de producir una estimación por cada ciclo de reloj. Por ejemplo, si nuestro sistema estereo es capaz de funcionar a 50 MHz, debemos ser capaces de alcanzar 50 millones de estimaciones por segundo de disparidades. En esta situación difícilmente podemos intentar la compartición de recursos porque los circuitos utilizados requieren de un paralelismo masivo para alcanzar nuestras especificaciones (pese a ello hemos analizado cómo esto puede llevarse a cabo en el Capítulo 5 para un dispositivo de estimación de movimiento). Este efecto ha sido parcialmente compensado gracias al análisis detallado de la profundidad de bits de las diferentes etapas que hemos llevado a cabo para los diferentes sistemas. Además, la gran cantidad de recursos que actualmente proporcionan los dispositivos de hardware reconfigurable permite que nuestros diseños, aunque masivamente paralelos, mantengan disponible una gran cantidad de recursos del sistema. Por último debemos tener en cuenta que las técnicas usadas, en especial el diseño de cauces finamente segmentados, tienen otras ventajas aparte del incremento de la potencia de cómputo. Como se muestra en [SHE92], [SUT03], el consumo de potencia en dispositivos FPGA se reduce gracias a la segmentación del cauce ya que las transiciones espureas de los niveles lógicos, responsables de hasta el 70% del consumo de potencia de los dispositivos basados en lógica reconfigurable, se reducen significativamente. Con ello vemos que aunque el área del sistema aumente, por medio de la segmentación de cauce podemos disminuir la potencia total de nuestros sistemas.

Nuestra metodología de diseño se beneficia de una fina segmentación del cauce más un análisis detallado de la profundidad de bits y tipo de representación de las distintas etapas. Ambos métodos han sido descritos como técnicas efectivas para reducir el consumo de potencia en dispositivos FPGAs. Como trabajo futuro pretendemos cuantificar estos efectos y analizar de qué manera el diseño de sistemas complejos finamente segmentados ayuda a disminuir la potencia consumida.

1.8. Contenidos de la tesis

El resto de la presente memoria, que describe el trabajo realizado en esta tesis, ha sido estructurada en los diferentes apartados y capítulos que se indican a continuación:

I. Modelos de visión por computador

- *Capítulo 2: Técnicas de procesamiento de imágenes para estimación de rasgos locales: fase, energía y orientación.* En este capítulo describimos las diferentes aproximaciones basadas en filtros en cuadratura que han sido comúnmente utilizadas en el campo de la visión por computador. Con ellos podemos extraer la información local relativa a fase, energía y orientación de la imagen. También revisamos las diferentes técnicas existentes de interpolación entre filtros orientados que permite estimar la fase y energía presente para cualquier orientación [FRE91], [HAG92], [FEL02a], [NES98].

- *Capítulo 3: Modelos de estimación de movimiento*. Aquí describimos los principios en los que se basan los diferentes métodos de cómputo de flujo óptico. Comparamos los compromisos entre precisión y eficiencia de estos modelos y concluimos que la aproximación de Lucas y Kanade [LUC81] es adecuada para nuestro sistema. Además, analizamos las modificaciones propuestas por Brandt [BRA97], cuyo modelo ha sido la base del diseño propuesto en la Sección 6.3.
- *Capítulo 4. Modelos de visión estéreo*. En este capítulo revisamos brevemente la viabilidad y precisión de los métodos de estimación de disparidad presentes en la literatura. Ello nos permite destacar una aproximación basada en fase [SOL01] que es utilizada en el Capítulo 7 para el diseño nuestro sistema estéreo.

II. Arquitecturas de procesamiento eficientes. Diseño hardware y evaluación de prestaciones.

- *Capítulo 5: Arquitectura hardware para cómputo de fase, energía y orientación*. Diseño de un banco de filtros orientable. En este capítulo proponemos una arquitectura eficiente para extraer esas propiedades locales de la imagen. También evaluamos cuantitativamente la degradación del modelo debido al proceso de cuantificación de bits y el rendimiento final del sistema desarrollado.
- *Capítulo 6: Procesamiento de movimiento: Diseño hardware de una arquitectura de alto rendimiento*. En este capítulo proponemos un diseño eficiente basado en las modificaciones del modelo de Lucas y Kanade [LUC81]. Evaluamos las distintas aproximaciones diseñadas basadas en este método, con diferentes compromisos entre precisión y consumo de recursos. Cabe destacar la versión masivamente paralela con un cauce finamente segmentado cuya potencia de cómputo es superior en más de un orden de magnitud a cualquier otro sistema descrito en la literatura hasta ahora. En este capítulo hemos realizado también un considerable esfuerzo en la evaluación de las arquitecturas. Los diferentes sistemas diseñados han sido evaluados usando un banco de pruebas mediante secuencias sintéticas de mapa de movimiento conocido. Ello nos ha permitido cuantificar la degradación del sistema debido al uso de un número restringido de bits y aritmética en punto fijo.

- *Capítulo 8: Arquitectura de estimación de estéreo de altas prestaciones*. Aquí describimos un diseño eficiente que proponemos para el cómputo de estéreo basado en un modelo de fase. Discutimos el consumo de recursos de diferentes aproximaciones que usan diferentes escalas espaciales y finalmente evaluamos la pérdida de prestaciones debido a la limitada profundidad de bits disponible en las distintas etapas de la arquitectura.
- *Capítulo 8: Ejemplo de aplicación: Sistema de ayuda de cambio de carril basado en movimiento para el seguimiento de vehículos*. La arquitectura de estimación de movimiento descrita en los capítulos anteriores es utilizada aquí conjuntamente con un sistema de seguimiento para la detección de adelantamientos de vehículos durante la conducción.

III. Discusión y conclusiones.

- *Capítulo 9: Conclusiones.* Es un resumen de las contribuciones originales de este trabajo.

Chapter 1
Introduction

We briefly explain the motivation and goals of this work. We frame the main contributions of this thesis in the context of a complete computer vision system, emphasizing which parts of this complete vision system have been developed in specific hardware and why. This work has been developed in the framework of two European projects. We briefly mention the purpouses of this work matched with the more general aims of the research projects. Finally, we summarize schematically the contents of the thesis.

1.1. Motivation

After many years of research in the field of computer vision we are still far from understanding how the human visual system works. Many research groups have been focused on the simulation of specific cortical processing tasks (motion computation, stereo computation, colour, etc). These simulations are very time consuming and that has forced many researchers to develop simplified models.

The main contribution of this work is the implementation of bio-inspired real-time vision processing datapaths. This is supported by several grounds:

a. *Real-time processing for embodied vision experiments.* Nowadays, it has become clear that simple off-line simulations are not enough to understand the way that different tasks are concurrently performed in the visual cortex. Furthermore, there is a strong working hypothesis called “embodiment concept” that states any realistic simulation of a biologically inspired processing system should be tested in the framework of a certain task. The way that this task is achieved can be used to validate the different parts in which is based the success of the system. The embodiment concept is based on the hypothesis that biology has developed the impressively smart systems in nature through evolution trying to optimize certain tasks that improve the individual survival and specie perpetuation.

b. *Active vision.* The perception process is active. It combines sensori-motor capabilities in an integrative manner. Not only haptics but also vision is believed to be an active process in which intentional primitives drive certain mechanisms (such as fixation, smooth pursuing for stabilization, etc.) that enhance the accuracy of the system. Furthermore, it is also believed that attention is a useful mechanism in order to achieve very high performance with constrained processing resources. But active perception processes can only be studied in the framework a perception-action closed-loops. This specifically requires real-time processing and represents a strong motivation for developing high performance vision processing architectures.

c. *Understanding by building.* From an engineering point of view, we only fully understand certain mechanisms if we are able to implement them. In the framework of computer vision systems, as engineers, trying to build efficient image processing architectures based on biological vision systems is a very interesting approach since we face the same limitations as nature also with constrained processing resources. According to the “neuromorphic engineering” paradigm we adopt an opportunistic attitude in which we try to emulate schemes that seem to be efficient in the biological systems and we avoid other features that are more intrinsically related with the tissues in which they are based. Furthermore, not being limited by some biological restrictions (such as power or conductance and switching capability of neuron wiring and connections) we can take full advantage of certain outstanding characteristics of electrical technology, such as high communication bandwidth, high speed state switching, etc.

d. *Smart vision systems in real world applications.* Real-time processing of local features, motion and stereo is interesting for a wide range of applications in real world scenarios. Therefore, the implementation of high-performance computing architectures has an interest in itself for solving real world problems.

1.2. Structuring a vision machine

Visual perception is a complex process that transforms (translates) signals (images) into cognitive information. Although there is no general agreement about how to structure such a complex system, for the sake of clarity, we can split vision in different layers dealing with information at diverse abstraction levels (as illustrated in Figure 1.1):

1. **Early cognitive vision.** (Low level vision).
    - 1.1. *Basic spatio-temporal sampling.* This stage is composed of spatio-temporal filters. Biological systems use cells whose receptive fields project onto the retinas. The vision models use a set of basic spatio-temporal filters of different size and temporal characteristics (corresponding to spatio-temporal derivatives of different orders).
    - 1.2. *Stereo vision.* Combination of filter responses from the two retinas to extract *depth* estimations.
    - 1.3. *Motion processing.* Combination of filter responses of the same retina in order to obtain *motion* estimations.

1.4. *Color processing.* Combination of different monochrome filters to robustly and efficiently encode natural colors with opponent cell responses.

1.5. *Structure extraction.* Integration of outputs of different spatial filters to enhance image areas that encode a significant quantity of image structure.

2. **Middle cognitive vision.** Integration mechanism that allows efficiently and constructively combining different visual modalities (motion, stereo, orientation, etc). At this stage different information channels converge leading to multimodal entities. Different mechanisms can be applied at this stage to extract only robust information discarding outliers (coplanar integration, collinear criteria, etc).
   Also in middle cognitive vision can be located segmentation mechanisms that integrate multimodal entities corresponding to real-world grounds (sources, or common cause effects), such as IMOs (Independent Moving Objects), egomotion, heading, etc. These segmentation mechanisms lead to a scene in which different elements are identified as object candidates or common cause effects (such as egomotion) and linked with specific multimodal characterizations that can be static or driven by certain structure constraints (such as the rigid object motion).
3. **High level vision.** This is a very high processing stage in which scene interpretation is performed through more specific sub-tasks, such as object recognition, effects prediction, comparison with already perceived scenarios, etc.

It is important to note that early cognitive vision is inherently dense, i.e. it requires processing of each pixel in the scene, while higher level tasks deal with discrete entities (multimodal entities, objects, common cause effects such as egomotion, etc). Current computer vision systems are not able to extract in real time the low level vision primitives (inherently dense) and therefore they are already limited at this processing stage.

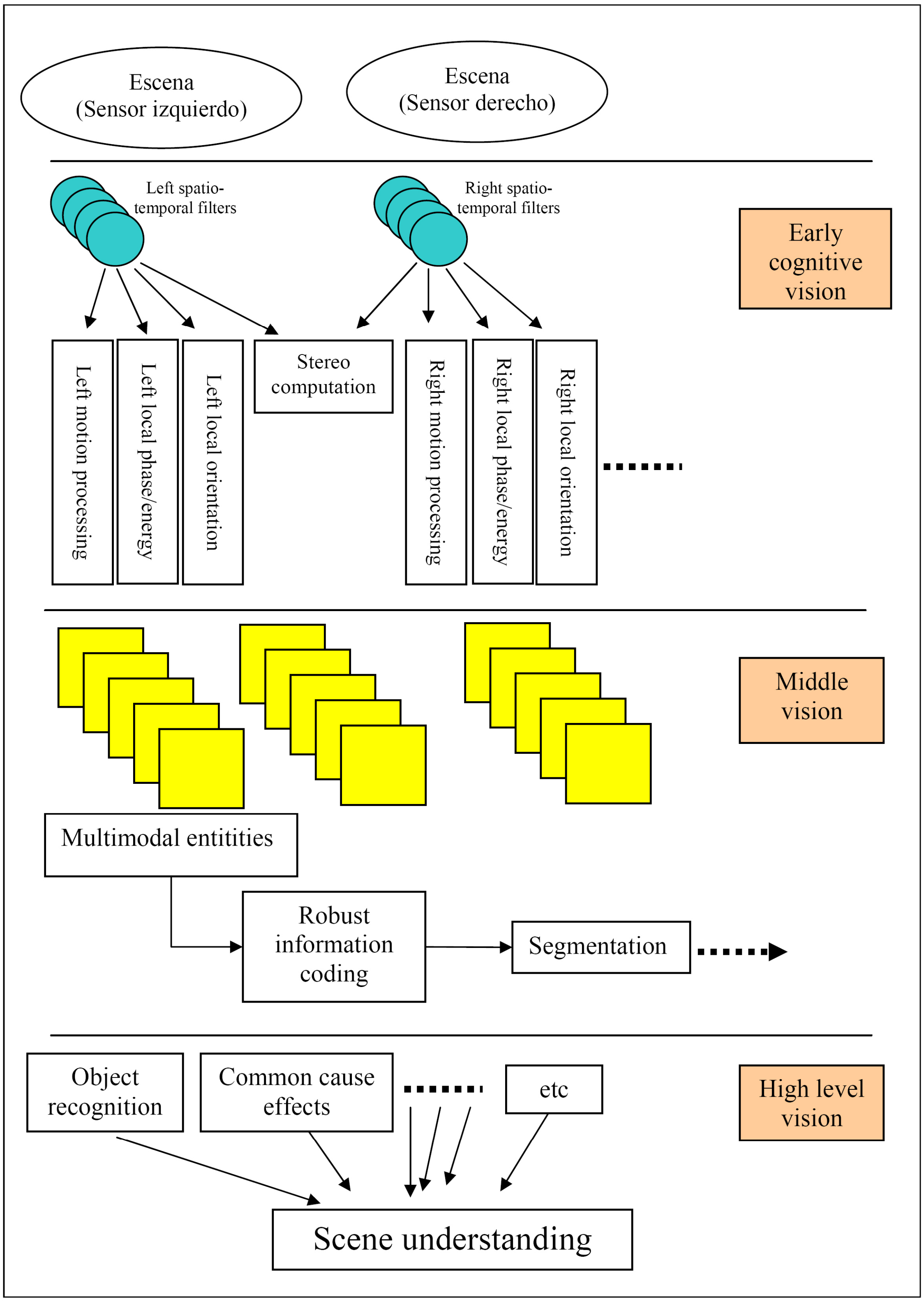


Figure 1.1. Schematic of the vision system structure.

One of the main motivations for the work of this Thesis is the implementation of a high performance vision processing architecture capable of computing in real-time the low vision primitives. This is of specific interest because higher vision levels (that are inherently discrete) suit better computing platforms based on a single (and powerful) processor while the early cognitive vision highly benefits of a fine-grained parallelism. This will be illustrated in chapters 5 to 7 in which we design highly parallel processing architectures based on the regular data flow processed at the very early vision stages. More concretely, in this work we focus on the implementation of high performance computing architectures for motion, stereo and local structure (orientation, phase and energy) analysis.

1.3. From biological models to real-time hardware systems

Engineering processing architectures designed for tasks that biological systems solve with impressive ease can benefit considerably by mimicking computing strategies developed by nature over long periods of evolution. But the adaptation of such techniques is not straightforward, since the physical principles upon which biological tissues are based are very different from those characteristically used in electronic technology. Furthermore, biological and electrical *"technologies"* face different restrictions which are overcome by resorting to different strategies.

Nevertheless, an "opportunistic attitude" which takes the key-functional principles that contribute to the outstanding performance of biological systems and also uses technology-motivated computing techniques to adapt those computing primitives must be of considerable interest. This opportunistic approach should on its own merits provide a suitable solution to the individual task in question, whilst also helping to identify and characterize the functional principles that support the high performance observed in biological systems. For example, biological systems widely use massive parallel processing to overcome the slow chemical-based principles that support most of the computing and transmission principles of neurons. On the other hand, whereas electrical technology allows faster devices (more than three orders of magnitude), the connectivity allowed by current silicon technology is restricted to 2-D patterns and so this massive parallelism becomes impossible to adopt in electronic devices.

To be able to adopt biologically inspired processing schemes we use a time-slicing technique and we develop very fast computing units that abstract the functional principles upon which the emulated scheme is based. In this way, for instance, we can process in stereo the disparity between two images several times (with different shifts and spatial scales) and thus obtain multiple disparity estimations which in a biological system would have been extracted by different populations of neurons. We then integrate all these estimations constructively to achieve the best performance.

We illustrate here several examples of such approaches. We have developed a system for processing local image features, optical flow and disparity estimation that are able to extract all these modalities at frame rates with large image resolution. This allows the exploration of integration schemes in the framework of real-time processing tasks. For example, this fast computation allows neural population coding based on the set of estimations obtained on multiple time slots.

Conventionally, parallel processing of different circuits is limited due to the limited transmission bandwidth. Especially significant are the constraints deriving from the external memory access; which is usually one of the important bottlenecks for FPGA processing capability, but due to the on-chip system management of external and internal memory, and since the described architecture consists of one single processing unit, with the whole system implemented on the same device (as a *System-on-a-Chip,* SoC), the access control is carefully designed and this bandwidth limitation is overcome. Furthermore, the proposed scheme is scalable; since we are plenty of available computing resources on the same chip and depending on the image features selected, two or more processing units can be used, if further parallelism is needed, to increase the frame-rate, extract more estimations to enlarge the population or increase the spatial resolution.

1.4. Framework of this work

This work has been developed in the framework of two European Projects:

- ECOVISION: Artificial Vision Systems based on early cognitive cortical processing (IST-2001-32114), (01-01-2002 till 30-12-2004), [ECO06].
- DRIVSCO: Learning to emulate perception action cycles in a driving school scenario (016276-2) (01-02-2006 till 31-01-2009) , [DRI06].

In both projects, one of the main goals is the implementation of real-time computer vision systems, in order to open the door to all the potential applications of such schemes. Furthermore, in both projects participates an industrial partner related

with the automobile industry that has defined specific potential applications in which such vision systems would be of great interest. In fact, a significant effort has been done along this work to validate this technology (real-time vision system in FPGA) for specific tasks in real world scenarios (for instance, car tracking in overtaking scenarios). DRIVSCO is an ongoing project that aims the implementation and validation of vision-based learning strategies to assist driving in night scenarios (see for instance Chapter 8). This requires motion and stereo in real-time processed in specific hardware due to its high computational load.
The vision circuits presented in this work are currently used in the European Consortium (DRIVSCO) to explore information integration mechanisms in middle and high level vision. Furthermore, this technology is of crucial importance to evaluate perception-action close loops.

1.5. A complete work with a good research vs. development trade off

The work described in this document presents the results of four years dedicated to analyze different vision primitives and their interrelation. There is not a very well defined border line to clearly distinguish between development and research tasks. In fact any research action requires of development tasks in the experiments and results extraction processes. More concretely, in our case the system design and its performance evaluation requires considerable development workloads.

The PhD work has well defined research objectives (working process represented at figure 1.2):

- Evaluation of the feasibility of specific circuits to extract vision modalities (local image features, motion and stereo) in real-time.
- Evaluation of the accuracy vs. efficiency of the different approaches. There were studies about different motion and stereo schemes and we have used these results. On the other hand, for the local image features extraction (phase, orientation and energy) we have carried out a serious study in order to be able to

choose a proper spatial-filter approach. To the best of our knowledge no comparative study about the different alternatives has been done before.

- Implementation of high performance datapaths for the different visual modalities:
    - Exploration of circuit design strategies for regular data flow models.
    - Implementation of fine grain pipelined and superscalar datapaths.
    - Evaluation of the performance of the proposed circuits.

The achievement of these goals has required a high development load. There are relevant topics that have been faced during this work. For instance, instead of proposing new algorithms for visual modalities we have implemented specific circuits to process them efficiently. But this requires the evaluation of the aimed model, its simplification and the evaluation of the accuracy with constrained computational circuits (fixed point arithmetic and restricted bit-width). In fact, the implementation can be considered a new model whose performance and accuracy needs to be properly evaluated comparing it with other approaches described in the literature. The adopted working methodology has forced to use benchmarking sequences or images (synthetic and real ones) for the evaluation of the performance and accuracy of the designed systems.

Summarizing, the field of high performance computing architectures requires high development loads not only towards the specific design to the proposed architecture but also when evaluating the obtained performance with benchmark sequences or images. Nevertheless, there are specific issues that represent the main research trends of this work:

d. The developed vision circuits can be considered as models that behave similarly to their software versions but due to their precision constraints lead to completely different computation speed versus accuracy trade off. Therefore, the implementations need to be evaluated and compared with other models and implementations as has been done in this dissertation.

e. The extensive use of deep pipelined superscalar computing architectures for the design of the different models is a quite new and innovative strategy seldom adopted by other authors. Furthermore, this strategy is the one that allows an efficient use of the inherent parallelism of the FPGA devices in order to obtain outstanding performance rates.

These two topics have facilitated the wide publication of the results of the research work as it is pointed out in the discussion chapter.

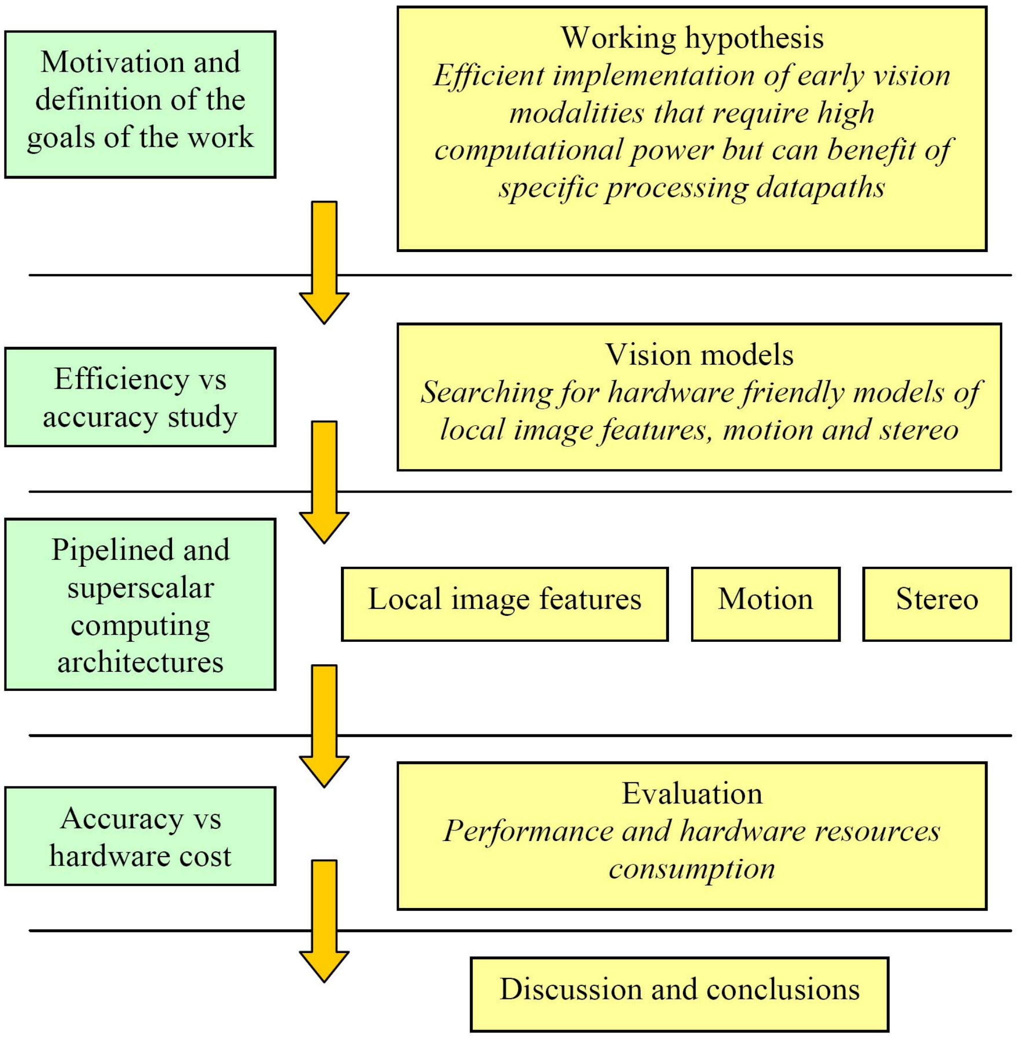

Motivation and definition of the goals of the work
Working hypothesis
Efficient implementation of early vision modalities that require high computational power but can benefit of specific processing datapaths
Efficiency vs accuracy study
Vision models
Searching for hardware friendly models of local image features, motion and stereo
Pipelined and superscalar computing architectures
Local image features
Motion
Stereo
Accuracy vs hardware cost
Evaluation
Performance and hardware resources consumption
Discussion and conclusions

Figure 1.2. Schematic of the work described in the Thesis.

1.6. Circuit definition strategy

The main contribution of this work can be considered the designed system and its performance evaluation. But the design of a complex system can be done at different abstraction levels and with different definition tools. Given the high complexity of the aim models we have used Handel-C [CEL06c] as hardware description language (HDL) because it allows the definition of the computing architecture at different levels of abstraction without paying a high cost when comparing it with other circuit description languages [ORT06b]. We have defined the circuits at a register transfer level (RTL) but with the abstractions provided by Handel-C (for instance the baseline state machine that allows efficiently testing different parallel processing schemes). Furthermore, the chosen description language (similar to standard C) has highly facilitated the implementation of vision models that are usually described as algorithms.

For the implementation of specific processing architectures we have used reconfigurable hardware (FPGA) devices. We have not used their dynamic reconfiguration capability to optimize the computation power in working time but this technology has highly facilitated the definition and test of different implementations. Furthermore, following Moore's law this technology keeps advancing in terms of number of resources that are allocated on a single chip and also because the devices already include highly optimized circuits (such as embedded memory resources, multipliers or high bandwidth I/O channels) of interest for a wide range of applications.

In this work, instead of trying to take full advantage of the available computing resources of a given architecture (such as a general purpose single processor), we have defined specific purpose computing architectures for specific tasks and we have shown that they clearly outperform the approaches based on general purpose processors. This is an outstanding result that cannot be obtained without the intensive use of the parallelism available at FPGA devices. In fact, it was not clear at the beginning of the work that the designed systems would outperform general purpose architectures (such as conventional processors) that run at clock frequencies almost two orders of magnitude higher than our circuits. Nevertheless, by designing specific purpose processing architectures with general purpose digital circuitry we can take full advantage of the continuous advances of digital technology instead of being stacked with a concrete computing architecture. All the circuits presented in this work can be recompiled (with only a moderate adaptation workload) to the future FPGA devices or low cost chips in order to address different application fields.

1.7 Hardware design tools and methodologies

The process of implementing on hardware a high level computer vision model requires several challenges to achieve success. The model has to be properly modified to be hardware friendly. Arithmetic representation needs to be revised (floating point representation used on software models do not fit well embedded devices) and the resulting system should be described on a proper way to achieve a good designing time versus hardware performance/consumption trade-off. After the whole process, the system requires to be evaluated to determine the final accuracy which can be significantly different of the original software models. This is specifically addressed for each developed circuit in Chapters 5, 6 and 7.

1.7.1. Motivation and tools for bit-width analysis

High performance embedded systems use specific computational resources for each pipelined processing stage. Therefore, it is crucial to use low cost computational circuits whenever is possible. Floating point arithmetic demand large hardware resources and thus, designers usually try to use fixed point arithmetic. The analysis of the system degradation (when comparing floating point with fixed point representations with a target precision) requires splitting the algorithm into multiple substages with limited and controlled bit-widths. As commented in [MAL06] the strategies can be roughly categorized into two groups. The first one is an analytical approach used by algorithm developers who analyze finite word length effects due to fixed-point arithmetic [CHA95], [GRA98]. The other approach is based on bit-true simulation techniques used by hardware designers [KED98].

There has been some work in the recent literature on automated compiler techniques for conversion of floating point representations to fixed point representations [SYN06a],[SYN06b]. The BITWISE compiler [STE00] determines the precision of all inputs, intermediate and output signals in a synthesized hardware design from a C program description. The MATCH compiler [NAY01] develops precision and error analysis techniques for MATLAB programs. Synopsys has a commercial tool called the Cocentric Fixed-Point Designer [SYN06b], which automatically converts floating point computations to fixed point within a C compilation framework. However, the code generated is not synthesizable. Constantinides [CON03] has developed a design tool to tackle both linear and nonlinear designs. Chang et al. [CHA02] have developed a tool called PRECIS for precision analysis in MATLAB. An algorithm for automating the conversion of floating point MATLAB to fixed point MATLAB was presented in [BAN03] using the AccelFPGA compiler. Their approach needs the default precision of variables and constants specified by the user which the compiler is unable to infer. In 2004, Roy et al. [ROY04] proposed automated algorithms to convert floating-point MATLAB programs into fixed point MATLAB programs using input profiling. The work is used to trade-off area and performance with respect to the quantization error.

As can be deduced from the previous discussion, proper bit-width selection is an active research area. Nevertheless, though hardware friendly, fixed point arithmetic are not always the best choice. Some contributions try to reduce the resources required using floating point representation based on a custom bit-width. For instance, based on the idea that some circuit nodes have a higher sensitivity to quantization noise than others and using differentiation techniques to find these nodes, in [GAF02] it was presented a mathematical formulation to customize floating point representation at each circuit node. A mixed alternative is the utilization of Dual fixed point arithmetic. As commented in [CHU04], this approach is an intermediate solution between floating point representation and fixed point in terms of dynamic range and arithmetic precision with a more affordable hardware cost.

There are other side consequences of choosing a proper arithmetic representation that go beyond hardware resources consumption vs. accuracy trade-off. As mentioned in [CON03], a proper bit-width design has significant importance in terms of power consumption. Low significant bits tend to switch their state very frequently and this shall be avoided if it does not drive any information. Due to that, the smart elimination of low significant bits allows decreasing the frequency of meaningless bit switching, reducing the switching power which is important in embedded systems and also for the migration to VLSI devices.

Unfortunately, although there are a large number of significant contributions, this problem is still only partially solved. Most of the previous methods are designed for the analysis of well defined circuits' substages such as FIR filters or well defined arithmetic operations such as *Arithmetic-Logic Units* (ALU). Complex designs use linear and non linear arithmetic operations and require more extensive analysis. Statistics probability can be also included in the analysis to evaluate the effective dynamic range of the variables. Other considerations such as power consumption can be involved on the design decision process. The whole system analysis requires splitting the system on simple substages, studying the required arithmetic representation at each of them and determining their bit-width which is still far beyond the possibilities of current tools.

Similarly to the methodology presented in [ROY04] we have developed a semi-automatic tool for bit-width analysis on the MATLAB environment. Our methods require manually dividing the model on substages and fixing the data range and arithmetic type of the main variables. Then, the tools make an extensive analysis of the different alternatives and generate the quantization noise data based on the comparison with the software approach using double floating point representation. As in Chapter 5, we can use the SQNR to evaluate the software versus hardware model degradation. Nevertheless, our method is very flexible and the error metrics can be defined in different ways, as the similarity to some real values. For example, for orientation estimation the error metric can be simply defined as the difference between the computed angle and the real angle of each pixel of a synthetic image or for optical flow the angular error measure can be defined as the difference between the computed and the real motion of each point. In these cases, synthetic sequences with known ground truth are required and the quantization error is included in the model error.

Our software is called *MCode for DSP Analyzer*. It consists on a set of libraries that allow making computations with fixed point or floating point data representation with customized bit-widths. These libraries also include functions to evaluate the degradation based on the SQNR as well as other common error metrics as the maximum error value, relative error, taking special attention to periodic functions such as the trigonometric ones. On an iterative way, we can explore which substages have higher quantization sensibility or which of them do not benefit of large bit-widths. We can also test the utilization of customized floating point representations and evaluate how well they fit the design requirements. As shown in Chapter 6, at some critical stages where the required data range is large, floating point representation becomes more hardware friendly than the equivalent fixed point data format.

An important difference with other approaches is the inclusion on the optimization process of some confidence information about the data computed. Our image features have been computed based on computer vision models which only represent approximate results. They are not analytical solutions of any equation and estimation errors are intrinsically assumed into the models. Therefore, these computations have some degree of uncertainty and, although quantization errors should be always kept below this level, there is no sense in increasing the accuracy further than this value. Keeping that in mind, we consider that SQNR is a good accuracy metric which easily can be combined with a confidence parameter of our vision features. This combination allows to measure quantization error only for reliable data outputs, which effectively allows reducing bit-widths at the different stages and the hardware area of our implementations. This is a significant difference from previous approximations that relay on our field of application (computer vision) and includes information about the model uncertainty on the bit-width analysis.

As drawback, our tools still require significant user expertise and are not automatic enough. At the present stage, the dimensionality of the problem makes unviable an extensive search of bit-widths configurations in the whole solutions space. This is the reason for manual data range introduction which is required to constraint the problem. Future work will try to make this process fully automatic improving this tool by using artificial intelligent methods to reduce the problem dimensionality. Furthermore, we also plan to include metrics for different valuable estimations such as hardware resources consumption, data-throughput, etc; which allows defining the problem as a multi-objective searching approach, maximizing accuracy and minimizing area and/or memory, power consumption, etc.

1.7.2. Hardware system specifications and tools

Taking into account our target technology (FPGAs) we have tried to use fixed point data representation with constrained bit-widths as described on the previous section. In order to evaluate the implementation feasibility, we need to focus on hardware resources and system performance as goals to optimize. The large number of parameters to consider in a hardware implementation makes necessary to constraint the problem to achieve a solution at a reasonable design time. The *MCode for DSP Analyzer* determine stages more sensible to quantization noise and provides the data bit-width relations, scaling methods, rounding techniques and other relations that effectively help the hardware design process.

Nevertheless, the parameters space (bit-width and arithmetic representation of each datapath element) exploration makes necessary a hardware coding style that allows full model parameters specification. The very high algorithm nature of the approaches discussed in this Thesis makes necessary a higher level of abstraction than standard RTL codes such as the VHDL or Verilog can achieve. Applications such as Catapult C Synthesis of Mentor [MEN06] are supposed to achieve this abstraction level but the low level system design is still not solved. For DSP design there are some box-connection based tools such as *System Generator for DSP* of Xilinx [XIL06a], *PixelStreams* of Celoxica [CEL06c], the *DSP Builder* of Altera [ALT06] or the *Codesimulink* [COD06] developed at the *Politecnico di Torino*. They are based on Simulink-like working methods to simplify the hardware system description. After boxes connection and parameterization, the system finally generates VHDL or Verilog code to be synthesized.

In our system a remarkable performance is required for the addressed applications on the framework of the projects ECOVISION [ECO06] and DRIVSCO [DRI06] (as reference, more than 25 fps with two cameras and images of resolution 1000x1000 pixels). This makes that very high level system specification tools hardly achieve the target specifications. The designing time is a very valuable factor since different visual modalities and systems are developed. We also need to consider the designing parameter exploration. For this task a HDL and a design tool capable to run on batch mode is a valuable option, specially taking into account that synthesis process is a very time consuming task.

We have chosen the DK synthesizer of Celoxica [CEL06b] that fits these requirements. The HDL is Handel-C [CEL06c] which allows easy description of algorithmic systems but the coding style also allows a RTL description to achieve high performance. The synthesis engine produces good results [ORT06b] and the functional simulation is integrated on the environment. DK (design environment) output is an *Edif* code which is the input to proprietary place and route tool.

The system specifications require a high parallelism system description and coding style which motivates the use of a fine grain pipelined architecture with parallelism growing across the different stages to achieve the maximum system throughput. Our goal is to obtain one pixel output per clock cycle (a computing system running at 50 MHz should be able to achieve 50 millions estimations per second). As consequence, the main drawback of this high performance requirement consists on high hardware resources utilization because resources sharing is not possible (though it has been explored in Chapter 5 in the context of an Optical flow processing architecture). Nevertheless, this is partially compensated by the optimized bit-width design methodology. Furthermore, the large hardware resources available on current FPGAs devices make possible the designed system and still leaving a large amount of resources on the same chip for other purposes (such as sensor and computer interfacing). Furthermore, the fine pipeline architecture produces other benefits than high performance. As shown in [SHE92], [SUT03], the power consumption of FPGA devices is reduced when large pipelines are utilized because it significantly reduces the circuit glitches. They can be responsible of up to 70% of the power consumption of this kind of devices. Therefore fine grain pipeline techniques effectively helps reducing power.

Our design methodology benefits from a carefully designed data bit-width and arithmetic in addition to the fine pipeline architecture. Both methods have been highlighted as effective techniques for power consumption reduction. Future work will try to quantify the benefits of these techniques analyzing also the effect of the fine pipelined datapaths on the power consumption issue.

1.8. Content of this thesis

This work has been structured in the following parts and chapters:

I. Computer Vision Models

- *Chapter 2: Image processing methods for computing the local image features: phase, orientation and ene*rgy. Here we describe the different quadrature filters commonly used on the literature for computing three basic local images features: orientation, phase and energy, furthermore we evaluate the interpolation methods to estimate the feature at the right orientation [FRE91], [HAG92], [FEL02a], [NES98].
- *Chapter 3: Motion processing models*. Here are described the principles in which different motion estimation models are based. We compare the accuracy versus efficiency of the different approaches and we conclude that the Lucas & Kanade (L&K) algorithm [LUC81] is a very good option. Furthermore, we choose a modified version of the original L&K approach [BRA97] whose hardware implementation is described in Chapter 6.
- *Chapter 4: Stereo vision processing models*. In this chapter we briefly review the different stereo models evaluating their accuracy and feasibility in specific hardware. The main objective of this study is to arrive at a specific well defined model in which we will focus in Chapter 7. We choose a phase-based hardware friendly model [SOL01].

II. Efficient processing architectures. Hardware implementation and performance evaluation.

- *Chapter 5: Hardware architecture for phase, orientation and energy computation*. Hardware implementation of a high performance steerable filter bank for phase, energy an orientation computation. In this part we specifically describe an efficient computing architecture to extract these signals. We also evaluate qualitatively the accuracy versus efficiency trade-off of the approach.
- *Chapter 6: Motion processing: Hardware implementation of a high performance computing architecture*. In this chapter we describe an efficient processing architecture for the modified L&K model. Different versions of the system are described and evaluated characterized with different accuracy versus hardware resources trade-offs. Particularly we present a superpipelined and superscalar processing architecture that outperforms any previous motion estimation system (described in the literature) by more than one order of magnitude. This chapter represents also a considerable effort in the evaluation of the presented processing architectures. For this purpose, we benchmark different approaches with sequences on known motion ground-truth in order to evaluate degradation of the model due to the use of fix point arithmetic with a restricted number of bits.
- *Chapter 7: High Performance stereo computing architecture*. This chapter focuses on the efficient implementation of a phase-based stereo model. We discuss the hardware cost of diverse approaches with different filter lengths. Finally, we evaluate qualitatively the accuracy loss due to the limited precision operations at different stages of the computing architecture.
- *Chapter 8: Application example: Lane change decision aid system based on motion-driven car tracking*. In this section we apply the optical flow computing architecture for detection of overtaking vehicles as application example.

III. Discussion and conclusions.

- Chapter 9: Conclusions. This is a summary of the main original contributions of this work.

Chapter 2
Image processing methods for computing the local image features: phase, orientation and energy

This chapter describes the different quadrature filters commonly used on the literature for computing three basic local images features: orientation, phase and magnitude. The methods based on Isotropic analytic filters (Monogenic Signals), Gabor filters and Gaussian derivatives are discussed in terms of accuracy and efficiency. Furthermore, for approaches where only a discrete number of oriented filters are presented, the interpolation methods to estimate the feature at the right orientation are evaluated. In this chapter we find that the second order Gaussian derivative is a good trade-off between accuracy and computing resources to be implemented on customized hardware.

2.1. Introduction

The analysis of image features such as colour, edges, corners, phase, orientation or contrast provides significant clues in the process of image understanding. They are used as base for higher level task such as object segmentation, object recognition, texture analysis, motion and stereo processing, image enhancement and restoration or special effects [GON02], [GRA95], [SON98].

We will focus on three basic image properties, phase, energy and orientation. They have been extensively used on computer vision [KRÜ02], [KRÜ04] and, as we will see, they can be obtained from the same set of image operations.

These features are low level primitives that can be extracted through convolution based operations with a set of spatial filters. In this chapter we describe the three more extended filter types used for this purpose and we make accuracy vs. efficiency study to define the best option at this low level stage (for extracting these low level features).

This chapter deeply focuses in the signal processing theory for computer vision. Very specific concepts are discussed and some previous knowledge on this material is required. Sections 2.1 to 2.3 are included as review and link to the discussed signal processing concepts, improving the chapter completeness. Nevertheless, our main contributions appear on Section 2.4 and depending on the reader interest, previous sections could be skipped. We encourage to the readers to focus on that section and just use previous sections for consulting further details.

2.1.1. Local orientation

The analysis of local orientation has received a considerable amount of attention in the literature over the past decade [GRA78], [BIG91], [PER92], [KAS87], [FRE91], [RAO91]. It is an early vision feature that encodes the geometric information of the image. The common assumption on computer vision is that sufficiently small image regions can be characterized as local one-dimensional signal, e.g., in terms of lines or edges. For natural images this assumption is usually correct except at specific points, e.g., corners, line junctions or crossings and textured regions. However, the size of the regions that have to be in order to appear as one-dimensional varies both between images and within an image. Also, in practice a local region is never exactly one-dimensional but can be approximated as such. Image regions which are in fact one-dimensional are also referred to as simple or intrinsic one-dimensional (i1D), [KRÜ03].

There are a wide variety of algorithms for the estimation of local orientation, with a wide range of applications from the simplest case of 2D orientation (which we are addressing here) up to the more complex cases of multiple simultaneous orientations and junction analysis [MIC94] or even multi-dimensional orientation [ADE85], [HAG92], [AND92], [WES94], where time can be taken into account.

Orientation has an ambiguity in the representation of local orientation coming from the 2 possible directions, for each orientation because it is a 180º periodic measurement, i.e. a line between two points has no given direction, but has a well-defined orientation, which can be defined in [0, π[or in [π,2 π[. That is to say, the two complex numbers $re^{i\theta}$ and $re^{i\theta+\pi}$ represent the same orientation. Averaging of these two vectors, however, will result in total cancellation. At first, it may seem that simply restricting the allowable orientation estimation values to a particular interval would eliminate this problem but this ambiguity can be solved in a proper way using the following methods:

1. The double angle representation: Proposed by G. H. Granlund in [GRA78], is to simply double the angle of each orientation estimate. While doubling the angle is unattractive for visualization purposes, mathematically it provides us with a meaningful representation for averaging, differentiation, and other related operations. The angle is of course halved for visualization purposes.
2. The tensor representation: it is a generalization defined for arbitrary dimensions of the image data [GRA95]. It applications include curvature estimation and tensor field controlled image and image sequence enhancement.

Related with these representations, a number of methods have been proposed for computing or estimating an orientation representation from image data [WIK06]. These include:

1. Quadrature filter based methods [FRE91], [KNU83], [HAG94].
2. Gradient based methods [KAS87].
3. The structure tensor [GRA95].
4. The Energy tensor [LAR05].
5. The Boundary tensor [KÖT06].
6. Local polynomial approximation [FAR99].

Nevertheless this classification is arbitrary and several of them could be considered belonging to different types (for example, the method of Haglund [HAG92] define a structure tensor based on quadrature filters and therefore can be classified as type 1 or 2.

From all of these approaches, concerning that our goal is the efficient hardware architecture design; we will focus on quadrature filter based methods, basically motivated by the following reasons:

1. They allow the computation of local orientation based on convolution with a set of kernels. This operation is hardware-friendly and can be efficiently implemented on digital hardware.
2. We can share the information coming from the filtered set of images to extract other valuable information, such as local phase and energy (see next section).
3. Quadrature-filter based methods can be readily extended to handle instances of multiple simultaneous orientations, as occuring at the intersection of lines and corners.

2.1.2. Phase and energy

Fourier transform of the image allows recovering the signal spectrum which can be used for enhancing or restoring the image. We assume that the signal is stationary (signals which are constant in their statistical parameters over time, e.g. sinewaves). If the signal is non-stationary, any abrupt change of the signal will be spread over the whole frequency axis and the spatial position of the discontinuity will be impossible to retain from the Fourier coefficients. The Fourier transform is apparently not sufficient for analyzing such signals. The *Short Time Fourier Transform*, or *Windowed Fourier transform*, is one way to modify the Fourier transform for better performance on non-stationary signals, allowing extending this concept for the characterization of local features. It has been widely used on bioinspired computer vision models [SIM98]. Using a bank of bandpass quadrature filters tuned to different orientations and spatial scales, the image can be convolved and, we can obtain a set of outputs for these filters. This filter bank should be designed to cover homogeneously the frequency domain as showed on Figure 2.1. On that way, the filter responses encode the frequency context of the image.

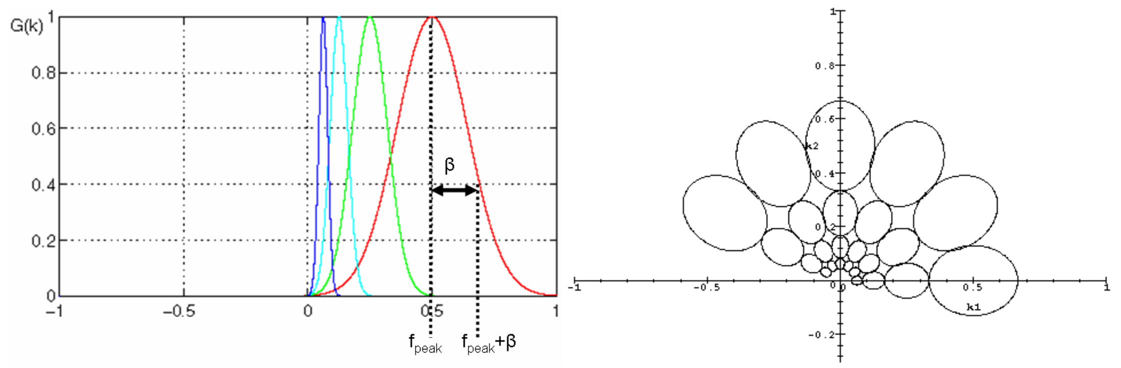


Figure 2.1. Bandpass filters covering different spatial frequencies and orientations (figures adapted from [GET06]. Left image represents the different filters spatial scales based on scaling by 2 of the main filter. The x-axe represents the normalized frequency values ($f/f_{Nyquist}$). Right image shows a polar representation of these scales across different orientations, using a logarithmic splitting of the frequency domain. Uniform coverage of the frequency domain allows properly decomposing the image signal on this domain and extracting multivalued local phase and energy information.

Quadrature filter is a complex filter that allows decomposing each output as phase and magnitude. If we note ***h****(x,*k*)* for a complex filter tuned to a spatial frequency k_0 then:

$$h(x;k_0) = c(x;k_0) + js(x;k_0) \tag{2.1}$$

Where c and s respectively represent the even and odd components of the filters, fulfilling the condition that the even and odd filters are Hilbert transforms of each other. The convolution with the image *I(x)* is expressed by equation (2.2):

$$I * h = \int I(\xi)h(x-\xi;k_0)d\xi = C(x) + jS(x) = \rho(x)e^{j\phi(x)} \tag{2.2}$$

Where ρ(x) denotes its *amplitude (*that we will also note *as magnitude* and *energy* to its square value), ϕ(x) is the phase and the components C(x) and S(x) will represent respectively the responses of the even and odd filter. Whereas the local amplitude is a measure for the local contrast of a structure, the local phase describes the structure or shape of the signal [OPP81] allowing splitting luminance and structural information.

Phase information has been widely used at the literature. As it is manifest on the literature [FLE93], phase information is more stable against change on contrast, scale, or orientation. It can be used to interpret the kind of contrast transition at its maximum [KOV99], e.g., a phase of π/2 corresponds to a dark–bright edge, while a phase of 0 corresponds to a bright line on dark background. It has been applied to numerous applications, specially for motion [FLE90], [FLE92], [GAU02], and stereo processing

[SOL01], [COZ97], [FLE91] , [SAN88]. Furthermore, the phase-based approaches extract subpixel information without requiring extra-processing or feature localization which makes simpler the computation of these primitives (as explained on Chapter 4).

2.1.3 Local features interrelation

The orientation encodes the geometric information of the local signal while the phase can be used to differentiate between diverse image structures ignoring orientation differences. Energy (or equivalently its square root, the magnitude) keeps the information about the local luminance and contrast (which is a valuable parameter on the estimation of confidence parameters for our features).

As commented, the estimation of the local phase and the local energy is an important step in many signal and image processing tasks. A second crucial task in image processing is the estimation of the local orientation. In most cases, the energy and phase are computed using a set of filters with some predefined orientation. Each complex filter is composed by an odd an even component, where one is the Hilbert transform of the other. Because local phase and energy information are intrinsically 1-dimensional features, the preferred orientation is necessary for its computation unless some spherical filter is used [FEL01].

If these features are not computed at their corresponding orientation, our estimation will be suboptimal and will not reflect the right values. It makes necessary a proper covering of the orientation space in order to obtain accurate estimations for these features. Furthermore, taking that under consideration, if the signal presented at an image position is not 1-D as happens in corners, junctions or textures, these features can be multi-valued and complex analysis will be required.

2.2. Quadrature filters approaches for local phase, energy and orientation estimation

In the previous analysis we review the different methods for computing the local orientation and its relation with the phase and energy. We mentioned the large number of applications that may use these features. The main limitation is the high computing power necessary for image processing of such features which reduces the applications fields. This motivates the development of customized hardware to address this problem.

From the previous Section 2.1 we conclude that quadrature filters are the best option for a real-time hardware system for the following reasons:

1. Quadrature filters are based on convolutions which are hardware friendly operations.
2. There are substantial contributions about these approaches which mean that this approach is mature enough for hardware implementation. Furthermore, there are a considerable number of applications that use this preliminary stage as input, allowing resources sharing on our processing architecture.
3. Quadrature filters represents a biological approach that models cells of the visual cortex [DEA91]. It can drive future experiments and be used as model for testing neural computation models.

Three different quadrature filters set will be considered for our study, Steerable filters based on Gaussian derivatives, Gabor filters and the isotropic analytic filters such as the Monogenic signal transform. We will study their accuracy, robustness and implementation feasibility as well as resources consumption in order to decide which filter fits better our system architecture.

2.2.1. Generic filter implementation considerations

There are several generic filter considerations that we want to highlight before come into details of the different approaches. Given a bandpass filter of peak frequency f_0 and Banwdith β (defined at the cut-off frequency corresponding to half of the base-band amplitude spectrum), we should consider:

1. Nyquist sampling condition: Using pixels as units, the sampling period is 1 pixel, which corresponds to a 1 pixel^{-1} sampling frequency. The maximum bandwidth of the filter to avoid aliasing is 0.5 pixels^{-1}. Taking this into account, given β the filter bandwidth, the maximum peak frequency of the filter f_0 can be derived from the following equation:

$$f_0 + \beta < 0.5 \tag{2.3}$$

It is worthy to mention that, since all the filters considered in this section are not bandlimited, some aliasing will occur regardless of the sampling density. In other words, by setting a filter bandwidth we decide how much aliasing we tolerate.

2. Multiscale frequency space coverage. The distance between neighboring frequency "channels" is determined by the spatial frequency bandwidth. A efficient implementation that covers different spatial scales is proposed by [BUR83]. Because this representation based on scaling factors power of two (logarithmic coverage), the minimum bandwidth to cover the frequency domain without empty areas is:

$$\beta >= f_0/3 \qquad (2.4)$$

3. Uniform orientation coverage condition (only for oriented filters such as Gabor or Gaussian derivatives). Because we should cover the 2-D frequency domain for different orientations, we need to consider a minimum number of oriented filters. This number depends on the filter bandwidth and is related with the desired orientation filter sensibility. According to [FLE90], we can estimate the desired bandwidth using:

$$2\pi f_0 <= N_{orientation} * 2\beta \qquad (2.5)$$

and define the orientation bandwidth B_θ in the frequency domain as (cf. [BOV90]):

$$B_\theta = \tan^{-1}(\beta / f_{peak}) \qquad (2.6)$$

Spatial frequency bandwidths are constant in octaves, and orientation bandwidths are constant in degrees, but there is freedom to choose the absolute magnitudes of these bandwidths (provided that they respect condition 1).

4. The spatial extent of the filter should not exceed the number of taps. Because the filters we consider have infinite extension, the condition to fulfill is that we keep at least the 95% of the energy of the filter.
5. DC removal. Gabor even filters has a significant DC response. Gaussian derivatives do not have it but, due to the sampling and windowing operations, they also can be affected. This does not happen for Monogenic filters because the even component is composed by differences of Poissons that can scaled after sampling to eliminate the DC component. For the Gaussian derivatives and Gabor filters, several approaches are possible. For example:
    - Convolve the image with a kernel: $I-I_{mean}$ where the mean value is computed on a square window of size half of the filter [DIA05a], [DIA05b].

- Numerical optimization to removal this component as [NES98].

2.2.2. Gabor filters approach

Widely used on the literature, Gabor filters are defined by harmonic functions modulated by a Gaussian distribution. Their main property is that they minimized the spatio-frequency uncertainty. An efficient filter implementation could be finding on [NES98] or [DIA06i]. Note that they are not separable filters but can be computed as sum of separable filters as described on [NES98].

Shown below in Figure 2.2 are the resulting Gabor filters at 8 different orientations.

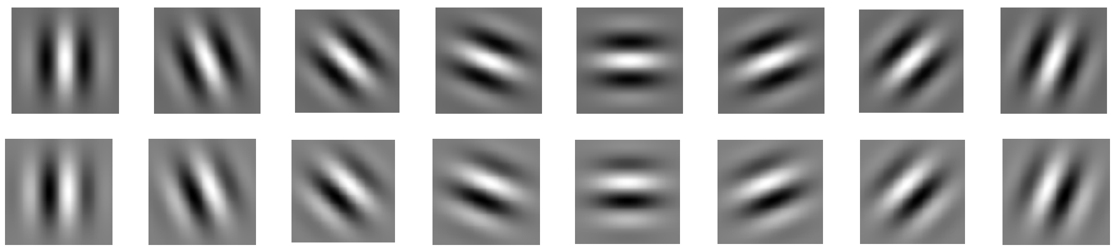


Figure 2.2. Example of a Gabor filters bank at 8 different orientations. First row shows the even filter and second row the odd filter. Each oriented quadrature filter is composed by this two filters, represented by columns.

The bandwidth used in [NES98] equals $f_0/3$ or 1 octave, which is the smallest one to properly cover scale space. In [NES98], only 4 orientations are supported but it can be extended as in [DIA06i] to 8 orientations. By exploiting the symmetry, all 8 even and odd filters can be constructed on the basis of 24 1-D convolutions (lower than the 32 theoretically needed because some of then can be reused). The main limitation of this approach is that if more orientations need to be considered, the number of convolutions grows exponentially and therefore the computing resources. For this approach, using a filters bank properly sampled in the frequency domain, features computations can be done accurately using different interpolation functions. This is described on Section 2.3.

2.2.1.2 Filter implementation

Basic equation for Gabor complex filters is given by equation 2.7, with f_0 the peak frequency, θ the main orientation and σ the Gaussian variance.

$$G(x,y,\theta)=\exp\left\{\frac{-(x^2+y^2)}{\sigma^2}\right\}*\exp\{i2\pi f_0(x\cos(\theta)+ysen(\theta))\} \qquad (2.7)$$

Basically, the Gabor filters are a Gaussian multiplied by a sinusoidal function. The spatial window extension is approximately [-2σ, 2σ], which has be considered to determine the number of taps for the filter and corresponding computing resources. We also need to consider the Variance-bandwidth relation. The bandwidth of the Gabor filters is equal to the bandwidth of its associate Gaussian. This is:

$$\frac{G(0)}{G(B)}=2 \quad \Rightarrow \beta=\frac{\sqrt{2*\ln(2)}}{2\pi\sigma} \qquad (2.8)$$

As in [NES98], we will consider a Gabor filter with $f_0 = f_{Nyquist}/2$ and 11 taps. The corresponding bandwidth makes necessary to use at least 8 orientations according to equation (2.5) and therefore as commented on [DRI06] 24 1-D separable convolution are required.

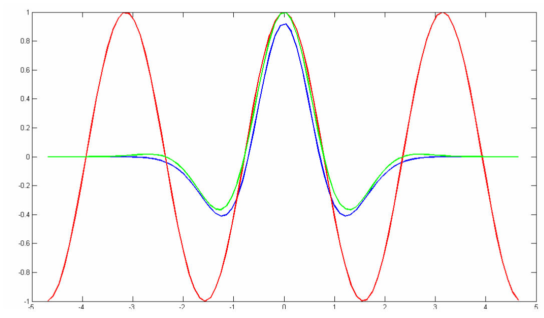
The complexity for computing *K* separable convolutions using a kernel of *N* taps is:

$$O_{sep}(K,N)=K*N \textit{ multiplications} + K*N-1 \textit{ additions} \qquad (2.9)$$

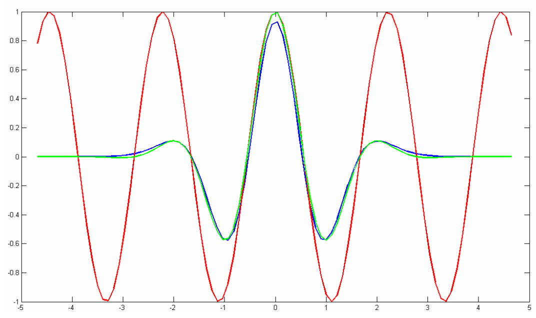
This means that the Gabor filter bank, using 24 total filters (odd and even) of 11 taps requires 264 multiplications and 240 additions.

2.2.2 Gaussian derivatives approach

Widely used on the literature, [KOE87], [BLO96], [DIA03], [MOT05], Gaussian derivatives allow the computation of any particular orientation based on a basic set of separable kernels, this property is usually referenced as stereability [FRE91]. The kernels have to be properly weighted to get the desired oriented kernel. Quadrature filters are computed using the Hilbert transform of the Gaussian derivative as described in [FRE91]. A key factor for its design is the derivative order, since the tuning frequency and bandwidth depend on this parameter. Figure 2.3 illustrates the effect of using different derivative orders and its relation with the Gabor filters.



(a)



(b)

Figure 2.3. Comparison of Gabor (green), Gaussian derivatives (blue) and cosine (red) functions tuned to the same peak frequency. (a) Gaussian derivative of order 2 is close to Gabor filters but the difference is not negligible. (b) Gaussian derivative of order 4. This time the similarity is larger and the filter is very close to the Gabor approach. Note that the number of waves increase according to the Gaussian derivative order, corresponding to higher orientation selectivity.

The main property of Gaussian derivatives is that we can compute the exact response at any orientation using a linear combination of these filters outputs. The base set for the case of Gaussian derivatives of second order is shown on Figure 2.4.

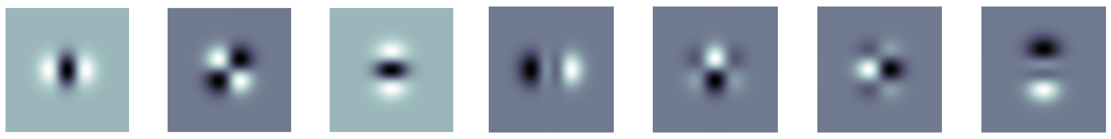


Figure 2.4. Second order Gaussian derivatives separable base set. The three first filters (from left to right) are the Gaussian derivatives and their Hilbert transforms are the four filters showed on their right. With this set of filters, we can estimate the output at any orientation just combining linearly the base set output. This allows building oriented quadrature filters banks as shows in Figure 2.3 but at any possible orientation.

2.2.2.1 Filter implementation

The well known equations of a 1-D Gaussian g and its derivatives are:

$$g_0(x) = e^{-\frac{x^2}{2\sigma^2}} \qquad g_n(x) = \frac{d^n}{dx^n} g_0(x) = P_{n,\sigma}(x) g_0(x) \tag{2.10}$$

This equation indicates that the nth derivative of a Gaussian can be written as the product of a polynomial (generalized Hermite polynomial) by the original Gaussian. In the frequency domain Equation (2.10) can be expressed as:

$$G_0(\omega) = \sigma e^{-\frac{\sigma^2\omega^2}{2}} \qquad G_n(\omega) = (j\omega)^n G_0(\omega) \tag{2.11}$$

where G is the Fourier transform of g and ω the frequency on rad/pixels. Because we focus on phase-based approaches, we need the quadrature pair of these filters. It can be obtained using its Hilbert transform [FRE91].

An important difference compared with Gabor filter can be extracted from equation (2.11). It can be derived that, given a predefined filter orientation, the spectrum distribution around the peak frequency is not 2-D symmetric (differently to Gabor approach). Gaussian derivatives present wider orientation bandwidth on the direction normal to the filter main axe resulting then in a broad tuning for local orientation.

The basic parameters to consider for these filters in their design process are:

1. Spatial window extension [-2σ, 2σ], where σ^2 stands for the variance.
2. Variance-bandwidth relation. From [KOE87] we get the asymptotic bandwidth can be computed as:

$$\beta \to \frac{1}{4\pi\sqrt{2}\sigma} \tag{2.12}$$

3. Its peak frequency f_0 is computed by derivation in the frequency domain as in [BLO96]:

$$f_0 = \frac{1}{2\pi}\sqrt{n/\sigma^2} \tag{2.13}$$

If we desire to design a filter similar to the previous described Gabor filter, it means that we use $f_0 = f_{Nyquist}/2$, fourth order Gaussian derivatives and also 11 taps. The steerability property allows deciding the filter orientation based on the image stimulus or just getting a set of oriented filters for predefined orientations. Both alternatives can be considered. Concerning the derivative order, we will study the properties of the 2 and 4 orders in Sections 2.3 and 2.4. The number of kernels to compute the oriented output of the filters k, depend on their derivative order n. We need $k'=2n+3$ separable 2-D kernels or $k=4n+6$ 1-D kernels. The complexity for computing these convolutions is, according to equation (2.9):

- n=2, k=14, 153 multiplications + 140 additions.
- n=3, k=18, 198 multiplications + 180 additions.
- n=4, k=22, 242 multiplications + 220 additions.

The conclusion is that, depending on the derivative order we can significantly reduce the resources consumption compared with Gabor filter if low derivative order is utilized but we need to check the accuracy in order to test our primitives. We address this point on Section 2.4.

2.2.3 Monogenic signals

This transform is based on a nobel generalization of the 1-D concept of phase for 2-D signals [FEL01], based on the Riesz transform, which is used as generalization of the Hilbert transform. The monogenic signal performs a split of identity, i.e., it orthogonally divides the signal into energetic information (indicating the likelihood of the presence of a structure), its orientation and its structure (expressed in the phase). Orientation is used as disambiguation information to extend the 1-D concept of phase.

The image is convolved with three 2-D non-separable filters which are *spherical quadrature filters,* allowing the estimation of the phase and energy of the signal for the main orientation (which it is also extracted). Because only the information of the main direction is provided, this approach is only recommended for 1-D signals as edges or lines.

At this point is important to introduce the concept of intrinsic dimensionality [FEL02a]. It is possible to analyze the local image structure and determine if it represents a 1-D feature such as an edge or line or a 2-D feature such as a corner or features presented on textures. This concept can be even extended to a continuous probabilistic formulation as proposed in [KRÜ03] and computed using on the monogenic signals.

The main advantage of the monogenic signals is that, thanks to using spherical quadrature filters, we always obtain the predominant orientation features. It is as well its main limitation, when we have 2-D structure, the multiple features existing at each orientation can be used on applications such as texture segmentation [DUN95] or transparent motion discrimination [SIM98]. These applications are lost using the monogenic signal approach in which these features are not extracted.

2.2.3.1 Filter Implementation

The monogenic signal allows computing *spherical quadrature filters* from 1-D quadrature filters as described on [FEL01]. We will use this transform applied to the *Difference of Poissons* (DOP) functions as in [FEL01], [FEL02a]. The three resulting functions for the kernels are:

a. Radial filters (with even symmetry with respect the to origin):

$$h_e(x) = \frac{s_1}{2\pi\left(x^2 + y^2 + s_1^2\right)^{3/2}} - \frac{s_2}{2\pi\left(x^2 + y^2 + s_2^2\right)^{3/2}} \tag{2.14a}$$

$$H_e(u) = \exp(-2\pi|u|s_1) - \exp(-2\pi|u|s_2) \tag{2.14b}$$

Where s_1 and s_2 represent the poles of the Poisson functions, h_e represents the signal on the spatial domain and H_e its Fourier transform. They fix their spatial extension and bandwidth.

b. X-Y filters (real and imaginary parts with odd symmetry)

$$h_o(x) = \frac{x + iy}{2\pi\left(x^2 + y^2 + s_1^2\right)^{3/2}} - \frac{x + iy}{2\pi\left(x^2 + y^2 + s_2^2\right)^{3/2}} \tag{2.15a}$$

$$H_e(u) = \frac{y - ix}{|u|}\left[\exp(-2\pi|u|s_1) - \exp(-2\pi|u|s_2)\right] \tag{2.15b}$$

c. Systems relations for filter design.
   1. Filter poles relation. According to [FEL02a], we take $s_2=2\cdot s_1$.
   2. Spatial window extension $[-s_2, s_2]$ for windowing.
   3. Peak frequency:

$$f_0 = \frac{1}{2\pi(s_2 - s_1)}\ln\left(s_2 / s_1\right) \tag{2.16}$$

   4. Poisson bandwidth:

$$\beta = \frac{\ln(2)}{2\pi s} \tag{2.17}$$

   5. DOP bandwidth (using $s_2=2\cdot s_1$):

$$\beta = \frac{1}{4\pi s_1}\left[\ln\frac{2}{1-\sqrt{1-2M(f_0)}} - \ln\frac{2}{1+\sqrt{1-2M(f_0)}}\right] \tag{2.18}$$

The DOP based filters requires not separables 2-D convolutions. Computing *K* non separable 2-D convolutions of *N* taps has a complexity:

$$O_{no\text{-}sep}(K,N)=K*N^2 \text{ multiplications} + K*(N^2-1) \text{ additions} \tag{2.19}$$

It means that the computing resources for this approach using a 11x11 taps filter implementation requires 363 multiplications and 360 additions. It is then the more expensive approach and, its implementation only can be justified for the sake of accuracy.

2.3. Features interpolation from a set of oriented filters

The previous stages describe several quadrature filters types. From them, there are several methods to compute the local image features phase, orientation and energy.

The monogenic signal extracts directly this information for the main orientation using the equations:

$$E_{local} = \sqrt{I_e^2 + I_o^2} \tag{2.20}$$

$$\theta_{local} = \arg(I_o) \quad \mathrm{mod}(\pi) \tag{2.21}$$

$$P_{local} = sign(\zeta\{I_o\}) * \arg(I_e + i|I_o|), \quad \zeta\{I_o\} = real(I_0) * \cos\theta + imag(I_0) * \sin\theta \tag{2.22}$$

Where we note E_{local}, θ_{local} and P_{local} the magnitude, orientation and phase computed from the monogenic signal.

If we consider 8 oriented filters (computing using Gabor or Gaussian Derivatives), is likely that the local orientation of some features do not fit this discrete number of orientations. Under this circumstance, we require to interpolate the feature values computed from this set of outputs in order to estimate the filter output at the proper signal orientation. Different methods can be used. We note E_i and P_i to the magnitude and phase of the filter oriented with angle=i*π/N and noted by h_i. This filter is expressed by:

$$h_i = c_i + js_i, \tag{2.23}$$

And the primitives features are computed with this filter orientation and computed as:

1. Filter energy → $E_i = [c_i]^2 + [s_i]^2$ (2.24)
2. Filter phase → $P_i = \arg(c_i, s_i)$ (2.25)

If only the main orientation information is required (1-D local signals), we can apply several strategies to interpolate the primitives from this multivalued set:

i. Winner- take-all. We will take for each pixel the phase, energy and orientation of the filter with maximum energy.

$$E_{local} = E_{\max} \quad P_{local} = P_{E_{\max}} \quad \theta_{local} = \theta_{E_{\max}} \tag{2.26}$$

ii. Weighted-average: (we consider linear case, though the energy can be power to different orders).

$$E_{local} = \frac{\sum_i E_i}{N} \quad P_{local} = \frac{\sum_i E_i P_i}{\sum_i E_i} \quad \theta_{local} = \frac{\sum_i E_i \theta_i}{\sum_i E_i} \tag{2.27}$$

where all angles are properly shifted for avoiding angle wrapping effects.

iii. Tensor-based method [HAG92]. Based on a local tensor that projects the different orientations, information can be computed as follows:

$$E_{local} = \frac{\sum_i E_i}{N} \tag{2.28a}$$

$$\theta_{local} = \arg\left(\sum_i \frac{4}{3}\sqrt{c_i^2 + s_i^2}\, \exp\{2\theta_i\}\right) \tag{2.28b}$$

$$\begin{aligned} c &= \sum_i c_i \cos^2(\theta_i - \theta_{local}) \\ s &= \sum_i s_i \bullet sign(\cos(\theta_i - \theta_{local})) \bullet \cos^2(\theta_i - \theta_{local}) \\ P_{local} &= a\tan\left(s/c\right) \end{aligned} \tag{2.28c}$$

iv. Energy Fourier series expansion for Gaussian derivatives based approach. As described on [FRE91], using the nth Gaussian derivative G_n and its Hilbert transform H_n as band pass filter oriented to the angle θ, we have that the energy at this orientation is expressed by:

$$E_n(\theta) = [G_n^\theta]^2 + [H_n^\theta]^2 \tag{2.29}$$

Writing these functions using the separable basic filter outputs, this equation can be expressed as a Fourier series in angle and described as:

$$E_n(\theta) = C_1 + C_2\cos(2\theta) + C_3 sen(2\theta) + high\ order\ terms \tag{2.30}$$

Note that values of coefficients C_i can be found on [FRE91] for n=2 case. From equation (2.28), local orientation is computed based on the lowest frequency term as:

$$\theta_{local} = \frac{\arg(C_2, C_3)}{2} \tag{2.31}$$

The energy is estimated using equation (2.29) and the phase using (2.25) taking into account that, for *n* even, c_i is G_n and H_n is s_i and the opposite for odd values of *n*.

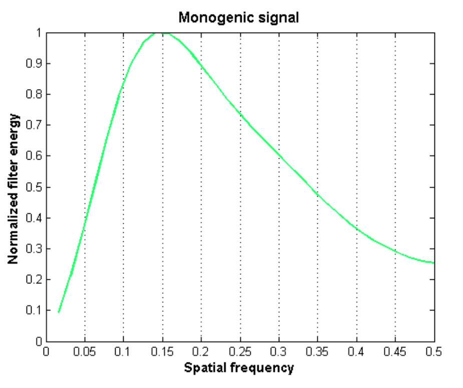
2.4. Quantitative analysis of the accuracy for feature estimation of the presented approaches.

In Section 2.2 we study the computing complexity of the different filters and conclude that Gaussian derivatives with low derivative order are the less computational load approach and monogenic signal the most expensive one. Section 2.3 shows the underlying equations and methods that we can use to estimate the local image features from each filter type. Now we are going to evaluate the accuracy of the different approaches using the equations presented on previous section.

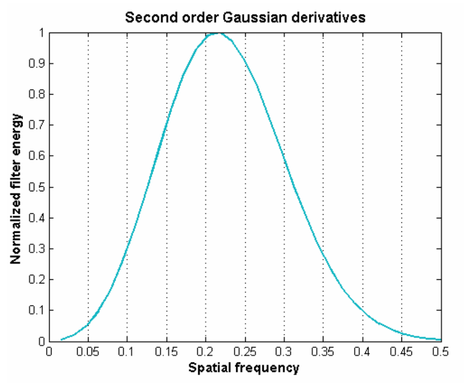
Comparing the different approaches is a hard task to do due to the large number of variables to consider for filtering design. Furthermore, the parameter choice can significantly bias the results, possibly leading to wrong conclusions. Because of that, we will focus on a most affordable task; we will use some fixed filter parameters that exploit each signal type properties. For instance, Gabor and 4-order Gaussian derivatives allow very fine filter tuning capabilities and orientation selectivity. As in [NES98], our designed filter will have a peak frequency of f_0=0.25 pixels^{-1} and bandwidth β= f_0/3=0.083 pixels^{-1}. Monogenic signals and second order Gaussian derivatives have broad bandwidth and therefore, peak frequency should be lower to fulfil equation (2.3). We use f_0=0.21 pixels^{-1} and bandwidth β= 0.1 pixels^{-1} for the Second order Gaussian derivative as in [FRE91]. For the Monogenic signal, the design values are S_1=1 and S_2=2, which gives us the higher frequency filter based on this approach. It gives a peak frequency of f_0=0.11 pixels^{-1} and mean bandwidth β= 0.14 pixels^{-1} (bandwidth curve is not symmetric and therefore we only provide its mean value). All these values have been computed using the equations described on Section 2.2.

In order to test the different approaches, we use two different kinds of signals. First, a set of synthetic sinusoidal gratings with different orientations and spatial scales is used. For this stimulus image features are known and we can numerically test the accuracy of the filters. Second, we also have used real images to get some qualitative results.

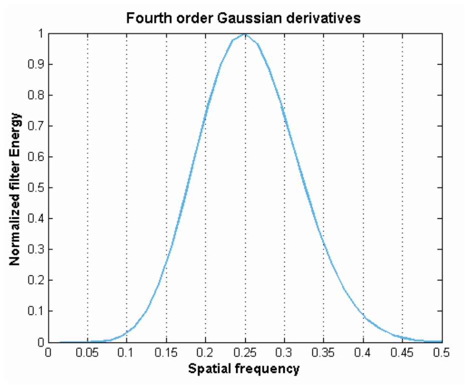
From the sinusoidal gratings set, the experimental energy values of the different filters responses across the scales is represented on Figure 2.5 (note that these are the experimental results, which consider for example quantization problems or finite spatial kernel size). It confirms our numerical bandwidth values and shows that for our design, Gabor filters have the narrowest bandwidth and Monogenic signal the widest one.



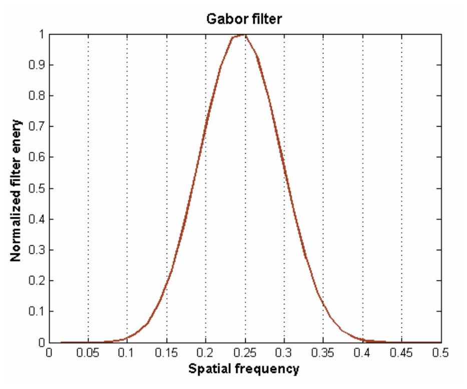
(a)



(b)



(c)



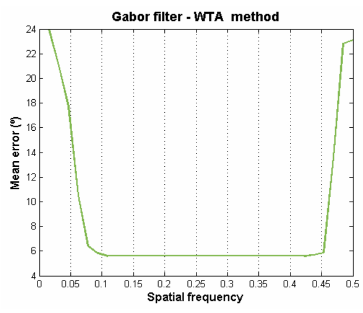
(d)

Figure 2.5. Normalized energy distribution across the spatial frequency scales, experimental results using a sinusoidal gratins test (x-axe use units on pixels^{-1}) for the Monogenic signal (a), Second order Gaussian derivatives (b), Fourth order Gaussian derivatives (c) and Gabor filters (d). Filters bandwidth decrease from (a) to (d).

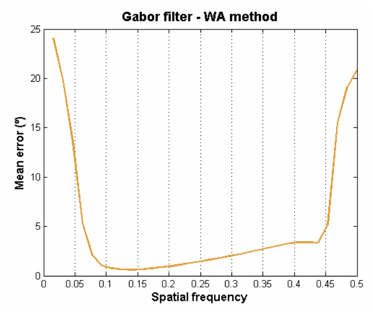
Our goal is to compare the different alternatives accuracy taking into account on their hardware implementation feasibility. We have three features to evaluate but we will focus on the local orientation estimation accuracy. Local energy is valuable as reference to discriminate areas with low or high contrast and therefore, its numerical value is not important but rather its relative value compared with closer areas or with respect the whole image. Because of that, there is not reason for evaluating numerically this feature between filters types in this context. Local orientation is necessary to compute phase and therefore, error or bias on its estimation significantly can degrade the phase accuracy. The phase information is related with the filter shape and therefore numerical evaluation is quite complex and far away from the presented analysis. It has more sense on an application specific context as in [DIAZ06i] where the phase coming from different filters is used for optical flow and stereo disparity computation. Given the previous discussion we focus on orientation selectivity to decide between the different approaches.

In the Figure 2.6 we measure the mean error vs. sinusoidal grating spatial scale. Data outputs are unthresholded and therefore, large errors are not significant if the filter energy value is close to zero. For approaches that need of filter responses interpolation, the tree methods presented in section 2.3, Winner-take-all, weighted-average and Haglund tensor are compared. Several conclusions can be extracted from these figures.

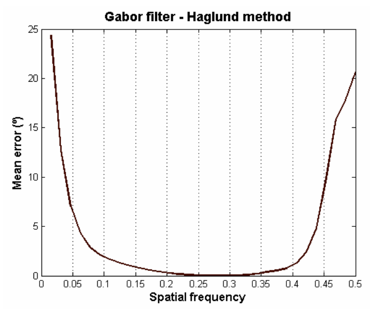
1. The best interpolation method is the Haglund approach. It produces the smaller error on the filter frequency band.
2. All the filters have high accuracy for orientation estimation, less that 1° of error.
3. The filters that cover the wider range areuses of the Monogenic signals and the second order Gaussian derivatives. This confirms the theoretical analysis in Section 2.2 relative to its bandwidths.
4. For the second order Gaussian derivatives, Haglund tensor approach and the Freeman Fourier series expansion produce equivalent results.



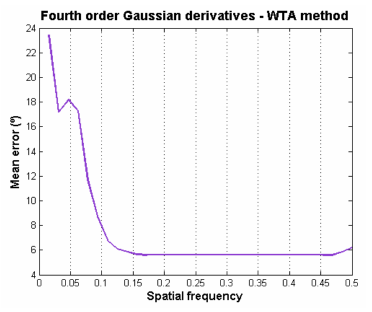
(a.1)



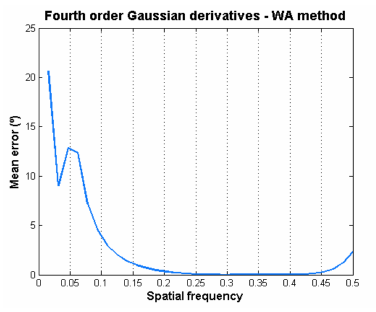
(a.2)



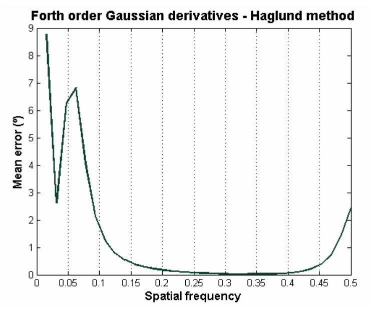
(a.3)



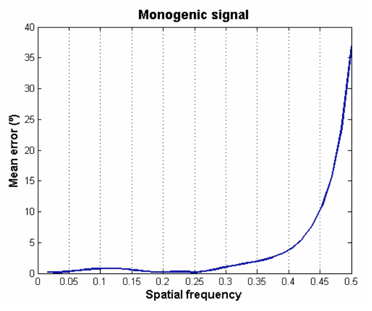
(b.1)



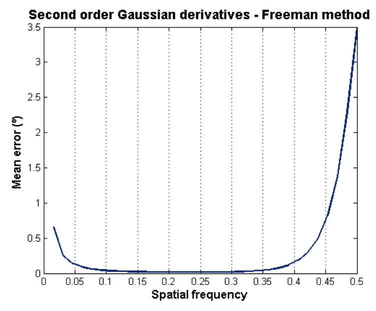
(b.2)



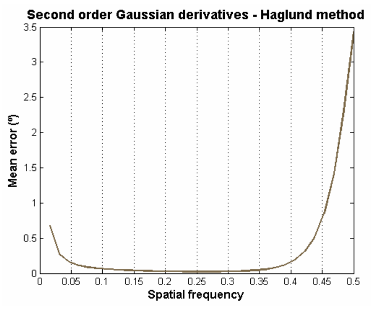
(b.3)



(c)



(d)

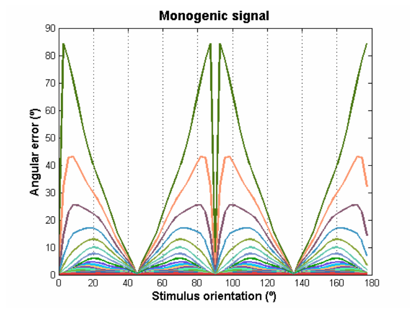


(e)

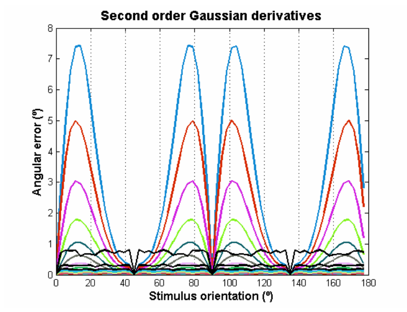
Figure 2.6. Mean orientation error measured for each grating spatial frequency. We have used sinusoidal gratings as input, oriented to 64 different angles and with spatial scales from 0.5 pixels^{-1} to 0.0078 pixels^{-1}. Note that, for the sake of clarity, we use different y-axe scales but error values are quite different for the different approaches.(a) Gabor filters are computed at 8 orientations and three interpolation methods are used: (a.1) Winner take all (*WTA)*, (a.2) Weighted average (*WA*) and (a.3) Haglund approach. (b) The same methods are utilized using the Fourth order Gaussian derivatives. (c) Monogenic signal results. (d) Second order Gaussian derivatives, orientation computed using the Freeman and Adelson approach [FRE91]. (e) Second order Gaussian derivatives, orientation computed using eight oriented filter and the Haglund [HAG94] interpolation method.

We also have measure the different filter behaviours against several noise types (multiplicative, Gaussian white noise and salt and pepper). As expected, the error grows approximately linear by all the approaches and therefore, robusness to noise will not drive and affect significantly the decision between the different filters.

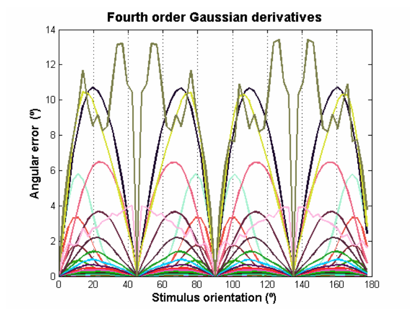
The error presented at each orientation and scale is showed in Figure 2.7, using Haglund interpolation filter approach when required. There are 32 different scales from 0.5 to 0.0078 pixels^{-1} that makes difficult to use colors legends to mark each case. Therefore, they are only used for qualitative error hints, where large error is presented at scales far away from the filter tuning peak frequency. Smooth error curves descend indicating gratings from high to lower spatial frequencies. For the filter tuning frequency, error is close to 0 and therefore it is not visible on the graphics. These figures also show error curves with flat responses or multiple narrow peaks. It happens when we pass from the tuning scale to coarse scales where the filter response confidence is quite low (energy is close to 0) but this reponses can be easily filtered using energy thresholds.



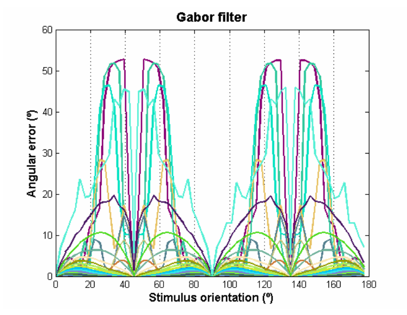
(a)



(b)



(c)



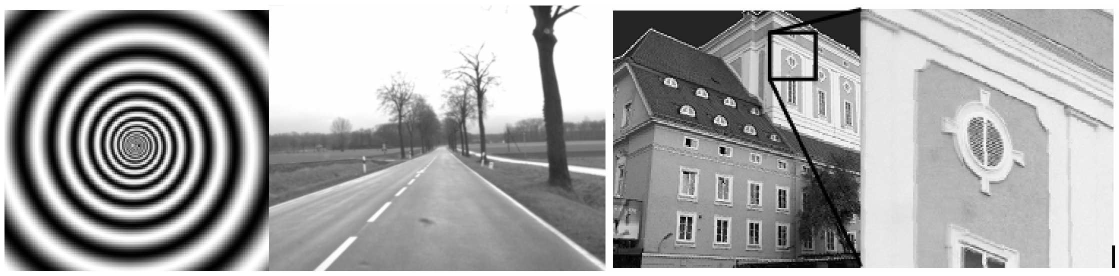
(d)

Figure 2.7. Error evolution across the different stimulus orientations. We have used sinusoidal gratings as input, oriented to 64 different angles and with spatial scales from 0.5 pixels^{-1} to 0.0078 pixels^{-1}. Each spatial frequency filter output is represented on a different colour (there are 32 curves). (a) Monogenic signals results, error decreases with the spatial frequency in each plot. For low spatial frequencies, the filter provides quite accurate orientation estimations but it gets worse with high frequency gratings. (b) Second order Gaussian derivatives. There is a frequency range where the filter properly matches the stimulus orientation. For low frequency patterns, the error increases as represented on black lines at the bottom of the plot. (c) Fourth order Gaussian derivatives and (d) Gabor filters have a small bandwidth. Thus, in this case stimulus with spatial contexts far from the filter tuning frequency are prone to high orientation errors. Graphics (b), (c) and (d) use the Haglund approach for computing orientations based on a set of 8 oriented quadrature filters. Note that results are unthresholded. Large errors appear in zones of almost zero energy but this feature can easily be used as confidence parameter for tuning the filter to the best spatial scale.

The qualitative results of features computation for images in Figure 2.8 are illustrated on Figures 2.9, 2.10 and 2.11. Three examples are shown:

1. Synthetic spiral image (Figure 2.8.a) that covers all the orientation as well as different spatial scales. Results presented in Figure 2.9.

2. Forward view of a road (Figure 2.8.b) with a well defined structure that allows easy identifications of the image features. Two different image scales are shown, the original one and the image reduced by a factor 4 in Figure 2.10.
3. Finally, real image of a house is used. We focus on the details of a circular skylight (Figure 2.8.c) and the extracted primitives are computed and shown in Figure 2.11.

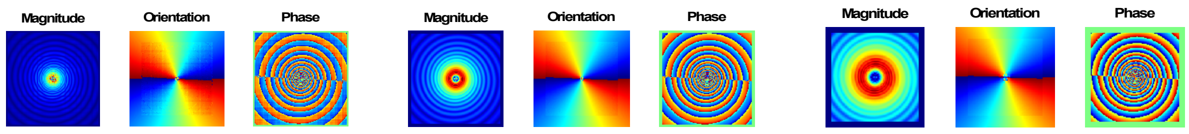


(a) **(b)** **(c)**

Figure 2.8. Original images used for qualitative evaluation of the different approaches.

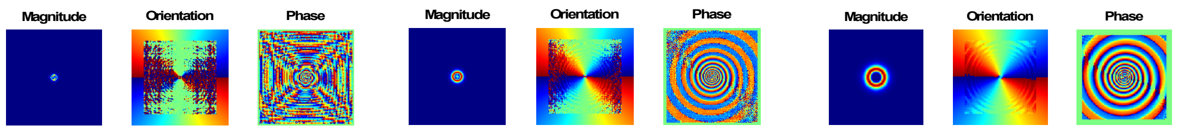
Monogenic signals

(a.1) **(a.2)** **(a.3)**



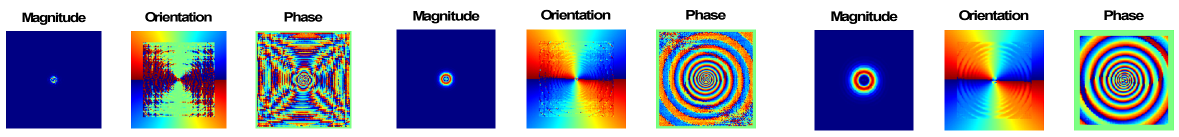
Gabor filters

(b.1) **(b.2)** **(b.3)**



Four order Gaussian derivatives

(c.1) **(c.2)** **(c.3)**



Second order Gaussian derivatives

(d.1) **(d.2)** **(d.3)**

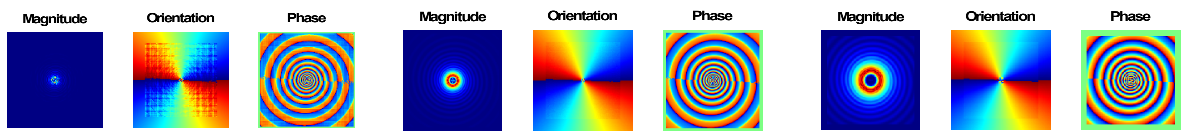
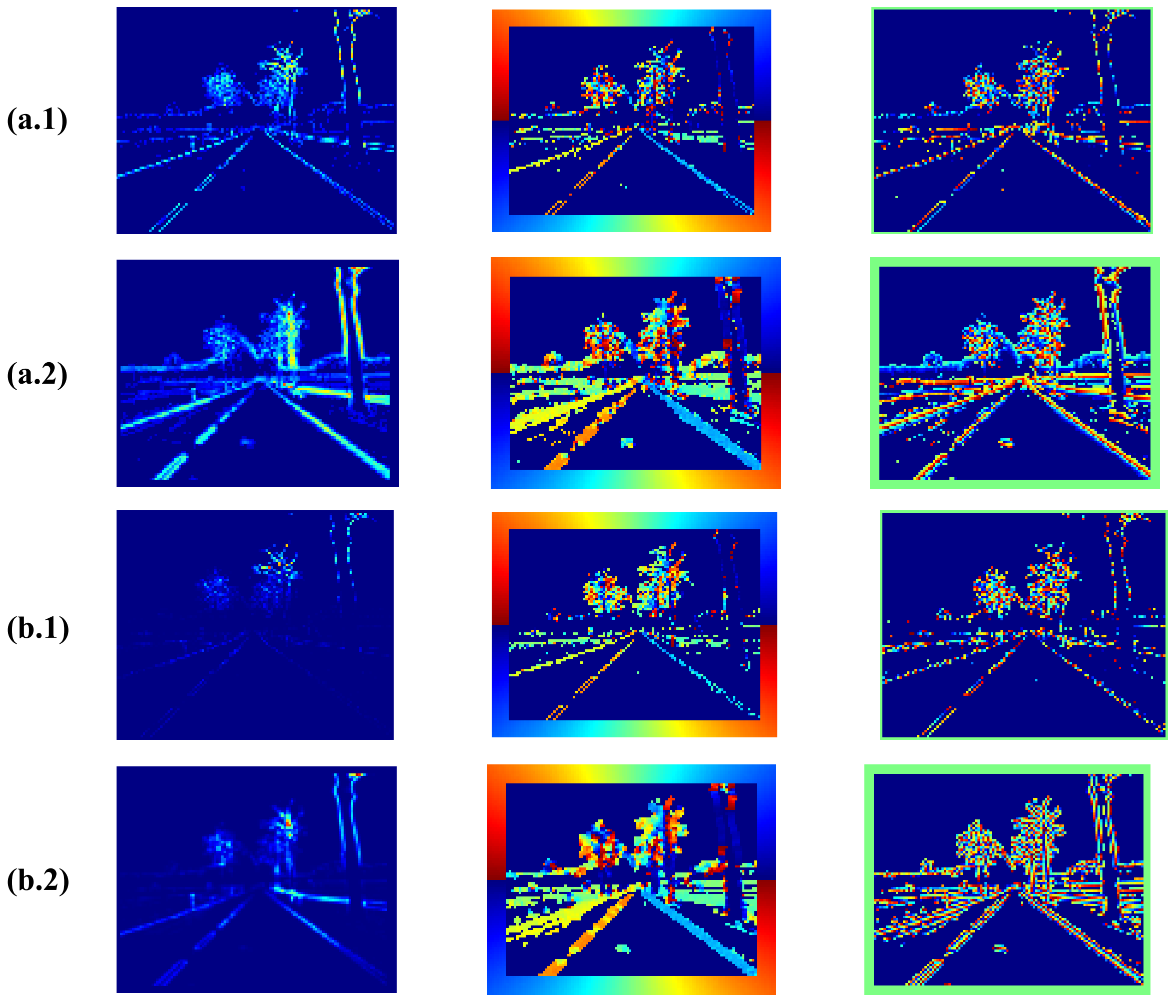


Figure 2.9. Orientation estimation for the image in Figure 2.8.a, using unthresholding results. Row (a) represents the results for the Monogenic signals, row (b) results of the Gabor filters, row (c) the fourth order Gaussian derivatives and (d) output from the second order Gaussian derivatives. Each column represent one scale, column 1 is the fine scale, column 2 represents a image resolution divided by 2 and column 3 represents the image with original resolution divided by 4. The results show that high resolution images are properly tuned only at the centre for the fourth order Gaussian derivatives and Gabor filters. The tuning region grows for lower resolution areas because the peak frequency is better tuned at these scales, as can be seen from the energy response images. Note that second order Gaussian derivatives and Monogenic signals, thanks to the wider bandwidth, allow the primitives computation at larger areas. An orientation frame is utilized for these images encoding with colours the different orientations. Note that we use the direction normal to the line (the filter axe) as orientation direction for the colormap.



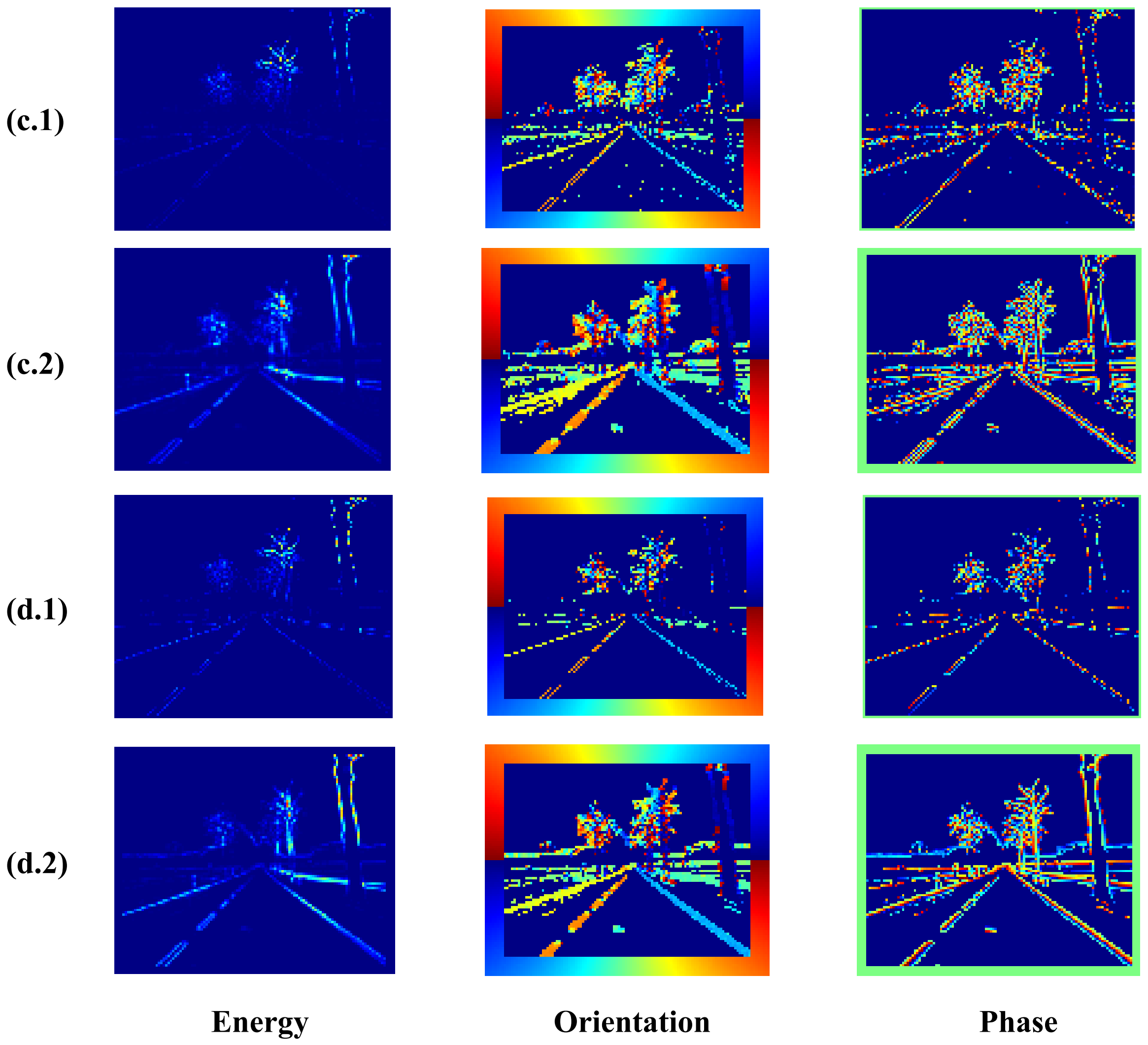
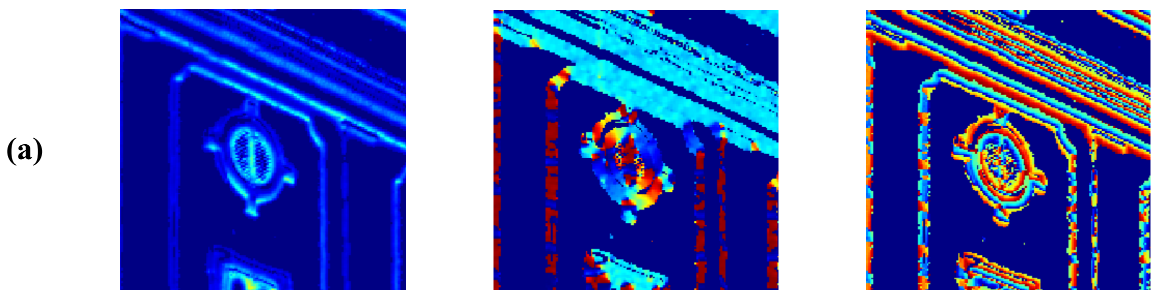


Figure 2.10. Image features computed the for road scene in Figure 2.8.b with a energy confidence threshold of 5e^{-3} times the maximum output. We use the previous filters: (a) Monogenic signals, (b) Gabor filters, (c) Fourth order Gaussian derivatives and (d) Second order Gaussian derivatives. For each filter, the image is computed at 2 scales, the original one and other resolution divided by four which is represented by the subindices x.1 and x.2 where x stand for a, b, c or d.



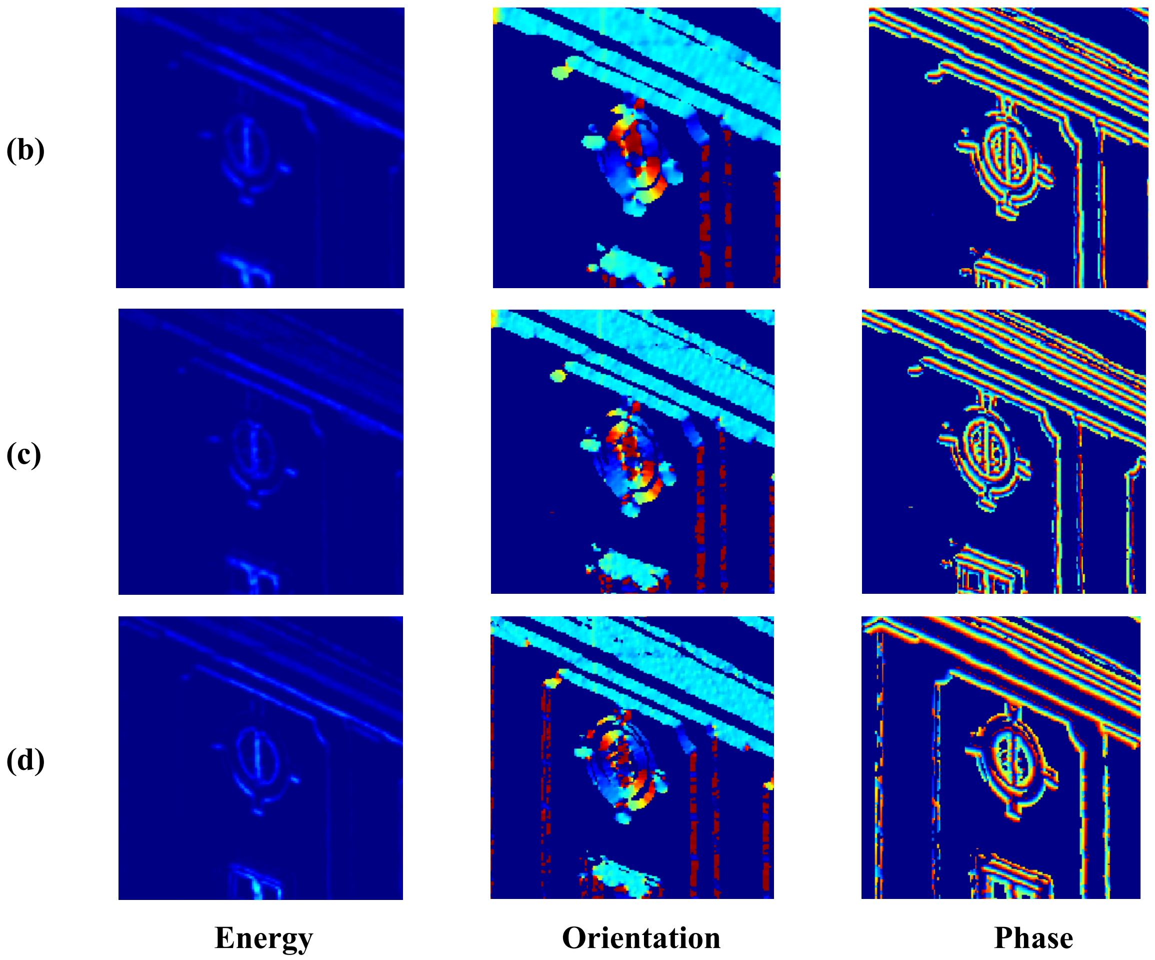


Figure 2.11. Image features computed for the circular skylight of Figure 2.8.c. We use the same filters: (a) Monogenic signals, (b) Gabor filters, (c) Fourth order Gaussian derivatives and (d) Second order Gaussian derivatives. We can appreciate that the phase information blur from (a) with the higher value to (b) and (c) with the lower value. A trade-off between these alternatives is represented by the (d) case based on the second order Gaussian derivatives.

2.4. Conclusions

From the previous analysis we conclude that the filters responses have high accuracy for spatial scales close to their peak frequency. The qualitative analysis gives us some qualitative hints for the evaluation of the different approaches. First, note that the bandwidth of the different approaches has a critical effect on feature computation at each scale. Multiscale approaches can benefit from narrow tuning filter but, for general applications with only one scale, this is a significant drawback. On [DIA06i], a multiscale algorithm for optical flow and stereo computation highlight the Gabor filter as the filter approach which produces the higher accuracy maps compared with the other alternatives. From our study, a monoscale approach benefits from a wider bandwidth. We conclude that, second order Gaussian derivatives are the best option because:

1. They require the minimum number of resources. Only seven 2-D separable convolutions are required.
2. They are a good trade-off between spatial resolution and spatial scales range. Fine resolution is extracted compared to Monogenic signals although is coarser than Gabor or higher order Gaussian derivatives.
3. Their orientation accuracy is quite high (less than one degree of error) and therefore it fulfils the requirements of most applications.
4. Arbitrary orientations can be computed from the basic set of filters just changing the filters interpolation coefficients because they are Steerable filters.

Several interpolation methods have been presented for these filters but the Freeman and the Haglund approaches show the best performance. Hardware complexity of both methods is similar and the accuracy differences are negligible. This motivates the implementation of the Haglund approach because in the future, if narrow tuning becomes necessary, the filter base can be changed to Gabor or higher order derivatives without changing the interpolation scheme. It is also a valuable factor to implement this filter base because most of the computing circuits can be reutilized although changing the filter parameters will be necessary.

Furthermore, as significantly different to the Monogenic signal, these filters provide mutivalue responses at each orientation that open the utilization of these results for texture segmentation, intrinsic dimension analysis or other 2-D image structure as corners or junctions.

Chapter 3
Motion Processing Models

This chapter focuses on motion or optic flow processing models. It briefly reviews the different alternatives and based on comparative studies in the literature concludes that the Lucas & Kanade algorithm is an approach with a good accuracy vs efficiency trade off. The chapter describes the model and also explains several modifications (proposed in the literature) that enhance its accuracy with a low cost in computational load.

3.1. Introduction

Horn and Schunck [HOR93] defined optical flow as follows:
"*The optical flow is a velocity field in the image which transforms one image into the next image in a sequence. As such it is not uniquely determined ... The motion field, on the other hand, is a purely geometric concept, without any ambiguity- it is the projection into the image of three-dimensional motion vectors.*"

Once that this slight difference between the two concepts "optical flow" and "motion field" is clarified it can be noted that the objective of extracting optical flow through mathematical methods is usually to recover the actual motion field of the scene. Therefore, in order to evaluate the performance of different algorithms we shall correlate the obtained optical flow with the ground-truth motion field. But this is only possible in synthetic sequences. Nevertheless, simple sequences such as regular repetitions of specific patterns (for instance a simple sine representation) may not be sufficient to evaluate the accuracy of a concrete algorithm given the complexity of natural scenes (mainly a wide range of spatio-temporal frequencies). This has motivated that many works evaluate different algorithms with complex synthetic sequences as benchmarks, see for example the test images available at [CVH06]. In these synthetic sequences the true 2-D motion field can be accessed and this facilitates the quantification of the model performance. Nevertheless, it must be taken into account that these sequences are clean (without any noise) and therefore the performance estimations obtained with these sequences will hardly be achieved with real sequences in which a wide variety of noise sources and luminance artefacts are present

There are many methods to extract optical flow but most of them can be structured in four functional stages:

A. Prefiltering and smoothing in order to extract image structure and enhance the signal-to-noise ratio.
B. Extraction of basic primitives (such as spatiotemporal derivatives or local correlation surfaces).
C. Integration of these primitives to produce a 2-D flow field. This step often involves assumptions about smoothness that are not always fulfilled.
D. Extraction of the subset of reliable measurements by thresholding the 2-D flow field with local confidence measurements.

Results are usually represented as shown in Figure 3.1 with a arrows map (Figure 3.1.b) or using a colormap for module an other for direction (Figures 3.1.c and 3.1.d).

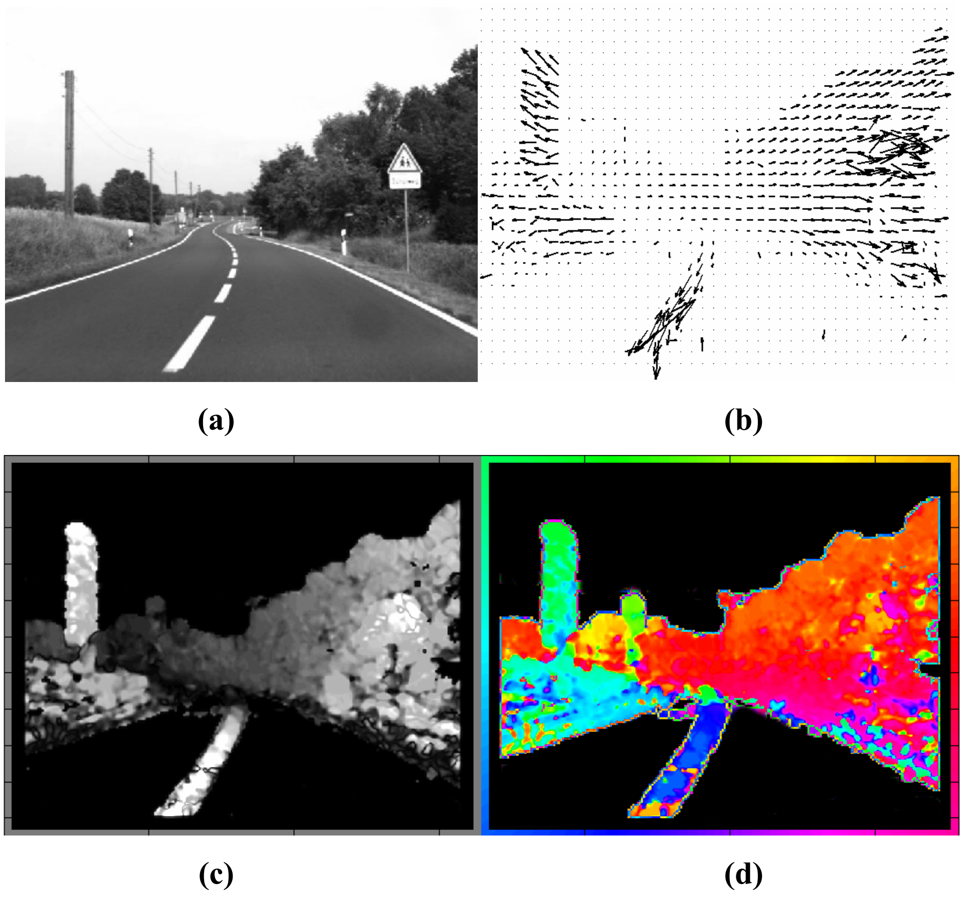


Figure 3.1. Two different optical flow representations. (a) Front-view driving vehicle scene. (b) Optical flow represented as arrow vectors scaled to indicate local pixels speed and direction. (c) and (d), Optical flow results encoded using a colormap. The velocities module uses a gray-scale as illustrated in c. Lighter colors indicate fast movements. In (d), the colormap encoding the velocity direction according to image frame colors. Note that closer objects due to the perspective look to move faster than remote objects.

3.2. Confidence measure of the optical flow estimation

Most of optical flow schemes provide one estimation per pixel. Nevertheless, there are regions in the images which are more problematic (prone to erroneous estimations) due to their spatio-temporal pattern; for instance, areas with singularities in their spatio-temporal components, or regions without any structure. In order to avoid a global increment of the error in a given scene, the optical flow fields are filtered with specific thresholds (that may depend on the spatio-temporal structure of the neighbourhood of the estimation pixel). Different optical flow algorithms use different confidence measures to enhance their accuracy keeping the maximum estimation density. In fact, a fair comparison between different optical flow approaches shall include the accuracy vs density trade-offs [BAR94].

Each algorithm benefits of its specific confidence measure, in fact the accurate choice of confidence measure and threshold significantly improves the performance. There are many confidence measures to predict the quality of optical flow estimates. Of course they depend on the optical flow method to which is applied. For instance if the optical flow is computed using differential methods the confidence measure is usually based solely on the matrix of spatial derivatives of intensity (therefore trying to reject estimations with low spatial structure). But there are many specific confidence measures such as the ones based on eigenvalues [BAR94], [SIM91], [JAH93], the determinant [BRA97], the condition number [SOB91] and the spatial gradient [BAR94]. In fact, Bainbridge-Smith and Lane [BAI96] specifically address the comparison of different confidence measure estimates concluding that the methods based on the inverse minimum eigenvalue provide a good combination of simplicity and performance.

3.3. Evaluation of optical flow models. Error metrics

A fair comparison between different optical flow algorithms requires a well defined error metric. But on the other hand, as commented in [MCC01], trying to summarize the performance of over a million flow vectors with a single number is a difficult task. It is further complicated if we try to define a general error metric not dependent of specific application tasks.

There are many of them proposed in the literature [BAR94], [FLE95], [GAL98]. The following expressions are examples of error metrics widely used:

a. Angular difference between the correct and estimated flow vectors.

$$\textit{Angular}\quad \textit{Error} = \cos^{-1}(\hat{c} \cdot \hat{e}) \tag{3.1}$$

Where *c* is the correct motion vector, *e* is the estimated optical flow vector and ^ denotes vector normalization. This error measurement has been widely used in the literature and therefore it is appropriate to compare our results with previous works. This measurement can be used with high- and low-velocity modules with the same estimators but with some bias. A more detailed explanation of this can be found in [BAR94].

b. *Average magnitude difference between the correct and estimated flow vectors for each pixel.*

$$\textit{Magnitude of Difference} = \|c - e\| \tag{3.2}$$

c. *Average error normal to the gradient.*

$$\textit{Error Normal to Gradient} = \left\|(c - e) \cdot \hat{g}^{\perp}\right\| \tag{3.3}$$

Where $g = \left(\frac{\partial I}{\partial x}(x,y), \frac{\partial I}{\partial y}(x,y)\right)$ and $\hat{g}^{\perp}$ denotes the vector perpendicular to $\hat{g}$.

This error metric measures how efficiently each algorithm compensates to the aperture problem, which is a very significant error source in optical flow models.

3.4. Comparative study of different optical flow models

Before comparing different optical flow methods, it should be noted that since we try to compute an approximation of the 2-D motion field (a projection of the 3-D velocities) only from spatiotemporal patterns of image intensity, some authors claim that only qualitative estimations can be given [VER87] or that in fact the measurement of optical flow is an ill-posed problem [ALO92]. Nevertheless, it is usually assumed that accurate motion estimations can be given if relative errors (due to projection artifacts, such as occlusion effects and other 2-D inherent errors, besides intensity variations) are bellow 10% [BAR90], [BAR94].

There are comparative studies of different optical flow algorithms [MCC01], [BAR94], [LIU98]. Among them one of the most widely referenced is the work done by

Barron et al. [BAR94] which compares a wide variety of approaches. We will flollow this comparative study reviewing several techniques.

The differences between approaches rely in the mathematical model and how neighborhood pixels information is used to drive the information of central pixel. In optical flow we should assume several well known situations illustrated in Figure 3.2. The main problems to solve, aperture problem and blank wall problem are schematically depicted in Figures 3.2.a and 3.2.c.

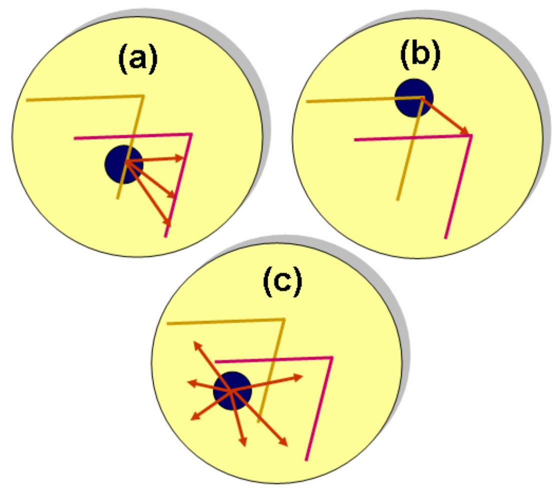


Figure 3.2. Several pixel areas configuration to compute optical flow. (a) Our local window has only information on an edge (1-D structure) and therefore only normal flow can be recovered. It is known as *aperture problem*. (b) For a corner pixel, 2-D motion field can be computed. (c) In bland areas no structure is present and there is not any vector field information. This is called *the blank wall problem*.

3.4.1. Differential Techniques

Differential techniques compute velocity from spatiotemporal derivatives of image intensity or filtered versions of the image (using low-pas or band-pass filters). Most of them use first-order derivatives and are based on image translation [FEN79], [HOR81], [NAG83], [LUC81] that is:

$$I(x,t) = I(x - vt, 0) \tag{3.4}$$

Where $\boldsymbol{v}=(v_x,v_y)^T$ and $\boldsymbol{x}$ stands for spatial vector of two components. From an assumption that intensity is conserved, i.e. $dI(x,t)/dt = 0$, the gradient constraint equation can be derived:

$$\nabla I(x,t)\cdot v + I_t(x,t) = 0 \tag{3.5}$$

Where $I_t(x,t)$ denotes the partial time derivative of $I(x,t)$, $\nabla I(x,t) = (I_x(x,t), I_y(x,t))^T$, and $\nabla I \cdot v$ denotes scalar product. Expression (3.5) represents a single linear equation with two unknown components. Therefore, further constraints are necessary to extract the two velocity components.

Some authors have used second-order derivatives [NAG83], [NAG87], [TRE84], [URA88] or even higher derivatives orders [JOH99]. But the inclusion of second-order spatial derivatives means that first-order deformations of intensity (e.g. rotation or dilation) should not be present, which represents a very strong restriction.

Another way to obtain further constraints for expression (3.5) is combining local estimates through space and time [BAI97]. One method that can be applied for this purpose is fitting the measurements in each neighbourhood for instance using least-squares minimization or a Hough transform [FEN79], [KEA87], [LUC81], [SIN90], [WAX85]. Another approach uses global smoothness constraints (regularization) in which the velocity field is calculated as the minimum of a functional defined over the image [HOR81], [NAG83], [NAG87].

The use of differential techniques requires that $I(x,t)$ must be differentiable. This means the temporal smoothing is required to avoid aliasing and the numerical differentiation must be done carefully. Gradient-based models work with high accuracy in frameworks in which the intensity is nearly linear, with velocities less than 1 pixel/frame. Among the differential or gradient techniques Barron et al. [BAR94] includes four approaches in the comparative study:

- Horn and Schunk model (H&S) [HOR81].
- Lucas and Kanade (L&K) approach [LUC81].
- Nagel (NAGEL) method [NAG83], [NAG87], [NAG86].
- Uras, Girosi, Verri adn Torre (URAS) model [URA88].

Differential methods also have been extended including color information [AND03] and their limitation to luminance changes also has been addressed [KIM05]. Multiscale approaches with warping techniques (coarse to fine scales) can be used to avoid achieve more accurate estimations in the presence of large movements. This is usually solved on the framework of multiscale approaches [BUR83] to deal with this problem. This idea is illustrated in Figure 3.3. For example Simoncelli et. al [SIM99] uses probability distribution of optical flow and an extended Kalman filtering across spatial scales to improve the range and accuracy of the optical flow base on multiscale with warping scheme. In a different way, Weber and Malik [WEB95] combines the information extracted across scales based on confidence parameters.

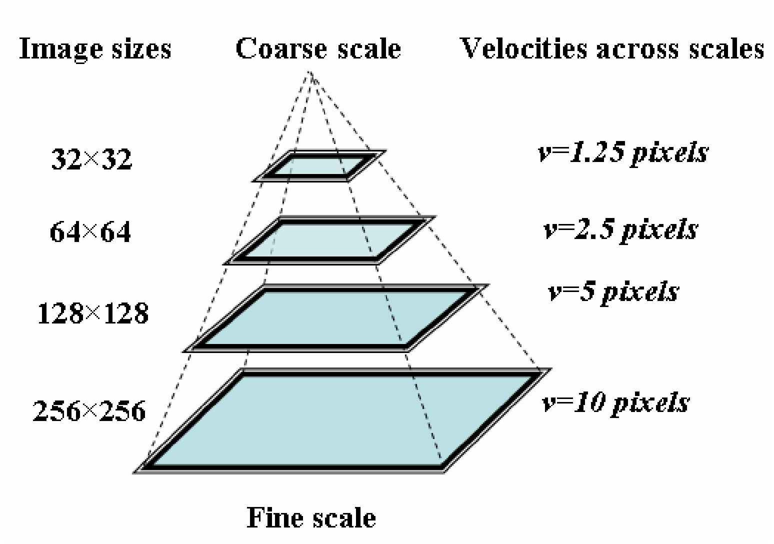


Figure 3.3. Multiscale concept. An image is reduced several times using lowpass filtering and subsampling to get an image pyramid, usually scaling sizes by 2. Motion speed is reduced at each scale until it can be considered small enough for the optical flow computing method. The velocity information can be combined across scales or we can use it iteratively to warp the image from coarse to fine scales, reducing the motion range and avoiding models limitations.

3.4.2. Region-Based Matching

Due to noise, aliasing in the image acquisition process and other factors that reduce the quality of the original images, accurate numerical differentiation may be impractical. In this case, region-based matching ([ANA89], [CAM97], [SIN92a]) become a very valid option. These approaches define velocity v as the shift *d=(d$_x$, d$_y$)* that yields the best fit between image regions at different times (along a sequence). Therefore, these approaches require "distance measure" (such as the sum-of-squared difference, SSD) which needs to be minimized (as estimation of maximum similarity):

$$SSD_{1,2}(x;d)=\sum_{j=-n}^{n}\sum_{i=-n}^{n}W(i,j)\times[I_1(x+(i,j))-I_2(x+d+(i,j))]^2=W(x)*[I_1(x+d)]^2 \qquad (3.6)$$

where *W* denotes a discrete 2-D window function, and *d=(d$_x$,d$_y$)* takes on integer values. This approach is similar to the differential-based techniques, since the difference in equation (3.6) can be viewed as a window-weighted average of the first-order approximation of the temporal derivative of *I(x,t)*.

The comparative study of Barron et al. [BAR94] includes two region-based algorithms:

- Anadan model [ANA87], [ANA89].
- Singh model. A two stage matching approach [SIN90], [SIN92b].

3.4.3. Energy-Based Methods

This class of methods is based on the output energy of velocity-tuned filters [ADE86], [BIG91], [HAG92], [HEE88], [GRZ90], [SIM98], [DIA03], [ROS04]. They are also called frequency-based methods because they require the design of velocity-tuned filters in the Fourier domain [ADE85], [FLE92], [WAT83]. The Fourier transform of a translating 2-D pattern (3.4) is the following:

$$\hat{I}((k,\omega)=\hat{I}_0(k)\delta(\omega+v^T k) \qquad (3.7)$$

where $\hat{I}(k)$ is the Fourier transform of $I(x,0)$, $\delta(k)$ is a Dirac delta function, ω denotes temporal frequency, and $k=(k_x,k_y)$ denotes spatial frequency. Interestingly, it has been shown that certain energy-based methods are equivalent to correlation-based methods [ADE85], [SAN85] and to the gradient-based approach of Lucas and Kanade [ADE86], [SIM93].

The comparative study of Barron et al. includes only the model proposed by Heeger [HEE87], [HEE88].

3.4.4. Phase-Based Techniques

The name of this class of methods comes from the velocity definition in terms of phase behaviours of band-pass filter outputs [FLE90], [FLE92], [GAU02], [WAX88]. The comparative study of Barron et al. [BAR94] includes two phased-based methods:

- Waxman, Wu and Bergholm [WAX88].
- Fleet and Jepson [FLE90].

Note that for some of these models, see for example [FLE90], [FLE92] and [GAU02], the use a first order Taylor expansion of the phase in a very similar way than gradient models as [LUC81]. This manifest that the difference between some gradient and phase models rely in that gradient uses the luminance (energy) of the signal whilst phase model use the phase information.

The comparative study done by Barron et al. [BAR94] uses a set of synthetic sequences with known ground-truth for the performance evaluation of the different approaches under consideration. These sequences are available in [VIS06] and have been widely used to evaluate other approaches. The confidence measured used for each approach is proven to be crucial for obtaining reliable (accurate) estimations. Therefore, certain approaches with appropriated confidence measures achieve very high performance if the unreliable estimations are discarded effectively.

The conclusions of the comparative study highlighted differential techniques and phased-based methods as the most reliable models for estimating optic flow. Concretely, the first-order local differential method of Lucas and Kanade [LUC81], [LUC84] and the local phase-based method of Fleet and Jepson [FLE90] were shown as the best methods. Only these methods performed well over all the image sequences tested by Barron et al. [BAR94]. It was reported the importance of very good confidence measures for these methods (as the size of the smallest eigenvalue of the normal equations (left expression in (3.10) in the framework of the Lucas and Kanade model). It was also reported the importance of accurate numerical differentiation and spatiotemporal smoothing. In particular, some degree of spatiotemporal presmoothing to remove small amounts of temporal aliasing and spatial discontinuities are shown to be very relevant improvements for most of the original algorithms. Finally, most of the implementations considered in this study involved only one scale of filtering and would significantly improve with multiscale implementations. This is true of most techniques, including the ones that achieved the best results (already using a single scale), such as Lucas and Kanade [LUC81], [LUC84] and the phase-based approach of Fleet and Jepson [FLE90], [FLE92].

3.4.5 Other approaches

There are other approaches which are difficult to classify in the previous types. For example, Nestares et. al [NES01] uses an 3-D directionally oriented bandpass to reduce speed over channels and combine them on a reliable way. These methods combine the ideas from energy models, such as [SIM98] with a differential method such as in [SIM99].

On [BRU05a] the advantages of local and global differential techniques are combined to achieve high optical flow accuracy, avoiding the over-smoothed optical flows produced by global methods but solving the local constraint inherent to local approaches. There are also modifications [BRU05b], where a varational method is used to speed up the optical flow in standard processors achieving near real-time performance.

In [GIA97] they present an interesting method. Using a feed-forward scheme, images are warping based in previous optical flow estimations to enlarge the algorithm velocity range. This method solves the limitation of gradient optical flow approaches without requiring multiscale techniques, which are complex to implement in customized hardware devices.

Finally, a variation of [NAG86] based on an anisotropic diffusion method with a diffusion tensor has also been proposed by Alvarez et. al [ALV00]. This method creates flow fields with 100 % density over the entire image domain and it can recover displacement fields that are far beyond the typical one-pixel limits which are characteristic for many of the differential methods commented before.

3.5. Accuracy vs. Efficiency

There are many potential application fields in which optical flow is of high interest. But in order to be used, the estimations need to be computed and delivered efficiently (with short response time). Nevertheless, if we make up a new algorithm that is very simple and gives optic flow estimations very fast, it will only have interest if it also accurate enough to allow a subsequent interpretation of the results. Therefore, both accuracy and efficiency are important as far as real world applications are concerned. There are specific studies such as [LIU98] that address the evaluation of *accuracy vs. efficiency* (AE) trade-offs of different optic flow approaches. These studies conclude that the L&K approach achieves high accuracy at different densities and requires affordable computing resources. The study indicates other simpler approaches that are easier to be implemented in real-time but at the cost of achieving lower accuracy. But in our research work, since we have implemented specific processing datapaths we address the implementation of a high accuracy model (L&K) taking advantage of dedicated computing resources. In [BAI97] this method is also compared with other differential approaches (using for example higher derivative order) and concludes that the least squares fitting with first order derivatives of L&K present the best results. Finally, McCane [MCC01] also give L&K a good score and conclude that the computational power required by this approach is affordable. This has prompted later researchers to focus on the L&K algorithm [BAK04].

3.6. Description of the Lucas & Kanade model

Although the original algorithm was proposed as a method to estimate the disparity map in stereo-pair images [LUC81], we have applied Barron's description of the L&K algorithm to optical-flow computation [BAR94]. We have implemented different versions (see Chapter 6) with several modifications to improve the accuracy of the original model and the feasibility of its hardware implementation.

In the following equations we describe briefly the computations upon which the L&K approach is based. A detailed description of the L&K model is provided in [BAR94], [LUC81].

The algorithm belongs to gradient-based techniques characterized by a gradient search performed on extracted spatial and temporal derivatives. Upon the assumption of constant luminance values through time, the first-order gradient constraint equation (3.8) is obtained as:

$$\nabla_{xy} I(x,y,t) \cdot (v_x, v_y) + I_t(x,y,t) = 0 \tag{3.8}$$

This equation only allows us to estimate velocity in the direction of maximum gradient, i.e. in the normal direction of moving surfaces. To overcome this limitation the L&K method constructs a flow estimation based on the first-order derivatives of the image. By least-square fitting, the model extracts an estimation of motion based on the hypothesis of similarity of velocity values in the neighborhood of a central pixel. This is described mathematically by expression (3.9):

$$\min \sum_{x \in \Omega} W^2(x) \left[\nabla_{xy} I(x,y,t) \cdot (v_x, v_y) + I_t(x,y,t) \right]^2 \tag{3.9}$$

where *W(x)* weights the constraints with values near the centre of the spatial neighborhood Ω.

The known solution to this problem is expressed in equations (3.10) to (3.12).

$$\vec{v} = \left[A^T W^2 A\right]^{-1} A^T W^2 \vec{b} \tag{3.10}$$

Where:

$$A^T W^2 A = \begin{bmatrix} \sum\limits_{x\in\Omega} W^2 I_x^2 + \alpha & \sum\limits_{x\in\Omega} W^2 I_x I_y \\ \sum\limits_{x\in\Omega} W^2 I_x I_y & \sum\limits_{x\in\Omega} W^2 I_y^2 + \alpha \end{bmatrix} \tag{3.11}$$

$$A^T W^2 \vec{b} = \begin{bmatrix} -\sum\limits_{x\in\Omega} W^2 I_x I_t \\ -\sum\limits_{x\in\Omega} W^2 I_y I_t \end{bmatrix} \tag{3.12}$$

An inherent limitation to these models appears in *blank wall* or *aperture problem* situations. In these cases the problem has no solution (matrix $A^T W^2 A$ is not invertible) and the model cannot provide any estimation of motion. To overcome this we have added a small constant, α, to the matrix diagonal, as suggested in [SIM91], which allows us to estimate the normal velocity field in situations where 2-D velocity cannot be extracted due to the lack of contrast information. In summary, we have to compute the 2x2 matrix of equation (3.11), its inverse, and the 2x1 matrix indicated in equation (3.12) and the velocity values using equation (3.10). The value of α is set to zero in the original model [LUC81], [BAR94], in fact not even defined in the original model.

Before computing the image derivatives in the matrix elements of equation (3.11), the images are pre-processed by Gaussian smoothing, which reduces image noise and generates a higher correlation between adjacent pixels. Typically, Gaussian space-time filters of 2 pixels variance plus a temporal derivative of 5 pixels are used. All the temporal operations require storage of 15 images for the entire process. This is hardly affordable in embedded hardware systems; therefore, as indicated in [FLE95], an alternative tactic can be implemented by using IIR temporal recursive smoothing and derivative filters. In this way the temporal storage requirement is reduced to 3 frames and the computation time improved at a cost of only slightly reduced accuracy. The temporal filter can be computed as follows in the next subsection.

3.6.1. IIR filters for the L&K algorithm

Let us consider a separable space-time smoothing filter. After the spatial filtering operation we can use a causal temporal filter based on a truncated exponential.

$$E(t) = \begin{cases} \exp(-t/\tau)/\tau & t \geq 0 \\ 0 & t < 0 \end{cases} \tag{3.13}$$

where τ is the time constant of the filter. The temporal derivative of the images can be calculated using this filter. The digital filter equations described in [FLE95] are:

$$\begin{aligned} &w(t) = I(t) - 2rw(t-1) - r^2 w(t-2) \\ &R_2(t) = q^2 w(t) + 2q^2 w(t-1) + qw(t-2) \\ &y(t) = R_2(t) - ry(t-1) \end{aligned} \tag{3.14}$$

where we store and update: w(t-1), w(t-2), y(t-1). The parameters *q* and *r* are calculated from *τ* according to equation (3.15):

$$q = \frac{1}{1+2\tau} \qquad r = \frac{1-2\tau}{1+2\tau} \tag{3.15}$$

Finally, the smoothed temporal image and its derivative are computed with equation (3.16):

$$\begin{aligned} &I_{smooth}(t) = qy(t) + qy(t-1) \\ &I_t(t) = (I_2(t) - I_{smooth})/\tau \end{aligned} \tag{3.16}$$

The design strategy (IIR vs. FIR temporal filters) relay on the number of taps required for some specific environments and tasks because difference in hardware resources is not very significant (thought memory availability could be critical). If low frame rate cameras are used, we need to filter temporal high frequency information to avoid aliasing and therefore a larger number of taps is required, making the IIR approach a better option (external memory resources are not augmented). Nevertheless, it means that the motion and light conditions should be continuous and this is not always a realistic situation. Therefore, we consider that the use of high frame-rate cameras with FIR filters is a better choice if these sensors are available. It provides higher accuracy and can be used on more generic environments. In Chapter 6 both implementations are described and compare their accuracy with a benchmark synthetic sequence.

3.6.2. Improved gradient-based optical flow estimation. Modified Lukas & Kanade algorithm

The previous sections present the L&K model as a good candidate for real-time optical flow computation. The L&K algorithm belongs to gradient-based techniques which means that the estimation of pixel velocities is based on image derivatives and the assumption of constant luminance over a temporal window is required.

Most of the literature works utilize the derivative kernels and model parameters presented in [BAR94] but, as J. Brandt described in [BRA97], they are suitable of significant improvement. We encourage the reading of that work for the correct understanding of such modifications. To summarize, the processing stages developed in our system (also explained in Section are the following:

1. Pre-filtering with a separable kernel of 3x3x3, P=[1, 2, 1]/4 The utilization of this small smoothing kernel allow high optical flow estimation density because it does not reject the high frequency terms and at the same time also contributes as anti-aliasing filter.
2. Complementary derivative kernels (2-D smoothing and 1-D derivation for each axe derivative) as designed by Simoncelli [SIM94]. These kernels increase the computation architecture complexity compared with other approaches [DIA04, DIA06] but significantly improve the accuracy of the system [BRA97]. In terms of performance, they represent a computation load increment of a factor of 3 but this is not a problem when designing customized hardware because it can be implemented in the pipeline structure without throughput degradation.
3. The image derivatives I_x, I_y and I_t (subscript stands for axe direction derivative) are cross-multiplied to get the five products $I_x \cdot I_x$, $I_y \cdot I_y$, $I_x \cdot I_y$, $I_x \cdot I_t$, and $I_y \cdot I_t$ and then are locally weighted on a neighborhood area Ω. The weighing operation is implemented as separable convolution operations over the derivatives products using the 2-D spatial central-weighting separable kernel [0.0625, 0.25, 0.375, 0.25, 0.0625].
4. Finally, the weighted image derivatives products are combined to get each pixel velocity estimation [BAR94].

The overall support of the system is 11x11x7 pixels using the parameters described above, thus just 7 images storage is required which is feasible on systems embedded on a single chip. This makes this approach affordable in specific hardware, even without adopting the alternative IIR temporal filters proposed by Fleet et al.

[FLE95], and it achieves higher accuracy when using restricted fixed-point bit-width to compute the filter values.

Table 3.1. Accuracy evaluation of different implementations of the L&K model using the Yosemite flow-through synthetic sequence with known ground truth. The error measure is described in [FLE92]. Note that the confidence thresholds are slightly different for each version to arrive at similar optical flow densities. The threshold parameter used is the product of eigen-values because it is hardware friendly and it gives a good error discrimination sensibility [BRA97].

Sequence	**Average Error (º)**	**Standard deviation (º)**	**Density (%)**
Standard L&K implementation (from [BAR94], using minimum eigenvalue as confidence parameter)	4,28	11,41	35,1
Standard L&K implementation (using eigenvalues product as confidence parameter)	4,57	12,77	36,44
L&K using IIR temporal filters suchs as described on [FLE95]	6.4716	13.0057	36,46
Improved L&K (based on [BRA97])	3.3757	8.9263	36,44

In Chapter 6 are described different hardware implementations of the L&K model at different accuracy vs. efficiency trade offs. In the most accurate one we adopt the first two modifications of the original L&K implementation that improve the motion accuracy as well as the density of the flow, as can be seen on Table 3.1. Furthermore, for the spatial integration of constraints we have used the spatial kernel *[0.0625, 0.25, 0.375, 0.25, 0.0625]* instead of the *[0.2, 0.2, 0.2, 0.2, 0.2]* commented by [BRA97] because enhance the accuracy. The overall support of the algorithm is reduced from 19x19x15 pixels [BAR94] to 11x11x7 pixels with Brandt modifications [BRA97], thus just 7 images storage is required. In other implementations (see Chapter 6) we use the Fleet et al. [FLE95] IIR temporal filter which requires only 3 images storage. The drawback of that approach is that IIR filters produce lower accuracy in the estimated optical flow and need higher fixed-point bit-width to compute filter values. The 7 images storage requirement of the modified L&K model is less than half of the previous Barron approach [BAR94] and feasible on embedded systems.

All the accuracy tests commented above are conditioned by the used reliability test parameter. The equations to estimate the pixels velocities are often under-determined or ill-conditioned because the gradient direction does not vary sufficiently within the integration neighbourhood. Several measures can be used to detect these situations [BAI96]. According to the results from Brandt [BRA97] we use the product of the eigenvalues of the matrix of equation (3.10) since it provides similar results than the well known minimum eigenvalue criteria from Barron et al. [BAR94]. Furthermore, it is more hardware-friendly because it corresponds to the determinant of equation (3.10) already computed in the motion estimation datapath (see Chapter 6 for hardware implementation description).

3.7. Conclusions

In this Chapter we have defined optic flow and briefly revised different alternative to compute it. This represents a summary of comparative studies about the different optic flow methods evaluating their accuracy and efficiency (computational requirements). This chapter represents an review of the state of the art in optic flow methods, with the goal of selecting a candidate model to be implemented in real-time designing specific purpose processing datapath (as will be described in Chapter 6).

The final conclusion of this chapter is that although there are different optic flow methods that achieve high accuracy ([BRU05a], [ALU00], [FLE90, FLE92]), concretely the model of Lucas & Kanade represents a very good candidate for several reasons: a) this approach achieves high accuracy, b) it has an efficient and well defined confidence measure easy to extract from temporal variables required by the model (no extra computation required for this purpose) and c) it requires affordable computing resources.

Chapter 4
Stereo vision processing models

This chapter briefly reviews different techniques used in computer vision for stereo disparity estimation. We describe briefly the camera calibration process and also how to use stereo for 3D reconstruction but the main contribution of this chapter is the comparative discussion of different image correspondence methods for stereo disparity estimation. We use this analysis to choose a proper hardware friendly stereo processing model based on local phase shift. This is the approach in which is focused Chapter 7 for designing a real-time stereo-vision hardware architecture. Here we explain the main motivations and the characteristics of this model. We describe the algorithms in which it is based and the main advantages of this model with respect to other ones described in the literature.

4.1. Introduction

The problem of determining 3-dimensional structure of a scene from two or more images taken from distinct viewpoints is addressed on this chapter. Extraction of three-dimensional scene structure has been an intense area of research for decades, with an extensive literature available such as computer vision books [FAU93], [TRU98], [FAU01], [HAR04] and comparative studies about the different topics involved, for example different image correspondence techniques are addressed in [BRO92], [ASC93], [SCH02], [BRO03] and camera calibration in [ZHA98], [HEM03], [SAL02], [ARM03]. Stereopsis is useful in many visual domains such as autonomous navigation, three-dimensional reconstruction, active tracking or face recognition [TSA05], [ZIY00], [BER98], [OIS03].

This is information with strong biological fundaments. This task is accomplished in the visual cortex by a specialized receptive field structure [DEA91]. Significant studies have shown that a substantial proportion of neurons in the striate and extrastriate cortex of monkeys have stereoscopic properties; that is, they respond differentially to binocular stimuli, thus providing cues for stereoscopic depth perception ([HUB62], [BAR67], [DEA98]).

We will focus on the case of two cameras, which is called *binocular stereo vision or binocular stereopsis*. The fundamental basis for stereo is the fact that a single three-dimensional physical location projects to an unique pair of image locations in two observing cameras, as illustrated in Figure 4.1. As a result, given two camera images, if it is possible to locate the image locations that correspond to the same physical point in space, then it is possible to determine its three-dimensional location.

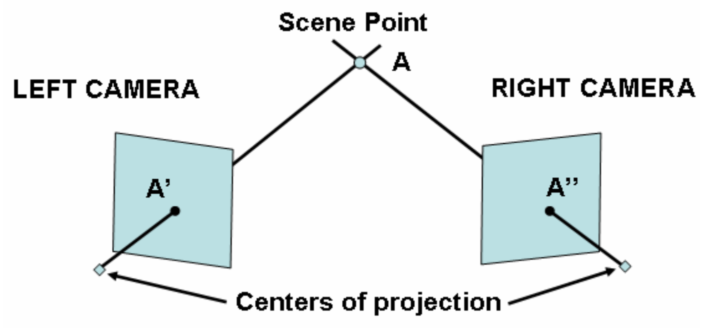


Figure 4. 1. Binocular scene geometry: The projection of a spatial point *A* into the left and right images is depicted by points *A'* and *A''*, respectively.

The primary problems to be solved in computational stereo are *camera calibration*, *image pair correspondence* and *scene reconstruction*:

1. *Camera Calibration* is the process of determining camera system external geometry (the relative positions and orientations of each camera) and internal geometry (focal lengths, optical centers and lens distortions). Accurate estimates of this geometry are necessary in order to relate image information (expressed in pixels) to an external world coordinate system.
2. The *image pair correspondence* consists on determining the locations in each camera image that are the projection of the same physical point in space. This vector displacement field that we compute is called *disparity map*. Therefore, our goal is given two images, and from the information contained in them, we must compute the disparities field. No general solution to the correspondence problem exists, due to ambiguous matches (e.g., due to occlusion, specularities, or lack of texture). Thus, a variety of constraints (e.g., epipolar geometry) and assumptions (e.g., image brightness constancy and surface smoothness) are commonly exploited to make the problem tractable.
3. The *scene reconstruction problem* consists of determining 3-dimensional structure from a disparity map, based on known camera geometry. Based on the parameters extracted during the calibration camera stage and using the disparity field computed using the image correspondence techniques, for each point of the space we can compute its depth in the three dimensional space. This process is called *triangulation*.

These topics are further explained on section 4.2. The relations between optical flow and stereo computation should also be taken into consideration. Both methods are based on the correspondence between images and that is the reason that several authors look for generic techniques feasible to be used on both vision modalities [VAL96]. Nevertheless, the goals and problems to solve are quite different. On one hand, for optical flow we are interested on extracting motion information with high accuracy. This can be done properly if the time dimension is properly sampled. The movements usually involve small image displacement and very high spatial resolution is not of significant importance. Despite of that, the computing power is of significant importance because we use 2-D search and high frame-rate is desirable. On the other hand, for Stereo vision computation different problem and specifications are considered. The quality of the 3-D reconstruction is directly related with the cameras calibration, large image resolutions as well as large correspondence search areas are wanted. Furthermore, the differences between cameras due to imbalance on their parameters (for example differences on cameras luminance gain) can significantly affect some of the stereo correspondence methods whilst in the optical flow domain, with only one camera, this problem would be related to the temporal illumination changes.

4.2. Stereo Vision Basics

The problem of estimating calibration is at this point well understood, and high-quality toolkits are available (e.g., [BOU06] and links therein). This method characterize each camera as a projective device which means that scene points are projected through the camera focus to the projective plane. This is modeled as a projection matrix which converts a real word 3-D point to a 2-D camera point. The matrix which involves the intrinsic (camera origin, focal length, pixel size or exe deviation) and extrinsic (rotation matrix, displacement vector) parameters is called *Fundamental Matrix,* and calibration is the process to obtain this matrix. This topic is not addressed on this Thesis. For interesting discussions of recent work on calibration, see [HAR04] and [FAU01].

Concerning to cameras configuration, we consider the arrangement shown in Figure 4.2. This schematic illustrates how distance is calculated from disparity. The depth of a point A in a three-dimension space imaged by two cameras with optical centers C_l and C_r is defined by intersecting the rays from the optical centers through their respective images of A, which are A' and A''. The baseline of the stereo pair is defined to be the line segment joining the optical centers. For non-verge geometry as schematically shown on Figure 4.2, this line is parallel to the camera x coordinate axis. With this camera configuration, the disparity d is defined as the shift between two corresponding points which can be expressed by equation (4.1).

$$d = \mathrm{x} - \mathrm{x}' \tag{4.1}$$

The set of all disparities is called a *disparity map*. From that, depth, Z, can be calculated as:

$$Z = f\frac{T}{d} \tag{4.2}$$

where f is the camera focal length and T is the distance between the optical centers of the two cameras, C_l and C_r. This is the process called triangulation. Equation (4.2) shows that given a fixed focal length and camera separation, distance is inversely proportional to disparity. So, the larger the disparity, the closer the scene point is to the cameras.

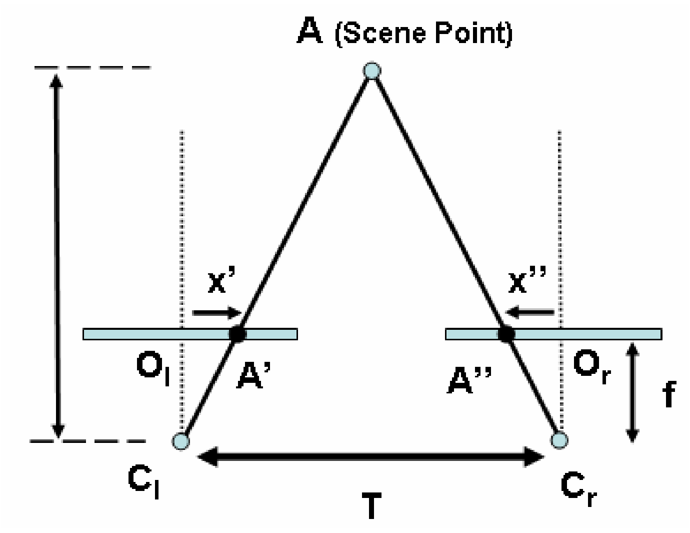


Figure 4.2. The geometry of non-verged stereo, computing depth from disparity. The optical centers are C_l and C_r. The point A in the scene is imaged by the left and right cameras respectively as points A' and A''.

To solve the correspondence problem, the first approach is to pick one pixel in one image and then search through a 2-D region around that pixel location in the other image to find the corresponding point. However, it can be shown that a 1-D search is sufficient due to the *epipolar constraint*. This constraint guarantees that if A' is the projection of a scene point A in one image, then the corresponding point, A'' in the other image will lie on a straight line, *epipolar line*, which is the intersection of the image planes with a plane that contains point A' and the two centers of projection.

In a pair of cameras that are vertically aligned and have parallel optical axes, the epipolar line will coincide with a scan line of the image (Figure 4.3). This property makes the search process simpler compared with non-horizontal epipolar lines. This will be of great advantage for a hardware implementation because it makes feasible data-flow structures without complex memory management architectures.

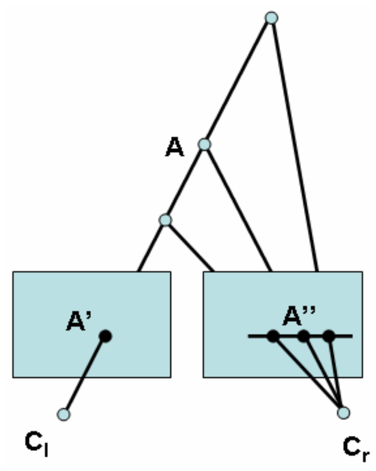


Figure 4.3. Epipolar constraint for non-verged cameras.

In practice, it is difficult to build stereo systems with non-verged geometry. However, it is well-known that arbitrary stereo image pairs (i.e., with verged geometry) may also be rectified (resampled) to non-verged geometry by the *epipolar constraint*. This is illustrated on Figure 4.4. A point *A* in the scene is imaged by the left and right cameras respectively as points *A'* and *A''*. The baseline *T* and optical rays C_l to *A* and C_r to *A* defined a plane of projection for the point A, which called the *epipolar plane*. This plane intersects the image planes in lines called *epipolar lines*. The epipolar line through a point *A''* is the image of the opposite ray, C_l to *A* through point *A''*. The point at which an image's epipolar lines intersect the baseline is called the *epipole* (*e'* and *e''* for *A'* and *A''* respectively), and this point corresponds to the image of the opposite camera's optical center as imaged by the corresponding camera. Given this unique geometry, the corresponding point *A''* of any point *A'* may be found along its respective epipolar line. By rectifying the images in such a way that corresponding epipolar lines lie along horizontal scan-lines, the two-dimensional correspondence search problem is again reduced to a scan-line search, greatly reducing both computational complexity and the likelihood of false matches. In this situation (Figure 4.3), the epipoles are in the infinite. See [TRU98] or [CVO06] for details on algorithms to compute this rectification.

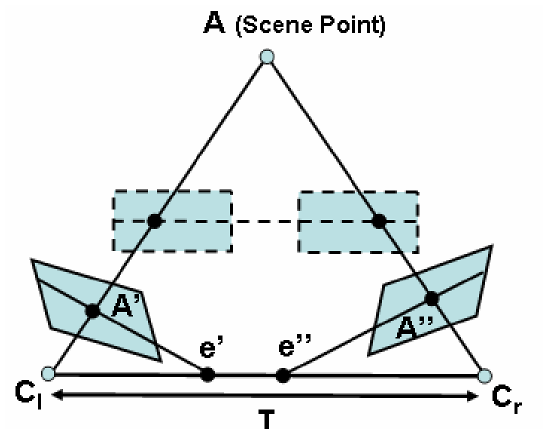


Figure 4.4. Two arbitrary images of the same scene may be rectified along epipolar lines (solid) to produce collinear scan lines (dashed). (Adapted from [BRO03]).

The calibration process is usually a stage that is done only one time before running the stereo system (for example in a grasping application domain as shown in Figure. 4.5). Therefore, camera calibration is not a topic related with real-time issues. After calibration, rectification process improves 1-D matching but this modification can also introduce artifacts. Because of this and also for simplicity, careful camera alignment is usually preferred on robotics approaches [FRO96]. Nevertheless, this process can be easily introduced into the frame-graber for a customized DSP but we have not addressed this topic. In this chapter and the following we have made the assumptions of cameras with the same focal length and vertically aligned with parallel optical axes. On that way, camera calibration issues and image rectification are not addressed here because they are out the scope of the presented contribution.

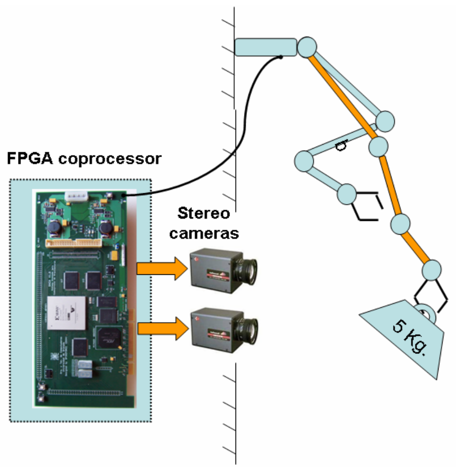


Figure 4.5. Example scenario of binocular stereo-vision application. Grasping task application with a robotic arm. The camera calibration process in this scenario can be done before stereo image processing without requiring any modification in the calibration parameters during working time.

4.2.1. Stereo Vision quality metrics

The evaluation of the performance of a stereo algorithm allows ranking the different approaches based on their accuracy and evaluating also potential target application fields. Two general methodologies are used:

a. Computation of the error statistics with the ground truth data or real images (for of natural well calibrated scenarios) [SCH02].
b. Evaluation the error using synthetic images obtained by warping the reference image by the computed disparity map [SZE99].

The first method is more common and databases of images with known ground truth are available in the internet [SCH06], [GER06]. There is a large number of quality metrics used on the literature, for example:

1. RMS (root-mean-squared) error (measured in disparity units) between the computed disparity map and the ground truth.
2. Percentage of bad matching pixels, with error larger than a predefined threshold.

Furthermore, these metric can be measured over different image types and specific areas within these images, such as textureless regions, occluded regions and depth discontinuity regions which allow evaluating the performance of the different methods under very specific situations. More details can be found in [SCH02].

4.3. Image pair correspondence methods

As mentioned before, stereo disparities may be determined in a number of ways and by exploiting a number of constraints. All of these methods attempt to match pixels in one image with their corresponding pixels in the other image.

Although the correspondence problem is also discussed for motion computation in Chapter 3, as commented in Section 4.1 the specific characteristics and constraints presented in stereopsis make necessary a new analysis of the different correspondence methods.

For simplicity, we refer to constraints on a small number of pixels surrounding a pixel of interest as *local constraints*. Similarly, we loosely refer to constraints on scanlines or on the entire image as *global constraints*. A good review about this topic is presented in [BRO03] and [SCH02]. Table 4.1 summarizes the main methods for exploiting both local and global constraints, focusing on binocular stereo methods.

Table 4.1. Summary of the most common image correspondence techniques applied in stereopsis.

TECHNIQUE	REFERENCES	DESCRIPTION
LOCAL METHODS		
Region-Based Matching	[KAN94, [COR97], [MUH02]	Search for maximum match score or minimum error over a small region, typically using variants of cross-correlation or robust rank metrics.
Differential Techniques	[LUC81], [SIM91], [NAG83]	Minimize a functional, typically the sum of squared differences, over a small region.
Energy or Phase Based Techniques	[SAN88], [FLE94], [FLE96], [SOL01], [FEL02b], [CHE04]	Analyze the image on the Fourier domain focussing on the energy or phase of the signal after convolving with quadrature filters.
Feature Matching	[BEN02], [KRÜ04]	Match specific features rather than intensities themselves.
GLOBAL METHODS		
Dynamic Programming	[BEL96], [BOB99], [FOR04]	Determine the disparity surface for a scanline as the best path between two sequences of ordered features. Typically, order is defined by the epipolar ordering constraint.
Global optimization	[TER86], [ROY98], [BOY01]	Determine the disparity surface as energy minimization process.

4.3.1. Local search correspondence methods

Region-Based Matching also called Block matching methods seek to estimate disparity at a point in one image by comparing a small region around that point (the template) with a series of small regions extracted from the other image (the search region). Differently to what happens for optical flow, the epipolar constraint reduces the search to one dimension. These methods have extensively been used for real-time approaches as [COR97], [KAN94] because they fit in well with specific hardware architectures (highly parallelizable, fits well on fixed point arithmetic, well known at the algorithmic level, etc.). Several classes of metrics are commonly used for block matching:

- SAD, Sum of Absolute Differences. This technique is often used for computational efficiency. Based on the intensity differences.
- SSD, Sum of Squared Differences. Simpler than NCC (see below), narrower tuning curves than SAD, and it can also be normalized.
- NCC, Normalized Cross Correlation. This is the standard statistical method for determining similarity. Its normalization, both in the mean and the variance, makes it relatively insensitive to radiometric gain and bias at the cost of higher computing power requirements.
- Local non-parametric transforms: SAD based on rank transform or Censos transform with Hamming distance as matching technique [ZAB94]. This transform approximately eliminates sensitivity to radiometric bias or gain but reduces the accuracy.

This similarity functions can also be extended to include color information [MUH02]. This method achieves good performance on image areas with enough structure but it does not produce results on bland areas. Furthermore, information of occlusion areas and boundaries has no any special treatment (which would significantly can improve the disparity map). An extensive comparison of these metrics has been done by Aschwanden and Guggenbuhl [ASC93].

Differential Techniques, [LUC81], seek to determine small local disparities between two images by formulating a differential equation relating motion and image brightness. In order to do this, the assumption is made that the image brightness of a point in the scene is constant between the two views. Then, the horizontal translation of a point from one image to the other is computed by a simple differential equation. Chapter 3 makes a review of these techniques. Their main drawback is that, in order to achieve large disparity estimation, multiscale approaches are required [BUR83] which significantly increases the architectural complexity. Furthermore, these techniques are affected by radiometric problems such as cameras gain luminance or illumination changes which make these approaches more suitable for only well controlled scenarios.

Energy and Phase Based Methods based on the signal properties on the Fourier domain are also a common technique for optical flow computation (see Chapter 3). Population coding based on quadrature filter energy can be applied combined with phase based approaches [FLE96] or [CHE04]. Approaches based only on Phase information have extensively been used for stereo disparity estimation [SAN88], [FLE91], [FLE94], [SOL01], [FEL02]. The advantage of phase based techniques has been highlighted by some contributions as [FLE93], [COZ97]. As commented in Chapter 2, phase information rely on structural information instead of intensity, this makes this approach unbiased to luminance change as well as it leads to a better behaviour against affine transformations (for instance due to different cameras perspectives). Furthermore, this approach allows direct subpixel disparity information without any post-processing as required for block-matching methods. This has motivated that real-time approaches based on this technique have recently been proposed [POR02], [DAR03] and [DAR06].

Feature matching techniques using some specific features such as corners, edges or even multimodal information to implement the matching process have been recently proposed [BEN02], [KRÜ04]. This sparse map has the advantage of requiring less computational resources, allowing large region search and because of that it has been widely used in the past when there was not possible to estimate dense disparity maps on a reasonable computing time. Furthermore, features are more stable signals than just image regions and do not suffer from luminance change problems. The main drawback is that the disparity map that they produce is very sparse which is not suitable for applications such as 3-D reconstruction but it is still interesting for embedded system applications when only small computer power is available.

4.3.2. Global search correspondence methods

In contrast to local methods, global methods try to compute the disparity field based on prior assumptions such as disparity smoothness. Keeping that in mind, many global methods are formulated in an energy-minimization framework [TER86].

The objective is to find a disparity function *d* that minimizes a global energy as expressed in equation (4.3).

$$E(d) = E_{data(d)} + \lambda E_{smooth(d)} \qquad (4.3)$$

The data term, $E_{data(d)}$ measures how well the disparity function *d* agrees with the input image pair. This is given by equation (4.4).

$$E_{\text{data(d)}} = \sum_{(x,y)} C(x, y, d(x, y)) \tag{4.4}$$

where *C* is the matching cost function which evaluates how well the image pairs are matched. The smoothness term $E_{\text{smooth(d)}}$ encodes the smoothness assumptions made by the algorithm. To make the optimization computationally tractable, the smoothness term is often restricted to only measuring the differences between neighboring pixels' disparities. This is mathematically formulated in equation (4.5)

$$E_{\text{smooth(d)}} = \sum_{(x,y)} \delta\big(d(x, y), d(x, y+1)\big) + \delta\big(d(x, y), d(x, y-1)\big) \tag{4.5}$$

where δ is a monotonically increasing function of disparity difference.

The kind of functions used for C and δ determine the different types of global optimization approaches such as max-flow [ROY98] or graph-cuts [BOY01]. For a review and comparison of different approaches see [SCH02] and [BRO03].

A different class of global optimization algorithms is composed by the ones that are based on *dynamic programming*. This technique can be applied to scanline optimization problem [BOB99], [BEL96]. These approaches work by computing the minimum-cost path through the matrix of all pairwise matching costs between two corresponding scanlines. Partial occlusion is handled explicitly by assigning a group of pixels in one image to a single pixel in the other image.

There are other global approaches such as the nonlinear diffusion and belief propagation ones also widely used on the literature. In fact, for some of these algorithms [SCH98] it is possible to explicitly state a global function that is being minimized. The description of these techniques is out of the scope of this chapter. We recommend the works [SCH02] and [BRO03] for a more complete review.

Global methods as the one presented here do not fit on regular dataflow structures and require an intensive iterative process across image areas or scanlines. This kind of processing cannot take full advantage of high parallelism and requires fast processors to achieve high performance. Therefore, they do not fit well in specific hardware architectures (which usually work at low clock frequency) but rather in standard processors that work at higher clock frequencies. According to this, recently has appeared a real-time systems based on this dynamic programming which used Graphic Processing Units *(GPUs)* as coprocessor to speed-up the stereo computation process [FOR04].

4.4. Hardware-friendly Phase-based Stereo

From the previous discussion we conclude that local methods are well suited for hardware implementation. Hardware-based approaches normally rely on correlation-based models because they fit in well with specific hardware architectures [BRO03]. Nevertheless, over the last decade phase-based computational models have been proposed as an interesting alternative [FLE93] against other methods mainly because they are based on local operations and produce dense depth maps with direct subpixel resolution. Furthermore, as shown in Chapter 2, phase information is quite stable to luminance changes and affine transformation which are very common problems due to unbalanced cameras parameters and makes this approach more suitable for reliable stereo computation than the others. This has motivated several real-time approaches based on these techniques [CRE98], [POR02] and [DAR06].

The adopted computing model has been proposed by Solari & Sabatini [SOL01]. In a first approach the positions of corresponding points are related by a 1-D horizontal shift, the disparity, along the epipolar line. Formally, the left and right observed intensities of the two eyes, $I_L(x)$ and $I_R(x)$, can be expressed in terms of equation (4.6):

$$I^R(x) = I^L[x + \delta(x)] \tag{4.6}$$

where $\pm\delta(x)$ is the (horizontal) binocular disparity. This is represented in Figure 4.6.

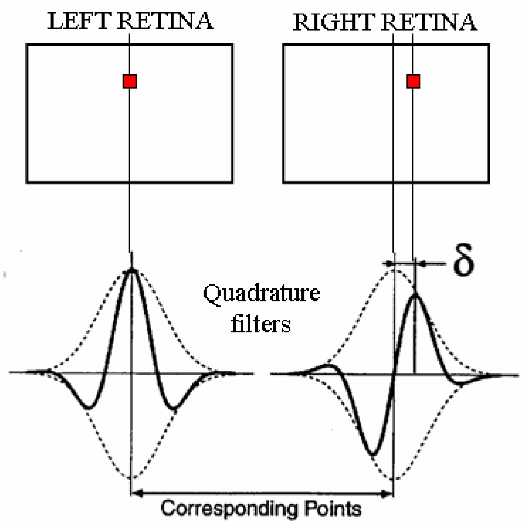


Figure 4.6. Phase-based disparity estimation using neurons with receptive fields as quadrature filters.

Disparity can be estimated in terms of phase differences in the spectral components of the stereo-image pair [POG84]. Since the two images are locally related by a shift, within the neighborhood of each image point the local spectral components of $I_L(x)$ and $I_R(x)$ are related by a phase difference shown in equation (4.7).

$$\Delta\phi(k) = \phi^L(k) - \phi^R(k) = k\delta \qquad (4.7)$$

Spatially localized phase measurements can be obtained by filtering operations with complex-valued, quadrature-pair, bandpass kernels (e.g. Gabor filters or Gaussian derivatives), approximating a local Fourier analysis of the retinal images. For example, if we consider a complex Gabor filter *(h)* with a peak frequency k_0 and corresponding Gaussian variance σ_2 can be described as equation (4.8).

$$h(x;k_0) = \exp(-x^2/\sigma^2 + jk_0x) = h_C(x;k_0) + jh_S(x;k_0) \qquad (4.8)$$

the resulting convolutions with the left and right binocular signals can be expressed as in equation (4.9):

$$Q(x) = \int I(\xi)h(x-\xi;k_0)d\xi = C(x) + jS(x) = \rho(x)e^{j\phi(x)} \qquad (4.9)$$

where $\rho(x)$ and $\phi(x)$ denote their amplitude and phase components, while C(x) and S(x) are the responses of the quadrature filter pair (C and S stand for cosine and sine respectively). Local phase measurements are stable, with a quasi-linear behaviour over relatively large spatial extents, except around singular points, where the amplitude of Q(x) vanishes and the phase becomes unreliable. This property of the phase signal yields good predictions of binocular disparities by equation (4.10).

$$\delta(x) = \frac{\phi^L(x) - \phi^R(x)}{k(x)} = \frac{\lfloor \phi(x) \rfloor_{2\pi}}{k(x)} \qquad (4.10)$$

where we note $\lfloor\ \rfloor_{2\pi}$ as the principal part of the argument (i.e. ϕ belongs to $[-\pi, \pi]$) and k(x) is the average instantaneous frequency of the bandpass signal, measured using the phase derivative from the left and right filter outputs (x subscripts indicate differentiation along the x axis):

$$\mathrm{k(x)} = \frac{\phi_x^{\ L}(x) + \phi_x^{\ R}(x)}{2} \qquad (4.11)$$

As a consequence of the linear-phase model, the instantaneous frequency is generally constant and close to the tuning frequency of the filter ($k(x) \approx k_0$), except near singularities, where abrupt frequency changes occur as a function of spatial position. Therefore, a disparity estimation at a point x is accepted only if $|(\phi_x - k_0)| < k_0\tau$, where τ is a proper reliability threshold.

It should be noted that equation (4.10) does not require the explicit calculation of the left and right phases and thus we can compute the phase difference in the complex plane directly using the following identities:

$$\begin{aligned} &\lfloor \phi(x) \rfloor_{2\pi} = \lfloor \arg(Q^L Q^{*R}) \rfloor_{2\pi} = \\ &\arctan 2(\mathrm{Im}(\arg(Q^L Q^{*R}), \mathrm{Re}(\arg(Q^L Q^{*R})) = \\ &\arctan 2(C^R S^L - C^L S^R, C^L C^R + S^L S^R) \end{aligned} \qquad (4.12)$$

This approach is hardware friendly. It avoids the logic dedicated to the wrap-around mechanism. Furthermore, it reduces the number of divisors by 50%. This is crucial since the precision required for the operations in the last stages is already significant, making the division an expensive operation. All this is achieved at the cost of more multiplications compared to other models that explicitly calculate the phase. Nevertheless, multiplications are not so expensive operations since current FPGAs include specific circuits for this purpose.

4.5. Conclusions

In this chapter we have briefly described the different issues related with stereo computation: camera calibration, images correspondence and 3-D reconstruction. We have focused on discussing different techniques for image correspondence in order to choose a hardware friendly model for this task. A brief overview of the methods described in the literature makes clear that local methods are the ones that can be structured in regular dataflows which is an important factor that makes them feasible on specific hardware. Furthermore, although region matching models are the most widely used in the literature, phase based techniques achieve high accuracy, allow subpixel resolution and are more robust to luminance changes. This has prompted us to choose a phase based hardware friendly approach proposed in [SOL01] to be implemented in specific hardware (see Chapter 7).

Chapter 5
Hardware architecture for phase, orientation and energy computation

This chapter describes the hardware architectures that we implemented for local energy, phase and orientation estimation. Based on the analysis of Chapter 2, the Steerable Second order Gaussian derivatives have been chosen as the best suited approach. The Haglund methods will be employed for interpolation image features at any orientation from a computed set of 8 images. We describe here the system accuracy vs. bit-width trade-off, we analyze the different architectural approaches and we describe the solution adopted. We finally show the global system resources consumption, performance and some features computation qualitative results.

5. 1. Introduction

Based on the analysis of Chapter 2, the Steerable Second order Gaussian derivatives have been chosen as the best approach. The Haglund methods will be employed for interpolation image features at any orientation from a computed set of 8 basic orientations. The computation stages described on Chapter 2can be summarized as follows:

1. Bidimensional separable convolutions. The Second order Gaussian derivatives G_{xx}, G_{xy} and G_{yy} and their Hilbert transforms H_{xx}, H_{xy}, H_{yx} and H_{yy} are computed. The kernel coefficients can be computed from equation (2.10), for details see Appendix B and [FRE91].
2. Filters steering. This is one of the main properties of Gaussian derivatives. From this set of 2-D filter outputs, a linear combination of them allows the computation of any orientation. We use 8 basic orientations, sampling the angle interval $[0,\pi[$ (half circumference) with a regular spacing of $22,5^{o}$. The quadrature filter at orientation θ, h_{θ}, is calculated using equation (5.1), where c_{θ} and s_{θ} are respectively the even and odd components of the filters.

$$h_{\theta} = c_{\theta} + js_{\theta} \,,$$
$$c_{\theta} = \cos^2(\theta)G_{xx} - \cos(\theta)\sin(\theta)G_{xy} + \sin^2(\theta)G_{yy} \tag{5.1}$$
$$s_{\theta} = \cos^3(\theta)H_{xx} - 3\cos^2(\theta)\sin(\theta)H_{xy} - 3\cos(\theta)\sin^2(\theta)H_{yx} + \sin^3(\theta)H_{yy}$$

3. Features computation. Energy, phase and orientation are computed as described on equation (2.28) but some operations and dependencies are not hardware friendly. Therefore, we have made some modifications that are described on the next section.

The effect of the arithmetic representation and bit-width quantization on the embedded system design is also addressed in this chapter. On one hand, current standard processors use floating point representation which has very large data range and precision. On the other hand, digital circuits usually use fixed point representation with a very restricted number of bits. Unfortunately, embedded systems with specific circuitry for each computation can only afford floating representation at critical stages and most of the computation should be done using fixed point arithmetic. The quantization effects should be properly studied to ensure that the data dynamic range is properly adjusted and the bit quantization noise is kept at a small value. We will compute this hardware quantization degradation using a standard measure which is the *Signal to Quantization Noise Ratio* SQNR, which is defined by equation (5.2).

$$SQNR = \frac{E(S)}{E(e_Q)} = \frac{\left(\sum_i S_i^2\right) / N}{\left(\sum_i (S_i - S_{qi})^2\right) / N} \tag{5.2}$$

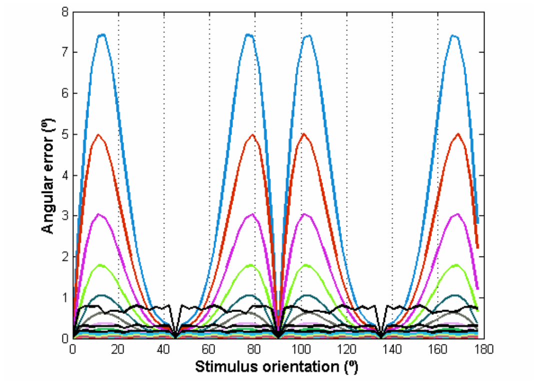
Where S_i are the data values computed using floating point representation, S_{qi} are the data values computed using fixed point representation, E() stands for the mean value operator, and N is the number of data. For the analysis described on the following sections we use this error metric to evaluate the quantization error of the different approaches.

5.1.1. Modifications for improving the hardware feasibility

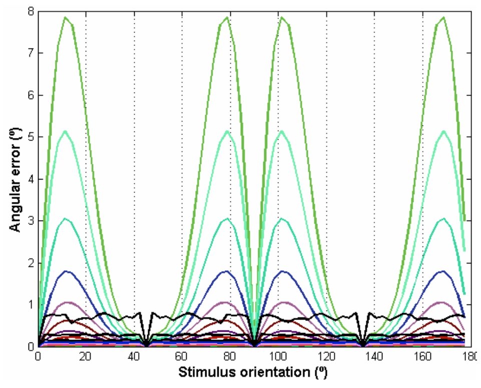
We will focus on the three main drawbacks or problems to overcome to arrive at an affordable high performance system:

I. *Magnitude/energy for orientation computation.* Equation (2.28.b) requires a square root operation which is difficult to compute on reconfigurable hardware.
II. *Image features dependencies.* Equation (2.28.b) uses the local energy extracted at each orientation, also computed on equation (2.28.a). Equation (2.28.c) uses the orientation computed at equation (2.28.b). This data dependency enlarges the pipeline datapaths and can be quite resource demanding.
III. *Arctan function computation.* It can be computed using *Look-Up-Tables* LUT based methods or the Cordic approach [VOL59].

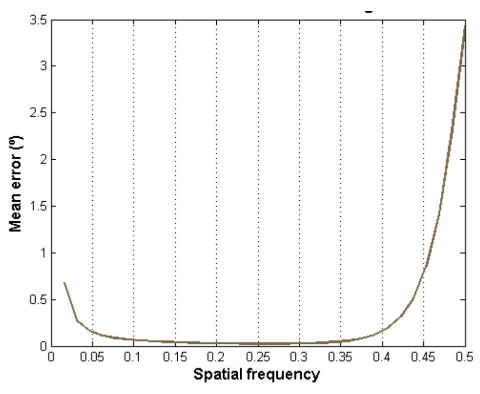
The first problem (I) has been solved using the energy instead of its square value on equation (2.28.b). Magnitude/energy works in this equation as weights for each orientation. The use of energy instead of magnitude implies a heavier weight for stronger outputs. If the filter tuned range is narrow enough, this modification does not affect significantly the orientation estimation. Figure 5.1 shows the orientation results (angular errors) when we use energy and magnitude as weights. This figure illustrates that there is almost no system degradation. We only have detected a small increase of 0.4º in the error estimation at very high frequency where the filters are not properly tuned. At other frequencies, the differences are negligible.

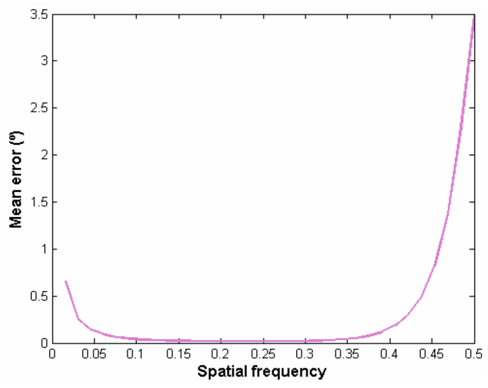


(a.1) Magnitude weights



(a.2) Energy weights





(b.1) Magnitude weights **(b.2) Energy weights**

Figure 5.1. Accuracy computation: energy vs. magnitude weights. (a) Error evolution across the different stimulus orientations, using sinusoidal gratings as test input. (b) Mean orientation error measured for each grating spatial frequency. Note that the differences of both approaches are negligible. Only a slight difference can be seen between the top curves of Figs 5.1.a but the filter is not tuned at this spatial scale and therefore this is not representative.

Using Figure 2.8.a, we have also measured the difference between original model and the modified version, considering this modification as a noisy signal. The SNR for this modification is 57.4 dB which is large enough to ensure a very high quality.

The problem (II) is related with the image feature dependencies. This point needs to take into consideration two aspects. First, the local energy is used for mean energy computation as well as for orientation estimation. Since local energy only requires even and odd filters squaring and addition, this is not a critical dependency because it can be solved just using a delay buffer. Second, the dependency between orientation and phase estimator circuits is more relevant. Orientation estimation involves the *arctan* function which usually requires a deep pipeline for high performance. The 8 even and 8 odd filters outputs require a buffering technique to make available these data when orientation is ready. Furthermore, the filters output should be accessible in parallel to maximize the circuits performance which requires a large number of memory resources or registers. This requires a large amount of resources and should be avoided. Following the indications of [HAG92], equation (2.28c) can be simplified as described on equation (5.3). The differences between the two choices, as indicated by [HAG92], are negligible. Furthermore, in the case of a one-dimensional signal there will be no difference at all between the methods. We will use the new method because is more hardware friendly since it does not suffer from features dependencies.

$$\begin{aligned} c &= \sum_i c_i \\ s &= \sum_i s_i \\ P_{local} &= \arctan\left(s/c\right) \end{aligned} \tag{5.3}$$

Finally, relative to the *arctan* function implementation (presented as problem III at the beginning on this section) the main goal is to optimize the accuracy vs. resources consumption trade-off. The techniques used for this issue are briefly explained in Sections 5.2 and 5.3.

5.2. Bit-width analysis

We illustrate in this section the main results produced by the *MCode for DSP Analyzer*, software and methodology described on Section 1.7.1. Since our goal in this chapter is to estimate three different features from the same primitives, we choose a common error metric for all of them which is the mentioned SQNR.

We use an image of periodic spatial patterns with multiple orientations as benchmark, (Figure 2.8a.), and we compute the different local features (energy, orientation and phase). Each of the main stages requires a careful substages bit-width design but here we are only interested on highlighting the main designing decisions. Therefore, on this section we focus on the required bit-width of the following stages:

1. Convolution kernel weights.
2. Convolution results of the Gaussian derivatives base set.
3. Oriented quadrature filters.
4. Trigonometric functions (*sin*, *cos* and their powers or multiplications) that are used at the steering and interpolation stages.
5. Main non linear operations: division and *arctan*.
6. Estimated goal features.

We have chosen the energy as the confidence parameter for computing these features has been the energy. Pixels with lower energy values than a given threshold represent areas of low contrast, where filters outputs are prone to noise errors. We have used a low restricted value to reject from the bit-width analysis only very low confidence estimations.

Relative to the first point, convolution kernel weights, the methodology is the following. We keep all the computation using double precision floating point representation but the kernel values and compare the results with the complete software version. The results from the MCode *for DSP Analyzer* are shown on Figure 5.2.

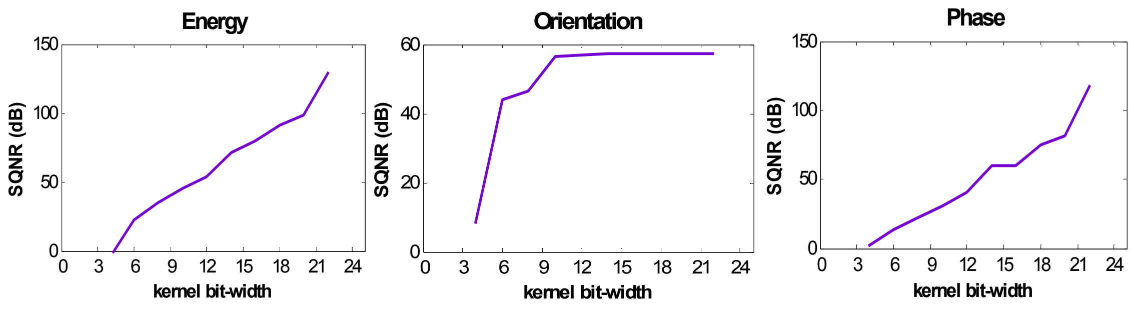


Figure 5.2. System SQNR vs. Gaussian derivatives kernels quantization. Energy and phase have approximately linear behaviour. For orientation, bit-widths larger than 11 bits do not produce any significant accuracy improvement. This is due to the fact that, as commented on Section 5.1.1, on the hardware simulator we use energy instead of magnitude as weight for equation (2.28.b) that makes not possible to achieve higher SQNR than 57.4 dB. Nevertheless, there is not significant error degradation and the quality of the system is very high already for a bit-width of 11.

Our first conclusion is that, values higher than 11 bits for the kernels do not improve the accuracy on orientation but do for energy and phase features. The next stage is the analysis of the bit-width used to store the output produced after convolution with these quantized kernels. These results are illustrated in Figure 5.3. Different kernels bit-widths are represented with different curves for each image feature. On this experiment we have considered (without loss of generality) 8 bits for the integer part and a variable number of bits for the fractional part of the convolution outputs (represented in the x axis). Different plots represent different bit-widths used for the kernel weights mask. This is a general method because a proper scaling was used.

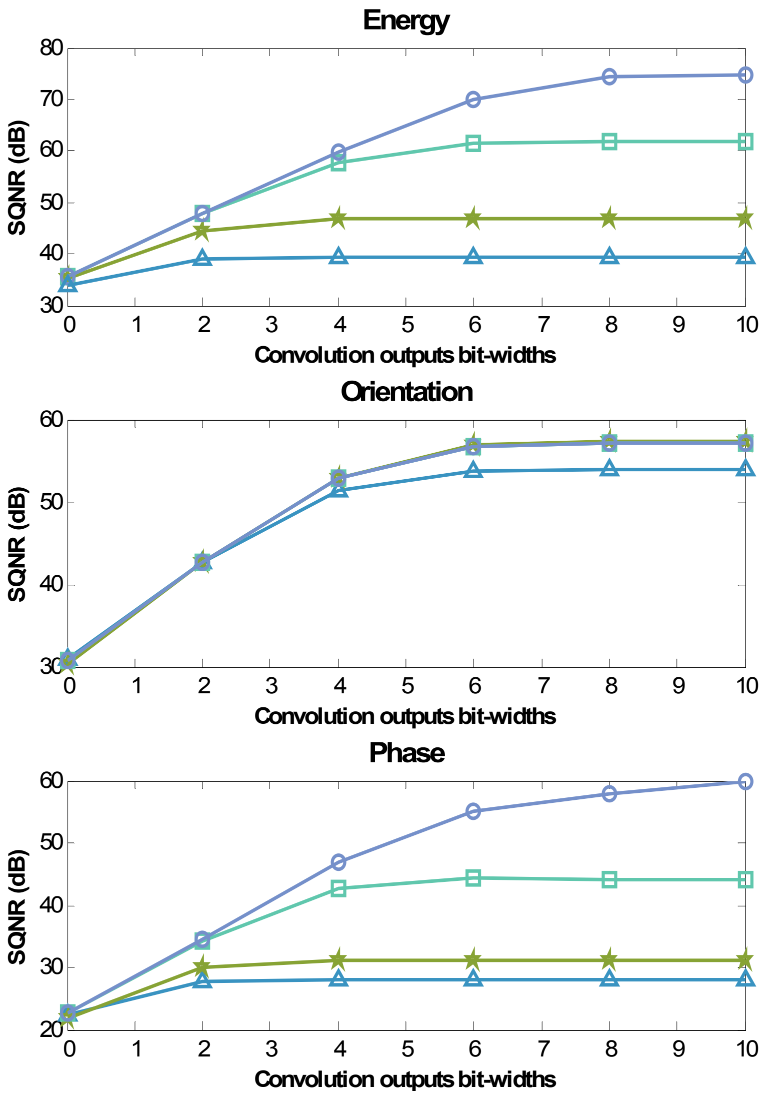


Figure 5.3. SQNR (db) evolution for different bit-widths configuration for kernel and convolution outputs quantization. The y-axe represents the SQNR values and x-axe the convolution outputs fractional part bit-widths. The integer part is fixed to eight bits. Four different curves are presented at each plot. They represent different kernels bit-width configurations. Triangles represent 9, stars 11, squares 13 and circles 15 bits to store the convolution outputs.

The main conclusion is that, in order to take full advantage of larger bit-widths of the convolution kernel, a larger bit-width should be used for the convolutions output.

For example, focusing on the Phase curve, for a kernel bit-width of 9 bits the optimal choice for the fractional part is 2 bits, 4 for a 11 bits kernel, 6 for 13 bits kernel, etc. Similar results are for the Energy but not for the orientation. As commented before, there are not significant differences for the kernels curves with more than 11 bits-widths and convolution outputs of 6 fractional bits.

Table 5.1. SQNR at different stages for two well defined configuration examples. Example A uses 11 bits for the kernel quantization and 9 bits for the convolution stage (only one fractional bit). Example B uses 13 bits for the kernels and 11 for the convolution outputs. Each stage (0 to 3) represents the SQNR of the Energy, Orientation and Phase after consecutive bit-widths quantization. At stage 0, only the kernels and convolution bits have been quantized. Stage 1 represents the SQNR after the previous quantization plus the quantization of the oriented quadrature filters with the same number of bits than the convolution outputs. Stage 2 shows the results with the addition of the next quantization stage, the trigonometric and non linear functions (*arctan* and division). Trigonometric functions require only 9 bits. *Arctan* function uses (2*bit-width-convolution outputs+2) bits to avoid degrading the system. Finally, stage 3 shows the SQNR of the whole system quantized using fixed-point data representation. Orientation and phase are angles and therefore their dynamic range is well defined (-π to π). We have used 9 bits to represent their values. For the energy, his dynamic range depend on the convolution outputs bit-widths. We have used (2*bit-width-convolution outputs) bits to achieve satisfactory results.

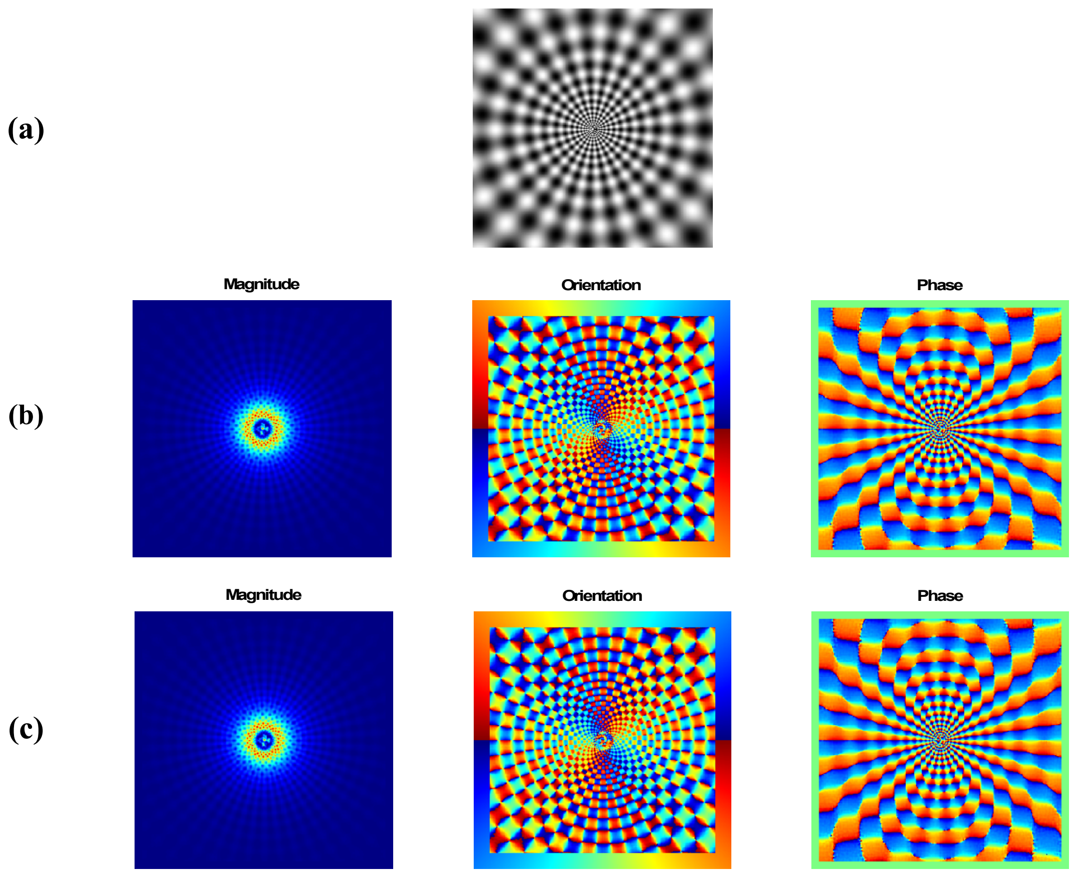
Stage	**SQNR (dB)**	**Energy**	**Orientation**	**Phase**
A .0	*Kernel and convolution outputs quantization*	40.739	36.323	27.036
A. 1	*Orientation quadrature filters quantization*	39.853	35.404	26.061
A. 2	*Trigonometric and non linear functions quantization.*	39.849	35.403	26.061
A. 3	*Image features results quantization.*	39.842	35.027	26.008
B. 0	*Kernel and convolution outputs quantization*	53.234	48.238	39.194
B. 1	*Orientation quadrature filters quantization*	48.560	46.923	40.512
B. 2	*Trigonometric and non linear functions quantization.*	48.557	46.922	40.512
B. 3	*Image features results quantization.*	48.550	46.272	40.163

At this point we need to define some accuracy goals. On Table 5.1 we see that at this quantization stage the SQNR of our estimation for two well balanced low area cases. We consider acceptable SQNR values larger than 30 dB.

Already in Fig. 5.2 was shown that the phase estimation is the most demanding (in terms of bit-width) to achieve high SQNR. For instance with 11 bits we obtain approximately SQNR=40 dB.

The results presented in Table 5.1 show that the most sensible stages are the kernel and convolution outputs quantization. Trigonometric, non linear function as well as the image features themselves can be quantized with a relatively small number of bits without extra accuracy degradation as can be deduced from the SQNR results. From this table we also conclude that fixed point arithmetic can be properly used for the system design on embedded systems due to the large value of the SQNR. This is especially true for the example B with quite high values that validate the bit-width configuration and data representation. Nevertheless, the final decision about system design should use the information about the hardware resources consumption. This is addressed in Section 5.3.

We have also included a qualitative evaluation of the previous precision examples. The software, quantized version A and quantized version B are shown on Figure 5.4 for a synthetic image and also for a real image (Figure 5.5).



(d)

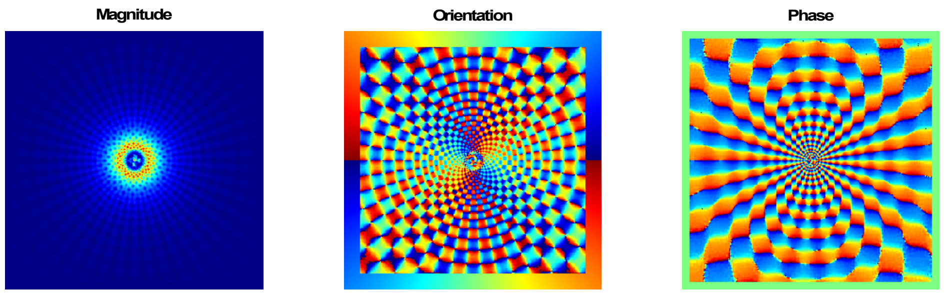


Figure 5.4. Images features computation with several data representations and bit-widths. Results obtained without any confidence threshold. (a) Synthetic image with multiple orientation and spatial scales used for testing. (b) Image primitives computed using software with double precision floating point representation. (c) Image primitives computed using fixed point data representation with bit-widths choices at example B of Table 5.1. (d) Image primitives computed using fixed point data representation with bit-widths choices of configuration A in Table 5.1. Note that the differences between floating point and fixed point representation, although numerically significant are not visible on the qualitative evaluation. Only some small differences can be found for the figures of row (d). This validates the previous choices as good alternatives for hardware implementation.

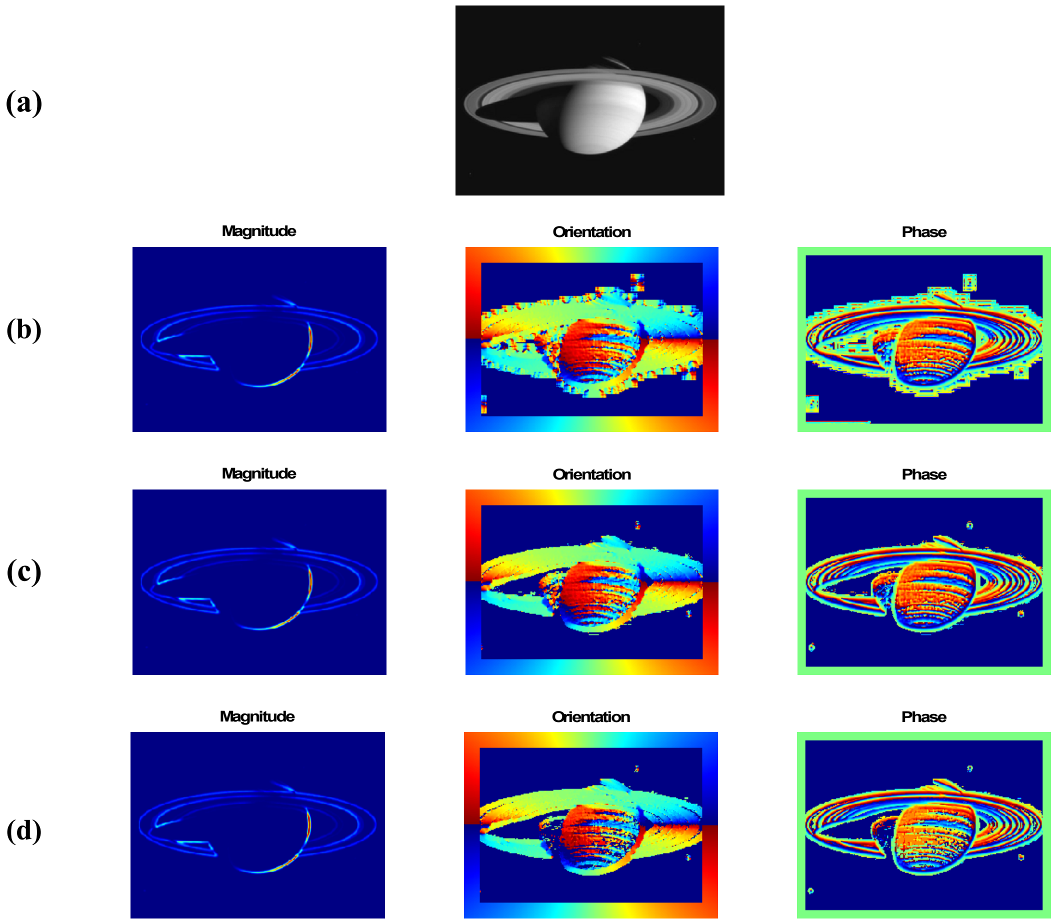


Figure 5.5. Images features computation with several data representations and bit-widths. (a) Real image of Jupiter captured in the Cassini-Huygens space mission. (b) Image primitives computed using software with double precision floating point representation. (c) Image primitives computed using fixed point data representation with bit-widths choices of configuration B in Table 5.1. (d) Image primitives computed using fixed point data representation with bit-widths choices of configuration A in Table 5.1. This time we use an energy threshold that rejects pixels with energy bellow the maximum energy • 10^{-5}. This allows seeing small differences between floating point and fixed point approaches. The threshold rejects more pixels for fixed point that for floating point. This can easily be justified because of the energy quantization effect. Despite that, the results are quite similar and the differences only reject unreliable data estimations. Although not all of them are rejected because we consider a very low restrictive threshold.

5.3 Hardware architecture based on a fine grain pipeline

According to discussion of Section 1.7.2, we have designed a very fine grain pipeline architecture whose parallelism grows across the stages to keep our throughput goal of one pixel output for clock cycle. The 3 main stages described in Section 5.1 are the coarse pipeline stages of our design. Each of them is finely pipelined to achieve our throughput goal.

The main circuit stages are described on Figure 5.6. Note that the coarse parallelism level of the circuit is very high (for example state S_1 requires 16 parallel paths). The stage S_0 basically is composed of 7 separable convolvers. The separability property of the convolution operation allows computing first the convolution by rows and after that by columns. This configuration requires only 8 double port memories shared by all the convolvers. They are arranged storing one image row at each memory to allow parallel data access at each column. This stage S_0 is finely pipelined in 24 microstages as indicated in brackets in the upper part of Fig. 5.6.

Stage S_1 is used to compute the oriented quadrature filters. This is achieved by multiplying each output of stage S_0 by some interpolation weights as shown on equation (5.1). These coefficients are stored using distributed registers for parallel access. In a sequential approach they could be stored using embedded memory but the required data throughput makes necessary full parallel access to all of them making unviable this option.

Stage S_2 compute the image features from this set of even and odd complex filters. Figure 5.6 shows the three different computation datapaths but, due to their complexity, we are going to explain them with more details.

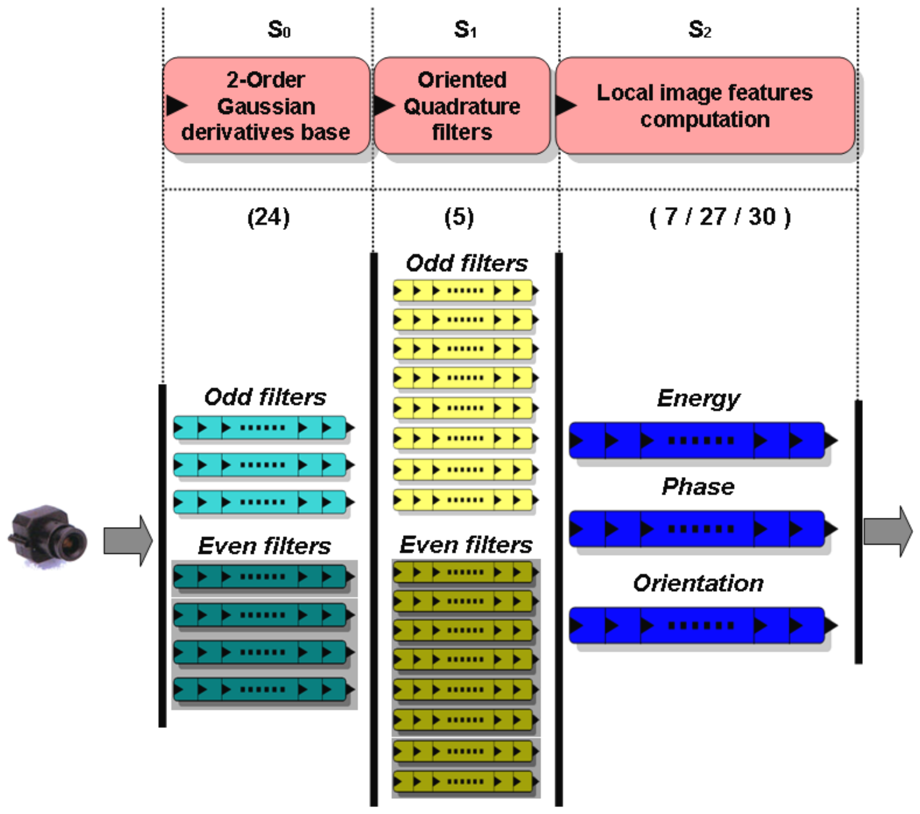


Figure 5.6. Image features processing core. Coarse pipeline stages are represented at the top and superpipelined scalar units at the bottom. The number of parallel datapaths increase based on the algorithm structure. The whole system has more than 59 pipelined stages (without counting other interfacing hardware controllers such as memory or video input/output interfaces). This allows computing the three image features at one estimation per clock cycle. The number of substages for each coarse-pipeline stage is indicated in brackets in the upper part of the figure.

Note that the last stage S_2 has different latencies (7, 27 and 30) for each feature (Energy, phase and orientation). Nevertheless, since we need all of them at the same time we use delay buffers to synchronize the outputs.

The three main datapaths are illustrated in Figures 5.7, 5.8 and 5.9. Note that the parallelism level grows to maintain the circuit's throughput. Orientation and Phase require the *arctan* function. We have tested a customized implementation based on LUTs and the CORDIC core taken from the Coregenerator tool of Xilinx [XIL06b]. Our results are summarized in Table 5.2. Although LUT approach requires less resources, it have lower precision (we use a 8 Kwords LUT, using the symmetries of the arctan function, this circuit has an equivalent data input precision of 15 bits). LUT table extension requires increasing the memory decoding logic which actually affects the speed of the whole circuit (only 42 MHz). Although this logic can be also pipelined to increase the clock rate, the logic required make the approach based on the CORDIC core more suitable for our implementation and therefore this is the option that has been chosen for our architecture.

Table 5.2. *Arctan* function implementation approaches. Cordic core uses data input with 21 bits. The LUT uses 8 Kwords to sample the *arctan* fuction. Decision logic allows to extend the range from the sampled interval [0, π/2[to the whole circumference.

Method	**Slices**	**EMBs**	**Multipliers**	**f_{clk_max} (MHz)**
CORDIC core	1,020	0	0	118
LUT-Arctan	841	5	0	42

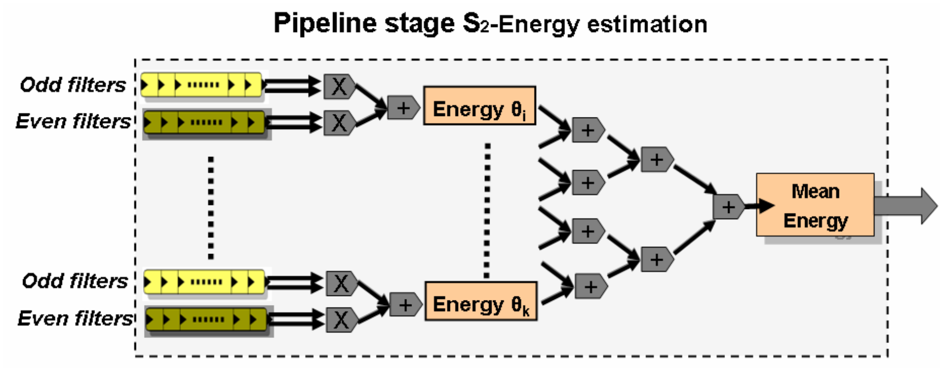


Figure 5.7. Image features core, stage S_2, Energy computation. This stage corresponds to the implementation of equation (2.28a). The mean energy is computed using a binary adder's pipelined tree. Normalization is computed by shifting operations. All these operations are computed using 7 fine pipeline stages but a memory buffer is connected to its output for synchronization with other image features.

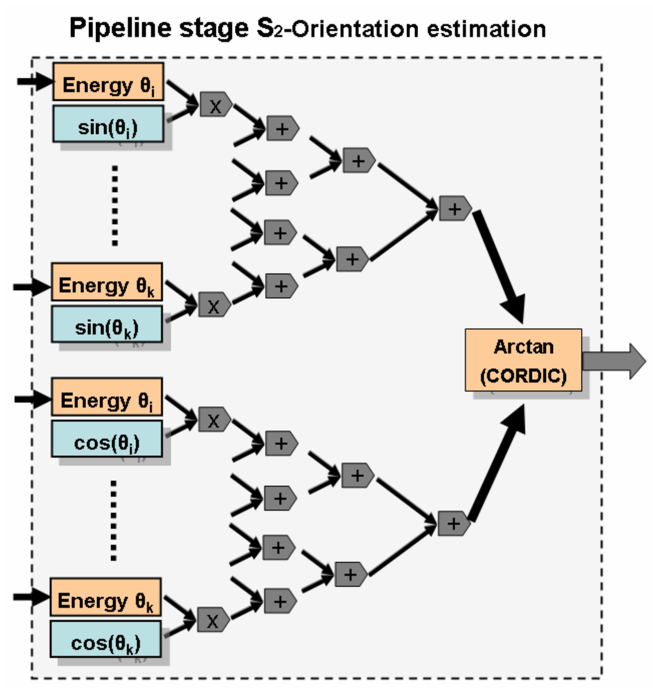


Figure 5.8. Image features core, stage S_2, Orientation computation. This stage corresponds to the implementation of equation (2.28b). The local energy computed at each orientation on the energy path is the input of this stage. Each oriented energy is multiplied by their coefficient and then is added using a binary adder's pipelined tree. Orientation angle is computed using a CORDIC core customized for our application. All these operations are computed using 30 fine grain pipeline stages and this is the largest path of stage S_2.

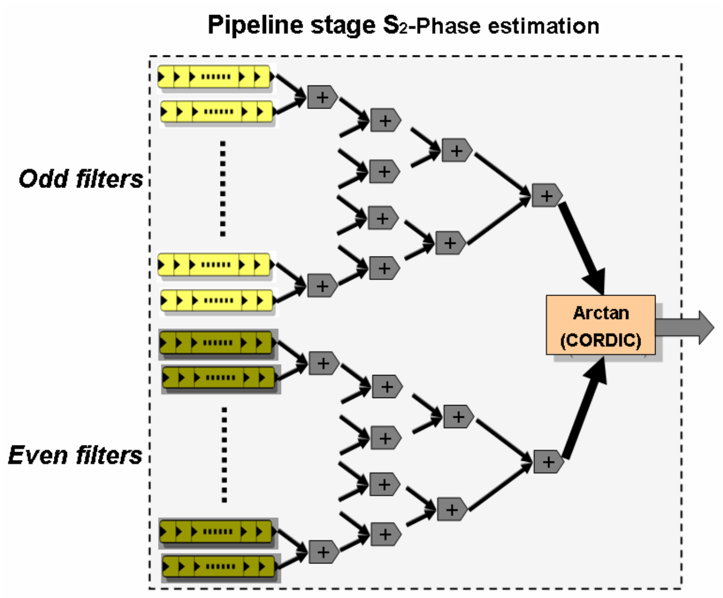


Figure 5.9. Image features core, stage S_2, Phase computation. This stage corresponds to the implementation of equation (5.3). The sum of even and odd filters is computed using a binary adder's pipelined tree. Phase angle is computed using a CORDIC core customized for our application. All these operations are computed using 27 fine grain pipelines stages, and thus we need to use a memory buffer connected to its output for synchronization with other image features.

5.3.1 Hardware resources consumption analysis

This architecture has been designed allowing easy parameter specifications. This allows exploring the resources-consumption vs. accuracy trade-off. Based on the analysis of Section 5.2, we focus on the analysis of the convolution output bit-widths. The different pipeline stages of our design scaling using the conclusions extracted from this study. Kernels coefficients bit-width have been selected from the values of Figure 5.3 that optimize the system (bit-widths values that produce flat horizontal curves). This means in ranges from 9 to 21 bits for the kernels and from 9 to 22 bits (8 of them taken for the integer part and the other for the fractional part) for the convolution outputs.

The hardware resources vs. bit-widths evolution is represented in Figure 5.10. Note that the trade-off depends on the proposed architecture and parameter relation and due to that, conclusions can only be extended to similar implementations. The Number of slices grows approximately exponentially. This is mainly caused by the non linear stages whose expansion does not follow a quadratic tendency. The clock frequency of the system decreases down to a minimum value of (approximately) 50 MHz. This stable minimum can be explained based on the synthesis tool properties. Our design uses the *retiming* capabilities of the DK synthesizer. This allows redistributing combinational logic across a registers path to increase the circuit speed. When the bit-width grows, we increase the number of pipeline stages to compensate this effect. For small bit-widths, *retiming* does no produce any effects but, for larger values, *retiming* is able to reduce the combinational paths, increasing the maximum system clock frequency up to this value. Nevertheless, the synthesis process involves two different tools which work at different abstraction levels, the ISE Foundation [XIL06d] and the DK Design Suite [CEL06b]. Because they are proprietary tools, the information available about this process is very limited and the analysis could be biased by unknown tasks. Furthermore, the frequency change is less than the 14% of the peak value. This percentage is not of significant relevance on the synthesis process and therefore it could be biased by the optimization tools parameters.

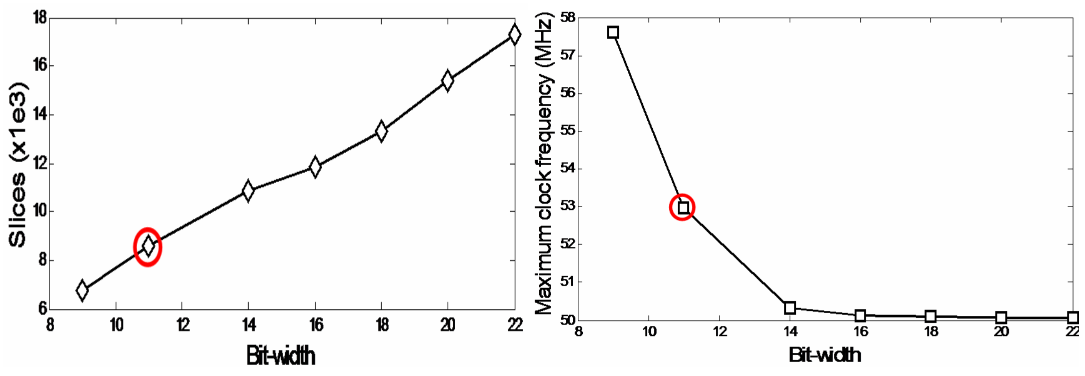


Figure 5. 10. Hardware resources consumption and system clock frequency vs. convolution output bit-width. We can address the different configuration studies because the design technique allows automatically scanning of data bit-width and scaling changes (just adjusting some predefined parameters). We have marked with a circle our design choice which corresponds to the configuration B analyzed in Table 5.1.

We have chosen the bit-widths described in Table 5.1, configuration B. Figure 5.3 shows that the hardware resources for this approach represent a good trade-off between accuracy (SQNR of all features larger than 40 dB) and resources consumption. The whole core has been implemented on the RC300 board of Celoxica [CEL06d]. This prototyping board is provided with a Virtex II XC2V6000-4 Xilinx FPGA as processing element including also video input/output circuits and user interfaces/communication buses. The final required hardware consumption and performance measures of our implementation are shown in Table 5.3.

Table 5.3. Complete system resources required for the local image features computing circuit. The circuits have been implemented on the RC300 prototyping board [CEL06d]. The only computing element is the Xilinx FPGA Virtex II XC2V6000-4. The system includes the image features processing unit, memory management unit, camera Frame-grabber, VGA signal output generation and user configuration interface. (*Mpps*: *mega-pixels per second* at the maximum system processing clock frequency, *EMBS*: embedded memory blocks).

Slices / (%)	**EMBS / (%)**	**Embedded multipliers / (%)**	**Mpps**	**Image Resolution**	**Fps**
9135 (27%)	8 (5%)	65 (45%)	56.5	1000x1000	56.5

Detailed information about the image features core and the coarse pipeline stages are shown in Table 5.4. Note that the sum of the partial stages is not the total number of resources. This can be easily explained based on the synthesis tools. For the whole system is more likely that the synthesizer engine can share some resources and reduce hardware consumption. This also explains why the whole system has lower clock frequency than the slowest subsystem. This is a trade-off that should be taken into account into the design process.

The previous argument does not explain why the number of multipliers grows on the complete system with respect to the sum of the multipliers used at each substage. We have used automatic inference of multipliers on the system. Because the number of multiplications by constants is quite high, it is difficult to manually determine when is better to use an embedded multiplier or compute it using FPGA logic. Synthesis tools define cost functions that are able to evaluate based on technological parameters for each multiplication which is the best option. We have compared the manual and the automatic inference of multipliers and, though differences are not large, automatic inference usually achieves higher clocks rates than manual generation with slightly less logic. This automatic inference changes depending on the circuit that is being synthesized and produces the differences presented in Table 5.4.

Table 5. 4. Partial system resources required on a Virtex II XC2V6000-4 for the coarse pipeline stages described for this circuit. (*EMBS* stands for embedded memory blocks). The differences between the sum of partial subsystems and the whole core are explained in the text.

Circuit stage	**Slices / (%)**	**EMBS / (%)**	**Embedded multipliers / (%)**	**f_{clk} (MHz)**
S_0 *Gaussian base convolutions*	4,170	8	50	85
S_1 *Oriented quadrature filters*	1,057	0	0	69
S_2 *Features Energy, Phase and Orientation computation*	2,963	0	6	89
Whole processing core	7627	8	65	58.8

5.4 Conclusions: System results summary

The presented implementation combines a well justified bit-width analysis with a high performance computing architecture. The circuits SQNR is quite high (more than 40 dB) which provides a high accuracy computation, as can be seen from the results of Figures 5.11 and 5.12.

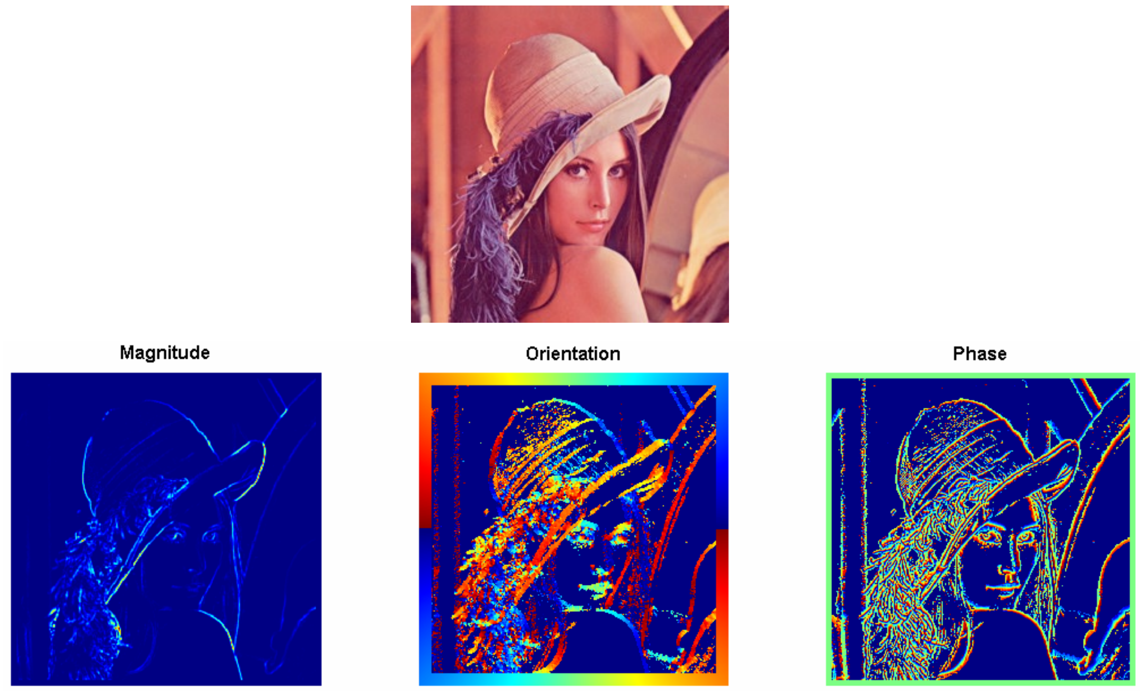


Figure 5. 11. System results for the known image of Lena (left image). The three right pictures present the computed features for this image. Note that although restricted fixed-point arithmetic is used, the quality of the features is quite high as can be seen looking on small details such as presented on hat's plumes.

Note that as illustrate Figure 5.12, the contrast independency of the phase information can be a quite interesting property for applications which requires high quality image edges detection [VAR04], [VAR05], [ROS06].

The computation performance allows processing more than 56 Mpixels per second, where the imagen resolution can be adapted to the target application. For instance, using VGA resolution of 640x480 pixels, the device can compute up to 182 fps. This is achieved thanks to the fine grain pipeline computing architecture and to the highly parallelism employed. This outstanding power performance is required for the target specifications addressed on the DRIVSCO project [DRI06], where stereo pair images of resolution 1024x1024 require real-time computation. The superscalar and superpipelined proposed architecture fulfils our requirements and illustrates that, even with a low clock rate, a well defined architecture can achieve an outstanding computing performance unviable with current standard processors running more than 60 times faster than FPGA circuits.

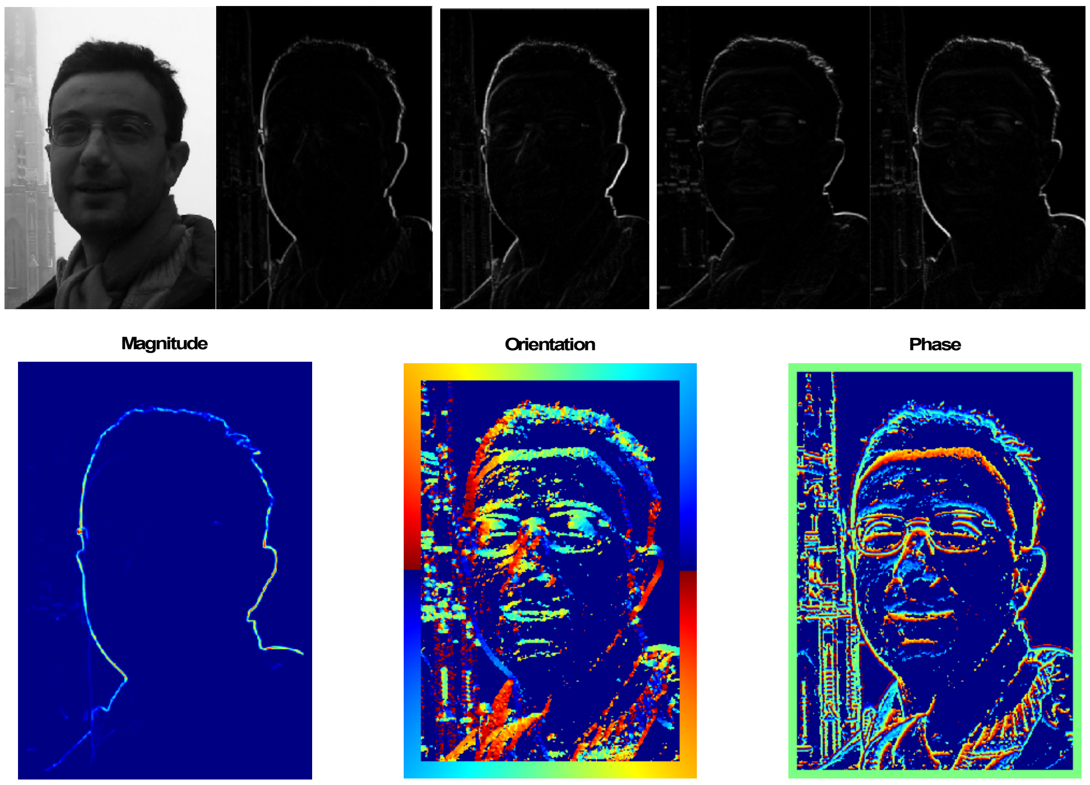


Figure 5. 12. System results for a real image. First row, left side, represent a photograph of a close face with a tower in the left side. The tower is blurred due to the fog and therefore has low contrast. Four images to the right represent the quadrature filters energy output at 4 defined orientations (0, π/4, π/2 and 3π/4). The second row represents the computed features for that image. Note that at the energy image, the tower is almost invisible but not for the orientation and phase images. This illustrates that the structure based features have not got contrast dependency, and therefore the tower appears clearly visible.

Chapter 6
Motion processing: Hardware implementation of a high performance computing architecture

This chapter proposes several specific purpose architectures to compute optic flow. The different systems described in this chapter represent solutions with different trade-offs of several characteristics: accuracy, hardware cost and data throughput. We have implemented different alternatives of the original Lucas and Kanade model. The performance evaluation section include comparisons with the state-of-the-art approaches; in fact the comparisons represent a considerable effort that has been worth to do since it helps to highlight the outstanding performance obtained by the presented systems.

6.1. Introduction

Although in Chapter 3 we concluded that we would focus on the Lucas and Kanade model, we described different alternatives (mainly FIR vs. IIR temporal filters) and modifications suggested by Brandt [BRA97] that significantly improve the model accuracy at a moderate cost in terms of computing load. Instead of focusing on a single implementation, since once defined the designing strategy the implementation of different alternatives is not very time consuming, we have designed and evaluated different versions of the L&K model.

We have used a high level Hardware Description Language (HDL) to define the circuits. More concretely we have used Handel-C [CEL06c]. We have also integrated specific modules (cores), such as a floating point processing unit when considered appropriate. This design strategy has facilitated the possibility of defining the system at high abstraction level. Being very easy to modify and create different versions of the same circuit with different hardware cost vs. computing power trade-offs.

We have focused on two alternative systems:

- L&K original model with temporal IIR filters. This system represents a solution that achieves frame-rate computation at VGA resolution. In order to achieve this at moderate hardware cost we adopt the IIR temporal filters which require only 3 images to be stored and we design a coarse grain pipeline datapath.
- L&K modified model according to the improvements suggested by Brandt [BRA97]. This alternative represents a high performance computing architecture able to process high frame-rates at high image resolution (see Section 6.3 for more details). For this purpose we have used FIR temporal filters, that although requiring higher computational (storage) resources, they represent a more stable solution when restricted fixed-point arithmetic is used. Furthermore, since in this case we address a high performance system we have designed a fine grain pipelined datapath of more than 70 stages, able to deliver one motion estimation per clock cycle. To the best of our knowledge, this system outperforms any existing optic flow system described in the literature by more than one order of magnitude.

A significant effort has been made in evaluating the hardware resources and the accuracy of the different approaches. This has revealed to be a very time consuming study but important to facilitate the comparison of the proposed processing architecture with other approaches described in the literature. This evaluation study has been critical to publish the approaches described in this chapter.

6.2. Coarse grain pipelined optic flow processing architecture based on IIR temporal filters for the original model of Lucas and Kanade

In this section we describe the design of a customized DSP circuit in a single chip of high computational power based on an intensive use of the inherent parallelism of FPGA devices.

For this approach we have chosen the original L&K model and adopted the IIR filters for the temporal filters as proposed in [FLE95] (see Chapter 3, Section 6.1 for more details).

Another slight modification makes possible to provide estimations when the aperture problem appears in the direction of the maximum gradient [DIA04a], [DIA04c]. We have added a small constant, α to the matrix diagonal as suggested in [SIM91], which allows estimating the normal velocity field in situations where 2-D velocity cannot be extracted due to the lack of contrast information.

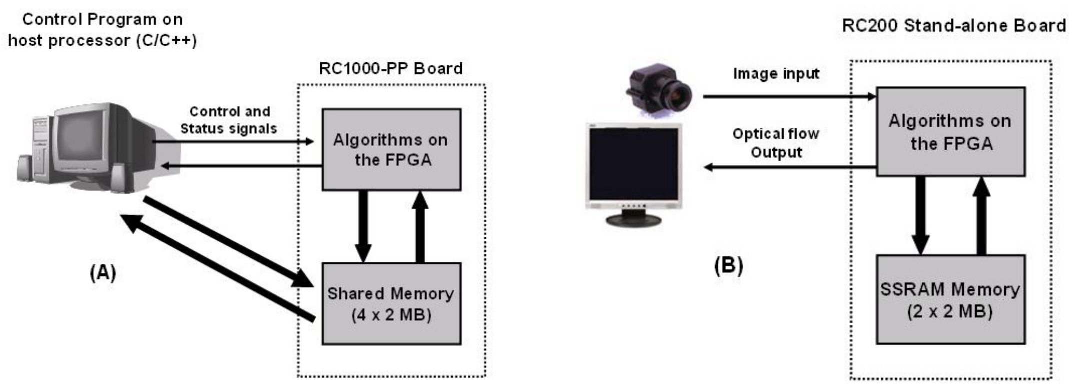
Other authors have recently described the hardware implementation of optical-flow algorithms [NII04], [COB98], [MAY03], [COR02] but most of them provide no results to evaluate the performance of the system, i.e. the accuracy and the computation speed. We describe a fully stand-alone working system at conventional camera frame rates of 30 Hz, with image sizes of 320 x 240 pixels.

6.2.1. Hardware Description

For our design we have used two platforms: the first one is the RC1000-PP board from Celoxica [CEL06e]. This is a PCI bus board connected to the PC and can be used as a hardware accelerator board or as a prototype board containing a Virtex 2000E-6 Xilinx FPGA The second platform is the stand-alone RC200 board from Celoxica [CEL06f]. This board includes camera input, video/VGA output, two 2 MB SSRAM memory banks and a XC2V1000-4 FPGA. It is a very suitable test system for embedded applications. The conection schemes are illustrated in Figure 6.1.

We have used Handel-C [CEL06c] as hardware specification language to generate the Edif input to the Xilinx ISE environment. This high-level hardware language allows us to describe RTL circuits in a very algorithmic-like way. This is relevant due to the algorithmic nature of the proposed method that makes an RTL approach more difficult to adopt. The drawback is the cost in terms of number of gates but the design time is reduced significantly [ORT06a], [ORT06b].

Four our discussion, we will focus on the PCI board implementation [DIA06F]. The version running on the stand-alone platform implements the same processing architecture (including new I/O controller modules). In Section 6.2.2 we will only outline the resource requirements and the improvements for the design based on the Virtex II FPGA family.



(a) (b)

Figure 6.1. Platforms for the optical flow computing system. (a) PCI-board scheme. (b) Stand-alone board scheme.

6.2.1.1 Hardware development

The efficient implementation of the algorithm with an FPGA device requires the intensive exploitation of the intrinsic processing parallelism of this kind of device.

We use a pipeline architecture, as shown in Figure 6.2, which basic computational stages can be summarized as follows:

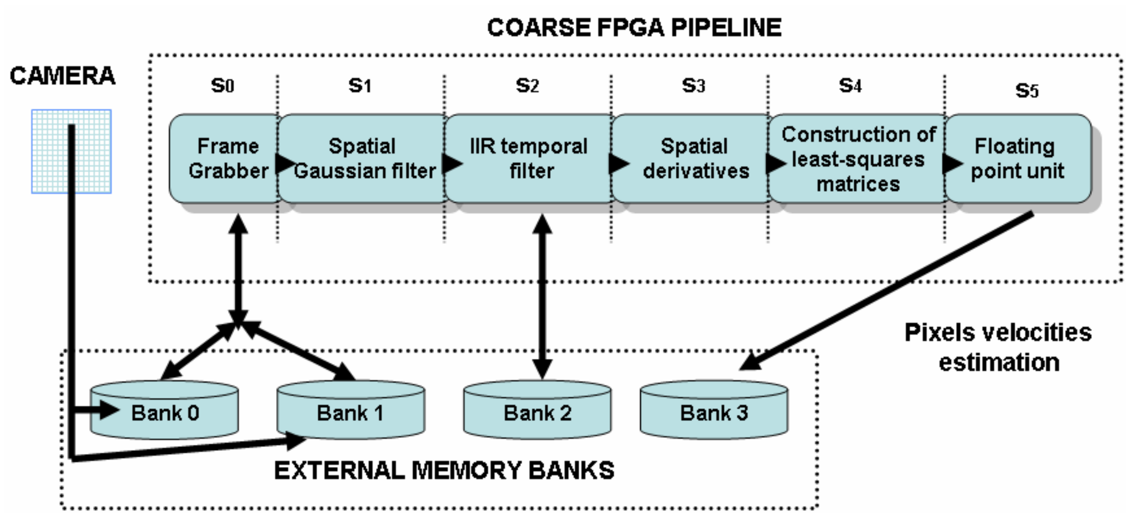
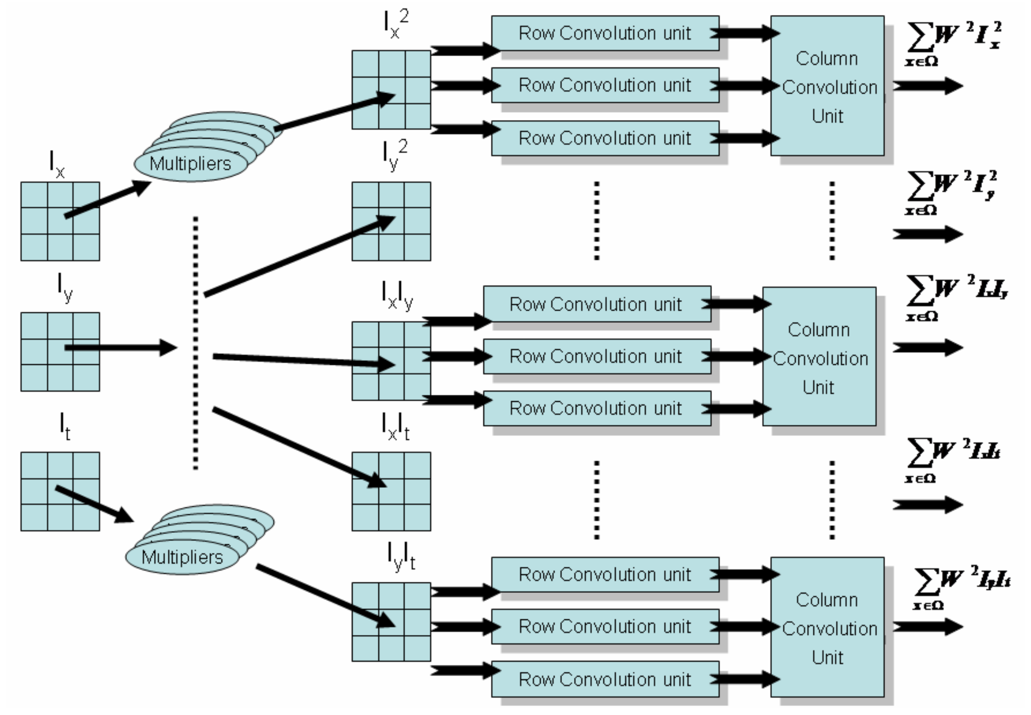


Figure 6.2. Coarse pipeline processing architecture

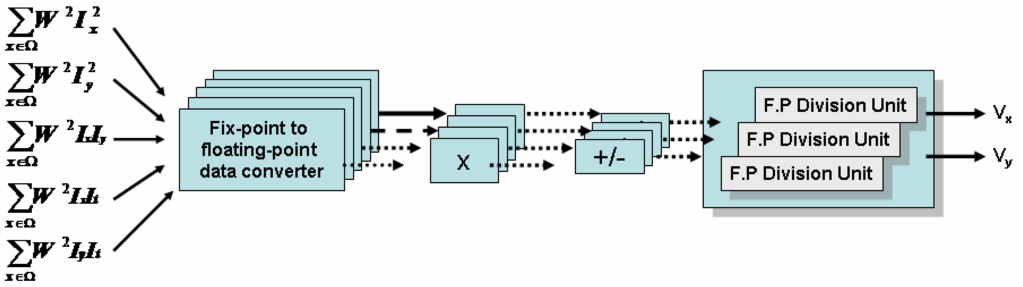
- S_0. The frame-grabber receives the pixels from the camera and stores them in one of the memory banks, using a double-buffer technique to avoid temporization problems.
- S_1. Spatial-Gaussian-filter smoothing stage.
- S_2. The IIR temporal filter computes temporal derivative and space-time smoothed images.
- S_3. Spatial derivatives stage.
- S_4. Construction of least-square matrices for integration of neighborhood velocities estimations [BAR94].
- S_5. Custom floating-point unit. Final velocity estimation requires the computation of a matrix inversion, which includes a division operation. At this stage the resolution of the incoming data bits is significant and expensive arithmetic operations are required. Thus fixed-point arithmetic becomes too expensive, prompting us to design a customized floating-point unit.

The computation bit-width increases throughout the pipeline structure. For example, for a high precision system with low accuracy degradation, we use 8 bits in the first two stages, 12 bits in the third and fourth stages, 24 in the construction of the least-square matrices and 25 for the floating-point unit. The computation of the least-square matrices (S_4) is the most expensive stage in terms of computational resources. Different parallelism strategies can be adopted at this point.

The basic parameters of the pipeline structure are *Latency* (L) and the *Maximum Number of Cycles* (MNC) required during the longest stage, which is the limiting factor of the computing speed. The pipeline circuit scheme provides a computing speed (data throughput) in pixels per second (pps) that depends on the MNC and the frequency clock (f_{clk}) according to the expression $pps=f_{clk}/MNC$.


I_x
I_y
I_t
Multipliers
Multipliers
I_x^2
I_y^2
I_xI_y
I_xI_t
I_yI_t
Row Convolution unit
Column Convolution Unit
$\sum_{x\in\Omega} W^2 I_x^2$
$\sum_{x\in\Omega} W^2 I_y^2$
$\sum_{x\in\Omega} W^2 I_x I_y$
$\sum_{x\in\Omega} W^2 I_x I_t$
$\sum_{x\in\Omega} W^2 I_y I_t$

(a)


$\sum_{x\in\Omega} W^2 I_x^2$
$\sum_{x\in\Omega} W^2 I_y^2$
$\sum_{x\in\Omega} W^2 I_x I_y$
$\sum_{x\in\Omega} W^2 I_x I_t$
$\sum_{x\in\Omega} W^2 I_y I_t$
Fix-point to floating-point data converter
X
+/-
F.P Division Unit
F.P Division Unit
F.P Division Unit
V_x
V_y

(b)

Figure 6. 3. Architecture details for some specific pipeline stages. (a) Pipeline stage S_4, Least squares matrices circuit builder for a 3x3 neighborhood. (b) Pipeline stage S_5, Floating-point unit scheme.

There are two main critical stages: S_4 and S_5 whose architecture are shown in Figure 6.3. The construction of least-square matrices is done in S_4 where the trade-off between efficiency and cost can vary widely. Equation (3.10) requires the previous computation of five products: I_x^2, I_y^2, I_xI_y, I_xI_t, I_yI_t. Thus we make a weighted sum in a window (Ω) over a neighborhood of size $w_x \bullet w_y$. Due to memory limitations we save the I_x, I_y, and I_t values instead of the five crossed products. Therefore, the operations required are:

- Computation of the products for all the elements within a neighborhood. We need to calculate five $w_x \bullet w_y$ multiplications.
- Row-convolution operation. We compute five multiplications by w_y convolutions.
- Column-convolution operation, requiring the computation of five more convolutions.

This is an important stage where we can bias the trade-off between efficiency and hardware cost. For example, if we use a 3x3 neighborhood, we need between 1 to 45 multipliers, 1 to 15 row-convolution units and 1 to 5 column-convolution units. This choice allows us to compute the weighted sum values in one clock cycle with a highly parallel hardware unit or to compute it sequentially.

The second critical stage is the computation of final velocities using a custom floating point unit which architecture is shown in Figure 6.3.b. At this stage equation (3.10) is computed. Until now the arithmetic operations have been done using integer or fixed-point arithmetic with truncation operations. Convolution operations work well with this representation but when bit-width is too large, a floating-point representation of the data is better suited for hardware implementation. This is done with a customized superscalar floating-point unit. Since during the previous stage (S_4), a high bit-width (24 bits) is used to preserve computational accuracy, the current stage (S_5) becomes very expensive in terms of hardware resources. Therefore the design of S_5 is critical as it exerts an important influence on the accuracy *vs* processing speed trade-off.

The calculations in this stage involve the following basic arithmetical operations: subtraction, multiplication and division. When arithmetical operations are made with high bit-width, the signal delays associated with carry lines degrade overall performance, decreasing the maximum system frequency. To avoid this, pipeline arithmetic operators or sequential iterative operators can be used. The first approach allows us to make the computation in 1 or 2 clock cycles, after a given latency at a high cost in terms of hardware resources. The second option takes several clock cycles, therefore degrading the MNC of the system, but allows us to use the same hardware for each iteration. We define a system which uses one-cycle floating-point hardware circuits because this works at the desired maximum clock frequency (without becoming the limiting stage) for all the operations except the division. We have used a hardware sequential divisor instead of a pipelined divisor that needs 21 cycles to compute the division of 25 bits of floating numbers. But in this case the MNC is too high and imposes a considerable limit on pipeline performance. To counter this we use up to 3–way division units and, depending on the performance required, we can synthesize more or less ways. Each floating-point unit needs:

1. One to five fixed-point to floating-point converter units.
2. One to six 25-bit floating point multipliers.
3. One to three subtractors.
4. One or two divisor units. If an n-ways divisor scheme is chosen, we use n to 2n divisor units.

6.2.2. Hardware resources and processing performance

This section presents the main results of the system. First, we analyze the resources consumption of the different stages as well as different implementation possibilities with different levels of parallelism. After that we measure the performance and finally, we estimate the accuracy of the system using synthetic and real sequences for evaluation.

6.2.2.1. Hardware resources consumption and flow estimations accuracy

The system is designed in modules so that parallelism and bit accuracy at the different stages can be easily modified. Due to the high level of abstraction that Handel-C provides [CEL06c] it is easy to manage the parallelism of the computing circuits and the bit-width at the different stages. The hardware resources of the different stages using a XCV2000E Virtex FPGA for a specific implementation (called HSHQ in the Table 6.2) are summarized in Table 6.1.

Table 6. 1. Detailed sub-circuit hardware requirements on a Virtex XCV2000E. Note that the sum (% of the device in the first column) is larger than 100%, this can be explained because these data have been obtained by partial compilations and the synthesis tool makes a wide use of the available resources. When the whole design is compiled it consumes 99% of the device.

		Number of slices / (% of the device) / equivalent gates	Computing cycles	ISE maximum Clock frequency (MHz)	Memory requirements / (% of the device)
S_1	Spatial Gaussian (17 taps)	220 / (1%) / 270,175	8	29.2	16 / (10%)
S_2	IIR filter	134 / (1%) / 51,971	7	38.5	3 / (1%)
S_3	Spatial derivative convolution	287 / (1%) / 121,296	7	28.0	7 / (4%)
S_3	Least square matrices construction	15,288 / (79%) / 642,705	10	20.3	24 / (15%)
S_5	Superscalar floating point unit	5,720 / (29%) / 90,993	10	17.4	0

The last two stages have the larger MNC values. Note that a lower MNC is possible for other stages but there is no reason to improve them due to the other existing limiting stages. The maximum clock frequency is taken from Xilinx timing analyzer thought they are not always accurate. In fact it usually underestimates the speed at which a circuit can run: the maximum frequency allowed by the system has been experimentally measured and it is 10-20 MHz higher than the very conservative results given by ISE. This arises because the analyser looks at the static logic path rather than the dynamic one (cf. [CEL06g]) and because of that we measure experimentally the maximum working frequency. As can be seen in Table 6.1, we have designed a system with a pipeline structure which has a global latency of 42 cycles and a maximum working frequency of 17.4 MHz evaluated by ISE (35 MHz measured experimentally).

One important aspect is that of the various possibilities for configuring the system. We have evaluated several configurations to explore different trade-offs between accuracy, hardware cost and computing speed. In all these configurations we have used the same basic architecture but with different levels of parallelism, mainly customizing stages S_4 and S_5.

Table 6. 2. Performance and hardware cost of different configurations on a Virtex 2000-E FPGA (2 million gates and 640 Kbits of embedded memory). (Kpps → kilopixels per second, Fps → frames per second). All the performance values were measured using a clock frequency of fclk=27MHz. These measurements (Kpps and Fps) are underestimations because the computing time measured also included data transmission to the prototyping board.

Version	% device occupation	% on-chip memory	Kpps	Image resolution	Fps (f_{clk}=27MHz)	Max. f_{clk} (MHz)
HSHQ	99	17 31	1776	160x120 320x240	95 24	35
HSMQ	65	16 31	1776	160x120 320x240	97 24	35
MSMQ	43	16	625	160x120	33	35
LSLQ	36	8	400	120x90	38	35

Table 6.2 summarizes the main properties of the different configurations. The ones using a 5x5 average window for the least-square-matrix neighborhood are called high quality (HQ) approaches, and the ones using a 3x3 window, medium quality (MQ). Other modifiable parameters are the smoothing and spatial derivative filter sizes. HQ and MQ approaches include 5-pixel derivative filters and 9-pixel Gaussians. A low cost (LQ) version uses 3-pixel derivatives and a Gaussian filter of the same size. If we fix the optical-flow quality of the system, another factor to take into account is the performance vs. hardware cost trade-off. If the system works with maximum parallelism the MNC is 10. Lower cost approaches are possible if we reduce the parallelism level,

thus increasing MNC. For example, we implemented a high-speed (HS) version with MNC=10 cycles using a three-way division unit and maximum parallelism. A slower version was implemented reducing the parallelism and thus resulting in a medium speed (MS) version. Finally, we implemented a low-speed (LS) version. Table 6.2 summarises the performance of the systems and hardware costs.

It is important to note that in our experiments data transmission of the images to the prototyping board through the 33 MHz PCI bus takes about 30-40% of the total processing time and therefore higher frame rates might be expected using a direct connection between the camera and the FPGA. Furthermore, the theoretical data-throughput of the HSHQ is 2700Kpps at this clock frequency. This topic is amply discussed in [BEN03].

We also have tested the design using the stand-alone prototyping platforms RC200 and RC203 [CEL06f] to avoid the PCI bus bottleneck (see Table 6.3). The first platform includes an XC2V1000-4 FPGA, the second includes a XC2V3000-4 FPGA, both with embedded multipliers. In these approaches we have implemented the whole optical flow system plus Video input, VGA and memory arbitration controller. A LUT for visual color representation of the velocities vector for the VGA output has also been included in the FPGAs. The optical flow system implemented in the RC203 is the HSHQ. In the RC200 is a customization for this specific platform. It shares the main properties of HSHQ PCI board version but uses the embedded multipliers and several clock domains. This version also has a limited level of parallelism getting a MNC value of 14 cycles and can be considered as *Medium Speed, High Quality* (MSHQ) version.

Table 6.3. Performance and hardware costs of stand-alone systems for optical flow computation with camera input and VGA output. First row contains values correspond to the HSHQ version using RC203 board provided with a Virtex II XC2V3000-4 FPGA (3 million gates and 1728 Kbits of embedded memory). Second row implement the MSHQ version using the Virtex II XC2V1000-4 FPGA (1 million gates and 720 Kbits of embedded memory) which RC200 board is provided.

Version	% device occupation	% on-chip memory	% Embedded multipliers	Kpps	Image resolution	Fps	Max. f_{clk} (MHz)
Stand-alone RC203 board	99	29	41	4100	340x280	53	41
Stand-alone RC200 board	99	70	40	2857	340x280	37	40

The computing speed measured at the maximum clock frequency (40 MHz) for the system running in the RC200 platform was 2,857 kpps (30 fps of 340x280 images). Now the system is faster due to the improved technology of the Virtex II and the elimination of the PCI bus. The use of a customizable approach with a high level description language facilitates the implementation of this system on an FPGA of only 1 million gates (99% of resources was used). In fact, the optical flow processing algorithm only consumes 80% of the number of slices whilst the rest is occupied by the I/O controllers. The use of the embedded multipliers saves 662 slices (13% of the system resources). In the RC203 platform, higher level of parallelism is achievable and due to that, the system is able to process up to 4100 Kpps with the HSHQ system version.

6.2.2.2. Performance evaluation

As commented in Chapter 3, the accuracy of the computation of the optical flow in real-life sequences is difficult to assess because the real flow of these sequences is unknown. Therefore to evaluate the accuracy of our design, which depends on the bit-width of the different stages, we have adopted the test scheme and synthetic sequence from the comparative study made by Barron *et al.* [BAR94], with the error measurement proposed in [FLE90]. This error measurement has been widely used in the literature and therefore it is appropriate to compare our results with previous works. This measurement can be used with high- and low-velocity modules with the same estimators but with some bias. A more detailed explanation of this can be found in [BAR94].

In the hardware implementation some simplifications are made to the original model. Table 6.4 shows the accuracy of the model after the modification of several parameters. Unthresholded results (100% density) are included to enable an easy comparison between the hardware and software versions. The second row in Table 6.4 includes the evaluation results with reliable estimations as indicated in [BAR94].

The fourth and fifth rows include results of the implementation of the algorithm with IIR filters computed with fixed point arithmetics using 12 bit-width. In the sixth row of Table 6.4 the accuracy of the L&K algorithm (with hardware-system parameters) is computed by a standard PC using double precision variables and unthresholded results. Finally, the seventh row includes the performance achieved with our hardware implementation. It can be seen that accuracy is reasonably high, bearing in mind that fixed-point variables and restricted bit-widths are used in this approach. It can be seen that the performance of the hardware is only slightly worse (2.48º increase in error) than the software version with a data precision of 64 bits. Furthermore, the results of the hardware implementation described here are comparable with other software approaches evaluated by Barron *et al.* [BAR94].

Table 6.4. Yosemite sequence results using the angle error measurement of Fleet *et al.* [FLE90]. Comparison between software models (including FIR and IIR filters, and computed with double precision variables) with different parameters. Final row also includes the hardware system accuracy. The fourth, fifth and sixth rows use the simplifications adopted in the hardware implementation. For a description of the parameters significance see [BAR94][FLE95].

Model	Average Error º	Standard deviation º	Density %	Parameters
LK FIR	11.29	17.72	100	$\lambda_{min}=0$, $\alpha=0$, $\sigma_{xyt}=1.5$
LK FIR	4.54	11.31	33.3	$\lambda_{min}=0.75$, $\alpha=0$, $\sigma_{xyt}=1.5$
LK IIR	11.97	16.027	100	$\lambda_{min}=0$, $\sigma_{xy}=1.5$, $\tau=2$, $\alpha=0$.
LK IIR ***(with hardwarized filters)***	11.47	15.34	100	$\lambda_{min}=0$, $\sigma_{xy}=1.5$, $\tau=2$, $\alpha=0$
LK IIR ***(with hardwarized filters)***	13.71	15.99	100	$\lambda_{min}=0$, $\sigma_{xy}=1.5$, $\tau=2$, $\alpha=1/16$;
LK IIR ***(version implemented in hardware)***	15.91 º	11.5 º	100	$\lambda_{min}=0$, $\sigma_{xy}=0.8$, $\tau=2$, $\alpha=1$
Hardware system	18.30 º	15.8 º	100	$\lambda_{min}=0$, $\sigma_{xy}=0.8$, $\tau=2$, $\alpha=1$

We have also compared the performance of the software and the hardware implementations using sinusoidal grating sequences. We used different stimulus frequencies ($f_0=0.02$ and $f_0=0.05$) and velocities (V=0.25 ppf and V=1 ppf). With these tests the hardware achieved results very similar to those of the software (less than 5% error in the speed of calculation).

Real sequences: overtaking-car segmentation

Only qualitative differences were estimated with both the hardware and software optical-flow approaches using real sequences (since the real flow is unknown). In this section we include some real image sequences for a qualitative evaluation.

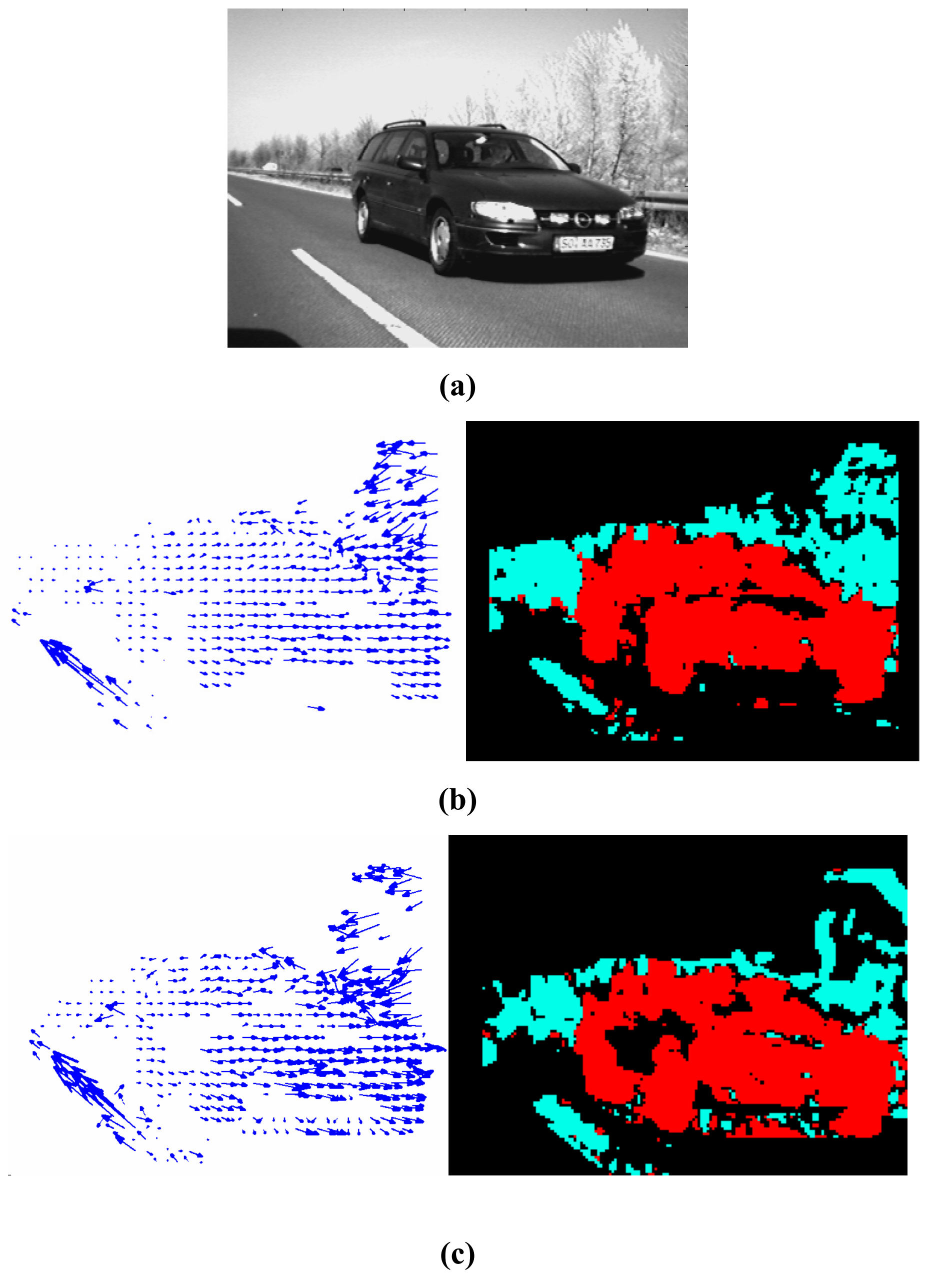


Figure 6.4. Optical flow for the overtaking car. Software vs. hardware estimations. (a) Original image extracted from the sequence. (b) Software result and (c) hardware result. The left-hand images use arrows to represent velocity vectors. In the right-hand images, for the sake of clarity, only leftwards (light colours) due to the landscape and rightwards (dark colours) due to the overtaking-car are used to indicate the motion. From this information the car segmentation is straight-forward.

Figure 6.4 contains the image of an overtaking-car sequence seen from the rear-view mirror, together with the results of software and hardware optical-flow estimations. This is a good example of how optical flow can be used for certain real-life applications in a very straightforward way (see Chapter 8 for more details about this real world application). In this example, the goal is the segmentation of the overtaking car, which can easily be done relying on optical flow, since the motion pattern of the overtaking car (moving rightward in the images) contrasts sharply with the landmarks, moving leftwards due to the egomotion of the host car. Note that the presented approach is based on a dense feature map but the motion segmentation problem can be addressed in multiple ways, using for example sparse feature maps as in [MOT06a], [MOT06b].

As shown in Figures 6.4.b and 6.4.c, the software results are smoother than those produced by the hardware. This is due to the bit-width restriction of the hardware approach. Nevertheless the results are quite similar and the accuracy of the hardware seems to be enough to obtain good qualitative results and address further processing stages such as car tracking.

6.3. Fine grain superpipelined optic flow architecture based on FIR temporal filters for the modified model of Lucas and Kanade

Temporal aliasing is one of the major error sources in motion estimation algorithms when they are forced to work with conventional sensors working at 30 fps. In order to reduce this temporal aliasing effect, some approaches implement multiscale schemes. These models require very complex architectures that imply very high hardware resources consumption in order to achieve real-time processing and in some cases their inherent parallelism is constrained (for instance by the warping method in multiscale approaches or due to the use of iterative algorithms). The approach described in the next sections, focuses on a different alternative. It consists in implementation of a very regular motion estimation approach that allows the utilization of high frame-rate cameras in an efficient way. Advances in imaging sensor technology make possible to acquire more than 1000 frames per second (fps) (see products from the web sites:

http://www.coreco.com, http://www.hitachi-service.net/, http://www.ims-chips.com/index.php3). The utilization of such kind of hardware reduces the motion range presented at the video sequences, allowing the simplification of the optical flow models and increasing the accuracy of the system [LIM05]. Although the 1000 fps frame-rate is still far away from our processing capabilities, the presented specific purpose computing architecture is able to compute at frame-rates significantly higher than the standard 25 fps.

This motivates the development of a high frame-rate optical flow computing system. Although we have presented in previous sections a description of a system able to process 41 Kps *(Kilopixels per second)* using the RC203 stand-alone platform, the approach presented here adopts a different algorithmic strategy according to the modifications proposed in [BRA97] (see Section 3.6.2 for more details). Furthermore, here we implement a novel superpipelined processing architecture capable of computing one pixel per clock cycle. A deep analysis of the circuit arithmetic has been required to achieve accurate results at affordable hardware resources. These innovative aspects make the presented approach outperform by one order of magnitude any previous system described in the literature. This outstanding performance mark allows real-time processing of oversampled frame-rates, which opens the door to the use of advanced image sensors in real-time motion estimation systems and their potential applications.

The addressed challenge is to design a novel architecture capable of processing optical flow at oversampled frame-rates. The state-of-the-art processing architectures (see Section 6.4) are unable to process 640x480 resolution images (we will assume this resolution in the rest of the work unless explicitly mentioned) at frame-rates higher than 13 *frames per second* (fps) for subpixels methods or 26 for correlation based approaches (see Table 6.9 in Section 6.4) and therefore a novel design strategy is needed. We will show in the following sections the architecture of a customized DSP designed on FPGA for this goal. We illustrate how the presented superpipelined architecture outperforms the up-to-now fastest processing system in more than one order of magnitude.

Appendix A reviews 3-D spatio-temporal sampling theory and investigates the effects of motion aliasing as well as the main limitations of the L&K model.

6.3.1 High frame-rate system motivation

The analysis in Appendix A motivates the use of a first-order-Gaussian-derivative kernel of 5 taps. According to the sampling theorem and as described on [LIM05], we can not compute reliably the velocity of high spatial frequency contents objects. We can just compute fast motion for low spatial-frequency objects using complex multiscale methods as described in [WEB95], [FLE90], [SIM91], [GAU02]. Therefore, we propose to increase the temporal sampling frequency to recover fast motion patterns and improve model assumptions.

We comment in the introduction the availability of image sensors with acquisition rates of more than 1000 fps at different image resolutions. The combination of such high frame-rate image sensors and a specific processing system capable of computing the incoming data stream can be of significant interest for a wide range of real world applications. The main improvements of such a system are the following:

1. The processing scheme is simplified, avoiding the complex multiscale approaches which require complex architectures and translate in higher system costs. The improved version of L&K optical flow model that we adopt combines high accuracy and implementation feasibility [LIU98], [DIA06a], [DIA06e].
2. Temporal aliasing is reduced significantly through high frame-rate sampling. This allows the computation of high spatial frequency contents of the image motion, increasing the density of the flow field.
3. Constant luminance condition is better satisfied [LIM05] through high frame-rate sampling. Therefore, first order constraint is better satisfied improving the accuracy.

Up to date there are only a few approaches capable of processing optical flow in real-time at standard video rates of up to 30 fps and faster processing systems are required to deal with high frame-rates in real-time. Nevertheless, even though we are able to compute the flow one order of magnitude faster than previous approaches, there is a trade-off between flow accuracy and image camera SNR. In general, SNR is proportional to the square root of the exposure time. Thus, higher frame-rate implicates lower exposure time which can make the noise increase dramatically. As showed in [LIM05], an oversampled factor of 3 (90 fps) dramatically increases the accuracy of motion estimation models but this strategy is not recommend for oversample factors higher than 6 due to the degradation of the image SNR.

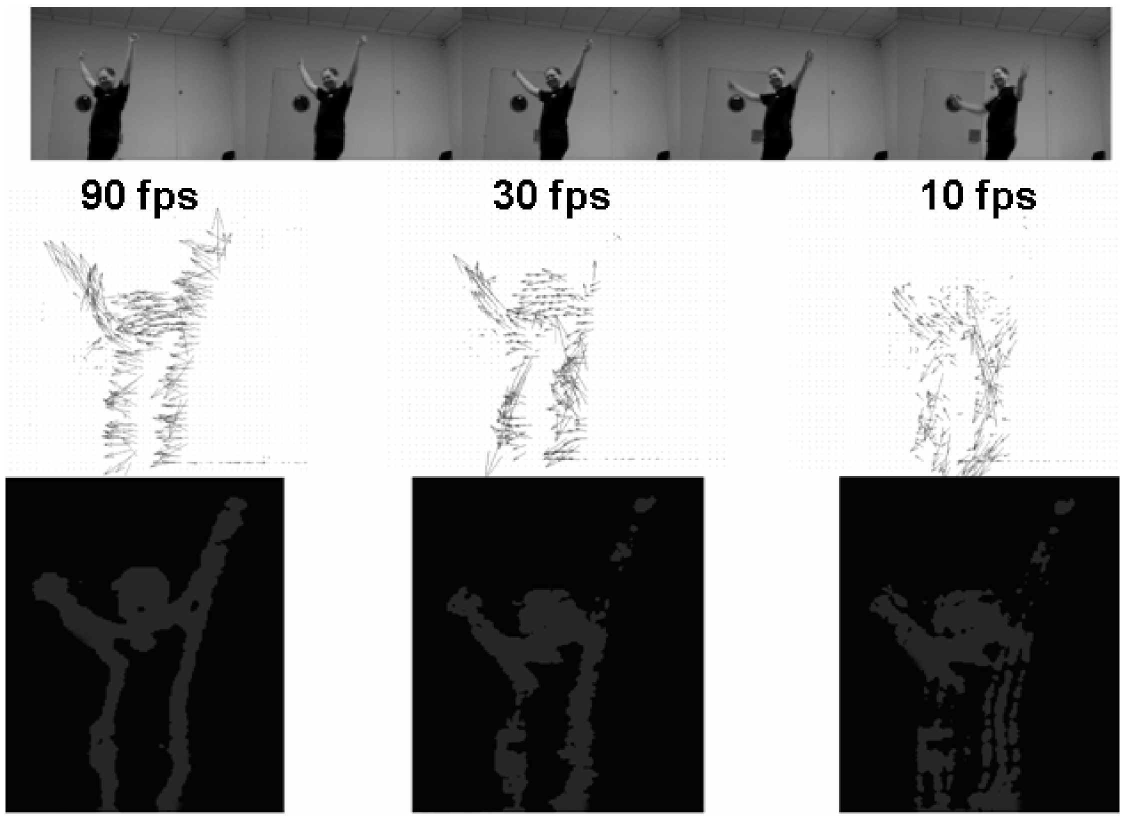


Figure 6.5. Qualitative effects of different frame-rates sequences acquisitions. The top row shows the example of a walking and moving the arms up sequence captured at 90 fps. The low level structure of the clothes allows us to focus on the motion at the body boundaries. Second row shows three optical flows computed at 90, 30, and 10 fps (using sequence subsampling) respectively. Finally, third row shoes the pixels over the confidence threshold for the sequences computed at the three different frame-rates. Note that though the walking movement is slow (the human-camera distance is approximately 4 m), the system is not able to compute its motion pattern at 10 fps. Still at 30 fps the flow is noisy, and some important details (such as the left arm pattern) are lost.

In Figure. 6.5 we show the qualitative results of an optical flow sequence computed at 90, 30 and 10 fps using a progressive area scan CCD sensor. Because the oversampling factor is small, as a first approximation we consider that image SNR is constant at the different frames-rates. The results shows that the flow computed at 90 fps is very homogeneous and stable and the confidence areas clearly identify the motion boundary which corresponds with the areas of higher spatio-temporal structure. The results computed at 10 fps are quite noisy, which can be easily derived from the reliability areas of the third row. Note that at 30 fps the flow quality is significantly degraded (compared with the 90 fps flow) and the fastest movements (for instance the left arm pattern) are lost. This shows the high interest of a computing system able to process the data stream of 90 fps sequences.

6.3.2. Hardware description

For the design of the presented architecture, although we maintain the coarse structure described in Section 6.2.1 [DIA04c], [DIA06b], each stage has been carefully redesigned. We would like to highlight the following points which are completely novel with respect to our previous work (Section 6.2):

1. Improved optical flow parameters according to [BRA97] are adopted. We have designed a 3-D complementary smoothing-derivative FIR filters based on Simoncelli kernels [SIM94] to improve the derivation operations. These kernels are first order Gaussian derivatives which properties have been described in Chapter 2. Although orientation requires second order derivatives, for optical flow computation, first derivative order has enough angular accuracy and requires lower resources. Their structure is shown in Figure 6.6.

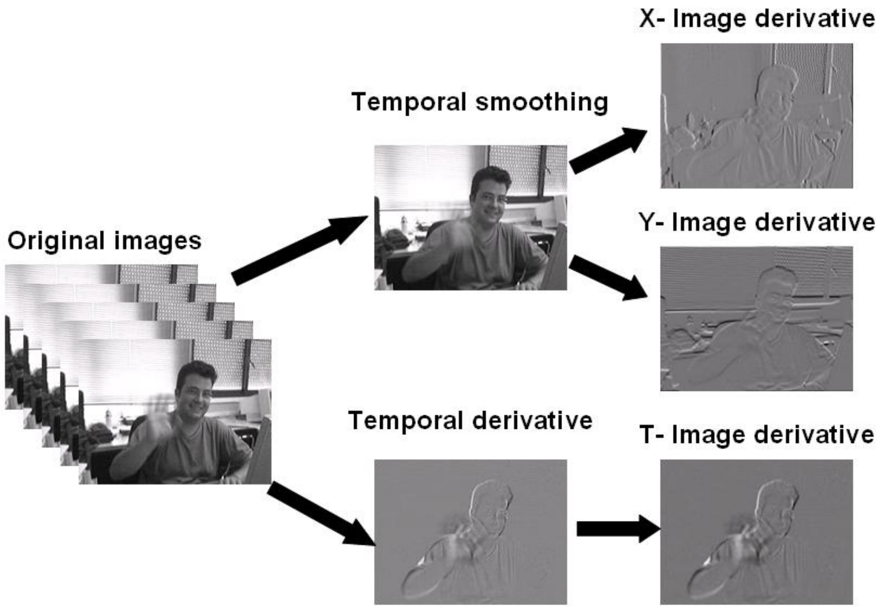


Figure 6.6. 3-D complementary smoothing-derivative filters structure. There are two basic primitives corresponding to the smoothing and derivation operation. First, we compute these two primitives on the temporal domain. Second, complementary smoothing/derivation are carried out on the spatial dimension. The final results correspond to a 3-D derivation over each spatio-temporal axe. For a high performance

design, all the operations (2 temporal and 6 spatial FIR filtering) are processed in parallel based on separable convolutions.

2. In our previous works, Section 6.2, we used the Fleet & Langley [FLE95] IIR filters for optical flow computation that are more hardware-friendly than FIR ones because they reduce the system memory requirements to three images. The drawback of this approach is that, due to the iterative process required for IIR operations, fixed-point arithmetic magnifies errors leading a significant degradation in the temporal derivative accuracy. The modifications adopted from [BRA97] allow using smaller FIR filters in the presented approach which makes this option more reliable.
3. We have implemented a superpipelined processing architecture able to compute one pixel per clock cycle as described in the next section.

6.3.3. From coarse to superpipeline architecture

We have motivated the interest of a high frame-rate optical flow processing architecture. The previous scheme (Section 6.2), with a pipelined structure divided on 5 basic stages, would lead to a remarkable performance but still far from high frame-rate processing requirements. The main reason is that the architecture in Figure 6.2 is similar to a DSP processor. There is a trade-off between pipeline length and system performance based on the dependence problems (branch conditions often break the pipeline which represents a significant waste of time). Therefore, long pipelines are not presented on standard DSPs and microprocessors. On the other hand, we describe here a specific purpose processing architecture that highly benefits of a fine grain pipeline datapath. In this way, we take full advantage of the regular data flow of the L&K algorithm. According to [FOR02], the best architecture is a superscalar and superpipelined structure. This design strategy has been adopted and it has been proven to be successful on other problems such as the ones described on [SAM06], [AGI06], [ROS06]. In Figure 6.7 we present the global scheme. Each coarse stage has been finely pipelined leading to a processing datapath of more than 70 stages. The number of scalar units grows at stages in which L&K model requires to maintain the system throughput. This parallelism expansion is outlined in the following:

- Stage S_0 uses one scalar unit for spatial smoothing.

- Stage S_1 uses two scalar units, one for temporal smoothing and another one for temporal differentiation.
- Stage S_2 uses three scalar units; corresponding to the three dimensions (I_x, I_y, and I_t) in which are computed the image derivatives.
- Stage S_3 uses five scalar units, corresponding to the five cross-products ($I_x \cdot I_x$, $I_y \cdot I_y$, $I_x \cdot I_y$, $I_x \cdot I_t$, and $I_y \cdot I_t$) which are used on the weighted squared sum at the least-squares fitting stage of the L&K approach (see Section 3.6).
- Finally, stage S_4 uses one scalar unit to compute the final motion for each pixel but internally several parallel pathways drive the data process. Therefore, this stage must be seen as a superscalar floating point unit.

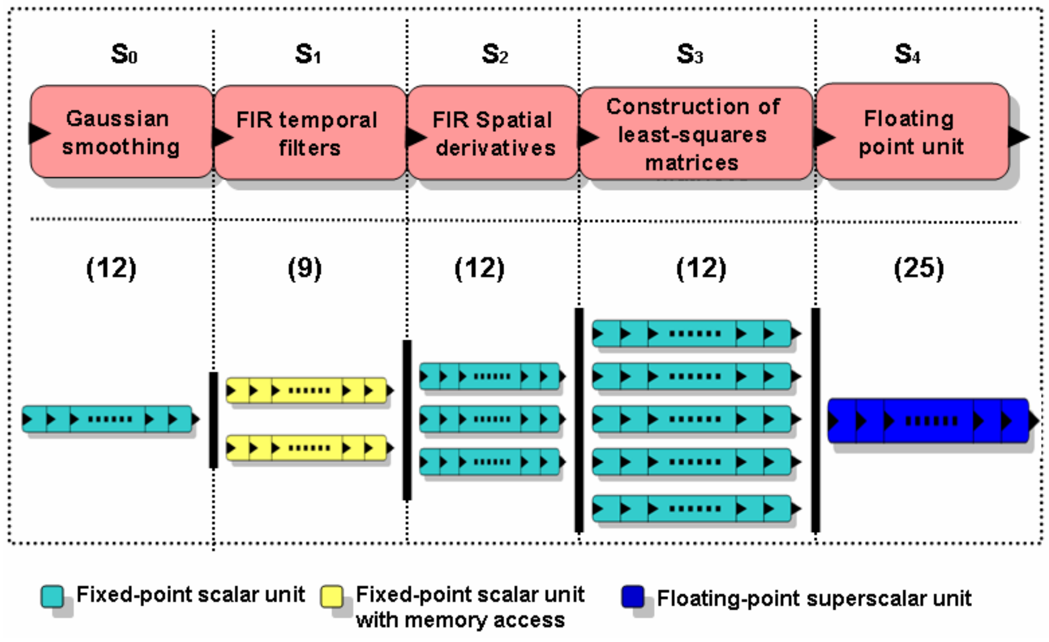


Figure 6.7. Optical flow processing core. Coarse pipeline stages are represented at the top and superpipelined scalar units at the bottom. Colors indicate the type of scalar units according to the legend at the bottom of the figure. The number of parallel datapaths increase based on the optical flow algorithm structure. The whole system has more than 70 pipelined stages (counting the motion processing core and other interfacing hardware controllers). This allows computing one optical flow measurement per clock cycle. The number of substages for each coarse-pipeline stage is indicated in brackets.

Figure 6.7 legend indicates the internal data representation of the scalar units. The number of parallel units is driven by the intrinsic model parallelism. Note that parallelism level is only schematically represented at each stage. There are some internal operations computed on parallel at each scalar unit to get the final throughput of one estimation per clock cycle (therefore some of these datapaths can be seen as superscalar units). We used fixed-point representation for all the stages but S_4, which uses floating-point representation. This has been a critical decision motivated by the large incoming bit-width at this stage which makes fixed-point representation very expensive in terms of computational resources. This topic is widely discussed in Section 6.3.4.

Details about the most representative device stages are presented on the next sections.

6.3.3.1. Memory Management Unit

The FIR temporal filters are highlights on Section 3.6.2 as the best solution for our system, but its specifications have special effect on the designing process considerations. The limited number of memory banks accessible on board constraints the available system parallelism (which translates in performance degradation) and increments the design complexity. Therefore, an efficient *Memory Management Unit (MMU)* becomes of great interest to abstract the sequential access inherent to this kind of devices. For this purpose, we create *Virtual Memory Ports (VMP),* whose behavior emulates parallel independent real memory ports. *High abstraction Hardware description Languages* (HDL) make it feasible to define systems at a high abstraction level, but finally, low level hardware imposes strong constraints on the feasibility of the system. We have designed a shell to expand the parallelism of this sequential elements in such a way that the design process of the system can be carried out without taking into account this low level considerations. According to this strategy, algorithmic implementations as the one proposed here can be designed at a higher abstraction level. The main idea for this implementation is to combine the following concepts/properties:

1. Nowadays, long memory words (36 bits) make it feasible to store up to four 9-bit-width data at each memory address with more than 512Kaddress [IDT06] (up to 5 images of 720x576 pixels per memory chip).
2. A throughput of one pixel per cycle is possible by using pipelined packing and unpacking circuits, which only requires to access memory once in 4 clock cycles.

We have designed an MMU which benefits from the previous architectural descriptions. Depending on the number of VMPs required and packing/unpacking possibilities (provided by the memory word bit-width), a state machine is used to feed the VMP registers sequentially, achieving a final performance of one data per cycle. Furthermore, this architecture is scalable because an increment of N in the number of VMPs available on one memory only modifies the required access cycles on a factor of N. This can be further optimized by incrementing the MMU clock frequency by this factor with respect to the global system clock frequency. There is only one limitation, due to the packing and unpacking circuits, random access is limited to an addresses multiple of 4. Besides, for an efficient data management, they should be stored on memory in a consecutive packed way.

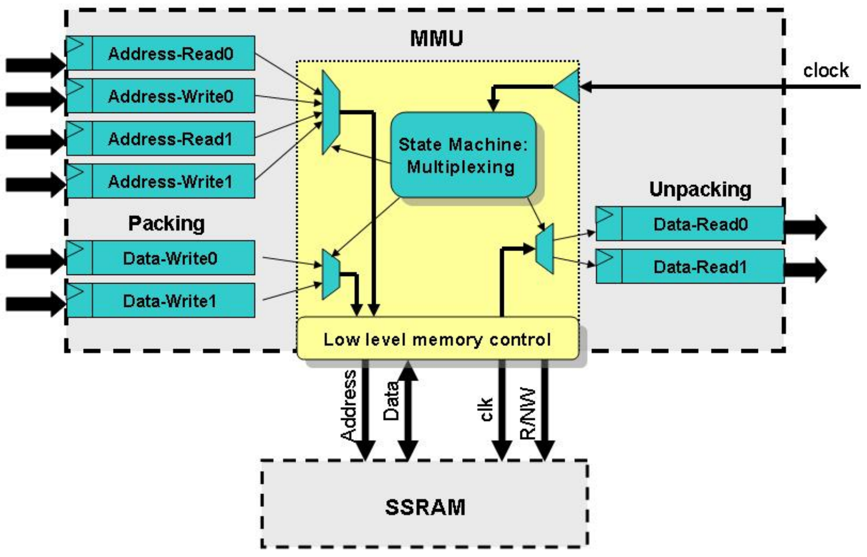


Figure 6.8. MMU schematic for a 4 VMPs expansion-case. VMPs are represented by one address register (type Read or Write) and a Data-Write or Data-Read register. Low level memory control manages the data and address signals as well as the SSRAM clock, read-no-write signal (R/NW), etc. The state machine feeds four VMPs sequentially and manages the low level memory access. Packing/Unpacking circuits achieve a total throughput of one pixel per clock cycle. This architecture allows us to multiply by 4 the equivalent memory parallel access.

The MMU architecture is illustrated on Figure 6.8 for a four VMP case. Note that a VMP is composed of a 4 addresses register (read or write type) plus a data-write register with packing circuits or by a data-read register with unpacking circuitry.

The new high level abstraction provided by the MMU makes the implementation of FIR temporal filters feasible by using a high abstraction level description. Previous implementation of L&K used IIR filters to reduce the memory access, but the drawback was the accuracy degradation [FLE95]. The presented architecture allows the easy management of a large number of read-write processes necessary for FIR temporal filters with a minimum FPGA logic, which clearly justifies the design of the presented MMU architecture.

6.3.3.2. Stage S_3 architecture

This stage is the one which requires more hardware resources. As showed on Figure 6.9, this stage is expanded to compute the five cross-products utilized for the least squares fitting process. This is implemented as five parallel segmented scalar units. Furthermore, incoming data for this stage require 18 bits, which makes the arithmetic circuits consume a considerable circuit area.

As illustrated on Figure 6.9, the main computing circuit of this stage is the separable convolution unit which implements the weighted average calculation.

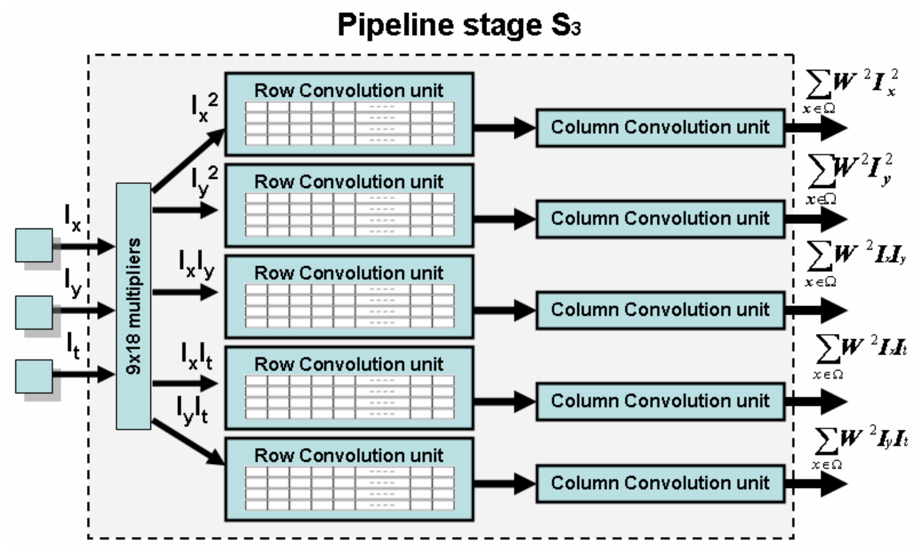

Pipeline stage S3
Row Convolution unit
Column Convolution unit
9x18 multipliers
I_x
I_y
I_t
I_x^2
I_y^2
I_xI_y
I_xI_t
I_yI_t
$\sum_{x\in\Omega} W^2 I_x^2$
$\sum_{x\in\Omega} W^2 I_y^2$
$\sum_{x\in\Omega} W^2 I_xI_y$
$\sum_{x\in\Omega} W^2 I_xI_t$
$\sum_{x\in\Omega} W^2 I_yI_t$

Figure 6.9. Architecture schematic of the least squares matrices construction. Pipeline stage S_3.

6.3.3.3. Stage S_4 architecture

Stage S_4 is critical in terms of system frequency, resources, and accuracy. The incoming data use fixed point representation of 18 bits and this stage requires the operations of multiplication, addition/subtraction and division without losing accuracy. As discussed on Section 6.3.4., the best arithmetic representation for this stage is floating point data which allows obtaining the required precision with reasonable resources consumption (as is shown on Table 6.5). Figure 6.10 presents the architecture of this stage, based again on a high pipelined and parallel datapaths to achieve a high system throughput. The whole stage requires 25 steps. Data conversion, multiplication, addition, and subtraction are computed in just one cycle. The superscalar division unit of n-ways presented in Section 6.2 does not suit well the specifications of this new architecture. The requirement of MNC=1 constraints the division units to pipelined version because a superscalar version is more expensive for very large parallelism level. Therefore, a pipelined floating point divider has been designed, requiring 15 steps for the current custom float type.

S_4 is the stage limiting the system clock frequency and it could be even further pipelined to increase the clock frequency if necessary.

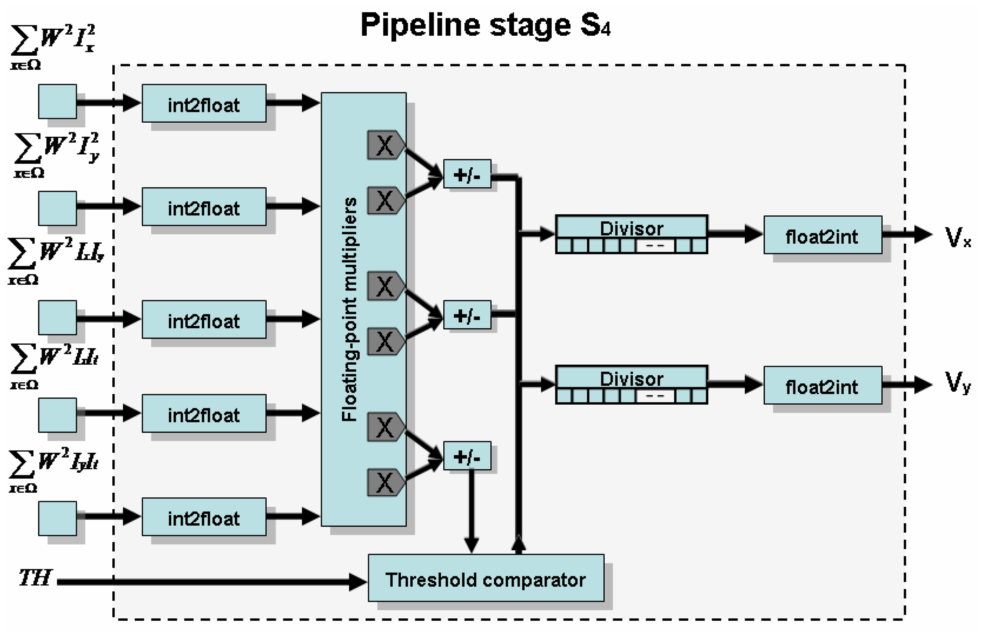

Pipeline stage S4
$\sum_{x\in\Omega} W^2 I_x^2$
$\sum_{x\in\Omega} W^2 I_y^2$
$\sum_{x\in\Omega} W^2 I_x I_y$
$\sum_{x\in\Omega} W^2 I_x I_t$
$\sum_{x\in\Omega} W^2 I_y I_t$
int2float
Floating-point multipliers
X
+/-
Divisor
float2int
Vx
Vy
TH
Threshold comparator

Figure 6.10. Architecture schematic of the floating-point unit. Pipeline stage S_4.

6.3.3.4. Global system superpipelined datapath description

We have described the main computing stages on the previous subsections. The parallelism of the system is expanded according to the algorithmic structure and the final architecture is described on Figure 6.11. Note that the MMU units are critical allowing the management the memory accesses of the different elements as described in the figure.

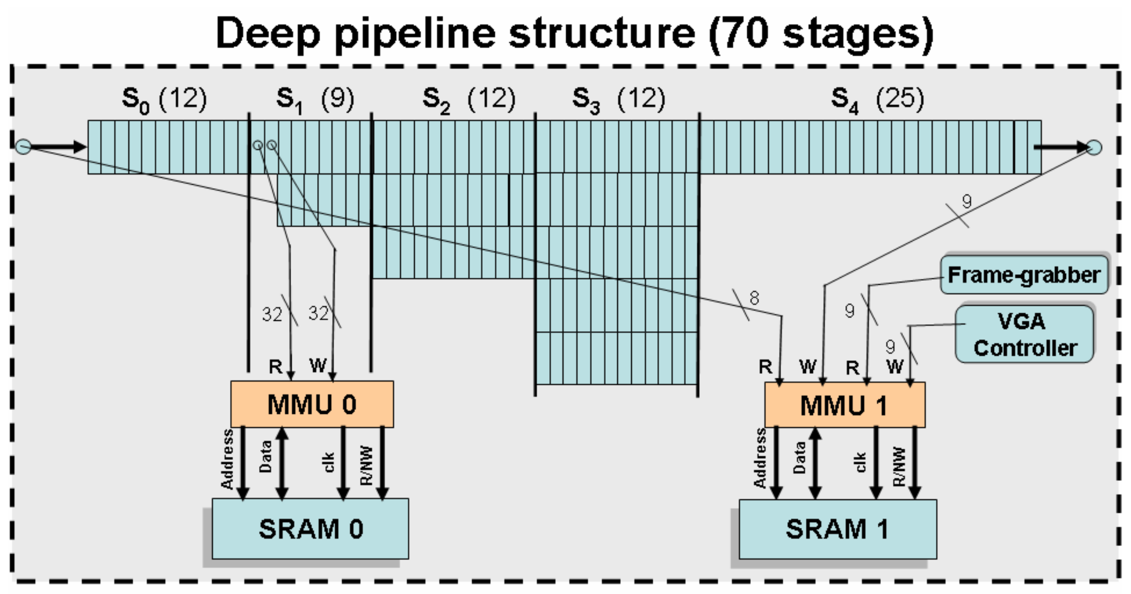


Figure 6.11. Architecture schematic of the global system. We represent the number of functional units (as parallel pathways), fine grain pipeline stages (indicated as rectangular cells and in brackets in the top labels corresponding to each coarse grain pipeline stage), and memory access elements (MMU channels are included).

The synchronization among the different processing units (frame-grabber, motion processing, VGA, and user interface) is accomplished by using specific external memories as data buffers, which solves the problem associated to the different clock frequencies. The memory interchange strategy makes use of delays between processing units as synchronization technique. This enables the design of a very deep pipeline processing structure without using branch predictions that would degrade the overall performance. The high system throughput is based on this deep pipeline and on the parallel scalar units of different stages designed according to the Lucas & Kanade algorithmic complexity. Well balanced units are used to achieve a final system throughput of one estimation per clock cycle. Only on specific points (for example VGA controller) interprocess communication is needed. On these situations, we use an interface module between the two processes, with a synchronized and buffered point-to-point communication scheme. This module blocks the communication until both modules (sender and receiver) are ready and data is transferred, allowing synchronizing hardware controllers with different clocks or other characteristics.

6.3.4 Bit-width optimization

The proper selection of the data structure is a key factor for a successful implementation of a customized system. First we need to consider the system accuracy vs. resources consumption trade-off. The best accuracy is obtained with double floating point representation, (this is the choice mostly adopted in software approaches). This choice leads to a maximum precision at the cost of low performance. Custom processors for real-time systems lack of large resources compared with the current general purpose processors. Because of that, customized datapaths need to be carefully designed in order to achieve comparable of higher performance. Typically, the customized systems use fixed point data representation because it fits better on current digital technology but this strategy requires a careful analysis to avoid accuracy degradation. Second, a well bit-width design has significant importance in terms of power consumption [CON03]. Low significant bits tend to switch their state more frequently and this shall be avoided if it does not drive any information. Due to that, the smart elimination of low significant data allows to decrease the bit values transition, reducing the switching power which is important for migration to VLSI devices.

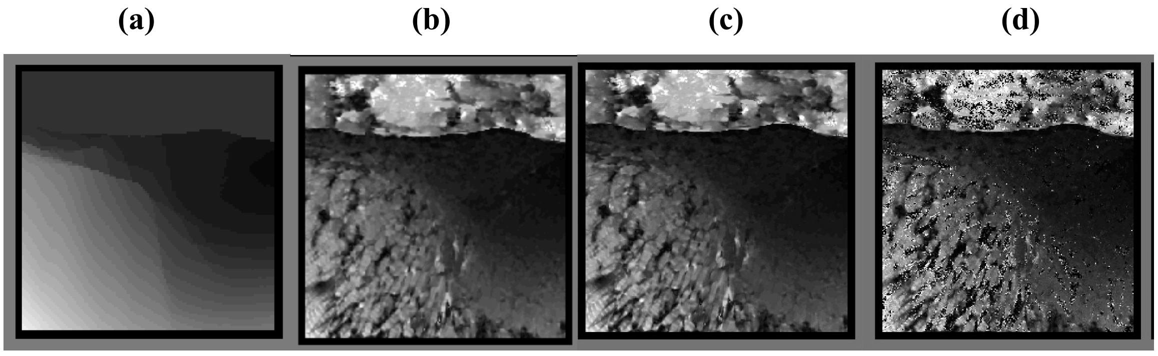


Figure 6.12. Module of the velocity for the Yosemite valley through flow sequence (thresholds not used). From left to right: (a) velocity real ground-truth module, (b) module computed with the software program, (c) module computed with the designed system, (d) module computed with inadequate bit-width for stage S_4. Note that the quantization errors are visible as a salt and pepper like noise but there is no significant difference between the module of the software (b) and our hardware implementation (c).

In our system designing process we have used several measures and strategies to evaluate how well the specifications are fulfilled. First, concerned that our goal is to optimize the optical flow accuracy, we have used the *MCode for DSP Analyzer* with several bit-width configurations and the well known synthetic sequence of flow-through across Yosemite Valley [BAR94] as reference to measure the degradations of each approach with respect to a double precision data representation. The decision of the bit-widths at each stage has an important impact on the accuracy of the system. Figure 6.12 shows the effects of insufficient bit-width utilization for the stage S_4, which leads to a significant quantization noise (visually similar to salt and pepper noise).

Stages S_0 to S_3 have been implemented using fixed point arithmetic because these operations are based on convolutions. After an extensive analysis of the accuracy vs. resources consumption trade-off, we have decided to implement stages S_0 to S_2 with 9 bits and stage S_3 with 18 bits. This corresponds to the use of one fractional bit on the image derivatives. With this configuration and using a threshold that allows an optical flow density of 36.47 %, the angular error and its variance changes from 3.38º (8.93º variance) for double floating point to 3.43 (8,96º variance) for fixed point representation. This can be seen in Figure 6.13. This bit-width choice is also motivated by hardware device structure (some of the internal hardware resources, such as embedded multipliers and memories, are optimally used for a maximum of 18 bits) and to the fact that the absolute error increment remains very low (0.05 º).

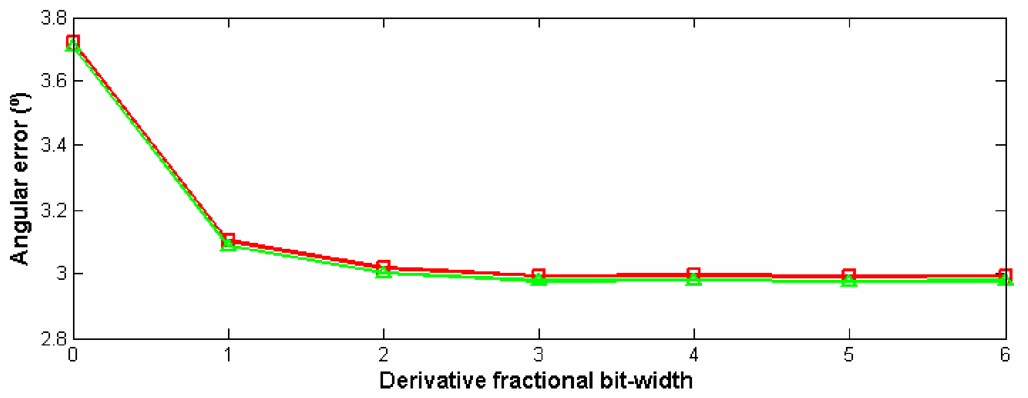


Figure 6.13. Angular error considering image derivatives of different bit-widths. We take an integer part of 8 bits plus different values of the fractional part. The next circuit's stages are designed according to this decision. The dark line (red) corresponds to a mantissa of 6 bits utilized at stage S_4. Light one (green) corresponds to a 12 bits implementation. There is no appreciable improvement for larger mantissa bit-widths. It is clear that the fixed point stages dramatically compromises the whole system accuracy compared to bit-width decision at the mantissa in stage S_4, i.e., stage S_4 can not take advantage of larger bit-widths if the precision is restricted in previous stages.

Note that on the datapath, previous stages accuracy limits the maximum precision achievable by next stages and due to that their requirements shall be carefully studied taking into account the whole datapath. According to this, increasing the bit-width of these stages only makes sense if the next stages also increase their bit-widths in order to obtain higher system accuracy.

At stage S_4 the incoming bit-width is too large and specific arithmetic representation is required. With these precision requirements, customized floating point arithmetic produces faster circuits with lower system resources. In table 6.5 an implementation using 36 bits fixed point arithmetic achieves similar accuracy to the one using a customized representation with 19 bits but the resources consumption is 6 times larger and the maximum system clock frequency is more constrained (by a factor 0.7). This short study motivates the use of floating point arithmetic in S_4.

Table 6.5. System accuracy vs. resources consumption trade-off based on stage S_4 precision and data format. Hardware resources in terms of gates consumption taken from the DK synthesizer [CEL06b]. Optical flow error measure using Fleet & Jepson method [FLE92] at a density of 36.44%. (*man* stands for mantissa and *exp* for exponent).

Pipelined stages	**NAND gates**	**FFs**	**Max clock frequency (MHz)**	**Angular error (º)**	**Error variance (º)**
S_4 Floating point unit (11 man + 7 exp)	57167	3488	45	3.42º	8.95º
S_4 Fixed point unit (36 bits)	345981	1080	31	3.37º	8.93º

For the bit-width decision of stage S_4 we have carried out an extensive batch of simulations of different cases. The main parameters under study being are the angular error, its variance and the SQNR of the signal. These parameters are represented against the mantissa bit-width of S_4 on Figure 6.14.

Once decided the bit-widths at different stages we evaluate the degradation due to quantization errors of our design. In this way, we check the accuracy of different implementations measuring the results in the same pixels (and thus same densities). Table 6.6 summarizes these results. As it is shown, the system still achieves a very high accuracy. It validates our previous data analysis and confirms that the high bit-width used at the standard computer can be significantly reduced.

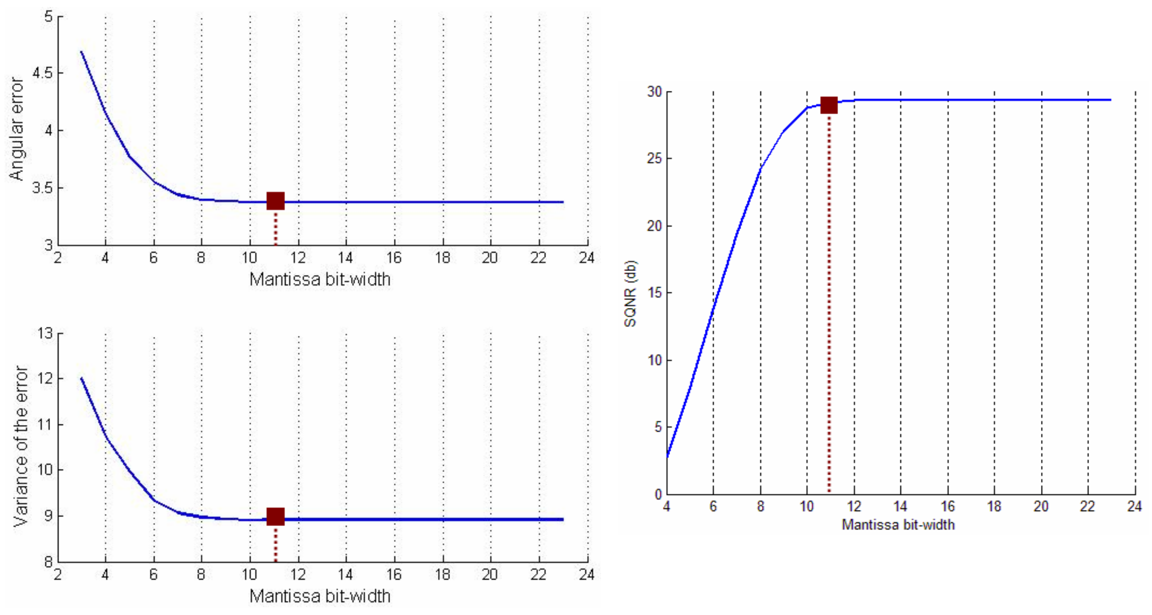


Figure 6.14. Angular error (top-left), variance (bottom- left) and SQNR (right) against the number of bits of the mantissa in stage S_4. Please note that error and variance are approximately constant for values larger than 10 bits and SQNR becomes also stable for values higher than 12 bits. As good trade-off, we choose a mantissa of 11 bits that allows a larger accuracy similar to double floating point arithmetic with high SQNR.

Table 6. 6. Comparing angular errors between the software double data type and the customized data structure used in the hardware implementation. Note that though the bit-width has been dramatically reduced, due to the previous analysis the results obtained with the hardware approach are only slightly degraded even at significant estimation densities.

Density	**Software double data implementation**		**Customized hardware implementation**	
(%)	**Angular error (º)**	**Variance (º)**	**Angular error (º)**	**Variance (º)**
36.47	3.4330	9.1056	3.5166 deg	9.2406
42.14	4.0731	9.8389	4.2128	10.0986
57.20	6.9166	12.9283	7.8611	14.5013
91.95	17.3303	20.3685	18.3185	20.6841

A comparative analysis of the optical flow reliability areas vs. the high quantization error areas highlights that the scenarios with lower confidence are the candidates to produce noisier results due to the quantization effects. Fig. 6.15 illustrates this effect. In Figure 6.15.a we visualize confidence values at each image pixel. Bright grey levels mean higher confidences. In the right image (Fig. 6.15.b) we represent the angular error difference between software and hardware estimations. In this image, light areas represent pixels with higher quantization noise. This image clearly shows that the areas prone to larger quantization noise are the ones with lower optical flow confidences. Therefore, the confidence filtering efficiently helps to neglect misestimating driven by quantization errors.

This effect can be easily understood based on the image structure. Image pixels with low confidence correspond to areas of low structure which produce smaller image derivatives and therefore the division operations required to estimate motion is prone to errors. When these results are computed using a limited bit-width the relative importance of their quantization is larger and translates into erroneous estimations. This motivates the utilization of a proper threshold for the optical flow computation, allowing us to get high accuracy circuits with lower resources consumption and rejecting not reliable estimations (i. e. prone to quantization errors).

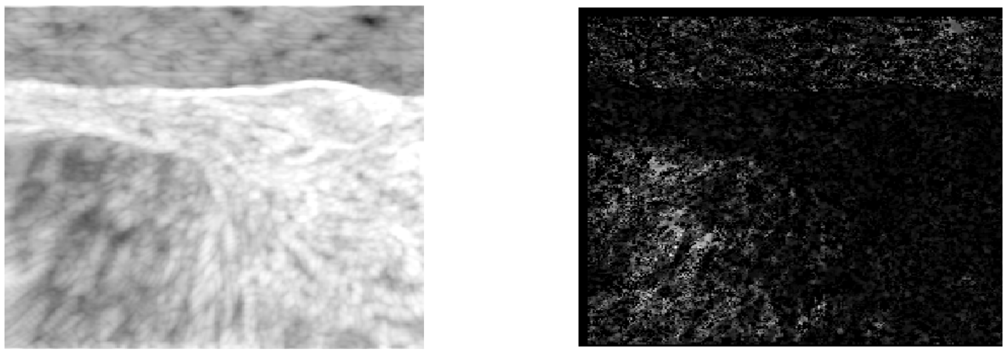

(a) **(b)**

Figure 6.15. Confidence areas and quantization error. (a) Logarithm of optical flow confidence values (light grey means high confidence estimations). (b) Software-hardware angular error difference. Data range (logarithmic) scaling is done to improve visualization. Note that areas prone to higher quantization noise correspond with the lower optical flow confidence areas.

6.3.5 System resources and performance

The whole system benefits of the accuracy analysis of the previous section and has been successfully implemented on a stand-alone board for embedded image processing applications. The used prototyping board is provided with a Virtex II XC2V6000-4 Xilinx FPGA as processing element including also video input/output circuits and user interfaces/communication buses. Illustrative results are shown in Figure 6.16. Tables

6.7 and 6.8 show the hardware costs of the motion processing core as well as the control/interface logic. After synthesis, the whole system consumes 24 % of the FPGA slices, 20% of the available embedded memory and 8% of internal multipliers. Its maximum frequency clock rate is 45.5 MHz. The image resolution can be selected according to image input camera standard or processing capabilities.

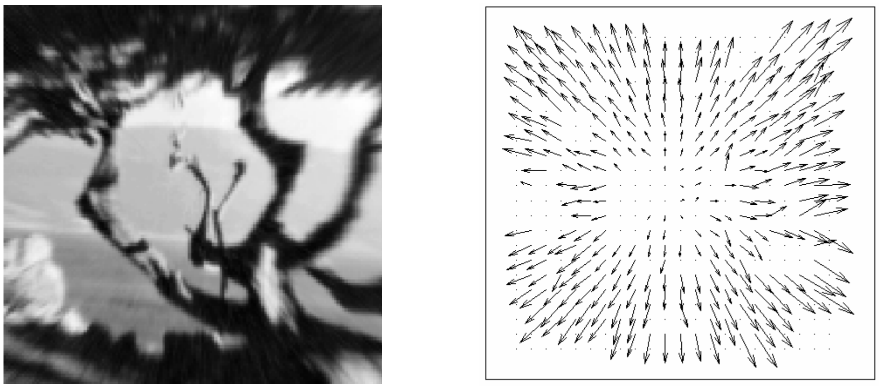
Table 6. 7. System stages gates resources consumption (results taken from the DK synthesizer [CEL06b]). First row: interfaces and hardware controllers (camera frame-grabber, MMUs, VGA signal output and user configuration interface). Second row: motion processing core.

Pipelined stages	NAND gates	FFs	Memory bits	Max clock frequency (MHz)	Image Resolution	Fps
Interfaces + hardware controllers	65881	2363	18208	45	640x480 1280x960	148 37
Motion Processing core	1145554	6529	516096	45,5		

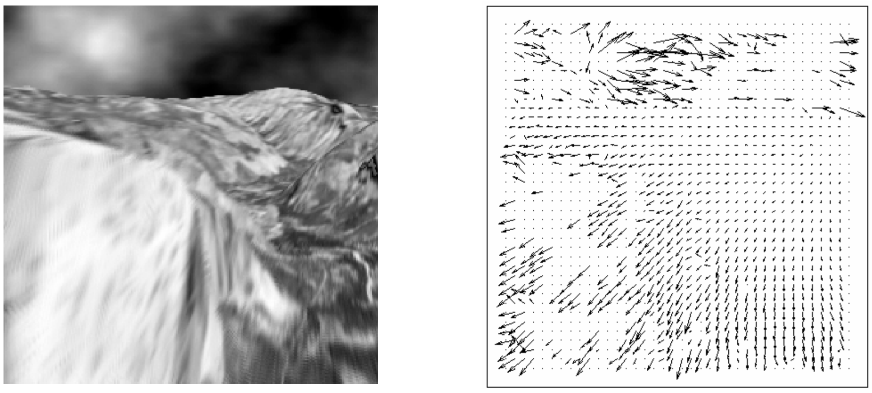
Table 6. 8. System resources required on a Virtex II XC2V6000-4 after bit-stream generation with Xilinx ISE Foundation [XIL06d]. The system includes the optical flow processing unit, memory management unit, camera Frame-grabber, VGA signal output generation and user configuration interface. (*Mpps*: *mega-pixels per second* at the maximum system processing clock frequency, *EMBS*: embedded device memory).

Slices / (%)	EMBS / (%)	Embedded multipliers / (%)	Mpps	Image Resolution	Fps
8250 (24%)	29 (20%)	12 (8%)	45.49	640x480 1280x960	148 37

This architecture is modular and scalable, being possible to reduce the system parallelism (and performance) to fit on smaller devices as illustrated in Section 6.2. Furthermore, the processing core can be replicated (more than 75% of system resources are available) and the Frame-grabber can be easily modified (thanks to the MMU architecture) to split the image and send it to several processing units. This high level scalability allows multiplying the processing performance if it is required.



(a)



(b)

Figure 6.16. Optical flow processing results for a coupled of sequences used in [BAR94] and available at [VIS06]. (a) Diverging tree sequence. (b) Yosemite Fly-Through sequence. The flow is represented as vectors with module proportional to the motion speed.

6.4 Systems performance comparison with other approaches

The implementation of the optical-flow algorithm with FPGAs has only been addressed by some authors in very recent years. Table 6.9 summarizes the performances obtained by the different approaches. In our previous work described on Section 6.2, [DIA04c], [DIA06b], the basic implementation of L&K model was proposed and we presented a detailed study about the performance vs. system resources trade-off. Although the performance was already high, neither the image resolution or frame-rates complaint the high frame-rate requirements addressed by the system described in Section 6.3. It can be seen that the approach described in Section 6.3 outperforms any previous system in more than one order of magnitude.

The iterative algorithm of Horn & Schunk (H&S) [HOR81] has also been implemented by different authors. Martin et. al. [MAR05] presented a system implementation that fits quite well the specification of a standard frame-rate optical flow system. The main disadvantage of that approach is that the model itself obtains poor accuracy (compared to L&K) as shown by Barron et al. [BAR94]. There is also a work from Cobos et al. [COB98] which described the implementation of the H&S model but using modest resources and therefore achieving lower performance. Using the block-matching approach, the implementation described by Niitsuma & Maruyama [NII04] achieves 30 fps of image size 640x480 but with high hardware cost (90% slices of a XC2V6000 FPGA) and without sub-pixel accuracy.

Based on the L&K approach, Correia & Campilho [COR02] presented a real-time implementation of the system using a MaxVideo200 pipeline image processor. Though still far from our results, they obtain a high performance (1666 Kpps) because they take full advantage of the pipeline architecture Nevertheless, the use of an acceleration processor (such as the MaxVideo200) makes difficult the transference to embedded platforms.

Table 6. 9. Comparison to prior works. Our data uses the maximum available system clock frequency for comparison with previous approaches.

Implementation	Throughput (Kpixels/s)	Image size (pixels)	Image rate (frames/s)
Improved L&K (described section 6.3)	45490	640x480 1280x960	148 37
L&K (stand-alone board, table 6.4, first row)	4100	320x240	53
L&K (PCI-board, table 6.4, second row)	2303	320x240	30
H&S (Martin et. al. [MAR05])	3932	256x256	60
Block-matching, (Niitsuma & Maruyama [NII04]*)*	9216	640x480	30
L&K (Correia & Campilho [COR02]*)*	1666	256x256	25
H&S (Cobos et al. [COB98]*)*	47.5	50x50	19
Variational (Bruhn, et al. [BRU05a]) Intel Pentium 4, 3 GHz	1444.7	316x252	18
Intel Pentium 4 HT (3.2 GHz)	914	160x120	47.6

For standard PC platform, the approach described in [BRU05a] presented a high accuracy but performance is still far away from the presented system and the used method is not suitable of easy implementation on embedded devices.

Finally, the model described here, running in software on an Intel Pentium 4 HT, 3200 MHz, can compute 47.6 fps of 160x120 pixels (914 Kpps as indicated on the last row of Table 6.9) though this can be further optimized using MMX and SSE instructions. But in any case this requires the full computing power of the machine.
Since the referenced works are very recent (some of then using even the same evaluation devices), the outstanding performance of our approach is not provided by technology improvements but rather by a very efficient processing architecture (superpipelined and superscalar datapath) that extensively uses the parallel resources of the FPGA device.

6.5. Conclusions

This chapter presents two optical flow systems which represent the state-of-the-art in the field of high performance motion estimation systems on chip. The chapter describes mainly two different alternatives:

1. A coarse grain pipelined datapath for flow estimation based on IIR temporal filters and the original Lucas & Kanade algorithm. This system shows how an optical-flow estimation circuit can be implemented using an FPGA platform as a customized DSP for a specific purpose. We describe a scalable architecture that can work with large image data at a conventional video-frame rate (30 fps). System performance, customization feasibility and scalability, due to the FPGA technology and design strategy, allow the use of the system in diverse application fields as explained in previous sections. The modularity of the system also enables the easy alteration of the computing scheme to target different computing speed vs. hardware cost trade-offs.

   The accuracy of the estimated flow is essential for some applications (for instance the one described in Chapter 8). We have studied how the restricted bit-width of the different computations affects the quality of the extracted optic flow and compared the results obtained with software implementations of the algorithm computed with double precision. The results of the hardware implementation described are in the range of other software approaches considered in the study of Barron et al. [BAR94]. Therefore, the performance of the hardware is of reasonable quality provided that it computes in real time (at a speed of 1776 Kpps with our PCI board system, 4100 Kpps with the RC203 stand-alone platform).

2. Fine grained superpipelined datapath based on FIR temporal filters and the modified version [BRA97] of the Lucas and Kanade algorithm. The necessity of a system for high frame-rate optical flow systems has been clearly motivated. Current image sensors make possible very fast image acquisition which lead to significant improvements on the optical flow accuracy [LIM05]. Simple gradient based optical flow approaches seem to be a suitable alternative for moderate cost systems (compared with complex multiscale approaches). According to this we have implemented an improved version of the L&K model [BRA97] which complements the capabilities of high frame-rate cameras providing real-time image motion analysis of high accuracy.

   We have presented an architecture proved to be scalable, modular, and versatile and we have described how parallel and superpipelined structures can be utilized on the implementation of algorithms to design novel high performance architectures for image processing. The system outperforms in more than one order of magnitude any previous approach validating the adopted computing scheme. From the accuracy analyses we conclude that, with a much reduced bit-with data representation, the designed system achieves accuracy close to the software implementation (with double precision floating point representation).

   We finally have evaluated the system resources consumption and performance of an implementation on a stand-alone platform which fulfils the high frame-rate optical flow requirements. The comparison with previous works clearly shows the outstanding performance of the system and opens the door to a wide range of application fields.

The second model can be considered a fully parallel implementation of model described in Section 6.2 but the spatio-temporal differentiation stages plus the fine-pipeline require an especial design which motivates considering it as a complete new circuit.

It is also worthwhile to comment that the resources required for the version described in Section 6.3 are lower than the ones used for the HSHQ model (section 6.2) implemented on the same family device (see Tables 6.3 and 6.8). This can be understood based on the careful bit-width design (for example, the system described in

Section 6.3 uses less bits in the floating point unit than the circuit presented in Section6.2, as indicated by the accuracy analysis done) and also for the architectural improvement (on sequential designs not all the hardware is shared and therefore some parts of the circuits are not optimally used). This analysis support our hypothesis that high parallel devices can be optimal also in term of hardware resources compared with sequential circuits if a carefully design is done.

Chapter 7
High Performance stereo computing architecture

This chapter proposes a high performance stereo computing architecture. The system is structured in a fine grain pipelined datapath (with some superscalar critical stages) able to provide one stereo estimation per clock cycle.

The final implementation of a specific model in a specific purpose datapath with constrained precision and fixed-point arithmetic can be considered as a novel approach (though with behavior close to the software model based on high precision computation resources). This chapter includes a study of how the limited precision in the critical computations affects the global accuracy of the system and how many hardware resources are required when increasing the bit-width at these critical stages.

In the final part of the chapter the performance of the presented architecture is evaluated and compared with other approaches described in the literature.

7.1. Introduction

In Chapter 4 was described the stereo model in which we will focus now. It is a hardware friendly phase-based model in which we compute the phase difference (disparity estimation) without an explicit calculation of the phase of each of the two images. We have used Gabor as quadrature filters but, depending on the resource-sharing strategy, the Second Order Gaussian derivatives described in Chapter 5 also can be used.

This approach has several advantages that make the system hardware-friendly [DIA06h]. Although equation (4.12) increases the number of multiplications compared to circuits with direct phase subtraction, current FPGA devices include embedded multipliers for DSP operations, which make this technology of particular interest for vision tasks. In fact, the main advantage provided by this approach is that of avoiding the explicit logic required for the wrap-around mechanism. This implies reducing comparison logic considerably. Furthermore, the number of division operations is reduced by 50%. This reduction is important because this division using fixed-point arithmetic requires high precision. In fact, quantization errors make the former approach noisier and thus demand more hardware resources to achieve similar accuracy.

To address the hardware implementation of this approach the basic steps can be summarized as follows:

1. DC component image removal using the local image contrast $I\text{-}I_{mean}$ operator for the even quadrature filters.
2. Even (C) and odd (S) 1-D quadrature filters convolution of left and right images.
3. Direct phase-difference calculation from (4.12).
4. Disparity computation using equation (4.10), assuming $k(x) \approx k_0$.

7.2. Hardware implementation

Most of the previous real-time contributions described in the literature are based on correlation techniques [BRO03] because this approach fits in quite well with customized hardware but the choice of a phase-based stereo approach is also justified because of its capacity to reduce illumination problems. As mentioned in [COZ97], a contrast test shows that this approach is not very susceptible to differences in local contrast. It also seems to be capable of dealing with imbalanced images too, which are usual in real cameras since they have slightly different luminance gains.

The phase-based approach can also be of interest in biological studies to extract real-time working models based on features (phase) similar to the ones that are thought to be used by biological systems [FLE96] or [CHE04].

7.2.1. Camera calibration

Setting up the system requires image rectification and camera calibration (which is a critical stage). After a manual calibration to arrange the cameras in parallel, the current implementation only involves a simple pre-processing method based on image displacement, which runs in a set-up system configuration stage as follows:

a. We define a plane of null disparity and we allocate a flat object or picture on it with some texture (for instance a chessboard pattern).

b. A frame-grabber shift of up to 32 pixels is explored iteratively along the horizontal and vertical co-ordinates to obtain the best overall matching value within that range (integrating 4 times in each iteration to reduce the error due to camera noise and image flickering). This plane defines the zero disparity distance. Closer and farther objects will lead to positive and negative disparities respectively. This reference plane is defined depending on the cameras configuration, system scale and target application to properly tune the filter disparity range to the target scenario (close to the reference plane). With this method we reduce the range of disparities presented at the image close to this reference plane, which allows us to recover disparities with only small quadrature filters kernels.

c. When the calibration process finishes, the system is auto-reprogrammed from external Flash memory with the new configuration file and the stereo computation starts.

This simple calibration process takes about 32 seconds using a 40 MHz FPGA clock but this time is not critical since the calibration is computed only once at the initial configuration stage. In fact, up to a 70 MHz clock frequency is supported by this circuit, but we only use a 40 MHz clock to facilitate the on-line generation a VGA output of the imaged matching process. In this way, we are able to visually monitor the calibration process (iterative matching). This allows discarding wrong initial camera settings that lead to poor matching results. Future work will address an improved calibration pre-processing including image rectification techniques.

7.2.2. System architecture

Handel-C [CEL06c] allows us to define very straightforwardly the level of parallelism and pipelined structure, which can be easily grouped on the basis of functionality and finely sub-divided to get well balanced pipelined structures of high data throughput.

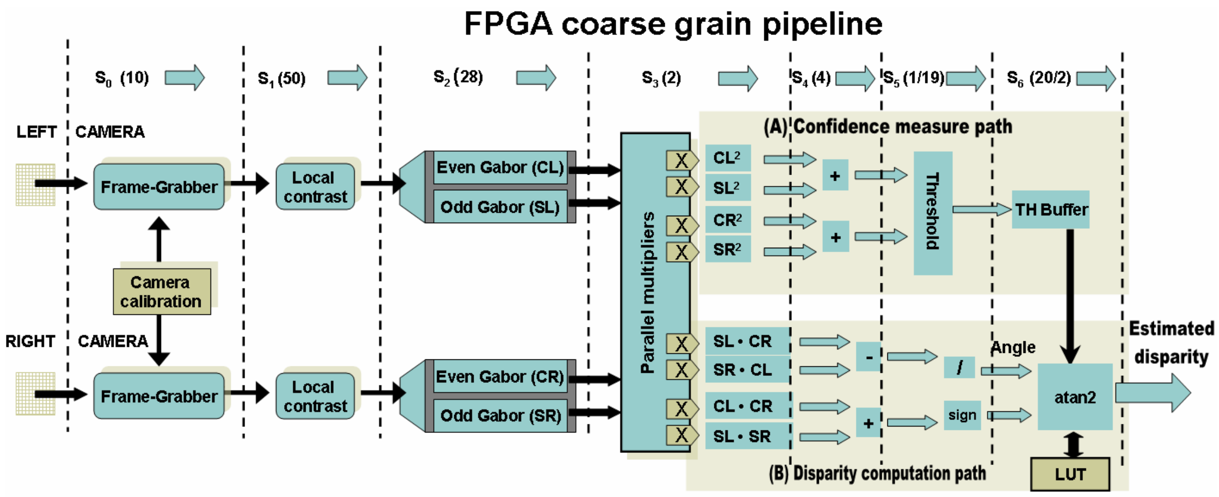


Figure 7.1. Stereo-system hardware architecture. System calibration parameters are stored and used as input shifts (horizontal and vertical) for the camera frame-grabbers of stage S_0. At stage S_1 the local contrast is removed to eliminate even quadrature filter DC response. At S_2 we compute the even and odd Gabor outputs (where ‘S’ stands for the sine or odd Gabor filter and ‘C’ for the cosine or even filter). Note that left and right images have parallel pathways during these stages (high processing performance is enhanced by replicating scalar units). ‘L’ and ‘R’ denote the responses coming from the left and right images. Stages S_3 to S_6 implement the direct phase-difference computation as described in Equation (6). Left and right image responses are combined during these stages into two different datapaths. The upper pathway (A) computes the quadrature filter output energy, which is used as a confidence measure. The bottom pathway (B) computes the phase difference. Note that the efficient use of the intrinsic parallelism available in the FPGAs is achieved by a customized pipeline processing architecture based on well balanced parallel computing blocks at different stages. It allows the computation of one estimation per clock cycle. For this purpose we have designed a micropipelined architecture. In the upper part of the figure we indicate in brackets the number of micropipelined steps at each functional stage; in stages S_5 and S_6, ‘/’ is used to indicate the different number of micropipelined steps of each of their parallel datapaths (A/B).

According to this strategy, the system is configured in 7 functional stages (coarse grain pipelined structure) which are divided into fine-grain pipelined sub-stage data-paths. This leads to a total latency of 115 clock cycles (equal to the number of fine pipeline stages) and a data throughput of one estimation per clock cycle.

The stereo architecture according to this strategy is shown in Figure 7.1. There are two parallel pathways which process each camera image to compute the Gabor-filtered values (implemented as optimized convolution circuits). The level of parallelism at each stage has been expanded to achieve a data throughput of one estimation per clock cycle. The direct phase-difference calculation of equation (4.12) is based on two different paths, (A) and (B) of the circuit. The unit (B) computes the disparity value and the unit (A) measures the confidence estimation (module of the quadrature filters' output). We use this confidence measurement since phase is not clearly defined near module singularities and therefore no reliable information is present at these points [COZ97]. The TH Buffer is a memory buffer used to balance the two processing paths (A and B). The system has been fully implemented in a stand-alone board as a prototype for embedded applications (the RC300 board [CEL06d]). This complete system set-up is shown in Figure 7.2. All the processing operations are computed in the FPGA device as a System-on-a-Chip (SoC), which also contains the camera's frame-grabbers, memory management units, VGA controller and user interface.

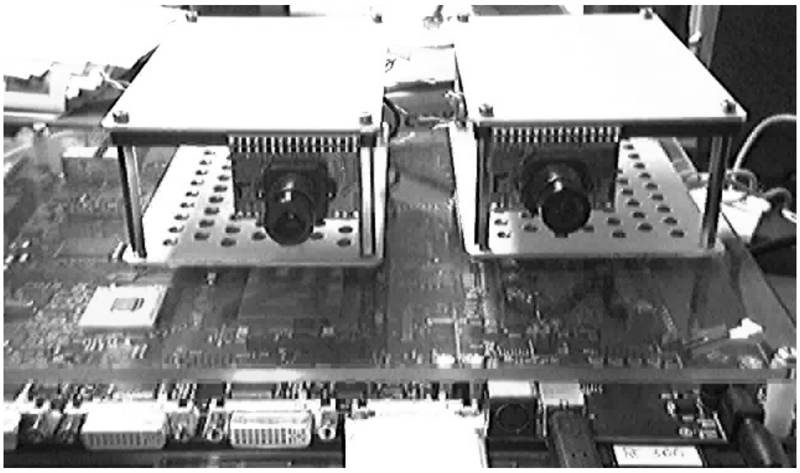


Figure 7.2. Stereo processing system set-up.

7.3. Analysis of System requirements

The full system has been successfully implemented on a Xilinx Virtex-II FPGA [XIL06c]. The system frequency is 65 MHz and due to the regular data-path of the proposed model, we achieve one pixel per clock cycle. This means that we can compute up to 65 megapixels per second (arranged as 52 fps of 1280x960 pixels per image for instance). The consumption of system resources has also been evaluated. The factors to take into account for implementation are those of model degradation due to limited fixed-point bit-width and to the Gabor-filter wavelength (large values improve the disparity range but consume more resources).

7.3.1 Bit-width study

Several decisions have been made about the data representation and bit-width in each pipeline stage based on our utilization of *MCode for DSP Analyzer*. The bit-width of the convolved images with the Gabor filters is critical because its precision affects the following stages in two ways: firstly the bit-width of the next computation grows concomitantly with the square of the number of bits of this stage and secondly, any limitations in precision are transferred to the following stages reducing the system accuracy.

Therefore, in order to optimize the accuracy vs. efficiency trade-off we focus on this stage (S_2). We process a couple real images (shown in Figure 7.4) with a software implementation of the model using double floating-point data precision. We store the disparity values for future evaluations. Then, the filters output is recalculated using signed fix-point arithmetic of different bit-width (from 2 to 32) and we obtain the disparity estimation using this limited precision implementation. In this implementation the RMS error metrics has been used instead SQNR (used in Chapter 5 for local image features) for the sake of clarity (disparity error are easy to evaluate). This show how we benefit from the flexibility of *MCode for DSP Analyzer* that easily adapts to different quantization error metrics.

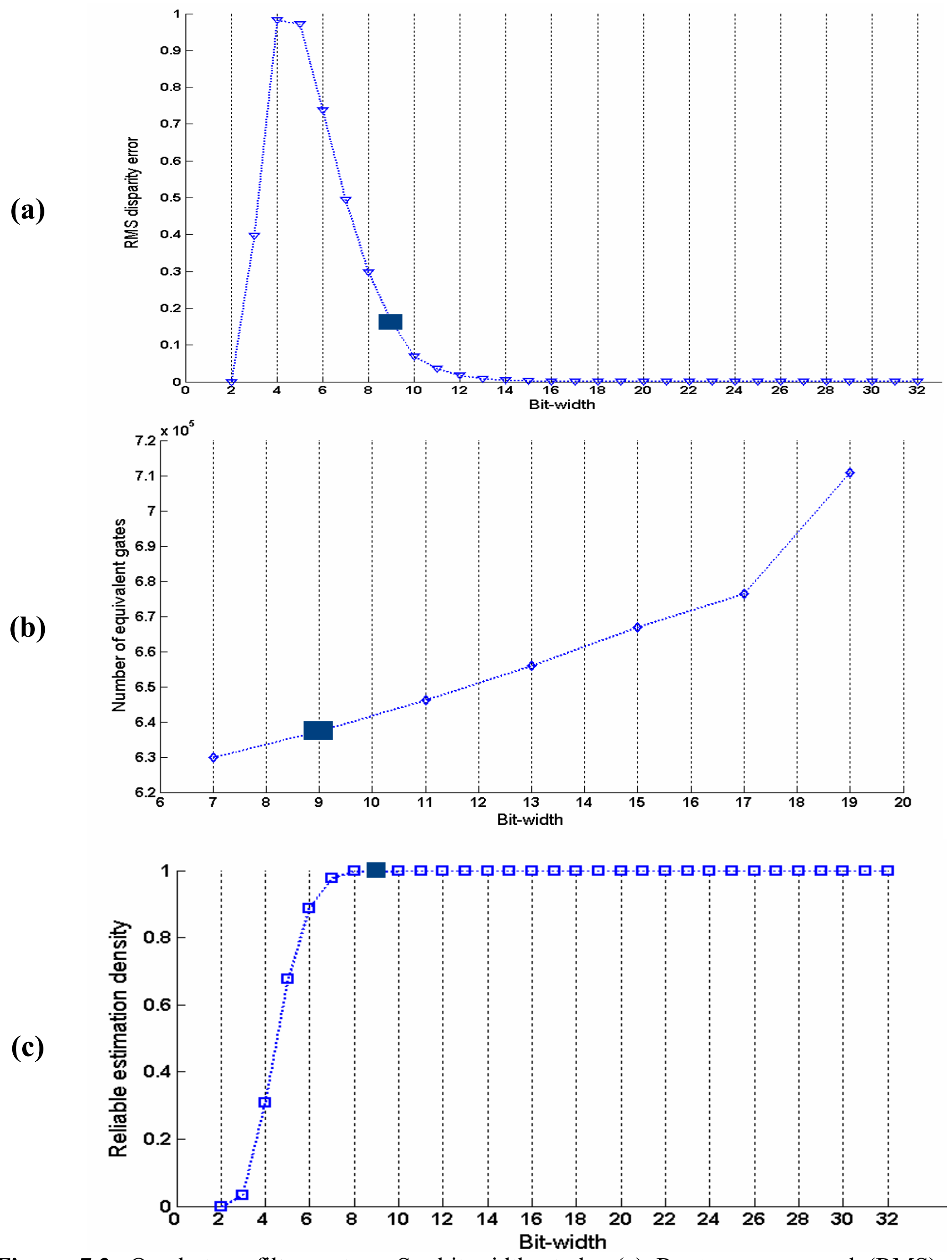


Figure 7.3. Quadrature filters, stage S_2, bit-width study. (a) Root-mean-squared (RMS) disparity error. (b) Number of equivalent gates (hardware cost) vs. bit-width. (c) Number of reliable data (normalized to 1) for the different bit-width choices. The filled squared represent our 9 bits choice which represents a well balanced trade-off between system accuracy, density and hardware cost.

Figure 7.3 represents the study of the bit-width influence in the global system accuracy, reliable density estimations and hardware cost. The RMS error between double floating-point and restricted precision fix-point arithmetic version is represented in Figure 7.3.a. The difference is caused by the data representation bit-width which is fixed independently of image quality. We also calculate a confidence measurement that helps us to filter unreliable estimations. As future work we plan to study if we could use this confidence measurement to adapt dynamically the bit-depth of the data representation. This would require using dynamic reconfiguration.

The hardware cost of the whole system depending on the precision on this stage is illustrated in the 7.3.b. image. These data are extracted synthesizing the whole system width different bit-widths. Finally, using the confidence measure parameters, Figure 7.3.c. shows that low densities are selected for very restricted bit-width. Note that in Figure 7.3.a. the RMS values are very low but this is not important since these points represent very low density values, as it is shown in Figure 7.3.c.

Based on this study, our stages are developed as follows:

- At the convolution stages the processing is done with fixed-point data representation of 9 bits.
- Intermediate data precision is 19 bits using fixed-point arithmetic, avoiding bit wrapping or saturation operations.
- The division operation is implemented using a Xilinx pipelined division core [XIL06b] with 24 bits (19 bits from the above data plus a fractional part of 5 bits for the arctan function).
- The arctan function is implemented using a look-up table of 1024 addresses of 10 bits with 5 fractional bits. Using symmetry, only the $[0, \pi/2]$ interval is sampled. A decision logic based on the input data sign allows us to recover the quadrant of the angle within the full range $[-\pi, \pi]$. This simple scheme allows a maximum estimation error of 0.03 rad for the arctan function with a very simple logic and therefore complex circuits, such as CORDIC [VAL02] for example, are not required.

7.3.2. Hardware resources consumption

Table 7.1 shows the required resources for the whole system with the choice described above (Section 7.3.1). The first row indicates the calibration-system circuit resources and the following rows show the consumption for each Gabor scale, which grows for longer scales. We consider these bit widths as good trade-offs between the system accuracy and hardware resource requirements. Note that in Figure 7.3.b, for bit-widths higher than 18 there is a clear increment in the FPGA resource usage. This is because the width of the FPGA internal multiplier is exceeded.

Each design is characterized by the Megapixels per second and is completely modular. Therefore we can choose different resolution versus frames per second trade-offs. The 65 Mpps of performance also has been used for population coding simulation, using the system as coprocessing unit based on time-slicing techniques, see [DIA06j] for details.

Besides the calibration stage, the FPGA reconfigurability also allows different image scales computation. Provided that stereo techniques work better for small disparities, we have designed three different scales, with Gabors filters of 15, 31 and 55 taps. In that way, depending on the image structure, our FPGA can be reconfigured for different scales to estimate the range of disparities that match better the image structure. It is important to note that larger filters work as low pass filters and high frequency image structures are lost; therefore, Gabor filters must be tuned to the desired application to get the best results.

Table 7. 1. System resources required on a Virtex II XC2V6000-4. First row: simple camera calibration system. Following rows: phase-based stereo device using several Gabor scales. (Mpps: mega-pixels per second at its maximum system processing clock frequency.) EMBs stands for embedded memory blocks.

Slices / (%)	EMBs / (%)	Embedded multipliers / (%)	Mpps	Gabor spatial scale (filter length)	Image Resolution	Fps
2864 / (8%)	1 / (1%)	0 / 0	70	-	640x480	56
6411 / (18%)	15 / (10%)	21 / (14 %)	65	15	640x480 1280x960	211 52
9197 / (27 %)	39 / (27%)	31 / (21 %)		31		
13048 / (38%)	71 / (49%)	59 / (49 %)		55		

7.4. Performance evaluation

Figure 7.4 shows the disparity estimation for a couple of real binocular image pairs. Gray levels encode depth information (lighter levels indicate closer objects). Software (with double floating-point representation data) and hardware (with limited fixed-point data representation) approaches are compared. Qualitatively, the degradation is very low.

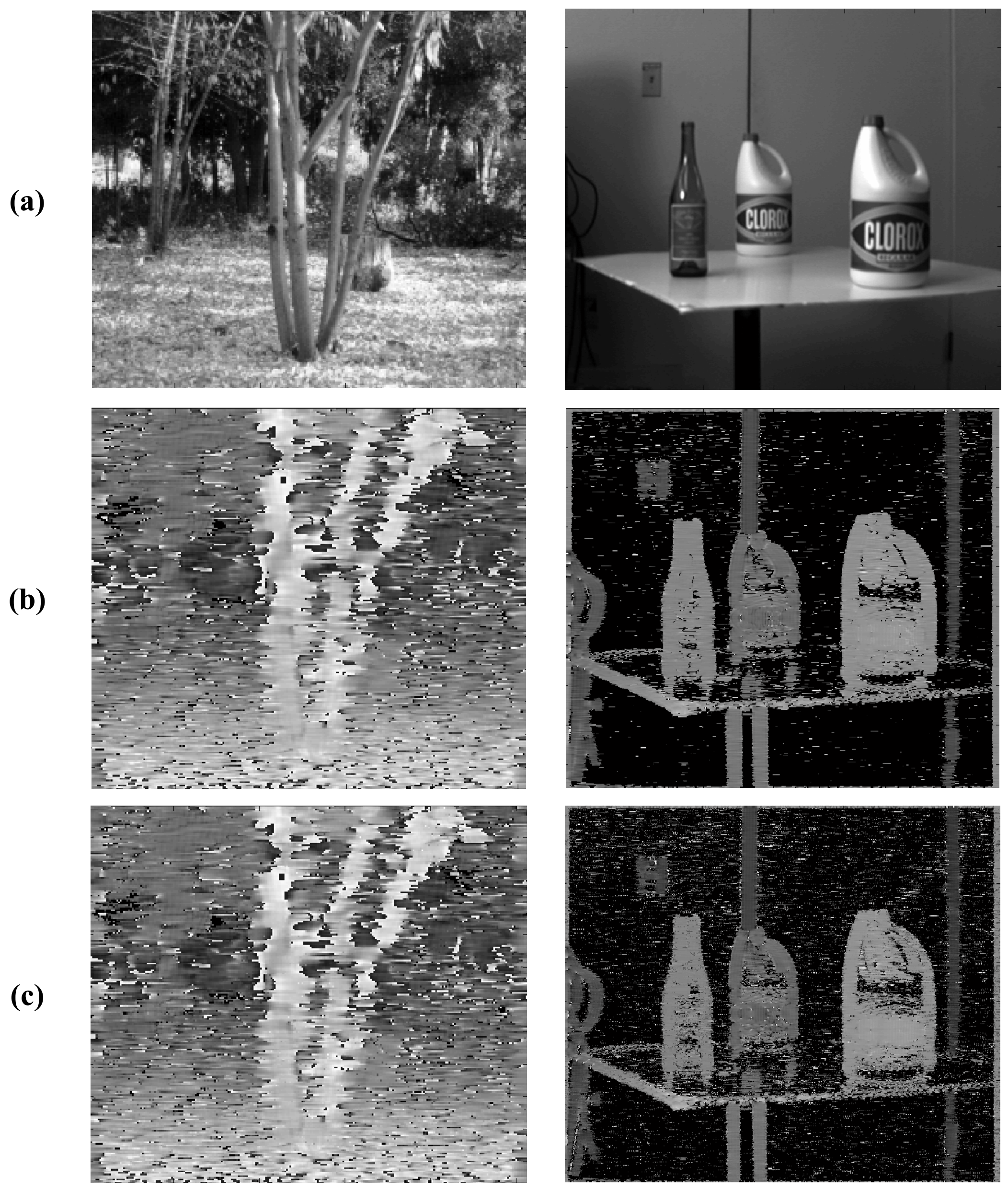


Figure 7.4. Software vs. hardware implementation: (a) original images, (b) software stereo processing, and (c) hardware stereo processing. The image on the left was processed using a small scale (Gabor filters with a length of 15 pixels) and that on the right with a medium scale (Gabor filters with a length of 31 pixels). Note that only small differences are visible as an increase in salt-and-pepper noise (more visible in the right-hand image) in the hardware results due to the restricted precision available in the hardware implementation.

For a quantitative study, we measure the quantization error of the images in Figure 7.4, which have significantly reduced bit-width. We measure the mean disparity error for the left and right images obtaining values in the interval [0.05, 0.06] pixels. This represents a negligible noise contribution compared to the error coming from the stereo estimation model itself which hardly achieves a precision higher than 0.1 pixels on arbitrary scenes [BRO03]. We conclude from these measurements that the system quantization degradation versus the hardware resources consumption (table 7.1) trade-off is appropriate. For more details on the bit cutting procedure see previous Section and [DIA05a], [DIA05b].

Given that stereo techniques work better for small disparities, we have designed three different scales, using quadrature filters with a length of 15, 31 and 55 pixels. The Gabor-filter wavelength determines system quality according to image resolution and disparity range. The disparity range for each circuit is +/- 4 pixels, +/-8 and +/- 14 pixels respectively [COZ97]. It is important to note that larger filters work as low-pass filters and high-frequency image structures are lost; therefore, although the first stage of camera calibration reduces overall image displacement, the Gabor filters must be tuned to the desired application to get the best results.

According to this strategy, the implementations presented here, which only compute one scale, must be scale-tuned according to the application addressed and image structure required. This is done by the user (or agent), who can reconfigure the circuit (i.e. the FPGA device can be reprogrammed in less than 400 ms) from the system interface or PC command line to change the disparity scale. Future studies will try to combine different scales dynamically based on the image structure without the need for intervention by the user.

The evaluation of the system performance should consider the image resolution, frames per second and number of cameras. It is also important to consider the searching area where the two images are compared (small searching areas or filter lengths require less computing performance than larger areas). According to that, we use a common comparison metric of stereo-vision systems [DAR03], [DAR06] in order to rank the system, the performance is given by measuring the number of disparities computed per second. This is the PDS (Point-time Disparity per Second), measured as: $PDS=N \bullet D \bullet (C-1)$, where N is the number of pixels processed per second, D the number of disparity values estimated (the disparity range) and we also include C which is the number of cameras to extend the metric to multi-baseline stereo approaches. Please note that this metric only measures the system performance and not the architecture complexity which is based on different factors such as the disparity estimation model and the computing platform.

Using this metric our binocular system achieves different performances according to the Gabor scale and consequently shows different system-resource consumptions. Expression (8.1) calculates the performance of the system in terms of the PDS of the different configurations based on scales of 15, 31 and 55 Gabor-filter lengths. The disparity range for each circuit is +/- 4 pixels, +/-8 and +/- 14 pixels, which gives us equivalent D values of 9, 15, and 29 respectively. Thus, the corresponding PDS values are as follows:

$$PDS = N \cdot D \cdot (C-1) = N \cdot f_{clk} \cdot 1 = \begin{cases} 9 \cdot 65 = 585M \\ 15 \cdot 65 = 975M \\ 29 \cdot 65 = 1885M \end{cases} \tag{8.1}$$

A comparative performance study is shown in Table 7.2. The large differences between architectures (standard processors, custom hardware, Graphical cards and FPGAs) makes difficult a direct comparison but it is still worthwhile to illustrate how well each approach fits the stereo computation task. We use the system raw performance in PDS as performance metric unit.

There are recent Software based approaches but most of the processing platforms are based on FPGAs [DRA03] and they use different FPGA families with different performance that bias the comparison due to technology advances. Because of that we are not taking into account the resources consumption. In FPGA based approaches we try to reduce the bias due to technology advances by normalizing the PDS performance by the clock frequency as shown on Table 7.2. The performance obtained by our system is faster than the earlier block-matching-based binocular implementations commented in [BRO03] (whose fastest version is included in the sixth row of Table 7.2).

In terms of PDS our system outperforms the fastest non-comercial implementation in Table 7.2 (more concretely, the approach of Niitsuma and Maruyama [NII04], by a factor between 2.3 and 7.5, depending upon the chosen Gabor scale). Similar outperforms are achieved using the normalized PDS. Note that this system [NII04] uses the same FPGA technology and fast clock frequency but our outstanding performance is based most significantly on the optimized computing architecture described in this chapter. This becomes clearer when we compare the normalized PDS to evaluate the computing parallelism of the different approaches. Our system achieves far higher normalized PDS through the intensive use of the parallel processing resources available at the FPGA device (mainly due to the fine pipeline architecture).

Table 7. 2. Performance comparison of selected real-time binocular systems. We have only included systems with performance information available in the literature. In the systems based on FPGAs we also include the normalized performance measures PDS/fclk in order to facilitate the evaluation of the efficient use of the parallel resources available at these devices.

Real-time system	Image resolution	fps	Disparity range	PDS x10^6 / [PDSx10^6/f_{clk}]	Method	Processor type
Proposed here	*1280x960*	*52*	*9* *15* *29*	*585 / 9* *975 / 15* *1885 /29*	*Phase based*	*Custom FPGA, Xilinx Virtex-II (65 MHz)*
Gong and Yang [GON05] (2005)	512x384	14.7	40	117 / --	Correlation-based with image-gradient-guided cost aggregation	Pentium 4 3GHz equipped with an ATI 9800 XT (412 MHz)
Forstmann et al. [FOR04] (2004)	256x256 640x480 1024x1024	30.4 7.23 2.2	100	200 / -- 222 / -- 230 / --	Dynamic programming	AMD AthlonXP 2800+ and MMX optimization
Niitsuma and Maruyama [NII04] (2004)	640x480	30	27	248.8 / 3,66	Correlation. SAD	Custom FPGA, Xilinx Virtex-II (68 MHz)
Darabiha et al. [DAR06] (2006)	360x256	30	20	55.3 / 1,1	Correlation phase-based	Custom FPGA, Xilinx Virtex, (50MHz)
Woodfill, and Herzen [WOO97] (1997)	320x240	42	24	77.4 / 2,35	Census matching	Custom FPGA Xilinx XC4000 (33MHz)
Focus Robotics [FOC06]	752x480	30	94	1017/ --	Correlation. SAD (9x9)	Xilinx Spartan3
Videre Design [VID06]	640x480	30	64	589 / --	Correlation. SAD (15x15)	Xilinx Spartan3
T. Kanade et. al. [KAN94] (1994)	256x240	24,4	20	30 / --	Multi-baseline Correlation. SSAD	Custom HW & C40 DSP (2-6 cameras)

Table 7.2 also includes approaches with software implementations using graphic cards and MMX extensions [FOR04], [GON05]. Despite the computing performance of such systems being quite high, they consume all the resources of the computer, rendering it impossible to compute higher level algorithms based on stereo in real-time. Furthermore, it is difficult to use these approaches on mobile platforms for embedded applications such as robotics or smart sensors.

We have also included some recent commercial products based on FPGAs processors [FOC06], [VID06]. These are very good engineering solutions from companies which use well-known block-matching technique with SAD as similarity function. Despite their high performance, especially for the approach of [FOC06] with performance similar to our design, their main limitation coming from the chosen approach. As commented in [SCH02], they have regular disparity estimation quality. Furthermore, correlation based approaches suffers from radiometric problems and present a high dependence on light conditions and other environment options which make these systems a valuable solution but only for specific applications.

Finally the last row of Table 7.2 shows a very interesting approach [KAN94]. In this case the comparison with our system is not fair because they build a full custom system which can not be easily updated (while our approach easily takes advantage of the continuous technology advances). Furthermore this system has very high power consumption while in ours everything is built on the same chip that significantly reduces the required power.

There are also commercial products such as Bumblebee and Digiclops® from Point Grey Research [POI06]. These devices consist of calibrated stereo cameras plus software libraries to compute the stereo. We have not included these devices in the comparison in Table 7.2 because the information about processing performance on standard computing platforms is obsolete.

7.5. Improved system based on population coding

The main limitation of the previous system is the limited range of disparities available due to the linear approximation of the phase model. Theoretically this is $\lambda/2$ (being $\lambda=2\pi/k0$ the period of the tuning frequency of the Gabor filter) but experimentally is about $\lambda/3$ (for details see [COZ97]). Usually the solution found in the literature consists of a coarse-to-fine approach, using confidence values from coarse scales to warp the image at fine scales. The problem of such an approach is that wrong estimations propagate from coarse to fine scales.

Contrary to this approach, a parallel processing of spatial scales with a fusion integration stage is more biologically plausible. In a similar way to [FLE94], the scales are processed in parallel and integrated using a similarity measure. Shift neurons could also be added ([FLE94], [POR02]) to improve the disparity range using neurons with overlapping disparity tunings. Contrary to Fleet's approach ([FLE94]), which uses Gabor filter correlation and sub-pixel estimations by linear interpolation, we propose a scheme which uses Sum of Absolute Differences (SAD) over the energy of the shifted cells (which is more hardware-friendly because it avoids square roots and division operations) [DIA06j]. At this stage, the cell with the lowest response encodes the winner shift value which achieves the best disparity tuning. Phase difference for sub-pixel estimation (instead of linear interpolation methods) is used to obtain sub-pixel disparities values. The shift offset obtained with SAD, is calculated with the value obtained from the basic model providing the improved subpixel disparity estimation. This is schematically shown in Figure 7.5.

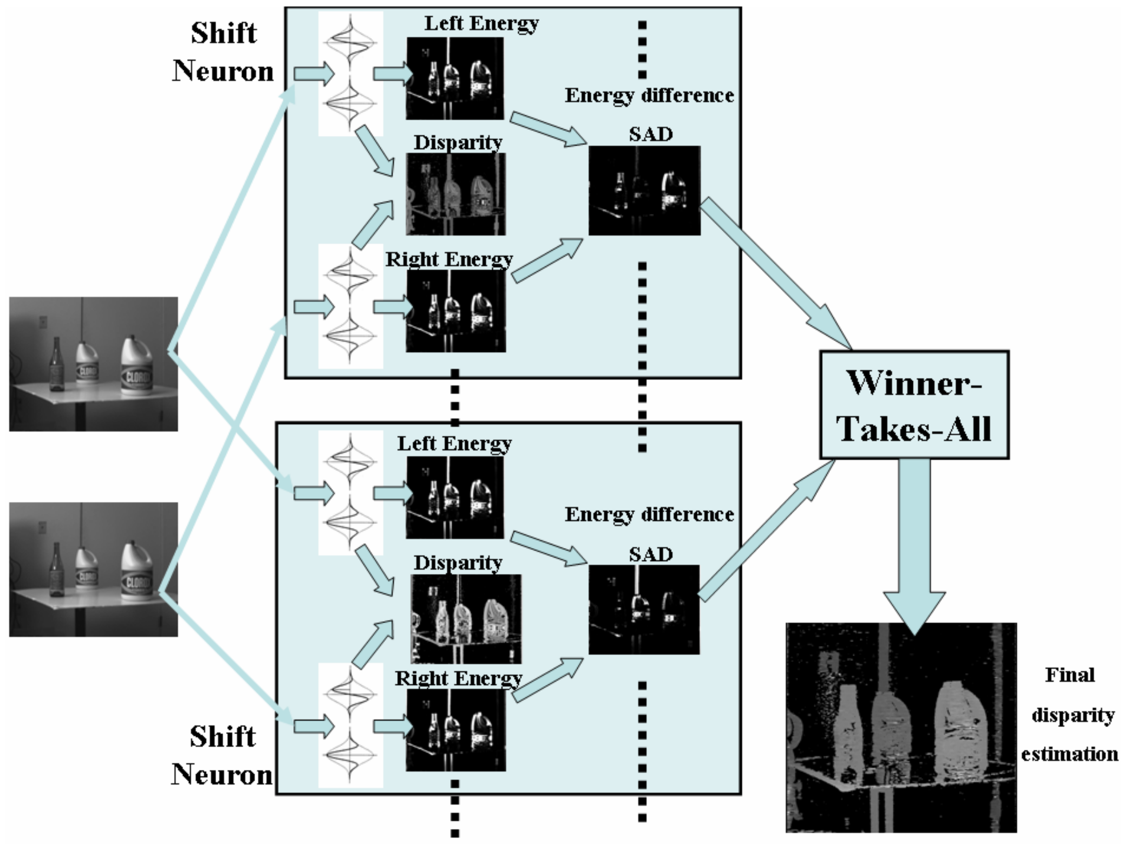


Figure 7.5. Population coding for binocular disparity estimation based on shifted neurons and SAD similarity measure.

Qualitative results for this model are shown in Figure 7.6. Note that the disparity range and resolution are improved, obtaining smooth variation and disparity details.

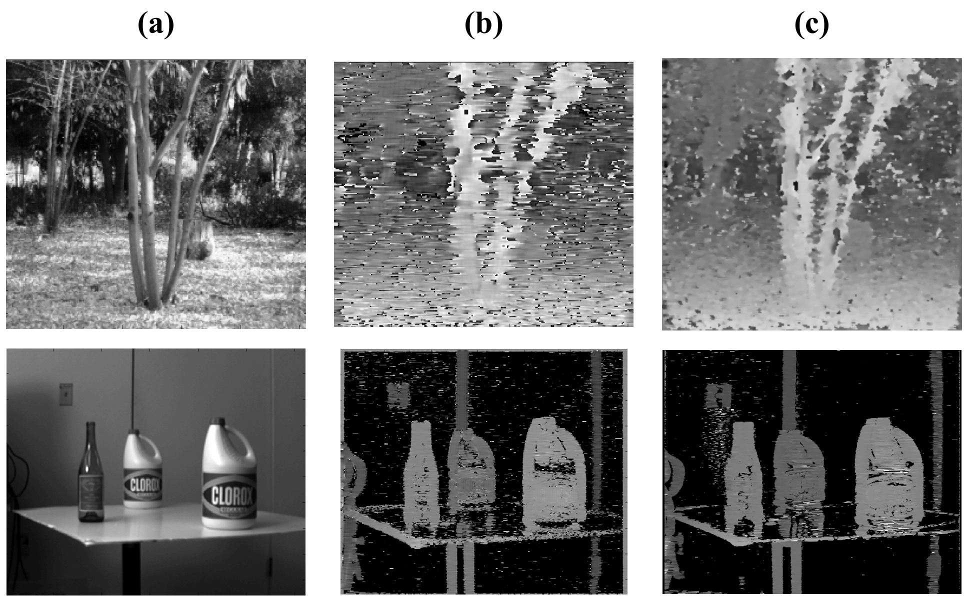


Figure 7.6. Basic vs. modified stereo model. (a) Original images, (b) Original stereo model described in Section 4.4. (c) Results using the multiple estimation based model described on this section. The disparity is encoded in grey levels, light pixels indicate short distances.

The processing speed of our system using a customized frame-grabber allows us to test several population types and fusion methods in real time, using for example the basic system described in Sections 7.2 as a coprocessing board. For example, we can process each image pair 8 times, using 3 spatial scales and a shifted distribution of 5 neurons with overlapping disparity tuning to increase the available range of disparities obtaining an equivalent circuit running up to 26 fps of image sizes of 640x480 pixels using approximately the same system resources (memory resources demand is increased). Shift neuron just implies offset values in the frame-grabber of one of the cameras and the different scales imply just changing the Gabor filter coefficients. Therefore, we use the same primitives described in Section 7.2. Furthermore, the outstanding processing speed achieved by our approach allows us to use the same circuits to process the images repetitively (with different shifts and filter scales) storing the results to be integrated in a simple winner-takes-all stage. In this fusion module we just take at each pixel the disparity value (among candidates) with the highest confidence value.

7.6. Conclusion

We present a bio-inspired model implemented onto programmable hardware that runs on a stand-alone chip for embedded applications. The pipeline processing structure, including some well balanced parallel processing modules, efficiently computes phase-based disparity estimations. The most important contribution of this chapter is the efficient implementation of a vision model on specific circuits adopting a well structured design strategy, as described in Section 7.2. The regular data-path is able to compute one pixel per system clock cycle. This efficient use of the parallel computing resources available on FPGAs plus a fine-grain pipeline design lead to an outstanding processing speed (65 Megapixels per second, which can be arranged as 52 fps of 1280x960 pixels per image).

The system includes an automatic pre-calibration stage to improve the system disparity range as well as the possibility of switching between spatial filters scales according to the application addressed and image structure in the target scenario. We have measured the system degradation due to bit-width restrictions and decided upon a good trade-off between degradation and resource consumption.

In the future we plan to study the implementation of a multiple-scale stereo system that takes advantage of the designed architecture and combines the different scales according to the image structure presented in the neighborhood of each image point. We are also planning to address the integration of population schemes as showed in Section 7.5 in order to analyze plausible biological models for stereo vision computation.

Chapter 8
Lane change decision aid system based on motion-driven car tracking

This chapter proposes a lane-change decision aid system for monitoring vehicle overtaking scenarios. The system is based on the real-time optic flow systems described in the Chapter 6. We describe the system and evaluate its performance with a benchmark database of real overtaking sequences taken using instrumented cars which makes possible to know parameters such as the detection distance at which the vehicle that is approaching is reliably tracked.

8.1. Introduction

The work presented this chapter was carried out within the framework of the ECOVISION Project [ECO06]. One of the objectives of the ECOVISION consortium was the development of pre-cognitive visual models for real-world environments. In particular, a rear-view-mirror blind-spot monitor and a driver-distraction alert system are presented as a feasible problem in which motion processing can provide useful cues for the motion pattern based segmentation of an overtaking car.

Current techniques concerning car tracking usually focus on road-traffic monitoring. The use of video cameras with computer vision techniques offers an attractive alternative to current sensors due to their ability to measure a greater variety of traffic parameters (e.g. entry/exit statistics, journey times and incident detection) [ATE06], [SET01]. Although the use of an image-processing system in a car is not straightforward since it requires an embedded processor capable of computing images from moving cameras in real-time, blind spot and driver distractions are such important sources of accidents that the European Commission is studying specific actions to eliminate the blind spot on motor vehicles [DIR03]. A lane-change assistant should recognize vehicles in the blind spot and warn the driver if he starts changing lane. A standardization committee has been formed on the subject of a Lane-Change Decision-Aid System (*LCDAS*). To evaluate this system a preliminary draft of the ISO standard is applied here (ISO / TC204 / WG14 / N40.27).

Over the last few years driver-assistance systems have become a priority for car manufacturers. Nowadays, on-board image processing platforms and cameras are more in demand for help in lane keeping and the detection of impending collisions from fast-approaching or lane-changing vehicles [HIT06], [MOB06], even to the extent of including stereo cameras [RCM06] or radar-stereo fusion [BRO05] to estimate the distance to collision. Some companies, such as Mobileye N.V. [MOB06], Volvo [VOL06], and Fico S.A. [FIC06], have apparently developed some aids to lane-change decision making but no reports on their technical details or the performance of these approaches have as yet been published. Their initiatives only cover the application itself but with no benchmarking information to validate their systems, making it impossible to compare the different approaches. There is also a product based on radar sensors [HEL06a] to solve the same problem but again it lacks any validation information.

Furthermore, the impossibility of acquiring such sensors indicates that they are still at a predevelopment stage, with some problems still to be solved.

In our approach we use a monocular camera within the car that allows us to detect the overtaking vehicle by using optical-flow algorithms. This system can be used to generate alert signals to the driver. The optical-flow-driven scheme has several properties that can be very useful for car segmentation. Basically, by focusing on the optical-flow field we should find static objects and landmarks moving backwards (due to our ego-motion) and the overtaking cars moving forward towards our vehicle. Nevertheless, there are several artefacts such as perspective deformation and camera vibration that can affect the performance of the system. The proposed scheme needs to address these kinds of artefacts.

The application involves significant challenges. Most of the contributions developed for traffic analysis work with static cameras [SET01], [HSU04], [DES05]. On-board cameras increase the complexity of the system considerably, partly because the algorithm needs to deal with non-static scenarios (which means complex algorithms to analyse the scene), and partly because the processing frame rate becomes a critical factor for such analysis. On-board cameras have been used for lane tracking [APO04], [MCC06], and also in front/back vision for obstacle avoidance [RCM06], [DAG04], but the application we present here focuses on a different field of view, the rear-view mirror. It is important to emphasise that we have to deal with such important factors as perspective deformation [MOT04a], [MOT04b] and in order to perform satisfactorily the proposed system needs to overcome this problem.

One important implementation issue concerns the co-design strategy, i.e. deciding the software/hardware code partitioning, which will have an important impact on the final flexibility of the system and its cost. The working scheme that we have adopted is composed of two very different stages. In the first step we have customised a FPGA device (to be used with embedded systems) for real-time motion processing [DIA04b], [DIA04c]. The chosen optical-flow scheme uses a gradient model based on the classical approach of Lucas & Kanade [LUC81], [BAR94]. As mentioned in [DIA04c], this model achieves satisfactory optical-flow accuracy using affordable hardware resources. We have used a high-level hardware description language (Handel-C, see [CEL06c]), which allows us to describe hardware using high-level (C-like) algorithmic structures. This makes it easy to decide the critical code to be implemented on a customised DSP. In the second step, based on the previous motion-salience map, we combine Kalman filtering techniques with appropriate filtering operations to compensate the effects of perspective deformations in order to arrive at a reliable estimation of a car's position in the scenario.

8.2. Car Tracking

What is the aim of the system? The system has to warn the driver of impending critical situations during a lane change. Critical situations occur in different possible scenarios:

1. A vehicle is beside the lane-changing car in the so-called blind spot and the driver does not realise that his lane change would cause a critical situation.
2. A car is coming up at relatively high speed, which would also result in a dangerous situation if the driver were to change lane.
3. The absent-minded driver begins to change lane without noticing that an overtaking car is approaching.

Within this context, we are only interested in the approaching car closest to us, so we begin the local searching within the right-hand area of the image (Figure 8.1)[1]. We are interested in detecting the car as soon as possible and not losing track of it, especially when the car is close to us. Furthermore, the proposed application needs to direct alert signals at the driver to prevent an accident. Therefore we estimate the car's position and the confidence level. This facilitates the generation of the alert signal.

8.2.1. Pattern selection and optical-flow filtering templates

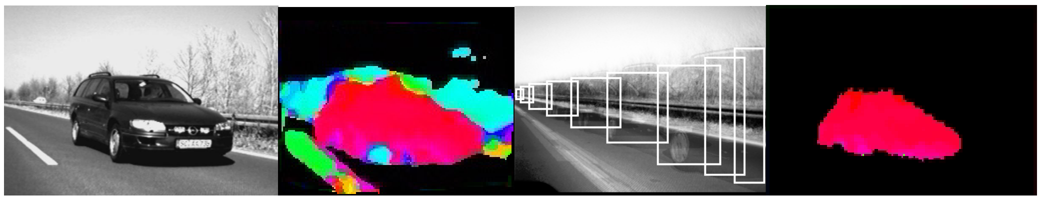
Optical flow is a well known method used for motion-based segmentation [WEB97], [YAN03] and according to our previous results [MOT05] we have validated this approach for the on-board segmentation of overtaking vehicles. In our system some simplifications can be made because of the structure of the problem addressed. We consider only rightward movements. During overtaking manoeuvres the approaching car is moving to the right-hand side of the rear-view image so we do not need to

[1] For the sake of clarity we only consider right-hand driving with the steering wheel on the left and left-hand overtaking.

consider leftward velocities (Figure 8.1). Wrong velocity estimations of the optical flow are frequent; therefore we need to clean up these erroneous patterns in the next steps. If V_x is the x component of the velocity, V_y is the y component and k the minimum reliable velocity component module, the velocity set that we use should verify $V_x>k$ and $|V_y|<V_x$. This allows us to consider only rightward motion and take into consideration the focus of expansion in the rear-view mirror, which apparently produces both vertical and horizontal patterns for the moving objects.

The proposed system uses templates that filter the motion-saliency map. We use them to clean up the optical flow of the previous stage and maintain only the more reliable data to compute the position of the overtaking vehicle. This scheme fits quite well into specific hardware because the operation can be implemented as convolvers.

An object will pass to the next stage if it has enough active points in the neighbourhood circumscribed by the templates. The template forms are rectangles that grow along the x axis towards the right-hand side of the image, where the vehicle is expected to be larger (cf. Figure 8.1). Each spatial position has an associated template that establishes the minimum number of points and the neighbourhood area to carry out the search. Pixels without enough active neighbours are neglected. The values of active neighbours are experimentally determined using the overtaking-car sequence database provided by Hella KGaA Hueck & Co. [HEL06b]. A typical value of the threshold of active points related to each template is 50% of all the points inside the template.



(a) **(b)** **(c)** **(d)**

Figure 8.1. Vehicle segmentation example. (a) Overtaking car sequence. (b) Original optical-flow saliency map (colors indicate velocity orientations: red encodes rightward motion and blue-green pixels encode leftward motion). (c) Templates for filtering the optical flow based on rectangles to estimate the car's position. (d) Final result (segmented car) after applying the filtering.

Centroid computation of the saliency map gives us an estimation of the car's position but this is correct only for continuous overtaking. Some more complex and realistic situations need to be solved:

1. Static overtaking: An overtaking car seems to stop (and its optical flow vanishes) because it maintains the same velocity as the car being overtaken. In this situation we need to maintain the car's estimated position for a certain time.
2. Multiple car overtaking. This is a very common situation on highways and one that we need to solve.

Another subject to address is the minimum number of overall valid pixels that gives us reliable car-position estimations. When there are no cars in the sequence, or they maintain the same speed as our own car, no valid data should appear in the saliency map. We use a confidence threshold for the remaining data to keep only reliable features active. The threshold is dynamically adapted according to the system's recent record, using a threshold function which decreases linearly with time in the absence of inputs (enhancing the system sensibility) and increases when a high number of inputs are presented (improving the system's reliability). Besides, this threshold also varies according to the car's estimated position using our *a priori* knowledge about the mirror perspective deformation. Higher thresholds are used in the right-hand area of the image, where candidate cars are expected to be larger. Furthermore, the motion extracted in this area is noisier because speeds are higher. A study concerning perspective deformation and the different techniques available to minimise this deformation can be found in [MOT04a].

8.2.2. Solution for static overtaking. Kalman filtering

We need to use a memory system to retain the vehicle position when it remains stationary relative to our car. Traditionally Kalman filtering has proved to be satisfactorily in resolving many problems involved in predicting the position of moving targets [DEL97], [GAO05] and is even useful for complex motion prediction [JUN97]. It is also advisable because of the inherent latency of the system's processing. Although the proposed platform can compute 25 fps, the optical-flow processing unit has a latency of 3 frames. This means that the estimated position of the car undergoes a short delay with respect to its real position. This is not a problem for low relative velocities but when the velocity is high it might result in the system's underestimating the car's position. The capability of Kalman filtering to predict position allows us to overcome the artefact produced by this inherent processing latency, thus increasing the system's reliable detection distance.

As far as hardware feasibility is concerned, we have used simple Kalman filter equations that basically act as a short-term memory system with prediction capability. The process model we use is described by equation (8.1), where σ_Q^2 is a model parameter.

$$s_{k+1} = \phi s_k + \xi_k \quad s_k = \begin{pmatrix} x_k \\ y_k \\ x'_{k+1} \\ y'_{k+1} \\ v_k^x \\ v_k^y \end{pmatrix} \quad \phi = \begin{pmatrix} g_m & 0 & (1-g_m) & 0 & g_m & 0 \\ 0 & g_m & 0 & (1-g_m) & 0 & g_m \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \qquad Q_k = \sigma_Q^2 \begin{pmatrix} 1000 \\ 0100 \\ 0010 \\ 0001 \end{pmatrix} \tag{8.1}$$

The system state depends on the position and velocity estimations at the previous instants using the memory gain parameter g_m. This parameter is a constraint which implies smooth velocities and which, for this application, can be $g_m(x)$. The variables x'_{k+1}, y'_{k+1}, v^x_k, v^y_k are measured using an iterative centroid computation as described in the section below. The vector ξ represents a random Gaussian white vector of zero mean that models the additive noise and has a diagonal covariance matrix, Q_k, which is also defined in equation (8.1). This model makes the assumption that the velocity is constant and that the noise can be seen as an acceleration of the object. For the measurement model we use equation (8.2).

$$z_k = Hs_k + \mu_k \quad H = \begin{pmatrix} 100000 \\ 010000 \end{pmatrix} \qquad \mu_k = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} \tag{8.2}$$

Vector μ, as ξ in the previous equations, represents a random Gaussian white vector of zero mean that models the additive noise and has a diagonal covariance matrix R_k with σ_R^2 as model parameter.

8.2.3. Solution for multiple car overtaking: iterative process

For our application, a multi-target tracking system is unnecessary. We only need to know whether there is at least one car in a relevant situation. We use an iterative computation with several steps to compute the car's position. In the first step we use all the saliency-map points of the whole image to give the estimated position of the car, which will be the correct position if there is only one car. When there are several targets in the system, however, the main goal is to detect the position of the car closest to us.

Therefore, we focus on the right-hand area of the image, using the computed centroid position as the left-hand image boundary. We try to calculate a centroid of this restricted area in the image if we have significant features; otherwise we take the previously calculated value. We can repeat this computation several times until the estimation converges or we can use a limited number of iterations. For our system we have used only three iterations to get adequate results. A qualitative example with some frames of an overtaking sequence is shown in Figure 8.2. The car labelled (1) is tracked (frame A); once it passes, the vehicle estimation searches for a new vehicle (frame B). The car labelled (2) is found and tracked (frames C and D); when this vehicle has overtaken the system looks for the next car and finds the one labelled (3) (frame E).

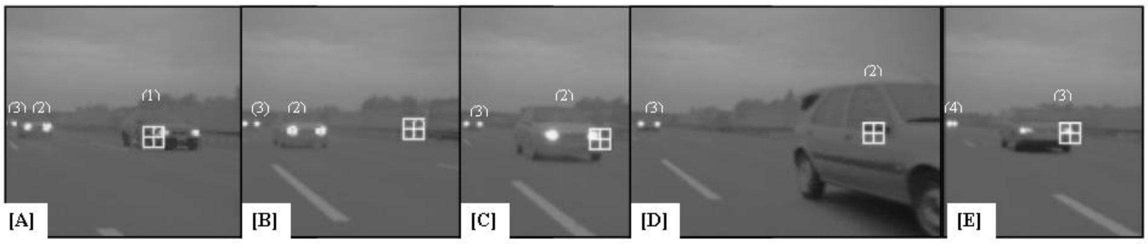


Figure 8.2. Multiple cars overtaking on a highway on a cloudy day. Number labels on the top of the cars are added to clarify the process. The reliable position is marked automatically by the system in the figures using a white cross.

8.2.4. Confidence measure estimator discussion

After the optical-flow filtering step, the resulting image contains only reliable points for the centroid computation. If there are only a few points remaining, no reliable information can be obtained and no estimation can be arrived at. The number of confidence points (*NCP*) also varies with the evolution of the system. Closer cars appear larger in the image due to perspective so *NCP* must increase with the estimated car position moving rightwards. Nevertheless, with a small confidence threshold (*CTH*) we can effectively get a stable vehicle position and a stability signal that indicates the confidence of our system on the acquired data. Sometimes the optical flow is very noisy (on a bumpy road for example) and, despite the number of thresholds imposed to compensate them, some mistakes appear (such as the isolated points shown in Figure 8.3). These spurious estimations should not be allowed to trigger the alarm signal or they would compromise the driver's confidence in the monitoring system. Although from an engineering point of view the number of errors presented as isolated dots in Figure 8.3 is not significant (less than 2%), from a psychological point of view, false positive alarms significantly affect the driver's confidence. Therefore our alarm system only triggers in the shadowed areas in Figure 8.3. The system can also benefit from methods for monitoring the driver vigilance (such as eyetrackers [BER06]) whose information can be efficiently used to inhibit or increase the alarm signal generation. This is important to reduce the number of unnecessary alarms which represents an interesting topic to be explored in future works.

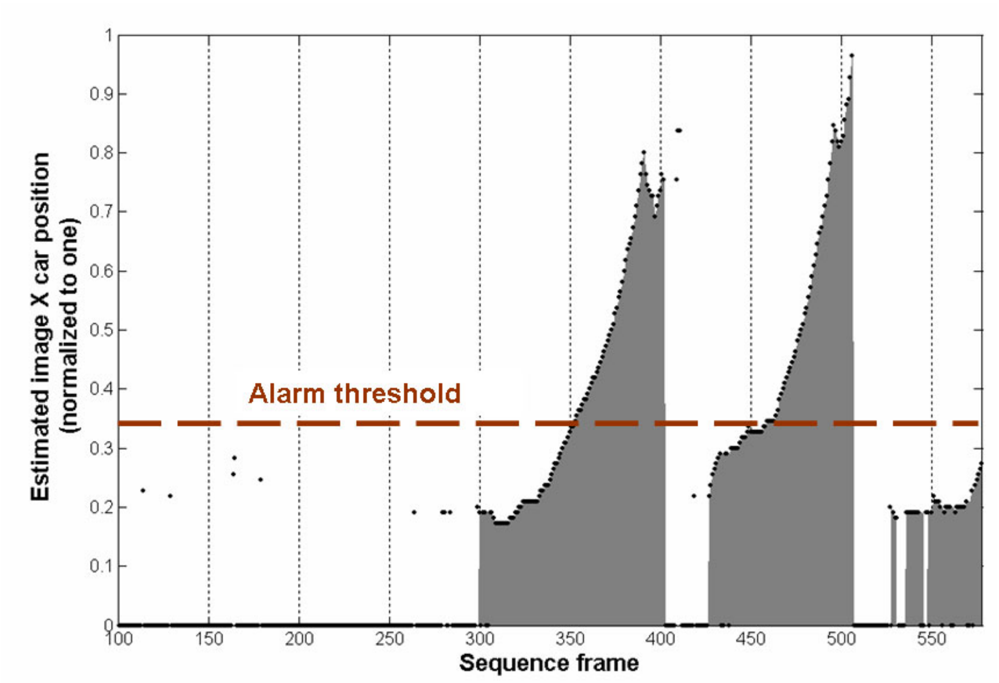


Figure 8.3. Example of confidence car-tracking estimators for an overtaking sequence of three cars. Target vehicle closing speed is about 15 m/s. Black dots represent the estimation of the car's position using centroid computation from optical-flow data. There are still some isolated errors. Filled areas represent high confidence frames in which the car's position is estimated reliably (i.e., exceeding the confidence thresholds) and isolated points represent unreliable data. The horizontal line represents the vehicle-position threshold to trigger the alarm signal.

To solve this situation a very simple hardware-friendly scheme is adopted. Using a temporal memory window of 7 frames and median filtering of the stability signal we finally achieve high confidence, high stability and high reliability in deciding when we are in critical situations. The temporal persistence of the stimulus allows us to reject noisy inputs and thanks to the large detection distance (provided by subpixel optical-flow configuration) we can utilise this median filtering without loss of performance.

The final system output is represented by the grey areas in Figure 8.3, corresponding to three different overtaking cars. These areas represent the alarm signal which a driver will see if he tries to steer towards the overtaking car lane.

The scheme shown in Figure 8.4 summarizes all the computing stages described above.

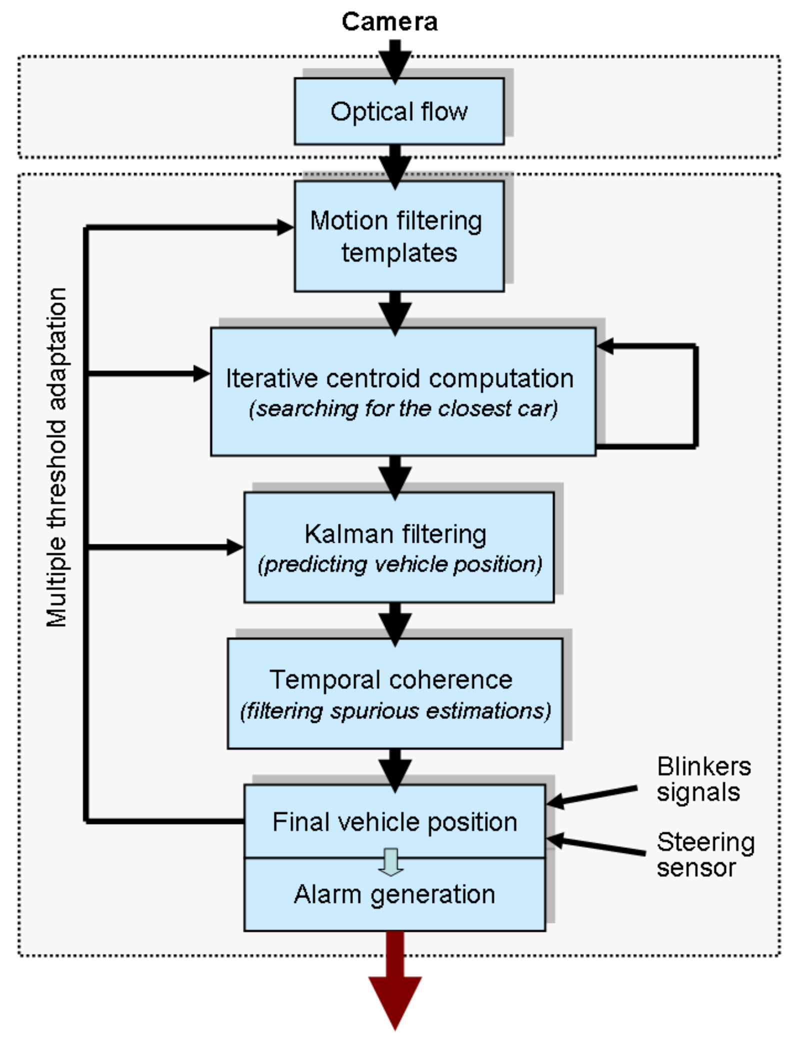

Camera
Optical flow
Multiple threshold adaptation
Motion filtering templates
Iterative centroid computation
(searching for the closest car)
Kalman filtering
(predicting vehicle position)
Temporal coherence
(filtering spurious estimations)
Blinkers signals
Steering sensor
Final vehicle position
Alarm generation

Figure 8.4. System functional blocks. Note that the different thresholds adapt dynamically according to the evolution of the recent scenario. Final alarm decision uses overtaking-car position, car-steering sensors and blinkers.

8.3. System architecture and FPGA resources consumption

The global system architecture is represented in Fig. 8.5. We have implemented a very regular datapath (without requiring specific interrupt handling) with a very deep pipeline structure (more than 70 stages) in order to achieve high performance.

The synchronization between the different processing units (frame-grabber, motion processing core and tracking unit) is done using specific memory data buffers which solves the problem associated to the different clock frequencies. The computing platform used to ZBT SSRAM memories whose capabilities have been exploited using a specifically designed Memory Management Unit (MMU) described in Chapter 6 that minimizes data delays and latencies. It is especially useful for the temporal filtering stage of the motion processing unit because it enables the use of FIR temporal filters which provide more stable estimations.

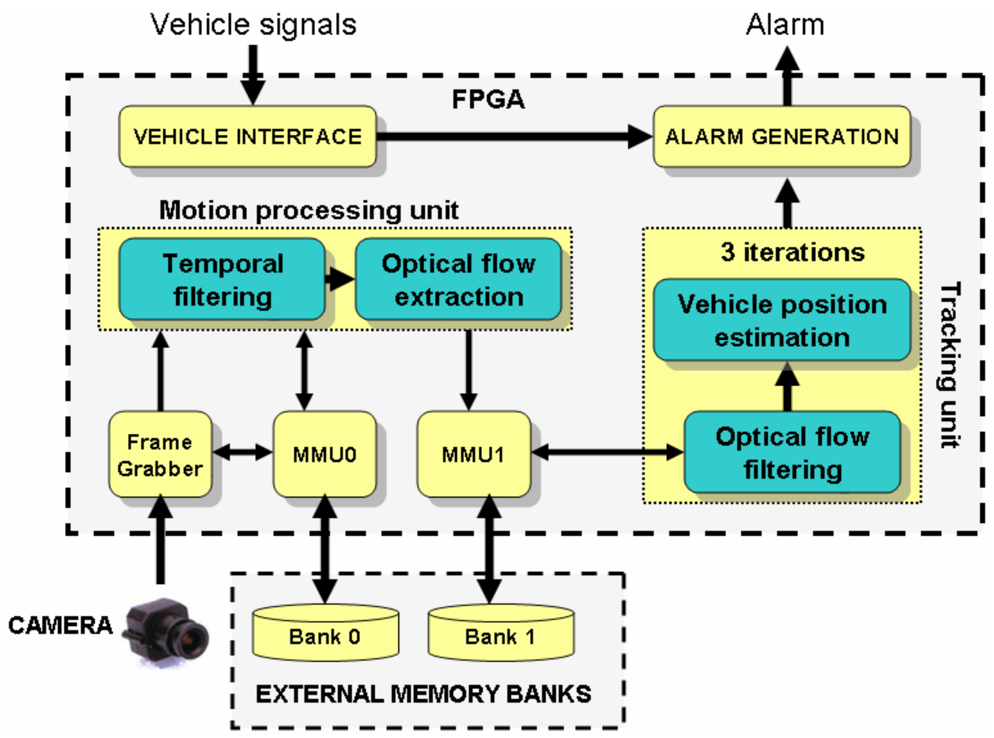


Figure 8.5. Overtaking monitor system architecture. All the processing stages and interfaces have been implemented using the FPGA as control element and processing unit. The whole system requires two external memory banks, a camera and vehicle interfaces for the alarm generation and external inputs encoding vehicle information such as speed, steering or lateral indicators.

The memory interchange strategy makes use of delays between processing units as synchronization technique. This makes possible the design of a very deep pipeline processing structure without using branch predictions that would degrade the performance. The high system throughput is based on this deep pipeline and on the parallel scalar units of different stages designed according to the Lucas & Kanade algorithmic complexity. Well balanced units are used to achieve a final system throughput of one estimation per clock cycle.

The performance of the optical flow unit makes possible to take advantage of high frame-rate cameras reducing the speed range to be processed (more time resolution) and leading to accurate tracking. Each stage has been designed with customized bit-widths from 8 (in the first stage) to 19 bits (in the last stage) with fixed-point and floating point data representation depending on required precision. More details about this architecture are given in Chapter 6.

In the tracking unit the templates computation has been implemented using convolution kernels which collect the information of the neighbourhood of each pixel [DIA06d]. The iterative process only requires some boundary image control to choose the area in which the centroid is computed. Finally, the Kalman filtering uses simple arithmetic operations which are computed once per frame.

Table 8.1. Basic stages gates resources consumption (results taken from the DK synthesizer [CEL06b]).

Pipelined stages	NAND gates	FFs	Memory bits	Max clock frequency (MHz)
Interfaces + hardware controllers	65881	2363	18208	45
Motion Processing core	1145554	6529	516096	45,5
Tracking core	12087	751	0	71

Table 8.2. System resources required on a Virtex II XC2V6000-4. First row contain the Optical flow processing system resources, taken from Table 6.8 and copied here for the sake of easy systems comparison. The whole overtaking car system monitor resources are shown in the second row. (Mpps: mega-pixels per second and it's the maximum system processing clock frequency, EMB stands for *embedded memory blocks*).

Slices / (%)	EMBS / (%)	Embedded multipliers / (%)	Mpps	Image Resolution	Fps
8250 (24%)	29 (20%)	12 (8%)	45.49	640x480	148
10073 (29%)	29 (20%)	12 (8%)	45,5	640x480	148

The gates consumption estimation of the different subcircuits is given on Table 8.1. Note that the tracking unit, provided that is implemented using iterative computation, allows efficient resources sharing (thus representing a relatively inexpensive stage). On the other hand, the motion processing unit requires the intensive exploitation of the parallelism capabilities of the FPGA device (representing the most expensive module in terms of chip area). The interfaces and hardware controllers also require a considerable number of resources. Global system resources are shown in Table 8.2 after synthesis. It requires less than 2 million gates in a Virtex-II FPGA.. The tracking stage is processed sequentially only requiring 5% more of the whole FPGA slices. This represents 17% of the global hardware resources consumed by the complete system.

8.4. System performance evaluation

Evaluating the accuracy and efficiency of the system for real-image sequences is not easy. A visual inspection of the results gives us some “quality hints” to evaluate the performance but this is not a valid “quality evaluation procedure”. Several authors have addressed this validation using forward or backward cameras mounted on cars [DEL97], [BET96].

For our application we have used a camera mounted in the rear-view mirror, this experimental setup is shown in Figure 8.6. We have tested the algorithm in different overtaking sequences provided by Hella KGaA Hueck & Co [HEL06b] with different vehicles and weather conditions. There are 20 sequences composed of more than 9,000 frames. Our goals are:

1. To detect the overtaking car as soon as possible.
2. To track it reliably.

It is a complex task because if we use a very sensitive system, continuous false alerts can render the approach useless and make the driver lose confidence in the system. The next section shows some qualitative results. Section 8.3.2 describes the system benchmark procedure. It should be remembered here that although there are some commercial initiatives working towards similar systems [MOB06], [VOL06], [FIC06], no performance evaluation or scientific benchmarking methodology seems to have been applied to date. This makes it impossible to compare the different approaches and estimate their applicability.

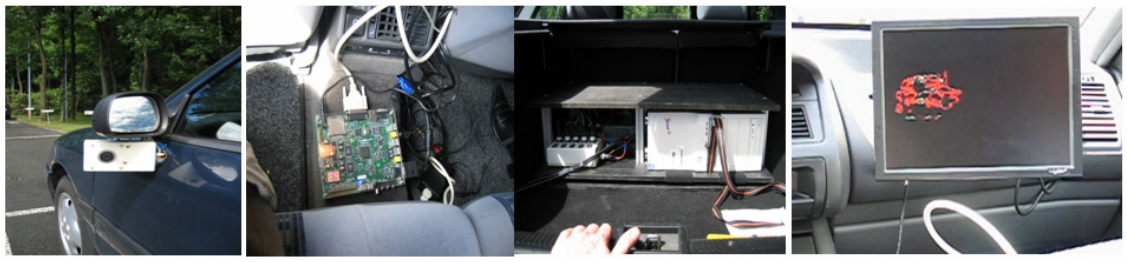


Figure 8.6. Experimental setup utilized for testing the system, in collaboration with Hellla [HEL06b].

8.4.1. Illustrative system results

In this section we illustrate some of the qualitative results obtained using different overtaking car sequences provided by Hella [HEL06b] with different vehicles and weather conditions. At the beginning of the overtaking maneuvering, when the vehicle is very small our system confidence measure is not reached. This means that we have not enough information but we have already unreliable position estimations. This has been marked as black squares in the figures. When the car is larger, confidence thresholds begin to be reached but without temporal consistency and, finally, the system is able to track accurately the vehicle until the end of the overtaking sequence. Reliable position is drawn in the figures using a white cross. For all the evaluated sequences, this situation is reached for very far distances of the overtaking car so the system performance is good for safe distances.

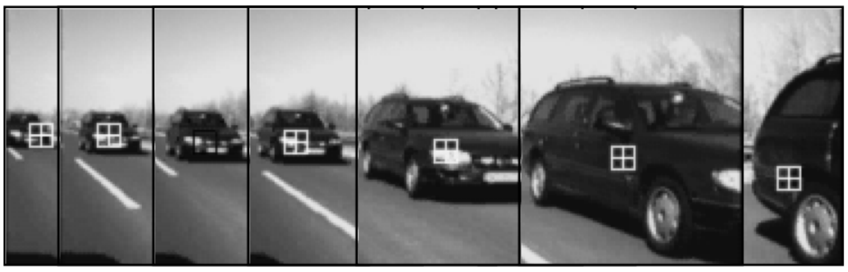


Figure 8.7. Overtaking with relative static situation with a black car in a sunny day. Sequence recorded using a conventional CCD camera.

An important problem occurs when the overtaking car velocity is equal to our car velocity, so the relative vehicle velocity will be around zero. In this situation the Kalman filtering allows us to keep the car position but the confidence value will not be reached, as it is seen in Figure 8.7. The system memory allows us to keep the car position under the confidence threshold (see black square in the third frame). The Alert signal system can use the estimation position and memory consistency to decide if we are in a dangerous situation or not.

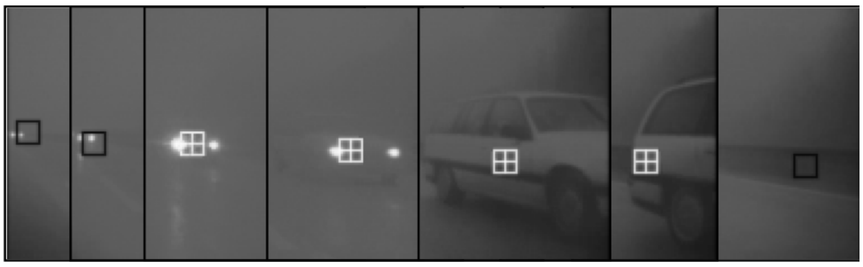


Figure 8.8. Car in a foggy and rainy day. Sequence recorded using a high dynamic range camera.

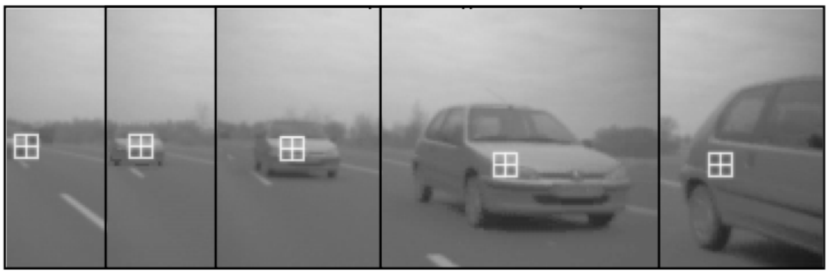


Figure 8.9. Car in a cloudy day. The car moves with the lights switched off. Sequence recorded using a high dynamic range camera.

In different weather and light conditions the kind of camera sensor is crucial and strongly motivates the use of high dynamic range cameras. The sequence of Figures 8.8 and 8.9 tests our system capability for very low contrast sequences. The weather conditions in the sequence of Figure 8.8 are really bad, in these situations lights become a very important source of information. Here the system needs closer cars to reach the confidence value to begin the car tracking reliably. In Figure 8.9 we test the robustness of the system to low contrast scenarios. This sequence has more contrast but the car has switched off the lights. As it can be seen the results are correct.

The sequence of Figure 8.2, Section 8.2.2.2, shows a complex scene. Several cars are overtaking in a highway. Each car is numbered using brackets. The figure shows different frames of the sequence and the dangerous car position estimations. As we explained in section 8.2.2.2, a multi-target system is not necessary and the system only marks the closest car (the most dangerous in the scene). One important problem occurs when we have multiple lanes. Motion information from monocular viewing can not give us information about car distance so it is difficult to know in which lane is detected the approaching car. We can use the road white lines to do that but the important issue is to be able to discriminate whether the situation is dangerous or not.

Our system is useful if it prevents us of changing lane when another vehicle is present in a dangerous situation. This problem will be addressed in the future.

In this figure we can see the car estimation inertia (Figure 8.2.B). It should be noted that when the system looks for a new car, the estimation is over the confidence threshold but in a wrong position. This occurs because the saliency map obtained from the optical flow has reliable information about the car position but the Kalman filter needs two or three frames to update its parameters. We can use a more complex model for the car tracking but, thinking in hardware implementation of an embedded system, it can represent an unnecessary computation overload, since for a real time system that computes 25 frames/s (or even more) this delay of the alert signal is not significant.

8.4.2. Benchmark methodology and system description

The idea that the car in question is the closest to us and therefore must be in the right-hand area of the image has important implications for our test. We are interested in detecting the car as soon as we can and not losing track of it, especially when it is close to us. We measure the distance in which reliable tracking starts to evaluate the quality of the system.

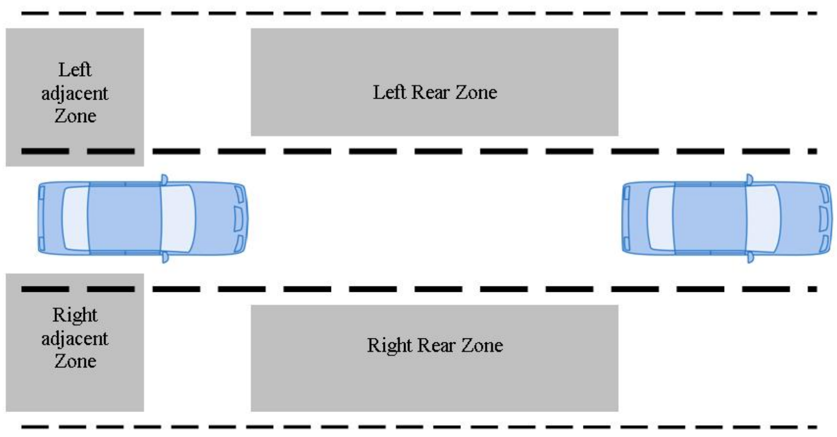
For benchmarking, special test sequences were recorded by Hella KGaA Hueck & Co [HEL06b] according to the preliminary version of the ISO standard (ISO / TC204 / WG14 / N40.27). Three systems are considered, based on the areas they cover (see Figure 8.10.a):

- Type I: *Blind Spot Warning*. This system is intended to warn only about target vehicles in the adjacent zones (the zones on the left and right of the subject vehicle). It is not required to provide warnings of target vehicles approaching the subject vehicle from the rear.
- Type II: *Closing Vehicle Warning*. This system is intended to warn about target vehicles that are approaching the subject vehicle from the rear.
- Type III: *Lane Change Warning*. This type combines the Blind Spot and Closing Vehicle functions.

We deliberately did not take into account in the first steps situations where cars enter the blind spot from the front, when the ego-vehicle overtakes cars in the adjacent right-hand lane.

The Type II specifications consider different closing speeds: A → 5-10 m/s; B →15-20 m/s; C → 25-30 m/s. For the evaluation we used two instrumented test cars, the target car (*TC*) which is the overtaking car, and the subject car (*SC*) which is equipped with the camera and the system described for tracking the target car. *TC* has a LIDAR sensor installed at the front [HEL06c] (illustrated in Figure 8.10.b) to measure the distance between the two vehicles. Both instrumented data-acquisition systems are synchronised to match the recorded frames of *SC* with the LIDAR information of *TC* at any time. An onboard computer stores this information for off-line analysis.

(a)



(b)

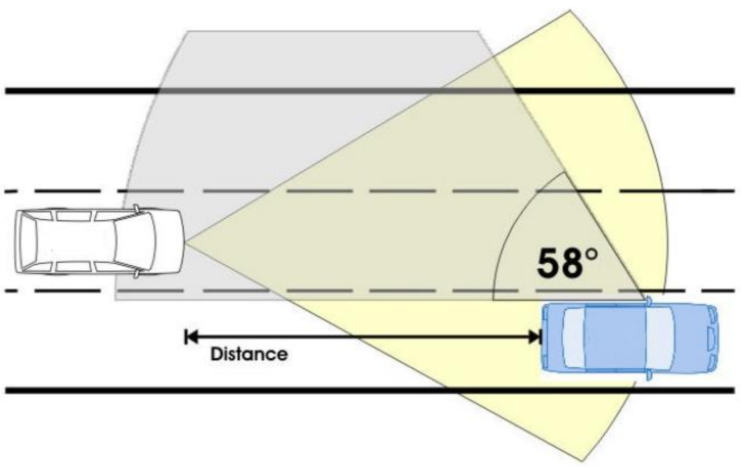


Figure 8.10. Vehicle areas and distances. (a) Car areas for device type classification. (b) Inter-car distances using the LIDAR sensor and camera view angle to cover the blind spot areas.

In the test scenario it is possible to get image data from *SC* and, as a reference, the value of the distance between both vehicles from the LIDAR sensor of the overtaking car. Most of the recorded video streams have corresponding LIDAR measured distances. Nevertheless, due to technical problems only a limited number of cross-validated sequences were recorded. Day and night scenarios where tested but we include here only the results from the day scenarios, which present the more difficult situation since at night the headlights of overtaking cars facilitate the tracking task.

8.4.3. Benchmark evaluation results

The results from our 20 cross-validated test sequences shown in Figure 8.11 indicate the distance between cars as measured using the LIDAR sensor. Basically, we have two different kinds of recorded sequences, one for the Type I system test (dark bars) and another for the Type II system test (light bars). In this case, three different approaching speeds are possible according to our preliminary standard (except for 3 lanes and 25-30 m/s where no distance information is available due to technical problems).

Figure 8.11 shows that cars approaching faster are reliably detected at longer distances (in the two-lane bars). This is highly desirable since the time-to-contact (*TTC*) is shorter in these situations. It is possible because *TC* is approaching faster and the motion cues become significant even when *TC* is still far away.

The sequences were taken under different visibility and weather conditions. This also affects the system performance (significantly in the third bar of the three-lane case in Figure 8.11).

These results show the high potential for a possible application within the framework of a driver-assistance system [DIA06g]. With these data we have been able to evaluate and classify the system, using the ISO draft, as nearly fulfilling the requirements for a Type III system (lane-change warning) with the subtype C (relative velocities up to 20 m/s).

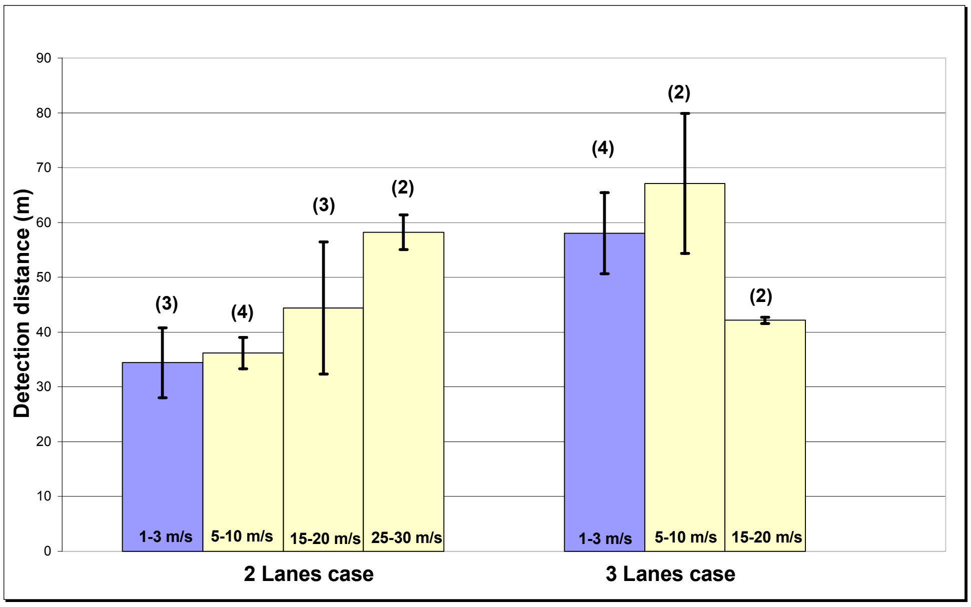


Figure 8.11. System evaluation results. The average detection distances (in meters) with their typical deviation are presented. Two different cases are considered: two-lane motorways and three-lane motorways. At the top of the bars we include the number of cross-validated sequences.

8.5. Conclusions

We have described a system to track overtaking cars using the rear-view-mirror perspective. Basically, we implement it in two steps: firstly we compute the optical flow, and then, after a filtering stage, this motion-saliency map represents reliably car points that are used to compute the overtaking car's position. We implemented a customized DSP for optical-flow computation combined with a tracking unit for alarm generation. Finally, we have also applied a benchmarking methodology with a wide set of diverse overtaking sequences to evaluate the system's performance. The results shown are very promising because the system is very reliable and stable, even for very difficult image sequences in poor visibility.

From Figure 8.11 we can also compute the *TTC* when the alarm signal is generated. Using the fastest velocity of each interval and considering the lowest detection distance (average value minus typical deviation), the worst case is 1.61 seconds, presented in the bar corresponding to two lanes and a closing speed of 15-20 m/s. Based on driver behaviour studies [GRE00], the worst reaction time for a driver is 1.5s for braking (less if we consider that the lane-change manoeuvre just implies a steering action, which is more than 0.15s faster than braking, and therefore the reaction time becomes 1.35s). Therefore we believe that our system can effectively alert the driver and leave him enough time to react.

There are still some open issues for future work, however. (1) On three-lane roads an overtaking car in the outside lane should not generate a warning signal. This implies a distinction between overtaking manoeuvres in the other two lanes. (2) Inverse overtaking scenarios, when the *SC* is overtaking the *TC*, the warning signal should be generated since lane changing would also generate a dangerous situation. (3) Smart warning strategy (human-machine interface field). Future work will cover these points and test the whole system into the car.

Capítulo 9
Conclusiones

Este capítulo constituye un resumen comentado de todo el trabajo presentado en esta memoria de tesis. Discutimos los diferentes resultados obtenidos interrelacionándolos y comentando sus campos de aplicación potencial.

El principal objetivo de este capítulo es presentar un resumen claro de las principales contribuciones de este trabajo y sus aspectos innovadores. Finalmente resaltamos explícitamente las conclusiones y contribuciones científicas.

9.1. Discusión

El principal objetivo de este trabajo era investigar diferentes modalidades visuales y su implementación en tiempo real utilizando hardware de propósito específico. Las modalidades visuales que han sido estudiadas e implementadas son las siguientes:

- Características locales de las imágenes (fase, energía y orientación). Véanse Capítulos 2 y 5.
- Movimiento. Véase los Capítulos 3 y 6.
- Estéreo. Véanse los Capítulos 4 y 7.
- Ejemplo de aplicación. Véase el Capítulo 8.

Después de los estudios de viabilidad de las diferentes técnicas para la extracción de cada una de esas características hemos diseñado arquitecturas de altas prestaciones de los modelos que permiten una implementación más eficiente. Estas arquitecturas son capaces de extraer estas modalidades visuales en tiempo real (a distintas resoluciones espacio-temporales, esto es importante ya que la frecuencia de muestreo temporal es crítico para la estimación de movimiento mientras que la resolución espacial es más relevante para el estéreo).

La implementación de arquitecturas de procesamiento específicas para visión es un campo interesante que requiere estrategias de diseño radicalmente distintas a las que se utilizan para arquitecturas basadas en un solo procesador de propósito general. Hemos estructurado las técnicas de procesamiento visual de forma apropiada para adaptarlas a un mejor aprovechamiento de flujos de datos regulares. Además hemos estudiado las operaciones en que se basan estas técnicas para identificar las etapas de procesamiento críticas y evaluar su viabilidad. Finalmente, hemos diseñado caminos de datos segmentados con grano fino para obtener arquitecturas de altas prestaciones. De hecho, los sistemas presentados superan en potencial de cálculo a todas las soluciones encontradas en la literatura (por ejemplo el sistema de procesamiento de movimiento supera en más de un orden de magnitud a todas las implementaciones previas publicadas hasta la fecha).

El trabajo se ha estructurado en las siguientes etapas:

- Evaluación de la viabilidad y compromiso entre eficiencia y precisión de las distintas técnicas de procesamiento de imágenes para la extracción de las modalidades visuales estudiadas. El objetivo de esta etapa preliminar es la elección de un modelo concreto en el que centrar el diseño de circuitos específicos.
- Adaptación del mejor modelo a un flujo de datos regular. En esta etapa adoptamos una estrategia de procesamiento que permite computación segmentada eficiente.
- Evaluación de las operaciones y los requerimientos de precisión en cada etapa de procesamiento. En este estudio evaluamos la profundidad de bits necesaria en cada paso del modelo y escogemos una representación de datos con aritmética en punto fijo o aritmética en punto flotante. En vez de realizar una búsqueda exhaustiva estudiamos brevemente las distingas operaciones y nos centramos en las etapas de procesamiento críticas. Para ello, hemos utilizado estrategias de evaluación apropiadas basadas en secuencias o imágenes sintéticas (con unos valores reales conocidos de los distintas características visuales que se pretenden extraer) y métricas de error utilizadas ampliamente en la literatura.
- Tras la implementación hemos evaluado los recursos hardware que requiere cada uno de los sistemas diseñados.
- Finalmente, hemos hecho un esfuerzo considerable en la realización de un estudio comparativo de las arquitecturas presentadas con otros sistemas e implementaciones publicadas por otros autores.

Esta metodología de trabajo ha sido adoptada para la implementación de tres sistemas: características locales de la imagen (fase, orientación y magnitud), movimiento y estéreo. Estas modalidades visuales no han sido escogidas arbitrariamente, requieren de modelos de procesamiento computacionalmente muy pesados (fundamentalmente basados en operaciones de convolución espacio-temporales). Consumen más del 90% de la carga computacional de los sistemas de visión complejos que también incluyen procesamientos de más alto nivel (como fusión multimodal, etc). De hecho, los tres sistemas presentados no deben ser vistos como independientes, sino que comparten la misma metodología de procesamiento (caminos de datos con segmentación de cauce de grano fino basados en circuitos de computación superescalares) y están basados en primitivas comunes (convoluciones espacio-temporales). Aunque los recursos computacionales de los distintos caminos de datos no pueden compartirse sin degradar las prestaciones, las tres arquitecturas se han implementado en un mismo dispositivo y juntas pueden considerarse como un sistema de visión multimodal de bajo nivel en un chip.

La implementación de este sistema visual multimodal ha sido motivada por distintos aspectos:

- **Arquitecturas de propósito específico.** La implementación de arquitecturas radicalmente diferentes a las de los procesadores de propósito general (ampliamente utilizados) es interesante en ciertos campos. El hecho de que los sistemas desarrollados en este trabajo se hallan podido estructurar en caminos de datos regulares nos ha empujado a investigar e implementar caminos de datos muy segmentados en microetapas basadas en circuitos superescalares. Este tipo de arquitecturas no se utiliza en el diseño de procesadores de propósito general debido a la flexibilidad que requieren sus recursos computacionales. Las condiciones del flujo de control de los algoritmos que se ejecutan en estas máquinas hacen que sistemas con caminos de datos supersegmentados no sean eficientes para la ejecución de cualquier tipo de código.
- **Aprender construyendo.** Una fuerte motivación para la implementación de esquemas de procesamiento bio-inspirados (en nuestro caso esquemas de procesamiento de modalidades visuales biológicas) es el aprendizaje del modo de funcionamiento de estos sistemas biológicos. Como ingenieros tratamos de copiar sistemas de procesamiento (como el esquema de estimación de movimiento) de los sistemas biológicos, en los que se consiguen unas prestaciones impresionantes con un número limitado de recursos computacionales. Durante la construcción de estos sistemas artificiales nos planteamos los mismos objetivos que los sistemas biológicos (por ejemplo estimación de movimiento en el área cerebral MT con una cantidad de recursos computacionales reducida). Esto nos fuerza a adoptar una actitud que nos ayuda a entender la función de determinados procesamientos intermedios que se dan en los sistemas nerviosos centrales. Por ejemplo, las convoluciones espacio-temporales (que es una primitiva ampliamente utilizada por los sistemas naturales en tareas de visión) que son combinadas de forma eficiente en los

sistemas de visión más avanzados que conocemos (que son los sistemas biológicos). De hecho, algunas de las técnicas adoptadas (por ejemplo, el modelo de extracción de estéreo) representan sistemas muy bio-inspirados. Pero en todo caso hemos adoptado una actitud oportunista: utilizando estrategias de computación que llevan a altas prestaciones y gran precisión pero no emulamos otras propiedades que no son fáciles de adaptar a la tecnología que estamos utilizando. Por ejemplo, implementamos caminos de datos supersegmentados en vez de computación distribuida y cooperativa de elementos de procesamiento independientes que sería una aproximación mucho más bio-inspirada.

- **Visión en tiempo real en un chip.** La disponibilidad de diferentes modalidades visuales en un solo chip abre las puertas a la exploración de esquemas de fusión sensorial en el marco de sistemas de visión de más alto nivel. Este tema está siendo estudiado en otros grupos de investigación internacionales en el marco del proyecto Europeo DRIVSCO. De hecho, está planificado que a corto plazo el sistema desarrollado se utilice para este propósito en cuatro centros europeos de investigación.
- **Aplicaciones reales.** Como se ha descrito en el Capítulo 8, algunas modalidades visuales, como el movimiento tienen una aplicación directa en algunos campos. Por ejemplo en sistemas embebidos para la vigilancia de vehículos durante maniobras de adelantamiento. Estamos explorando otros campos de aplicación como robots con navegación dirigida por visión (movimiento), robots para la manipulación de objetos utilizando estéreo, sistemas de visión aumentada (incluyendo diferentes modalidades visuales) para pacientes de baja visión, etc. Pero una correcta evaluación de las prestaciones de los sistemas presentados en esta memoria en estos campos de aplicación es parte de nuestro trabajo futuro.

Aunque el trabajo presentado se debe considerar un sistema de visión global podemos extraer conclusiones diferentes de las distintas arquitecturas desarrolladas:

- **Características locales (fase, orientación y magnitud).** Estas características pueden ser extraídas con distintos tipos de filtros que han sido estudiados. No se ha encontrado publicado ningún estudio comparativo entre las distintas alternativas a nivel de filtros. Distintos autores escogen una u otra alternativa a priori, presuponiendo que son aproximadamente equivalentes en precisión y carga computacional. Sin embargo, la implementación de un sistema eficiente nos ha forzado a realizar este estudio preliminar encontrando importantes diferencias entre los distintos filtros en cuanto a la precisión, cobertura de amplios rangos en frecuencias espaciales (típicos de escenas naturales) y carga computacional.

- **Movimiento.** En este caso existen estudios comparativos entre distintas técnicas para la extracción de movimiento. Esto ha facilitado la selección de un modelo concreto (el algoritmo de Lucas y Kanade) que alcanza un buen compromiso entre precisión y eficiencia. Para el procesamiento de movimiento el muestreo temporal es muy importante y es un tema que rara vez se ha estudiado o comentado. Para esta modalidad visual el error debido a la baja frecuencia de muestreo temporal es importante porque convencionalmente se utilizan cámaras comerciales que trabajan a 25 o 30 imágenes por segundo. Hemos desarrollado una arquitectura de altas prestaciones capaz procesar hasta 95 imágenes por segundo. Esto hace que se pueda utilizar de forma eficaz con sensores avanzados o camaras digitales de bajo coste capaces de adquirir imágenes con mayor frecuencia. La calidad de las estimaciones de movimiento que se pueden extraer con esta combinación de arquitectura de altas prestaciones y sensor avanzado es mucho mejor que la que se obtiene por métodos más complejos en plataformas que no pueden procesar en tiempo real más que secuencias tomadas a 30 imágenes por segundo.
- **Estéreo.** La extracción de información estéreo con alta precisión requiere imágenes de mucha resolución espacial y áreas de estimación de disparidad amplias. Por lo tanto en este caso, una arquitectura de altas prestaciones es de interés para computar áreas amplias de estimación de disparidades (filtros espaciales grandes). Para esta modalidad visual hemos adoptado un modelo que se puede implementar en hardware específico de forma eficiente ya que no requiere del cálculo directo de diferencia de fases espaciales.

La implementación de sistemas de cierta complejidad es difícil con los lenguajes de descripción de hardware más ampliamente utilizados (como VHDL o Verilog). Nosotros hemos escogido un lenguaje de más alto nivel (Handel-C) que permite la definición de arquitecturas de cómputo de altas prestaciones, además facilita la gestión del paralelismo durante el diseño de distintas arquitecturas. Hemos definido circuitos específicos como la unidad de manejo de memoria para multiplexar eficientemente los accesos a memoria, lo cual es un punto crítico cuando se diseñan caminos de datos muy segmentados. Este tipo de recursos ha permitido la definición de sistemas de cierta complejidad a un alto nivel de abstracción (ya que, por ejemplo, no es precisa la gestión directa de los accesos a memoria). Además la utilización de un lenguaje de descripción de hardware muy algorítmico ha facilitado la implementación de modelos descritos originalmente de esta manera. Aunque sin embargo, la implementación de arquitecturas de altas prestaciones ha hecho necesario el diseño de los caminos de datos a un nivel de transferencia de registros (RTL).

9.2. Publicación de resultados

Se pueden destacar dos conceptos o tópicos sobre el trabajo que se ha presentado:

a) Los circuitos de visión pueden considerarse modelos de visión genuinamente nuevos, con comportamientos parecidos a los correspondientes modelos software pero con compromisos totalmente diferentes entre precisión y velocidad de procesamiento. Por lo tanto, se han presentado como "modelos nuevos", se han descrito y evaluado comparando sus resultados con otras soluciones de otros autores. Esta metodología científica ha facilitado la publicación de los circuitos desarrollados.
b) Todos los sistemas han sido diseñados como arquitecturas supersegmentadas (con unidades de computación superescalares en ciertas etapas críticas) lo cual permite una utilización eficiente de los recursos inherentemente paralelos de los dispositivos de tipo FPGA. Esta metodología de diseño es bastante novedosa y ha sido adoptada en pocas ocasiones por otros autores. Sin embargo, se ha probado que es una metodología de diseño muy eficiente para obtener sistemas de altas prestaciones de propósito específico.

Estos dos argumentos constituyen las bases de los sistemas presentados en este trabajo y han recibido muy buenas críticas en las revisiones científicas que hemos recibido. Además algunos de los resultados relacionados con esta tesis se han publicado en los siguientes artículos:

Revistas internacionales con índice de impacto (SCI)

[1] R. Agís, J. Díaz, E. Ros, R. Carrillo, E. M. Ortigosa, **"**Hardware Event-driven Simulation Engine for Spiking Neural Networks**"**, *International Journal of Electronics* (Taylor & Francis Group),2006, (en publicación)..

[2] J. Díaz, E. Ros, F. Pelayo, E. M. Ortigosa and S. Mota, "FPGA based real-time optical-flow system," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 2, pp. 274-279, 2006

[3] J. Díaz, E. Ros, S. Mota and R. Rodriguez-Gomez, **"**FPGA based architecture for motion sequence extraction**"**, *International Journal of Electronics* (Taylor & Francis Group), 2006 (en publicación).

[4] J. Díaz, E. Ros, S. Mota, F. Pelayo, and E. M. Ortigosa, "Sub-pixel motion computing architecture," *IEE Proc. Vision, Image & Signal Processing*, 2006, (en publicación).

[5] J. Diaz, E. Ros, A. Rotter, M. Muehlenberg, "Lane-change decision aid system based on motion-driven car tracking," *IEEE Transactions on Intelligent Transportation Systems*, 2006, actualmente en revisión.

[6] J. Díaz, E. Ros, R. Carrillo and A. Prieto, "Real-time system for high-image-resolution disparity," *IEEE Trans. on Image Processing*, 2006, actualmente en revision.

[7] E. M. Ortigosa, A. Cañas E. Ros, P. M. Ortigosa, S. Mota, J. Díaz, "Hardware description of multi-layer perceptrons with 3 different abstraction levels," *Microprocessors and Microsystems*, 2006, (en publicación).

[8] J. Díaz, E. Ros, S. P. Sabatini, F. Solari y S. Mota, "A Phase based stereo system-on-a-chip," *BioSystems journal,2006,* to be published.

[9] S. Mota, E. Ros, J. Díaz, E. M. Ortigosa, and A. Prieto, "Motion-Driven Segmentation by Competitive Neural Processing," *Neural Processing Letters*, vol.22, no 2, pp. 125-147, 2005.

[10] F. Vargas-Martin, M. D. Peláez-Coca, E. Ros, Díaz J. and S. Mota, "Optoelectronic Visual Aid Based on Reconfigurable Logic for Severe Peripheral Vision Loss Rehabilitation", *Ophthalmic Research*, vol. 36, no. 1, pp. 60, 2004

Revistas Internacionales con congreso asociado

[1] R. Agís, J. Díaz, E. Ros, R. Carrillo, E. M. Ortigosa, **"**Event-driven simulation engine for spiking neural networks on a chip," *Lecture Notes On Computer Science*, Springer-Verlag, 2006, (en publicación).

[2] J. Díaz, E. Ros, S. Mota and R. Rodriguez-Gomez, "Highly paralellized architecture for image motion estimation, " *Lecture Notes On Computer Science*, Springer-Verlag, 2006, (en publicación).

[3] J. Díaz, E. Ros, S. Mota and R. Agis, "Real-time embedded system for rear-view mirror overtaking car monitoring," *Lecture Notes On Computer Science*, Springer-Verlag, 2006, (en publicación).

[4] S. Mota, E. Ros, J. Diaz, and F. de Toro, "General purpose real-time image segmentation system, "*Lecture Notes On Computer Science*, Springer-Verlag, 2006, (en publicación).

[5] S. Mota, E. Ros, J. Díaz, R. Agis and F. de Toro, "Bio-inspired motion-based object segmentation," *Lecture Notes On Computer Science*, Springer-Verlag, 2006, (en publicación).

[6] E. Ortigosa, A. Cañas, R. Rodriguez, J. Diaz, S. Mota, "Towards an optimal implementation of MLP in FPGA," *Lecture Notes On Computer Science*, Springer-Verlag, 2006, (en publicación).

[7] E. Ros, J. Díaz, S. Mota, F. Vargas-Martín and M.D. Peláez-Coca, "Real time image processing on a portable aid device for low vision patients," *Lecture Notes On Computer Science*, Springer-Verlag, 2006, (en publicación).

[8] J. Díaz, E. Ros, S. Mota, R. Carrillo, R. Agís, "Real time optical flow processing system," *Lecture Notes in Computer Science,* vol. 3203, pp.617-626, 2004.

[9] S. Mota, E. Ros, J. Díaz, E. M. Ortigosa, R. Agís and R. Carrillo R, "Real-time visual motion detection of overtaking cars for driving assistance using FPGAs," Lecture Notes in Computer Science vol. 3203, pp. 1158-1161, 2004.

[10] J. Díaz , S. Mota, E. Ros and Guillermo Botella, "Neural Competitive Structures for Segmentation Based on Motion Features," *Lecture Notes in Computer Science*, vol. 2686, pp. 710-717, 2003

Conferencias internacionales

[1] J. Díaz, E. Ros, S. Mota, E. M. Ortigosa, and B. del Pino, "High performance stereo computation architecture," in *Proc. of IEEE Int. Conf. on Field Prog. Logic and Applications (FPL'05),* Tampere, Finland, August 2005, pp. 463-468.

[2] F. Vargas-Martín, M. D. Peláez-Coca, E. Ros, S. Mota and J. Díaz, "A general real-time video processing unit for low vision," presented at Vision 2005, London, England, April 2005

[3] J. Díaz, E. Ros, S. Mota, E.M. Ortigosa, "Real-time optical flow computation using FPGAs", presented at the Early Cognitive Vision Workshop, Isle of Skye, Scotland, UK, 28 May- 1 June 2004. [Online] Available: http://www.cn.stir.ac.uk/ecovision-ws/schedule.php

[4] S. Mota, E. Ros, J. Díaz, G. Botella, F. Vargas, and A. Prieto, "Motion driven segmentation scheme for car overtaking sequences," In *Proc. of 10th. International Conference on Vision in Vehicles (VIV'2003)*, Granada, España, 2003.

[5] S. Mota, E. Ros, J. Díaz, S. Tan, J. Dale and A. Johnston, "Detection and tracking of overtaking cars for driving assistance," presented at the Early Cognitive Vision Workshop, Isle of Skye, Scotland, UK, 28 May- 1 June 2004. [Online] Available: http://www.cn.stir.ac.uk/ecovision-ws/schedule.php

[6] E. Ros, J. Díaz, S. Mota, "Neural multilayer structure for motion pattern segmentation," presented at Brain Inspired Cognitive Systems (BICS-2004), Stirling, Scotland (UK), August 2004

[7] F. Vargas-Martin, M. D. Peláez-Coca, E. Ros, J. Díaz and S. Mota, “Video processing based on reconfigurable logic for low vision aids,” presented at the II EOS Topical Meeting on Physilogical Optics*,* Granada, España, September 2004

[8] F. Vargas-Martin, M. D. Peláez-Coca, E. Ros, J. Díaz. and S. Mota, “Augmented view visual aid based on reconfigurable logic for peripheral vision loss”, presented at the Conference and Workshop on Assistive Technologies for Vision and Hearing Impairment, CVHI’2004, Granada, España, June 2004.

[9] F. Vargas-Martín, M.D. Peláez-Coca, E. Ros, J. Díaz, S. Mota, “Optoelectronic visual aid base don reconfigurable logia for severe peripheral vision loss rehabilitation,” presented at the European Association for Vision and Eye Research (EVER2004), Vilamoura, Portugal, October 2004

[10] J. Díaz, E. Ros, S. Mota, G. Botella, A. Cañas, and S. Sabatini, “Optical flow for cars overtaking monitor: the rear mirror blind spot problem,” presented at 10th. International Conference on Vision in Vehicles (VIV’2003), Granada, España, September 2003

Conferencias nacionales

[1] J. Díaz, E. Ros, R. Rodríguez-Gómez and B. del Pino, ““Análisis y diseño de una arquitectura súpersegmentada de altas prestaciones para estimación de movimiento,” VI Jornadas sobre Computación Reconfigurable y Aplicaciones (JCRA 2006), Cáceres, España, September 2006, (en publicación).

[2] J. Díaz, E. Ros, and S. Mota, “Arquitectura para cómputo de estéreo en imágenes de alta resolución mediante hardware reconfigurable,” presented at *Jornadas sobre computación reconfigurable y aplicaciones,* Granada, España, September 2005, pp. 167-172.

[3] J. Díaz, E. Ros, S. Mota, E. M. Ortigosa, R. Carrillo, and R. Agis, “Cómputo de flujo óptico en tiempo real mediante FPGAs,” presented at *Jornadas sobre computación reconfigurable y* aplicaciones, Barcelona, España 13-16 septiembre 2004, pp. 481-488.

[4] E. Ros-Vidal, J. Díaz, S. Mota, and F. Vargas-Martín, “Procesamiento de imágenes en tiempo real para ayuda a pacientes con baja vision”, presented at the *XXI Congreso Anual de la Sociedad Española de Ingeniería Biomédica, (CASEIB 2003),* Mérida, España, November 2003, pp. 153-156.

9.3. Conclusiones

Finalmente resumimos las principales contribuciones del trabajo presentado:

- Hemos estudiado distintas alternativas basadas en diferentes filtros espaciales para la extracción de características locales. Hemos evaluado las distintas alternativas con imágenes sintéticas para comparar sus precisiones y hemos estimado la carga computacional de cada una de ellas. Las *derivadas Gausianas de segundo orden* han resultado las más ventajosas por su buen compromiso entre precisión y eficiencia.

- Hemos implementado una arquitectura de altas prestaciones para la extracción de características locales. Hemos evaluado los requerimientos de recursos computacionales, las prestaciones del sistema y la pérdida en precisión debida al número limitado de profundidad de bits en los datos en las distintas etapas.
- Hemos diseñado dos sistemas alternativos para el procesamiento de movimiento. Las dos aproximaciones tienen distinta precisión y requerimientos computacionales. Estos dos sistemas son los siguientes:
    - Una arquitectura con un camino de datos segmentado en grano grueso basado en el modelo original de Lucas y Kanada pero utilizando filtros temporales IIR para reducir los recursos de almacenamiento temporal. De hecho hemos definido distintas versiones de este sistema con diferente grado de paralelismo en las etapas críticas para permitir su implementación en dispositivos de distinto coste y potencia.
    - Una arquitectura con un camino de datos supersegmentado (en grano fino) basado en el modelo modificado de Lucas y Kanade propuesto por Brandt [BRA97] que utiliza filtros temporales FIR. Este sistema representa una opción de muy altas prestaciones capaz de procesar secuencias a más de 30 imágenes por segundo (a 95 imágenes por segundo con una resolución VGA). Esta arquitectura tiene gran interés cuando se utiliza con sensores avanzados (capaces de captar secuencias con alta resolución temporal). El sistema presentado supera en más de un orden de magnitud las soluciones publicadas hasta el momento.
- Hemos aplicado la arquitectura de procesamiento de movimiento en tiempo real en el marco de un sistema de asistencia a la conducción para monitorizar el cambio de carril durante maniobras de adelantamiento. Los resultados prueban claramente la utilidad de la solución presentada, ya que el sistema es capaz de detectar y seguir vehiculos en proceso de adelantamiento de forma fiable a distancias en las cuales los conductores tienen tiempo para reaccionar.
- Hemos implementado una arquitectura de altas prestaciones para la extracción de información estéreo basada en un modelo bio-inspirado. Hemos evaluado las

etapas de computación que más afectan a la precisión del sistema. Hemos estudiado los requerimientos hardware del sistema y sus prestaciones comparándolos con otras plataformas descritas por otros autores.

- Hemos mostrado a lo largo del diseño de los diferentes sistemas una metodología para el estudio de la profundidad de bits de las distintas etapas y del tipo de aritmética a utilizar, permitiendo de esta manera un compromiso optimizado entre el recursos utilizados y precisión del sistema.
- Hemos mostrado que el uso de sistemas altamente paralelos (múltiples unidades escalares y cauce supersegmentado) permite obtener una alta potencia de procesamiento. Este método de trabajo es poco utilizado en la literatura pero hemos demostrado su viabilidad y eficiencia para el diseño de caminos de datos en circuitos específicos de procesamiento de imágenes.

Chapter 9
Conclusions

This chapter summarizes the whole work that has been done and presented in this PhD memory. We discuss the different results relating them with each other and with potential application fields.
The main goal of this chapter is to summarize in a clear manner the different contributions and their innovation aspects of the presented work. Finally we highlight specific conclusions and scientific contributions.

9.1. Discussions

The main purpose of this work was to investigate different vision processing modalities and their implementation in real-time using specific hardware. The vision modalities that have been studied and implemented are the following:

- Local image features (phase, energy and orientation). See Chapters 2 and 5.
- Motion. See Chapters 3 and 6.
- Stereo. See Chapters 4 and 7.
- Application example: See Chapter 8.

After feasibility studies about the different techniques for extracting each of these features, we have designed efficient computing architectures of hardware friendly models able to extract these features in real-time (at different frame-rates since the temporal sampling may be critical for some of them, for instance motion estimation, while spatial resolution may be more relevant for stereo).

The implementation of specific processing architectures for vision is a highly challenging field which requires design strategies radically different to the general purpose single-processor architectures. We have structured the vision processing techniques in a proper manner to adapt them to regular data flows. After this we have studied the different operations in which they are based evaluating the feasibility and critical stages of each technique. Finally, we have designed fine grain pipelined datapaths in order to achieve high performance computing architectures. In fact, to the best of our knowledge the presented architectures outperform any existing solution published in the literature (for instance the motion processing system outperforms in more than one order of magnitude any existing solution published so far).

The work has been structured in the following stages:

- Evaluation of the feasibility and efficiency vs accuracy trade-off of the different techniques. The goal of this stage is to choose a concrete approach to be implemented in hardware.
- Adaptation of the best model to a feed-forward regular data flow. In this stage we adopt a processing strategy that allows efficient pipelined computing.

- Study of the operations and accuracy requirements of each processing stage. In this study we evaluate the bit width necessary at each step of the design of the model and we choose between fix point arithmetic and floating point data type. Instead of doing an exhaustive search in this stage we evaluate briefly all the different operations and we explicitly focus on the critical stages. For this purpose, we have used a proper benchmarking strategy based on synthetic sequences or images (with known ground-truth of the goal visual modality) and error metrics widely used in the literature.
- After the implementation, we have evaluated the hardware resources requirements of the designed systems.
- Finally, we have done a considerable effort in benchmarking the proposed computing architectures in terms of accuracy and performance with previously published approaches.

This working methodology has been adopted for the implementation of mainly three systems: local image features (phase, orientation and energy), motion and stereo. These visual modalities have not been chosen arbitrary. They represent very expensive processing models (mainly based on extensive spatio-temporal convolution operations). They consume more than 90% of the computational load of complex vision systems that also include higher vision tasks (such as multimodal fusion, etc). Furthermore, the three designed systems should not be seen as independent processing architectures. They share the same processing methodology (very deep pipelined datapaths composed of superscalar computation circuits) and they are basically based on common primitives (spatio-temporal convolutions). Although the computing resources of the different datapaths cannot be shared without loosing performance, the three processing architectures can be implemented on a single device and all of them together can be seen as a low level multimodal vision system on a chip.

The implementation of such a low level multimodal vision system has several motivations:

- **Specific purpose architectures.** Implementation of computing architectures radically different of the most widely used general purpose single-processor platforms. Since the systems that have been developed in this work can be structured in very regular data flows this has prompted us to investigate and implement superpipelined and superscalar datapaths. These architectures are not affordable in general purpose processors due to the required flexibility of their processing resources. The branching conditions of the dataflow of general processors make superpipelined approaches not convenient for processors that need to run arbitrary codes.

- **Understanding by building.** A very strong motivation for the implementation of bio-inspired processing schemes (in our case processing schemes of biological visual modalities) is the understanding of biological systems. As engineers we try to copy processing systems (such as motion estimation) in which biological approaches show impressive performances with a restricted number of computational resources. We address the same goals as biological systems (for instance motion estimation in the brain area MT) with constrained resources. This forces us to adopt an attitude that helps to clarify intermediate computations that take place in the biological systems. For instance, this is the case of spatio-temporal convolutions (which is a primitive widely used for vision tasks by nature) which are efficiently combined in the smart biological visual systems. In fact, some of the techniques adopted (such as the stereo model) implement highly bio-inspired approaches. Nevertheless, we have adopted an opportunistic attitude: using computing strategies that lead to high accuracy and performance but we do not adopt or emulate other properties which do not fit the goal technology in which our system will be implemented (for instance we implement very deep pipelined datapaths instead of distributed and cooperative computing of independent and asynchronous processing elements).
- **Real-time vision on a chip.** The availability of different visual modalities in real-time on a chip opens the door to explore sensory fusion schemes in the framework of higher level vision system. This issue is being studied by other European labs in the framework of the European project DRIVSCO. In fact, in the short run, the vision system developed in this work is planed to be soon working on 4 European Universities.
- **Specific real applications.** As described in Chapter 8, some visual modalities, such as motion have very high potential applications in the framework of embedded systems for instance for car tracking during overtaking maneuvers. We are exploring other application fields such as robot navigation based on optic flow, robotic object manipulation based on stereo, augmented vision

systems (including different vision modalities) for low vision patients, etc. But a correct evaluation of the performance of our approaches in these fields is part of our future work.

Although the presented work should be seen as a global vision system we can extract different conclusions from the different implemented architectures:

- **Local features (phase, orientation and energy).** These features can be extracted with different kinds of filters which have been studied. This comparative study has not been found in the literature before, normally one or other approach is just adopted by different authors a priori, because they consider them more or less equivalent in terms of accuracy and computational load. But the implementation of an efficient system extracting these features requires a preliminary study. We have found out significant differences between the different filters in terms of accuracy, robustness against wide spatial frequencies (usually present in natural scenarios) and computational load.
- **Motion.** The extensive comparative studies about different techniques available in the literature have facilitated the selection of an approach (the Lucas and Kanade algorithm) with a very good accuracy vs efficiency trade off. For motion processing, temporal sampling is very important and is an issue which is seldom addressed or commented. For this vision modality is critical the temporal aliasing, since the conventional cameras work at 25 or 30 frames per second. We have developed a high performance motion computing architecture able to process up to 95 frames per second which can be used with advanced sensors or even low cost digital cameras able to acquire "oversampled sequences". The quality of the motion estimations extracted with this combination of high performance computing architecture and advance sensors clearly outperforms more sophisticated approaches which cannot process sequences at more than 30 frames per second in real-time.
- **Stereo.** Accurate stereo computation requires high spatial resolution and large disparity estimation areas. Therefore, in this case high performance is highly desirable since it allows the computation of larger disparity estimation areas (larger spatial filters). For this vision modality we have adopted a hardware friendly scheme which avoids unnecessary computations of phase differences.

The implementation of systems of a certain complexity becomes difficult with the most widely used hardware description languages (such as VHDL or Verilog). Because of that we have used a higher level description language (Handel-C) which allows the definition of high performance computing architectures and easily managing the computing parallelism of the architecture. We have defined specific circuits such as the memory management unit to efficiently multiplex the memory accesses which is a critical issue when defining very deep pipelined datapaths. This allows designing systems of a certain complexity at a high level that properly abstracts the description of specific low level issues (such as external memory access).

We have used an algorithmic-like hardware description language which facilitates the implementation of models described as algorithms. Nevertheless, the design of high performance computing architectures has required the definition of the datapaths at a register transfer level.

9.2. Publication of the results

There are two general points that can be specifically highlighted about the work presented here:

a. The described vision circuits can be seen as genuine new models (with behaviors similar to their software counterparts) but with completely different processing speed vs accuracy trade-offs. Therefore, as “new models” they have been presented, discussed and evaluated comparing their results with other approaches described in the literature. This scientific methodology has facilitated the publication of the presented circuits.
b. All the systems have been designed as deep pipelined computing architectures (with superscalar datapaths at certain critical stages) which enables the efficient use of the available parallel processing resources at FPGA devices. This design methodology is quite novel and seldom adopted by other authors. Nevertheless, it has been proven to be a very valid tool to obtain high performance specific purpose systems.

These two issues are a strong motivation for all the implementations presented in this dissertation and when reviewed in scientific publications have received very good marks. Besides the works that are currently under review, some of the results related with this thesis have been published in the following papers:

International Journals with Scientific Impact (SCI)

[1] R. Agís, J. Díaz, E. Ros, R. Carrillo, E. M. Ortigosa, **"**Hardware Event-driven Simulation Engine for Spiking Neural Networks**"**, *International Journal of Electronics* (Taylor & Francis Group),2006, to be published.

[2] J. Díaz, E. Ros, F. Pelayo, E. M. Ortigosa and S. Mota, "FPGA based real-time optical-flow system," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 2, pp. 274-279, 2006

[3] J. Díaz, E. Ros, S. Mota and R. Rodriguez-Gomez, **"**FPGA based architecture for motion sequence extraction**"**, *International Journal of Electronics* (Taylor & Francis Group), 2006 to be published.

[4] J. Díaz, E. Ros, S. Mota, F. Pelayo, and E. M. Ortigosa, "Sub-pixel motion computing architecture," *IEE Proc. Vision, Image & Signal Processing*, 2006, to be published

[5] J. Diaz, E. Ros, A. Rotter, M. Muehlenberg, "Lane-change decision aid system based on motion-driven car tracking," *IEEE Transactions on Intelligent Transportation Systems*, 2006, submitted for publication.

[6] J. Díaz, E. Ros, R. Carrillo and A. Prieto, "Real-time system for high-image-resolution disparity," *IEEE Trans. on Image Processing*, 2006, submitted for publication.

[7] E. M. Ortigosa, A. Cañas E. Ros, P. M. Ortigosa, S. Mota, J. Díaz, "Hardware description of multi-layer perceptrons with 3 different abstraction levels," *Microprocessors and Microsystems*, 2006, to be published.

[8] J. Díaz, E. Ros, S. P. Sabatini, F. Solari y S. Mota, "A Phase based stereo system-on-a-chip," *BioSystems journal,2006,* to be published.

[9] S. Mota, E. Ros, J. Díaz, E. M. Ortigosa, and A. Prieto, "Motion-Driven Segmentation by Competitive Neural Processing," *Neural Processing Letters*, vol.22, no 2, pp. 125-147, 2005.

[10] F. Vargas-Martin, M. D. Peláez-Coca, E. Ros, Díaz J. and S. Mota, "Optoelectronic Visual Aid Based on Reconfigurable Logic for Severe Peripheral Vision Loss Rehabilitation", *Ophthalmic Research*, vol. 36, no. 1, pp. 60, 2004

International Journals with associate Conferences

[1] R. Agís, J. Díaz, E. Ros, R. Carrillo, E. M. Ortigosa, **"**Event-driven simulation engine for spiking neural networks on a chip," *Lecture Notes On Computer Science*, Springer-Verlag, 2006 to be published.

[2] J. Díaz, E. Ros, S. Mota and R. Rodriguez-Gomez, “Highly paralellized architecture for image motion estimation, “ *Lecture Notes On Computer Science*, Springer-Verlag, 2006, to be published.

[3] J. Díaz, E. Ros, S. Mota and R. Agis, “Real-time embedded system for rear-view mirror overtaking car monitoring,” *Lecture Notes On Computer Science*, Springer-Verlag, 2006, to be published.

[4] S. Mota, E. Ros, J. Diaz, and F. de Toro, “General purpose real-time image segmentation system, “*Lecture Notes On Computer Science*, Springer-Verlag, 2006, to be published.

[5] S. Mota, E. Ros, J. Díaz, R. Agis and F. de Toro, “Bio-inspired motion-based object segmentation,” *Lecture Notes On Computer Science*, Springer-Verlag, 2006, to be published.

[6] E. Ortigosa, A. Cañas, R. Rodriguez, J. Diaz, S. Mota, “Towards an optimal implementation of MLP in FPGA,” *Lecture Notes On Computer Science*, Springer-Verlag, 2006, to be published.

[7] E. Ros, J. Díaz, S. Mota, F. Vargas-Martín and M.D. Peláez-Coca, “Real time image processing on a portable aid device for low vision patients,” *Lecture Notes On Computer Science*, Springer-Verlag, 2006, to be published.

[8] J. Díaz, E. Ros, S. Mota, R. Carrillo, R. Agís, “Real time optical flow processing system,” *Lecture Notes in Computer Science,* vol. 3203, pp.617-626, 2004.

[9] S. Mota, E. Ros, J. Díaz, E. M. Ortigosa, R. Agís and R. Carrillo R, “Real-time visual motion detection of overtaking cars for driving assistance using FPGAs,” Lecture Notes in Computer Science vol. 3203, pp. 1158-1161, 2004.

[10] J. Díaz , S. Mota, E. Ros and Guillermo Botella, “Neural Competitive Structures for Segmentation Based on Motion Features,” *Lecture Notes in Computer Science*, vol. 2686, pp. 710-717, 2003.

International Conferences

[1] J. Díaz, E. Ros, S. Mota, E. M. Ortigosa, and B. del Pino, “High performance stereo computation architecture,” in *Proc. of IEEE Int. Conf. on Field Prog. Logic and Applications (FPL’05),* Tampere, Finland, August 2005, pp. 463-468.

[2] F. Vargas-Martín, M. D. Peláez-Coca, E. Ros, S. Mota and J. Díaz, “A general real-time video processing unit for low vision,” presented at Vision 2005, London, England, April 2005.

[3] J. Díaz, E. Ros, S. Mota, E.M. Ortigosa, “Real-time optical flow computation using FPGAs”, presented at the Early Cognitive Vision Workshop, Isle of Skye, Scotland, UK, 28 May- 1 June 2004. [Online] Available: http://www.cn.stir.ac.uk/ecovision-ws/schedule.php

[4] S. Mota, E. Ros, J. Díaz, G. Botella, F. Vargas, and A. Prieto, “Motion driven segmentation scheme for car overtaking sequences,” In *Proc. of 10th. International Conference on Vision in Vehicles (VIV'2003)*, Granada, Spain, 2003.

[5] S. Mota, E. Ros, J. Díaz, S. Tan, J. Dale and A. Johnston, "Detection and tracking of overtaking cars for driving assistance," presented at the Early Cognitive Vision Workshop, Isle of Skye, Scotland, UK, 28 May- 1 June 2004. [Online] Available: http://www.cn.stir.ac.uk/ecovision-ws/schedule.php

[6] E. Ros, J. Díaz, S. Mota, "Neural multilayer structure for motion pattern segmentation," presented at Brain Inspired Cognitive Systems (BICS-2004), Stirling, Scotland (UK), August 2004.

[7] F. Vargas-Martin, M. D. Peláez-Coca, E. Ros, J. Díaz and S. Mota, "Video processing based on reconfigurable logic for low vision aids," presented at the II EOS Topical Meeting on Physilogical Optics, Granada, Spain, September 2004.

[8] F. Vargas-Martin, M. D. Peláez-Coca, E. Ros, J. Díaz. and S. Mota, "Augmented view visual aid based on reconfigurable logic for peripheral vision loss", presented at the Conference and Workshop on Assistive Technologies for Vision and Hearing Impairment, CVHI'2004, Granada, Spain, June 2004.

[9] F. Vargas-Martín, M.D. Peláez-Coca, E. Ros, J. Díaz, S. Mota, "Optoelectronic visual aid base don reconfigurable logia for severe peripheral vision loss rehabilitation," presented at the European Association for Vision and Eye Research (EVER2004), Vilamoura, Portugal, October 2004.

[10] J. Díaz, E. Ros, S. Mota, G. Botella, A. Cañas, and S. Sabatini, "Optical flow for cars overtaking monitor: the rear mirror blind spot problem," presented at 10th. International Conference on Vision in Vehicles (VIV'2003), Granada, Spain, September 2003.

National Conferences

[1] J. Díaz, E. Ros, R. Rodríguez-Gómez and B. del Pino, ""Análisis y diseño de una arquitectura súpersegmentada de altas prestaciones para estimación de movimiento," VI Jornadas sobre Computación Reconfigurable y Aplicaciones (JCRA 2006), Cáceres, September 2006, to be published.

[2] J. Díaz, E. Ros, and S. Mota, "Arquitectura para cómputo de estéreo en imágenes de alta resolución mediante hardware reconfigurable," presented at *Jornadas sobre computación reconfigurable y aplicaciones,* Granada, Spain, September 2005, pp. 167-172.

[3] J. Díaz, E. Ros, S. Mota, E. M. Ortigosa, R. Carrillo, and R. Agis, "Cómputo de flujo óptico en tiempo real mediante FPGAs," presented at *Jornadas sobre computación reconfigurable y* aplicaciones, Barcelona, Spain 13-16 septiembre 2004, pp. 481-488.

[4] E. Ros-Vidal, J. Díaz, S. Mota, and F. Vargas-Martín, "Procesamiento de imágenes en tiempo real para ayuda a pacientes con baja vision", presented at the *XXI Congreso Anual de la Sociedad Española de Ingeniería Biomédica, (CASEIB 2003),* Mérida, Spain, November 2003, pp. 153-156.

9.3. Conclusions

We now summarize the main contributions of the presented work:

- For the local image features extraction we have studied diverse alternatives based on different spatial filters. We have benchmarked the different alternatives with synthetic images to evaluate their accuracy and we have also estimated the computational load of the different approaches. The *Second Order Gaussian derivatives* are highlighted as the approach with the best accuracy versus efficiency trade off.
- We have implemented a high performance computing architecture to extract local image features. We have evaluated the lost of accuracy due to the restricted bit-width at the different stages. We have evaluated the hardware resources requirements and the performance of the system.
- We have designed alternative systems for motion processing with different accuracy versus hardware resources trade offs. Mainly two approaches:
    - A coarse grain pipelined datapath based on the original model of Lucas & Kanade but with IIR temporal filters to reduce the temporal storage resources. This system has also been defined with different levels of parallelism at the critical stages to allow its implementation on devices of very diverse costs and available resources.
    - A fine grain pipelined datapath based on the modified Lucas & Kanade algorithm proposed by Brandt [BRA97] using FIR temporal filters. This system represents a very high performance approach able to process "oversampled sequences" (at frame rates of 95 images per second at VGA resolution) which can highly benefit of the current advanced sensors (able to acquire images at high frame rates). The presented system outperforms in more than an order of magnitude any previous approach found in the literature.
- We have applied the real-time motion computing architecture in the framework of a driver assistant system for lane change monitor during overtaking maneuvers. The results clearly prove the utility of the approach, since it is able

to robustly detect overtaking vehicles at distances in which the driver is able to react.

- We have implemented a high performance stereo computing architecture based on a bio-inspired hardware friendly model. We have evaluated the computing stages which affect most significantly the system accuracy. We have evaluated the hardware resources of the approach and its performance comparing it with previous approaches described in the literature.
- Across the design process of the different stages we have validate a methodology for bit-width study and arithmetic type decision. Our technique allows achieving a good trade-off between resources consumption and system accuracy.
- We have developed massive parallel architectures (based on superscalar and superpipelined units) capable to achieve high computing power. This architectural strategy is quite uncommon in the literature but the results presented at this dissertation show that these datapaths are feasible and valuable alternatives for specific image processing computing devices.

Appendix A
First order gradient model limitation analysis

This appendix reviews 3-D spatio-temporal sampling theory and investigates the effects of motion aliasing (being this, as first approximation, the main limitation of the L&K model). It is discussed in a simplified but insightful way.

L&K model is based on a first order Taylor expansion of the image, equation (3.5), which is correct only if quadratic and further terms can be neglected. This is true for small velocity vectors but errors grow fast when high order terms become significant. Nevertheless, the consideration of how small or large a velocity can be depends on the image structure presented in the neighbourhood of each pixel position. According to the Nyquist-Shannon theorem, the maximum velocity that can be measured in an image without aliasing is limited by the local spatial bandwidth. According to Weber et al. [WEB95], if we consider a sinusoid grating of wavelength λ, we can limit the maximum acceptable displacement given by expression (A.1.a):

$$V_{Max-theoretical} < \lambda/2 \qquad \text{(A.1.a)}$$

$$V_{Max-\exp erimental} < \lambda/2\pi \qquad \text{(A.1.b)}$$

For real images, they show a velocity limit even smaller, about the third part of the theoretical bound, equation (A.1.b) [WEB95]. This equation implicates that the maximum velocity is strongly correlated with the spatial frequencies presented at each image position. The maximum frequency in units of pixels is 0.5 pixels^{-1} which means λ=2 pixels thus, the maximum theoretical speed that can be recovered is less than 1 pixel per frame at the lowest spatial frequency. Thus, using pixels as units, the sampling period is 1 pixel and we can not recover 1 pixel motion of λ=2 sinusoidal gratings. But, this also means that the maximum value of the velocity can be very high for images with spectral contents of large wavelengths. For example, if we consider λ=100 pixels, algorithms could theoretically recover motion up to 49 pixels/frame (first integer value below than 100/2) and experimentally 16 pixels/frame. But note that in order to get this estimation we need to tune the image derivative to the proper frequency in order to get response from the filters

The next consideration is related with the pre-filters and the derivative kernels sizes. Usually the derivative operation is computed as a convolution with Gaussian derivatives which works as band-pass filters with optimal frequency response given by equation (A.2) (also presented in equation (2.13) and replicated here for the sake of clarity) where σ^2 represents the variance and n the derivative order [BLO96].

$$f_0 = \frac{1}{2\pi}\sqrt{n/\sigma^2} \tag{A.2}$$

The utilization of large filters allows us to recover fast motion because it corresponds to large wavelengths but, high frequency image information is lost. Furthermore, the Gaussian derivative bandwidths are approximately constant and asymptotically equal as expressed in (A.3) [KOE87] (also presented in equation (2.12) and replicated here for the sake of clarity). This means that the spatial extension of the Gaussian derivative filters is inversely proportional to the filter bandwidth.

$$\beta \to \frac{1}{4\pi\sqrt{2}\sigma} \tag{A.3}$$

According to equation (A.3), smaller kernels allow us higher flow densities because a larger frequency range is considered (although filters are not optimally tuned for the whole range). Nevertheless, the drawback is that these small spatial resolution filters provide estimations prone to noise, which typically affect more significantly high spatial frequencies. Then, for low noise sequences, the utilization of a small smoothing kernel could be profitable for real images but the final decision of the optimal pre-filters and derivative kernels must be chosen taking into account the SNR of the input images and their spectral properties.

Other important hypothesis is the implicit assumption of constant luminance. Large temporal filters impose a high restriction to the illumination condition that is not always preserved on real scenarios. This motivates the utilization of shorter temporal windows for computing the optical flow. But the drawback of this approach is that higher temporal frequencies are available thus, incrementing the aliasing artefacts.

We decide to utilize first-order-Gaussian-derivative kernels of 5 pixels length which is widely used in most of the implementations and evaluations because it represents a good trade-off between accuracy and computing resources. This strategy implies to adopt two basic assumptions:

1. Low noise. Standard micro-cameras achieve SNR >45 dB in standard environments (not industrial) with homogenous illumination.
2. Only small velocities can be computed, at least for high spatial frequencies. This is a more restrictive assumption. First-order-Gaussian-derivative kernels of 5 pixels have a variance of 1 pixel and a top cut-off frequency of 1.35 rad/pixels (using equation (A.2) and (A.3), $f_0+\beta$) which corresponds to a wavelength of $\lambda=1.48\pi$ pixels and gives us an experimental maximum velocity for such frequency of 0.74 pixels/frame. This highly motivates the use of high frame-rate cameras for motion estimation in real scenarios.

Appendix B
X-Y separable basis set for second derivative of Gaussian

This appendix presents the equations of the Second order Gaussian derivatives G_{xx}, G_{xy} and G_{yy} and their Hilbert transforms H_{xx}, H_{xy}, H_{yx} and H_{yy}. Their shape can be seen in Figure 2.4. Their equations are, for the Gaussian derivatives:

$$G_{xx} = 0.9213 \cdot \left(2x^2 - 1\right) \cdot e^{-\left(x^2+y^2\right)} \tag{B.1.a}$$

$$G_{xy} = 1.843 \cdot x \cdot y \cdot \left(2x^2 - 1\right) \cdot e^{-\left(x^2+y^2\right)} \tag{B.1.b}$$

$$G_{xx} = 0.9213 \cdot \left(2y^2 - 1\right) \cdot e^{-\left(x^2+y^2\right)} \tag{B.1.c}$$

And for their Hilbert Transform:

$$H_{xx} = 0.9780 \cdot \left(-2.254x + x^3\right) \cdot e^{-\left(x^2+y^2\right)} \tag{B.2.a}$$

$$H_{xy} = 0.9780 \cdot \left(-0.7515 + x^2\right) \cdot \left(y\right) \cdot e^{-\left(x^2+y^2\right)} \tag{B.2.b}$$

$$H_{yx} = 0.9780 \cdot \left(-0.7515 + y^2\right) \cdot \left(x\right) \cdot e^{-\left(x^2+y^2\right)} \tag{B.2.c}$$

$$H_{yy} = 0.9780 \cdot \left(-2.254y + y^3\right) \cdot e^{-\left(x^2+y^2\right)} \tag{B.2.d}$$

We use a 9 taps kernel, with a sampling period of 0.67. The peak frequency is f_0=0.21 pixels^{-1} and bandwidth β= 0.1 pixel·s^{-1} as in [FRE91]. Note that thanks to the exponential function properties, the separable implementation of these kernels is straightforward.

The filters steering is done by equations (B.3) (replicated here from equation (5.1) for the sake of clarity). The quadrature filter at orientation θ, represented by $h_\theta = c_\theta + js_\theta$, is calculated using equation (B.3), where c_θ and s_θ are respectively the even and odd components of the filter.

$$\begin{gathered} c_\theta = \cos^2(\theta) G_{xx} - \cos(\theta)\sin(\theta) G_{xy} + \sin^2(\theta) G_{yy} \\ s_\theta = \cos^3(\theta) H_{xx} - 3\cos^2(\theta)\sin(\theta) H_{xy} - 3\cos(\theta)\sin^2(\theta) H_{yx} + \sin^3(\theta) H_{yy} \end{gathered} \tag{B.3}$$

Note that the minus sign in the components of equation (B.3) selects the direction of θ to be counter-clockwise.

Appendix C
Acronyms and abbreviations list

CCD: *Charge-Coupled Device*

CTH: *Confidence Threshold*

DRIVSCO: *Learning to emulate perception action cycles in a driving school scenario*

DSP: *Digital Signal Processor*

ECOVISION: *Artificial Vision Systems based on early cognitive cortical processing*

EDIF: *Electronic Design Interchange Format*

EMB: *embedded memory blocks*

FPGA: *Field Programmable Gate Array*

GPU: *Graphics processing Unit*

HDL: *Hardware Definition Language*

IMO: *Independent Moving Object*

LCDA: *Lane-Change Decision-Aid System*

LIDAR: *LIght Detection And Ranging*

MMU: *Memory Management Unit*

MMX: *MultiMedia eXtensions*

MNC: *Minimum Number of Cycles*

NCP: *Number of Confidence Points*

PDS: *Point-time Disparity per Second*

RMS: *Root Mean Square*

RTL: *Register Transfer Level*

SAD: *Sum of Absolute Differences*

SC: *Subject Car*

SNR: *Signal to Noise Ratio*

SOC: *System-On-a-Chip*

SQNR: *Signal to Quantization Noise Ratio*

SSD: *Sum o f Squared Differences*

SSE: *Streaming SIMD Extensions*

TC: *Target Car*

TTC: *Time-To-Contact*

VHDL: *Very high speed integrated circuit Hardware Description Language.*

VMP: *Virtual Memory Port*

Bibliography

[ADE85] E. Adelson, and J. Bergen, “Spatiotemporal energy models for the perception of motion,” *Journal of the Optical Society of America*, vol. 2, no. 2, pp. 284-299, 1985.

[ADE86] E. H. Adelson and J. R. Bergen, “The extraction of spatiotemporal energy in human and machine vision,” in *Proc.of IEEE Workshop on Motion: Representation and Analysis*, Charleston, SC, 1986, pp. 151–156.

[AGI06] R. Agís, J. Díaz, E. Ros, R. Carrillo, E. M. Ortigosa, **“**Hardware Event-driven Simulation Engine for Spiking Neural Networks**”**, *International Journal of Electronics* (Taylor & Francis Group),2006, to be published.

[ALO92] Y. Aloimonos and Z. Duric, “Active egomotion estimation: A qualitative approach,” presented at European Conference on Computer Vision, Santa Margarita Ligure, May 1992, pp. 497--510.

[ALT06] Altera, DSP Builder, [Online] Available: http://www.altera.com/products/software/products/dsp/dsp-builder.html

[ALV00] L. Alvarez, J. Weickert, J. Sánchez, “Reliable Estimation of Dense Optical Flow Fields with Large Displacements,” *International Journal of Computer Vision*, vol. 39, no.1, pp. 41-56, Aug. 2000.

[ANA87] P. Anandan, "Measuring Visual Motion From Image Sequence," Phd dissertation and COINS Technical Report 87-21, University of Massachusetts, Amherst, 1987.

[ANA89] P. Anandan, “A Computational Framework and an Algorithm for Measurement of Visual Motion,” *International Journal of Computer Vision*, vol. 2, pp. 283-310, 1989.

[AND03] R. J. Andrews, and B. C. Lovell, “Color Optical Flow,” presented at Workshop on Digital Image Computing, Brisbane, 7 February, 2003 vol. 1, no. 1, pp. 135-139.

[AND92] Mats T. Andersson, “Controllable Multidimensional Filters and Models in Low Level Computer Vision,” Ph.D. thesis, Linköping University, 1992.

[APO04] N. Apostoloff, and A. Zelinsky, “Vision In and Out of Vehicles: Integrated Driver and Road Scene Monitoring,” *International Journal of Robotics Research*, vol. 23, no.4-5, pp. 513-538, 2004.

[ARM03] X. Armangué and J. Salvi, “Overall View Regarding Fundamental Matrix Estimation,” *Image and Vision Computing*, vol. 21, no. 2, pp. 205-220, February 2003.

[ASC93] P. Aschwanden and W. Guggenbuhl, “Experimental Results from a Comparative Study on Correlation-Type Registration Algorithms,” *Robust Computer Vision,* Forstner and Ruwiedel, eds., pp. 268-289, Wickmann, 1993.

[ATE06] S. Atev, H. Arumugam, O. Masoud, R. Janardan and N. P. Papanikolopoulos, “A vision-based approach to collision prediction at traffic intersections”, *IEEE Trans. On Intelligent Transportation Systems*, vol. 6, Issue 4, pp. 416- 423, 2006.

[BAI96] A. Bainbridge-Smith and R. G. Lane, “Measuring Confidence in Optical Flow Estimation,” *IEEE Electronic Letters,* vol. 10 pp. 882-884, May 1996.

[BAI97] A. Bainbridge-Smith and R.G. Lane, “Determining Optical Flow Using a Differential Method,” *Image and Vision Computing,* vol. 15, no. 1 pp. 11-22, January 1997.

[BAK04] S. Baker and I. Matthews, “Lucas-Kanade 20 Years On: A Unifying Framework,” *International Journal of Computer Vision*, vol. 56, nº. 3, pp. 221 – 255, March, 2004.

[BAN03] P. Banerjee, D. Bagchi, M. Haldar, A. Nayak, V. Kim, R. Uribe, “Automatic Conversion of Floating Point MATLAB Programs into Fixed Point FPGA Based Hardware Design,” In *Proc. of FPGA Based Custom Computing Machines (FCCM)*, Napa Valley, CA., 2003, pp. 263-264.

[BAR67] H. B. Barlow, C. Blakemore, and J. D. Pettigrew, “The neural mechanism of binocular depth discrimination,” *Journal of Physiology,* vol. 193, no. 2, pp. 327-342, 1967.

[BAR90] J. L. Barron, A. D. Jepson and J. K. Tsotsos, "The Feasibility of Motion and Structure from Noisy Time-Varying Image Velocity Information,” *International Journal of Computer Vision*, vol. 5, issue 3, pp. 239-269, December 1990.

[BAR94] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, “Performance of optical flow techniques,” *International Journal of Computer Vision*, vol. 12, no. 1, pp. 43-77, 1994.

[BEL96] P. N. Belhumeur, “A Bayesian approach to binocular stereopsis,” *International Journal of Computer Vision*, vol. 19 no. 3, pp. 237–260, 1996.

[BEN02] A. Bensrhair, A. Bertozzi, A. Broggi, A. Fascioli, S. Mousset, and G. Toulminet, “Stereo vision-based feature extraction for vehicle detection,” presented at IEEE Intelligent Vehicle Symposium, 2002, vol. 2, 17-21 June 2002, pp. 465 - 470 vol.2

[BEN03] D. Benitez, “Performance of reconfigurable architectures for image-processing applications,” *Journal of Systems Architecture: the euromicro journal*, vol 49, no. 4-6, pp. 193-210, 2003.

[BER06] L.M. Bergasa, J. Nuevo, M. A. Sotelo, R. Barea, and M. E. Lopez, “Real-time system for monitoring driver vigilante*,” IEEE Trans. On Intelligent Transportation Systems*, vol. 7, Issue 1, pp. 63 – 77, March 2006.

[BER98] M. Bertozzi, and A. Broggi, “GOLD: a parallel real-time stereo vision system for generic obstacle and lane detection”, *IEEE Trans. on Image Processing*, vol. 7, issue 1, pp. 62-81, Jan 1998.

[BET96] M. Betke, E. Haritaoglu and L. S. Davis, “Multiple Vehicle Detection and Tracking in Hard Real Time’, University of Maryland, College Park, Technical Report CS-TR-3667, 1996.

[BIG91] J. Bigün, G. H. Granlund, and J. Wiklund. “Multidimensional orientation estimation with applications to texture analysis and optical flow,” *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 13, Issue 8, pp. 775-790, August 1991.

[BLO96] J.A. Bloom, and T.R. Reed, “A Gaussian derivative-based transform,” *IEEE Trans. Image Processing*, vol. 5, no. 3, pp. 551-553, 1996.

[BOB99] A. F. Bobick and S. S. Intille, “ Large occlusion stereo,” *International Journal of Computer Vision*, vol. 33 no.3 pp. 181–200, 1999.

[BOU06] J. -Y. Bouguet, "Camera Calibration Toobox," [Online]. Available: http://www.vision.caltech.edu/bouguetj/calib_doc/

[BOV90] A.C. Bovik, M. Clark, and W.S. Geisler, "Multichannel texture analysis using localized spatial filters," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 12, Issue 1, pp. 55–73, 1990.

[BOY01] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 23, issue 11, pp. 1222–1239, 2001.

[BRA97] J.W. Brandt, "Improved Accuracy in Gradient Based Optical Flow Estimation," *International Journal of Computer Vision,* vol. 25, issue 1, pp. 5-22, October 1997.

[BRO03] M. Z. Brown, D. Burschka, G. D. Hager, "Advances in Computational Stereo," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* vol. 25, issue 8, pp. 993-1008, 2003.

[BRO05] A. Broggi, C. Caraffi, R. I. Fedriga and Paolo Grisleri, "Obstacle Detection with Stereo Vision For Off-Road Vehicle Navigation, "in *Proc.* CVPR'05 – Workshops *Intl. IEEE Wks. on Machine Vision for Intelligent Vehicles,* San Diego, USA, June 2005, p. 65.

[BRO92] L. G. Brown, "A Survey of Image Registration Techniques," *ACM Computing Surveys*, vol. 24, issue 4, pp. 325-376, December 1992.

[BRU05a] A. Bruhn, J. Weickert, and C. Schnörr, "Lucas/Kanade Meets Horn/Schunck: Combining Local and Global Optic Flow Methods," *International Journal of Computer Vision*, vol. 61, no. 3, pp. 211 – 231, February 2005,

[BRU05b] A. Bruhn, J. Weickert, C. Feddern, T. Kohlberger, and C. Schnorr, "Variational optical flow computation in real time," *IEEE Transactions on Image Processing*, vol. 14 no.5, pp. 608-615, May 2005.

[BUR83] P. J. Burt, and E. H. Adelson, "The Laplacian pyramid as a compact image code," *IEEE Trans. Communication,* vol. com-31, no. 4, pp. 532-540, 1983.

[CAM97] T. Camus, "Real-time quantized optical flow," *Journal Real-Time Imaging,* vol. 3, no. 2, pp. 71–86, 1997.

[CEL06a] Celoxica, PixelStreams, [Online] Available: http://www.celoxica.com/products/pxs/default.asp

[CEL06b] Celoxica, DK Design Suite, [Online] Available http://www.celoxica.com/products/dk/default.asp

[CEL06c] Celoxica, Handel-C Language Reference Manual. [Online] Available: www.celoxica.com/techlib/files/CEL-W0410251JJ4-60.pdf

[CEL06d] Celoxica, RC300 board. [Online] Available : http://www.celoxica.com/products/rc300/default.asp

[CEL06e] Celoxica, RC1000-PP board. [Online] Available : http://www.celoxica.com/support/view_category.asp?NodeID=52

[CEL06f] Celoxica, RC200/RC203 board. [Online] Available : http://www.celoxica.com/products/rc203/default.asp

[CEL06g] Celoxica application note AN 68 v1.1, "*Timing Analysis. Timing Analysis and Optimisation of Handel-C Designs for Xilinx Chips*".

[Online]. Available: http://www.celoxica.com/support/view_article.asp?ArticleID=384

[CHA02] M.L. Chang, S. Hauck, “Precis: A Design-Time Precision Analysis Tool,” *IEEE Design & Test of Computers*, vol. 22 no. 4, pp. 349-361, July-August 2005.

[CHA95] K. H. Chang, and W. G. Bliss, “Finite word-length effects of pipelined recursive digital filters,” *IEEE Transactions on Signal Processing*, pp. 1983 –1995, Aug. 1994

[CHE04] Y. Chen and N. Qian, “Coarse-to-fine Disparity Energy Model with both Phase-shift and Position-shift Receptive Field Mechanisms,” *Neural Computation*, vol. 16, issue 8, pp. 1545-1577, 2004.

[CHU04] Chun Te Ewe, Peter Y. K. Cheung, George A. Constantinides, "Dual Fixed-Point: An Efficient Alternative to Floating-Point Computation", Field Programmable Logic and Applications, Lecture Notes in Computer Science, Springer-Verlag, 2004. pp.1-10 (2004).

[COB98] P. Cobos, and F. Monasterio. "FPGA implementation of the Horn & Shunk Optical Flow Algorithm for Motion Detection in real time Images," in *Proc. of the XIII Design of Circuits and Integrated Systems Conference,* Madrid, Spain. Nov. 1998, pp. 616-621.

[COD06] Codesimulink, [Online]. Available: http://polimage.polito.it/groups/codesimulink.html

[CON03] G. Constantinides, “Perturbation Analysis for Word-length Optimization,” in *Proc. IEEE Sym. on Field-Programmable Custom Computing Machines (FCCM’03)*, Napa, CA. Sept. 2003, pp. 81-90.

[COR02] M. V. Correia, and A. C. Campilho, “Real-time implementation of an optical flow algorithm,” in *Proc. international Conference on Pattern Recognition (ICPR2002)*, 2002, p. 40247, vol. 4. pp. 247-250.

[COR97] P. Corke and P. Dunn, "Real-Time Stereopsis Using FPGAs," in *Proc. IEEE TENCON-Speech and Image Technologies for Computing and Telecommunications*, Brisbane, Qld., Australia, 1997, vol. 1, pp. 235-238.

[COZ97] A. Cozzi, B. Crespi, F. Valentinotti, and F. Wörgötter, “Performance of phase-based algorithms for disparity estimation,” *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 9, no. 5-6, pp. 334-340, 1997.

[CRE98] B. Crespi, A. Cozzi, L. Raffo, S. P. Sabatini, “Analog computation for phase-based disparity estimation: continuous and discrete models,” *Machine Vision and Applications*, vol. 11, no. 2, pp. 83-95, 1998

[CVH06] Computer Vision Homepage, [Online]. Available: http://www.cs.cmu.edu/~cil/vision.html.

[CVO06] On-line Compendium of Computer Vision, [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/CVonline/CVentry.htm

[DAG04] E. Dagan, O. Mano, G. P. Stein and A. Shashua, “Forward collision warning with a single camera,” in *IEEE Intelligent Vehicles Symposium*, Parma, Italy, 14-17 June 2004, pp. 37-42.

[DAR03] A. Darabiha, J. Rose, and W. J. MacLean, “Video-Rate Stereo Depth Measurement on Programmable Hardware”, in *Proc 2003 IEEE Computer Society Conference on. Computer Vision and Pattern Recognition*, Madison, Wisconsin, June 2003, vol. 1, pp. 203-210.

[DAR06] A. Darabiha, W. J. MacLean and Jonathan Rose, “Reconfigurable Hardware Implementation of a Phase-Correlation Stereo Algorithm,” *Machine Vision and Applications Journal*, March, 2006.

[DEA91] G. C. DeAngelis, I. Ohzawa, and R. D. Freeman, "Depth is encoded in the visual cortex by a specialized receptive field structure," *Nature* 11, 352(6331) pp. 156-159, 1991.

[DEA98] G. C. DeAngelis, B. G. Cumming, and W. T. Newsome, "Cortical area MT and the perception of stereoscopic depth," *Nature,* vol. 394, issue 6694, pp. 677-680, 1998.

[DEL97] F. Dellaert and C. Thorpe, "Robust car tracking using Kalman filtering and Bayesian templates," in *Proceedings of SPIE: Intelligent Transportation Systems*, Pittsburgh, Pennsylvania, October 1997, vol. 3207, pp. 72-83.

[DES05] X. Desurmont, A. Bastide, C. Chaudy, C. Parisot, J. F. Delaigle and B. Macq, "Image analysis architectures and techniques for intelligent surveillance systems," *IEE Proc. Vision Image and Signal Processing*, vol. 152 issue 2, pp. 224-231, 2005.

[DIA03] J. Díaz , S. Mota, E. Ros and Guillermo Botella, "Neural Competitive Structures for Segmentation Based on Motion Features," *Lecture Notes in Computer Science*, vol. 2686, pp. 710-717, 2003

[DIA04a] J. Díaz, E. Ros, S. Mota, E. M. Ortigosa, R. Carrillo, and R. Agis, "Cómputo de flujo óptico en tiempo real mediante FPGAs," presented at *Jornadas sobre computación reconfigurable y* aplicaciones, Barcelona, Spain 13-16 septiembre 2004, pp. 481-488.

[DIA04b] J. Díaz, E. Ros, S. Mota, E.M. Ortigosa, "Real-time optical flow computation using FPGAs", presented at the Early Cognitive Vision Workshop, Isle of Skye, Scotland, UK, 28 May- 1 June 2004. [Online] Available: http://www.cn.stir.ac.uk/ecovision-ws/schedule.php

[DIA04c] J. Díaz, E. Ros, S. Mota, R. Carrillo, R. Agís, "Real time optical flow processing system," *Lecture Notes in Computer Science (Int. Conf. on Field-programmable logic and its applications (FPL 2004), 30 Ago - 1 Sept, Antwerp, Belgium)*, vol. 3203, pp.617-626, 2004.

[DIA05a] J. Díaz, E. Ros, and S. Mota, "Arquitectura para cómputo de estéreo en imágenes de alta resolución mediante hardware reconfigurable," presented at *Jornadas sobre computación reconfigurable y aplicaciones,* Granada, Spain, September 2005, pp. 167-172.

[DIA05b] J. Díaz, E. Ros, S. Mota, E. M. Ortigosa, and B. del Pino, "High performance stereo computation architecture," in *Proc. of IEEE Int. Conf. on Field Prog. Logic and Applications (FPL'05),* Tampere, Finland, August 2005, pp. 463-468.

[DIA06a] J. Díaz, E. Ros, R. Rodríguez-Gómez and B. del Pino, ""Análisis y diseño de una arquitectura súpersegmentada de altas prestaciones para estimación de movimiento," VI Jornadas sobre Computación Reconfigurable y Aplicaciones (JCRA 2006), Cáceres, September 2006, to be published

[DIA06b] J. Díaz, E. Ros, F. Pelayo, E. M. Ortigosa and S. Mota, "FPGA based real-time optical-flow system," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 2, pp. 274-279, 2006

[DIA06c] J. Díaz, E. Ros, S. Mota and R. Rodriguez-Gomez, "Highly paralellized architecture for image motion estimation, "*Lecture Notes On Computer Science*, Springer-Verlag, 2006, to be published.

[DIA06d] J. Díaz, E. Ros, S. Mota and R. Agis, "Real-time embedded system for rear-view mirror overtaking car monitoring," *Lecture Notes On Computer Science*, Springer-Verlag, 2006, to be published.

[DIA06e] J. Díaz, E. Ros, S. Mota and R. Rodriguez-Gomez, **“**FPGA based architecture for motion sequence extraction**”**, *International Journal of Electronics* (Taylor & Francis Group), 2006 to be published.

[DIA06f] J. Díaz, E. Ros, S. Mota, F. Pelayo, and E. M. Ortigosa, “Sub-pixel motion computing architecture,” *IEE Proc. Vision, Image & Signal Processing*, 2006, to be published

[DIA06g] J. Diaz, E. Ros, A. Rotter, M. Muehlenberg, “Lane-change decision aid system based on motion-driven car tracking,” *IEEE Transactions on Intelligent Transportation Systems*, 2006, submitted for publication.

[DIA06h] J. Díaz, E. Ros, R. Carrillo and A. Prieto, “Real-time system for high-image-resolution disparity,” *IEEE Trans. on Image Processing*, 2006, submitted for publication.

[DIA06i] J. Díaz, K. Pauwels, S. Sabatini and G. Andreani, “Analysis of different filters design approaches for the Drivsco system,” Drivsco Technical report TR2, May 2006.

[DIA06j] J. Díaz, E. Ros, S. P. Sabatini, F. Solari y S. Mota, “A Phase based stereo system-on-a-chip,” *Byo-systems journal,* to be published.

[DIR03] 10 November 2003, Directive 2003/97/EC of the European Parliament and of the Council, *Official Journal of the European Union*, L 25/1-45. “On the approximation of the laws of the Member States relating to the type-approval of devices for indirect vision and of vehicles equipped with these devices, amending Directive 70/156/EEC and repealing Directive 71/127/EEC”. [Online]. Available:

http://europa.eu.int/rapid/pressReleasesAction.do?reference=IP/04/193&format=HTML&aged=0&language=EN&guiLanguage=en

[DRA03] B.A. Draper, J.R. Beveridge, A.P.W. Bohm, C. Ross, and M. Chawathe, “Accelerated image processing on FPGAs”, *IEEE Trasactions on Image Proccesing*, vol. 12, issue 12, pp. 1543-1551, Dec. 2003.

[DRI06] DRIVSCO, project web page, [Online]. Available: http://www.pspc.dibe.unige.it/~drivsco/

[DUM99] C. Dumontier, F. Luthon, and J. P. Charras, “Real-time DSP implementation for MRF-based video motion detection,” *IEEE Transactions on Image Processing*, vol. 8, no.10, pp. 1341-1347, Oct. 1999.

[DUN95] D. Dunn, and W.E. Higgins, “Optimal Gabor Filters for Texture Segmentation,” *IEEE Transactions on Image Processing,* vol.4, no. 7, pp. 947-964, 1995.

[ECO06] ECOVISION Web Page (2006, May 15) [Online]. Available: http://www.pspc.dibe.unige.it/~ecovision/

[FAR99] G. Farnebäck, “Spatial Domain Methods for Orientation and Velocity Estimation,” Lic. Thesis LiU-Tek-Lic-1999:13, Dept. EE, Link¨oping University, SE-58183 Link¨oping, Sweden, March 1999. Thesis No. 755, ISBN 91-7219-441-3.

[FAU01] O. Faugeras, Q.-T. Luong and T. Papadopoulo, *The Geometry of Multiple Images*, MIT Press, 2001.

[FAU93] O. Faugeras, *Three dimensional computer vision, a geometric viewpoint*. MIT Press, 1993.

[FEL01] M. Felsberg, and G. Sommer, "The monogenic signal," *IEEE Trans. on Signal Processing,* vol. 49, no. 12, pp. 3136-3144, December 2001.

[FEL02a] M. Felsberg, “Low-Level Image Processing with the Structure Multivector,” Ph.D. thesis, Institute of Computer Science and Applied Mathematics, Christian-Albrechts-University of Kiel, 2002.

[FEL02b] Felsberg, M., “Disparity from monogenic phase,” presented at 24. DAGM Symposium Mustererkennung, Zürich, 2002, vol. 2449, pp. 248-256

[FEN79] C. Fennema and W. Thompson, “Velocity determination in scenes containing several moving objects,” *Computer Graphics and Image Processing* vol. 9 pp. 301-315, 1979.

[FIC06] Ficosa Digital blind spot detector. [Online]. Available: http://www.ficosa.com/eng/home_noticiaseventos.htm

[FLE90] D. J. Fleet and A. D. Jepson: "Computation of component image velocity from local phase information," *Int. Journal of Computer Vision,* vol. 5, no. 1, pp. 77-104, August 1990.

[FLE91] D. J. Fleet, A. D. Jepson, M. R. M. Jenkin, “Phase-Based Disparity Measurement,” *CVGIP: Image Understanding*, vol. 53, issue 2, pp. 198-210, 1991.

[FLE92] D. J. Fleet, *Measurement of Image Velocity*, Norwell, MA: Engineering and Computer Science, Kluwer Academic Publishers, 1992.

[FLE93] D. J. Fleet, A. D. Jepson, “Stability of phase information,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, issue 12, pp. 1253-1268, 1993.

[FLE94] D. J. Fleet, “Disparity from local weighted phase-correlation,” IEEE Int. Conf. on Systems, Man and Cybernetics,” San Antonio, USA, October 1994, 1, vol. 1, pp. 48-54.

[FLE95] D.J. Fleet and K. Langley, “Recursive filters for optical flow,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*,” vol. 17, no.1, pp. 61-67, 1995.

[FLE96] D. J. Fleet, H. Wagner, and D. J. Heeger, “Neural encoding of binocular disparity: Energy model, position shifts and phase shifts,” *Vision Research*, vol. 36, no. 12, pp. 1839-1857, 1996.

[FOC06] Focus Robotics, Stereo products, [Online]. Available: http://www.focusrobotics.com/products/systems.html.

[FOR02] M. J. Forsell, “Architectural differences of efficient sequential and parallel computers,” *Journal of Systems Architecture: the EUROMICRO Journal*, vol. 47, issue 13, pp. 1017-1041, July 2002.

[FOR04] S. Forstmann, S. Thüring, Y. Kanou, J. Ohya, and A. Schmitt, “Real Time Stereo By Using Dynamic Programming”, presented at the Conference on Computer Vision and Pattern Recognition Workshop, Washington, D.C., June 27 - July 02, 2004,Vol. 3, pp.29.

[FRE91] W. Freeman and E. Adelson. “The design and use of steerable filters,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 13, issue 9, pp. 891-906, September 1991.

[FRO96] T. Frohlinghaus, and J.M. Buhmann, "Real-time phase-based stereo for a mobile robot," in *Proc. 1st Euromicro Workshop on Advanced Mobile Robot*, Kaiserslautern, Germany, 1996, pp. 178-185.

[GAF02] A. Gaffar, O. Mencer, W. Luk, P. Y.K. Cheung, N. Shirazi, “Floating Point Bitwidth Analysis via Automatic Differentiation,” in *Proc. IEEE Int. Conference on Field Programmable Technology,* Hong Kong, Dec. 2002

[GAL98] B. Galvin, B. McCane, K. Novins, D. Mason and S. Mills, "Recovering Motion Fields: An Evaluation of Eight Optical Flow Algorithms," in *Proceedings of the British Machine Vision Converence (BMVC) '98,* September 1998.

[GAO05] J. Gao, A. Kosaka and A. C. Kak, "A multi-Kalman filtering approach for video tracking of human-delineated objects in cluttered environments," *Computer Vision and Image Understanding*, vol. 99, issue 1, pp. 1-57, 2005.

[GAU02] T. Gautama, and M. M. Van Hulle, "A Phase-based Approach to the Estimation of the Optical Flow Field Using Spatial Filtering," *IEEE Trans. Neural Networks,* vol. 13, issue 5, pp. 1127-1136, September 2002.

[GER06] V. Gerdes, Stereo Images with Ground Truth Disparity and Occlusion, [Online] University of Bonn, Germany. Available: http://www-dbv.cs.uni-bonn.de/stereo_data/.

[GET06] Gabor filter introduction from The Computer Vision Lab at GET, University of Paderborn, Department of Electrical Engineering. Online resource available at: http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/TRAPP1/filter.html (Last access, May 06).

[GIA97] P. R. Giaccone and G. A. Jones. ``Feed Forward Estimation of Optical Flow," presented at the *IEEE Conf. on Image Processing and its Applications*, Dublin, July 1997, pp. 204-208

[GON02] Gonzalez and Woods, *Digital Image Processing*. 2nd Edition, Prentice Hall, 2002

[GON05] M. Gong; R. Yang, "Image-gradient-guided real-time stereo on graphics hardware", in *Proc. Fifth International Conference on 3-D Digital Imaging and Modeling,* Ottawa 13-16 June 2005, pp. 548-555.

[GRA78] G. H. Granlund, "In search of a general picture processing operator," *Computer Graphics and Image Processing*, vol. 8, no. 2, pp. 155-173, 1978.

[GRA95] G. H. Granlund and H. Knutsson, *Signal Processing for Computer Vision*. Norwell, MA: Kluwer Academic Publishers, 1995.

[GRA98] R. M. Gray, D. L. Neuhoff, "Quantization," *IEEE Transactions on Information Theory*, vol. 44 Issue 6, pp. 2325 –2383, October 1998.

[GRE00] M. Green, ""How Long Does It Take to Stop?" Methodological Analysis of Driver Perception-Brake times," *Transportation Human Factors*, vol. 2, no.3, pp. 195–216, 2000.

[GRZ90] N. M. Grzywacz and A. L. Yuille, "A model for the estimate of local velocity by cells in the visual cortex," in *Proceedings of the Royal Society of London B* 239, pp. 129–161, 1990.

[HAG92] L. Haglund, "Adaptive Multidimensional Filtering," Ph.D. thesis, nº 284, Linköping University, Sweden, SE-581 83 Linköping, Sweden, 1992.

[HAG94] L. Haglund and D. J. Fleet, "Stable estimation of image orientation," in *Proc. 5th Int. Conf. Computer Vision*, Austin, TX, USA, 1994, pp. 68-72.

[HAR04] R. Hartley, and A. Zisserman, *Multiple View Geometry In Computer Vision,* 2nd Ed., Cambridge University Press, 2004.

[HEE87] D. J. Heeger, "Model for the extraction of image flow," *Journal of the Optical Society of America,* vol. 4, issue 8, pp. 1455–1471, 1987.

[HEE88] D. J. Heeger, "Optical flow using spatio-temporal filters," *Int. Journal of Computer Vision,* vol. 1, no. 4, pp. 279- 302, 1988.

[HEL06a] Hella KGaA Hueck & Co. Lane Change Assistant system. [Online]. Available: http://www.hella.com/produktion/HellaCOM/WebSite/Channels/AutoIndustry/Electronics/DA_Syst

ems/Lane_Change_Assistant.jsp

[HEL06b] Dept. of predevelopment EE-11, Hella KG Hueck & Co., Germany, [Online]. Available: http://www.hella.de

[HEL06c] Hella KGaA Hueck & Co Press text. *Proximity control creates safety*. [Online]. Available: http://www.hella-press.de/search_detail.php?text_id=433&archiv=0&language=e&newdir=eng

[HEM03] Elsayed E. Hemayed, "A Survey of Camera Self-Calibration," in *Proc IEEE Conference on Advanced Video and Signal Based Surveillance*, 2003, p. 351.

[HIT06] Hitachi: Image Processing Camera. (2006, May 15) [Online]. Available: http://www.hitachi.co.jp/en/products/its/product/platform/2004602_12387.html

[HOR81] B. K. P. Horn and B. Schunck, "Determining optical flow". Artificial Intelligence Lab., MIT, Cambridge, MA 02139, U.S.A. Rep. AIM-572

pages 185-204, 1981.

[HOR93] B. K. P. Horn and B. G. Schunck, "Determining Optical Flow - a Retrospective," *Artificial Intelligence* 59(1993). pp. 81-87.

[HSU04] W.-L. Hsu, H. Liao, B.-S Jeng, K.-C. Fan, "Real-time traffic parameter extraction using entropy," *Vision, Image and Signal Processing, IEE Proceedings*, vol. 151, no. 3, pp. 194-202, 2004

[HUB62] D. H. Hubel, and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *Journal of Physiology,* vol. 160, pp. 106-154, 1962.

[HUB88] D. H. Hubel, *Eye, brain and vision*, Scientific American Library, New York, 1988.

[IDT06] IDT SRAM ZBT memories, part number: 71T75602 Datasheet, [Online] Available: www.idt.com.

[JAH93] B. Jähne, "Spatio-Temporal Image Processing: Theory and Scientific Applications," *Lecture notes in computer science,* Springer-Verlag, Berlin, 1993.

[JOH99] A. Johnston, P. W. McOwan and C. Benton, "Robust velocity computation from a biologically motivated model of motion perception" in *Proc. Royal Society B. Biological Sciences* 1999 vol. 266, no. 1418, pp509-518.

[JUN97] S. K. Jung and K. Y. Wohn, "3-D tracking and motion estimation using hierarchical Kalman filter," in *Proc. IEEE Vision, Image and Signal Processing,"* vol. 144, issue 5, pp. 293 – 298, 1997.

[KAN94] T. Kanade, "Development of a Video-Rate Stereo Machine," in *Proc. of the 1994 ARPA Image Understanding Workshop* (IUW'94), November, 1994, Monttey Ca, pp. 549–558.

[KAS87] M. Kass and A. Witkin, "Analyzing oriented patterns," *Computer Vision Graphics and Image Processing*, vol. 37 issue 3, pp. 362-385, March 1987.

[KAT02] T. Kato, Y. Ninomiya and I. Masaki, "An obstacle detection method by fusion of radar and motion stereo," *IEEE Trans. on Intelligent Transportation Systems*, vol. 3, issue 3, pp. 182 - 188, Sept. 2002.

[KEA87] J. K. Kearney, W. B. Thompson, and D. L. Boley, “Optical flow estimation: An error analysis of gradient-based methods with local optimization,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, issue. 2, pp. 229-244, 1987.

[KED98] H. Keding, M. Willems, M. Coors, and H. Meyr, “FRIDGE: a fixed-point design and simulation environment,” In *Proc. of Design Automation Test in Europe*, Paris, 1998, pp. 429 –435.

[KIM05] Y. Kim, A.M. Martinez and A.C. Kak, “Robust Motion Estimation under Varying llumination,” *Image and Vision Computing*, vol. 23, no. 4, pp. 365-375, 2005.

[KNU83] H. Knutsson and G. H. Granlund. “Texture analysis using two-dimensional quadrature filters,” presented at Workshop on Computer Architecture for Pattern Analysis and Image Database Management, Pasadena, pp. 206-213, 1983.

[KOE87] J. J. Koenderink and A. J. van Doom, “Representation of local geometry in the visual system,” *Biological Cybernetics*, vol. 55, no. 6, pp. 367-375, 1987.

[KÖT06] U. Köthe, “Low-level Feature Detection Using the Boundary Tensor,”

in *Visualization and Processing of Tensor Fields, Series on Mathematics and Visualization,* J. Weickert, H. Hagen Eds. Berlin: Springer, 2006, pp. 63-79.

[KOV99] P. Kovesi. (Summer 1999). Image Features From Phase Congruency. *Videre* [Online]. *1(3).* Available: http://mitpress.mit.edu/e-journals/Videre/001/v13.html

[KRÜ02] N. Krüger, and F. Wörgötter, “Multi Modal Estimation of Collinearity and Parallelism in Natural Image Sequences,” *Network: Computation in Neural Systems,* vol. 13, no. 4, pp. 553-576, November 2002.

[KRÜ03] N. Krüger, and M. Felsberg, “A continuous formulation of intrinsic dimension,” In *Proc of the British Machine Vision Conference,* 2003.

[KRÜ04] N. Krüger and M. Felsberg, “An Explicit and Compact Coding of Geometric and Structural Information Applied to Stereo Matching,” *Pattern Recognition Letters,* vol. 25, Issue 8, pp. 849-863, June 2004.

[LAR05] K. Larkin, "Uniform estimation of orientation using local and nonlocal 2-D energy operators," *Opt. Express,* vol. 13, pp. 8097-8121, 2005.

[LIM05] S. Lim, J.G. Apostolopoulos, and A. E. Gamal, “Optical flow estimation using temporally oversampled video,” *IEEE Transactions on Image Processing*, vol. 14 no. 8, pp. 1074-1087, August 2005.

[LIU98] H.C. Liu, T.S. Hong, M. Herman, T. Camus and R. Chellappa, “Accuracy *vs* Efficiency Trade-offs in Optical Flow Algorithms,” *Computer Vision and Image Understanding*, vol.72 (3), pp. 271-286, Dec. 1998.

[LUC81] B.D. Lucas, and T. Kanade: “An Iterative Image Registration Technique with an Application to Stereo Vision,” In *Proc. of the DARPA Image Understanding Workshop*, April 1981, pp. 121-130.

[LUC84] B.D. Lucas, “Generalized Image Matching by the Method of Differences, “ PhD thesis, Dept. of Computer Science, Carnegie-Mellon University, 1984.

[MAL06] A. Mallik, D. Sinha, P. Banerjee, H. Zhou, “Smart Bit-width Allocation for Low Power Optimization in a SystemC based ASIC design Environment”, presented at the Design, Automation and Test in Europe (DATE-06), Munich/Germany, March, 2006.

[MAR05] J.L. Martín, A. Zuloaga, C. Cuadrado, J. Lázaro, and U. Bidarte, "Hardware implementation of optical flow constraint equation using FPGAs," *Computer Vision and Image Understanding*, vol.98, no. 3, pp. 462-490, June 2005.

[MAR06] W. van der Mark and D. M. Gavrila, "Real-time dense stereo for intelligent vehicles," *IEEE Trans. on Intelligent Transportation Systems,* vol. 7, issue 1, pp. 38- 50, March 2006.

[MAY03] S. Maya-Rueda, M. and Arias-Estrada, "FPGA Processor for Real-Time Optical Flow Computation," *Lecture Notes in Computer Science*, vol. 2778, pp. 1103-1016, 2003.

[MCC01] B. McCane, K. Novins, D. Crannitch and B. Galvin, "On Benchmarking Optical Flow," *Computer Vision and Image Understanding*, vol. 84, pp 126–143, 2001.

[MCC06] J. C. McCall and M. M. Trivedi, "Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation," *IEEE Trans. On Intelligent Transportation systems,* vol. 7, no. 1, pp. 20-37, March 2006.

[MEN06] Mentor, Catapult C Synthesis, [Online]. Available: http://www.mentor.com/products/c-based_design/catapult_c_synthesis/index.cfm

[MIC94] M. Michaelis and G. Sommer, "Junction classification by multiple orientation detection," In *Proc. 3rd European Conf. Computer Vision*, Stockholm, May 1994, vol. 1, pp. 101-108.

[MOB06] Mobileye N.V. Blind Spot Detection and Lane Change Assist (BSD/LCA). (2006, May 15) [Online]. Available: http://www.mobileye.com/general.shtml

[MOT04a] S. Mota, E. Ros, J. Díaz, S. Tan, J. Dale and A. Johnston, "Detection and tracking of overtaking cars for driving assistance," presented at the Early Cognitive Vision Workshop*,* Isle of Skye, Scotland, UK, 28 May- 1 June 2004. [Online] Available: http://www.cn.stir.ac.uk/ecovision-ws/schedule.php

[MOT04b] S. Mota, E. Ros, J. Díaz, E. M. Ortigosa, R. Agís and R. Carrillo R, "Real-time visual motion detection of overtaking cars for driving assistance using FPGAs," Lecture Notes in Computer Science vol. 3203, pp. 1158-1161, 2004.

[MOT05] S. Mota, E. Ros, J. Díaz, E. M. Ortigosa, and A. Prieto, "Motion-Driven Segmentation by Competitive Neural Processing," *Neural Processing Letters*, vol.22, no 2, pp. 125-147, 2005.

[MOT06a] S. Mota, E. Ros, J. Diaz, and F. de Toro, "General purpose real-time image segmentation system," *Lecture Notes On Computer Science*, Springer-Verlag, 2006, to be published.

[MOT06b] S. Mota, E. Ros, J. Díaz, R. Agis and F. de Toro, "Bio-inspired motion-based object segmentation" *Lecture Notes On Computer Science*, Springer-Verlag, 2006, to be published.

[MUH02] K. Muhlmann, D. Maier, J. Hesser, and R. Manner, "Calculating Dense Disparity Maps from Color Stereo Images, an Efficient Implementation," *International Journal of Computer Vision*, vol. 47, no. 1-3, pp. 79-88, 2002.

[NAG83] H. Nagel, "Displacement vectors derived from second-order intensity variations in image sequences," Comput. Graphics Image Process. 21, no. 1, pp. 85–117, 1983.

[NAG86] H. Nagel and W. Enkelmann, "An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 8, pp. 565–593, 1986.

[NAG87] H. Nagel, “On the estimation of optical flow: Relations between different approaches and some new results,” *Artificial Intelligence* vol. 33, pp. 299–324, 1987.

[NAY01] A. Nayak, M. Haldar, A. Choudhary, P. Banerjee, “Precision and Error Analysis Of MATLAB Applications During Automated Hardware Synthesis for FPGAs,” In *Proc. of Design Automation and Test in Europe*, Berlin, Mar. 2001.

[NES01] O. Nestares and R. Navarro, "Probabilistic estimation of optical flow in multiple band-pass directional channels," *Image and Vision Computing*, vol. 19, pp. 339-351, 2001.

[NES98] O. Nestares, R. Navarro, J. Portilla and A. Tabernero, “Efficient Spatial-Domain Implementation of a Multiscale Image Representation Based on Gabor Functions,” *Journal of Electronic Imaging*, vol. 7, no. 1, pp. 166-173, 1998.

[NII04] H. Niitsuma, and T. Maruyama, “Real-Time Detection of Moving Objects,” *Lecture Notes in Computer Science, FPL 2004*, vol. 3203, pp. 1153-1157, September 2004.

[OIS03] L. Oisel, E. Memin, L. Morin, and L, F. Galpin, “One-dimensional dense disparity estimation for three-dimensional reconstruction”, *IEEE Transactions on Image Processing*, vol. 12, issue 9, pp. 1107-1119, Sept. 2003.

[OPP81] A. Oppenheim, J. Lim, “The importance of phase in signals,” in *Proc. of the IEEE,* 1981 vol. 69 pp. 529-541.

[ORT06a] E. Ortigosa, A. Cañas, R. Rodriguez, J. Diaz, S. Mota, “Towards an optimal implementation of MLP in FPGA,” *Lecture Notes On Computer Science*, Springer-Verlag, 2006, to be published.

[ORT06b] E. M. Ortigosa, A. Cañas E. Ros, P. M. Ortigosa, S. Mota, J. Díaz, “Hardware description of multi-layer perceptrons with 3 different abstraction levels,” *Microprocessors and Microsystems*, 2006, to be published.

[PER92] P. Perona, “Steerable-scalable kernels for edge detection and junction analysis,” In *Proc. 2nd Europ. Conf. Comput. Vision*, G. Sandini (Ed.), LNCS-Series Vol. 588, Springer-Verlag, pages 3-18, 1992.

[POG84] G. F. Poggio, and T. Poggio, “The Analysis of Stereopsis,” *Annual Review of Neuroscience*, vol. 7, pp. 379-412, 1984.

[POI06] Point Grey Research, Stereo products, [Online]. Available: http://www.ptgrey.com/products/stereo.asp.

[POR02] B. Porr, B. Nürenberg, F. A. Wörgöter, “VLSI-Compatible Computer Vision Algorithm for Stereoscopic Depth Analysis in Real-Time”, *International Journal of Computer Vision*, vol. 49, no. 1, pp. 39–55, 2002.

[RAO91] A. R. Rao and B. G. Schunck. “Computing oriented texture fields,” in *Proc Comp. Vision, Graphics and Image Processing*, Orlando, FL, March 1991, vol. 53, issue 2, pp. 157-185.

[RCM06] RC MODULE: Car Collision Avoidance System (Project). [Online]. Available: http://www.module.ru/products/dsp/ccas.shtml

[ROS04] E. Ros, J. Díaz, S. Mota, “Neural multilayer structure for motion pattern segmentation,” presented at Brain Inspired Cognitive Systems (BICS-2004), Stirling, Scotland (UK), August 2004

[ROS06] E. Ros, J. Díaz, S. Mota, F. Vargas-Martín and M.D. Peláez-Coca, “Real time image processing on a portable aid device for low vision patients, *Lecture Notes On Computer Science*, Springer-Verlag, 2006, to be published.

[ROY04] S. Roy and P. Banerjee, “An Algorithm for Converting Floating Point Computations to Fixed Point Computations in MATLAB based Hardware Design,” In *Proc. of Design Automation Conference (DAC 2004)*, San Diego, Jun. 2004, pp. 484-487

[ROY98] S. Roy and I. J. Cox, “A maximum-flow formulation of the N-camera stereo correspondence problem”, In *Proc. Int. Conf. on Computer Vision*, 1998, pages 492–499.

[SAL02] Salvi, X. Armangué and J. Batlle, “A Comparative Review of Camera Calibrating Methods with Accuracy Evaluation,” *Pattern Recognition*, vol. 35, no. 7, pp. 1617-1635, July 2002.

[SAM06] S. Samavi, S. Shirani and N. Karimi, “Real-Time Processing and Compression of DNA Microarray Images,” *IEEE Transact. On Image Proc*, vol. 15, no. 3, pp. 754-766, March 2006.

[SAN85] J. P. H. van Santen and G. Sperling, "Elaborated Reichardt detectors," *Journal of the Optical Society of America A* vol. 2, pp. 300—321, 1985.

[SAN88] T. Sanger, “Stereo disparity computation using Gabor filters,” *Biological Cybernetics*, vol. 59, no. 6, pp. 405-418, 1988

[SCH02] D. Scharstein and R. Szeliski, “A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms”, *International Journal of Computer Vision*, vol. 47, nos. 1-3 pp. 7-42, April 2002.

[SCH06] D. Scharstein, R. Szeliski, Stereo Vision Research Page, Middlebury College [Online]. Available: http://cat.middlebury.edu/stereo/data.html.

[SCH98] D. Scharstein and R. Szeliski, "Stereo Matching with Non-Linear Diffusion," *Int. Journal of Computer Vision*, vol. 28, no. 2, pp. 155-174, 1998.

[SET01] C. Setchell, , E.L. Dagless, “Vision-based road-traffic monitoring sensor,” *Vision, Image and Signal Processing, IEE Proceedings*, vol. 148, no. 1, pp. 78 – 84, 2001.

[SHE92] A. A. Shen, A. Ghosh, S. Devadas, and K. Keutzer, “On average power dissipation and random pattern testability of CMOS combinational logic networks,” in *Proc. IEEE International Conference on Computer Aided Design*, Santa Clara, California, United States , November 1992, pp. 402-407.

[SHI02] B. E. Shi and K. A. Boahen, “Competitively coupled orientation selective cellular neural networks,” *IEEE Transactions on Circuits and Systems I*, vol 49, no 3, pp388-394, 2002.

[SIM91] E.P. Simoncelli, E.H. Adelson and D.J. Heeger, “Probability distributions of optical flow,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Mauii, Hawaii, June 1991, pp. 310-315.

[SIM93] E. P. Simoncelli, “Distributed analysis and representation of visual motion,” PhD thesis, Dept of E. Eng. Comp, Sci., Massachusetts Institute of Technology, Cambridge, MA., 1993.

[SIM94] E. P. Simoncelli, "Design of multi-dimensional derivatives filters," In *Proc. IEEE International Conf. on Image Processing*, Austin Tx, 1994, pp. 790-794.

[SIM96] E. P. Simoncelli, H. Farid, “Steerable wedge filters for local orientation analysis,” *IEEE Transactions on Image Processing,* vol. 5, issue 9, pp. 1377-1382, 1996.

[SIM98] E. P. Simoncelli and D. J. Heeger, “A model of neuronal responses in visual area MT,” *Vision Research* vol. 38, issue 5, pp. 743-761, 1998.

[SIM99] E P Simoncelli, "Bayesian multi-scale differential optical flow," in *Handbook of Computer Vision and Applications,* volume 2, B Jahne, H Haussecker, and P Geissler, editors, Academic Press, San Diego, April 1999, Chapter 14, pp 397--422.

[SIN90] A. Singh. "An estimation-theoretic framework for image-flow computation," presented at the Third international conference on computer vision, Osaka, Japan, 1990, pp. 168-177.

[SIN92a] A. Singh, and P. Allen, "Image-Flow Computation: An Estimation-Theoretic Framework and a Unified Perspective," *Computer Vision, Graphics and Image Processing*, vol. 56, pp. 152-177, Sept 1992.

[SIN92b] A. Singh, *Optic Flow Computation: A Unified Perspective*, IEEE Computer Society Press, 1992.

[SOB91] P. Sobey and M.V. Srinivasan, "Measurement of optical flow by a generalized gradient scheme," *Journal Optical Society of America A*, vol. 8, no.9, pp. 1488-1498, September 1991.

[SOL01] F. Solari, S. P. Sabatini, G. M. Bisio, "Fast technique for phase-based disparity estimation with no explicit calculation of phase," *Electronics Letters*, vol. 37, issue 23, pp. 1382 -1383, 2001.

[SON98] M. Sonka, R. Boyle, V. Hlavac; Image Processing: Analysis and Machine Vision, PWS, 1998

[SPE94] G. Sperling, C. Chubb, J. A. Solomon and Z-L. Lu, "Full-wave and half-wave processes in second order motion and texture," presented at Wiley (Ciba Foundation Symposium, 184) *Higher-order processing in the visual system*, Chichester, U.K., 1994, pp. 287-303.

[STE00] M. Stephenson, J. Babb and S. Amarasinghe, "Bitwidth Analysis with Application to Silicon Compilation," In *Proc. of the SIGPLAN conference on Programming Language Design and Implementation*, Vancouver, British Columbia, June 2000, pp. 108-120.

[SUT03] G. Sutter, S. López-Buedo, E. Todorovich, E. Boemo, "Logic Depth, Power, and Pipeline Granularity: Some Examples on FPGAs," presented at JCRA, Madrid/Spain, 2003, pp. 201-208.

[SYN06a] Cocentric SystemC Compiler, [Online], Available: http://www.synopsys.com

[SYN06b] Cocentric Fixed Point Designer, [Online], Available: http://www.synopsys.com

[SZE99] R. Szeliski, "Prediction error as a quality metric for motion and stereo," presented at Int. Conf. on Computer Vision, 1999, vol. 2, pages 781–788.

[TER86] D. Terzopoulos, "Regularization of inverse visual problems involving discontinuities," *IEEE Trans. On Pattern Analysis and Machine Intelligence*, vol. 8, no.4, pp. 413–424, 1986.

[TRE84] O. Tretiak and L. Pastor, "Velocity estimation from image sequences with second order differential operators," in *Proc. of Seventh IEEE International Conference on Pattern Recognition*, 1984, pp. 16—29.

[TRU98] E.Trucco and A.Verri, *Introductory Techniques for 3D Computer Vision*, Prentice-Hall, 1998

[TSA05] F.Tsalakanidou, S. Malassiotis, M.G. Strintzis, "Face localization and authentication using color and depth images", *IEEE Trans. on Im. Proc.*, vol. 14, issue 2, pp. 152-168, Feb. 2005.

[URA88] S. Uras, F. Girosi, A. Verri, and V. Torre, "A computational approach to motion perception," *Biological Cybernetics* no. 60, pp. 79–97, 1988.

[VAL02] J. Valls, M. Kuhlmann, and K. K. Parhi, "Evaluation of CORDIC Algorithms for FPGA design," *Journal of VLSI Signal Processing*, vol. 32, no. 3. pp. 207-222, Nov 2002.

[VAL96] F. Valentinotti, G. Dicaro, B. Crespi, "Real-Time Parallel Computation of Disparity and Optical-Flow Using Phase Difference," *Machine Vision and Applications* vol. 9, no. 3, pp. 87-96, 1996.

[VAR04] F. Vargas-Martin, M. D. Peláez-Coca, E. Ros, Díaz J. and S. Mota, "Optoelectronic Visual Aid Based on Reconfigurable Logic for Severe Peripheral Vision Loss Rehabilitation", *Ophthalmic Research*, vol. 36, no. 1, pp. 60, 2004

[VAR05] F. Vargas-Martín, M. D. Peláez-Coca, E. Ros, S. Mota and J. Díaz, "A general real-time video processing unit for low vision," presented at Vision 2005*,* London, England, April 2005

[VER87] A Verri, and T Poggio, "Against quantitative optical flow," in *Proc. of the First International Conference on Computer Vision*, London, 1987, pp. 171-180.

[VID06] Videre Design, Stereo on a chip, [Online]. Available: http://www.videredesign.com/stereo_on_a_chip.htm

[VIS06] The Vislist, ftp Server, Optical flow sequences, [Online]. Available: ftp://ftp.vislist.com/SHAREWARE/CODE/OPTICAL-FLOW/

[VOL06] Volvo BLIS system. [Online]. Available: http://www.mynrma.com.au/blis.asp

[VOL59] J. Volder, "The CORDIC Trigonometric Computing Technique," *IRE Trans. Electronic Computing*, vol. EC-8, pp. 330-334, Sept. 1959.

[WAT83] A. B. Watson and A. J. Ahumada, "A look at motion in the frequency domain," J.K. Tsosos, (eds.), *Motion: Perception and representation*, pp. 1–10. NewYork, 1983.

[WAT85] A. Watson and A. Ahumada, "Model of human visual-motion sensing," *Journal of the Optical Society of America A,* vol. 2, issue 2, pp. 322--342, 1985.

[WAX85] A. Waxman, and K. Wohn, "Contour evolution, neigh- bourhood deformation and global image flow: Planar surfaces in motion" *International Journal of Robotics Research*, vol. 4, pp. 95-108, 1985.

[WAX88] A. M. Waxman, J. Wu, and F. Bergholm, "Convected activation profiles and receptive fields for real measurements of short range visual motion," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 1988, pages 717--723.

[WEB95] J. Weber and J. Malik, "Robust computation of optical flow in a multi-scale differential framework," *International Journal of Computer Vision*, vol. 14, no.1, pp. 67-81, 1995.

[WEB97] J. Weber and J. Malik, "Rigid body segmentation and shape description from dense optical flow under weak perspective," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 19, no. 2, pp. 139-143, 1997.

[WES94] C.-F. Westin "A Tensor Framework for Multidimensional Signal Processing," PhD thesis, Linköping University, 1994.

[WIK06] Wikipedia online encyclopaedia. Orientation, [Online]. Available: http://en.wikipedia.org/wiki/Orientation_(computer_vision). Last access May 2006.

[WOO97] J. Woodfill, B. Von Herzen, "Real-time stereo vision on the PARTS reconfigurable computer", presented at the Conf. IEEE Symposium on Field-Programmable Custom Computing Machines, Napa, April 1997, pp. 201.

[XIL06a] Xilinx, System Generator for DSP, [Online] Available: http://www.xilinx.com/ise/optional_prod/system_generator.htm

[XIL06b] Xilinx, Coregenerator, [Online] Available: http://www.xilinx.com/products/design_tools/logic_design/design_entry/coregenerator.htm

[XIL06c] Xilinx Virtex II FPGAs, [Online]. Available:

http://www.xilinx.com/products/silicon_solutions

/fpgas/virtex/virtex_ii_platform_fpgas/resources/index.htm.

[XIL06d] Xilinx, ISE Foundation , [Online] Available: http://www.xilinx.com/ise/logic_design_prod/foundation.htm

[YAN03] H. S. Yan and T. Tjahjadi, "Optical Flow Estimation and Segmentation through Surface Fitting and Robust Statistics," in *Proc. 2003 IEEE International Conference on Systems, Man and Cybernetics*, Washington, DC USA, October 2003, pp. 1390-1395.

[ZAB94] R. Zabih and J. Woodfill, "Non-parametric local transforms for computing visual correspondence," *Lecture Notes in Computer Science*, vol. 801, (T*hird European conference on Computer Vision (Vol II)),* Stockholm, Sweden, 1994 Jan-Olof Eklundh, editor, volume 801 of, pages 151--158. Springer-Verlag, 1994.

[ZHA98] Z. Zhang, "Determining the Epipolar Geometry and its Uncertainty: A Review," *International Journal of Computer Vision*, vol. 27, Issue 2, pp. 161–195, March 1998.

[ZIY00] Lu Ziyi, B.E. Shi, "Subpixel resolution binocular visual tracking using analog VLSI vision sensors," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 47, issue 12, pp. 1468-1475, Dec. 2000.