UNIVERSIDAD DE GRANADA

Escuela Técnica Superior de Ingenierías
Informática y de Telecomunicación

Departamento de Ciencias de la Computación
e Inteligencia Artificial

Un modelo para el desarrollo de sistemas de detección de situaciones de riesgo capaces de integrar información de fuentes heterogéneas. Aplicaciones.

María Dolores Ruiz Lozano

Granada, Octubre 2010

Editor: Editorial de la Universidad de Granada
Autor: María Dolores Ruiz Lozano
D.L.: GR 2004-2011
ISBN: 978-84-694-1180-3

UNIVERSIDAD DE GRANADA

Escuela Técnica Superior de Ingenierías
Informática y de Telecomunicación

Departamento de Ciencias de la Computación
e Inteligencia Artificial

Un modelo para el desarrollo de sistemas de detección de situaciones de riesgo capaces de integrar información de fuentes heterogéneas. Aplicaciones.

Memoria de tesis presentada por

María Dolores Ruiz Lozano

para optar al grado de Doctor en Ingeniería Informática

por la Universidad de Granada

Directores de la Tesis:

Dr. Juan Luis Castro Peña
Dr. Miguel Delgado Calvo-Flores

Granada, Octubre 2010

La memoria de tesis titulada '**Un modelo para el desarrollo de sistemas de detección de situaciones de riesgo capaces de integrar información de fuentes heterogéneas. Aplicaciones.**', que presenta **María Dolores Ruiz Lozano** para optar al grado de Doctor en Informática, ha sido realizada en el **Departamento de Ciencias de la Computación e Inteligencia Artificial** de la Universidad de Granada bajo la dirección de los doctores **Juan Luis Castro Peña** y **Miguel Delgado Calvo-Flores**.

María Dolores Ruiz Lozano
Doctorando

Juan Luis Castro Peña
Director

Miguel Delgado Calvo-Flores
Director

A Javi

La inteligencia consiste no sólo en el conocimiento, sino también en la destreza de aplicar los conocimientos en la práctica.

– ARISTÓTELES

Agradecimientos

En primer lugar, quiero mostrarle mi agradecimiento a los doctores D. Miguel Delgado Calvo-Flores y D. Juan Luis Castro Peña por su excelente labor a cargo de la dirección de esta Tesis. Gracias por la confianza depositada en mí que ha hecho posible la realización de este trabajo. Ha sido, y es, un honor trabajar con vosotros.

A D. Antonio Albiol, D. Jorge Moragues y D. Luis Vergara, doctores de la Universidad Politécnica de Valencia, y, a D. Javier Albusac y D. David Vallejo, doctores de la Universidad de Castilla La Mancha, por la colaboración prestada y el tiempo invertido en este estudio.

Al Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada por las facilidades prestadas para la realización de este proyecto.

También me gustaría mencionar a mis compañeros del Departamento de Ciencias de la Computación e Inteligencia Artificial, especialmente a Clara, Migue, Sergio, María y Aida, por su apoyo profesional y su amistad.

A Paco, por haber leído mi trabajo y haber aportado su granito de arena.

A mi familia y amigos, por su apoyo y comprensión durante la elaboración de esta Tesis y siempre.

Mis agradecimientos también van dirigidos al Proyecto Hesperia [61] (proyecto CENIT-2005), que ha financiado este trabajo y me ha brindado la oportunidad de trabajar en el área de vigilancia inteligente, de gran interés para mí.

En último lugar, y no menos importante, quería agradecer especialmente el gran apoyo recibido de Javier, compañero de trabajo, de penas y de alegrías, compañero de vida. Gracias por cada uno de los minutos que has dedicado a este estudio, por apoyarme siempre en todo, por contagiarme la ganas de aprender y mejorarme en la vida profesional y personal. Gracias por tu confianza, amor y apoyo.

A todos ellos, muchas gracias.

Resumen

En la actualidad, existe una importante demanda de seguridad, tanto en espacios públicos como en privados, para proteger a las infraestructuras y a las personas de ciertos riesgos a los que estamos expuestos (intrusiones, robos, terrorismo, accidentes de tráfico...).

Los sistemas de vigilancia han sido, y son, ampliamente usados para mantener la seguridad en entornos monitorizados. Sin embargo, la vigilancia tradicional, que consiste en publicar, mediante monitores, el vídeo que recogen las cámaras de vigilancia, implica la atención constante de un operador humano para que no pase desapercibido cualquier peligro existente.

De este modo, surgen nuevos sistemas de seguridad inteligentes para paliar la carga del vigilante y evitar la presencia de riesgos no identificados por la falta de atención o la presencia de fatiga en el operador. Estos sistemas tienen como objetivo analizar e interpretar de forma automática la escena y llamar la atención del personal únicamente en los momentos que sea necesario, con el fin de avisarles de ciertos riesgos o peligros en tiempo real.

En los últimos años, se han llevado a cabo grandes avances en la vigilancia inteligente, pero aún existen muchos aspectos a mejorar en los sistemas de seguridad. Muchos de ellos se han diseñado para ser aplicados en entornos muy concretos y cumplir un fin específico, lo que repercute en que no puedan ser aplicados en otros espacios y que su escalabilidad sea baja.

La Inteligencia Artificial y las técnicas de Visión por Computador están jugando un papel importante en la vigilancia inteligente. Un buen ejemplo de estas novedades tecnológicas es el análisis de contenido de vídeo, presente en la mayoría de los sistemas de seguridad inteligentes. Además, existen otras tecnologías, menos explotadas, que permiten obtener información que el análisis de vídeo no puede obtener. En estos contextos destaca el análisis del audio, y también, la presencia de sensores que aportan otra perspectiva de los acontecimientos de una escena.

A pesar de todo, es difícil encontrar sistemas que abarquen de forma conjunta el análisis de vídeo, audio y sensores y que puedan ser aplicados en entornos diferentes. Este hecho se debe a la dificultad para trabajar con información heterogénea. Aún así, si todas esas tecnologías se unen, y se lleva a cabo un proceso de integración de la información obtenida, podemos construir un sistema de vigilancia mucho más potente que disponga de todos los datos unificados en la etapa de toma de decisiones.

Por ello, nace el propósito de crear una tecnología inteligente que sea capaz de integrar toda la información que se pueda recoger de diferentes fuentes y que realice un análisis conjunto de los distintos eventos que ocurren en un espacio protegido, dando lugar a una herramienta escalable, flexible y potente que contribuya a mantener la seguridad de los espacios monitorizados y permita ayudar a los operadores en el proceso de vigilancia, con el objetivo de avisar de riesgos o peligros, en tiempo real.

En este contexto y en esta tesis, presentamos un *Modelo* que será la base para el desarrollo de sistemas de seguridad inteligentes, que se caractericen por ser *flexibles* ante la inclusión de otras fuentes de información sobre el entorno, *escalables* para introducir nuevas funcionalidades y *portables* a cualquier escenario de estudio.

Como ejemplo de *aplicaciones* del Modelo propuesto, presentamos tres sistemas inteligentes para la detección de las siguientes situaciones de alerta: la *presencia de riesgo de atropello*, la *identificación de peligro por niños en zonas de tráfico* y la *detección de intrusiones*.

Índice

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

En este capítulo se exponen las distintas circunstancias que han motivado la presente tesis y se plantean los objetivos a abordar.

En la sección 1.1 presentamos, con breves pinceladas, la situación actual sobre la demanda de vigilancia inteligente y las carencias relevantes que presentan hoy día algunas de las soluciones ya propuestas en este área. También conoceremos los campos más destacados que demandan este tipo de vigilancia y que se convierten en los principales ámbitos de aplicación de esta línea de investigación.

En la sección 1.2, definimos los objetivos de este estudio, fruto del interés en cubrir algunas carencias existentes en la vigilancia inteligente actual.

Finalmente, el capítulo concluye con una breve descripción sobre la estructura de la memoria de esta tesis.

1.1. Motivación

1.1.1. Demanda de Vigilancia Inteligente

El interés por cubrir la seguridad en espacios públicos se ha intensificado recientemente en pro del bienestar social. Existe una

tendencia creciente hacia la globalización de la sociedad. Este hecho, junto con el aumento de la accesibilidad de las personas a los espacios públicos y el derecho de su calidad de vida en las infraestructuras, ha aumentado la necesidad de proteger a los ciudadanos ante posibles amenazas. De este modo, la vigilancia se hace cada día más necesaria para así ofrecer una mayor seguridad a las infraestructuras y a las personas, tanto en espacios públicos como en privados.

En las últimas décadas, se han desarrollado diversos sistemas de monitorización para otorgar seguridad a un espacio protegido. El comienzo de este tipo de tecnologías lo constituyen los sistemas de Circuito Cerrado de Televisión, más conocidos por su acrónimo CCTV, *(Closed Circuit Television)*. Estos sistemas integran una tecnología de vídeo vigilancia visual diseñada para supervisar una diversidad de ambientes y actividades. El circuito se compone, simplemente, por una o más cámaras conectadas a uno o más monitores o televisores, que reproducen las imágenes capturadas. Aunque, para mejorar el sistema, las cámaras se suelen conectar directamente o enlazar por red a otros componentes como vídeos u ordenadores.

Sin embargo, si se desea tener una vigilancia plena, los sistemas CCTV presentan un gran inconveniente: es necesario que exista un controlador humano visualizando las pantallas constantemente. Este hecho no siempre se puede controlar, ya que la presencia del operador no está garantizada permanentemente y cabe la posibilidad de que surja un riesgo que no sea observado. Incluso en el caso de que una persona permanezca constantemente visualizando los vídeos, existen estudios [137] que demuestran que tras 20 minutos observando imágenes, el rendimiento del vigilante disminuye considerablemente, por lo que aumenta la posibilidad de que se den situaciones de interés que no sean detectadas.

Las dificultades o carencias que presenta la vídeo-vigilancia tradicional demandan nuevas herramientas que ayuden al personal en los procesos de monitorización. Surge así la necesidad de desarrollar nuevas tecnologías, más potentes e inteligentes, que sean capaces de detectar riesgos de forma automática y disminuyan la carga del operador humano.

Como veremos en el capítulo 2 de esta memoria, las nuevas tecnologías han robustecido los sistemas de seguridad. Además, en estos últimos años, la Inteligencia Artificial ha jugado un papel muy importante en este área. Un buen ejemplo de estas novedades tecnológicas es el análisis del contenido del vídeo o vídeo inteligente, que se hace cada vez más presente en los sistemas de vigilancia. Este avance científico-tecnológico ha permitido que los sistemas CCTV evolucionen a sistemas semi-automáticos, cuyo objetivo es la detección de determinados eventos en tiempo real usando algoritmos de procesamiento de imágenes e Inteligencia Artificial.

La Vigilancia Inteligente es un campo que está desarrollándose actualmente tanto en el área de investigación como en el sector comercial; aunque en este último, de forma menos intrusiva. La comunidad científica está llevando a cabo una gran labor dentro de este campo. Fruto de este trabajo, en la literatura específica, podemos encontrar un gran abanico de Sistemas de Vídeo Vigilancia Inteligente. Entre ellos, algunos de los más destacados son los mostrados en [17, 18, 32, 121, 135]. Otros ejemplos destacados de Sistemas de Seguridad que podemos encontrar en el mundo comercial son Detect [64], Sivicam [70] y Gotcha [66].

Dentro de los Sistemas de Vigilancia Visuales, se pueden diferenciar varias etapas (o subproblemas): detección y seguimiento de objetos en movimiento [13, 23, 79], clasificación de dichos objetos [48, 52, 95] y análisis del comportamiento los mismos [21, 23, 138].

Actualmente, las primeras etapas están bastante explotadas, por lo que existen una gran cantidad de trabajos en el mundo académico que hacen referencia a dichas fases.

En el estudio realizado sobre el estado del arte, descrito en el capitulo 2, veremos cómo *la mayoría de los sistemas de monitorización visual otorgan casi toda la importancia a la creación y prueba de algoritmos y técnicas de procesamiento de imágenes (visión artificial)*, con el fin de realizar una detección, una clasificación y un seguimiento de objetos. En cambio, son más escasos los trabajos que desarrollan aplicaciones que intentan incorporar inteligencia dándole más importancia a la etapa de

análisis del comportamiento de objetos, una vez que han sido detectados.

Es necesario otorgar más énfasis al análisis de los acontecimientos de un entorno monitorizado, ya que es una etapa de vital importancia dentro de un sistema de seguridad inteligente. También es cierto que este tema está siendo puntero en estos últimos años y se está despertando un gran interés por profundizar en este aspecto. Esto puede verse reflejado en que el objetivo de la última generación de sistemas de seguridad inteligentes es realizar una buena interpretación de la escena, analizar los eventos acontecidos en el área monitorizada y alertar al operador sólo en el momento que sea necesario. En definitiva, creando herramientas de ayuda y soporte al vigilante en su trabajo.

Recientemente, la mayoría de los desarrollos realizados dentro de esta última etapa *se limitan a detectar situaciones simples o muy concretas* [47, 81]. Por consiguiente, surge la demanda de desarrollar sistemas más potentes, capaces de detectar situaciones ricas y complejas.

Hasta ahora, sólo hemos hablado de sistemas de vigilancia visuales (cuya única fuente de información es el vídeo). Sin embargo, hoy día, se desarrollan otros estudios que analizan audio con el fin de detectar sonidos cruciales en un entorno de seguridad [31, 124]. Aun así, *es muy difícil encontrar sistemas de vídeo-vigilancia que realicen un análisis de audio* que complemente la información visual. Esto es debido a que los datos obtenidos tras un análisis de audio tienen un contexto diferente a los obtenidos tras un análisis de vídeo. El hecho de tener información tan diferente, hace que el proceso de integración de audio y vídeo, para llevar un procesamiento conjunto, no sea fácil.

Si además, el análisis de una escena es complementado por la información proveniente de otros sensores (detectores de movimiento, detectores de humo... o, por ejemplo, etiquetas de un mundo marcado), el potencial del Sistema de Seguridad crece. Sin embargo, como se ha dicho anteriormente, la integración de información heterogénea es un proceso tedioso, lo que implica que no existan muchas herramientas que usen diversas fuentes de información (vídeo, audio, u otros sensores).

Por otro lado, un aspecto a tener en cuenta en los sistemas de vigilancia es el *tratamiento de la incertidumbre*. En muchos casos, dada

la naturaleza del problema, los sistemas que analizan vídeo o audio obtienen del entorno información imprecisa o con vaguedad. Este hecho se debe a que una detección exacta y rotunda de un tipo de evento es difícil de obtener, ya que los sistemas siempre están condicionados por 'ruidos' que impiden ver u oír con claridad. Además, las técnicas desarrolladas, por el momento, no han conseguido esa exactitud en la detección de eventos. Este hecho podría compararse en la vida real a encontrarnos ante un vigilante de seguridad "un poco ciego y un poco sordo".

Una carencia de las aplicaciones de vigilancia actuales es su *falta de escalabilidad*. La mayoría de ellas son sensibles a tres aspectos:

- *Cambio del escenario de estudio*. Muchos sistemas de seguridad están diseñados para proteger un entorno específico y no funcionarían si el área monitorizada cambia, o bien, el trabajo y el tiempo que llevaría adaptarlos al nuevo espacio protegido hacen que sea inviable la escalabilidad en este aspecto.

- *Variación del número y del tipo de sensores que monitorizan el entorno*. Muchas herramientas de vigilancia suelen ser poco flexibles cuando se modifica el número de sensores que alimentan el sistema. También son poco escalables si se añade un nuevo tipo de fuente de información. Este último es el caso de los sistemas que han sido exclusivamente diseñados para procesar el contenido del vídeo.

- *Modificación o ampliación de las situaciones de estudio*. Otro aspecto poco escalable en los sistemas de monitorización actuales se origina al estudiar una situación de riesgo diferente a la actual. En este sentido, muchos de ellos no están diseñados para ampliar el campo de análisis del sistema.

Otro punto débil que existe en los sistemas complejos, tanto los que estudian situaciones de riesgo complejas, como los que adquieren información de diversos sensores, es *el elevado tiempo que emplean en analizar e interpretar la escena*, lo que conduce al fracaso del sistema. El tiempo empleado en detectar un riesgo es vital en un entorno de

seguridad. Detectar un evento cuando ya ha pasado no es útil en algunas situaciones. En estos casos, es importante predecir el riesgo con antelación, y así evitar una situación peligrosa. Para estos propósitos, es necesario crear sistemas que analicen situaciones y ofrezcan resultados en tiempo real.

A modo de resumen, en la actualidad existen muchos edificios o espacios físicos que cuentan con un entorno provisto de cámaras de vigilancia, sensores o micrófonos. El número y tipo de estos dispositivos de monitorización es extenso. Sin embargo, hoy día, es muy difícil encontrar sistemas de seguridad inteligentes que usen información heterogénea proveniente diversas fuentes y que puedan ser aplicables sobre cualquier área.

Por ello, nace el propósito de esta tesis: diseñar una tecnología inteligente que integre toda la información que se pueda recoger de diferentes fuentes y realice un análisis conjunto de los distintos eventos que ocurren en un espacio protegido, dando lugar a una herramienta escalable, flexible y potente que contribuya a mantener la seguridad de los espacios monitorizados y permita ayudar a los operadores en el proceso de vigilancia, con el objetivo evitar o alertar situaciones desagradables o peligrosas.

1.1.2. Demanda de seguridad en distintos campos de aplicación

Los sistemas de vigilancia inteligentes han sido aplicados a diferentes ámbitos prácticos. Los campos de aplicación más destacados son: la monitorización del tráfico [32, 97], la protección del transporte público [43] y la vídeo-vigilancia de interiores o en el hogar [94].

Hoy en día, los accidentes de tráfico son una de las principales causas de muerte en Europa y en muchos de los países desarrollados. Este hecho ha despertado un gran interés en el diseño de sistemas de seguridad dentro de este ámbito. Esto puede verse reflejado en la gran cantidad de investigaciones llevadas a cabo sobre la monitorización del tráfico. En general, este tipo de sistemas inteligentes persiguen la detección, clasificación y seguimiento de personas y vehículos, junto con

la identificación de algunas situaciones anómalas dentro de este área: movimiento lento, parada, conducción al revés, aglomeraciones...

En estos últimos años se ha creado una gran expectación sobre la protección del viandante, ya que muchos peatones mueren o resultan heridos en accidentes de tráfico diariamente. Gracias a un estudio europeo *EuroTest* "Programa de evaluación de pasos de peatones" [62,65] podemos conocer un análisis con datos estadísticos sobre los heridos graves y las muertes de viandantes registradas tanto en carreteras como en zonas urbanas. Este estudio lleva realizándose cada año desde el 2005 y analiza y compara 310 pasos de peatones de 31 ciudades europeas importantes. España es uno de los países que está a la cabeza de muertes de viandantes. Actualmente, uno de cada cuatro fallecidos en accidente de tráfico en España es un peatón.

En la actualidad existen algunas carencias que deben ser solventadas en el área de tráfico para incrementar la seguridad del ciudadano que circula por la vía. Y a su vez, *surge la demanda de crear sistemas de vigilancia inteligentes que puedan ayudar a disminuir los accidentes de personas atropelladas*.

En este contexto, y con el fin de aumentar la seguridad de los transeúntes, surge la motivación para desarrollar una solución que detecte la presencia de riesgo de atropello cuando una persona se encuentre en peligro ante un vehículo. En este caso, nuestro objetivo se centrará en predecir la posibilidad de un accidente con suficiente antelación, con la intención de prevenirlo y evitarlo.

Otro hecho que *demanda seguridad* en zonas donde existe tránsito de vehículos, es la presencia de niños que no están bajo la protección de personas adultas y que pueden moverse según sus instintos. Un niño puede realizar una imprudencia sin ser consciente: correr o jugar cerca de vehículos en movimiento, cruzar una vía por un lugar no adecuado o en un momento inoportuno...

Existen entornos, como colegios, zonas de juegos o residenciales, que disponen de una señal de tráfico que avisa del peligro por la proximidad de un lugar frecuentado por niños. Sin embargo, hay otras áreas urbanas que también pueden ser visitadas por niños y que no

disponen de ese tipo de aviso. En este caso, los niños se convierten en puntos vulnerables que pueden originar un accidente de tráfico o ser víctimas del mismo. Por ese motivo, desarrollar un sistema que identifique este tipo de eventos, puede resolver muchas *situaciones de riesgo ocasionadas por niños en una zona de tráfico*. Es importante destacar, que en estos casos debemos alertar tanto a los niños como a los conductores.

Como se ha dicho anteriormente, no solo la monitorización del tráfico es un campo de aplicación clave en sistemas de seguridad. Por otro lado, aparte de enfocar la seguridad para proteger la vida de las personas, existen otros aspectos donde la vigilancia es útil.

En concreto, la protección de determinadas infraestructuras (museos, edificios políticos, casas privadas, comercios...) es un factor de gran preocupación para un amplio sector de la población.

La intrusión de individuos en espacios físicos donde está restringido el acceso supone un riesgo tanto para los edificios como para las personas. Muchas de estas intrusiones acaban en robos, asaltos o incluso agresiones a personas.

Cuando se busca la protección y la seguridad de una infraestructura y de los seres que residen en ella, la sociedad recurre a medios tradicionales para alertar de posibles intrusiones. Habitualmente se contratan guardias de seguridad o se instalan cámaras de vigilancia; y en raros casos, también, se implantan sensores de movimiento que activan alarmas sonoras. De esta manera, surge la motivación de desarrollar una herramienta inteligente que use diversas fuentes de información (cámaras, sensores, micrófonos) y sea capaz de *detectar intrusiones* en cualquier escenario de estudio, para así poder alertar de la presencia de aquellas circunstancias que puedan desembocar en un robo, un asalto personal, etc.

1.2. Objetivos

Con el fin paliar las limitaciones existentes en el área de Vigilancia Inteligente y ofrecer mejoras ante los problemas mostrados en la sección

1.1 nos proponemos alcanzar los siguientes objetivos:

1. **Diseñar una tecnología que permita el desarrollo de sistemas de vigilancia inteligentes** que se caractericen por ser escalables, flexibles y portables a cualquier entorno.

   La tecnología que proponemos, se alimentará de la información procedente del nivel sensorial y ofrecerá como salida la identificación de situaciones de interés.

   Hemos de enfatizar, que nuestro estudio pretende avanzar en aspectos de integración de información heterogénea y análisis de situaciones de alerta, centrándonos más en la etapa de razonamiento sobre los acontecimientos de una escena. Por ello, supondremos que las salidas de la capa sensorial son resultados de un análisis cognitivo preliminar de señales de audio, vídeo y otros sensores.

   Por lo tanto, no pretendemos hacer una detección, clasificación y seguimiento básico de objetos a partir del vídeo, ni una identificación de eventos de sonido; sino que partiremos de los datos obtenidos de sistemas de extracción de conocimiento que realizan dichos análisis preliminares. Como veremos en el estado del arte, estas etapas han sido estudiadas profundamente en otras investigaciones. Por ello, los esfuerzos de esta tesis se centran en avanzar en otros aspectos posteriores, concretamente en el análisis de situaciones más abstractas a partir de la integración de información procedente de diversas fuentes.

   - Se pretende crear un **Modelo que permita representar cualquier tipo de información sobre un entorno vigilado**. Para ello, se describirán *ontologías* que permitan definir de forma homogénea el conocimiento obtenido por diversas fuentes (audio, vídeo, y otros sensores).
   - Dotaremos a la tecnología con **mecanismos de extracción de nuevo conocimiento y procesos de *Inteligencia Artificial*** que consulten, a través de dicha Ontología, el conocimiento del entorno. Este razonamiento será diseñado con el fin de detectar anomalías o situaciones de riesgo que no son triviales.

- Queremos que los sistemas desarrollados con dicha tecnología sean robustos ante información imprecisa o borrosa. Para ello, se llevará a cabo el uso de métodos formales que permitan tratar la incertidumbre y la vaguedad. De esta forma, la *lógica difusa* representará un papel fundamental en la etapa de integración de información y de análisis del conocimiento.
- Así mismo, nos proponemos crear una **arquitectura basada en capas y componentes**, que permita que el sistema final sea escalable y flexible, tanto en la introducción de nuevas fuentes de información, como en la ampliación de nuevas situaciones de estudio.

2. **Aplicar el Modelo propuesto para la creación de sistemas de seguridad inteligentes**, desarrollando sistemas de vigilancia que analicen distintas situaciones reales. Este objetivo consiste en la implementación y puesta en práctica del modelo teórico.

   Concretamente, resolveremos tres tipos de riesgos que se caracterizan por ser situaciones contextualmente ricas y complejas, y además, de gran interés social. Nos referimos a:

   a) La identificación de **peligro de atropello**. El objetivo de este estudio será prevenir un accidente detectando peatones que puedan estar en riesgo porque un vehículo pueda atropellarlos.

   b) La identificación de **peligro por niños** que estén lejos de la tutela de adultos, **en una zona donde pueden transitar vehículos**.

   c) La **detección de intrusiones** en un escenario específico.

   Nuestro propósito es diseñar tres **sistemas expertos** independientes, cuyo análisis pueda ser aplicado en *cualquier escenario* monitorizado.

3. Finalmente se llevará a cabo un **estudio experimental** de las diversas tecnologías desarrolladas.

Como paso preliminar de esta investigación, y para poner en contexto el trabajo realizado, se ha llevado a cabo un **análisis sobre**

el estado del arte[1] de los distintos Sistemas de Vigilancia actuales, existentes tanto en el mundo académico como en el comercial. Este estudio previo nos ha permitido conocer los antecedentes y el estado actual de la Vigilancia Inteligente.

Tras el trabajo desarrollado sobre el estado del arte, somos conscientes de que los sistemas de seguridad inteligentes de última generación presentan muchas ventajas con respecto a los sistemas de vigilancia tradicionales. Sin embargo, aún existen algunas carencias que pueden mejorarse. De ahí el interés de alcanzar los objetivos citados anteriormente.

1.3. Estructura de la memoria

El presente documento describe el trabajo desarrollado en esta tesis doctoral y se divide en 5 capítulos.

En este **primer capítulo**, se han mostrado los principales aspectos que han motivado este estudio: la demanda de vigilancia inteligente mediante herramientas flexibles, escalables y robustas, que ayuden al operador en su trabajo. También, hemos definido un marco de trabajo y hemos descrito, de forma general, los objetivos que se pretenden alcanzar en esta tesis.

En el **capítulo 2**, se muestra un estudio sobre la situación actual de los distintos Sistemas de Vigilancia, los diferentes campos de aplicación y las diversas técnicas empleadas en el desarrollo de dichos sistemas. Esta investigación ha sido el punto de partida de este trabajo y enmarca el estado de la cuestión en la actualidad.

Conocer el estado del arte, nos ha permitido analizar las debilidades existentes en la Vigilancia Inteligente (presentadas de forma resumida en la sección 2.5), las cuales han marcado la dirección a seguir dentro de este área y han impulsado el desarrollo de una nueva investigación que quedará reflejada en este documento.

En el **capítulo 3**, se describe el Modelo que proponemos para el

[1] Este estudio es descrito de forma detallada en el capítulo 2

desarrollo de sistemas de vigilancia inteligentes, que se caracterizan por analizar distintas situaciones de riesgo, a partir de información heterogénea proveniente de diversas fuentes de monitorización de entornos.

En primer lugar (sección 3.1), analizaremos con detalle nuestra propuesta, los requisitos que proponemos y las dificultades a los que nos enfrentamos. Para ello, mostramos una visión global de la estructura y las características de los sistemas de seguridad que pretendemos abarcar. En segundo lugar (sección 3.2), especificamos el tipo de escenario a monitorizar. En tercer lugar (secciones 3.3 y 3.4), definimos formalmente el Modelo teórico propuesto, inspirado en el paradigma de desarrollo de software basado en capas y componentes. Finalmente (en la sección 3.5), describimos las Ontologías que dan soporte al Modelo presentado.

En el **capítulo 4**, se proponen tres sistemas expertos que analizan distintas situaciones de riesgo que requieren vigilancia:

1. La identificación de la presencia de *peligro de atropello*, (sección 4.1).
2. La identificación de *peligro por niños en zona de tráfico* porque no estén bajo la tutela de adultos, (sección 4.2).
3. La *detección de intrusiones* en un escenario específico, (sección 4.3).

Estos sistemas han sido desarrollados basándose en el Modelo presentado en el capítulo 3. Para cada una de las tres herramientas propuestas, hemos estructurado los siguientes apartados: la *estructura general* y el *objetivo* que se intenta conseguir, su *arquitectura* y cada uno de los componentes que constituyen dicha arquitectura, el desarrollo y la implementación final del Sistema y los resultados obtenidos en la etapa experimental. Para detectar la presencia de las distintas situaciones de riesgo, se proponen 3 controladores difusos que son descritos en detalle en dicho capítulo.

Finalmente, en el **capítulo 5**, exponemos las conclusiones finales obtenidas tras el desarrollo de esta tesis y mostramos las principales vías de investigación abiertas.

La memoria concluye con la **bibliografía** usada en este estudio.

Capítulo 2

Estado del Arte

La *Vigilancia Inteligente* está despertando un gran interés en el mundo científico y comercial por las ventajas que aporta. Su objetivo es construir sistemas capaces de interpretar de forma automática una determinada escena, con el fin de detectar riesgos y alertar de su presencia en tiempo real. Para ello, se analiza el comportamiento de los objetos existentes en un escenario vigilado, basándose en la información que transmiten una serie de sensores. En los últimos años, el interés en desarrollar en esta línea de investigación se ha incrementado de forma considerable. Actualmente, existe una amplia variedad de trabajos sobre sistemas de monitorización inteligentes.

En este capítulo, describimos el estado del arte de los Sistemas de Vigilancia. Primero, mostraremos la evolución que han sufrido estos sistemas desde sus inicios hasta la actualidad. Posteriormente, conoceremos las diferentes etapas que constituyen el proceso de Vigilancia Inteligente, y cuáles son las principales técnicas empleadas en cada fase.

Además, presentamos un gran abanico de diferentes herramientas para la monitorización inteligente, a las que hemos clasificado en función del campo de aplicación para el que han sido desarrolladas. Distinguiremos entre sistemas que sólo usan la información de vídeo como fuente de conocimiento y sistemas que se alimentan de datos heterogéneos provenientes de diversas fuentes de información (vídeo, audio, sensores...). Estos últimos son más escasos porque provienen de

una reciente línea de investigación, que aún no ha sido explotada debido a la dificultad para integrar información heterogénea.

Finalmente, exponemos una serie de anotaciones sobre el estado del arte que nos permitirán conocer las principales carencias que actualmente existen en la Vigilancia Inteligente, así como, destacaremos las principales líneas de investigación abiertas.

2.1. Distintas Generaciones en los Sistemas de Vigilancia

A lo largo de la historia las sociedades han sufrido grandes cambios. En algunos aspectos, estos avances han desembocado en un entorno inseguro en el que las personas y las infraestructuras están expuestas a nuevas amenazas y riesgos (vandalismo, robos, accidentes de tráfico, terrorismo,...). Estas circunstancias han provocado la demanda de protección en muchos ámbitos. De esta forma, gracias al avance científico y tecnológico, nacen los Sistemas de Vigilancia para dar respuesta a estos problemas, aumentando el nivel de seguridad en espacios públicos y privados, y mejorando la calidad de vida de las personas y las infraestructuras.

Los sistemas de vigilancia han evolucionado a lo largo de los años, lo que permite distinguir varias generaciones de sistemas. Concretamente, Valera y Velastin realizaron un estudio (descrito en [151]) donde diferencian tres generaciones principales:

Primera generación.

El comienzo de los sistemas de Vigilancia lo constituyen los *sistemas de circuito cerrado de televisión*, cuyo acrónimo en inglés es *Closed Circuit Television, CCTV*. Éstos fueron introducidos en Estados Unidos e Inglaterra en los años 60 y 70. Los primeros sistemas eran bastante simples; consistían en cámaras de baja resolución en blanco y negro conectadas por un cable coaxial. Cada cámara estaba ligada a un monitor en blanco y negro, lo que implicaba que si se usaban 15 cámaras

se necesitaban 15 monitores.

La evolución de esta tecnología comenzó casi en el mismo momento que empezó a existir. Al principio se añadió una caja de conmutación, lo que permitía al operador cambiar de cámara desde su punto de vigilancia. Gracias a esto, los operadores podían ver varias cámaras desde un solo monitor (aunque no simultáneamente).

Los años 70 trajeron multiplexores, VCRs *(Video Cassette Recorder)* o vídeos de seguridad y cámaras más elaboradas. Los multiplexores permitían dividir la pantalla de un monitor en cuatro partes con imágenes diferentes. Los VCRs permitían grabaciones y distribución de vídeo. Las nuevas cámaras daban fiabilidad y mejor resolución, como también mayor compatibilidad con otros equipos.

Al principio hubo muchos problemas, sobre todo con los VCRs. La calidad de las grabaciones era muy pobre. La combinación de la baja resolución de las cámaras y la poca calidad de las cintas de vídeo se traducía en imágenes poco claras y con niebla. La tecnología VCR no permitía al operador revisar y grabar eventos simultáneos y llevaba mucho tiempo encontrar ciertos momentos específicos. Tampoco existía la manera de poder hacer una gestión remota desde otro lugar.

La funcionalidad y rendimiento de estos sistemas en sus orígenes eran muy básicos y a su vez muy caros, tanto el equipamiento como la instalación. Sin embargo, éstos fueron evolucionando, y a mediados de los noventa, la nueva tecnología trajo innovaciones como el DVR *(Digital Video Recorder)* o vídeo grabador digital que permitía grabar imágenes a una resolución mucho mayor que su antecesor. También solucionó un inconveniente de los antiguos métodos de grabación, la cinta de vídeo.

Los DVRs son automáticos y requieren poca intervención del operador. Las imágenes se etiquetan con su correspondiente fecha y hora y es muy fácil de acceder a ellas. Los DVRs, usando tecnología IP *(Internet Protocol)*, permiten a usuarios remotos autorizados acceder a las grabaciones desde Internet, una red LAN o WAN e incluso controlar el movimiento de las cámaras distantes, surgiendo así los sistemas de vigilancia-IP.

Las cámaras actuales vienen con increíbles mejoras tales como

alta resolución, zoom y una buena cantidad de lentes que nos permite tener una amplia panorámica de lo que se está vigilando. Con algunos suplementos, como por ejemplo los infrarrojos, podemos obtener una visión nocturna. Además, se pueden configurar para que empiecen a grabar si salta alguna alarma que previamente se haya determinado.

Hoy en día, los sistemas CCTV están mucho más evolucionados que las configuraciones básicas de cámaras y monitores iniciales. Ahora son escalables, flexibles y compatibles con casi toda la tecnología existente en el ámbito de la seguridad. La evolución ha sido vertiginosa en unos pocos años y se espera que lo siga siendo.

Actualmente, la mayoría de los sistemas CCTV usan técnicas analógicas para el almacenamiento y la distribución de imágenes. Este hecho supone un problema, ya que las cámaras convencionales generalmente usan un '*dispositivo de cargas [eléctricas] interconectadas*' llamado CCD *(charge-coupled device)* para capturar imágenes. El inconveniente está en que la imagen digital es convertida en señal de vídeo analógica, la cual es conectada a la matriz de vídeo CCTV, monitores y equipo de grabación. La conversión de digital a analógico provoca una degradación en la imagen, además la señal analógica es susceptible al ruido.

Aunque es menos común, también es posible tener sistemas CCTV digitales que mantengan la ventaja de seguir usando el formato digital inicial.

En estas últimas décadas, estos sistemas se han empleado y se emplean mundialmente para vigilar espacios protegidos. Se suelen instalar sobre todo para la monitorizaron de estaciones de trenes, aeropuertos, estadios, tiendas, centros comerciales, garajes, parkings, centros oficiales.... Un ejemplo de ello puede verse en [100], donde se discute la integración de diferentes sistemas CCTV para monitorizar sistemas de transporte.

El uso de sistemas de vigilancia se ha incrementado en los últimos años, especialmente en sectores como el bancario y financiero, así como en otras áreas que requieren de rigurosas medidas de protección como complejos urbanísticos cerrados.

Si se pretende que los sistemas CCTV sean usados para detectar situaciones anómalas en tiempo real, existe un gran inconveniente que radica en la necesidad de un operador humano, que constantemente esté pendiente del vídeo de las diferentes cámaras que constituyen el sistema de vigilancia de un entorno concreto. Esto implica que durante las 24 horas debe de haber personal visualizando las pantallas. Este hecho no siempre puede ser controlado, ya que pueden surgir circunstancias que conlleven a la ausencia del vigilante durante un cierto tiempo, en el que puede darse la posibilidad de que surja un riesgo que no pueda ser visto por el ojo humano.

Por otro lado, si se requiere que haya una persona constantemente viendo los vídeos, existen estudios (como el descrito en [137]) que demuestran que tras 20 minutos visualizando imágenes, el rendimiento del operador humano disminuye considerablemente, ya que es inevitable que aparezcan síntomas de fatiga y falta de atención. En esas circunstancias, también puede darse el caso de que ocurran situaciones de interés que no sean detectadas.

A pesar de que el fin ideal de la instalación de un sistema de cámaras de seguridad debería ser visualizar anomalías en tiempo real para poder evitar un riesgo o peligro, en muchos casos, estas aplicaciones no son usadas con tal fin, debido a los problemas anteriormente descritos (ausencia del operador humano, fatiga, falta de atención...). De esta forma, el uso principal que se hace de estos sistemas es recurrir al vídeo grabado una vez que el riesgo ya ha ocurrido; bien para afrontar la situación, o bien para usarlo como prueba en trámites legales en el caso que sea necesario.

Para solventar los problemas de la vídeo-vigilancia tradicional, surge la necesidad de una interpretación inteligente de las escenas que alerte de los peligros de forma automática, dando lugar a la segunda generación de sistemas.

Segunda generación.

El avance tecnológico permitió que los sistemas de primera generación evolucionaran a *sistemas semi-automáticos*, constituyéndose

así la segunda generación. Este tipo de sistemas pretenden reducir la carga de trabajo de un vigilante de seguridad, ya que se basan en la creación de algoritmos inteligentes para la detección de eventos en tiempo real con el fin de cooperar con el operador humano en el reconocimiento de riesgos.

Con esta generación de sistemas, nace una nueva línea de investigación sobre *Vigilancia Inteligente*. Actualmente, este área de estudio está totalmente en auge. Como veremos en las próximas secciones, existen muchos trabajos realizados en este ámbito, pero todavía queda mucho por avanzar, ya que aún persisten ciertas carencias en los sistemas de seguridad inteligentes.

La segunda generación de sistemas usa la tecnología CCTV y vigilancia IP para obtener una señal de vídeo que pueda ser analizada y estudiada con el fin de detectar eventos en tiempo real. Este proceso se conoce como Análisis del Contenido del Vídeo o VCA *(Vídeo Content Analysis)*.

Estos sistemas usan como única fuente de conocimiento el vídeo, y tan solo usan la información que proviene de una sola cámara. Esto implica que si el recurso de entrada se daña, el sistema, deja de funcionar. Los sistemas que se basan en el análisis del contenido del vídeo son conocidos en la literatura con el término de *Sistemas de Vigilancia Visual*.

De este modo, los sistemas de esta generación tratan de interpretar, de forma inteligente y automática, los acontecimientos que tienen lugar en el escenario monitorizado. Su propósito es identificar los objetos que se mueven en la escena grabada por una cámara de vigilancia y seguir sus trayectorias intentando comprender las acciones que realizan. Posteriormente, el estudio de los comportamientos de dichos objetos puede usarse para conocer si realizan anomalías que supongan un peligro de amenaza en el espacio vigilado. Por ejemplo, conocer si una persona o un vehículo entra en un área protegida, o bien, si un vehículo estaciona en un lugar prohibido, etc.

Para llevar a cabo el proceso de análisis inteligente sobre la interpretación de una escena, se crean algoritmos de *Visión por Ordenador* y

se emplean técnicas de *Inteligencia Artificial*. Estos dos tecnologías juegan un papel fundamental en la Vigilancia Inteligente.

Todavía en esta generación, la detección de eventos en tiempo cercano al real es un reto a conseguir. Como veremos en posteriores secciones, en muchos sistemas de este tipo, el tiempo de procesamiento se demora. Este hecho es vital en un proceso de vigilancia en el que se requiere que los riesgos sean detectados con tiempo para poder ser solventados. Además, para poder mejorar la vídeo-vigilancia tradicional, el objetivo de la vídeo-vigilancia inteligente radica en la detección automática de amenazas en tiempo real.

Tercera generación.

Los sistemas que constituyen la tercera generación se caracterizan por ser altamente distribuidos. Son definidos para monitorizar una área extensa con muchos puntos de supervisión, es decir, usando un gran número de cámaras de vigilancia distribuidas en el entorno, con el objetivo de reflejar la naturaleza jerárquica y distribuida del proceso de vigilancia humano. Incluso, algunos de los últimos sistemas que actualmente se están desarrollando, incluyen también la instalación de otro tipo recursos (micrófonos, sensores...) que aportan información diferente a la del vídeo y que también es relevante conocer en el proceso de vigilancia, para enriquecer y fortalecer el proceso de razonamiento.

Frente a los sistemas de segunda generación, la distribución de las capacidades de procesamiento sobre la red y el empleo de dispositivos empotrados ofrecen a los sistemas de tercera generación, escalabilidad y robustez. Escalabilidad para introducir nuevos recursos de monitorización que aporten mayor información al sistema. Robustez en cuanto a que si un recurso deja de funcionar, el sistema puede seguir trabajando con la información de los demás recursos. Por otro lado, el hecho de que el procesamiento no esté centralizado, hace que el sistema gane eficiencia y los tiempos de razonamiento sean menores, y por lo tanto, más cercanos al procesamiento en tiempo real.

La finalidad que se persigue con los sistemas de esta última generación es que éstos sean capaces de emitir alarmas cuando un

riesgo o un peligro es detectado. Para ello, deben de realizar una buena interpretación de la escena, basándose en la información procedente de varios sensores que monitorizan el entorno protegido. De esta forma, no existe la necesidad de que el operador esté constantemente visualizando los vídeos de las cámaras de vigilancia. El objetivo es que estos sistemas llamen la atención del operador humano en tiempo real, sólo en los momentos que sea necesario, reduciendo así la carga de trabajo del vigilante.

En la literatura, estas aplicaciones son conocidas como *sistemas multi-sensor*, ya que emplean un amplio número de sensores como fuentes de información del sistema. Los recursos de monitorización usados pueden ser cámaras, micrófonos, sensores de movimiento, detectores de humos.... Cuando los sensores que participan en el espacio son únicamente cámaras de vigilancia, se califican como *sistemas multi-cámara*.

Disponer de un amplio conjunto de recursos para monitorizar un espacio físico, observando la escena desde diferentes puntos y transmitiendo datos simultáneamente, hace posible la creación de un sistema más potente para realizar una buena interpretación de la escena, ya que disponen de una gran cantidad de información diversa sobre los acontecimientos que tienen lugar en el entorno protegido. Esto es una gran ventaja, pero sin embargo, juega en contra del desarrollo de sistemas, ya que es difícil combinar diferentes recursos en una misma red de sensores.

Además, la integración de información heterogénea no es un proceso trivial. Cuando un sistema se alimenta del conocimiento obtenido de diversas fuentes, es necesario un proceso de fusión e integración de información que sea capaz de identificar los datos que refieren a un mismo objeto.

Esta generación de sistemas de vigilancia es la más reciente y por lo tanto la menos explotada. Actualmente existen numerosas investigaciones centradas en el desarrollo de sistemas de este tipo. En las secciones posteriores veremos algunos sistemas que son clasificados como sistemas de última generación.

Los sistemas de segunda y tercera generación se basan en el uso

de las tecnologías CCTV y vigilancia-IP más modernas y avanzadas.

En la tabla 2.1, (basada en el estudio de Valera y Velastin [151]), exponemos las principales técnicas, ventajas e inconvenientes, y el estado de investigación actual de cada una de las distintas generaciones.

	1ª Generación	**2ª Generación**	**3ª Generación**
Técnicas	Sistemas CCTV analógicos	Sistemas de Vigilancia Visual Inteligentes que combinan la tecnología CCTV con técnicas de Visión por Ordenador	Sistema de Vigilancia Inteligentes que usan un gran número de sensores para monitorizar una gran área
Ventajas	-Tecnología madura -Buen funcionamiento -Ampliamente utilizados en la práctica	-Vigilancia Inteligente -Incremento de la eficiencia de los sistemas CCTV -Ayuda al operador humano en el proceso de vigilancia	-Vigilancia Inteligente con mejores resultados tras la combinación de diferentes tipos de sensores -Distribución del procesamiento

Inconvenientes	-Uso de técnicas analógicas para la distribución y almacenamiento de imágenes -Necesidad de la intervención humana para la detección de riesgos en tiempo real	-Necesidad de crear algoritmos de detección y seguimiento de objetos robustos para un posterior análisis del comportamiento -Soluciones todavía en desarrollo ante la detección de situaciones muy concretas	-Plataformas multi-sensor -Dificultad ante la comunicación de los distintos sensores -Integración de la información heterogénea es un proceso complejo -Metodología del diseño

Investigación actual	-Analógico vs Digital -Técnicas de compresión de vídeo -Almacenamiento y recuperación de vídeo digital	-Creación de algoritmos y técnicas de visión artificial que ofrezcan resultados robustos en tiempo real -Reconocimiento de acciones desarrolladas en la escena -Aprendizaje automático del conocimiento del entorno -Análisis estadístico de la escena vs interpretaciones en lenguaje natural	-Inteligencia centralizada o distribuida -Integración de información heterogénea -Marco de razonamiento probabilístico -Identificación de riesgos importantes en tiempo real, evitando falsas alarmas -Ayuda a la toma de decisiones

Tabla 2.1: Resumen de la evolución técnica de los Sistemas de Vigilancia

2.2. Fases de la Vigilancia Inteligente y técnicas aplicadas.

El proceso de vigilancia inteligente llevado a cabo en sistemas de segunda y tercera generación es bastante complejo, por lo que puede dividirse en diferentes etapas. Estas fases han sido definidas en diferentes investigaciones [151, 154] (aunque en estos trabajos se centran solo en sistemas de vídeo-vigilancia). En esta sección nosotros hacemos una nueva propuesta basada en las investigaciones anteriores, pero contemplando ya la existencia de sistemas que usan fuentes heterogéneas de información. De este modo, las posibles etapas de procesamiento que consideramos que un Sistema de Vigilancia puede abarcar son:

1. Detección y reconocimiento de objetos en movimiento.
2. Clasificación de objetos.
3. Seguimiento de objetos.
4. Análisis del comportamiento de los objetos.

Estas fases se caracterizan porque realizan análisis independientes. En cada etapa se realiza un procesamiento que genera nuevo conocimiento que puede ser empleado en la etapa siguiente.

2.2.1. Detección y reconocimiento de objetos en movimiento

Es la primera etapa del proceso y consiste en detectar los objetos dinámicos dentro del entorno vigilado; en otras palabras, identificar aquellos elementos que no forman parte del escenario y que aparecen y desaparecen de forma dinámica.

En esta fase, se pretende conocer cuántos objetos hay en la escena y cuál es su posición (y si es posible, conocer incluso otras características como su tamaño, velocidad...). Evidentemente, el método para detectar

los elementos móviles en el escenario varía en función del tipo de sensores que monitorizan el entorno. Actualmente, existe una amplia variedad de sensores, desde cámaras de vigilancia y micrófonos hasta sensores de movimiento, volumétricos...

Realmente, la fuente más usada, con diferencia, para la detección de objetos es el vídeo, ya que puede ofrecer información más enriquecedora e interesante para este aspecto: la detección de objetos en movimiento. Este proceso a veces es denominado proceso de segmentación y consiste en el procesamiento de los frames o imágenes obtenidas directamente a partir del flujo de vídeo de las cámaras.

Se han aplicado diversas técnicas de procesamiento de imágenes para realizar esta tarea. Actualmente, existen tres métodos básicos que permiten detectar el movimiento en una secuencia de frames de vídeo:

- Mediante la **extracción del fondo de la imagen**. Consiste en la substracción del fondo de la imagen de la cámara para usarlo como modelo de referencia sobre el que se compararán los frames del vídeo. Este método compara, píxel a píxel, cada frame con la imagen de referencia para identificar aquellas regiones que son distintas, lo que supone la detección de movimiento. Algunos trabajos destacados que usan esta técnica se pueden leer en [42,87].

  El modelo de substracción de fondo, a pesar de ser un método robusto para extraer información sobre objetos en movimiento, presenta una gran desventaja: la sensibilidad a cambios dinámicos en el entorno (iluminación y sombras en la imagen). La variabilidad de las condiciones del entorno conlleva la detección de falsos objetos. Este motivo hace que esta técnica sea mayormente utilizada para la vigilancia de espacios interiores donde las condiciones de luz no suelen cambiar. Para su aplicación en espacios abiertos, en muchos estudios se propone crear un modelo de substracción de fondo que se esté actualizando continuamente, con el fin de evitar una pobre detección de objetos si cambian las condiciones de la escena [119].

- Mediante la **diferencia temporal**. Esta técnica consiste en comparar frames consecutivos dos a dos. Las regiones de movimiento

en una imagen son extraídas analizando la imagen del frame actual con respecto al anterior.

La técnica de diferencia temporal, al ser adaptativa, ofrece buenos resultados en entornos dinámicos, mejorando el modelo de substracción de fondo en espacios exteriores, donde las calidades de luz y sombras son variables. Pero presenta un pequeño problema, se obtiene un peor rendimiento en la extracción de los píxeles pertenecientes a objetos en movimiento.

En [15, 88] se pueden leer trabajos basados en la técnica de diferencia temporal para la detección de objetos en movimiento.

- Mediante el **flujo óptico**. La segmentación del movimiento basada en el flujo óptico usa las características de los vectores de movimiento de los objetos en el tiempo, para detectar las regiones de movimiento de una secuencia de imágenes. Algunos trabajos dentro de este ámbito son descritos en [14, 96].

  Esta técnica presenta buenos resultados con cámaras móviles, pero, en muchos casos, requiere un alto coste computacional que impide que pueda ser aplicada en sistemas de tiempo real.

Además de estas tres técnicas, existen investigaciones que proponen otros métodos de detección de objetos en movimiento. Por ejemplo, en [32], se describe un sistema denominado VSAM donde se aplica un algoritmo híbrido para la segmentación de movimiento, combinando una técnica de substracción de fondo adaptativa con un modelo de diferencia temporal cada 3 frames.

Otro trabajo que presenta otra técnica basada en un modelo de clasificación Gausiano mixto es descrito en [50]. Este modelo clasifica los valores de píxeles en tres distribuciones independientes predeterminadas correspondientes a los conocimientos previos del fondo y las sombras. También actualiza el componente mixto de forma automática para cada clase, de acuerdo a la probabilidad de pertenencia.

No es menos importante, en algunas situaciones, la detección de sonidos del entorno, ya que al fin y al cabo son objetos dinámicos que pueden irrumpir en la normalidad del entorno. No son objetos móviles,

como personas o vehículos que accedan al escenario, pero si son eventos interesantes de detectar. Existen estudios que detectan eventos de sonido captados por micrófonos para procesos de vigilancia, algunos de ellos son [31, 35, 124]. En estos trabajos, la detección de eventos va ligada con procesos de clasificación.

2.2.2. Clasificación de objetos

Un aspecto de gran trascendencia es conocer la clasificación del objeto y sus características tipológicas, es decir, ¿se trata de un vehículo? ¿es una persona o grupo de personas?..., o incluso, si es un sonido ¿qué clase de sonido es?. Este hecho es relevante en la etapa de análisis de situaciones de riesgo para analizar el comportamiento de objetos conociendo qué tipo de objeto estamos estudiando. Existen acciones que son normales si son realizadas por personas, pero que conllevan una anomalía si las realiza un coche. Por ejemplo, acceder a un comercio por su puerta para clientes viandantes. Si esta acción es llevada a cabo por un vehículo, no es normal, puede ser que se trate de un alunizaje para un intento de robo.

Dependiendo de la fuente de información usada, las técnicas empleadas para clasificar eventos varían. Como en estos casos, para cubrir este objetivo, la fuente más usada es el vídeo, vamos a ver las dos técnicas generalmente empleadas para la clasificación de objetos en dicha fuente:

- **Clasificación basada en formas.** Esta técnica se basa en comparar la región donde se ha detectado el movimiento, (es decir, donde ha sido ubicado el objeto) con una plantilla de patrones o formas geométricas de distintos y posibles objetos. De este modo, la clasificación del objeto se hace en base a su silueta.

  Esta técnica comprueba cada objeto con cada patrón de la plantilla de siluetas definida previamente. En cada caso, se obtiene un valor numérico que indica el grado de pertenencia del objeto detectado al patrón aplicado. Cada patrón es asociado con un determinado tipo de objeto. El máximo valor obtenido indica que el

objeto es clasificado como el tipo de objeto correspondiente al patrón para el que se ha obtenido dicho valor.

Un trabajo que identifica a las personas estudiando la forma geométrica de su silueta es [84]. Otro trabajo destacado que se basa en esta técnica es el sistema VSAM [32], que usa una plantilla de patrones de formas para clasificar los objetos en cuatro clases: persona, vehículo, grupo de personas y aglomeraciones desconocidas. Usa una red neuronal de 3 capas como clasificador.

- **Clasificación basada en movimiento.** Otro modelo para la clasificación de objetos es analizar los movimientos que estos realizan [123]. Existen investigaciones que estudian la forma de andar de las personas y se basan en la idea de que éstas varían su silueta con facilidad al moverse, realizando movimientos periódicos en sus trayectorias [39, 41]. Los vehículos, en cambio, no suelen cambiar su silueta.

Igual que en la etapa anterior existen otros trabajos que ofrecen nuevas propuestas para la clasificación de objetos. Por ejemplo, en [28] se propone un nuevo método basado en una matriz de co-ocurrencias de los objetos en el tiempo, para clasificar jerárquicamente los objetos y comportamientos. Existen otros estudios, como [45], donde se estudian características adicionales, tales como el color y la velocidad, para obtener resultados más precisos en la clasificación; en este caso, de personas en entornos urbanos.

En muchos trabajos la clasificación de objetos ofrece resultados difusos, un ejemplo puede verse en [136]. En estos casos, la clasificación obtenida para un objeto va acompañada de un índice de certeza que indica la posibilidad de que este objeto pertenezca a dicha clasificación. Por ejemplo, un objeto podría considerarse persona con una posibilidad de que lo sea de 0.8 y vehículo con una posibilidad de 0.2.

2.2.3. Seguimiento de objetos

El seguimiento de un objeto, también conocido en la literatura como *tracking de objetos*, consiste en seguir sus movimientos a lo largo

del transcurso de ese objeto en el escenario. Esta etapa se lleva a cabo usando el conocimiento adquirido desde la fuente de vídeo, ya que permite seguir a un objeto con mayor facilidad.

Como es obvio, la etapa de *tracking* de objetos se realiza de forma posterior a la detección de los mismos. Primero se identifica la existencia de un objeto y posteriormente se siguen sus movimientos. En cambio, la clasificación puede ser anterior o posterior al tracking y, en muchos estudios, la etapa de clasificación y seguimiento se realizan de forma paralela (ya que la clasificación de un objeto puede depender de las trayectorias que sigue). En cambio, el seguimiento siempre es una etapa anterior al análisis de comportamientos de los objetos. Para poder interpretar las acciones que realiza un determinado objeto, es necesario que se realice un seguimiento sobre él.

El seguimiento de objetos es un proceso que se expone a diferentes dificultades; por ejemplo, la existencia de objetos que circulan por la escena muy próximos entre sí y que pueden ser confundidos como un único objeto; y la posible oclusión de un objeto cuando éste es ocultado por otro elemento y objeto de la escena. En este último caso, lo que se pretende es seguir el seguimiento cuando el objeto ocultado vuelve a aparecer y no crear objetos nuevos.

Existen diversos modelos para realizar el seguimiento de objetos, los más usadas son:

1. **Seguimiento basado en regiones**. Consiste en analizar las regiones de movimiento de la imagen. Estas regiones suelen corresponder a objetos que han sido identificados. Este método es el más comúnmente utilizado entre los distintos trabajos existentes en la literatura que realizan seguimiento de objetos.

   Los Filtros de Kalman, tanto los básicos como los extendidos son la técnica más usada en los sistemas que siguen esta filosofía. El filtro de Kalman, suaviza errores ya que permite eliminar ruidos en la imagen producidos por sombra o variaciones en la iluminación. Algunos trabajos que llevan a cabo esta técnica son [24,94,122,125], basado en el Filtro de Kalman y [32,82] usando un Filtro de Kalman Extendido.

2. **Seguimiento basado en contornos activos**. Consiste en calcular y considerar los modelos geométricos de los contornos que envuelven a los objetos dinámicos detectados. Los modelos geométricos de contornos activos están basados en la evolución de curvas o superficies por flujos geométricos. En dichos modelos, la curva o superficie se deforma según la velocidad de los objetos y depende de parámetros geométricos intrínsecos y de la adaptación a los contornos de la imagen. Un ejemplo en el que se aplica esta técnica puede verse en [9].

3. **Seguimiento basado en modelos**. Esta técnica se basa en la creación previa de modelos o patrones sobre los objetos que pueden aparecer en el escenario vigilado. Estos modelos se construyen con antelación, bien manualmente, con herramientas de CAD o con técnicas de visión por computador. Los algoritmos de tracking basado en modelos siguen los objetos, haciendo coincidir los modelos de patrones de objetos, previamente definidos, con los datos de la imagen.

   El seguimiento puede hacerse usando modelos 2D o modelos 3D, la mayoría de los trabajos usan patrones 2D. Por ejemplo, en [158] se aplica un modelo rectangular para seguir los coches al pasar cerca de la cámara y un modelo en forma de U para seguir la parte trasera del coche cuando está delante de la cámara. Otras propuestas de modelos geométricos 3D son [125], [48] y [82]. En estos casos, se emplea un conocimiento geométrico a priori de los distintos objetos que pueden aparecer en una escena (personas, vehículos -coche, furgón, camión. . . - . . .).

Como veremos a continuación en la sección 2.3, existe un alto porcentaje de sistemas que realizan el proceso de seguimiento de objetos de forma local a las cámaras, es decir, no abarcan la posibilidad de tener múltiples cámaras con diferentes vistas de la misma escena. Estos sistemas se basan en un modelo de seguimiento sobre la imagen o seguimiento 2D.

Actualmente, los sistemas más novedosos intentan incluir el seguimiento multi-cámara. En estos casos, es necesario conocer la posi-

ción relativa de los objetos respecto un punto de referencia empleando técnicas de calibración de la cámara [57]. De esta forma se puede realizar un tracking 3D sobre los objetos. Este seguimiento es más robusto porque tiene menos problemas de pérdida de perspectiva y presenta una gran solución al problema de la oclusión de objetos. Ejemplos de tracking multi-cámara puede verse en [24, 91].

2.2.4. Análisis e interpretación del comportamiento de los objetos

Esta etapa consiste en analizar los distintos comportamientos de los objetos en una escena. Este hecho conlleva un reconocimiento y una interpretación de las actividades que realizan los objetos seguidos.

Esta fase es de vital importancia en el proceso de vigilancia, ya que al analizar e interpretar la escena de forma computacional e inteligente, se pueden llegar a detectar situaciones anómalas que conlleven riesgos o peligros. Estas anomalías pueden implicar la activación automática de alarmas que llamen la atención del operador humano, con el fin de evitar, en muchos casos, una fatalidad. De esta forma, se logra construir herramientas que sirvan de apoyo al vigilante.

Existen dos filosofías para la detección de peligros:

- Desde el punto de vista de la *normalidad*. Existen trabajos que se inclinan por definir lo que es normal que ocurra en un entorno, para posteriormente compararlo con los eventos analizados. Si al estudiar el comportamiento de los objetos se encuentra una acción que no ha sido contemplada como normal, se considera que es una anomalía y por lo tanto, un riesgo. Algunas investigaciones basadas en esta idea son [5, 7, 19].
- Desde el punto de vista de la *anormalidad*. En este caso, se define directamente cuáles son los eventos anormales. Por lo tanto, un riesgo es detectado si al analizar el comportamiento de los objetos, se identifican circunstancias o eventos anormales que han sido predefinidos con anterioridad. Esta filosofía es la más aplicada. Algunos ejemplos que siguen este planteamiento son [6, 37, 47].

En esta etapa han sido aplicadas multitud de técnicas, entre las más destacadas se encuentran:

- **Modelos de Redes Probabilísticas**, como:
  - *Redes Bayesianas*. Una red bayesiana es un modelo probabilístico multivariado que relaciona un conjunto de variables aleatorias mediante un grafo dirigido que indica explícitamente una influencia causal. Gracias a su motor de actualización de probabilidades, el Teorema de Bayes, las redes bayesianas son una herramienta extremadamente útil en la estimación de probabilidades ante nuevas evidencias.

    Algunos ejemplos de sistemas que analizan comportamientos usando redes bayesianas son [99, 157].
  - *Modelos Ocultos de Markov, (HMM, Hidden Markov Model).* HMM es un modelo probabilístico en el que se asume que el sistema que se va a modelar es un proceso de parámetros desconocidos. El objetivo es determinar los parámetros desconocidos (u ocultos) de dicha cadena a partir de los parámetros observables. Los parámetros extraídos se pueden emplear para llevar a cabo sucesivos análisis, por ejemplo, en aplicaciones de reconocimiento de patrones. Un HMM se puede considerar como la red bayesiana dinámica más simple.

    Algunos ejemplos de sistemas de vigilancia inteligentes basados en Modelos Ocultos de Markov son descritos en [10, 110].
- **Dinamic Time Warping (DTW)** es una técnica basada en la comparación de dos secuencias variables en el tiempo y analizar sus similitudes. Esta técnica es ampliamente utilizada en el reconocimiento de voz y en los patrones de imagen, pero está siendo recientemente aplicada para identificar movimientos humanos a partir de un análisis de vídeo. Se trata de que coincidan con un patrón de referencia.

Un reciente trabajo que usa DTW es descrito en [80], en el que se propone un sistema basado en DTW para reconocer

secuencias multimodales de diferentes longitudes generadas a partir de múltiples sensores con salidas discretas y continuas.

- **Redes Neuronales**. Las redes neuronales artificiales (RNA) son un paradigma de aprendizaje y procesamiento automático inspirado en la forma en que funciona el sistema nervioso humano. Este método ha sido recientemente utilizado para el análisis del comportamiento de los objetos en escenas dinámicas.

  Por ejemplo, en [27], se presenta un sistema para detectar actividades anormales en estaciones de metro, usando un modelo basado en una red neuronal para clasificar dichos comportamientos.

- **Algoritmos de Agrupamiento (clustering)**. Se trata de un procedimiento para la agrupación de una serie de vectores de acuerdo con un criterio, normalmente de distancia, mediante el que se agrupan los vectores por características comunes.

  En [148] se describe un trabajo en el que aplican un algoritmo de clustering para la identificación de las trayectorias que siguen los objetos.

- **Gramáticas independientes de contexto**. Son gramáticas formales en la que cada regla de producción es de la forma $V \rightarrow w$, donde V es un símbolo no terminal y w es una cadena de terminales y/o no terminales. El término libre de contexto se refiere al hecho de que el no terminal V puede siempre ser sustituido por w, sin tener en cuenta el contexto en el que ocurra.

  Un ejemplo de aplicación que realiza una interpretación semántica de los eventos que tienen lugar en una zona de aparcamiento protegida y que emplea una gramática formal independiente del contexto para realizar el análisis de comportamientos de los objetos es presentado en [77].

- Algunos sistemas se basan en un modelo *basado en reglas* o modelos declarativos para representar el razonamiento lógico con el fin de reconocer acciones predefinidas y detectar situaciones anómalas; es el caso de los sistemas descritos en [37, 72, 128].

- **Modelos Difusos**. Consiste en sistemas expertos que hacen uso de la lógica difusa para interpretar los sucesos que ocurren en el escenario vigilado.

  En el ámbito de la vigilancia inteligente, es muy apropiado emplear sistemas que permitan trabajar con posibilidades e incertidumbre, ya que en muchos casos los datos obtenidos en las diferentes etapas de detección, clasificación y seguimiento de objetos van ligados a índices de creencia que indican posibilidades sobre dichos resultados.

  Además, durante el análisis de la interpretación de escenas, pueden surgir conceptos difusos susceptibles de estudiar (por ejemplo, la cercanía entre objetos, conocer si dos objetos están muy cerca o poco cerca...).

  Por ejemplo, en [30] los autores proponen una red neuronal difusa para la detección, seguimiento y clasificación de vehículos en una autopista. El sistema realiza el conteo de vehículos.

  En [98], se presenta un sistema multiagente para controlar el tráfico y detectar infracciones, donde se usa un sistema basado en reglas difusas. Otra aportación difusa es descrita en [46], en la que se presenta un sistema basado en reglas difusas para detectar accesos a zonas restringidas.

2.3. Sistemas de Video-Vigilancia. Aplicaciones.

En esta sección vamos a conocer distintas aplicaciones de sistemas de vigilancia basados en fuentes de vídeo. Veremos sistemas de primera, segunda y tercera generación. Éstos se mostrarán por orden cronológico, entremezclando sistemas de las distintas generaciones, ya que en los últimos años se están desarrollando, de forma paralela, sistemas de las tres generaciones.

Las aplicaciones desarrolladas en el mundo comercial suelen ser de primera generación, ya que la mayoría no incluyen técnicas inteligentes para la vigilancia. En cambio, los trabajos desarrollados en el mundo científico se centran en desarrollar sistemas inteligentes de segunda y

tercera generación. Como veremos a continuación, aquellos de última generación se caracterizan por ser altamente distribuidos al incluir varias cámaras de vigilancia.

Como se señaló en secciones anteriores, con los sistemas de primera generación, se necesita la presencia constante del un operador humano, el cual, viendo las imágenes de vídeo de las cámaras, puede visualizar los eventos acaecidos dentro del entorno vigilado y detectar personalmente las anomalías.

Una gran concentración sobre los vídeos, como se dijo anteriormente, puede provocar fatiga y falta de atención y darse circunstancias importantes que pasan desapercibidas ante la presencia del operador. Por ello, para facilitar este trabajo, durante las últimas dos décadas, se está llevando a cabo un gran esfuerzo por desarrollar Sistemas de Vigilancia Visuales Inteligentes que sirvan de soporte al ser humano, y que sólo alerte al vigilante de forma automática en circunstancias inusuales o de cierto riesgo.

Los Sistemas de Vigilancia Visuales se caracterizan por *analizar el contenido del vídeo* con el fin de detectar peligros en un entorno monitorizado por cámaras de vigilancia. Si en el espacio protegido existen varios puntos donde hay fuentes de supervisión que alimentan al sistema, éste suele ser clasificado como sistema *multi-cámara* o *multi-sensor*.

El vídeo es la fuente más explotada para llevar a cabo un proceso de monitorización sobre el entorno. La gran variedad de aplicaciones referentes a este tema lo demuestra. En la literatura específica se puede encontrar un gran número de investigaciones relacionadas con sistemas de seguridad inteligentes basados en vídeo. Esta línea de investigación es muy fértil. Este hecho puede reflejarse en los numerosos Workshop y Conferencias celebrados sobre Vigilancia Visual [1–4, 33, 73, 75, 76, 101–109], y en la existencia de revistas especializadas en este área de investigación [73–76], donde se pueden encontrar muchos trabajos.

En esta sección pretendemos dar una visión general sobre las distintas investigaciones desarrolladas en este área durante los últimos años. Para ello, vamos a mostrar un resumen sobre diferentes sistemas

de segunda y tercera generación. Veremos tanto sistemas comerciales, como sistemas destacados en el mundo académico.

Las tecnologías más empleadas dentro este ámbito son los *algoritmos de visión artificial y procesamiento de imágenes*. La tolerancia a fallos de los sistemas de Vigilancia visuales depende enormemente de la calidad de las imágenes procesadas y de los algoritmos diseñados. Éstos se pueden ver afectados por distintos tipos de sabotajes (oclusión, desenfoque, cambio del campo de vista de la cámara) [112] que se deben de tener en cuenta para la obtención de buenos resultados.

2.3.1. Sistemas de Vídeo-Vigilancia comerciales

El incremento y la demanda de la seguridad en la sociedad han provocado un aumento de la existencia de este tipo de sistemas. Existen Sistemas de Vigilancia Visuales que se han creado con un propósito comercial y que son diferentes de aquellos diseñados en el mundo académico.

Los sistemas comerciales, se caracterizan sobre todo por ser sistemas de primera generación (algunos en estados muy avanzados). Este tipo de sistemas tienen tendencia a usar un hardware de propósito específico e incrementar el uso de redes de cámaras digitales, dejando un poco al margen las tareas inteligentes (de los sistemas de segunda y tercena generación), mientras que los existentes en la literatura se basan en el desarrollo de la inteligencia del sistema, mejorando las tareas de procesamiento de imágenes, los algoritmos de detección, seguimiento y clasificación de objetos, el reconocimiento de actividades, la evaluación de herramientas...

En la actualidad existen muchos sistemas de vídeo-vigilancia comerciales, algunos ejemplos son DETECT [64], SIVICAM [70] y GOTCHA [66]. Son sistemas de primera generación con tecnologías modernas y muy avanzadas. Como veremos a continuación, solo uno de ellos, incorpora algún componente inteligente básico.

Detect [64] ofrece un sistema de vigilancia remota para ver las imágenes de vídeo desde una estación de trabajo remota mediante una

LAN o Internet, desde cualquier parte del mundo, usando una de las tecnologías más avanzadas de sistemas de Circuito Cerrado de TV.

Sivicam [70] es el denominador que usa la empresa *Siadasa* para ofrecer soluciones a medidas sobre sistemas de vigilancia con las últimas novedades en tecnología de la imagen (interiores/exteriores, visión nocturna por infrarrojos, detección de movimiento....). Permite el acceso instantáneo o remoto a la imagen de la cámara, a tiempo real o a los registros de grabación almacenados. Además, tienen un software específico para la transmisión de las imágenes sobre teléfonos móviles y PDAs. También ofrecen diversos paquetes de programas de visión inteligente para: la detección de movimiento en zonas restringidas, la detección de desaparición de objetos, el reconocimiento de patrones, el conteo de personas y el conteo de sucesos. Cuando detectan una alarma, realizan la generación de la misma por mail, SMS o llamadas telefónicas para avisar del suceso acontecido.

Gotcha es uno de los software más completos para utilizar las pequeñas cámaras en labores de vigilancia. Este software puede ser descargado desde el sitio [66], en una versión de prueba con algunas restricciones y que funciona durante 30 días. El sistema solo captura cuando detecta movimiento, y cuando efectúa la captura añade automáticamente la hora. De este modo, se puede visionar lo que se desee de forma fácil, con tan solo conocer la hora. Cuando se detecta movimiento, además se puede enviar un email para avisar al usuario o colocar imágenes en una página web a través de FTP *(File Transfer Protocol)*. El sistema incorpora, además, una base de tiempos de disparo de la cámara. Con este software, se puede transformar las *webcams* en efectivos sistemas de vigilancia y conocer si alguien ha usado el ordenador o ha visitado el escritorio mientras el usuario estaba ausente.

Los sistemas comerciales están diseñados, de manera estandarizada, con el objetivo de vender el producto en diversos campos de aplicación. Estas herramientas pueden tener múltiples usos y ser aplicadas en diferentes ámbitos. Por ejemplo: para la protección de comercios, del control de mercancías, de obras a distancias, para proteger a los seres queridos, visión remota de niños y familiares, control de la propiedad...

Sin embargo, en el mundo académico, por lo general, los sistemas de vigilancia se desarrollan para abarcar propósitos específicos. Las áreas de aplicación más comunes para el desarrollo de investigaciones sobre vigilancia inteligente han sido la monitorización del tráfico y la vigilancia en estaciones públicas. A continuación, vamos a ver un resumen con los principales sistemas, clasificándolos según su campo de aplicación.

2.3.2. Sistemas de Vídeo-Vigilancia en el área de Tráfico

La monitorización del tráfico es uno de los usos a los que más partido se está sacando en el desarrollo de Sistemas de Vigilancia Visuales. Existen numerosas aplicaciones dentro de este campo de aplicación. El sistema AUTOSCOPE [97], desarrollado en USA en los años 80, ha sido uno de los primeros ejemplos más conocidos sobre monitorización de tráfico en carreteras, basada en vídeo. Este sistema sólo realiza una detección de los vehículos a partir del flujo de imágenes obtenido de las cámaras.

Uno de los primeros prototipos de los 90 se presenta en [34]. Fue propuesto para vigilar aeropuertos y llevar un control sobre el tráfico. El sistema abarca la transmisión en tiempo real y la descripción de escenas complejas. Sin embargo, tiene un gran inconveniente, está limitado a casos en los que se conoce tanto la estructura de la escena, como los objetos que aparecen y cómo se van a comportar.

Otro trabajo de la época se describe en [144]. En él se muestra un sistema con gran capacidad para la detección y el seguimiento de vehículos en carreteras. En él, se emplea un procesamiento de imágenes basado en regiones espacio-temporales y métodos de probabilidad basados en contorno. Sin embargo, como en el trabajo anterior, no puede usarse para casos reales porque es necesario conocer a priori el número de objetos presentes en la escena. Además, tiene la dificultad de su alta complejidad computacional.

Unos años más tarde, en el 97, se presenta un sistema de vigilancia basado en vídeo para la medición de los parámetros de tráfico [16]. Este sistema realizada una detección, clasificación y seguimiento de vehículos,

con el fin de controlar las congestiones. El modelo captura el vídeo de las cámaras, que posteriormente es digitalizado y procesado por dispositivos hardware embebidos. La información obtenida se envía al Centro de Gestión de Transporte (TMC).

Los autores de [16] usan modelos basados en características y en contornos, que son rastreados. Una vez que los contornos han sido rastreados, son agrupados por el módulo de subcaracterísticas en vehículos candidatos. Este módulo de agrupación construye un grafo donde los vértices son componentes de subcaracterísticas, los bordes agrupan relaciones entre vértices, y los componentes conectados corresponden al vehículo candidato. Cuando la última parte de un componente conectado entra en la región de salida, un nuevo vehículo candidato es generado y el componente es eliminado de la agrupación en el grafo. El sistema se compone de un ordenador personal conectado a una red de 13 DSPs (procesadores de señal digitales). Seis de estos DSPs realizan el rastreo, cuatro la detección de bordes, y uno actúa como el controlador. De hecho, éste último está conectado a un DSP, que actúa como un capturador, y a otro DSP que hace de display. La actualización del rastreador es enviada al ordenador. El sistema ofrece buenos resultados tanto en condiciones de mucho tráfico, como de noche o en intersecciones urbanas.

Dentro del área de tráfico existen aplicaciones con el fin de monitorizar los parkings o zonas de aparcamiento. En ese mismo año, en [125] se muestra un sistema dividido en dos módulos visuales, uno capaz de identificar y seguir vehículos y el otro peatones, realizando un proceso de seguimiento híbrido.

Podemos observar que, en los inicios, exclusivamente se construían sistemas que cubrían sólo las dos primeras etapas definidas anteriormente, tratando la detección y el seguimiento como la única tarea de vigilancia, sin realizar ninguna interpretación semántica de los resultados. Sin embargo, es importante realizar una etapa posterior donde se lleve a cabo un análisis de los acontecimientos que tienen lugar en el entorno vigilado.

En el 98, se presentó un sistema de monitorización de tráfico

para carreteras de Italia (que fue descrito en [117]). El sistema usa cámaras de color PTZ (pan, tilt and zoom). Éstas se instalaron en los lugares donde las cámaras CCTV no ofrecen buenos resultados (por ejemplo, bajo malas condiciones atmosféricas). La aplicación tiene dos puntos principales de control en diferentes lugares geográficos y nueve puntos secundarios. La imágenes se recogen de forma centralizada, y se envían a cada uno de los puntos secundarios, usando el formato MPEG-1. Se transmiten 2Mbps. En algunos de los centros secundarios, como los del control de túneles, se han instalado subsistemas comerciales para la detección de colas o vehículos parados. El sistema es capaz de reconocer situaciones determinadas y alertar al operador humano. Los autores resaltan la importancia de tener una estructura jerárquica para el control y procesamiento de imágenes. El problema es que no es completamente automático.

En [32], se muestra un resumen final sobre el proyecto Video Surveillance and Monitoring (VSAM) que duró tres años (1997-1999) y fue desarrollado por Robotics Institute, Carnegie Mellon University (CMU) y Sarnoff Corporation. En este proyecto se construyó un sistema que usa múltiples cámaras que cooperan entre sí, recogiendo y distribuyendo la información en tiempo real. Se realiza una detección de objetos híbrida diferenciando entre personas y vehículos. Se emplea una red neuronal para distinguir entre una persona, un grupo de personas y un vehículo; y un método de clasificación basado en un análisis discriminatorio lineal para hacer distinciones entre vehículos (camión, furgoneta...). El sistema es capaz de detectar si una persona está andando o corriendo, basándose en las figuras que adquieren las personas al moverse. Consideran como anomalía que una persona corra. Otra finalidad del sistema es identificar las trayectorias de los objetos. Este sistema ha sido diseñado con tales propósitos y es difícil de escalar con el fin de aumentar las capacidades del mismo.

En VSAM, para el seguimiento de objetos, los autores amplían la noción de filtro de Kalman básica con el fin de tener una lista de múltiples hipótesis o alternativas de movimiento para manejar casos donde existe cierta ambigüedad entre múltiples objetos de movimiento. Las propuestas que usan el filtro de Kalman básico tienen un uso

limitado porque están basadas en la densidad Gaussiana unimodal que no puede soportar la hipótesis de movimiento alternativo simultáneo [90].

En el 99, un sistema de vigilancia basado en vídeo que controla el flujo de tráfico en carreteras centrándose en la detección de ciclistas y peatones se describe en [55]. La aplicación tiene dos módulos de procesamiento principales: el módulo de seguimiento que realiza diferentes fases -detección de movimiento, filtrado, y extracción de características- y el módulo de análisis que usa un método basado en la cuantificación de un vector de aprendizaje para obtener el cómputo final. Uno de sus inconvenientes es que los algoritmos de procesamiento de imágenes no son lo suficientemente robustos, ya que dependen de la posición de la cámara. Además, el módulo de análisis no es en tiempo real, sino que el razonamiento se hace a posteriori (offline).

También a finales de los noventa y dentro del problema de monitorización del tráfico en parkings, en [77] se describe uno de los primeros trabajos donde se empieza a contar con la etapa de interpretación de escenas de forma más desarrollada. La aplicación propuesta realiza una interpretación semántica de los eventos que tienen lugar en una zona de aparcamiento protegida. Su objetivo es realizar un seguimiento de los vehículos y de los peatones, para identificar cuando una persona aparece en la escena para subirse a un vehículo y abandonar el parking; y la situación contraria, cuando un vehículo aparece en la escena para estacionar, y de él se baja, al menos, una persona.

Existen 3 partes importantes en el sistema [77]: el 'tracker' (seguidor), que rastrea los objetos y recoge sus movimientos; el 'generador de eventos', que genera eventos discretos a partir de los movimientos detectados, basándose en un modelo de un único entorno (aparece una persona, aparece un vehículo, desaparece una persona, desaparece un vehículo, un vehículo se detiene...); y finalmente, el 'analizador gramatical' que analiza los eventos, según un modelo de gramática independiente de contexto estocástica *(stochastic context-free grammar, SCFG)* y describe estructuralmente actividades posibles: 1)una persona sube a un vehículo y sale del parking, 2)un vehículo estaciona en el aparcamiento. A pesar de que se realice una interpretación semántica

de las actividades de los objetos, el sistema tiene el inconveniente de que usa una única cámara inmóvil y no está maduro, porque los resultados obtenidos ofrecen todavía bastantes interpretaciones falsas.

En el año 2003, otra aplicación de vigilancia para áreas de aparcamiento es descrita en [82]. En este trabajo se presenta un sistema constituido por tres fases: un módulo de movimiento; un modelo de visualización y refinamiento; y el módulo de seguimiento e interpretación de las actividades que realizan los vehículos. Los autores aplican el Filtro de Kalman Extendido para llevar a cabo la fase de seguimiento. Calculan el levantamiento 3D del vehículo sobre el plano de la imagen, usando la información de calibración de la cámara. El módulo de interpretación semántica se realiza en 3 fases: primero, una clasificación de la trayectoria; después, un paso de clasificación en línea usando clasificadores Bayesianos; y finalmente, se aplican descripciones en lenguaje natural a los patrones de trayectorias de los coches que han sido identificados.

Al igual que los anteriores, y también en el 2003, se presenta en [24] un nuevo sistema de vigilancia para su uso en parkings. La arquitectura consiste en uno o varios subsistemas de cámara estáticos y uno o varios subsistemas de cámara activos. Primero, tiene lugar la detección y el seguimiento del objeto por los subsistemas estáticos. Una vez que el objeto ha sido seleccionado, un PTZ (pan, tilt and zoom) que forma el subsistema activo, es activado para capturar el vídeo del objeto a alta resolución. El modelo de seguimiento está diseñado para usar múltiples cámaras. La fusión de los datos obtenidos de las diversas cámaras está basada en la distancia de Mahalanobis. Este sistema se caracteriza por realizar un seguimiento multi-cámara, pero no realiza una interpretación sobre las actividades de la escena. Los autores han empleado el filtro de Kalman. Como podemos ver, este método es uno de los más usados en la vigilancia visual para realizar el seguimiento y el reconocimiento de objetos.

Otro ejemplo de un sistema, diseñado a principios del nuevo milenio, para reconocer patrones de comportamiento inusuales sobre peatones y vehículos en espacios abiertos es DETER [149] [116]. Deter lleva a cabo la detección de acontecimientos para la evaluación y el

reconocimiento de amenazas, como la detección de movimiento, el exceso de velocidad y la detección de patrones de movimiento anteriormente definidos. El sistema consta de dos partes: el módulo de visión y la evaluación de amenazas o módulo de gestión de alarmas. El primero, realiza la detección, el reconocimiento y el seguimiento de objetos a través de las cámaras, el cual se caracteriza por ser multi-cámara. La segunda parte, consiste en el reconocimiento semántico de alto nivel, un entrenamiento offline y una clasificación de amenazas online.

Unos años más tarde, en el 2006, se presenta un nuevo algoritmo en [25] para la detección y el seguimiento personas y vehículos basado en el uso de identificadores de creencia. Emplea una estrategia de fusión temporal en la que se usa la historia de los eventos acontecidos con el fin de mejorar las decisiones instantáneas. Además, se utilizan unos indicadores de creencia normalizados y actualizados en cada frame que resumen la historia de eventos específicos. El control de actualización de cada pixel está basado en un indicador de estabilidad estimado a partir de las variaciones entre frames. El algoritmo de seguimiento usa una propuesta basada en regiones. Existe un indicador de creencia que representa la consistencia de seguimiento para cada objeto y permite resolver ambigüedades. También hay un segundo indicador que representa la calidad de identidad de cada objeto. Los indicadores permiten propagar incertidumbres sobre los niveles más altos de la interpretación. Este modelo presenta buenos resultados, pero no realiza ninguna interpretación de la escena ni detección de anomalías.

En ese mismo año, en [81] se propone un nuevo sistema para analizar el vídeo de tráfico en carretera y generar una alarma cuando se detecte una situación anormal. Este proceso se realiza en 3 etapas. Primero, tiene lugar la inicialización del sistema donde se determina la región de interés de la escena y se lleva a cabo la calibración de la cámara para eliminar el efecto de perspectiva de la imagen. La segunda etapa realiza el proceso de detección y seguimiento de vehículos. En la tercera etapa, se analizan las actividades realizadas por los vehículos. En esta última fase, ellos consideran 6 situaciones: movimiento lento, parada, accidente de tráfico debido al choque entre vehículos, conducción al revés, etc.

Dentro de la monitorización del tráfico, existe una línea de investigación basada en la idea de que el estudio del comportamiento del peatón puede evitar situaciones peligrosas. Por ejemplo, en ese mismo año, se desarrolla otra aplicación que se muestra en [122], con el propósito de llevar a cabo un procesamiento de imágenes para detectar situaciones en las que las personas pueden encontrarse en peligro o realizando movimientos sospechosos. Esta investigación se basa en el estudio de las posiciones y las velocidades de los viandantes que son detectados en la escena para detectar diferentes situaciones: el acceso a un área protegida, un peatón que se mueve a gran velocidad o el merodeo de peatones que pasan mucho tiempo en una determinada zona. Cuando la aplicación detecta uno de estos incidentes, atrae la atención de personal de seguridad. Tiene dos grandes inconvenientes, por un lado, el sistema no es robusto frente a los cambios de iluminación y la aparición de sombras en la escena, lo que conlleva la detección de falsas personas, y por otro lado, tampoco distingue entre peatones que van corriendo o en bicicleta.

NerveCenter [37] es otro sistema para vigilancia, presentado en el 2007. Se centra en analizar la presencia de diferentes situaciones concretas: vehículo a gran velocidad, vehículos que estacionan en zonas prohibidas, personas merodeando... Para ello, se usa un sistema basado en reglas. Estas reglas son definidas previamente por el usuario usando un esquema XML. Las reglas definen los eventos que se van detectar. Cuando el sistema descubre la existencia de comportamientos sospechosos previamente definidos en las reglas, emite una alarma. El usuario puede recibir la alarma mediante distintos métodos (sms, fax, mail, por teléfono...). Lo novedoso es que ante la alarma, realiza un proceso de respuesta, mediante el que intenta resolver dicha situación, asignando tareas al personal y realizando un seguimiento de la respuesta desde el centro de mandos. Cuando se notifica una alarma, se usa un mapa geográfico para su localización.

Muchos de los sistemas de monitorización del tráfico que hemos citado *están diseñados para obtener información sobre los diferentes parámetros de tráfico*, por ejemplo, el número de vehículos por unidad de tiempo, la clasificación de vehículos, el promedio velocidad, la velocidad

individual de cada vehículo, etc. Algunos trabajos, como [147], se han llevado a cabo para gestionar inteligentemente la información que se muestra a los conductores, en función de las condiciones de la carretera o de las circunstancias del tráfico.

En los últimos años se está despertando un nuevo interés en desarrollar herramientas para la vídeo-vigilancia del tráfico con otros fines. Por un lado, *se comienzan a desarrollar sistemas con el objetivo de detectar infracciones* y así poder sancionarlas. Por otro lado, *se innova en la creación de herramientas que alerten de peligros que puedan ocasionar accidentes de tráfico*, con el objetivo de evitar una situación lamentable en la que una persona pueda resultar herida de gravedad. Aún así, estos campos de aplicación son más novedosos y todavía no existen muchos trabajos en este ámbito.

Uno de los sistemas más recientes que podemos encontrar en la literatura especializada para la monitorización del tráfico se describe en [47] (año 2008). En él se combina la monitorización de tráfico tradicional (donde se obtiene información de diferentes parámetros sobre el tráfico) con la monitorización para detectar accidentes de forma automática. Se trata de un sistema con conocimiento dirigido capaz de controlar el tráfico en carreteras o autopistas en una sola dirección. En dicho sistema, se lleva a cabo un proceso de segmentación de la imagen. Comparando una imagen segmentada con la anterior, se determina el movimiento de los vehículos en la escena. También se realiza una clasificación de vehículos según su categoría (coche, moto, vehículo pesado...). Los incidentes que se detectan son: presencia de un objeto extraño en la carretera, vehículo viajando en el arcén, vehículo pesado viajando en zona restringida, vehículo a gran velocidad o que viaja extremadamente despacio, vehículo parado, congestión y movimiento extraño de un vehículo.

Un año más tarde, otro trabajo, con un fin similar al anterior, es descrito en [19]. Este sistema realiza un seguimiento de vehículos en la autopista para detectar anomalías en el tráfico y alertar con una alarma al vigilante de seguridad. La aplicación se basa en tres procedimientos principales. El primero inicializa el sistema, detecta la región de interés de la escena, y realiza la calibración de la cámara para eliminar el

efecto de perspectiva de la imagen entrante. El segundo realiza el seguimiento de los vehículos en tiempo real. En el tercer procedimiento, se analizan las actividades de los vehículos a partir de una serie de situaciones normales predefinidas que pueden pasar en la autopista. En este módulo se comprueban la información detallada de cada vehículo y la información estadística global para identificar cualquier caso anormal (basándose en un estudio sobre la normalidad), y consecuentemente disparar una alarma.

Muchos de los accidentes de tráfico son originados por el choque de dos vehículos. Por lo tanto, un aspecto importante es el estudio de la probabilidad de colisión entre objetos en zonas de tráfico, como realizan los autores en [86] (año 2007), que presentan una investigación para la prevención de las colisiones en la autopista. Este trabajo se basa en un modelo probabilista que predice la probabilidad de accidentes en las autopistas, en función de las condiciones de flujo de tráfico, con el fin de ser aplicado para la prevención de accidentes en tiempo real. El modelo está basado en los registros y datos obtenidos en la autopista de Gardiner, en Toronto, durante 13 meses. Un inconveniente que presenta este modelo es la necesidad de volver a realizar un nuevo análisis estadístico si se pretende cambiar el entorno de estudio. Para que realmente este método sea aplicado en la prevención del accidente, debe establecerse una relación entre el nivel de probabilidad de que exista un posible accidente y la intervención de algún medio de seguridad en tiempo real.

La mayoría de las investigaciones cuyo objetivo es realizar un seguimiento de los objetos para la detección automática de accidentes se centran sobre todo en detectar las irregularidades llevadas a cabo por los vehículos. Sin embargo, no se puede dejar atrás el estudio del comportamiento del peatón, ya que éste puede verse involucrado de una manera u otra en muchos accidentes de tráfico.

El atropello de peatones es un problema actual debido a que muchas personas mueren o son heridas diariamente en zonas de tráfico. Sin embargo, poco se sabe acerca de la exposición de los viandantes al riesgo de colisión, especialmente cuando se compara con la cantidad de conocimiento disponible para el tráfico motorizado. Este hecho demanda

más conocimiento y análisis acerca del peatón para estudiar los eventos que involucran a los peatones en las colisiones. Las estadísticas de colisiones en sí, no son suficientes para el estudio de atropello.

En el 2009, en [72] se presenta un sistema de vídeo-vigilancia que puede: a) detectar y realizar un seguimiento de los usuarios de la carretera en una escena de tráfico y los clasifica como peatones o usuarios de vehículos motorizados de carretera, b) identificar los eventos importantes que pueden dar lugar a colisiones, y c) calcular la gravedad de varios indicadores de conflictos. El sistema tiene por objeto clasificar los eventos importantes y los conflictos de forma automática, pero también se puede utilizar para resumir grandes cantidades de datos que pueden ser revisadas por expertos en seguridad. En este procesamiento se usan reglas simples para detección de los indicadores de riesgo.

A pesar de que la mayoría de las últimas investigaciones encontradas en la literatura específica se centran en el análisis del comportamiento de objetos, éstas son todavía escasas, queda mucho por avanzar en esta etapa de vigilancia. A continuación, en la sección 2.4, veremos nuevas investigaciones para la monitorización del tráfico centradas en la fase análisis de situaciones, con el reto de usar otras fuentes además del conocimiento adquirido a partir del vídeo.

Por otro lado, *aunque las tareas de detección, seguimiento y clasificación de objetos son las más estudiadas, aún necesitan reducir su tasa de errores y la ambigüedad de ciertas situaciones*. Muchos algoritmos de seguimiento suelen fracasar cuando se analizan situaciones complejas. Por ello, en la actualidad surgen nuevos trabajos centrados en estas etapas iniciales del proceso de vigilancia inteligente, con el fin de mejorar las soluciones existentes y conseguir resultados más exactos con la realidad en un tiempo de procesamiento mínimo. En este aspecto, encontramos investigaciones como [30, 142], publicadas en 2009.

En [30] los autores proponen una red neuronal difusa para la detección, seguimiento y clasificación de vehículos en una autopista. El sistema realiza el conteo de vehículos.

En [142], se presenta una extensión de un sistema de seguimiento estándar que utiliza un marco de conocimiento específico sobre el

entorno para resolver los problemas del seguimiento general. Existe una capa que representa el conocimiento conocido sobre el contexto del escenario de análisis. Este conocimiento está definido por un conjunto de reglas. La representación del conocimiento del contexto y los métodos de razonamiento son generales y pueden ser fácilmente adaptados a diferentes escenarios. Este método basado en contexto mejora los resultados del seguimiento sin sobrecargar los tiempos de procesamiento.

Otro trabajo más reciente, publicado en el 2010, que realiza un seguimiento de vehículos es descrito [93]. En este caso, los autores presentan un algoritmo detección y seguimiento basado en la técnica de sustracción de fondo. El objetivo que persiguen y consiguen es mejorar el seguimiento aún en malas condiciones de iluminación, congestión...

Como podemos observar, las técnicas de procesamiento de imágenes son clave para el desarrollo de Sistemas de Vigilancia Visuales. Un resumen sobre técnicas de procesamiento de vídeo para aplicaciones de tráfico se puede encontrar en [150].

2.3.3. Sistemas de Vídeo-Vigilancia en el Área de Espacios públicos

A pesar de que la monitorización del tráfico es el campo de estudio que ha causado un mayor interés para la aplicación de vigilancia inteligente y, por lo tanto, el más explotado, también son muchos los Sistemas de Vigilancia diseñados para mantener la seguridad en espacios públicos, como estaciones de tren, aeropuertos, museos, edificios emblemáticos, centros comerciales...

El aumento de la accesibilidad de las personas a espacios públicos y la dependencia de su calidad de vida en las infraestructuras ha aumentado el riesgo y la necesidad de proteger a esos mismos ciudadanos antes nuevas amenazas.

Durante estos últimos años la vigilancia en el transporte público ha sido especialmente demandada. Este hecho queda reflejado en una gran cantidad de proyectos financiados por el estado para mantener

el orden y la seguridad en las estaciones del transporte de viajeros. En estos ambientes se realizan tareas que necesitan seguridad, por ejemplo: el progreso entre trenes, la regulación de velocidad, el peligro de choque... Además, debido a la posibilidad de ataques terroristas, el estudio del posible abandono de objetos extraños y peligrosos requiere especial interés. Por estas razones, se han desarrollado diversos sistemas de vídeo-vigilancia en el transporte público, con el fin de obtener una protección contra riesgos accidentales o intencionados, y aumentar la seguridad de los pasajeros.

Entre los primeros prototipos de los 90, encontramos en [20] un sistema cuyo objetivo es detectar situaciones peligrosas, por ejemplo, el vandalismo en una estación de metro y avisar al operador con una señal de alarma. El sistema es capaz de reconocer personas y otros objetos y realizar un seguimiento de los mismos. La dificultad principal está asociada a la gran variabilidad de ambos en escenas reales.

Otro sistema de la época de los 90 que realiza la detección y el seguimiento de objetos en espacios abiertos se presenta en [49]. Este modelo usa una nueva medida 'spectrum', que indexa una gran cantidad de imágenes que contienen diferentes vistas de los objetos. La limitación del sistema es su dependencia de las condiciones de iluminación y de presencia de sombras que provocan una posible detección objetos falsos.

En [54], se describe un nuevo sistema que concretamente realiza una detección y un seguimiento de personas en entornos abiertos, pero las capacidades del mismo también son limitadas.

Al igual que en las aplicaciones en el área de tráfico, vemos que en estas tempranas investigaciones, sólo se realiza detección y seguimiento de objetos, sin llevar a cabo un proceso de razonamiento posterior donde se interpreten los hechos acaecidos en los espacios vigilados.

En el año 98, se presenta una arquitectura para un sistema de vigilancia en un puerto marítimo [120]. La arquitectura está basada en el diseño cliente/servidor. Existen dos subsistemas, uno para la adquisición y otro para la visualización de imágenes. El subsistema de adquisición tiene un módulo servidor que es capaz de manipular hasta cuatro cámaras al mismo tiempo. Este módulo se encarga de recoger

las imágenes de las cámaras y transmitirlas a la red mediante TCP/IP. También es el encargado de almacenar las imágenes en discos duros. El módulo de visualización es llevado a cabo por los clientes, los cuales podrán ver las imágenes de las cámaras por internet. Se trata de un sistema que únicamente realiza: adquisición, distribución y visualización de imágenes, sin llevar a cabo ningún proceso de análisis sobre la escena. El punto de interés de este trabajo es el uso de la arquitectura cliente-servidor para la manipulación de imágenes digitales. El sistema es capaz de soportar más de 100 clientes y 100 cámaras al mismo tiempo.

Otro campo relevante donde inicialmente se aplicaron este tipo de sistemas es en estaciones de tren, donde encontramos trabajos como [126, 139, 140], publicados en la década de los 90. El estudio expuesto en [139] destacó por resolver un hecho puntual de cierta importancia: detectar objetos abandonados en entornos desatendidos (salas de espera, estaciones de trenes...). Los autores usan cámaras de TV monocromáticas de donde obtienen el flujo de imágenes que se procesan. Se emplean técnicas de recuperación basadas en contenido y técnicas del procesamiento de imágenes. Se consigue una probabilidad de éxito del sistema de un 89.9 %, entendiendo como éxito la detección correcta del abandono de un objeto.

Un trabajo posterior, publicado en el año 2000, y siguiendo la misma línea, fue presentado en [26]. En él se muestra un sistema de vídeo-vigilancia distribuido para la detección de situaciones peligrosas ante la presencia de objetos abandonados en las salas de espera de las estaciones de ferrocarril. No está limitado a este tipo de entornos, también se podría aplicar a parkings, aeropuertos.... Cuando un objeto es reconocido como objeto abandonado, el sistema transmite una alarma al centro de control remoto. Se realiza una comunicación multimedia basada en técnicas de acceso múltiple de división de código en secuencias directas (direct sequence code-division multiple-access: DS/CDMA) para asegurar una transmisión inalámbrica, segura y robusta al ruido.

Una aplicación más reciente, ya en el 2008, que detecta objetos que han sido abandonados se puede encontrar en [119]. Se presenta un método que emplea planos duales para extraer regiones de imagen estáticas temporalmente. Dependiendo de la aplicación, estas regiones

indican los objetos que no constituyen el fondo original, pero fueron traídos a la escena en un tiempo posterior, como artículos abandonados y quitados o vehículos ilegalmente aparcados. Se usan modelos Gaussianos multivariables para construir los fondos a corto plazo y largo plazo. Los parámetros de fondo son adaptados usando un mecanismo de actualización Bayesiano. Comparando cada frame con estos modelos, se estiman dos primeros planos (fondos). Se deduce una puntuación o evidencia para cada píxel aplicando un conjunto de hipótesis y comparando sobre el primer plano. Posteriormente se agrega la evidencia para proporcionar la consistencia temporal. A diferencia de las propuestas basadas en el flujo óptico que estropean las fronteras, el método propuesto puede segmentar con exactitud los objetos incluso si son totalmente son ocluidos.

En [127], en el año 2000, se presenta un sistema de vigilancia para estaciones de ferrocarril en Italia basado en cámaras CCTV. El sistema tiene una arquitectura jerárquica y distribuida entre los puestos de control principales (central de vigilancia) y los puestos secundarios (estación). Estos últimos se encargan de recoger las imágenes de las cámaras y enviarlas a los puestos centrales, donde se capturan y se visualizan. Este sistema no realiza una detección y un seguimiento de objetos, lo que hace que el papel de un operador humano sea crucial.

En relación con la seguridad en las estaciones de tren, se puede leer en [153] un resumen técnico de *sistemas comerciales* con fines al campo ferroviario.

La Unión Europea ha financiado algunos proyectos dentro del área de investigación de la seguridad en el transporte público. Por ejemplo, en el año 99, CROMATICA [43] [60] *(CROwd MAnagement with Telematic Imaging and Communication Assistance)*, con el objetivo principal de mejorar la vigilancia de pasajeros en el transporte público, permitiendo al empleo y la integración de tecnologías como la detección basada en el análisis de vídeo y la transmisión inalámbrica. El fin del sistema es medir continuamente el flujo de personas en estaciones de metro para descubrir las condiciones anormales (p.ej., el abarrotamiento, patrones de movimiento inesperados, colas) y prevenir situaciones peligrosas relacionadas con caídas, actos vandálicos, ataques personales, etc., que podrían causar problemas serios a un gran número de pasajeros.

Otro proyecto europeo paralelo, con el fin de reforzar la seguridad del transporte urbano por ferrocarril, es AVS-PV [6] *(Advanced Video Surveillance-Prevention of Vandalism in the Metro)*. El objetivo principal de este proyecto es descubrir los comportamientos que son típicos para "vándalos potenciales" en las estaciones de metro. El sistema de procesamiento de imágenes utilizado permite analizar e indicar algunos comportamientos, por ejemplo:

- "Comportamientos extraños" como el que una persona sola que permanezca de modo anormal durante mucho tiempo en el mismo lugar sin tomar un tren;
- "Comportamientos de grupo" como el distinguir que un número de personas están actuando en grupo (sin que formen un grupo de forma visual);
- "Comportamientos inquietos" de una persona sola o de un pequeño grupo de personas.

Una investigación posterior en la misma línea de AVS-PV es descrita en [27] (año 2001). Se presenta un sistema para detectar actividades anormales en estaciones de metro, usando un modelo basado en una red neuronal para clasificar dichos comportamientos. Las redes neuronales se han seguido usando en este tipo de sistemas, en [5] encontramos un sistema que usa también un clasificador basado en una red neuronal. Los autores se centran en el paradigma de que el conocimiento de la escena no genera una alarma, pero sí lo hace el desconocimiento de la misma.

ADVISOR *(Annotated Digital Video for Intelligent Surveillance and Optimised Retrieval)* [135], es otro sistema financiado con fondos europeos para la vigilancia de estaciones de metro subterráneas. El propósito del sistema es ayudar a los operadores humanos en la selección automática, grabación y almacenamiento de las imágenes que tienen acontecimientos de interés. En otras palabras, ADVISOR interpreta formas y movimientos de las imágenes captadas por CCTV para estudiar las actividades de las personas en la escena. El sistema lleva a cabo un proceso de anotación de vídeo, donde quedan reflejados los eventos

acontecidos. De esta forma, se pueden hacer consultas sobre los datos anotados y recuperar secuencias de vídeo. ADVISOR consiste en:

1. Una red de unidades (cada una de las cuales es instalada en una estación diferente y lleva a cabo una detección de objetos).
2. Un módulo de reconocimiento.
3. Un módulo para realizar el análisis conductual.
4. Un módulo de almacenamiento y recuperación de datos.

El sistema tiene una arquitectura abierta escalable (ver figura 2.1) y emplea hardware estándar comercial. A pesar de ello, puede ser considerado como un semi-distribuido, ya que puede ser visto como una red de nodos de procesador independientes dedicados (unidades). En cada nodo, el número de CPUs es directamente proporcional al número de módulos de procesamiento de imágenes existentes, lo que hace que el sistema sea difícil de escalar y de gran coste.

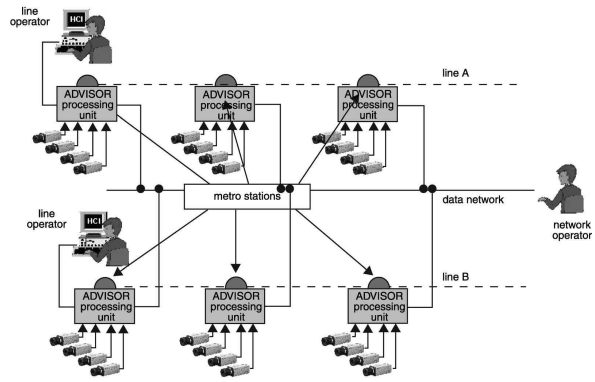


Figura 2.1: Arquitectura del Sistema Advisor

Un proyecto más reciente, también financiado por la Unión Europea, es AVITRACK [40, 111]. El objetivo de este proyecto es ayudar a mantener la seguridad en los aeropuertos. Para ello, el sistema analiza el comportamiento de personas y vehículos en dicho entorno. Concretamente, se centra en el control de la carga y descarga de mercancías en aviones, el repostaje de combustible...

Volviendo al campo de aplicación de los entornos marítimos, encontramos un sistema llamado ARGOS [17], que lleva implantado desde el 2007 en Venecia, para el control del tráfico marítimo. Esta herramienta estudia la velocidad de los barcos para la detección de aquellos que van a una velocidad superior a un determinado umbral. También identifica los barcos que se mueven en paralelo y muy próximos entre sí durante un largo tiempo, detecta si hay objetos que toman direcciones equivocadas en canales de un solo sentido y alerta de la existencia de barcos detenidos en áreas prohibidas.

Actualmente, siguen surgiendo nuevas investigaciones centradas en mantener la seguridad de los espacios públicos. Como veremos a continuación (ver sección 2.4), los nuevos proyectos pretenden nuevos retos, entre ellos, el uso de diferentes tipos de sensores para la monitorización del espacio, con el fin de complementar la información obtenida a partir del análisis del contenido del vídeo.

2.3.4. Sistemas de Vídeo-Vigilancia en otros ámbitos de aplicación

A finales del siglo XX, la monitorización del tráfico y la vigilancia de estaciones públicas dejan de ser el centro de estudio para Sistemas de Vigilancia y surgen nuevos campos de aplicación que requieren seguridad.

Existen sistemas que *detectan fuego a partir de un proceso de análisis de imágenes* de un vídeo [155]. En [146] (año 2003) se propone un método inteligente de detección de fuego en tiempo real, con el fin de usarlo como sistema de vigilancia para emitir una alerta que pueda prevenir un incendio.

Un año después, en [91] se muestra una aplicación de vigilancia visual para el *área de deportes*. El sistema consiste en ocho cámaras, ocho módulos servidores de características y un proceso de multiseguimiento. Las cámaras están instaladas sobre el área de juego, y las imágenes se envían por fibra óptica a distintos módulos servidores. Cada uno de éstos, realiza la detección, el seguimiento y la clasificación de objetos, y envía los resultados al módulo de seguimiento multi-cámara, que combina toda

la información usando el método del vecino más cercano, basado en la distancia de Mahalanobis.

No todos los Sistemas de Vigilancia están diseñados sólo para espacios abiertos. Existe un amplio abanico de sistemas cuyo fin es su uso en espacios cerrados o interiores. Los entornos exteriores e interiores tienen diferentes tipologías, lo que puede dar pie a diferentes algoritmos o niveles de implementación.

En el mundo académico, encontramos algunos *sistemas enfocados a su uso en espacios cerrados*. Por ejemplo, en el año 2000, en [78] se presenta un método de seguimiento multi-cámara en entornos interiores. Este modelo es incluido en un sistema de ambiente inteligente llamado 'EasyLiving' con el fin de ayudar a los inquilinos del entorno analizando sus actividades. Se consigue muy buenos resultados si hay menos de tres personas en la habitación y si llevan ropas de diferentes colores; en cambio, en otras situaciones los resultados son bastante pobres.

Otro sistema para usos interiores es CCN [71] (Co-operative Camera Network) (año 2002). Consiste en una red de nodos, donde cada uno está compuesto de una cámara PTZ conectada a un ordenador personal y una consola central, que se maneja por un operador humano. El sistema recoge la presencia de un individuo etiquetado dentro del edificio. El objetivo del sistema es detectar y supervisar a ladrones dentro de tiendas. Esta herramienta presenta una limitación, el sistema está condicionado por el número de objetos presentes en la escena, si este el tráfico de gente es escaso ofrece buenos resultados. En cambio, cuando existen muchos individuos el sistema presenta una alta tasa de errores.

Un año más tarde, en [94] se describe un nuevo sistema que realiza un seguimiento multi-cámara en entornos interiores. Consiste en una red de módulos de procesamiento de cámaras y un módulo de control que tiene una Base de Datos con los objetos actuales en la escena. También usan el Filtro de Kalman para el seguimiento. Además, los autores han construido un algoritmo que divide el seguimiento en tareas entre las cámaras, con el fin de asignar el proceso de seguimiento a la cámara que tenga mejor visibilidad del objeto en un momento dado (para la elección de la cámara usan una función que se basa en el tamaño de los objetos

en la imagen). Esta técnica potencia el procesamiento distribuido para mejorar la detección llevada a cabo.

Otro campo de aplicación que actualmente está despertando gran interés es la *vídeo-vigilancia en el hogar*, ya que puede ofrecerse como herramienta de ayuda a las personas, sobre todo a personas mayores o discapacitados que viven solos y con una limitada autonomía. En el año 2007, en [36] se presenta un modelo de visión multi-cámara para la detección y seguimiento de personas. El sistema identifica actividades que pueden ser peligrosas, como por ejemplo una caída. En estas situaciones el sistema crea una alarma que puede ser enviada mediante un mensaje al móvil (sms).

En ese mismo año, en [131] se presenta un sistema que puede ser empleado en cualquier entorno vigilado con diferentes cámaras. El sistema propuesto permite la visualización mediante dispositivos móviles de los vídeos provenientes de distintas cámaras de vigilancia. En este caso, el sistema no realiza ningún tipo de análisis de vídeo.

Uno de los trabajos más recientes es [118] (año 2008), donde se describe una metodología para la detección y el seguimiento de personas en secuencias de vídeo en tiempo real. Se emplean dos cámaras estáticas y una cámara de infrarrojos montada sobre un sistema robotizado, lo que hace posible aplicar esta técnica propuesta a imágenes adquiridas por condiciones diferentes y variables. Igualmente, permite una filtración a priori, basada en características de tales imágenes para dar más evidencia a los objetos que emiten un resplandor más alto (p. ej., la temperatura más alta).

En el sistema anterior, el reconocimiento del objeto durante el seguimiento se realiza usando una red neuronal artificial jerárquica. Este sistema tiene una arquitectura modular que posibilita la introducción de nuevas características, incluyendo nueva información útil para un reconocimiento más exacto. Este tipo de arquitectura modular permite la reducción de la complejidad local y, al mismo tiempo, la implementación de un sistema flexible. En caso de búsqueda automática de un objeto enmascarado u ocluido, se usa un paradigma de recuperación basado en contenidos para la recuperación y la comparación de las carac-

terísticas actualmente extraídas.

La vigilancia es otra actividad también aplicada en el *campo militar*, como podemos ver en [44], donde se pretende detectar movimientos anómalos en entornos militares. Otro trabajo dentro de esta temática es descrito en [114]. En él se presenta un sistema que establece relaciones espacio-temporales y las analiza, con el objetivo de predecir futuros movimientos de los objetos de la escena.

Por otro lado, la protección de edificios emblemáticos, museos o incluso de propiedades privadas es cada día más demandada. Sin embargo, es menos común encontrar en el mundo académico investigaciones que se centren en la tarea concreta de *detección de intrusos*. En cambio, en los sistemas comerciales las tareas más comunes de procesamiento son la intrusión y detección de movimiento [63, 68] y paquetes de detección [67–69].

Uno de los trabajos encontrados en la literatura donde se propone un sistema multi-agente para la detección y el seguimiento de intrusos es descrito en [115]. Otro sistema de monitorización visual para detectar cierto el acceso a zonas restringidas en escenas dinámicas puede verse en [156]. Este sistema consiste en la detección y el seguimiento de objetos y el reconocimiento de ellos como personas y vehículos...

Otra propuesta para la detección de intrusiones en zonas restringidas, basada en un sistema de reglas difusas, es descrita en [46]. Se trata de un sistema que puede ser instalado en una cámara de vigilancia.

2.4. Sistemas de Vigilancia basados en el conocimiento obtenido a partir de fuentes heterogéneas. Aplicaciones.

Actualmente, en muchas situaciones, existen circunstancias que son difíciles de detectar a partir de la información recogida por cámaras de vigilancia. En estos casos, añadir otro tipo de sensores que pueda recoger dicha información y que complemente al conocimiento adquirido del análisis de vídeo es vital. Por estas razones, la nueva generación

de Sistemas de Vigilancia demanda desarrollos que puedan analizar más fuentes de información a parte del vídeo. Este tipo de aplicaciones, junto con los sistemas multi-cámara, constituye la tercera generación de sistemas.

Los sistemas que usan información heterogénea procedente de diversas fuentes son conocidos también como sistemas *multi-sensor*, ya que usan la información obtenida de diferentes sensores ubicados en el entorno vigilado. En este caso, los sensores no tienen que ser únicamente cámaras de vigilancia, sino que también pueden ser: micrófonos, sensores de movimiento, detectores de humos, semáforos. . . .

Recordamos que en la literatura específica, el término multi-sensor ha sido ampliamente utilizado para referirse a aquellos sistemas que emplean varias fuentes de vídeo distintas (también conocidos como sistemas multi-cámara). No obstante, en esta sección, cuando hablemos de sistemas multi-sensor, nos referiremos a sistemas que usan distintos tipos de sensores (cámaras, micrófonos, infrarrojos, sensores de movimiento, de humos...), como se ha citado anteriormente.

El hecho de usar conocimiento de diferentes fuentes para la adquisición de datos sobre el entorno, puede dar lugar a sistemas más potentes, ya que disponen de una mayor cantidad y diversidad de información acerca de una escena. Este hecho repercute positivamente en el análisis de situaciones de riesgo.

La integración de información heterogénea es un proceso tedioso. Este es el principal motivo por el que aún no existen muchas aplicaciones de este tipo. Estos sistemas demandan un pre-análisis del conocimiento para poder realizar una buena integración de la información, donde se llevan a cabo la aplicación de técnicas de fusión de datos [11, 38].

A principios del nuevo milenio comienzan a surgir sistemas con fuentes heterogéneas. En el año 2002, se publica un trabajo en [53], en el que se describe una red de sensores distribuida para procesos de vigilancia con el fin de construir un sistema capaz de usar sensores heterogéneos (ópticos, infrarrojos, radar. . .) y fusionar la información de las distintas fuentes. El sistema se centra en el seguimiento de objetos para identificar sus trayectorias. La arquitectura propuesta permite que

sea fácilmente escalable y que su distribución en un entorno exterior de área amplia no sea compleja. Los autores afirman que el sistema ofrece buenos resultados, incluso de noche y en condiciones atmosféricas adversas.

Dos años más tarde, en una investigación posterior [83], se define el sistema anterior ya en un estado más avanzado, capaz de dar soporte durante las 24 horas del día, integrando la información de los distintos sensores ópticos e infrarrojos. Esta integración ofrece una mayor precisión sobre la localización del objeto.

Por otro lado, el audio es una fuente menos explotada que el vídeo, pero que en los últimos años se está aplicando en el desarrollo de sistemas de seguridad. La vigilancia podría verse complementada si existen sistemas que sean capaces de identificar sonidos asociados a actividades que pueden llegar a ser peligrosas. Además, muchas situaciones son más fáciles de identificar mediante eventos de sonido que mediante un análisis de vídeo, como es el caso de la identificación de un disparo [31].

Entre los diferentes sistemas que analizan audio, cada día son más los diseñados con el fin ser integrados en un proceso de vigilancia. Por ejemplo, en [35] (año 2004) se muestra una aplicación que analiza el audio que se percibe en un entorno y lo aprende, para poder detectar sonidos inusuales o diferentes a los que normalmente se dan en el ambiente estudiado. La detección de sonidos anormales se hace en tiempo real y puede aplicarse sobre diferentes situaciones. Tiene la limitación de que sólo puede usarse un micrófono.

Una año más tarde, en [124] se muestra otro análisis de audio para aplicaciones de vigilancia. Los autores se basan en que un sistema de vigilancia para la detección de eventos de sonido no se puede completar sin un marco de clasificación de audio supervisado. Por ello, proponen una solución híbrida con dos partes: una, donde se realiza un análisis de audio no supervisado y otra, donde se lleva a cabo un análisis basado en una clasificación de audio que fue obtenida tras el entrenamiento y el análisis de audio offline.

Otra propuesta de detección de eventos de sonido para vigilancia

es descrita en [12] (año 2006). En este trabajo se presenta un sistema capaz de distinguir entre sonidos vocales y no vocales y, dentro de esos grupos, los clasifica entre anormales o no.

Un proyecto dentro de este ámbito de investigación que incorpora subsistemas de análisis de audio, financiado por la Unión Europea a principios del siglo XXI, fue PRISMATICA ("PRo-active Integrated systems for Security Management by Technological, Institutional and Communication Assistance") [113, 134]. Este proyecto surge como continuación del proyecto CROMATICA, citado y brevemente descrito en la sección 2.3.3.

En PRISMATICA se presenta un sistema de vigilancia distribuido multi-sensor, que recibe información a través de redes de cámaras inalámbricas, cámaras CCTV, tarjetas de identificación y sensores de sonido. Consiste en una red de dispositivos que procesan las salidas de los sensores y envían/reciben mensajes a/desde un módulo central (llamado MIPSA). Este módulo gestiona la actividad de los dispositivos, recupera datos y establece el punto de contacto con un operador humano.

PRISMATICA (ver figura 2.2) usa una arquitectura estándar y escalable empleando hardware comercial. A pesar de que es clasificado como sistema distribuido, emplea una propuesta centralizada, ya que tiene una unidad central que controla y supervisa al resto y, un fallo en esta unidad, provocaría un fallo del sistema.

Algunos de los eventos detectados en PRISMATICA son: detectar que una persona se detiene en un pasillo, intrusiones, colas de espera con un tamaño anómalo... El sistema está diseñado para tener un uso específico en entornos interiores.

Por otro lado, en [89] se muestra el diseño de un sistema de vigilancia sin el servidor, para evitar la centralización dada en PRISMATICA. Esto hace que todos los subsistemas sean independientes y completamente autónomos. Posteriormente, se conectan todos estos nodos, el uno con el otro sin tener un punto de comunicación mutuamente compartido. Esta propuesta evita las desventajas del servidor centralizado y mueve todos los procesos directamente a la cámara haciendo del sistema un grupo de cámaras conectadas a través de la red. Pero para llevar a cabo

esta propuesta, es necesario construir un hardware específico comercial en el que cada unidad consta de una cámara, un procesador, un adaptador de redes y una base de datos. Por lo tanto, en aplicaciones sensibles al coste donde se requiera un número elevado de cámaras, esta propuesta podría ser inadecuada.

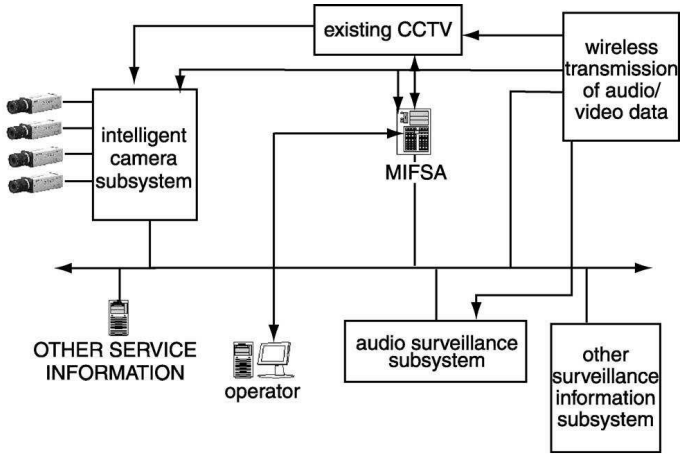


Figura 2.2: Arquitectura del Sistema PRISMATICA

En [11] se lleva a cabo una fusión de información sobre audio y vídeo procedente de diferentes sensores del entorno. El sistema usa un modelo probabilístico para la detección de eventos simples (hablar, estar de pie, andar, correr, llamar a la puerta...) y compuestos ("estar de pie y correr", "hablar y andar"...) en procesos de vigilancia. Este sistema no realiza el procesamiento del vídeo ni de audio, simplemente recibe la información, la fusiona y produce el evento de salida, con la información ya integrada.

Otra aplicación multi-sensor más reciente y que, entre otros, usa sensores de sonido se puede encontrar en [110]. Este trabajo, publicado en el 2008, describe un sistema que detecta una caída dentro de un ambiente/edificio inteligente equipado con la multitud de sensores (de

sonido, de vibración, y PIR–passive infrared–). Los sensores alimentan un detector de actividad, que analiza simultáneamente sus salidas para obtener una decisión final. El detector está basado en Modelos de Markov, entrenados para reconocer actividades, regulares o anormales (incluyendo la caída), de personas mayores y animales domésticos, para cada señal de sensor.

Para mejorar la vigilancia de aeropuertos surge un nuevo proyecto llamado Co-Friend [59], como continuación del proyecto AVITRACK. Se caracteriza por usar una red de sensores heterogéneos, compuesta por cámaras PTZ y sensores de localización GPS. Se realiza una fusión de datos para lograr una detección robusta y eficiente sobre el seguimiento de objetos en escenas complejas, con el objetivo de llevar a cabo una interpretación posterior de las actividades humanas en situaciones reales.

Otro proyecto de investigación más reciente, financiado con fondos nacionales, es HESPERIA: Homeland sEcurity: tecnologíaS Para la sEguridad integRal en espacIos públicos e infrAestructuras [61] (CENIT I+D 2006-2009). Hesperia tiene por objeto el desarrollo de tecnologías que permitan la creación de sistemas punteros de seguridad, vídeo vigilancia y control de operaciones de infraestructuras y espacios públicos. El proyecto surge para dar respuesta a una demanda sostenida a medio y largo plazo, en particular, en países de la Unión Europea y en Estados Unidos. En este proyecto participaron numerosas empresas y universidades de España, entre ellas, la Universidad de Granada. De hecho, la presente tesis, está enmarcada dentro del proyecto HESPERIA.

Regresando al campo de monitorización del tráfico, aparecen nuevas investigaciones multi-sensor, por ejemplo [98,152], que combinan la información procedente del análisis del contenido del vídeo con información procedente de un sensor: el semáforo, que indica si el semáforo está en verde, rojo o ámbar.

En [98] (año 2006), se presenta un sistema multiagente para controlar el tráfico y detectar infracciones en un cruce con tres intersecciones y semáforos. Para el análisis de situaciones anormales en dicho entorno se usa un sistema basado en reglas difusas. El gran inconveniente es que

este sistema ha sido diseñado para proteger un espacio concreto y no es portable a otro dominio.

Otra investigación más reciente, ya en el 2009, dentro de este ámbito es descrita en [152]. Se presenta un sistema para la monitorización del tráfico con el fin de detectar infracciones. En este trabajo, los autores estudian el control del tráfico en cruces de peatones con semáforos. Para ello, proponen un sistema semi-automático que estudia los comportamientos de los objetos en la escena. El sistema presenta una arquitectura multi-agente y realiza un análisis basado en reglas para detectar los comportamientos anormales. Unas de las ventajas de esta herramienta es que es fácilmente escalable a otros dominios de aplicación (diferentes cruces de peatones). Sin embargo, presenta una carencia, el sistema no es robusto frente a información imprecisa, y en este ámbito, este item es cada día más importante, ya que, por ejemplo, entre otras cosas, la mayoría de los algoritmos de clasificación de objetos detectados a partir del vídeo suele obtener información con incertidumbre, lo que hay que considerar.

Dentro del campo de los sistemas de seguridad, para evitar accidentes de tráfico, encontramos varios trabajos, publicados en 2010, centrados en proporcionar un mejor conocimiento de las condiciones de las carreteras a los conductores para reducir así el número de accidentes y que la conducción sea más cómoda y fluida.

Por ejemplo, en [29] se presenta una propuesta basada en enfoque difuso para la detección de posibles colisiones entre vehículos. En este trabajo los autores usan información heterogénea proveniente de diversos sensores instalados en el vehículo, ente ellos cámaras CCD y sensor GPS. Los datos son analizados para producir una alerta cuando el vehículo tiene posibilidad de chocarse con un obstáculo.

Otro trabajo similar al anterior es el descrito en [145]. En este artículo se presenta un sistema instalado a bordo de los vehículos con el fin de notificar el riesgo de colisión entre vehículos en una autopista. El sistema también detecta el riesgo de atasco. Se hace uso de redes VANET para el paso de mensajes entre vehículos.

El desarrollo de sistemas multi-sensor es el campo de estudio

más reciente dentro de los sistemas de seguridad, y por ello, también menos explotado. Hoy día, es difícil encontrar sistemas que empleen el conocimiento de diversas fuentes. De este modo, surge la demanda de buscar nuevas soluciones que incorporen información más diversa y complementaria, que fortalezcan el proceso de vigilancia en cualquier área de aplicación: monitorización tráfico, seguridad en espacios públicos...

2.5. Anotaciones sobre el Estado del Arte

Tras la realización de este estudio sobre el estado del arte de los sistemas de vigilancia, observamos que existen muchos proyectos desarrollados e importantes avances en investigación. Sin embargo, encontramos varios aspectos de la tecnología que pueden ser mejorados. Actualmente, la Vigilancia Inteligente es una área de trabajo abierta en la que aún queda mucho por hacer y avanzar.

Hemos visto, en las secciones anteriores, que la mayoría de los Sistemas de Monitorización Inteligente están basados en el análisis del contenido de vídeo, siendo ésta la única fuente de información en los mismos.

El análisis del contenido del vídeo supone un gran esfuerzo en la aplicación de técnicas de Visión por Ordenador. Este hecho influye notablemente en la mayoría de los sistemas para vídeo-vigilancia encontrados en el mundo académico. Éstos otorgan el mayor peso de la investigación a la creación, prueba y mejora de algoritmos de procesamiento de imágenes, con el fin de realizar una identificación y un seguimiento de objetos en movimiento.

De este modo, *destaca el extenso número de trabajos que abarcan las primeras etapas de vigilancia*: detección, clasificación y tracking de objetos. A pesar de ello, siguen existiendo algunas *carencias*:

- Muchas de las soluciones propuestas fallan en el análisis de escenas complejas, donde el número de elementos que intervienen en la escena es alto. Este comportamiento conlleva un mal funcionamiento en escenas reales.

- Otro problema todavía por resolver, es la sensibilidad a las condiciones de iluminación en zonas abiertas, que en algunos casos origina la detección de objetos falsos.

- El tiempo de procesamiento, la proximidad y la oclusión de objetos siguen siendo inconvenientes a solucionar con rotundidad en la literatura.

- En algunos casos, la utilización de una sola cámara de vigilancia limita el sistema. Todavía existen pocos trabajos que realicen seguimiento multi-cámara.

Una de línea de investigación abierta es el desarrollo de nuevas aplicaciones que lleven a cabo un *proceso más robusto en estas primeras etapas*. Este hecho se produce por el aumento de la demanda de datos válidos que describan con suficiente exactitud la realidad de las escenas, y que además, sean obtenidos en tiempo real.

Los resultados analizados tras una detección, clasificación y seguimiento de objetos son esenciales para un posterior razonamiento sobre ellos. La calidad de los datos conocidos acerca de los objetos influye notablemente en el análisis de sus comportamientos.

En muchos entornos, es imprescindible la implantación de sistemas que realicen un seguimiento multi-cámara. Siendo éste un campo de gran interés en muchas de las investigaciones actuales.

Valorando la gran cantidad de trabajos que se centran en las primeras etapas de vigilancia, es mucho más escaso el número de sistemas que aplican técnicas de Inteligencia Artificial para incorporar una capa de razonamiento dando más énfasis a la fase de análisis del comportamiento de objetos. Esta etapa es vital en el proceso de vigilancia, porque los resultados de la misma pueden servir al operador como herramienta de ayuda en su trabajo, al poder ser avisado de ciertos riesgos en el entorno monitorizado.

La interpretación de las escenas a alto nivel para detectar peligros con el fin de ser avisados y, en muchos casos, evitados, es una de las líneas de investigación abiertas. Todavía queda mucho por avanzar en

este campo. Algunas de las *mejoras* que pueden realizarse en la etapa de análisis son:

- *La identificación de situaciones complejas*. En la literatura específica, la mayoría de los sistemas que abarcan la fase de análisis del comportamiento de los objetos, se centra en resolver situaciones muy simples. Esto es debido, en muchos casos, a que previamente han realizado un proceso de detección, clasificación y seguimiento de objetos complejo y la etapa de interpretación de la escena queda en un segundo lugar.
- *El desarrollo de sistemas más exportables*. Muchos trabajos se centran en su aplicación para dominios concretos. En general, las aplicaciones son diseñadas para identificar situaciones de riesgo en ambientes específicos. Si cambia el escenario de estudio, el sistema deja de funcionar o el proceso de adaptación del mismo sería tan costoso que hace inviable su portabilidad a otro entorno.
- *La creación de sistemas más escalables.* La mayoría de los sistemas han sido diseñados para resolver un propósito específico y no son escalables. En muchos casos, las aplicaciones son creadas para vigilar un aspecto concreto y si se desea ampliar el análisis y estudiar un nuevo aspecto independiente, nos encontramos con sistemas con arquitecturas muy cerradas, que no permiten la inclusión de nuevos módulos de razonamiento.
- Mejorar los *tiempos de procesamiento*, que en muchos sistemas de la literatura son demasiado altos, a pesar de que los resultados obtenidos puedan ser buenos. En los sistemas de vigilancia, que exijan una detección de riesgos en tiempo real, este requisito es crítico.

Por esta razones, la vigilancia inteligente continúa siendo una línea de investigación joven y abierta donde existen *demandas* centradas en:

- *Escalabilidad* en los sistemas, creando *arquitecturas flexibles* que puedan incorporar nuevos módulos de análisis para detectar nuevas amenazas.

- El desarrollo de *sistemas que analicen escenas complejas*, detectando anomalías, riesgos o peligros que puedan ocurrir en entornos reales.

- Búsqueda de herramientas que puedan ser *aplicadas sobre cualquier dominio*. Por ejemplo, si se desarrolla un sistema para la detección de objetos abandonados, éste debe poder ser aplicado sobre cualquier entorno, ya sea en una gasolinera, en una plaza, o en una sala de espera. Si existiese un proceso de adaptación del sistema al nuevo entorno, debería realizarse fácilmente.

- *Sistemas eficientes* en sus cálculos, es decir, que inviertan un tiempo de procesamiento mínimo que permita la detección de riesgos en tiempo real.

Por otro lado, si estudiamos la cantidad de trabajos relacionados con sistemas que se alimentan de información heterogénea, es decir, recogida por diversas clases de sensores, el número de los mismos desciende considerablemente.

Sobre estos últimos, se están desarrollando estudios que analizan audio para detectar sonidos cruciales en un entorno de seguridad. Aún así, es muy difícil encontrar sistemas de vídeo-vigilancia que realicen un análisis de audio que complemente la información visual. Esto se debe a que los datos obtenidos tras un análisis de audio, corresponden a modelos de procesamiento diferentes a los que se obtiene en el análisis de vídeo. El hecho de trabajar con información diversa, hace que el proceso de integración para llevar un análisis conjunto requiera de un modelado de la información.

Si además contásemos con información procedente de otros sensores (de movimiento, de detección de humos...) que aportasen nuevo conocimiento sobre el entorno, el sistema sería más potente, ya que aumentaríamos los datos de la etapa de análisis. Sin embargo, como se ha dicho anteriormente, la integración de información heterogénea resulta un proceso tedioso y hace que no existan muchas herramientas que amplíen sus fuentes de información al campo de los sensores.

De esta forma, se constituyen otras dos nuevas líneas de investi-

gación abiertas, que están estrechamente ligadas:

- El desarrollo de *sistemas de vigilancia que se alimenten de información diversa* procedente de distintos tipos de sensores instalados en un espacio monitorizado.
- La *integración de información heterogénea*.

Otro aspecto a tener en cuenta en los sistemas de vigilancia es el *tratamiento de la incertidumbre*. En muchos casos, dada la naturaleza del problema, los sistemas que analizan vídeo o audio obtienen del entorno información imprecisa. No es realista pensar en obtener una detección exacta y rotunda de un tipo de evento, ya que los sistemas siempre están condicionados por 'ruidos' que impiden ver u oír con claridad y por las mismas técnicas desarrolladas que aún no han conseguido esa exactitud.

Este hecho podría compararse en la vida real como si nos encontrásemos ante un vigilante de seguridad que no ve bien y que es un poco sordo. Para entenderlo mejor, pongamos un ejemplo. Supongamos que se originase un sonido en el entorno, el vigilante podría dudar entre si es un ruido de cristales o un portazo, inclinándose más por un ruido de cristales. En un sistema inteligente en el que se detecten y clasifiquen sonidos, puede ocurrir lo mismo. En este caso, los resultados obtenidos irían acompañados de un grado de creencia, que indique la posibilidad de que el sonido sea del tipo identificado, por ejemplo, (rotura de cristales, 0.7) y (portazo, 0.3).

Lo mismo puede ocurrir para la clasificación de eventos de vídeo. Por ejemplo, cabe la posibilidad de no conocer con claridad si un objeto es un peatón o un vehículo. Este tipo vaguedad en la información hay que tenerlo en cuenta a la hora de analizar los comportamientos de los objetos. Un tratamiento adecuado de la incertidumbre en un entorno vigilado es crucial para el desarrollo de sistemas de vigilancia con el fin de obtener resultados robustos ante información difusa.

Nuestro estudio seguirá estas líneas de investigación. Partimos del propósito de diseñar un marco teórico que permita la creación de un **sistema multi-sensor capaz de detectar situaciones de interés**

a partir de la integración de diversas fuentes de extracción de conocimiento sobre vídeo, audio y otros sensores.

Como hemos visto, existen ya muchos algoritmos y técnicas que llevan a cabo una detección, clasificación y seguimiento de objetos a partir de vídeo, y sistemas que son capaces de detectar y clasificar los sonidos del entorno. En nuestra propuesta pretendemos reutilizar el conocimiento experto existente sobre las primeras etapas de vigilancia y centrarnos en la etapa de análisis del comportamiento de los objetos con el fin de detectar riesgos importantes.

De este modo, el tipo de Sistema propuesto se alimentará de un pre-análisis cognitivo de señales de audio y vídeo e información de otros sensores. Por lo tanto, no pretendemos hacer una detección y un seguimiento básico de objetos a partir del vídeo, ni una detección de eventos de sonido... sino que partiremos de la disposición de sistemas de extracción de conocimiento que lo realizan.

Esta tesis persigue avanzar en aspectos de integración de información heterogénea y en el complejo proceso de interpretación de escenas y detección de alertas. Trabajaremos con la información a alto nivel para generar nuevo conocimiento, que será analizado para la identificación de situaciones más abstractas y más ricas.

Nuestro propósito es dotar de inteligencia al sistema de seguridad, integrando toda la información que se pueda recoger y realizando un análisis conjunto de los distintos eventos que ocurren en un espacio protegido. Hoy día es muy difícil encontrar sistemas de seguridad de este tipo, que usen información heterogénea proveniente de diversas fuentes y que puedan ser aplicables sobre un cualquier área.

Las premisas que tomaremos como base para el diseño de sistemas de vigilancia inteligentes son **la escalabilidad y la flexibilidad** del sistema. Pretendemos crear una arquitectura flexible que permita escalar fácilmente el sistema, tanto para el estudio de nuevas situaciones como para la inclusión de nuevas fuentes de información. También se tendrá en cuenta la necesidad de desarrollar **herramientas portables** a cualquier ámbito de aplicación, es decir, que sean fácilmente adaptables si el escenario de estudio cambia.

Con el objetivo de integrar la diversa información proponemos el uso de **ontologías**, lo que permite la unificación de la información heterogénea obtenida. Para realizar el análisis de situaciones de alerta nos basaremos en el desarrollo de **controladores difusos**, que permitan realizar un tratamiento adecuado de la información imprecisa y obtener resultados robustos para la detección de situaciones de riesgo (en las que la vaguedad es un factor a tener en cuenta, cuando trabajamos con datos borrosos obtenidos de las fuentes de información).

En los siguientes capítulos conoceremos detalladamente la solución que proponemos para cumplir los requisitos propuestos. En el capítulo 3 describimos nuestra propuesta y definimos un Modelo formal para el desarrollo de sistemas de detección de alertas que usan información de diferentes fuentes heterogéneas y que se caracterizan por ser escalables y flexibles.

Posteriormente, en el capítulo 4, describiremos las aplicaciones desarrolladas para dicho Modelo. El rasgo peculiar de estas aplicaciones es la capacidad para analizar situaciones de riesgo complejas y de interés social en la actualidad: el peligro de atropello a peatones, el peligro ocasionado por la presencia de niños en zonas de tráfico y el riesgo de intrusiones a espacios restringidos.

Capítulo 3

Modelo para el desarrollo de Sistemas de Detección de Alertas

El objetivo principal de esta tesis, como ya hemos comentado, es el desarrollo de una tecnología que permita la creación de sistemas de vigilancia inteligentes multi-sensor, que sean escalables, flexibles y portables a cualquier entorno. Para cubrir este fin, en este capítulo, proponemos un **Modelo** general y abstracto como ***base*** *para el futuro desarrollo de sistemas de vigilancia inteligentes*.

Este *Modelo* constituye el pilar o punto de partida de la construcción de sistemas de detección de situaciones de riesgo en entornos vigilados. La instanciación de dicho *Modelo* para un propósito específico dará lugar a un sistema de detección de alertas concreto. De este modo, podemos ver este *Modelo* como el 'esqueleto' del sistema que se pretenda desarrollar. El *Modelo* será descrito con precisión y detalle en este capítulo, pero sin dejar de ser un marco abierto y adaptable según las funcionalidades de cada entorno de aplicación y de cada situación de alerta estudiada.

En primer lugar, en la sección 3.1 analizaremos con detalle las características que contemplan el tipo de sistemas que pretendemos abar-

car: *los Sistemas de Detección de Alertas*. Expondremos los requisitos y las dificultades a las que nos enfrentamos. De esta forma, ofrecemos una visión general del tipo de sistema que queremos construir.

Una vez expuesta nuestra finalidad, definiremos y describiremos, en las siguientes secciones, el *Modelo* que proponemos para dar solución a la propuesta planteada y cubrir todos los requerimientos.

En la sección 3.2, se describe formalmente las características del tipo de escenario de estudio.

Posteriormente, en las secciones 3.3 y 3.4 se define la arquitectura que constituye el pilar y la base del *Modelo* propuesto. Este *Modelo* ha sido pensado y diseñado para conseguir escalabilidad y flexibilidad en los sistemas. Por ello, proponemos una arquitectura basada en el paradigma del desarrollo de software basado en capas y componentes, que permite la inclusión de nuevos componentes y la desactivación de módulos existentes, en función de las necesidades de cada sistema.

En la sección 3.5, se definen las Ontologías que dan soporte al *Modelo* teórico presentado. Y finalmente, en la sección 3.6, se muestra la aplicabilidad del mismo.

Adelantamos, que en el siguiente capítulo, presentaremos tres *aplicaciones* de este *Modelo*, el cual ha sido instanciado tres veces para la construcción de tres sistemas de vigilancia diferentes.

3.1. Características de un Sistema de Detección de Alertas

Sistema de Detección de Alertas es el denominador que usaremos en esta tesis para referirnos al tipo de sistemas de vigilancia inteligentes que se abarcan en este estudio. Como veremos a continuación, son sistemas caracterizados por una serie de propiedades y requisitos.

En la introducción de este capítulo hemos hablado de la definición de un Modelo para el desarrollo de *Sistemas de Detección de Alertas*. Sin embargo, para pasar a definir ese Modelo, es necesario conocer qué

entendemos en este trabajo por Alerta y por Sistemas de Detección de Alertas. Por ello, en esta sección precisaremos estos conceptos y analizaremos los requisitos y las dificultades a las que nos enfrentamos.

A continuación, veremos qué es una alerta, qué es un sistema detección de alertas, cuál es su estructura general y qué características contempla.

3.1.1. Definición

El termino "Alerta" es definido por la Real Academia Española como *una situación de vigilancia o atención*. En esta tesis, este concepto tiene un ligero matiz, como veremos a continuación.

Definición. Una ***alerta***, en nuestro Modelo, es una situación de vigilancia o atención que es estudiada por un sistema[1] inteligente computacional con el fin de llamar la atención del usuario cuando detecte circunstancias que impliquen la presencia de dicha situación.

En este estudio, trataremos el concepto de alerta de forma difusa. De este modo, una alerta estará caracterizada con un grado de creencia (valor entre 0 y 1) que indica la posibilidad de la presencia de dicha situación de vigilancia en cada momento.

Otros términos que usaremos a lo largo de este documento para referirnos a una *alerta* son: situación de estudio, situación de riesgo, situación anormal, situación de vigilancia, situación de alerta...

Algunos ***ejemplos*** de posibles alertas son: identificación de objetos abandonados, detección de peligro de accidente de tráfico, identificación de intrusiones en zonas restringidas...

Una vez establecido el concepto de alerta, daremos una definición general sobre un Sistema de Detección de Alertas, que como su propio nombre indica, será un sistema que avise de la presencia de alertas en un entorno vigilado.

[1] El "Sistema de Detección de Alertas"

Definición. Un ***Sistema de Detección de Alertas, o SDA,*** es un sistema (P) cuya finalidad es la identificación de una o de varias situaciones de riesgo (S) a partir de la diversa información obtenida (E) en un entorno monitorizado. La detección de presencia de una anomalía que implique un riesgo definido con anterioridad, conllevará la activación de una alerta que indique el nivel de posibilidad de peligro detectado y las circunstancias que han conllevado a dicha activación. Formalmente, la estructura de un SDA se representa como una 3-tupla **SDA=<E,P,S>**, donde:

- **E** es el conjunto de *entradas* del Sistema. Constituye toda la información que puede recogerse de un entorno monitorizado por diversos sensores (éstos pueden ser cámaras de vigilancia y/o micrófonos y/u otro tipo de sensores).
- **P** es un módulo de *procesamiento* que constituye el nodo central del sistema. Se encarga de recoger e integrar todos los datos de entrada E, para procesarlos y analizarlos con el fin de identificar la presencia de un conjunto predefinido de alertas S en el entorno.
- **S** es el conjunto de *salidas* del Sistema, que muestra el estado de las Alertas estudiadas. El conjunto S puede estar constituido por una o varias situaciones de vigilancia. El sistema ofrece como salida los distintos niveles de alerta detectados para cada situación estudiada, junto con la explicación de los acontecimientos que han originado ese estado de los niveles de alerta.

Una alerta puede estar inactiva o activa. Está *inactiva* cuando el sistema no detecta la presencia de dicha situación de riesgo y *activa* cuando el sistema detecta circunstancias definidas con anterioridad que impliquen la presencia de dicha alerta. La identificación de una alerta, conllevará la activación de una *Alarma*. El concepto de *alarma* puede definirse, según el diccionario de la Real Academia Española, como un *aviso o señal de cualquier tipo que advierte de la proximidad de un peligro*.

A continuación, vamos a detallar la *estructura* de un SDA y veremos las características o *requisitos* ligados a dicha estructura que

tendremos en cuenta en el diseño del *Modelo*. Una vez conocidas las características, podremos ver ejemplos de SDAs en la sección 3.2.

3.1.2. Estructura general

Como hemos señalado anteriormente, el tipo de sistema que se pretende llegar a construir, *un Sistema de Detección de Alertas*, se caracteriza por poder recibir información heterogénea proveniente de distintas fuentes de conocimiento: vídeo, audio u otro tipo de sensores. El propósito del sistema es realizar una integración de la información de entrada para poder llevar a cabo un análisis y un procesamiento conjunto con el objetivo de detectar la presencia de determinadas situaciones de riesgo. Cuando una situación de estudio es identificada, el sistema debe notificar el estado de la misma al operador. De esta forma, se logrará que la herramienta le sirva de soporte en su trabajo.

En la figura 3.1 podemos ver la estructura general de los Sistemas de Seguridad que se abarcarán en este estudio.

Entradas

El Sistema propuesto recibirá como entrada el *flujo de información procedente del nivel sensorial*, por lo que debe ser capaz de recibir constantemente una serie de eventos que den información acerca de lo que ocurre en un entorno vigilado provisto de micrófonos, cámaras de vigilancia y otro tipo de sensores (de movimiento, incendio . . .).

Se pretende que el sistema pueda ser aplicado en un entorno *multifuente* y *multisensor*, es decir, en un espacio protegido por diversos tipos de fuentes de información (audio, vídeo, y otros sensores) donde pueden existir varios sensores para cada fuente (varias cámaras, varios micrófonos...). De esta forma, se requiere que el sistema esté diseñado para alimentarse de información heterogénea procedente de diferentes tipos de fuentes de información. Y a su vez, que sea flexible ante la existencia de varios sensores, del mismo tipo, para monitorizar un entorno.

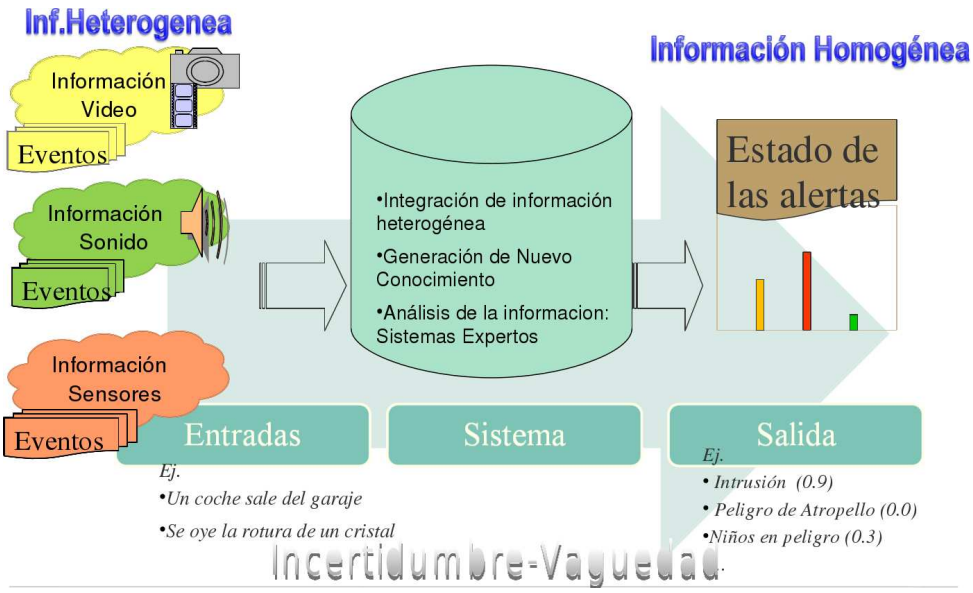


Figura 3.1: Estructura general de los Sistemas que se contemplan en este estudio. Entradas y salidas.

La información de entrada estará constituida por eventos obtenidos tras un análisis cognitivo básico de vídeo, audio y sensores. Como se expresó en los objetivos de este trabajo, nuestro estudio se centrará en el análisis de los comportamientos de los objetos que aparecen en un escenario monitorizado para estudiar distintas situaciones de riesgo. Por ello, para avanzar en este aspecto, no partimos del vídeo o del sonido en bruto, sino que partiremos de datos de entrada ya analizados. Como hemos visto en el capítulo 2, actualmente existen muchas herramientas que extraen conocimiento a partir de dichas fuentes. Lo que pretendemos es fomentar el uso de sistemas que ya realicen estas funciones y seguir avanzando en la presente línea de investigación.

El conocimiento de entrada debe de especificar los datos básicos para realizar un análisis posterior que permita llevar a cabo la interpretación de los acontecimientos que tienen lugar en una escena. De este modo, sería necesario y esencial que el sistema conozca: *cuántos objetos existen en el escenario de estudio en el momento actual, cuál es su*

tamaño o sus dimensiones dentro de la imagen, cuál es su posición en la imagen, qué tipo de objeto puede ser...

Todos estos datos pueden conocerse empleando herramientas que realicen algoritmos de detección y tracking 2D de objetos a partir del análisis de vídeo cognitivo.

Otra información útil para el sistema, que complementaría la información anterior, es conocer *qué trayectorias o circulaciones tiene asociadas un objeto en el momento actual*. Este conocimiento podría ser obtenido también tras un análisis de vídeo.

Como nos basamos en la idea de que el espacio monitorizado puede ser multifuente y, por lo tanto, puede estar provisto de micrófonos y otros sensores, también sería de gran utilidad que, en este caso, el sistema conozca: *qué tipo de sonidos son identificados en el ambiente, cuál es la localización donde ha tenido lugar dicho sonido, qué sensores han sido activados o desactivados, etc.*

Es importante destacar que la información de entrada puede contener datos con imprecisión y vaguedad dependiendo de las características de cada fuente de información.

Por consiguiente, desde un punto de vista general, como posibles *entradas* al Sistema destacamos:

- *Información sobre vídeo*:
  - Eventos que indican la posición, el tamaño y la clasificación de los objetos en la imagen. Éstos serían obtenidos tras un proceso de análisis cognitivo de imágenes para la detección de objetos. Por ejemplo, un evento de este tipo sería la "detección de un objeto con identificador 'id1', con un tamaño (15,10), situado en las coordenadas (300,214), clasificado como vehículo con una posibilidad de que lo sea de 0.8".
  - Eventos que indican la trayectoria que sigue un objeto. Estos eventos serían obtenidos tras un proceso de análisis del vídeo a un nivel más alto. Por ejemplo: "objeto 'id1' saliendo del garaje con una posibilidad de 0.6".

Los eventos de vídeo que indican la posición, el tamaño y la clasificación de los objetos suponen una información esencial para ser la base de un razonamiento posterior. En cambio, la identificación de trayectorias a alto nivel, puede ser una información adicional que, si el sistema la recibe, puede ser usada en su etapa de análisis si la situación estudiada lo requiere.

- *Información sobre audio*:
  - Eventos que indican la detección de determinados sonidos en el entorno. Por ejemplo: "rotura de un cristal con una posibilidad de 0.5", "disparo con una posibilidad de 0.9" . . .
- *Información sobre sensores*:
  - Eventos que indican la activación y desactivación de un determinado sensor. Por ejemplo: la activación de un sensor de movimiento, de un detector de humos. . .

Procesamiento

Toda la información de entrada será integrada en el sistema para poder ser procesada y analizada, generando nuevo conocimiento de alto nivel, con el fin de detectar circunstancias de riesgo tanto para las personas como para las infraestructuras.

De esta forma, el sistema se enfrenta a tres tareas principales:

- en primer lugar, a un proceso de *integración de información heterogénea*,
- en segundo lugar, a un proceso de *generación de nuevo conocimiento*,
- y en tercer lugar a un *razonamiento inteligente* que le permita detectar riesgos de forma automática. Un único sistema puede analizar una o más alertas.

Uno de los factores a tener en cuenta es la presencia de *vaguedad o incertidumbre* en la información de entrada. Esto exige que el sistema

realice un tratamiento adecuado de la información imprecisa para obtener resultados robustos en la detección de situaciones de riesgo.

Salidas

El sistema realizará un control constante sobre las distintas situaciones que sean objeto de estudio y ofrecerá como salida el estado de dichas alertas. De esta forma, uno de los requisitos será que el usuario pueda consultar el estado de las alertas siempre que lo desee.

Como se ha dicho anteriormente, una alerta es un concepto difuso caracterizado con un grado de creencia (valor entre 0 y 1) que indica el nivel de presencia detectado en cada momento sobre la situación de vigilancia. Los eventos de entrada podrán o no, incrementar o decrementar el nivel de las alertas.

Cada alerta tendrá asociado un umbral (valor entre 0 y 1). Cuando el grado de creencia de una alerta supere dicho umbral, el sistema lanzará un aviso o alarma que indique que cierta alerta está activada. De este modo, el sistema llamará la atención del operador en los casos que sean necesarios.

De forma general, algunos ejemplos de posibles situaciones susceptibles de vigilancia o alertas serían:

- *Peligro de atropello*. Si se dan las circunstancias para que una persona pueda sufrir un posible atropello por un vehículo.
- *Identificación de niños en peligro*. Si son detectados niños lejos de personas adultas en una zona peligrosa por el tránsito de vehículos.
- *Intrusión*. Si algún objeto entra en un espacio de seguridad cuando no le es permitido.
- *Peligro de choque de vehículos*. Si se dan las circunstancias para que dos vehículos puedan colisionar.
- *Merodeo en el exterior de un edificio*. Si una persona, grupo de personas o vehículo(s) deambulan cerca del edificio vigilado.

3.1.3. Dificultades y requisitos

La principal dificultad radica en la **diversidad y vaguedad** de las entradas que el sistema puede recibir. Este hecho se debe a varios factores:

- La *información de entrada es heterogénea*, proviene de diferentes fuentes (audio, vídeo, sensores), por lo que se reciben eventos muy variados entre sí. Además, pueden ser originados por distintos sistemas de extracción de conocimiento, los cuales usan una ontología propia, "ad hoc" para sus necesidades.
- El *nivel de vaguedad e imprecisión de los datos es elevado*. Mucha de la información extraída de las fuentes de audio y vídeo puede venir determinada por un grado de creencia. Un ejemplo sería la clasificación de los objetos: persona 0.8, coche 0.2. Esto nos obliga a construir un sistema que sea capaz de razonar con incertidumbre y que sea robusto ante este tipo de información.
- La *calidad de las salidas* del Sistema de Detección de Alertas *vendrá condicionada por la precisión y calidad de la información de los sistemas de extracción de conocimiento de audio, vídeo y sensores que se empleen como entrada*. Por este motivo, se procesará la información de entrada con el fin de enriquecerla y obtener eventos más complejos o información a más alto nivel que nos proporcione más conocimiento acerca de la escena.

A continuación describiremos los **requisitos** fundamentales que queremos que recoja un *Sistema de Detección de Alertas* y que tendremos en cuenta en la etapa de diseño de nuestro Modelo:

- El sistema de detección de alertas debe de ser capaz de recibir información procedente de diversos tipos de fuentes de información con contextos diferentes; por ejemplo, fuentes de audio, vídeo u otros sensores. Consecuentemente, el sistema de vigilancia debe poder **integrar toda la información heterogénea de entrada en información homogénea**, con el objetivo de realizar un

procesamiento conjunto y contar con una mayor información en la etapa de toma de decisiones.

La tecnología que proponemos, tiene como entrada información procedente del nivel sensorial, y como salida, la identificación y análisis de situaciones de interés. Hemos de enfatizar, que nuestro estudio pretende avanzar en aspectos de integración de información heterogénea y detección de situaciones de alerta, centrándonos más en la etapa de análisis de los acontecimientos de una escena.

Por ello, supondremos que las salidas de la capa sensorial son resultados de un análisis cognitivo preliminar de señales de audio, vídeo y otros sensores. Por lo tanto, como se ha dicho anteriormente, no pretendemos hacer una detección, clasificación y seguimiento básico de objetos a partir del vídeo, ni una identificación de eventos de sonido..., sino que partiremos de sistemas de extracción de conocimiento que realizan dichos análisis.

- **Se generará nuevo conocimiento a alto nivel** a partir de la información de entrada, con el objetivo de realizar un estudio más exhaustivo de los comportamientos de los objetos que aparezcan en el escenario de estudio.

- **El sistema debe ser robusto ante información imprecisa o borrosa**. Para ello, se llevará a cabo el uso de métodos formales que permitan tratar la incertidumbre y la vaguedad. La lógica difusa representará un papel fundamental en la etapa de análisis de la información.

- Debe **tener la posibilidad de estudiar diferentes situaciones de interés o riesgo de forma independiente**.

- Otro requisito del sistema es que sea **fácilmente escalable y flexible**, en tres aspectos:

  - Se desea que el sistema sea *escalable para poder insertar nuevos módulos futuros (de entrada) que obtengan otras características sobre las fuentes* de audio, vídeo y sensores. Por ello, el Sistema de Detección de Alertas propuesto se construirá de forma incremental, para que nueva información

de entrada pueda integrarse correctamente en el conocimiento del que se dispusiera con anterioridad.

- También se requiere *escalabilidad en cuanto a la inclusión de nuevos análisis sobre nuevas situaciones de vigilancia*. Por lo que el sistema debe realizarse también de forma incremental en este aspecto.
- Se requiere que el sistema pueda *ser aplicado a cualquier entorno físico* monitorizado, con el fin de crear aplicaciones portables a distintos escenarios.

- El sistema **publicará constantemente el estado de las alertas estudiadas** y **generará alarmas en tiempo real** cuando sea necesario.

  La identificación de la presencia de determinadas circunstancias será publicada de dos formas: por un lado, se debe llamar la atención de forma visual o sonora al operador; por otro lado, el sistema publicará esta información a través de un middleware para que pueda ser utilizada en un futuro por otros sistemas que requieran dicha información.

  En el primer caso, queremos conseguir que la herramienta propuesta sirva de ayuda al vigilante y lo avise sólo en los casos que sea necesario. En el segundo caso, perseguimos la idea de potenciar el uso de sistemas ya construidos con el fin de reutilizar para otros fines el conocimiento experto generado, por ejemplo, en el caso de desplegar un gestor de crisis que intente solventar la situación detectada.

- **Sensibilidad al tiempo real**: capacidad de actualizar el sistema al ritmo de los rápidos flujos de entrada (p.ej. el streaming de vídeo). Así como ofrecer una salida del sistema en tiempo real y adaptada a la las necesidades que solicite cada posible subscriptor del sistema.

- Se pretende conseguir una **correcta representación del conocimiento analizado** para así obtener una buena interpretabilidad de los resultados. El sistema debe poder informar en

todo momento del estado de las situaciones de estudio. En el caso de detectar alteraciones en las mismas, debe ofrecer una explicación sobre los distintos acontecimientos analizados en la escena, para así proporcionar una buena comprensión de los resultados obtenidos, tras analizar las distintas situaciones de riesgo.

- Para otorgar **movilidad** al sistema, incluiremos el aviso de las alertas, en tiempo real, mediante dispositivos móviles. De esta forma, cualquier operador podrá ser informado sin necesidad de estar permanentemente frente al sistema.

3.2. Escenario de estudio

La *vigilancia de un entorno* consiste en la continua observación realizada sobre el mismo con el fin de detectar anomalías. Si esta vigilancia se lleva a cabo usando dispositivos electrónicos (como cámaras de vigilancia, micrófonos, sensores...), el proceso de vigilancia puede llamarse proceso de monitorización.

En la mayoría de las ocasiones, el área que se quiere vigilar es extensa. Sin embargo, es cierto que, en muchos casos, de la amplia área sólo interesa monitorizar determinadas zonas o puntos conflictivos, ya sea en zonas exteriores o en interiores de edificios. Además, dentro de un amplio entorno vigilado, en cada uno de sus puntos monitorizados puede ser necesario que se analicen distintas situaciones de interés o riesgo.

Pongamos un ***ejemplo***: imaginemos que se pretende vigilar una amplia sede que consta de tres edificios (A,B y C), aparcamiento y zonas ajardinadas. El área que se quiere vigilar es muy amplia. Sin embargo, supongamos que el director de la sede solo está interesado en tres aspectos:

- La detección de intrusiones en el edificio principal (edificio A) para evitar posibles asaltos o robos.

- La detección del peligro de atropello a la salida del aparcamiento, ya que en los últimos meses se han ocasionado varios accidentes

debido a que hay niños del colegio colindante que acceden a esa parte en busca de balones perdidos y son víctimas de la salida de vehículos.

- La detección de objetos abandonados a la salida del aparcamiento.

Para solventar los problemas sugeridos, no es necesario vigilar todo el espacio global de la sede, ya que existen zonas que no aportarían información en la detección de las anomalías anteriores. Una solución muy apropiada pasaría primero por la identificación de los escenarios de estudio, por ejemplo, podríamos diferenciar 2 escenarios: 1)Área del edificio A, y 2)Entrada y salida del garaje.

Cada uno de los escenarios se equiparía con las cámaras, micrófonos u otros sensores que fuesen necesarios para la monitorización de los mismos y el análisis de las distintas situaciones de estudio.

En el escenario 1, la situación de riesgo que se desea vigilar es la intrusión, mientras que en el escenario 2, se estudiarían dos alertas: el peligro de atropello y la identificación de objetos abandonados. En este último escenario, ambas situaciones de estudio se alimentan de la misma información de entrada (porque ambas estudian los acontecimientos ocurridos en el mismo entorno), a pesar de que cada una realice un análisis diferente.

Lo más apropiado en este caso es tener dos Sistemas de Detección de Alertas independientes, uno para cada escenario. Cada SDA tendría las entradas pertinentes y los módulos expertos necesarios para el análisis y la detección de las anomalías específicas. De esta forma, conseguimos que un mismo sistema no analice información que no es relevante para su estudio; por ejemplo, para detectar una anomalía en el escenario 1 no es necesario analizar los acontecimientos que tienen lugar en el escenario 2. De aquí nace la idea de que *los Sistemas de Detección de Alertas se centren en analizar los acontecimientos que ocurren en un único* ***escenario local***.

Definición. Un ***escenario local de vigilancia*** es un lugar concreto y específico donde ocurren y se desarrollan varios eventos que son monito-

rizados bajo el uso de una o varias cámaras de vigilancia y/o uno o varios micrófonos y/o uno o varios sensores, con el objetivo de analizar la presencia de una o varias situaciones que requieren atención. Formalmente, un escenario local es una 7-tupla **E_L=<L,C,M,S,E,A,SDA>**, donde:

- **L** es el *lugar vigilado*. Este entorno monitorizado no se caracteriza por ser un área extensa, sino que suele ser un entorno local donde se analizará la presencia de riesgos concretos.
- **C** es el conjunto de *cámaras* de vigilancia existentes en L. Puede haber una o más cámaras. La presencia de múltiples cámaras indica la existencia de diferentes vistas sobre el mismo lugar. Son necesarias varias cámaras cuando una sola no puede abarcar toda la vista de la escena.
- **M** es el conjunto de *micrófonos* existentes en L.
- **S** es el conjunto de otros *sensores* existentes en L.
- **E** es el flujo de *eventos* recogidos desde C, M y S. Son los eventos preprocesados que constituirán las entradas del Sistema de Detección de Alertas.
- **A** es el conjunto de *alertas* que necesitan vigilancia en el entorno L.
- **SDA** es el Sistema de Detección de Alertas usado para vigilar el espacio físico L. El conjunto E constituyen las entradas del sistema. Y A es el conjunto de alertas o situaciones de estudio que analizará el sistema a partir de E con el fin de llevar a cabo un control sobre la presencia y el estado de las mismas en L.

Como se ha dicho anteriormente, cuando el entorno vigilado es amplio, la división del mismo en diferentes escenarios de estudio es muy importante. En la mayoría de los casos, para reconocer una situación de interés o anomalía en una zona concreta (como puede ser el peligro de atropello en la entrada y salida de coches de un garaje), no importa lo que esté pasando en otra zona (entrada del edifico A), por lo que solo es necesario analizar los eventos de la zona concreta para el estudio.

No siempre se parte de un entorno global en el que es necesario la identificación de escenarios locales. En muchas otras situaciones el entorno global de vigilancia será el escenario local de estudio. Este es el caso de entornos que se caracterizan por ser muy concretos y susceptibles de vigilancia para la detección de un fin concreto que ocurre dentro del espacio vigilado. Un posible *ejemplo* sería la monitorización de un cruce de peatones, en el que se espera vigilar la presencia de peligro de atropello.

La vigilancia de una misma situación de riesgo en distintos entornos es altamente demandado. Nos referimos al hecho de que en dos escenarios independientes se analice la misma situación de riesgo. Por ejemplo, supongamos que también se desea estudiar el peligro de atropello, pero esta vez en una travesía. El Sistema de Detección de Alertas podría ser el mismo que el aplicado en el cruce de peatones del ejemplo anterior, instanciado para este nuevo entorno. Éste es el ideal que perseguimos, crear herramientas portables, escalables y flexibles para su aplicación en entornos diferentes y poder reutilizar el conocimiento experto.

Como se observa en los ejemplos, un escenario local se caracteriza porque existe una fuerte relación contextual y espacial entre las alertas y el lugar o entorno de estudio.

Concluyendo, el **Modelo que proponemos** en este capítulo para el desarrollo de Sistemas de Detección de Alertas, **está basado en la idea de vigilar *escenarios locales***. No queremos sobrecargar a los sistemas con procesamiento innecesario, sino que nos basamos en la idea de que cada sistema debe analizar los acontecimientos esenciales, que son aquellos que ocurren únicamente dentro del entorno de estudio. Lo cual no quiere decir que se desarrollen herramientas para entornos específicos, sino aplicaciones fácilmente adaptables para vigilar cualquier tipo de escenario local.

3.3. Diseño de Sistemas de Detección de Alertas basado en capas

Un Sistema de Vigilancia Inteligente que cubra las características expuestas en la sección 3.1 es un sistema complejo y de gran magnitud. Una solución muy acertada para abarcar el diseño de este tipo de sistemas sería la división jerárquica de diferentes capas [22, 85].

El objetivo general es realizar una separación entre la capa de datos o fuentes de información, la lógica de procesamiento y la capa de presentación al usuario. En la figura 3.2 proponemos un esquema general para el diseño de este tipo de sistemas donde diferenciamos 3 capas principales:

1. **Capa de fuentes de información**. El objetivo de esta capa es recoger toda la información posible del entorno vigilado para así poder dotar al sistema con un conocimiento básico, inicial y esencial sobre la escena, el cual se empleará en un análisis posterior más avanzado.

   Se trata de la capa inferior y se encarga de recoger información directamente del nivel sensorial, es decir, de los sensores que existan en el entorno monitorizado: micrófonos, cámaras y otros sensores. Esta información se envía a la siguiente capa o capa intermedia: *la capa de procesamiento*. Antes de ser enviada, existirá un procesamiento previo de las señales de vídeo, audio y sensores. Este pre-procesamiento sería realizado por otros sistemas de extracción de conocimiento que usen como entrada las señales en bruto de dichas fuentes de información. Por ejemplo, nos referimos a sistemas que lleven a cabo algoritmos de detección y tracking de objetos a partir del vídeo, o algoritmos de procesamiento de señales acústicas para la clasificación de los sonidos del entorno, etc.

   Podría darse el caso de que no sea necesario realizar un procesamiento previo. De esta forma, la información de los sensores pasaría directamente a la siguiente capa. Sin embargo, en este caso, se sobrecargaría el procesamiento de la capa intermedia.

2. **Capa de procesamiento**. Se encarga de recoger toda la información obtenida en la *capa de fuentes de información* para realizar un análisis y un razonamiento más complejo sobre los acontecimientos que tienen lugar en el escenario vigilado. Dentro de esta capa podemos distinguir tres sub-capas:

   a) **Capa de fusión e integración de información (pre-procesamiento)**. En esta sub-capa toda la información heterogénea de entrada debe ser integrada de forma homogénea. Para ello, nos basaremos en el uso de *ontologías y técnicas de fusión de datos*. La dificultad radica en identificar cuál es la información que referencia a un mismo objeto de la escena y no a otro, a partir de los datos obtenidos de las diversas fuentes.

   En este nivel, puede darse el caso de que no sólo se realice una fusión e integración de la información, sino que al mismo tiempo los datos pueden ser procesados obteniendo nuevo conocimiento. Por ejemplo, si se recibiese la posición de los objetos detectados en la escena en coordenadas 2D de la imagen, estas posiciones podrían ser procesadas para obtener su coordenadas 3D en el mundo real, usando una técnica adecuada.

   Una vez que la información ha sido homogeneizada, pasa a la siguiente sub-capa: la capa de *análisis*.

   b) **Capa de análisis**. En esta etapa tiene lugar todo el razonamiento inteligente que se realiza sobre el conocimiento adquirido.

   La información recibida desde la *capa de fusión e integración de información (pre-procesamiento)* es almacenada en una *Base de Conocimiento*, a partir de la cual será consultada para cualquier tipo de procesamiento.

   Dentro de esta *capa de análisis* distinguimos dos niveles paralelos:

   - **Nivel de inteligencia**. Estará constituido por Sistemas Expertos que analizarán las distintas situaciones de estudio basándose en la información almacenada en la *Base de Conocimiento*.

Existirán tantos Sistemas Expertos como situaciones de vigilancia o de alerta se deseen estudiar. El estado de cada una de las alertas analizadas será transmitido a la siguiente capa: la *capa de transmisión de alertas*.

- **Nivel de extracción de conocimiento**. Al igual que el *nivel de inteligencia*, este nivel se alimenta de la información almacenada en la *Base de Conocimiento*.

  Estará constituido por distintos módulos que permitan la adquisición de nuevo conocimiento necesario para analizar las distintas situaciones de alerta.

  Por ejemplo, si se desea clasificar los objetos como niños o adultos y solo disponemos de la clasificación de los objetos como personas o vehículos, se crearía un nuevo módulo para lograr alcanzar ese conocimiento basándose en las características de los objetos. La nueva información adquirida en cada componente creado en este nivel, actualizará y complementará la *Base de Conocimiento*, haciéndola contextualmente más rica.

  En este nivel no sólo deben existir módulos que amplíen la información ya existente, sino que también es necesaria la creación de procesos que mantengan la coherencia en la *Base de Conocimiento* y, en algunos ocasiones, se deberá eliminar información no coherente. Por ejemplo, este sería el caso de identificar cuándo un objeto ha dejado de aparecer en la escena, para así dejar de tener en cuenta la información referente a éste, a la hora de realizar el análisis de las distintas situaciones de riesgo.

c) **Capa de transmisión de alertas**. Esta capa será la encargada de transmitir la información generada acerca del estado de las distintas situaciones de alerta analizadas en el *nivel de inteligencia*.

La existencia de esta capa se debe a que pretendemos que esta publicación de información sea dependiente de contexto. Este punto se explicará con más detalle en la sección 3.4.

Esta información será transmitida a la *capa de presentación de resultados*. Aunque en este estudio no se contempla,

esta información podría pasar a otra capa paralela a la de *presentación de resultados*, dedicada a la gestión de crisis. De esta forma, la detección de alertas sería recogida por a aquellos otros sistemas que requieran la información obtenida en la *capa de procesamiento* para seguir avanzando en otros procesos de seguridad, por ejemplo, el desarrollo de herramientas de ayuda a la decisión, con protocolos de actuación ante la identificación de situaciones críticas.

3. **Capa de presentación de resultados**. En esta capa se encuentran las herramientas de monitorización de los resultados obtenidos de la capa anterior, concretamente, existirán: a) una herramienta de escritorio donde se pueda visualizar y controlar el estado de las alertas y configurar los parámetros necesarios para llevar a cabo el análisis automático de las distintas situaciones; b) una herramienta ad hoc dispositivos móviles, que recibirá las notificaciones referentes al estado de las alertas.

Podemos observar la verticalidad de las capas que se caracteriza por tener una única dirección del flujo de información. La capa inferior o de fuentes de información transmite un flujo continuo de eventos a la capa intermedia o de procesamiento. Ésta, a su vez, transfiere un flujo constante de información a la capa de presentación de resultados.

La ventaja principal de diferenciar varias capas es que el desarrollo se puede llevar a cabo en varios niveles y, en caso de que sobrevenga algún cambio, sólo se ataca al nivel requerido. También, el diseño basado en capas le otorga escalabilidad y flexibilidad al sistema.

En la mayoría de los sistemas de vigilancia encontrados en la literatura, no se realiza una división clara de capas que separe el análisis de los comportamientos del resto de tareas, lo que conlleva una cohesión fuerte a la hora de procesar los datos y analizar situaciones que hace que sea poco escalable en el futuro.

Una división en capas, similar a la que aquí proponemos, se puede ver en [8], donde estamos de acuerdo con el autor en que una división clara entre la capa de fuentes de información y la capa intermedia

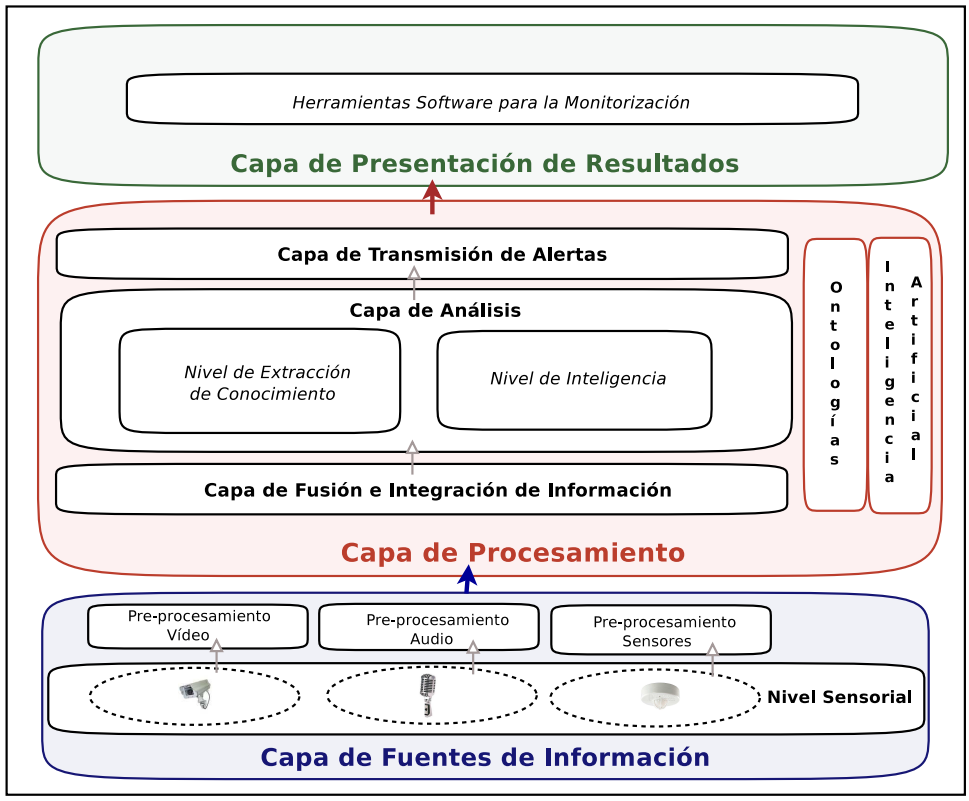


Figura 3.2: Esquema general del diseño propuesto basado en capas

permite que el estudio de las situaciones de alerta sea independiente de las técnicas o algoritmos usados para procesar las señales. De esta forma, se facilita la reutilización de conocimiento experto. El hecho de que se use un algoritmo u otro para el procesamiento de señales es transparente y, por lo tanto, no influye al proceso de análisis de un riesgo.

En este estudio **nos centraremos en la *capa de procesamiento* y en la generación de herramientas que permitan monitorizar los resultados** obtenidos tras el análisis de las distintas situaciones de vigilancia estudiadas.

3.4. Diseño de la Capa de Procesamiento basado en componentes

El Modelo propuesto se centrará en el desarrollo de sistemas cuya capa de *fuentes de información* estaría resuelta. Como se ha dicho anteriormente, en este estudio nos basamos en la reutilización de conocimiento experto y, por lo tanto, en el uso de sistemas inteligentes que realicen un análisis previo de las señales de vídeo o audio. Estos sistemas se incluirían dentro de la primera capa. Los resultados obtenidos en los mismos constituyen las entradas de los Sistemas de Detección de Alertas (sistemas de vigilancia que abordamos en este trabajo).

El diseño de un Sistema de Detección de Alertas, se centrará en la *Capa de Procesamiento*, definida anteriormente. Por ello, vamos a profundizar y a detallar el diseño y la arquitectura del Modelo propuesto dentro de dicha capa.

Como se describió en la sección 3.3, en la capa de procesamiento contamos con tres subcapas: *capa de fusión e integración de información*, *capa de análisis* y *capa de transmisión de alertas*. Como veremos a continuación, la primera se caracteriza por la existencia de uno o varios módulos *Traductores* (T), la segunda capa está constituida por una *Unidad de Procesamiento* (UP) y la tercera consta de un *Notificador de Alertas* (NA) en tiempo real y sensible al contexto.

3.4.1. Fusión e integración de información.

Recordamos que el tipo de sistema de vigilancia que se quiere desarrollar, se alimentará del flujo de datos procedente de la *capa de fuentes de información*. Si un escenario está dotado de cámaras de vigilancia, se reciben eventos de análisis del contenido del vídeo. Si además, está dotado de micrófonos, se recibirán también eventos sobre el sonido percibido. Y si se incluyen otro tipo de sensores, se recibirán eventos sobre la activación/desactivación de los mismos.

Los tipos de eventos que alimentan la *capa de procesamiento* del sistema dependerán de los sistemas de extracción de conocimiento empleados para realizar un pre-procesamiento de las señales recogidas por los sensores. Por ejemplo, un Sistema de Extracción de Conocimiento sobre Vídeo (SECV) podría ser uno que realice el proceso de detección y seguimiento de objetos. Un Sistema de Extracción de Conocimiento sobre Audio (SECA) podría ser uno que identifique ciertos ruidos o sonidos del entorno (por ejemplo: disparos, sirenas, roturas de cristales...).

Para cada uno de los sistemas de extracción de conocimiento empleados como fuente de información, se creará un **Traductor** que convierta la información obtenida de estas fuentes, en información que pueda ser manejada de forma homogénea por el sistema final.

El problema radica en que cada sistema de extracción del conocimiento usa una ontología específica. Si el sistema de vigilancia final *(SDA)* se construyera usando estas ontologías, éste sería poco escalable ya que los cambios en las fuentes de información modificarían todo el sistema. Además, la información de entrada es contextualmente muy diversa porque proviene de diferentes fuentes (audio, vídeo...).

Para paliar estos inconvenientes abstraeremos todas las posibles ontologías que se hayan generado y las agruparemos en una ontología general denominada *Ontología para la Representación Homogénea del Conocimiento (ORHC)* que definiremos a continuación en la sección 3.5. Dicha ontología nos permite realizar una representación del conocimiento, formulando un esquema conceptual riguroso dentro del dominio en el que estamos. Este Modelo pretende integrar los datos sin discriminarlos por su naturaleza (de audio, vídeo...).

Lo que se persigue con los *traductores*, *es transformar la información de entrada en un dominio que nuestro Sistema de Detección de Alertas sea capaz de manejar*, es decir, *representándola de acuerdo con la ontología de dominio que proponemos*.

Cada traductor, está constituido por un servidor que se mantiene constantemente a la escucha de los eventos que reciben como entrada. Esta información es analizada y procesada, pudiendo obtener nuevos datos a un nivel más alto. De esta forma, en los casos que se pueda, dotaremos a los traductores de procedimientos adicionales que permitan *obtener nueva información*.

A continuación pasaremos a definir formalmente un Traductor.

Definición. Un **Traductor** es un componente, perteneciente a un Sistema de Detección de Alertas, encargado de procesar un determinado flujo de información procedente de un pre-procesamiento de un tipo determinado de señales. Su objetivo es transformar la información de entrada en información representada bajo el esquema conceptual definido en la Ontología *ORHC* (definida en la sección 3.5). Su función principal es actualizar la Base de Conocimiento del sistema fusionando e integrando la nueva información con los datos ya existentes. Un traductor puede ser representado por una 3-tupla **T=<I,PA,I'>**, donde:

- **I** es un flujo de información de entrada procedente de un pre-procesamiento previo de un tipo de señal.
- **PA** es el conjunto de *procedimientos adicionales* con los que se dota al traductor para complementar la información I en los casos que se pueda y se requiera. Un traductor puede tener 0, 1 o varios PA.
- **I'** es la información de entrada I (complementada -en los casos que se pueda- con nuevo conocimiento generado por los PA) representada bajo el esquema conceptual descrito en la ontología *ORHC* (ver sección 3.5).

Veamos un ***ejemplo*** de posibles traductores: supongamos que disponemos de un sistema que realiza una detección y un seguimiento

de objetos en 2D a partir de las señales de vídeo y otro sistema que clasifica algunos sonidos a partir del procesamiento de las señales de audio. En este caso se construirán dos traductores, uno para procesar el flujo de información sobre el análisis de audio y otro para el procesamiento del flujo de información sobre análisis de vídeo. Como se ha dicho anteriormente, cada traductor se encargará de recoger los eventos correspondientes a cada análisis y los procesará con el objetivo de transformar los datos de entrada en información homogénea que sigue el esquema definido en la *ORHC*. Además, el módulo del traductor para el flujo de vídeo, por ejemplo, puede estar dotado de un procedimiento adicional que le permita obtener la posición real de un objeto en el escenario a partir de su posición en la imagen, y este dato sería contemplado en la información homogénea final generada por el traductor.

Siguiendo con el ejemplo, es importante resaltar que si existen varios sensores de un mismo tipo, por ejemplo dos micrófonos, la señal recogida desde cada sensor, en este caso de cada micrófono, pasaría por el pre-procesamiento del sistema de extracción de conocimiento para dicha fuente existente en la capa de Fuentes de Información. Los resultados obtenidos tras procesar las señales de ambos sensores pasarían a la Capa de Procesamiento. Para este ejemplo concreto, tendríamos como entrada en esta capa dos flujos de análisis de sonido. Ambos flujos de información usarían el mismo traductor, ya que la información de cada flujo tiene las mismas características.

Una vez que los eventos de entrada son transformados al esquema conceptual definido en la Ontología, esta información es enviada a la *capa de análisis*, concretamente a las Unidades de Procesamiento. El Traductor será el encargado de actualizar la Base de Conocimiento del sistema (creando nuevos Objetos o actualizando Objetos existentes) con la información obtenida de las fuentes de información.

3.4.2. Unidad de análisis y razonamiento

Constituyendo la Capa de Análisis, se encuentra la Unidad de Procesamiento.

Definición. Una ***Unidad de Procesamiento (UP)*** es un módulo único en un Sistema de Detección de Alertas, cuya función principal es el análisis y el procesamiento de toda la información obtenida sobre los acontecimientos que tiene lugar en el escenario de estudio. Se representa por una 3-tupla **UP=<BC, P, MDAs>**, donde:

- **BC** es la *Base de Conocimiento* del Sistema. Se trata de uno de los módulos más importantes ya que todo el conocimiento del sistema reside en él. En esta Base de Conocimiento queda almacenada de forma integrada toda la información, tanto la que se obtiene de los distintos sistemas de extracción de conocimiento que se empleen como entrada en el SDA, como el nuevo conocimiento generado en el mismo (tanto en la capa de fusión e integración como en la capa de análisis).

  La información recogida en este módulo será representada bajo el esquema de representación de conocimiento descrito en la *ORHC*. Para seguir ese esquema conceptual, nos basamos en la idea de que la Base de Conocimiento es un *almacén* que recoge todos los datos obtenidos sobre los *objetos* que aparecen en la escena, de ahí que exista una estructura conceptual para definir este tipo de información. Esto podrá verse de forma detallada en la Ontología de Representación Homogénea del Conocimiento, definida en la sección 3.5.

  Un cambio en uno de los objetos de la escena puede conllevar, o no, una modificación de los niveles de presencia de las situaciones de estudio o alertas.

  En la Base de Conocimiento diferenciaremos dos tipos de Objetos del Sistema, a pesar de que ambos se representen bajo el mismo formato conceptual:

  - *Objetos Dinámicos*. Estos *objetos* se crean, se actualizan y se eliminan de forma dinámica conforme vayan apareciendo o desapareciendo en el escenario vigilado. Con este tipo de objetos se puede representar a las personas, vehículos, etc. detectados a partir de un análisis de vídeo, o también para

representar activaciones de sensores concretos en un momento dado, o la detección de un sonido concreto en el entorno.

La principal característica de estos objetos radica en que tienen un tiempo de vida en la BC, es decir, son eliminados cuando se deja de tener constancia sobre ellos.

- *Objetos fijos*. Son aquellos *objetos* que pertenecen de forma permanente a la Base de Conocimiento. Este tipo de Objetos se caracterizan por tener una *cualidad* que indica que son objetos fijos y, por lo tanto, no pueden ser eliminados de la BC. Esto no impide que puedan actualizarse. Este tipo de objetos representa a *ítems* fijos del entorno, por ejemplo, a los micrófonos o sensores instalados en el escenario.

- **P** es el conjunto de *plugins* o módulos dotados con procedimientos adicionales cuya función principal es la generación de nuevo conocimiento a partir del conocimiento existente. Los plugins serán los encargados de actualizar la Base de Conocimiento, actualizando y complementando la información que se conoce sobre los objetos o acontecimientos que tienen lugar en la escena. Pueden crearse tantos plugins como se necesiten en el sistema.

  Uno de los plugins cuya existencia es necesaria y que en este Modelo proponemos es el *Plugin Eliminador de Objetos*:

  - El **Eliminador de Objetos (EO)** es un plugin básico para todo Sistema de Detección de Alertas. Consiste en un proceso que se activa cada cierto tiempo y que tiene como función comprobar que los objetos existentes en la Base de Conocimiento sean objetos que actualmente forman parte de la escena, lo que denominaremos Objetos Activos. Si el EO encuentra Objetos que no han sido actualizados durante un tiempo, los considerará como inactivos en el escenario y los eliminará.

    Para llevar a cabo este procedimiento, cada objeto tendrá asociado un grado de creencia que reflejará la posibilidad de que dicho objeto esté actualmente presente en la escena. Este grado de creencia es denominado *índice de vigencia del Objeto* (y representará el nivel de actividad del mismo).

Cuando un nuevo objeto aparece en el sistema, se crea con este grado de creencia igual a 1. El eliminador de objetos se encargará de decrementar poco a poco este índice de vigencia en el tiempo si no se van recibiendo eventos sobre dicho Objeto. Concretamente, si pasa un determinado tiempo t sin que un Objeto sea actualizado, entonces el EO decrementa el índice de vigencia del objeto con una cantidad c (índice de vigencia - c). Cuando el valor llega a 0.0, el Objeto en cuestión se elimina de la BC. t y c son parámetros de configuración de este plugin, que pueden ser ajustados para cada SDA desarrollado.

La idea es que si no se reciben eventos acerca de un Objeto, es porque probablemente haya desaparecido de la escena. Si el Objeto es actualizado, el grado de creencia que indica la vigencia del Objeto en la escena se actualiza a 1.

La existencia de este procedimiento es muy importante ya que el hecho de no realizar una buena limpieza de la Base de Conocimiento podría dar lugar a la detección de falsas alarmas.

- **MDAs** son los denominados *Módulos de Detección de Alertas*. Existirá un módulo independiente para cada situación de alerta que se estudie en el sistema. Estos módulos, consisten en *Sistemas Expertos* que se alimentan de la información de la Base de Conocimiento para llevar a cabo su proceso de análisis.

  Cada vez que se actualiza la Base de Conocimiento, cada Módulo de Detección de Alertas comprobará si existen objetos que por sus características o acciones modifiquen el grado de creencia de la situación de estudio que contempla. Si es así, el módulo que tras procesar la información detecte una modificación en la presencia de la situación que está analizando, actualizará el nivel de su alerta.

  Cada MDA puede especificar cuándo quiere ser ejecutado, por ejemplo, siempre que haya un cambio en la BC, sólo cuando cambie la localización de los objetos, o cuando se actualicen las acciones de los objetos...

  El diseño propuesto y la creación de MDAs independientes

permite que cada alerta pueda ser controlada de forma diferente. Las características de las distintas situaciones de estudio influirán en la elección de la técnica empleada para llevar a cabo el control de la alerta. Las distintas técnicas que se podrán usar son: controladores difusos, sistemas basados en reglas, los modelos de Markov, redes Bayesianas, redes Neuronales, técnicas basadas en lógicas ....

Los MDAs deben poder *generar de forma automática* ***explicaciones*** sobre los acontecimientos que influyen en la alteración de los niveles de alerta. Este hecho es posible gracias al uso de la Ontología Homogénea para la Representación del Conocimiento. Esta Ontología permite definir y detallar toda la información conocida sobre los objetos que intervienen en la escena. De esta forma, al analizar el comportamiento de los mismos, para identificar circunstancias que indican la presencia de un riesgo, se puede conocer cuáles son los objetos que influyen en la alerta y qué cualidades y acciones de los mismos intervienen en dicha detección.

Una vez conocidos estos datos, se pueden establecer una lista con aquellos objetos que incrementan el nivel de alerta (indicando cuáles son las características, acciones y/o cualidades que han modificado directamente el nivel de presencia de peligro) y generar una explicación, también en modo texto, fácilmente interpretable por un usuario.

Uno de los requisitos que deben tener en cuenta los MDAs es su **independencia del contexto**, es decir, deben ser diseñados para cualquier escenario de estudio, y por lo tanto, fácilmente adaptables a otros entornos de aplicación. Este requerimiento es fundamental para construir SDAs escalables, flexibles y portables.

3.4.3. Transmisión de alertas

El proceso de transmisión de la información acerca de las distintas situaciones de análisis también requiere importancia. De igual modo que en este trabajo nos basamos en la reutilización de sistemas inteligentes que aporten datos básicos sobre la detección de objetos en un entorno

monitorizado, también pretendemos que los datos obtenidos por los SDAs puedan ser empleados en otros estudios que se centren en otros aspectos posteriores al proceso de vigilancia, como los ya nombrados gestores de crisis que generen protocolos de actuación ante los peligros detectados.

No queremos que la transmisión de alertas sea trivial, sino que proponemos que se realice en *tiempo real* y de forma *sensible al contexto*, como exponemos a continuación.

Con el fin de que varios receptores conozcan el estado de las situaciones de riesgo estudiadas por el SDA (ver definición de alerta en sección 3.5), presentamos un nuevo componente denominado *Notificador de Alertas*. Este componente del sistema será el responsable de la publicación de las alertas.

Definición. Un ***Notificador de Alertas*** es un módulo o componente único en el SDA, cuya función es mantener informada a toda entidad (proceso, sistema, componente, etc.) interesada sobre el estado de las distintas situaciones críticas estudiadas en el sistema. Se puede representar por una 2-tupla **NA=<A,(E-a-C)*>**, donde:

- **A** es el conjunto de *Alertas* a notificar y que son objeto de estudio del sistema.
- **(E-a-C)*** es el conjunto de uno o varios triples (Entidad-alerta-Contexto). Un triple (E-a-C) indica que una determinada *Entidad 'E'* quiere ser informada acerca del estado de una determinada alerta *'a'* en función del *Contexto 'C'* en el que se encuentra (o en base a sus necesidades). Con *entidad* nos referimos a cualquier otro componente, proceso o sistema. Con el *contexto* nos referimos a la especificación (por parte de la entidad) de la alerta a la que se subscribe, a partir de qué umbral de alerta desea recibir información y cada cuánto tiempo desea ser informado.

La inscripción o *suscripción* será un requisito previo para que una entidad o cliente pueda recibir notificaciones sobre el estado de las alertas. Concretamente, en una suscripción, el suscriptor debe especificar tres parámetros:

- *Id*: es el identificador de la alerta de la que se desea recibir notificaciones.
- *Umbral*: es un valor entre 0 y 1 que indica que el usuario desea recibir información sólo cuando el nivel de alerta sea superior a este umbral.
- *Tiempo*: es el número de milisegundos que indica la frecuencia con la que el usuario quiere ser informado. Si este parámetro es mayor que 0, la notificación contendrá el máximo nivel de alerta en ese período de tiempo.

Si el cliente desea ser informado de todos los cambios, los parámetros de *umbral* y de *tiempo* (especificados anteriormente) deben ser 0. Estos parámetros se ajustan dependiendo de la capacidad del dispositivo del cliente o de las circunstancias del usuario. De esta manera, podemos observar que el proceso de notificación será sensible al contexto, es decir, de acuerdo con las necesidades y el contexto del suscriptor.

Para llevar a cabo, y de forma efectiva, este proceso de transmisión, en el Notificador de Alertas existirá una *hebra* que lleve el control de los diferentes tiempos de los clientes. Esta hebra dormirá cuando no haya notificaciones y se despertará cuando sea necesario notificar el estado de alerta a un cliente.

En la figura 3.3 se muestra un ejemplo sobre el nivel de alerta que el sistema notifica a dos clientes. Uno de ellos desea recibir información cada 5 milisegundos y sólo cuando el nivel de alerta sea mayor que 0,5. El otro quiere ser informado cada 3 milisegundos, y solo cuando el nivel de alerta sea superior a 0,8.

Una de las ventajas más importantes que presenta esta transmisión dependiente de contexto es la ***reducción del tráfico en la red***. Por ejemplo, supongamos que se recibiesen eventos de detección de objetos a partir de vídeo a 24 frames por segundo, lo que podría implicar que una situación de alerta se evalúe y actualice 24 veces por segundo. Una transmisión trivial publicaría el estado de la misma también a 24 veces por segundo. En este caso, si hay muchas alertas y muchos subscriptores,

el tráfico de eventos es muy denso.

El hecho de tener una notificación de alertas en tiempo real y sensible al contexto, permite que los clientes puedan recibir notificaciones sólo de determinadas alertas cada cierto tiempo, por ejemplo, cada segundo, y también filtrar la información por umbral. De esta forma, el tráfico de transmisión de eventos sobre las alertas se disminuye, evitando el envío de eventos innecesarios. De este modo, este Notificador evita la sobrecarga de la red.

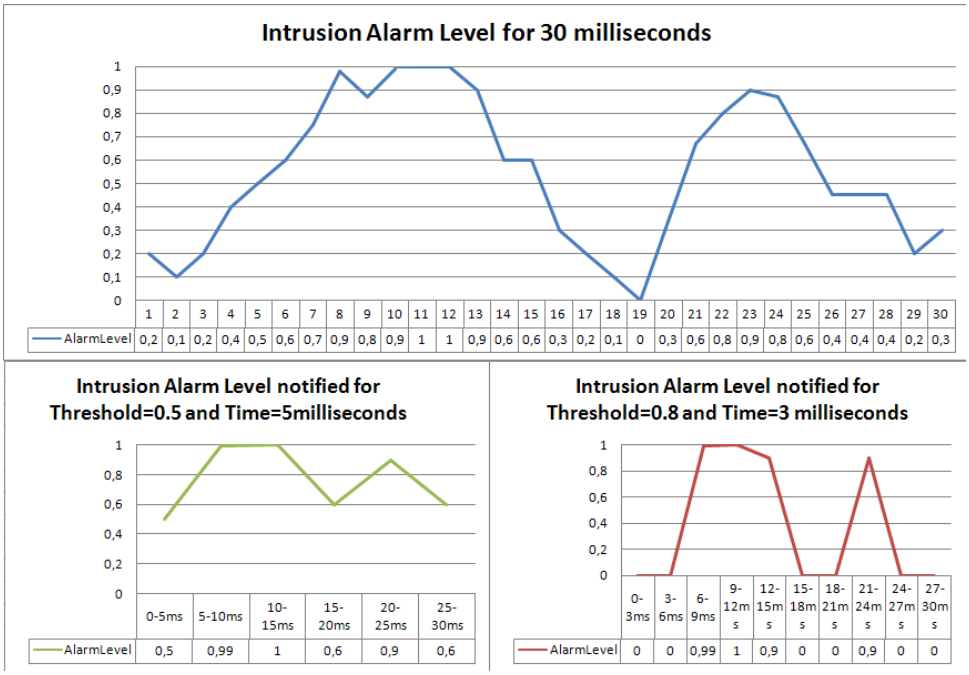


Figura 3.3: Notificación de Alertas en tiempo real y sensible al contexto.

3.4.4. Arquitectura

En la figura 3.4 podemos ver la arquitectura correspondiente al *Modelo* que proponemos.

Como podemos observar, se trata de una arquitectura inspirada en un diseño basado en capas [22, 85]. La información de entrada al

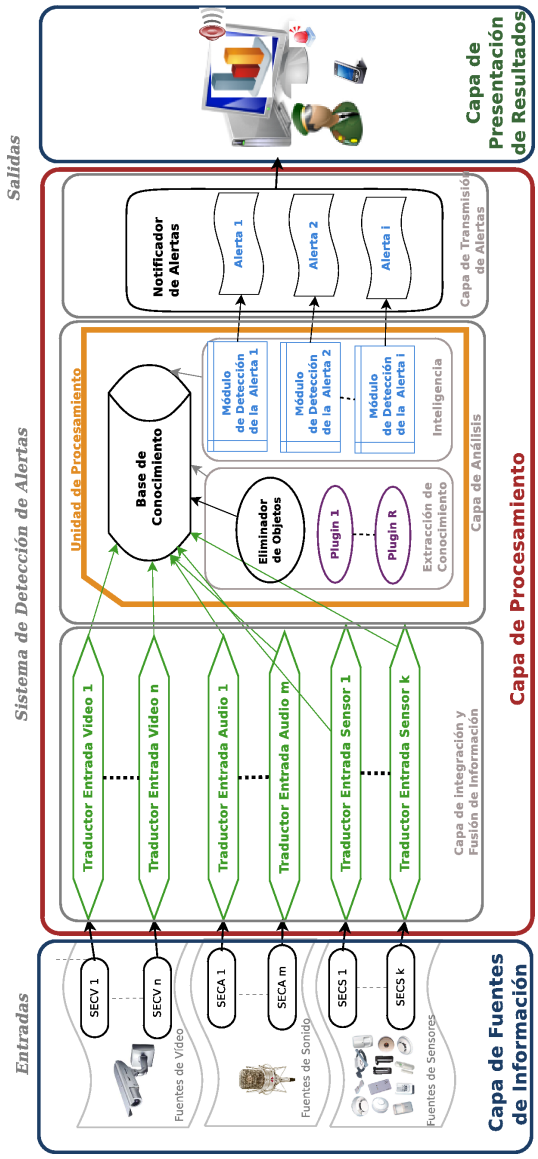


Figura 3.4: Arquitectura propuesta.

Sistema de Vigilancia pasa por dos fases, que coinciden con las dos capas principales diferenciadas en el Modelo: 1) Fase de fusión, integración y pre-procesamiento de la información de entrada al sistema, llevada a cabo en los Traductores; y 2) Fase de estudio, procesamiento y análisis del conocimiento de entrada y del generado en el sistema, llevada a cabo en las Unidades de Procesamiento. En esta última fase, se realiza el control de las situaciones de estudio. Posteriormente tiene lugar una tercera fase que consiste en la publicación de alertas.

Además de ser una arquitectura basada en capas, también está inspirada en desarrollo de software basado en componentes [51, 141], con el objetivo de crear sistemas distribuidos basados en componentes software reutilizables. El Modelo propuesto diferencia tres tipos de componentes principales: los Traductores, los Plugins, y los Módulos de Detección de Alertas.

A la hora de desarrollar los componentes, hay que tener en cuenta que un componente debe definirse de forma general y sin un contexto específico, de forma que se pueda permitir su adaptación a distintos sistemas y contextos.

Esta arquitectura distribuida en capas y componentes se caracteriza por:

- Permitir desarrollos paralelos, en las distintas capas y componentes, lo que agiliza el tiempo de desarrollo del sistema.
- Crear aplicaciones más robustas debido al encapsulamiento, tanto en las capas como en los componentes.
- Ofrecer un mantenimiento y un soporte sencillo, ya que es más fácil cambiar un componente que modificar una aplicación monolítica.
- El razonamiento del sistema está modularizado o dividido en módulos que realizan análisis paralelos e independientes, por lo que se agiliza el estudio inteligente de la información.
- Los componentes son unidades independientes de desarrollo y tienen múltiples usos en diferentes sistemas de seguridad, lo que implica la reutilización de los mismos.

- Alta escalabilidad.

- Tener mayor flexibilidad. Se pueden añadir nuevos módulos para dotar al sistema de nueva funcionalidad.

  Si se quiere que el sistema se alimente de una nueva fuente de información, se añade un nuevo Traductor para integrar el conocimiento obtenido desde esa fuente.

  Si es necesario calcular nueva información del entorno acerca de los objetos, como por ejemplo, conocer qué posibles objetos detectados en la escena han podido causar un determinado sonido captado en el entorno, solo hay que desarrollar y añadir un nuevo Plugin que analice y obtenga dicho conocimiento.

  Si se requiere analizar una nueva situación de riesgo, basta con añadir otro Módulo de Detección de Alertas para estudiar ese nuevo aspecto.

Es importante resaltar que a pesar de ser una arquitectura distribuida, la información se encuentra centralizada en la *Base de Conocimiento*. Como veremos a continuación, este motivo hace que en muchos casos sea necesario garantizar la exclusión mutua.

Exclusión Mutua

En los sistemas que efectúan accesos concurrentes a información centralizada hay que realizar una protección de los datos. Este es el caso de las Unidades de Procesamiento, concretamente en la Base de Conocimiento, donde se centraliza la información proveniente de las distintas fuentes y el conocimiento generado en el sistema.

Por la naturaleza del problema, se pueden llegar a recibir numerosos eventos que necesitarán actualizar la Base de Conocimiento. Si estas actualizaciones sobre los objetos se realizan de forma simultánea y además, concurrentemente tanto los Plugins como los MDAs hacen lecturas sobre los mismos, se puede llegar a producir incoherencias en los Objetos y, por lo tanto, provocar errores en la fase de Detección de

Alertas. Por esta razón, es necesario el uso de la exclusión mutua tanto para la lectura como para la escritura sobre la Base de Conocimiento.

Para garantizar exclusión mutua, este Modelo propone el uso de *cerrojos* con el fin de que se acceda a los recursos de forma controlada (en este caso los recursos son los Objetos del Sistema). Cuando haya varias hebras o procesos que intenten acceder a la BC, solo uno de ellos lo hará, llegando a ser dueño de un recurso en un determinado momento. Los demás procesos esperarán su turno. De esta manera, cuando los MDAs chequeen la Base de Conocimiento se hará en exclusión mutua. De igual modo, cuando el traductor realice la lectura de los Objetos o cuando necesite escribir la información de entrada en la BC, bien por la creación de un nuevo Objeto o por la actualización de uno existente, también se hará en exclusión mutua.

El hecho de controlar una exclusión mutua permite evitar posibles incoherencias y, por lo tanto, falsas alertas.

Un sistema en el que se llevó a cabo un proceso similar para garantizar exclusión mutua al trabajar con información centralizada es descrito en [129, 130].

Tiempos de procesamiento

Los Sistemas de Detección de Alertas son aplicaciones pensadas para detectar riesgos en tiempo real. Por estos motivos, los distintos componentes de la arquitectura deben de diseñarse de la forma más eficiente posible.

Uno de los aspectos a tener en cuenta es que, el hecho de usar como entrada resultados de sistemas que extraen un conocimiento inicial sobre el nivel sensorial, implica que dicha fase conlleve un tiempo de procesamiento. Este tiempo supone un retardo o *delay* sobre los acontecimientos que tienen lugar en el escenario. Por ello, el sistema final debe ser eficiente para que el delay de la alerta sea mínimo. Evidentemente, no se debe emplear como fuente de información en un SDA un sistema con altos tiempos de procesamiento.

Como se ha señalado anteriormente, la exclusión mutua permite

evitar posibles incoherencias. Sin embargo, al aplicarla, surge un problema que debemos tener en cuenta en cuanto a los tiempos de procesamiento, y es el tiempo de las conexiones. Cuando se recibe un evento que indica la activación de un determinado sensor, o un evento sobre la detección de un sonido concreto, se actualiza los datos de la BC de la forma correspondiente, en una sola conexión.

Sin embargo, el problema surge cuando se reciben datos sobre el análisis de vídeo, en este caso, el número de eventos puede ser muy elevado. Pongamos un ejemplo, supongamos que se analizasen unos 8 frames/segundo, y en cada frame se detectase información acerca de unos 10 objetos. Se crearían 10 conexiones por frame, cada una para realizar la actualización de cada Objeto en exclusión mutua. El tiempo de procesamiento global es más elevado, ya que cada actualización debe esperar a la anterior. Por este motivo, con el objetivo de disminuir los tiempos de procesamiento, en el Modelo propuesto se permitirán dos tipos de actualizaciones sobre la Base de Conocimiento:

- *Actualización en Bloque*. Se permite la actualización de varios Objetos independientes a la vez, en una sola conexión. Será usada al recibir eventos de vídeo. De esta manera se crea una conexión por frame y se reduce el tiempo de procesamiento.
- *Actualización Simple*. Se actualiza un único Objeto de la Base de Conocimiento.

De esta forma, se consigue actualizar el sistema al ritmo de los rápidos flujos de entrada (p.ej. el streaming de vídeo).

Middleware: Canales de Eventos

Para llevar a la práctica el desarrollo del Modelo de sistema de vigilancia que aquí proponemos, es necesario establecer un middleware de comunicación. El middleware es un software de conectividad que ofrece un conjunto de servicios que hacen posible el funcionamiento de aplicaciones distribuidas sobre plataformas heterogéneas. Funciona como una capa de abstracción de software distribuida, que se sitúa entre las capas de aplicaciones y las capas inferiores (sistema operativo y red).

El middleware nos abstrae de la complejidad y heterogeneidad de las redes de comunicaciones subyacentes, así como de los sistemas operativos y lenguajes de programación, proporcionando una API para la fácil programación y manejo de aplicaciones distribuidas.

Dada la naturaleza del Modelo planteado, la comunicación entre las capas y componentes de la arquitectura se realizará mediante canales de eventos. En este caso, hablamos de eventos en el sentido usual de los servicios en red.

En una arquitectura clásica cliente/servidor, el cliente se comunica con el servidor para solicitarle alguno de los servicios que éste ofrezca y el servidor, en esa misma conexión, devuelve los datos pertinentes al cliente. Sin embargo, en ocasiones es necesario, o más cómodo, que un cliente reciba la información sin que haya un proceso de solicitud, es decir, de forma asíncrona. Los canales de eventos representan la solución a este problema. Usando dichos componentes, los clientes se subscriben a un canal de eventos y se mantienen a la escucha de noticias (actuando como receptores). El servidor, que gracias a la suscripción conoce los receptores de los clientes, podrá enviarles datos cuando estos se produzcan.

Este modelo es usado, por ejemplo, en los correos electrónicos. Cuando un correo nuevo llega, si el cliente tiene la aplicación abierta es informado de ese nuevo correo. Anteriormente, sin el uso de canales de eventos, se tenía que efectuar peticiones regularmente y preguntar al servidor por nuevos correos. Evidentemente, la solución con canales de eventos es más rápida y sobrecarga menos al sistema.

Un ejemplo de integración de sensores de RFID mediante canales de eventos se describe en [132].

3.5. Representación del Conocimiento.

Consideramos tres niveles de representación del conocimiento:

1. Representación de las Entradas.

2. Ontología para la Representación Homogénea del Conocimiento (ORHC). En este caso, nos referimos al conocimiento interno del sistema.

3. Representación de las Salidas.

En cada uno de estos niveles proponemos una Ontología. Según la clasificación de Van Heist [56], estas ontologías son Ontologías del *modelado del conocimiento* ya que especifican conceptualizaciones del conocimiento. Se caracterizan porque contienen una rica estructura interna y están ajustadas al uso particular del conocimiento que describen.

Tanto la $ORHC$ como el esquema conceptual para la representación de las salidas son Ontologías fijas del Modelo propuesto. Sin embargo, la representación de las entradas es una propuesta orientativa de la información básica necesaria para un sistema de detección de alertas. En este último caso, la Ontología de la representación de las entradas puede cambiar dependiendo de las fuentes de información que se consideren en cada aplicación, a la hora de instanciar el Modelo propuesto.

3.5.1. Representación de las Entradas

Como se ha dicho anteriormente, para la representación de las entradas proponemos una Ontología orientativa con el fin de mostrar un ejemplo de un conjunto de datos básicos que puede necesitar un SDA. Esta Ontología se definirá concretamente y de forma específica para cada aplicación, una vez que se conozcan las fuentes de información del sistema a desarrollar.

El Modelo propuesto permite la creación de sistemas que se alimentan de fuentes heterogéneas. De este modo, un mismo sistema puede recibir eventos sobre análisis de vídeo, audio u otros sensores. Esto no implica que siempre sea así, sino que en cada aplicación se definirán cuáles son las entradas. Este hecho permite, por ejemplo, la creación de sistemas que sólo usen información de vídeo (si sólo con esa información es suficiente), o bien, sistemas que se alimenten de las tres fuentes.

La *información obtenida tras un análisis cognitivo de vídeo* constituye una información esencial y básica para cualquier tipo de Sistema de Detección de Alertas. Por consiguiente, proponemos que cualquier sistema basado en este Modelo reciba al menos como entrada eventos sobre la detección de objetos a partir del análisis del contenido del vídeo. Estos eventos deben de contener, al menos, los siguientes datos necesarios y básicos, representados por el siguiente cuádruple **(c,f,t,o)** donde:

- **c** denota el identificador de la cámara que emite el vídeo que se está analizando.
- **f** es el número del frame de vídeo.
- **t** es el tiempo en el que es anotado el frame de vídeo.
- **o** es una la lista de objetos detectados en la escena en ese frame determinado. Estos objetos son representados con la 4-tupla (i,p,t,c), donde:
  - **i** es el identificador del objeto.
  - **p** es la posición 2D del objeto dentro de la imagen de la cámara (x,y). Esta posición es medida en píxeles.
  - **t** representa el tamaño 2D (t_x, t_Y), medido en píxeles.
  - **c** la clasificación del objeto como persona, vehículo u otro, con un grado de posibilidad sobre esa clasificación.

Si el sistema recibe *eventos sobre el análisis de audio cognitivo*, esta información debería de contener al menos los siguientes datos representados por la siguiente dupla **(m,i)**, donde:

- **m** es el identificador del micrófono que ha detectado el evento.
- **i** es el identificador del evento detectado, es decir, la clasificación del sonido identificado, por ejemplo 'disparo'. Sería importante que este dato estuviese acompañado de un grado de creencia que indique la posibilidad de que dicho evento sea clasificado como 'i'.

Si además el sistema recibe *información sobre otro tipo de sensores*, estos eventos debería de contener al menos los siguientes datos representados por la siguiente dupla **(s,x)**, donde:

- **s** es el identificador del sensor.
- **x** es la acción que ha detectado el sensor, (*activación* o *desactivación*). Este dato también podría ir acompañado de un índice de creencia que indique el nivel de certeza con el que ha sido detectada dicha acción.

3.5.2. Ontología para la Representación Homogénea del Conocimiento (ORHC)

Con el objetivo de integrar la diversa información de entrada al sistema, proponemos una Ontología genérica que permite unificar la información heterogénea obtenida. Sobre dicha Ontología se llevará a cabo un proceso de análisis para identificar las situaciones de interés que se pretendan estudiar.

Esta representación está basada en la idea de que la mayor parte del conocimiento obtenido sobre los acontecimientos que tienen lugar en un espacio protegido está ligado a los objetos o eventos temporales que aparecen en el escenario. De ahí que el esquema conceptual, que vamos a definir, se base en una estructura que integrará toda la información correspondiente a los Objetos que aparecen en la escena. Con esta estructura conceptual pretendemos representar tanto a los objetos identificados a partir de vídeo, como los eventos detectados a partir de un análisis de audio o sensores. Esta estructura será denominada como *Objetos del Sistema*.

Definición: ***Objetos del Sistema***.

El concepto de *Objeto* es definido por una **7-tupla (i, g, li, t, c, a, loc)** donde:

- **i** es el *identificador del Objeto*. Se trata de una clave principal que identificará unívocamente al objeto.

- **g** es un *grado de creencia*, que varía entre 0 y 1 e indica la posibilidad de que el Objeto esté presente sobre el escenario de estudio en el momento actual. O dicho de otra forma, es el grado de creencia correspondiente a la vigencia del Objeto, ya que indica el nivel de actividad del mismo en el escenario.

  Inicialmente, un Objeto tiene este índice a uno, y se irá decrementando poco a poco en el tiempo, si no se van recibiendo eventos sobre él. El hecho de que llegue a cero, indicará que no ha sido actualizado desde hace un tiempo considerable y que, por lo tanto, podría eliminarse para dejarlo de tener en cuenta en procesamientos posteriores.

- **li** es una *Lista de identificadores*. Contiene otros posibles identificadores del Objeto. Es importante tener esta lista porque puede darse el caso de que dos sistemas de extracción de conocimiento o incluso en uno mismo, se haga referencia a un mismo Objeto con otro nombre. Debemos de abstraer este hecho y hacer que el Sistema propuesto sea consciente de cuándo se trata de un mismo objeto, evitando tener posibles Objetos duplicados.

- **t** es el *tiempo* de la última actualización del objeto en el sistema. Este parámetro es medido en milisegundos. El origen de tiempos considerado es el 01/01/1970.

- **c** representa *las **cualidades** del Objeto*. Una cualidad alude a un atributo o propiedad de un Objeto. Se va a representar con un triple *(c,v,g)*, donde:

  - **c** denota la clase o el tipo de cualidad. Se puede utilizar para agrupar cualidades de forma conceptual.
  - **v** es el valor de la cualidad.
  - **g** es un grado de creencia entre 0 y 1, que indica la posibilidad de que el objeto manifieste la cualidad de valor v.

  Para indicar que un objeto es fijo en la Base de Conocimiento y su índice de vigencia g no puede ser decrementado para una posible eliminación, se debe especificar la siguiente cualidad: *("objeto","fijo",1.0)*.

A continuación veamos otros ejemplos sobre posibles cualidades que puede tener un Objeto en forma de triple (clase, valor, grado de creencia):

- (“clase”,“persona”,1.0)
- (“clase”,“vehículo”,0.7)
- (“clase”,“desconocido”,0.2)
- (“velocidad”,“rápido”,0.7)
- (“sonido”,“alto”,0.9)
- (“tipo_evento”,“audio”,1.0)
- (“tipo_evento”,“video”,1.0)

- **a** representa *las* ***Acciones*** *del Objeto*. Cada objeto tendrá un histórico o lista con las distintas acciones que va desarrollando dentro de un escenario. Una acción se define por un cuádruple *(i,g,*t_i*,*t_f*)* donde:
  - **i** denota el identificador o descripción de la acción.
  - **g** es un grado de creencia entre 0 y 1 que indica la posibilidad de que el objeto de estudio esté realizando o haya realizado la acción de identificador i.
  - t_i representa el tiempo cuando se inició la acción. El tiempo es medido en milisegundos. El origen de tiempos considerado es el 01/01/1970.
  - t_f representa el tiempo de finalización de la acción. El tiempo es medido en milisegundos. El origen de tiempos considerado es el 01/01/1970.

    Cuando se detecta una determinada acción en un objeto, inicialmente el tiempo de finalización es nulo. Este dato no se recoge hasta que no se identifica que el objeto ha terminado de realizar la acción. Sin embargo, también puede ocurrir que una acción empiece y acabe en el mismo momento, y el tiempo de inicio y fin sería el mismo. Este es el caso de la detección de la activación de un sensor, o de la identificación de un sonido en un instante dado.

Algunos ejemplos de acciones en forma de cuádruple (identificador, certeza, inicio, fin) son:

- (“entrar al garage”,0.9,“10:00 10/09/10”,“10:01 10/09/10”).
- (“salir del edificio A”,0.7,“10:40 10/09/10”,“10:41 10/09/10”).
- (“rotura de cristales”,0.2,“11:00 10/09/10”,“11:00 10/09/10”).
- (“disparo”,0.4,“11:10 10/09/10”,“11:10 10/09/10”).
- (“activación sensor de movimiento 1”,1.0,“12:10 12/09/10”,“12:10 12/09/10”).
- (“desactivación sensor de movimiento 1”,1.0,“12:40 12/09/10”,“12:40 12/09/10”).

En estos *ejemplos*, se ha usado el formato de “hh:mm dd/mm/yyyy” para representar el tiempo, ya que es más comprensible para el lector. Sin embargo, el tiempo se medirá en los milisegundos que han trascurrido desde el 01/01/1970.

- **loc** representa la ***Localización*** *del Objeto dentro del Escenario*. Cada Objeto tendrá una lista con las distintas localizaciones que va adquiriendo durante su actuación en el Escenario. Entendemos por localización lo referente a las posiciones, tamaños y velocidades que el objeto adopta durante la escena. Una localización queda definida con el *quíntuplo (p,v,tam,t,i)* donde:
  - **p** es la *posición real* del Objeto en el escenario. Se trata de un punto en el espacio 3D, representado por un vector (x, y, z). Esta posición es medida en metros.
  - **v** denota la *velocidad* del objeto en el escenario. Se representa por el vector (v_x, v_y, v_z). Este parámetros es medido en metros por segundo.
  - **tam** representa el *tamaño* del objeto en la realidad. Tiene el formato (t_x, t_y), donde t_y representa la altura y t_x la anchura, ambos son medidos en metros.
  - **t** es el *tiempo* en el que se anota que el objeto tiene la posición, la velocidad y el tamaño anteriores. Es medido en milisegundos. El origen de tiempos considerado es el 01/01/1970.

- **i** indica el *incremento de tiempo*, es decir, la diferencia de tiempo existente entre la última vez que se actualizó la posición y el tiempo actual en el que se está actualizando. También se medirá en milisegundos.

Puede ocurrir que se desconozca la localización de un objeto que se haya creado a partir de eventos de sonido o sensores, ya que es un proceso complejo que la mayoría de los trabajos de este tipo no contemplan. En este caso estos datos de localización serán nulos.

Pongamos algunos **ejemplos** sobre distintos objetos. En los ejemplos, veremos que se ha usado el formato de "hh:mm dd/mm/yyyy" para representar el tiempo, ya que es más comprensible para el lector. Sin embargo, recordamos que los tiempos se medirán en los milisegundos que han trascurrido desde el 01/01/1970.

Un *Objeto creado a partir de eventos de vídeo* podría ser:

- Identificador: Objeto01
- Grado de creencia: 0.9
- Lista de identificadores: Objeto00,Objeto01
- Tiempo de la última actualización: "10:16:48 10/09/10" .
- Cualidades: ("clase","vehículo",0.7),("velocidad","rápido",0.7)
- Acciones: ("circular por vía principal",0.7,"10:15 10/09/10","10:25 10/09/10"),("entrar al garaje",0.88,"10:25 10/09/08","10:27 10/09/10"), ("activación SensorMov01",1.0,"10:30 10/09/10","10:30 10/09/10").
- Localización.
  - Posición: (40,0.6,40).
  - Velocidad: (30,45,45).
  - Tamaño: (4.4,1.23).
  - Tiempo: "10:10:00 10/09/10"

- Incremento de tiempo: 500 milisegundos

- Posición: . . .
- Velocidad: . . .
- Tamaño: . . .
- Tiempo: . . .
- Incremento de tiempo: . . .

- Posición: (50,0.6,40).
- Velocidad: (31,20,45).
- Tamaño: (4.2,1.36).
- Tiempo: “10:27:40 10/09/10”
- Incremento de tiempo: 500 milisegundos

Este objeto representa a un vehículo detectado a partir de eventos de análisis de vídeo. A parte de conocerse su posición, tamaño y velocidades, se conoce que hay una gran posibilidad de que haya circulado por la vía principal hasta llegar al garaje y que posteriormente haya activado un sensor de movimiento. Este último dato referente a la relación entre el vehículo y la activación de un sensor, podría ser obtenido tras conocer la detección del vehículo con posición próxima a la situación del sensor de movimiento cuando éste ha sido activado. Como podemos observar, esta estructura conceptual permite fusionar datos de fuentes heterogéneas (en este caso vídeo y sensor) que hagan alusión a un mismo objeto.

Un *Objeto creado a partir de eventos de sonido* que representa a un evento de un sonido identificado podría ser:

- Identificador: ‘rotura de cristal’
- Grado de creencia: 0.5
- Lista de identificadores:
- Tiempo de la última actualización: “10:00 10/09/10” .

- Cualidades: (“evento”,“sonido”,1.0),(“volumen”,“alto”,0.7)
- Acciones: (“rotura de cristal”,0.7,“10:00 10/09/10”)
- Localización.
  - Posición:
  - Velocidad:
  - Tamaño:
  - Tiempo:
  - Incremento de tiempo:

Otro objeto creado a partir de eventos de sonido, pero que en este caso representa al micrófono que ha recogido el sonido sería:

- Identificador: ‘Microfono01’
- Grado de creencia: 1.0
- Lista de identificadores:
- Tiempo de la última actualización: “10:00 10/09/10” .
- Cualidades: (“sensor”,“microfono”,1.0),(“objeto”,“fijo”,1.0)
- Acciones: (“rotura de cristal”,0.7,“10:00 10/09/10”)
- Localización.
  - Posición:
  - Velocidad:
  - Tamaño:
  - Tiempo:
  - Incremento de tiempo:

Un *Objeto creado a partir de la información de sensores* podría ser:

- Identificador: SensorMov01

- Grado de creencia: 1.0
- Lista de identificadores:
- Tiempo de la última actualización: “10:30 10/09/10” .
- Cualidades: (“sensor”,“movimiento”,1.0),(“objeto”,“fijo”,1.0)
- Acciones: (“activación”,1.0,“10:30 10/09/10”),(“desactivación”,1.0,“10:35 10/09/10”
- Localización.
  - Posición:
  - Velocidad:
  - Tamaño:
  - Tiempo:
  - Incremento de tiempo:

Como podemos observar, la *ORHC* permite representar bajo un mismo esquema la distinta información de entrada proveniente de los distintos contextos: audio, vídeo y sensores. De esta forma, la diversa información heterogénea obtenida de diferentes análisis y que alude a un mismo objeto en la escena, queda integrada en un único concepto: un único Objeto del Sistema.

3.5.3. Representación de las Salidas

El análisis del sistema ofrecerá como salida final el estado de las distintas situaciones de estudio, el cual vendrá reflejado bajo un esquema conceptual denominado $alerta$.

Definición: Alerta.

Una *alerta* es definida como la presencia de un hecho o conjunto de circunstancias que requieren atención y vigilancia, ya que representan una situación de interés o de riesgo. Una Alerta en el Modelo propuesto se representa conceptualmente como una 6-tupla *(i, g, u, t, eo, et)* donde:

- **i** es el *identificador* de la Alerta. Se trata de un descriptor que identifica unívocamente la Alerta.
- **g** es el *grado de creencia* de la alerta. Índice que varía entre 0 y 1 e indica el nivel de posibilidad de presencia de la situación de alerta estudiada.
- **u** es el *Umbral* de Alerta. Se trata de un índice perteneciente al intervalo [0,1]. Cuando el grado de creencia de la Alerta supere el umbral, se considerará que dicha Alerta está activada y se debe emitir una alarma que llame la atención del usuario.
- **t** es el *tiempo* de la última actualización de la alerta.
- **eo** es una estructura que resume la explicación de la activación de la alerta. Consiste en una secuencia de *Objetos del Sistema*. Esta secuencia tiene la peculiaridad de que está constituida por aquellos objetos del sistema que han influido en la modificación del grado de creencia de la alerta. Estos objetos tendrán en su lista de acciones sólo aquellas que originan la alteración del nivel de alerta. Igualmente para las cualidades, solo contendrán las cualidades que hayan influido en la modificación de la alerta.
- **et** es la explicación de la activación de la alerta en formato texto. Este parámetro está enfocado para ser visualizado como un mensaje de texto.

Pongamos un **ejemplo**, supongamos que se está estudiando el peligro de atropello, y que esta alerta se dispara porque un coche y una persona van a estar probablemente en el mismo punto dentro de poco tiempo. En este momento la Alerta se definiría:

- Identificador: PeligroAtropoello01
- Grado de creencia: 0.9
- Umbral: 0.8
- Tiempo: “12:40 12/09/10”

- Explicación: **Objeto**(Objeto01,0.9,{$Objeto00, Objeto01$},
  Cualidades {(“$tipo''$, “$vehiculo''$, 0,7)},
  Acciones{(“$entraralgarage''$, $0'88$, “$10:1510/09/10''$, “$10:1610/09/10''$)},
  Localización(Posición (40,0.6,40), Velocidad (30,45,45), Tamaño (4.4,1.23), Tiempo: “10:16:00 10/09/10”, 0.5 segundos),“10:16:48 10/09/10”),
  Objeto(Objeto02,0.7,{},
  Cualidades {(“$tipo''$, “$persona''$, 0,9), (“$velocidad''$, “$rapido''$, 0,9)},
  Acciones{(“$saliralgarage''$, 0,6, “$10:1610/09/10''$, “$10:1610/09/10''$)},
  Localización(Posición (40,0.3,39), Velocidad (30,45,45), Tamaño (0.4,1.63), Tiempo: “10:16:00 10/09/10”, 0.3 segundos),“10:16:48 10/09/10”).

- Explicación Texto: “Un vehículo se dirige con gran velocidad hacia una persona en movimiento”.

3.6. Aplicabilidad del Modelo propuesto

En este capítulo, se ha propuesto un *Modelo* que sirve como base para el desarrollo de Sistemas de Vigilancia que se alimentan de fuentes heterogéneas. Estos sistemas han sido caracterizados por una serie de requisitos y han sido denominados en este estudio como *Sistemas de Detección de Alertas (SDAs)* (ver sección 3.1).

Este Modelo ha sido diseñado con el propósito de crear herramientas escalables, flexibles y portables a cualquier entorno. Se compone principalmente de una *arquitectura* bien definida y una *ontología* para la representación del conocimiento (descritas en las secciones 3.3,3.4 y 3.5).

El Modelo se inspira inicialmente en la idea de construir una herramienta inteligente capaz de identificar situaciones de interés o de riesgo de forma automática. La mayoría de este tipo de situaciones, tiene lugar en un espacio específico que se caracteriza por estar dentro de un contexto y por no tener dimensiones extremadamente grandes. Por ello, el Modelo presentado favorece la confección de sistemas enfocados a vigilar los acontecimientos de un único escenario de estudio: el *escenario local* (ver sección 3.2).

Se trata de un Modelo estándar que puede ser desarrollado e implementado en cualquier ámbito de seguridad. Podemos observar que dicho Modelo es el *esqueleto* del sistema a construir. Está definida su estructura, arquitectura, el tipo de componentes y sus características, pero no deja de ser un Modelo abierto, donde queda por hacer la instanciación de cada uno de los módulos propuestos para una aplicación concreta.

Gracias a su diseño basado en componentes, se permite la creación de nuevos componentes dentro de la arquitectura, lo que llevaría a la especificación del sistema de vigilancia concreto. Es decir, la instanciación de dicho *Modelo* para detectar un propósito específico a partir de unas determinadas fuentes de información dará lugar a un sistema de detección de alertas concreto.

El primer paso para desarrollar un sistema basado en nuestra propuesta es conocer qué situaciones de estudio se pretenden analizar. El segundo paso es concretar las fuentes de información necesarias del sistema. Y posteriormente crear los componentes necesarios para el desarrollo final. Se crearán tantos *Traductores* como fuentes de información diferentes tengamos, tantos *Módulos de Detección de Alertas* como situaciones se estudien y tantos *Plugins* auxiliares como sean necesarios para obtener nuevo conocimiento con el que puedan trabajar los MDAs. Pero estos componentes no se podrán definir con detalle hasta no conocer sus funciones específicas, las cuales se definirán para cada aplicación del Modelo. El paso final consistirá en la integración de los diversos componentes en la arquitectura presentada.

Si los MDAs se realizan siguiendo los requisitos indicados en el Modelo, se lograría que el sistema final sea independiente del contexto y, por lo tanto, se garantizaría la puesta en práctica del mismo sobre cualquier escenario local de estudio.

Las principales ventajas que presenta el Modelo propuesto son:

- La creación de herramientas *escalables* y *flexibles*.
- Si el entorno de monitorización cambia, el sistema es fácilmente *adaptable al nuevo entorno*. Habría que adaptar los parámetros de

configuración de cada uno de los componentes del sistema (en el caso de que los tenga).

- La definición de componentes independientes facilita la *reutilización* de los mismos en otros sistemas de seguridad.

A continuación, en el siguiente capítulo, se muestran tres aplicaciones de Sistemas de Vigilancia basados en el Modelo presentado. Para cada una de estas aplicaciones, se han definido en detalle los Traductores, Plugins y MDAs necesarios en cada caso.

Capítulo 4

Aplicaciones: Desarrollo de Sistemas de Detección de Alertas

Existen muchas situaciones que demandan vigilancia y seguridad con el fin de poder evitar ciertos riesgos. De este modo, el desarrollo de Sistemas de Vigilancia Inteligentes para la detección automática de anomalías o peligros constituye un punto importante de ayuda y soporte para el mantenimiento de la seguridad en un entorno protegido.

En este capítulo se describe el desarrollo de distintas aplicaciones de Vigilancia Inteligente. Estas aplicaciones están basadas en el Modelo de diseño para el desarrollo de sistemas de detección de alertas propuesto en este estudio (y descrito en el capítulo 3). Este Modelo ha sido precisado, especificado e instanciado con el fin de crear tres aplicaciones para la detección inteligente de riesgos.

Cada sistema se encargará de analizar y detectar de forma inteligente una alerta diferente. Las situaciones de estudio que se contemplan en dichas aplicaciones son situaciones complejas, de interés actual y que demandan un proceso de vigilancia. De forma concreta, nos referimos a:

1. La identificación de la presencia de **peligro de atropello**, con el

fin de prevenir un accidente donde un peatón pueda estar en riesgo porque un coche pueda atropellarlo.

2. La identificación de **peligro por niños** que no estén bajo la tutela de adultos *en una zona donde pueden transitar vehículos*.

3. La **detección de intrusiones** en un escenario determinado.

Cada una de estas aplicaciones está basada en un *controlador difuso* (o sistema experto basado en reglas difusas), que será el encargado, en cada caso, de detectar la presencia de anomalías que implique cada uno de los riesgos anteriores.

A continuación, pasaremos a describir las aplicaciones que se han llevado a cabo. Para cada sistema desarrollado, se muestran los siguientes ítems:

1. La *estructura general* y el *objetivo* que se quiere conseguir en el sistema.

2. La *arquitectura*, basada en el Modelo propuesto en el capítulo 3.

3. Los componentes que constituyen la arquitectura: *las fuentes de información*, los componentes de *Traducción*, el *sistema experto* propuesto para identificar la presencia de la situación de alerta *(Módulo de Detección de Alerta)* y los *Plugins* adicionales.

4. Las distintas herramientas software que constituyen la implementación final del Sistema.

5. Los desarrollos realizados para la etapa experimental y los resultados obtenidos tras la evaluación del Sistema de Vigilancia Inteligente propuesto.

Queremos resaltar que los sistemas expertos difusos que se han propuesto en este capítulo para analizar la presencia de las distintas situaciones de peligro son descritos en las secciones 4.1.7, 4.2.5 y 4.3.5.

4.1. Vigilancia inteligente para el análisis del peligro de atropello

La colisión entre objetos móviles en el espacio es uno de los riegos más comunes de la vida diaria. En muchas situaciones, si una colisión es detectada a tiempo, un hecho desagradable puede ser evitado. Este hecho está presente, sobretodo, en los accidentes de tráfico, por ejemplo: cuando un vehículo atropella a un peatón, cuando dos vehículos chocan, cuando un avión colisiona con un obstáculo o un vehículo en un aeropuerto, etc. Por esta razón, y con el propósito de aumentar la seguridad en muchos casos, hemos desarrollado un Sistema Experto que usando resultados de análisis del contenido del vídeo es capaz de predecir colisiones, detectar objetos en peligro y alertar de este hecho en tiempo real.

En este contexto, proponemos un Modelo Abstracto (ver sección 4.1.7) que permite predecir una posible colisión futura entre dos objetos cualesquiera. Este Modelo consiste en un Controlador Difuso que realiza un estudio de las propiedades de los objetos (localización, velocidad, trayectorias seguidas...). Como aplicación de este Modelo, hemos desarrollado su adaptación al caso concreto de predicción de colisiones entre vehículos y peatones. De esta forma, queremos detectar el riesgo de que un peatón esté en peligro porque un vehículo pueda atropellarlo.

4.1.1. Objetivo. Estructura General.

El propósito es aplicar el Modelo propuesto en el capítulo 3 para desarrollar una aplicación concreta de un Sistema de Vigilancia, cuyo fin sea la detección de riesgo de accidente de tráfico donde un peatón pueda ser víctima de un vehículo. El escenario de estudio será cualquier zona donde exista tráfico de vehículos y peatones simultáneamente.

A partir de un análisis de vídeo se pueden conocer datos acerca de los objetos que aparecen en la escena, como ¿cuántos vehículos y peatones hay?, ¿cuáles son sus posiciones y velocidades?, etc. Como veremos a continuación, estos datos constituirán la información necesaria

para realizar el análisis del peligro de atropello.

Por otro lado, para estudiar esta situación de riesgo no es necesario que haya micrófonos u otro tipo de sensores que monitoricen el entorno. El audio recogido en una zona de tráfico, ya sea urbana o rural, coincide con el sonido de tránsito de vehículos y el bullicio de la gente al pasar, lo que no aportaría una información adicional a la que se puede recoger a partir de vídeo. El hecho de que existan otro tipo de sensores (de movimiento, ...) tampoco ofrecería datos relevantes para la resolución del problema.

Por consiguiente, la única fuente de información será el vídeo. De este modo, el escenario de estudio estará únicamente monitorizado por, al menos, una cámara de vigilancia. Esta situación es reflejo de la vida real, ya que en muchos aspectos de la monitorización de tráfico, la única fuente de información del entorno son las cámaras de vigilancia.

Concluyendo: el objetivo del Sistema inteligente que proponemos será predecir cualquier posible colisión entre un vehículo y un peatón a partir del análisis de vídeo. A grandes rasgos, diferenciamos tres partes en la estructura general del Sistema, que serán detalladas más adelante (ver figura 4.1):

- La entrada del Sistema será un flujo de eventos sobre la detección y el tracking de objetos, a partir de un análisis de vídeo.

- El Sistema procesará la información de entrada, complementándola con nueva información para ser estudiada por un módulo experto que analice el peligro de atropello.

- La salida del Sistema será el nivel de presencia de peligro de atropello detectado. O dicho de otra manera, el nivel de certeza de que exista una posible colisión vehículo-peatón en un futuro cercano.

A continuación, describimos la arquitectura del Sistema de Vigilancia para la detección del peligro de atropello.

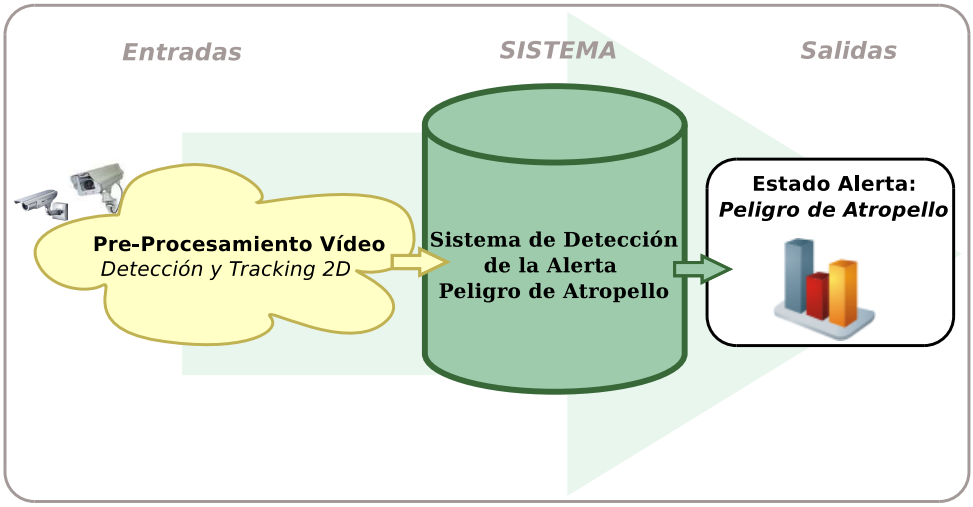


Figura 4.1: Estructura general del Sistema Inteligente para la Detección del Peligro de Atropello

4.1.2. Arquitectura

El Sistema de Vigilancia Inteligente para la Detección de Peligro de Atropello, según el Modelo propuesto en el capítulo 3, presenta una arquitectura basada capas y componentes que puede verse en la figura 4.2.

Como se ha señalado anteriormente, el Sistema recibe un flujo de entrada sobre un determinado análisis de vídeo (ver sección 4.1.3) y obtiene como salida el nivel de posibilidad de que exista una futura colisión entre un vehículo y un peatón.

La arquitectura está compuesta por los siguientes componentes:

- Un **Traductor**, que es el encargado de procesar la información de entrada para complementarla e integrarla en la Base de Conocimiento del Sistema. (Ver sección 4.1.4).
- La **Base de Conocimiento, BC**, que tiene toda la información referente a todos los *Objetos del Sistema*. En este componente se integra la información de entrada con la información obtenida tras

el procesamiento y el análisis interno del Sistema. Recordemos que en este componente, la información se representa bajo el esquema definido en la Ontología Homogénea para la Representación del Conocimiento (ver sección 3.5).

- Tres **Plugins**: (ver sección 4.1.8)
  - *Eliminador de Objetos, (Plugin EO)*. Es el encargado de eliminar los objetos, cuando estos hayan desaparecido del escenario.
  - *Visualización de la Base de Conocimiento, (Plugin VBC)*. Este plugin obtiene, en modo texto, toda la información actual residente en la Base de Conocimiento, con el objetivo de ser fácilmente legible por el usuario del Sistema.
  - *Identificación de Objetos Rápidos, (Plugin IOR)*. Se encarga de clasificar a los objetos como *rápidos* con un grado de creencia, en función de la velocidad de movimiento de los mismos.

- Un **Módulo de Detección de la Alerta Peligro de Atropello**, que será el encargado de analizar todo el conocimiento del Sistema con el objetivo de detectar circunstancias que impliquen la presencia de peligro de que un peatón pueda ser atropellado. (Ver sección 4.1.7)

- Un **Notificador de la Alerta**, en tiempo real y sensible al contexto. Este Notificador es un componente básico definido en el Modelo propuesto en este estudio y que es la base del Sistema actual. Este componente fue descrito en la sección 3.4.3.

Es importante resaltar que en este caso, tanto los *Plugins* como el *Módulo de Detección de la Alerta* se ejecutan cada vez que se detecte un cambio en la *Base de Conocimiento*.

A continuación, en los siguientes apartados, se definirán los distintos componentes de la arquitectura de forma detallada.

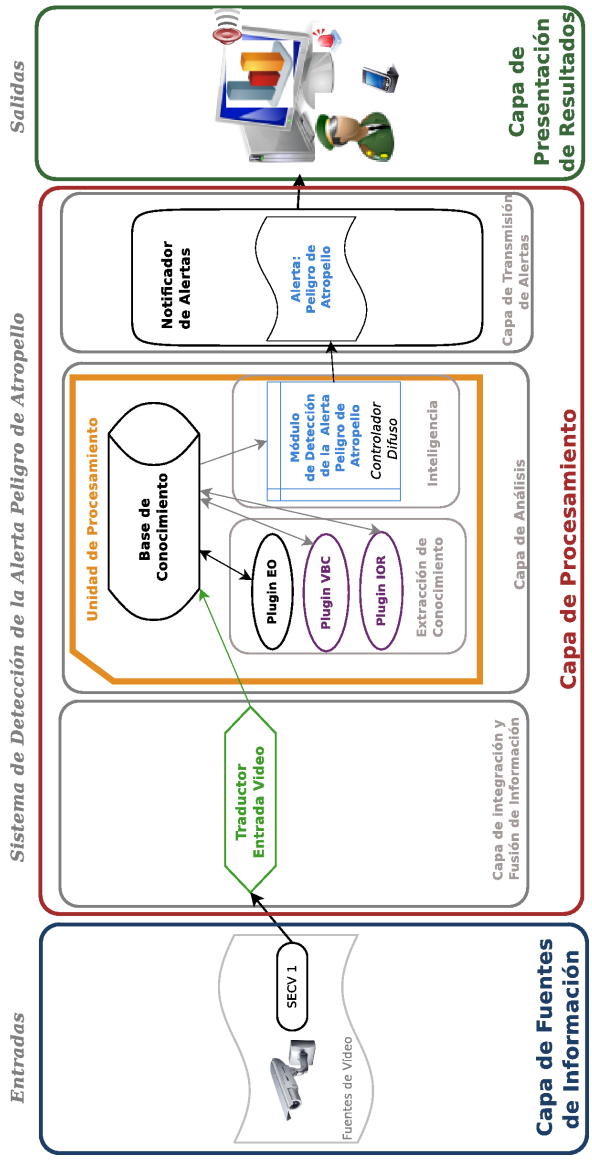


Figura 4.2: Arquitectura del Sistema de Vigilancia Inteligente para la Detección de Peligro de Atropello.

4.1.3. Fuentes de información

El módulo experto para la detección de futuras colisiones entre objetos, de forma concreta entre un vehículo y un peatón, necesita conocer determinados datos acerca de los objetos que aparecen en la escena. En este caso, los datos esenciales son:

1. La identificación de los objetos de la escena, es decir, ¿qué identificador tienen?
2. ¿Qué tipo de objeto es? ¿Es un vehículo o una persona?
3. ¿Cuál es su posición dentro del escenario?
4. ¿A qué velocidad se mueven?

Esta información puede conocerse a partir de un análisis cognitivo sobre vídeo. Cualquier sistema de extracción de conocimiento a partir de vídeo que realice una detección, clasificación y seguimiento de objetos, puede servir. Uno de nuestros propósitos es que el sistema que desarrollemos se diseñe sobre resultados de un trabajo real, con el fin de reutilizar código experto ya existente y construir un sistema robusto a resultados reales. De esta forma, logramos construir una herramienta basada en aplicaciones reales. Para ello, nos basaremos en un trabajo concreto que ha sido recientemente desarrollado por la Universidad Politécnica de Valencia y que se describe en [136].

El sistema de extracción de conocimiento sobre vídeo empleado realiza un proceso de detección y seguimiento de objetos en 2D. Este pre-procesamiento analiza el vídeo de cada cámara frame a frame y usa el formato MPEG-7 para hacer la anotación. En cada frame, se detecta un conjunto de objetos que son clasificados como *personas*, *vehículos* u *otros*, con un grado de creencia. Cada objeto detectado es englobado en una elipse, de la que se conocen sus dos radios (tamaño del objeto en dos dimensiones).

Para indicar el seguimiento 2D, el sistema de detección y tracking de objetos muestra, para cada objeto, cuál o cuáles son los identificadores de los objetos del frame antecedente de los que procede el objeto seguido.

De este modo, debemos tener en cuenta que un mismo objeto cambia de identificador de un frame a otro.

El flujo de eventos de vídeo generado por el análisis [136] inicial, que se recibirá como entrada en el Sistema, se define con el siguiente cuádruple **(c,f,t,o)** donde:

- **c** denota el identificador de la cámara que emite el vídeo que se está analizando.
- **f** es el número del frame de vídeo que se está analizando actualmente.
- **t** es el tiempo en el que es anotado el frame de vídeo. Es medido en milisegundos. El origen de tiempos considerado es el 01/01/1970.
- **o** es una la lista de objetos detectados en la escena en ese frame determinado. Estos objetos son representados con la 5-tupla (i,p,t,c,a), donde:
  - **i** es el identificador del objeto.
  - **p** es la posición 2D del objeto dentro de la imagen o frame (representada por el vector (x,y), medido en píxeles).
  - **t** representa el tamaño 2D (t_x, t_y), medido en píxeles. El tamaño consiste en el ancho y el alto de la elipse que envuelve al objeto en su detección.
  - **c** es un triple (cp,cv,co), donde cp es el grado de creencia (entre 0 y 1) de que el objeto i sea una persona, cv es el grado de creencia (entre 0 y 1) de que el objeto i sea un vehículo y cp es el grado de creencia (entre 0 y 1) de que el objeto i sea otra cosa.
  - **a** es el conjunto de identificadores de los objetos (en el frame anterior) de los que proviene el objeto actual que se está estudiando.

    Lo normal es que el objeto actual proceda de un único objeto del frame anterior, pero pueden darse las circunstancias de que el objeto detectado en el frame actual proceda de dos

objetos detectados en el frame anterior (fusión de objetos) o bien que dos objetos distintos en el frame actual procedan de un mismo objeto en el frame anterior (división de objetos). Estas situaciones son definidas con mayor detalle en la sección 4.1.6 y serán abordadas por el Traductor del Sistema.

Con el objetivo de usar los resultados de esta fuente de conocimiento, hemos creado un **Traductor** para dicha fuente.

4.1.4. Traductores

Existe una única fuente de información que alimenta al Sistema: el pre-procesamiento de vídeo que realiza una detección y seguimiento 2D de objetos [136]. De este modo, sólo es necesario crear un Traductor específico para dicha fuente.

Este componente recibe eventos sobre el flujo de vídeo descrito en la sección 4.1.3. Estos eventos son procesados obteniendo nuevos datos que complementan dicha información y que son necesarios para el análisis de la situación estudiada, como es el caso de las velocidades y de la posición real de los objetos en el escenario. Toda esta información (la de entrada y la procesada) es representada bajo el esquema conceptual definido en la Ontología (ver sección 3.5). Posteriormente, el Traductor se encarga de actualizar la Base de Conocimiento, creando o actualizando Objetos.

El Traductor creado para esta fuente concreta estará dotado de tres procedimientos que permiten generar nuevo conocimiento:

1. **Cálculo de velocidades**.

   Los autores en [136] no publican la velocidad del objeto (aunque la obtengan), porque el hecho de calcularla a nivel de píxel tiene un gran coste computacional que retarda la transmisión del evento.

   La velocidad 2D asociada a un objeto es necesaria en este estudio para poder evaluar las trayectorias de los objetos que pueden implicar un riesgo de atropello. Por este motivo, la calculamos y la añadimos en la información del objeto.

La velocidad es la magnitud física que expresa la variación de posición de un objeto en función del tiempo, o distancia recorrida en la unidad de tiempo. Como podemos obtener las diferentes posiciones del objeto en el mundo a lo largo del tiempo, podemos calcular su velocidad.

$$v_2 = \frac{(P_2 - P_1)}{(t_2 - t_1)}$$

Este resultado es ponderado con un valor ω respecto a la velocidad calculada para ese mismo objeto en el frame anterior. Esto disminuye posibles errores que pueden originarse con cálculos locales. De esta forma, la fórmula usada para el cálculo de la velocidad de un objeto en un instante t_2 sería:

$$v'_2 = v'_1\omega + v_2(1 - \omega)$$

Siendo v'_1 la velocidad final ponderada del objeto en el instante anterior t_1.

En el caso de que el seguimiento de un objeto indique que proviene de varios objetos del frame antecedente, la velocidad se estimará como la media de las distintas velocidades obtenidas con los respectivos objetos del frame anterior que dan origen al objeto seguido.

2. **Cálculo de la información de los objetos en el mundo real (posición y velocidad 3D)**, usando procesos de calibración de la cámara. Este procedimiento es descrito con mayor detalle en la sección 4.1.5.

3. **Seguimiento de los objetos a alto nivel**, basado en el posicionamiento 3D y en la clasificación de los objetos. Este proceso permite conocer cuándo hay que crear un objeto nuevo en el Sistema, o bien, cuándo existe el objeto y sólo habría que actualizarlo.

   Este seguimiento de alto nivel, se ha diseñado con el fin de realizar una buena gestión y un mantenimiento correcto de los objetos en la Base de Conocimiento, y además, conseguir un seguimiento multi-cámara. Este procedimiento es descrito con mayor detalle en la sección 4.1.6.

Este *Traductor* puede instanciarse tantas veces como cámaras de vigilancia existan en el escenario. Cada instancia del Traductor contendrá una configuración diferente sobre la calibración de cada cámara.

4.1.5. Cálculo de información sobre el mundo a partir de análisis de vídeo

En esta sección describimos un método que permite obtener el posicionamiento 3D de los objetos a partir de su posición 2D en la imagen, mediante procedimientos de calibración de la cámara. Como veremos a continuación, una vez calculada la posición real del objeto en el mundo, podemos generar nueva información acerca de su velocidad 3D, su altura real, su posición futura...

Uno de los puntos claves de nuestro trabajo es la interpretación semántica de fuentes de información, etapa de gran importancia dentro de un sistema de vigilancia. En este caso, vamos a procesar la información procedente de un análisis de vídeo para poder complementarla con nuevo conocimiento y, posteriormente, integrarla en el Sistema para realizar un razonamiento sobre los distintos acontecimientos que tienen lugar en el escenario.

El flujo de vídeo que el Sistema recibe como entrada es descrito con detalle en la sección 4.1.3. Recordamos que para cada frame, se recibe una lista de objetos detectados, donde para cada objeto se conoce el identificador, la posición 2D del objeto en la imagen, el tamaño 2D, el conjunto de identificadores de los objetos (en el frame anterior) de los que proviene el objeto seguido, y la clasificación en persona, vehículo u otro.

A este flujo de entrada, el Sistema añade automáticamente la velocidad 2D (v_x,v_y) de cada objeto (ver sección 4.1.4).

A partir de estos datos, podemos derivar otra mucha información como: la posición real de los objetos en el mundo, las velocidades reales del objeto, la posición futura de un objeto en el tiempo, el tamaño real de los objetos de la escena... que enriquecerá a nuestro Sistema para tomar futuras decisiones.

Nuestra propuesta calculará la información 3D mediante el calibrado de la cámara, de forma que los objetos de la escena queden representados unívocamente en el mundo real. Esto nos permitirá efectuar un seguimiento 3D que complemente y resuelva el problema múltiples cámaras.

Realizar un seguimiento de los objetos en tres dimensiones conlleva un paso previo muy importante que consiste en la transformación de las coordenadas 2D de una imagen en coordenadas 3D. Este paso no es trivial, por lo que requiere un procedimiento complejo basado en técnicas geométricas. Antes de mostrar el procedimiento que hemos llevado a cabo, sería interesante ver cómo una cámara realiza el proceso inverso, el cual consiste en plasmar el mundo real de tres dimensiones en objetos de dos dimensiones.

Modelo matemático para la transformación de coordenadas 3D en coordenadas 2D

Cuando una cámara toma una imagen de una escena, el mundo real de tres dimensiones desaparece, y queda proyectado en una imagen de dos dimensiones (ver figura 4.3).

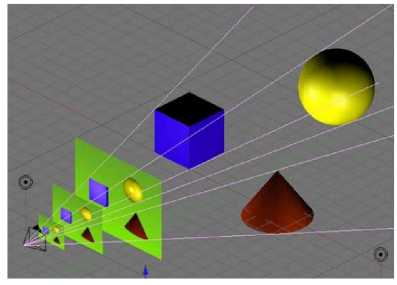


Figura 4.3: Imagen 2D que capta una cámara

El proceso de proyección, al fin y al cabo, es una transformación que convierte la representación tridimensional de una escena sobre un plano bidimensional. Por ejemplo, la proyección de un punto viene definida por la intersección entre el plano de proyección y el rayo que une dicho punto

con el centro de proyección. Y la proyección de una línea sigue siendo una línea, la cual se define uniendo las dos proyecciones de los extremos de la misma (ver figura 4.4).

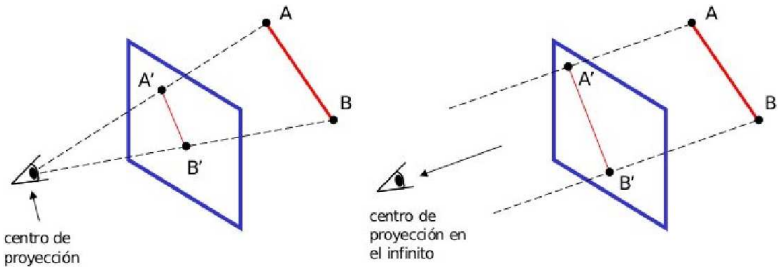


Figura 4.4: Proyecciones de una recta

Entre los distintos tipos de proyecciones existentes, la proyección *geométrica-planar* es la que se da cuando se obtiene una imagen de una cámara. Se denomina *geométrica* porque los rayos de proyección son rectos, y *planar* porque la superficie de proyección es un plano. La proyección geométrica-plana, a su vez, se divide en dos tipos: *perspectiva*, cuando la distancia del centro de proyección al plano es finita, y *paralela* cuando esa distancia es infinita (en este caso, sólo se especifica la dirección de vista y todos los rayos son paralelos).

Si pensamos en la escena que recoge una cámara, podemos observar que la imagen es una proyección geométrica-plana con perspectiva.

La proyección perspectiva también simula el comportamiento del ojo humano. Con esta proyección, se aumenta el realismo de la imagen (al dar sensación de profundidad) y el tamaño de un objeto varía inversamente proporcional a la distancia del objeto al plano de proyección. No es útil para reconocer formas ni medir longitudes ya que las distancias son falsas, los ángulos no se mantienen y las líneas paralelas dejan de serlo.

Supongamos que, a partir de un objeto del mundo en 3D, queremos calcular las coordenadas 2D del mismo objeto cuando la imagen de éste es captada por un observador, en este caso, una cámara. Para ello, colocamos un eje de 3 coordenadas, donde:

- El eje Y representa la vertical del observador.
- El eje X representa la horizontal del observador.
- El eje Z representa la dirección de vista.

La pregunta que nos haríamos sería: ¿Podemos calcular las coordenadas 2D de un punto *Q'=(x',y')*, proyectado sobre el plano de la imagen, a partir de un punto 3D del objeto *Q=(x,y,z)* en el mundo? (ver figura 4.5)

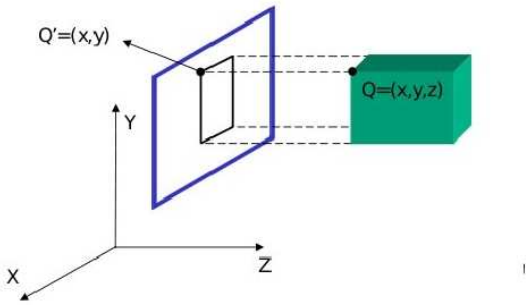


Figura 4.5: Proyecciones de un punto

Una solución podría ser aplicar el método de los triángulos semejantes (ver figura 4.6), suponiendo que la distancia 'd' al plano de proyección (distancia focal) es conocida:

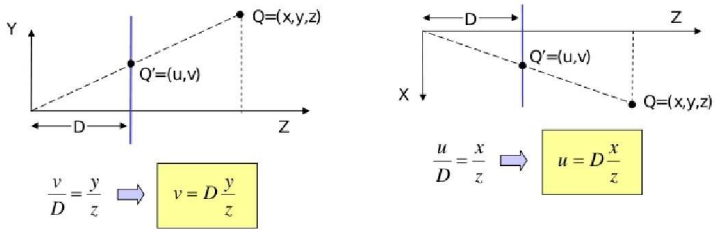


Figura 4.6: Método de los triángulos semejantes

donde obtendríamos,

- $\frac{x'}{d} = \frac{x}{z}$; $x' = d\frac{x}{z}$
- $\frac{y'}{d} = \frac{y}{z}$; $y' = d\frac{y}{z}$

La expresión final para la perspectiva sería:

$$x' = d\frac{x}{z};\ y' = d\frac{y}{z};\ z' = d$$

Los mismos resultados pueden obtenerse de una forma más automática, usando la denominada *matriz de perspectiva* P:

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \frac{1}{d} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

La transformación se obtendría de la siguiente manera:

$$Q' = QP$$

Tras aplicar la matriz, el resultado queda:

$$Q' = QP = (x, y, z, 1)P = (x, y, z, \frac{z}{d})$$

Si normalizamos, para poner a 1 la última componente, obtendríamos:

$$Q' = (x, y, z, \frac{z}{d}) = (x\frac{d}{z}, y\frac{d}{z}, z\frac{d}{z}, \frac{zd}{dz}) = (x\frac{d}{z}, y\frac{d}{z}, d, 1)$$

$$Q' = (x', y', z', h') = (x\frac{d}{z}, y\frac{d}{z}, d, 1)$$

$$x' = d\frac{x}{z}; y' = d\frac{y}{z}; z' = d$$

El problema es que no siempre es así de fácil. Este método sólo se puede aplicar para las condiciones establecidas, es decir, cuando el eje Z representa la dirección de vista del observador y los ejes X e Y representan la horizontal y la vertical del observador, respectivamente. En este caso, el 'ojo' estaría en el (0,0,0). Pero generalmente, el observador puede estar en cualquier posición, mirando en cualquier dirección, como ocurre en el caso de una cámara de vigilancia. En esta situación, la transformación de vista consiste en cambiar el sistema de coordenadas global de toda la escena a otro sistema centrado en el observador. Una vez que hayamos trasladado el sistema de referencia, el siguiente paso será realizar la proyección perspectiva en el nuevo sistema y obtener la foto final.

Con el fin de pasar del sistema de coordenadas global al sistema del observador, se realizan los siguientes pasos:

1. *Trasladar el observador al origen de coordenadas*. Para ello se aplica una traslación sobre los ejes X,Y y Z. Considerando (t_x, t_y, t_z) como la posición de la cámara respecto al origen, la matriz de traslación es:

$$T = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -t_x & -t_y & -t_z & 1 \end{pmatrix}$$

2. *Rotar la vista del observador hacia el eje Z*. Al estar en un sistema tridimensional, hay tres posibles ángulos que se podrían aplicar sobre el observador. Un ángulo θ sobre el eje Y, un ángulo ϕ sobre el eje X y un ángulo φ sobre Z. De acuerdo con esto, las matrices de rotación para cada ángulo son:

$$R_\theta = \begin{pmatrix} cos(\theta) & 0 & -sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ sin(\theta) & 0 & cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}; R_\phi = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\phi) & sin(\phi) & 0 \\ 0 & -sin(\phi) & cos(\phi) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix};$$

$$R_\varphi = \begin{pmatrix} cos(\varphi) & sin(\varphi) & 0 & 0 \\ -sin(\varphi) & cos(\varphi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix};$$

La matriz final M de transformación de vista se obtiene tras aplicar los procesos de traslación y rotación de la cámara:

$$M = TR_\theta R_\phi R_\varphi$$

Es importante resaltar que el orden de aplicación de las rotaciones influye en la solución. Nosotros hemos optado por el orden más intuitivo para un usuario: primero aplicamos la rotación sobre el eje Y, después la rotación sobre X y finalmente sobre Z.

Muchos autores, emplean también una matriz para representar la escala en el cálculo de M. En este estudio la hemos obviado, ya que la escala se puede definir mediante la distancia focal 'd'.

Resumiendo, cuando el ojo del observador no está en el origen, hay que aplicar un proceso de traslación y rotación para trasladarlo al origen y obtener así los nuevos puntos proyectados.

$$Q_n = QM$$

Una vez que tenemos las nuevas coordenadas de los puntos en el mundo, podemos aplicar las fórmulas que nos permiten obtener las coordenadas 2D de cada punto.

$$Q'_n = (x'_n, y'_n, z'_n) = (d\frac{x_n}{z_n}, d\frac{y_n}{z_n}, d)$$

Siendo el punto en 2D $Q'_n = (x'_n, y'_n) = (d\frac{x_n}{z_n}, d\frac{y_n}{z_n})$

Modelo matemático para deshacer la transformación y obtener las coordenadas 3D a partir de coordenadas 2D

El proceso más importante para poder realizar un seguimiento 3D es obtener las coordenadas 3D del objeto en el mundo, a partir

de las coordenadas 2D del objeto en la imagen de la cámara. Nuestro procedimiento consiste en deshacer lo que la cámara realiza y obtener la posición en la que está el objeto en realidad. A priori, sabemos que no es fácil, ya que se pierde una componente en la fotografía (la componente z). Como veremos, lo que podemos obtener es una recta como solución. Sin embargo, si además de los parámetros de la cámara, conocemos el suelo donde se proyectó el punto, sabremos determinar qué punto de la recta coincide con el plano.

El problema, ahora, sería el siguiente: Teniendo las coordenadas 2D de un punto $P = (x, y)$ en la fotografía ¿cuáles son sus coordenadas en el mundo real?

Además, sabemos que la fotografía está tomada a una distancia focal 'd'. Por lo tanto, sabemos que la coordenada z del punto es la distancia focal d. Todos los posibles puntos que pueden ser proyectados en esa misma coordenada (x,y,d), vienen representados por:

$$P = (x, y, d) = (2x, 2y, 2d) = ... = (\mu x, \mu y, \mu d)$$

donde μ, es un valor cualquiera.

Las coordenadas homogéneas se introducen con el fin de tratar de una manera uniforme las transformaciones y como anticipo de las transformaciones producidas por la perspectiva en los modelos tridimensionales. Por ello, agregamos una cuarta coordenada. Por ejemplo, en cualquier punto p(x,y,z) sería p(x,y,z,w) en coordenadas homogéneas, donde w tiene un valor arbitrario y representa un factor de escala. Finalmente, todos estos puntos $(\mu x, \mu y, \mu d)$ quedan representados como coordenadas homogéneas con la siguiente ecuación:

$$P = (x, y, d, \alpha)$$

donde α sería $\frac{w}{\mu}$

En el apartado anterior vimos que si el observador no cumplía las condiciones descritas, había que aplicar un proceso de traslaciones y rotaciones para situarlo en el origen y con vista hacia el eje Z, obteniendo

así los puntos proyectados:

$$Q_n = QM = QTR_\theta R_\phi R_\varphi$$

Ahora, tenemos que deshacer dichas transformaciones para regresar el ojo del observador a su sitio original y deshacernos de las proyecciones para hallar los puntos del mundo real.

$$Q = Q_n M^{-1} = Q_n {R_\varphi}^{-1} {R_\phi}^{-1} {R_\theta}^{-1} T^{-1}$$

Primero desharemos las rotaciones. Por ejemplo, para deshacer la primera rotación R_φ tenemos que multiplicar el punto por la matriz inversa de R_φ. No hay que efectuar un proceso de diagonalización, etc.. para obtener la inversa, sino que sabemos que la inversa consiste en aplicar el ángulo en negativo. Conociendo que

$$sin(-\varphi) = -sin(\varphi) y cos(-\varphi) = cos(\varphi)$$

obtenemos:

$${R_\varphi}^{-1} = \begin{pmatrix} cos(\varphi) & -sin(\varphi) & 0 & 0 \\ sin(\varphi) & cos(\varphi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

De forma análoga, podemos calcular las matrices inversas del resto de rotaciones:

$${R_\phi}^{-1} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\phi) & -sin(\phi) & 0 \\ 0 & sin(\phi) & cos(\phi) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$${R_\theta}^{-1} = \begin{pmatrix} cos(\theta) & 0 & sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -sin(\theta) & 0 & cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Como las matrices de rotación tienen que multiplicarse de forma inversa, el nuevo punto sin rotaciones será:

$$P_r(x0, y0, z0, \alpha) = PR_\varphi{}^{-1}R_\phi{}^{-1}R_\theta{}^{-1}$$

Obteniendo

$$\begin{aligned}\mathbf{x0} = x(cos(\varphi)cos(\theta) - sin(\varphi)sin(\phi)sin(\theta))\\ +y(-(sin(\varphi)cos(\theta) + cos(\varphi)sin(\phi)sin(\theta))\\ +z(-cos(\phi)sin(\theta))\end{aligned}$$

$$\begin{aligned}\mathbf{y0} = x(-sin(\varphi)cos(\phi))\\ +y(cos(\varphi)cos(\phi))\\ +z(sin(\phi))\end{aligned}$$

$$\begin{aligned}\mathbf{z0} = x(cos(\varphi)sin(\theta) + sin(\varphi)sin(\phi)cos(\theta))\\ +y(-sin(\varphi)sin(\theta) - cos(\varphi)sin(\phi)cos(\theta))\\ +z(cos(\phi)cos(\theta))\end{aligned}$$

Como el lector puede observar, los nuevos valores de (x, y, z), en este caso (x0, y0, z0), no están afectados por α.

Finalmente, calculamos la inversa de la matriz de traslación T. En un proceso parecido al cálculo de la inversa para las rotaciones, es evidente que:

$$T(-tx, -ty, -tz)^{-1} = T(tx, ty, tz) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ t_x & t_y & t_z & 1 \end{pmatrix}$$

El nuevo punto, sin rotaciones ni traslaciones es:

$$P_{r,t} = (x0 + \alpha \cdot tx, y0 + \alpha \cdot ty, z0 + \alpha \cdot tz, \alpha)$$

Para obtener las coordenadas no homogéneas, normalizamos el

punto, obteniendo P como punto normalizado,

$$P = P_{r,t} = (tx + \frac{x0}{\alpha}, ty + \frac{y0}{\alpha}, tz + \frac{z0}{\alpha}, 1)$$

Como indicamos, el punto real está en función de α, tiene infinitas soluciones. Para dar un valor concreto, usaremos un plano que represente el suelo donde fue tomada la imagen. Todo plano queda definido con un vector $\overrightarrow{N} = (Nx, Ny, Nz)$ y un punto S=(Sx,Sy,Sz). Bajo estos dos parámetros, la ecuación paramétrica que representa al plano es:

$$N = xN_x + yN_y + zN_z - |N \cdot S| = 0$$

Si igualamos la ecuación del plano N y del Punto P, podemos despejar α. Este valor representa el punto común del plano N y de la recta representada en $P_{r,t}$, es decir, la intersección.

$$\alpha = \frac{(Nx \cdot x0 + Ny \cdot y0 + Nz \cdot z0 - |N \cdot S|)}{(Nx \cdot tx + Ny \cdot ty + Nz \cdot tz)} = \frac{(|N \cdot P| - |N \cdot S|)}{(N \cdot T)}$$

Al sustituir α en P, obtenemos finalmente el punto P en 3D:

$$P(x, y, z) =$$
$$(\frac{(x0 \cdot |N \cdot T|)}{(|N \cdot P| - |N \cdot S|)} + tx, \frac{(y0 \cdot |N \cdot T|)}{(|N \cdot P| - |N \cdot S|)} + ty, \frac{(z0 \cdot |N \cdot T|)}{(|N \cdot P| - |N \cdot S|)} + tz)$$

Posición de los objetos en la imagen (Posición 2D)

En el proceso de detección de objetos descrito en [136] y sobre el que nos basamos aquí, cuando un objeto es detectado se crea un elipse que lo envuelve y lo identifica. El centro de dicha elipse es el punto que usan los autores para identificar la posición 2D del objeto. Nosotros haremos una pequeña modificación, cambiando las posiciones de los objetos con el fin de obtener posiciones más apropiadas para su estudio (ver figura 4.7):

- En el caso de que el objeto sea una persona, puede ser evidente que su posición está en el suelo. Por ello, tomaremos el punto de la elipse que obtenemos tras la intersección de ésta con la vertical.

- Si se trata de un vehículo, no podemos tomar el punto de la elipse que esté en el suelo, ya que no siempre sería lo más adecuado al desconocer la estructura del vehículo.... Por ello, tomaremos como posición 2D del objeto el punto intersección de la recta que indica la trayectoria de movimiento del vehículo con la elipse.

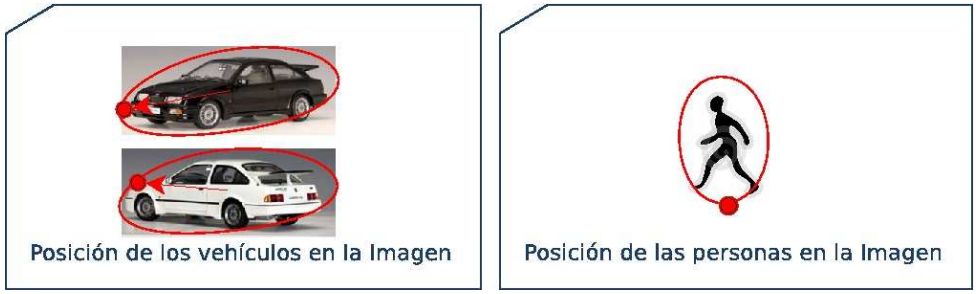


Figura 4.7: Posición escogida para los objetos en la imagen

Posición de los objetos en el mundo (Posición 3D)

Para conocer la posición real de los objetos en el mundo a partir de una imagen, se aplica el modelo matemático (basado en la calibración de la cámara) que fue definido anteriormente en el apartado 4.1.5. Este método permite obtener la posición 3D a partir de las coordenadas de la posición 2D. Para ello, es necesario conocer (con respecto a los ejes de coordenadas que tomemos de referencia): la localización de la cámara que toma la imagen, los ángulos de inclinación sobre los distintos ejes y la distancia focal. Todo esto son parámetros que se pueden medir, por ejemplo, usando instrumentos topográficos.

Para comprobar el comportamiento de la técnica propuesta consideramos un escenario de un cruce en una calle. Este escenario dispone de una cámara de vigilancia que capta los acontecimientos acaecidos

sobre dicho cruce. Construimos una aplicación que permite realizar la calibración de la cámara y el ajuste del plano que indica el suelo. Esta aplicación posibilita pulsar sobre la imagen de la cámara y obtener el punto 3D asociado al punto 2D seleccionado (ver figura 4.8). Como prueba, medimos sobre la realidad el paso de peatones existente en el escenario y comprobamos que los resultados obtenidos mediante la aplicación eran muy aproximados a la realidad.

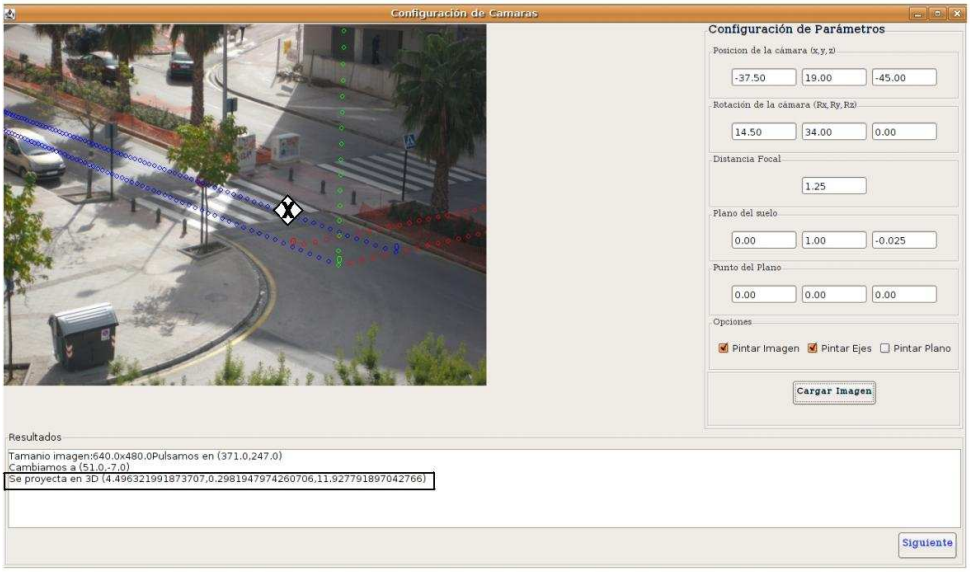


Figura 4.8: Aplicación que muestra el cálculo del posicionamiento 3D

Cálculo del tamaño real de los objetos

Calcular la altura real exacta de un objeto, a partir de sus dimensiones 2D en una imagen, es un proceso complejo, ya que existe una problemática: se desconoce la inclinación del objeto en el mundo (ver figura 4.9). Este hecho limita la posibilidad de conocer la altura exacta.

Nosotros, para resolver este problema, supondremos la verticalidad del objeto, es decir, partiremos de la idea de que el objeto en el mundo real carece de inclinación. Esta hipótesis junto con el

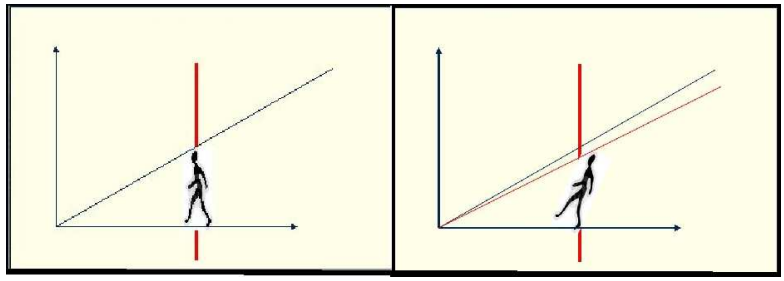


Figura 4.9: Verticalidad vs Inclinación

conocimiento de la distancia focal (d_1), la altura del objeto en 2D (h_{2D}) y la distancia que existe entre la posición real del objeto y la cámara $(d_2 = Distancia(Posicindelacmaraenelmundo, posicindelobjetoenelmundo))$, nos permite aplicar la siguiente *regla de tres* (ver figura 4.10):

$$h_{3D} = h_{2D}\frac{d_2}{d_1}$$

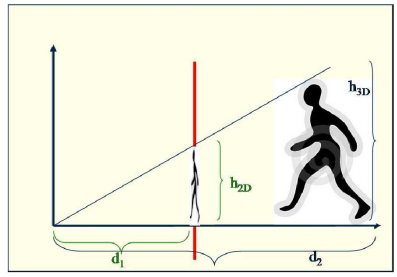


Figura 4.10: Cálculo de la altura real aproximada de un Objeto

La altura que obtenemos, es una altura "aproximada". Este método depende del tamaño 2D detectado.

Cálculo de la velocidad real de los objetos en la escena

La velocidad 3D se calcula de forma similar a la velocidad 2D simplemente que, en este caso, las posiciones de referencia son las

posiciones reales o posiciones 3D de los objetos en el escenario. Como podemos obtener las diferentes posiciones del objeto en el mundo a lo largo del tiempo, podemos hallar su velocidad.

No tomaremos como resultado final el valor de velocidad tal cual $v_2 = \frac{(P_2 - P_1)}{(t_2 - t_1)}$, sino que definiremos la velocidad actual en función de la anterior, ponderándola con un valor ω, con el fin de disminuir los posibles errores que puede haber en los cálculos locales. De esta forma, la fórmula usada para el cálculo de la velocidad de un objeto en un instante t_2 sería:

$$v'_2 = v'_1\omega + v_2(1 - \omega)$$

Siendo v'_1 la velocidad final ponderada del objeto en el instante anterior t_1.

Cálculo de la posible posición futura de un objeto

Al conocer la velocidad (v_t) de un objeto en el mundo en un instante de tiempo determinado (t), es fácil predecir dónde estará dicho objeto dentro de un intervalo de tiempo Δt. Bastaría con aplicar la siguiente fórmula:

$$P_{\Delta t} = v_t \Delta t$$

Este hecho nos permitirá adelantarnos, por ejemplo, a una posible colisión entre objetos.

4.1.6. Algoritmo de Seguimiento basado en la clasificación y la posición 3D de los objetos

En esta sección se propone un algoritmo de seguimiento de objetos más potente y robusto que complementa el seguimiento 2D descrito en [136].

Como veremos a continuación, es necesario realizar este seguimiento para recoger las situaciones que el seguimiento 2D no puede abarcar. Además, este método gestiona de forma adecuada la información que se recibe del análisis de vídeo, manteniendo la coherencia y evitando

réplicas de objetos dentro de la Base de Conocimiento de un escenario. Este algoritmo es flexible a la oclusión y a espacios con varias cámaras.

En [136], los autores realizan un seguimiento básico. Su propuesta otorga un identificador diferente en cada detección de objetos. Esto supone que un objeto puede llegar a tener varios identificadores según el frame en el que se encuentre. Entre sus resultados, para cada objeto detectado en un <*frame i*>, los autores dan como salida del algoritmo de tracking una lista de identificadores de objetos en el <*frame i-1*> de esa cámara que se corresponden con objetos de los que procede el objeto seguido. Para designar dichos objetos, introduciremos el término "antecesor".

Definimos un *antecesor* de un objeto, como el objeto del cual se ha derivado en el proceso de seguimiento. El seguimiento 2D inicial del que partimos, descrito en [136], puede detectar uno o varios antecesores para un objeto. Esto quiere decir que, en un frame, un objeto puede ser la fusión de diferentes objetos que aparecían en el frame anterior (situación 1) (ver figura 4.11). O la situación contraria, que varios objetos en un frame se han originado a partir de un mismo objeto del frame antecedente (situación 2) (ver figura 4.12).

Figura 4.11: Fusión de objetos en el proceso de Detección

La situación ideal sería que un único objeto enlazase como antecesor a un único objeto del frame anterior. Sin embargo, esta situación no es realista, ya que el algoritmo de detección de objetos no siempre ofrece ese resultado, sino que puede ocurrir cualquiera de

Figura 4.12: División de objetos en el proceso de Detección

las dos situaciones citadas anteriormente (fusión de objetos o división de objetos). Por esta razón, para crear un Sistema de Detección de Alertas robusto ante dichas circunstancias, proponemos un algoritmo de seguimiento a alto nivel, basado en la clasificación de los objetos y en el posicionamiento 3D de los mismos.

El algoritmo se aplicará cada vez que el Sistema reciba un evento con la información correspondiente a un conjunto de objetos detectados en un frame de vídeo.

El *objetivo* del algoritmo es identificar el antecesor más exacto para cada objeto del frame de entrada o, desde otro punto de vista, asociar los objetos que son identificados en el frame actual con objetos ya existentes en el Sistema. En el caso de que no exista ninguna posible asociación con un objeto existente, se crean como nuevos Objetos en el Sistema (proceso que se explicará más adelante).

Entradas y salidas del Algoritmo

Las *entradas* del algoritmo son:

- *ObjetosEscena*. Se trata de una lista que contiene todos los objetos detectados en la escena en el frame de vídeo actual. Cada uno de estos objetos se define con un quíntuple (i,a,c,p,t) donde:

- **i** es el identificador *(id)* del objeto,
- **a** representa una secuencia de objetos a los que referencia el objeto de estudio en el frame anterior *(antecesores)*,
- **c** es una lista de cualidades o propiedades de dicho objeto *(cualidades)*,
- **p** es la posición del objeto en el mundo real *(posicion3D)* y,
- **t** es una aproximación del tamaño real *(tam)*.

- *Tiempo*. Es el tiempo de anotación del frame. Es medido en milisegundos. El origen de tiempos considerado es el 01/01/1970.

La *salida* del algoritmo será la actualización correcta de la Base de Conocimiento, actualizando los Objetos existentes o creando nuevos Objetos según las circunstancias analizadas.

Estructuras empleadas

En el algoritmo propuesto, se van a emplear estructuras como una *matriz M* y un *vector de asociación*.

- **Matriz M**: Se trata de una matriz de mxn, donde m es el número de objetos detectados en el frame actual y n el número total que suman los antecesores de los objetos detectados. De esta forma, las filas de la matriz hacen referencia a los objetos de entrada, *ObjetosEscena* y las columnas hacen referencia a los objetos que constituyen una lista de candidatos, *Candidatos*.

  La lista de candidatos está formada por aquellos *Objetos del Sistema* que sean *antecesores* de los 'ObjetosEscena'. Los Objetos del Sistema tienen el formato descrito en Ontología propuesta y se encuentran almacenados en la *Base de Conocimiento*.

  Cada casilla de la matriz, **M(i,j)**, contendrá un valor perteneciente al intervalo [0,1]. Este valor indica el grado de creencia de que el objeto j sea el antecesor del objeto i. De este modo, para una misma fila i, el valor máximo obtenido (M(i,j)) indicará que el elemento j es el mejor candidato para el objeto detectado i.

- **Vector de Asociación**: Las posiciones del vector (i) hacen referencia a los *ObjetosEscena*, mientras que el contenido (j) referencia a los objetos Candidatos. El vector refleja la asociación del elemento *ObjetosEscena(i)* con el *Candidatos(j)*.

Cálculo de los valores de M

La matriz M es inicializada a 0. En el caso de que *Candidatos(j)* no sea un antecesor de *ObjetosEscena(i)*, este valor 0 permanecerá inalterado. Esto nos servirá para descartar la asociación entre ambos elementos. En cambio, si se trata de un antecesor, el valor M(i,j) se actualiza con la siguiente función:

$$M(i,j) = min(w_{dist}, w_{clase})$$

Donde:

- w_{dist} es el grado de cercanía asociado a la distancia existente entre dos posiciones: una, es la posición actual del elemento *ObjetosEscena(i)*, y otra, es la posición donde estaría posiblemente el elemento *Candidatos(j)* en este frame. En este último caso, se utiliza el método que nos permite saber la posible posición futura de un objeto pasado un tiempo (explicada en la sección anterior).

  Una vez hallada la distancia entre ambas posiciones, se calcula el grado de cercanía asociado a dicho dato. Como el término $cerca$ es un concepto difuso, aplicamos una función de pertenencia al conjunto difuso $ceñca$ (ver figura 4.13). De esta forma podemos saber si dos posiciones están “muy cerca”, sólo “cerca” o “poco cerca”.

$$w_{dist} = f(x) = \begin{cases} 1 & \text{si} \quad x \leq \frac{d}{2} \\ \frac{-1}{d}x + \left(\frac{d+\frac{d}{2}}{d}\right) & \text{si} \quad \frac{d}{2} < x < d + \frac{d}{2} \\ 0 & \text{si} \quad x \geq d + \frac{d}{2} \end{cases}$$

Como se puede observar, nos basamos en una distancia d de referencia que indica que dos objetos están cerca con grado 0.5. Con esta función, suponemos que cualquier pareja de posiciones cuya distancia de separación 'x' sea menor a la mitad de la distancia de referencia ($x < 0{,}5d$), están cerca. En cambio, si su distancia supera la suma de la distancia de referencia más la mitad de ésta ($x > d + 0{,}5d$), probablemente no lo estén. En casos intermedios $x \in (0{,}5d..d + 0{,}5d)$, se irá decrementando de forma gradual la certeza de que estén cerca, es decir, aquellos que estén menos separados que la distancia de referencia, estarán más cerca que aquellos que su separación supere dicha distancia.

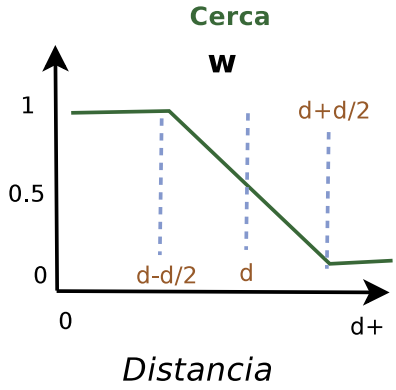


Figura 4.13: Función de pertenencia al conjunto difuso "estar cerca"

- w_{clase} es un valor de creencia asociado con la clasificación del objeto. Para su cálculo, se estudia que las cualidades del *ObjetosEscena(i)*, correspondientes a la clasificación de dicho objeto, estén contenidas en las cualidades del objeto *Candidatos(j)*.

  Recordemos que una cualidad se define por el triple ($clase$,$valor$,$grado\ de\ creencia$), por ejemplo, ("tipo", "persona", 0.8). En este estudio, consideramos que una cualidad está incluida en otra si tienen la misma $clase$ y el mismo $valor$. El grado de creencia asociado a la pertenencia de una cualidad en otra, es el mínimo de los dos grados de creencia que presenta dicha cualidad. Por ejemplo, la cualidad q1("tipo", "persona", 0.8) está incluida en la cuali-

dad q2(“tipo”, “persona”, 0.9) con un grado de creencia 0.8.

De este modo, un conjunto de cualidades A está incluido en un conjunto de cualidades B, si todas las cualidades de A están presentes en las cualidades de B. El grado de pertenencia de un conjunto en otro se mide como el mínimo de los grados de pertenencia de una cualidad en otra. Por ejemplo, siendo A el conjunto (Q1a(“tipo”, “persona”, 0.8), Q2a(“tipo”, “desconocido”, 0.2)) y B el conjunto (Q1b(“tipo”, "persona", 0.7), Q2b(“tipo”, “vehículo”, 0.1), Q3b(“tipo”, “desconocido”, 0.2)) podemos decir que A está contenido en B con un grado de creencia 0.2, pero no al revés (B $\not\subseteq$ A).

Por consiguiente, el grado de creencia w_{clase} es el grado de creencia con el que el conjunto de las cualidades del *ObjetosEscena(i)*, correspondientes a la clasificación, están contenidas en el conjunto de las cualidades del objeto *Candidatos(j)*.

En el caso en el que las cualidades referentes a la clasificación del objeto *ObjetosEscena(i)* no estén contenidas en las cualidades de clasificación de *Candidatos(j)*, w_{clase} es 0. Este dato nos servirá para descartar aquellos antecesores que no compartan la misma clasificación.

Finalmente, el valor correspondiente a un elemento M(i,j) es el valor mínimo de los valores w_{clase} y w_{dist}. Lo que se pretende es minimizar aquellos parejas de objetos donde un *Candidatos(j)* no es buen antecesor de *ObjetosEscena(i)*, bien porque no son objetos $cercanos$ o bien porque no comparten las misma clasificación. Además, se persigue maximizar aquellos, que siendo antecesores, tengan las mismas cualidades y sus posiciones sean cercanas (ya que puede tratarse del mismo objeto). De esta forma, el máximo de la fila indicará que el *Candidatos(j)* es el antecesor más apropiado para *ObjetosEscena(i)*.

Algoritmo

Entradas: time, ObjetosEscena.
Parámetros: Candidatos, M[][], VAsociación[]

BEGIN

1. **Construir la lista de Candidatos.**
   FOR $\forall$ *ObjetosEscena(i)* $\{i = 0..Num(ObjetosEscena)\}$

   a) Se obtiene los *Objetos del Sistema* a los que hacen referencia sus antecesores

   b) FOR cada $\forall$ antecesores

   1) Si no está en $Candidatos$, se añade.

   END FOR

2. **Crear M[ObjetosEscena.size][Candidatos.size]**

3. **Se rellena la matriz M**
   FOR cada M(i,j) $\{i = 0..ObjetosEscena.size, j = 0..Candidatos.size\}$

   a) IF $Candidatos(j)$ no es antecesor de $ObjetosEscena(i)$,

   $$M(i,j) = 0$$

   ENDIF

   b) ELSE IF $Candidatos(j)$ es antecesor de $ObjetosEscena(i)$,

   1) $dist$=Distancia($ObjetosEscena(i).posicion3D$, $Candidatos(j).posicin3DFutura(frame.tiempo)$)
   2) w_{dist}=Grado_Cercania($dist$)
   3) w_{clase}=Grado_Pertenencia($ObjetosEscena(i).cualidadesClasificacion()$, $Candidatos(j).cualidadesClasificacion()$)
   4) M(i,j)=min(w_{dist},w_{clase})

   ENDIF

   END FOR

4. **Crear Vector de Asociación** $VAsociacion[ObjetosEscena.size]$

5. **Se rellena el Vector de Asociación**
   DO(

   a) Se obtiene el MÁXIMO valor de M y su posición, ($maxValor$, i_{max},j_{max}).

   b) IF maxValor!=0

      1) Se asigna el $Candidatos(j_{max})$ como antecesor apropiado de $ObjetosEscena(i_{max})$ $VAsociacion[i_{max}] = j_{max}$
      2) Se actualiza el objeto del Sistema $Candidatos(j_{max})$ con la nueva información detectada del $ObjetosEscena(i_{max}$
      3) Se actualiza toda la fila i_{max} a 0

      ENDIF

)WHILE (true)

6. **Añadimos al Sistema los objetos que no tienen ningún asociado o antecesor** FOR $i = 0..ObjetosEscena.size$

   a) IF $VAsocicacion[i] = null$

   $$CrearnuevoObjetosEscena(i)$$

      ENDIF

   ENDFOR

END

La creación de un nuevo objeto

No siempre ha de crearse un nuevo Objeto en el Sistema. Existen situaciones que suponen una excepción. Podría darse el caso de que el objeto no tenga antecesores o que ninguno de los que tiene cumpla las propiedades necesarias para que sea considerado como su antecesor. Sin embargo, este hecho no quita que en otros frames anteriores sí esté realmente su antecesor y que, durante un tiempo, el objeto haya sido atrapado por otros (debido al problema de fusión de objetos en el algoritmo de detección, o a oclusiones, etc.).

Por este motivo, antes de crearlo en el Sistema, se comprueba si existe un posible Objeto en la Base de Conocimiento con el que se pudiera asociar. Para realizar esta comprobación, se calcula la posición donde estarían cada uno de los objetos del Sistema, en el tiempo en el que tiene lugar el frame de estudio. Si hay alguno que estuviera lo suficientemente cerca del objeto que va a ser creado como nuevo y, además, que pueda llevar uno o varios instantes sin actualizarse, asumimos que ese objeto podría ser el objeto que estamos analizando actualmente, por lo que lo asociamos con él. Sólo en el caso de que no encontremos un posible antecesor, crearemos un objeto nuevo en el Sistema.

Ilustrémoslo con un ejemplo (ver figura 4.14). Supongamos que aún no existe ningún objeto en el Sistema:

1. En el primer frame de una secuencia, se detectan dos objetos, uno parece ser una persona y otro un vehículo. Se crean ambos objetos en el Sistema, concretamente en la Base de Conocimiento.

2. En el segundo frame, se detecta un sólo objeto, que parece ser un coche, y que apunta a los dos objetos anteriores como antecesores (fusión de objetos).

   Podría darse el caso de que el objeto persona haya salido de la escena, o bien que haya subido al vehículo, o bien que esté muy cerca de él y por error se haya detectado como un sólo objeto, o bien que la persona haya quedado oculta tras el vehículo.

   Lo más evidente, y lo que hace nuestro algoritmo, es asociar el nuevo objeto vehículo con el objeto vehículo anterior, y actualiza su posición, velocidad... en el Sistema. De esta forma, el objeto persona se mantiene inalterado, ya que ante cualquiera de las situaciones que hayan podido pasar, no tenemos conocimiento exacto acerca de ella.

3. En el tercer frame, se detecta, de nuevo, un objeto vehículo que apunta al anterior. En este caso el seguimiento es trivial, se trata del mismo vehículo. Se actualiza el objeto del Sistema que lo representa.

4. En un cuarto frame, se detectan dos objetos, uno que parece ser una persona y otro un vehículo. Ambos objetos tienen como antecesor al objeto vehículo del frame anterior (división del objeto).

   En este caso, también es evidente que probablemente el nuevo vehículo siga siendo el mismo que el anterior. Nuestro algoritmo realiza dicha asociación. De esta forma, la persona se crearía como nuevo objeto. Pero en este último paso, como se ha explicado, hacemos una comprobación para saber si alguno del resto de objetos que están en la Base de Conocimiento podría ser dicha persona.

   Se obtiene que hay un objeto persona en el Sistema y que no se ha actualizado recientemente. Consecuentemente, comprobamos dónde estaría esa persona en el tiempo actual (tiempo del frame 4) si mantuviese la velocidad, con el fin de saber si podría tratarse de la misma persona. En el caso de que pudiese ser factible, porque ambos objetos estuviesen aproximadamente en la misma posición, actualizaríamos dicho objeto sin crear uno nuevo. De esta forma estamos realizando un seguimiento a un nivel de abstracción más alto.

Comportamiento del algoritmo frente múltiples cámaras

¿Cómo se comportaría el algoritmo ante el procesamiento de información procedente de varias cámaras? Si hacemos un estudio, vemos que para una determinada cámara *'A'*, llamaremos al algoritmo cuando llegue información de cada frame y se irán creando y actualizando los objetos en el Sistema. En el caso de haber otra cámara *'B'*, se hará lo mismo. Si estas cámaras graban escenas totalmente diferentes, cada instancia del algoritmo irá actualizando Objetos distintos en la Base de Conocimiento. Y así para múltiples cámaras.

La pregunta siguiente sería ¿Y si las cámaras A y B graban la misma escena desde diferentes puntos de vista? En este caso, deben tener como requisito la misma calibración, es decir, que ambas usen como referencia el mismo eje de coordenadas.

Supongamos que ambas cámaras graban una plaza con una fuente

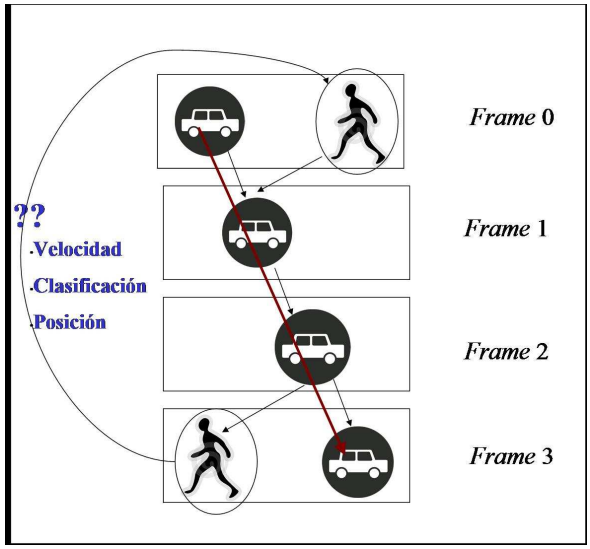


Figura 4.14: Ejemplo de una instancia del Algoritmo de Seguimiento propuesto

central. Si colocamos el eje de coordenadas en dicha fuente, debemos comprobar que para las distintas cámaras la fuente está en el (0,0,0). Una vez que se cumplan estas condiciones, al algoritmo no le afecta que haya múltiples cámaras actualizando los mismos objetos. En algún momento un objeto aparecerá en la escena, se podrá ver en una sola cámara o en las dos (dependiendo de la vista de las cámaras). En ambas situaciones, el objeto se creará en el Sistema originado por la información de una sola cámara, la que haya enviado antes el evento.

Recordamos que la creación y actualización de objetos se hace en exclusión mutua para evitar errores en dicho proceso. Cuando la otra cámara lo vaya a crear, primero se comprueba si existe otro objeto en el Sistema que pudiese ser dicho Objeto. Como trabajamos con coordenadas 3D en el mundo, si ambas cámaras están bien calibradas, el algoritmo lo reconoce y lo asocia al Objeto del Sistema que fue creado anteriormente, por lo que se realizará una actualización evitando una duplicación de objetos.

Reflexiones

Era necesario construir nuestro propio proceso de seguimiento para resolver determinadas situaciones.

- **Situación del 'tipo 1'**, cuando un *objeto en un frame surge por la fusión de 2 o más objetos del frame anterior*.

  Por ejemplo, ante un evento en el que un objeto-vehículo surge tras la fusión de dos objetos, persona y coche, del frame anterior, bien porque se ha subido la persona al coche o porque están tan cerca que solo se detecta un objeto, el más grande, ¿a qué objeto le actualizamos los nuevos datos como la posición, velocidad... a la persona, al coche?

  O en el caso de que un objeto-vehículo surge tras la fusión de tres objetos-persona: ¿Se trata de un vehículo que tapa a las personas desde la perspectiva de la cámara, o es que las personas caminan muy cerca? ¿Actualizamos a la vez los tres objetos que representan a las personas como si fuesen un grupo de gente y no creamos un nuevo objeto coche, o realmente ha aparecido un coche en la escena?

- **Situación del 'tipo 2'**, es la situación contraria a la anterior, cuando *varios objetos en un frame surge por la división de un objeto en el frame antecedente*.

  Este podría ser el caso de tres objetos clasificados principalmente como personas que surgen de un objeto-vehículo existente en el frame anterior, ¿Era un grupo de personas o un coche? ¿Creamos en nuestro Sistema tres nuevos objetos? . . .

Si se dan las condiciones ideales de que un objeto tenga un único antecesor, el seguimiento del objeto es trivial, nuestro algoritmo actualiza los nuevos datos sobre dicho objeto en el Sistema. En las dos situaciones puntuales de estudio, el método propuesto tiene en cuenta la clasificación y las posiciones de los objetos.

Resumiendo, en la primera situación, cuando un objeto tiene varios antecesores (fusión de objetos), el algoritmo propuesto descarta a los candidatos que no compartan la misma clasificación del objeto de estudio

o bien que se encuentren en posiciones no cercanas, lo que implicaría que no se trate del mismo objeto. Y se ofrece como solución la selección del candidato más apropiado (que es aquel que comparte la misma clasificación y se encuentra en posiciones muy cercanas en el mismo frame).

Este resultado, permite que los datos del objeto actual se actualicen sobre el objeto del Sistema que haya sido elegido como antecesor, mientras que los demás objetos permanecen inalterados. Por ejemplo, sería la situación de que un vehículo y una persona se fusionen, porque haya subido al coche o éste la ocluya. En este caso, actualizamos al objeto coche dejando el objeto persona sin modificar.

Si ninguno de los objetos anteriores comparte la misma clasificación, no se realiza ninguna asociación, sino que se crea un nuevo objeto en el Sistema (siempre que no exista otro en la Base de Conocimiento que pueda ser, y que no apareciese en el frame anterior). Este podría ser el caso en el que dos personas suben a un coche, o bien, son tapadas por el vehículo.

En la segunda situación, cuando varios objetos tienen el mismo antecesor, el algoritmo asociará el seguimiento del objeto original únicamente a uno de ellos, creando los demás como nuevos objetos del Sistema (si se comprueba que no existen otros objetos en la Base de Conocimiento que pudiesen serlo). La asociación se realiza siguiendo el mismo proceso descrito anteriormente.

Por ejemplo, si un coche en un frame se divide en dos personas en el frame posterior, se crearían dos nuevos objetos personas en el Sistema, dejando el objeto-coche sin modificar. En cambio, en el caso de que el objeto-coche se divida en coche y persona, se enlaza su seguimiento con el coche del segundo frame y se crea tan solo un objeto-persona nuevo. Recordamos que los objetos se crearán nuevos sólo y cuándo se haya comprobado que ningún objeto del Sistema pudiese ser él.

El algoritmo nos permite realizar un seguimiento a un mayor nivel de abstracción, resolviendo las situaciones de fusión y división de objetos en la detección. Se consigue llevar a cabo el proceso adecuado de gestión y mantenimiento de los Objetos con los que nuestro Sistema operaría,

manteniendo coherencia en la Base de Conocimiento.

4.1.7. Módulo de detección de la alerta peligro de atropello

En esta sección vamos a describir el Modelo Experto que proponemos para dar solución al problema planteado: identificar con antelación una posible colisión entre dos objetos cualesquiera. Posteriormente veremos su aplicación al caso concreto vehículo-peatón, ya que el fin principal del Sistema es detectar el peligro por la predicción de posibles atropellos a peatones.

A priori, nuestra primera propuesta para llevar a cabo la detección de posibles colisiones, fue estudiar por igual el comportamiento de los objetos móviles involucrados en una colisión. Sin embargo, después de varios tests, vimos que en algunas situaciones es necesario diferenciar diferentes roles entre los objetos que pueden dar lugar a una colisión. Por ello, en este estudio, distinguiremos entre objetos vulnerables y amenazadores.

Principalmente, nos basaremos en la idea de que un 'objeto-A' colisiona con un 'objeto-B'. En este caso, denominamos al 'objeto-A' como *objeto amenazador*, ya que consideramos que es el objeto que causa la colisión, y al 'objeto-B' como *objeto vulnerable*, porque suponemos que está en peligro por colisión. Es necesaria esta división de roles porque hay situaciones en las que un solo objeto es peligroso, como el choque entre un vehículo y un peatón.

Al estudiar colisiones donde un objeto amenazador choca contra un objeto vulnerable, nuestra propuesta podrá diferenciar entre colisiones relevantes y no relevantes. Una colisión es relevante cuando un objeto vulnerable es golpeado por un objeto amenazador y no será relevante en caso contrario.

Por ejemplo, si nos centramos en el caso de colisiones vehículo-peatón, o dicho de otro modo, en el caso del atropello de peatones, el vehículo es el objeto amenazador y la persona el vulnerable. Si un vehículo colisiona con una persona, las repercusiones pueden ser

graves, nos encontramos ante una colisión de relevancia. Sin embargo, si una persona choca contra un coche estacionado, se trata de una colisión no relevante que no debe ser motivo de alarma y, por lo tanto, nuestro Sistema no lo detectaría, ya que el objetivo es predecir colisiones causadas por objetos amenazadores.

Existen otras situaciones donde los dos objetos que ocasionan una colisión son peligrosos (o amenazadores), como cuando dos coches colisionan en una intersección. En estos casos, las situaciones se resolverán por simetría: nuestro modelo, primero, evaluará la situación tratando a un objeto como objeto amenazador y al otro como vulnerable y, posteriormente, se evaluará la situación opuesta. De esta forma, en estos casos, los dos serán tratados tanto como objetos vulnerables como amenazadores.

Sistema experto difuso para la detección de futuras colisiones

El análisis de la detección de futuras colisiones en el tiempo se basa en el estudio de conceptos difusos. Se evaluará ¿Cómo de *rápido* se mueven los objetos involucrados? ¿Cómo de *cerca* se encuentran los objetos? ¿Queda *poco tiempo* para la colisión?...

Para llevar a cabo un razonamiento con incertidumbre, presentamos un Modelo experto basado en lógica difusa, concretamente un **sistema basado en reglas difusas**.

Muchos de los sistemas expertos basados en reglas difusas, también denominados *controladores difusos*, diferencian varios módulos (ver figura 4.15):

- *Módulo Fuzzificador*, cuya función es tomar los valores numéricos provenientes del exterior y convertirlos en valores "difusos" que pueden ser procesados por el mecanismo de inferencia. Estos valores difusos son los niveles de pertenencia de los valores de entrada a los diferentes conjuntos difusos en los cuales se ha dividido el universo de discurso de las diferentes variables de entrada al Sistema.

- El *Marco de Conocimiento*. En él, se encuentran los conocimientos del sistema a desarrollar. Está formado principalmente por una base de datos, la cual contiene todos los datos relativos al sistema, variables y valores difusos posibles, y una base de reglas, que contiene todas las proposiciones que van a regir el sistema. La base de reglas es la manera que tiene el sistema difuso de guardar el conocimiento lingüístico, que le permite resolver el problema para el cual ha sido diseñado.

  Estas reglas son del tipo *IF antecedente, THEN consecuente*.

- El *Motor de Inferencia*. Se encarga de extraer las conclusiones partiendo de los datos analizados, aplicando las reglas que rigen el sistema. Una modificación del conocimiento conllevará una nueva evaluación de las reglas que puede dar como resultado unas conclusiones distintas.

- *Módulo Defuzzificador*. La salida que genera el mecanismo de inferencia es una salida difusa, lo cual significa que no puede ser interpretada por un elemento externo que únicamente manipule información numérica. Para lograr que la salida del sistema difuso pueda ser interpretada por elementos que solo procesen información numérica, hay que convertir la salida difusa del mecanismo de inferencia en una salida no difusa; este proceso lo realiza el defuzzificador.

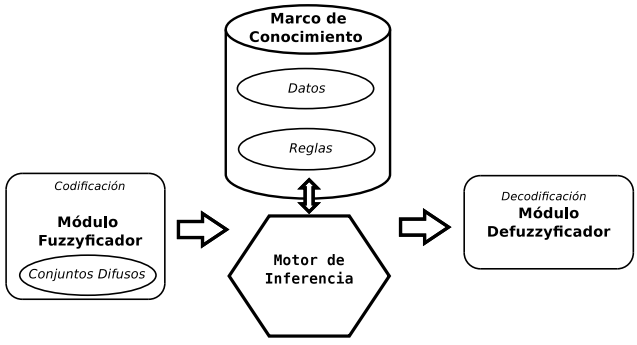


Figura 4.15: Estructura general de un Controlador Difuso

La estructura general del sistema difuso que aquí presentamos queda de la siguiente forma:

- Fuzzificador: El Sistema de vigilancia que proponemos en esta sección, recibe información en la que existen datos crips y datos con cierto grado de incertidumbre. Por lo tanto, ya disponemos de algunos valores difusos con los que se van a operar y que no necesitan un proceso de fuzzificación. Este es el caso de la clasificación de los objetos detectados a partir del vídeo. Recordemos que un objeto puede ser clasificado como *persona*, *vehículo* u *otro*, con un grado de posibilidad que indica el nivel de certeza con el que ha sido identificado como ese tipo de objeto.

  Sólo existirá un proceso de fuzzificación para los datos de entrada crisp que necesiten ser convertidos a datos difusos para ser procesados por el motor de inferencia del sistema y así poder operar con las condiciones de las reglas.

  Como veremos a continuación, en este controlador, el proceso de fuzzificación no se realiza directamente a partir de los datos de entrada. En primer lugar, los datos de entrada son procesados para obtener nuevo conocimiento requerido en el Sistema. De forma concreta, nos referimos a: la velocidad real de los objetos, la distancia entre dos posiciones (el punto de colisión y el objeto vulnerable), y, el tiempo que queda para que dos objetos puedan colisionar en un futuro.

  Una vez halladas dichas variables, tendrá lugar el proceso de fuzzificación para obtener los correspondientes datos difusos, es decir, los valores de pertenencia a los distintos conjuntos difusos estudiados: 'corto espacio de tiempo', 'cerca' y 'rápido'. Para una mejor comprensión para el lector, estos conjuntos han sido definidos en el motor de inferencia a la hora de realizar los cálculos.

- El ***Marco de Conocimiento***: que contiene todos los datos necesarios para el Motor de Inferencia. En este módulo se distinguimos entre: 1) la información que se va a analizar que, en este caso, coincide con el conjunto de objetos de la *Base de Conocimiento (BC)* del

Sistema de Vigilancia, 2) *Base de Reglas*: que contiene todas las reglas difusas que van a regir el Sistema.

- El ***Motor de Inferencia***: que se encarga de extraer las conclusiones, partiendo de los datos analizados, aplicando las reglas que rigen el Sistema. Una modificación del conocimiento conllevará una nueva evaluación de las reglas que puede dar como resultado unas conclusiones distintas.

El proceso de un defuzzificador no será necesario, ya que el objetivo del sistema difuso es obtener un nivel de presencia de peligro que indique el grado de creencia obtenido, sobre que pueda existir un posible atropello. Este dato de salida estará incluido dentro del intervalo [0,1].

Base de Reglas

El Modelo que proponemos se basa en el análisis de tres aspectos principales:

1. *Si falta poco tiempo para que el 'objeto amenazador' colisione con el 'objeto vulnerable'*.
2. *La cercanía o grado de proximidad entre el 'objeto vulnerable' y el punto de colisión de éste con el 'objeto amenazador'*.
3. *La rapidez a la que se mueve el 'objeto amenazador' en la escena*.

Actualmente, el Sistema está compuesto de una sola regla, ya que, como veremos más adelante, ésta es suficiente para la predicción de colisiones. Hemos de enfatizar que, antes de aplicar la regla, se ha realizado un procesamiento complejo, basado en un modelo geométrico y matemático, que permite obtener las posiciones de los objetos en el mundo real, sus velocidades, la actualización coherente de dichos objetos en la Base de Conocimiento (gracias al algoritmo de seguimiento propuesto y al plugin Eliminador de Objetos), el cálculo de la posición donde puede tener lugar una colisión, el tiempo que falta para que dicho choque ocurra... Estos procedimientos previos facilitan que la detección de riesgo de colisión mediante la regla difusa que proponemos.

Esta regla analiza los tres aspectos anteriores para detectar el peligro de colisión entre un 'objeto vulnerable' y un 'objeto amenazador'. Ésta puede ser definida, informalmente, como:

Si existe un 'objeto amenazador' que puede colisionar con un 'objeto vulnerable' en un 'corto espacio de tiempo' (w_0)*, y ese 'objeto vulnerable' está 'cerca' del punto de colisión entre ambos objetos* (w_1) *y el 'objeto amenazador' corre a 'gran velocidad'* (w_2)*, entonces, existe riesgo de que el 'objeto amenazador' colisione con el 'objeto vulnerable' (y ese riesgo está en función de los conceptos estudiados en las condiciones de la regla).*

De este modo, presentamos una **nueva variante de sistema experto difuso del tipo Sugeno**. El modelo de Takagi-Sugeno [143] emplea reglas difusas que presentan la siguiente estructura:

R_i : If x_1 is A_{i1} and ... and x_n is A_{in} Then y_i = F(x_1,...,x_n),

donde las x_i son las variables de entrada del sistema y F normalmente es una función lineal sobre dichas variables. La salida de un SBRD con una base de conocimiento con m reglas Takagi-Sugeno es la media ponderada de cada una de las reglas individuales, y_i , con i = 1,..., m:

$$Y = \frac{\sum_{i=1}^{m} h_i \cdot y_i}{\sum_{i=1}^{m} h_i}$$

siendo h_i = T($A_{i1}(x_1)$,..., $A_{in}(x_n)$) el grado de emparejamiento entre el antecedente de la i-ésima regla y la entrada actual al sistema (x_1,...,x_n), y siendo T una T-norma.

En la regla que rige nuestro sistema, el consecuente está en función de la entrada que tiene el Sistema en un momento dado. Sin embargo, nuestra y_i no está en función directa de las variables de entrada, pero sí en función de los grados de creencia sobre la pertenencia de las variables de entrada a los conceptos difusos estudiados. De este modo, los datos de entrada influyen en los datos obtenidos en las condiciones y, consecuentemente, en la salida. Por consiguiente, **la salida de la regla está en función del grado de cumplimiento del antecedente**. La variante de Sugeno que proponemos sigue el siguiente esquema para las reglas:

$IF\ x_0 is A_{i0}\ and ... and\ x_n is A_{in}\ THEN\ y_i = F(w_0, ..., w_n)$,

donde:

- w_0 es el grado de cumplimiento de $x_0\ is\ A_{i0}$,
- ...
- w_n es el grado de cumplimiento de $x_n\ is\ A_{in}$,
- y y_i está en función del grado de cumplimiento del antecedente: F(w_0,...,w_n) = $\alpha \cdot$ AND(w_0,...,w_n)=$\alpha \cdot (w_0 \wedge_{min} ... \wedge_{min} w_n)$,

Concretamente, la regla concreta que constituye el sistema actual se puede representar como:

$IF\ x_0\ is\ A_0\ AND\ x_1\ is\ A_1\ AND\ x_2\ is\ A_2\ THEN\ y = F(w_0, w_1, w_2)$,

donde:

- x_0 es el tiempo que falta para la colisión,
- A_0 representa el concepto difuso 'corto espacio de tiempo',
- x_1 es la distancia entre el 'objeto vulnerable' y el punto de colisión,
- A_1 representa el concepto difuso 'cerca',
- x_2 es la velocidad del 'objeto amenazador',
- A_2 representa el concepto difuso 'rápido'.
- w_0 es el grado de cumplimiento de que el tiempo que falta para la posible colisión sea corto,
- w_1 es el grado de cumplimiento de que la distancia entre los objetos estudiados indique cercanía,
- w_2 es el grado de cumplimiento de que la velocidad del objeto amenazador sea rápida,

- $F(w_0, w_1, w_2) = \alpha \cdot \text{AND}(w_0,w_1,w_2) = w_0 \wedge_{min} w_1 \wedge_{min} w_2$, donde $\alpha = 1$ porque al solo tener un tipo de regla en el sistema, ésta será la única que afecte al nivel de alerta final. Si hubiese otras reglas, tendría más sentido que cada una tenga un $\alpha \in [0, 1]$ que indique un nivel de importancia distinto para cada regla.

Motor de Inferencia

El motor de inferencia evalúa la regla para obtener un nivel de riesgo sobre una posible colisión entre los objetos evaluados. Esta regla es evaluada con cada pareja de objetos 'objeto amenazador'-'objeto vulnerable' que estén presentes en la escena en el momento actual.

Como se ha dicho anteriormente, la salida de la regla está en función del grado de cumplimiento del antecedente. Para evaluar cada una de las condiciones que constituyen dicho antecedente es necesario calcular las variables de estudio del controlador difuso. Por ejemplo, con el fin de conocer si queda poco tiempo para la colisión, primero se calcula el valor numérico de dicho tiempo y, posteriormente, se fuzzifica dicho valor aplicando la función de pertenencia al conjunto difuso '$\tilde{corto\ espacio\ de\ tiempo}$'.

A continuación vamos a explicar los cálculos que se realizan en el motor de inferencia para conocer las tres variables de estudio:

1. *El tiempo estimado que falta para que el 'objeto amenazador' colisione con el 'objeto vulnerable',* $t_{colision}$.

   En esta parte el objetivo es calcular cuánto tiempo falta para que el 'objeto amenazador' choque con el 'objeto vulnerable' y se ocasione una colisión. Este tiempo es denominado $t_{colision}$. Para obtener este dato, es necesario conocer el punto donde tendrá lugar la colisión *(P_c: Punto de Colisión)*. Este punto es definido como el punto intersección entre el plano de movimiento del 'objeto vulnerable' *(P_m)* y la recta de dirección del 'objeto amenazador' (R_a).

   Para obtener el P_m, conocemos que en términos geométricos, un plano es definido por un punto contenido en él y un vector perpendicular al mismo (vector normal al plano). El vector perpendicular

al Plano de Movimiento P_m del 'objeto vulnerable' es hallado como el producto vectorial entre el vector perpendicular al suelo de la escena *($\vec{P_s}$: Plano del suelo)* y el vector de dirección del 'objeto vulnerable', *($\vec{V_v}$: Velocidad del 'objeto vulnerable')*.

$$\vec{P_m} = (\vec{P_s} \times \vec{V_v})$$

Este vector, junto con la posición del 'objeto vulnerable' P_v (punto contenido en P_m), nos permite calcular la ecuación general de P_m:

$$\vec{P_m} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + |P_v \cdot \vec{P_m}| = 0$$

Por otro lado, podemos estimar cuáles van a ser las posiciones futuras del 'objeto amenazador' en función de la variable tiempo t, obteniendo la recta de dirección del 'objeto amenazador':

$$R_a = P_a + V_a \cdot t$$

Una vez que conocemos P_m y R_a, podemos hallar su intersección y obtener el punto P_c donde tendrá lugar la colisión,

$$P_c = R_a \bigcap P_m$$

También se conoce que el punto de colisión es la posición donde estará el objeto amenazador dentro del tiempo $t_{colision}$:

$$P_c = P_a + V_a \cdot t_{colision}$$

Si sustituimos el P_c en la ecuación general de P_m y despejamos $t_{colision}$, obtenemos:

$$t_{colision} = \frac{|P_v \cdot P_m| - |P_a \cdot P_m|}{V_a - P_m}$$

donde $t_{colision}$ es el tiempo que tarda el 'objeto amenazador' en

llegar a la intersección con el plano de movimiento ($\vec{P_m}$), a partir de su posición actual P_a con su velocidad actual V_a.

La representación geográfica de este método es representada en la figura 4.16.

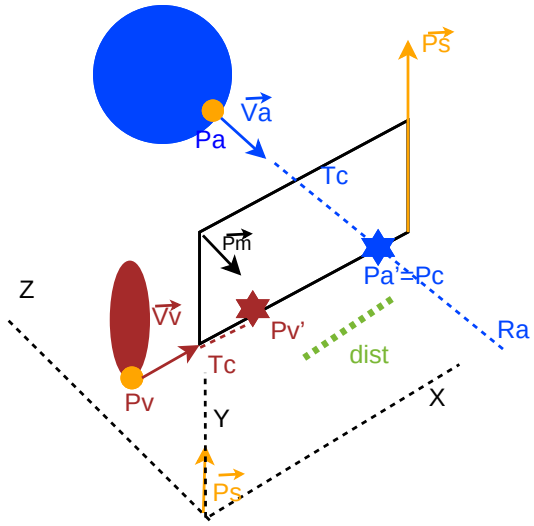


Figura 4.16: Geometría del Proceso

2. *La distancia existente entre el 'objeto vulnerable' y el punto de colisión de éste con el 'objeto amenazador',* $dist$.

   Una vez que se conoce el tiempo que queda para que tenga lugar la colisión $t_{colision}$, podemos determinar la futura posición dónde estará el 'objeto vulnerable' dentro de este tiempo:

$$P'_v = P_v + (V_v \cdot t_{colision})$$

   donde P_v y V_v son la posición y la velocidad del 'objeto vulnerable', respectivamente.

   Posteriormente, calculamos la distancia '$dist$' entre el 'objeto vulnerable' y el punto de colisión Pc, también hallado anteriormente:

$$P_c = P_a + V_a \cdot t_{colision}$$

$$dist = \sqrt{(P_v'^2 - P_c^2)}$$

3. *La velocidad a la que se mueve el 'objeto amenazador' en la escena,* V_a.

   Otro factor importante en el estudio de la predicción de colisiones es la velocidad. Por ejemplo, cuando se detecta una alta probabilidad de que dos objetos puedan colisionar (porque van a llegar a estar muy cerca), es importante conocer si uno de ellos se detiene poco antes de la posible colisión. También es muy importante la velocidad del 'objeto amenazador' V_a, ya que es más probable que una colisión ocurra, si este objeto va a gran velocidad. Este dato puede ser consultado desde la Base de Conocimiento.

Una vez conocidas las tres variables $t_{colision}$, $dist$ y V_a, para una pareja 'objeto amenazador'-'objeto vulnerable' tiene lugar el proceso de *fuzzificación* de las mismas, para obtener los valores de pertenencia (w_0, w_1 y w_2) a los conceptos difusos que se estudian en las condiciones del antecedente de la regla. w_0, w_1 y w_2 son valores pertenecientes al intervalo $[0, 1]$.

- **Evaluación de la primera condición. Obtención de** w_0.

  Una vez que se conoce el $t_{colision}$, se aplica una función trapezoidal para obtener el grado de pertenencia w_0 del $t_{colision}$ con respecto al conjunto difuso *'corto espacio de tiempo'*, (ver figura 4.17).

  Se usa una función trapezoidal porque es el mejor evaluador para la referencia humana en este caso. Esto es debido a que, primero, cuando el 'objeto amenazador' sobrepasa el plano de movimiento del 'objeto vulnerable', el tiempo $t_{colision}$ llega a ser negativo, por lo que no es relevante. El grado de pertenencia de este tiempo negativo al concepto *'corto espacio de tiempo'* debe ser 0. Segundo, existe un intervalo de tiempo crítico donde el grado de pertenencia debe ser 1. Este intervalo se da cuando queda muy poco tiempo para la colisión. Tercero, cuando el $t_{colision}$ es alto, el grado de pertenencia debe ser 0 otra vez, ya que el hecho de

que quede mucho tiempo para la colisión no es tampoco relevante. Entre el primer y el segundo caso, y entre el segundo y el tercero, consideramos que el cambio debe ser gradual. Todos estos motivos nos llevan a que la función trapezoidal sea el mejor evaluador en este caso, ya que se ajusta muy bien al razonamiento humano.

Esta función trapezoidal es definida como:

$$f(x) = \begin{cases} 0 \quad \text{si} \quad x \leq lA \\ \frac{1}{lB-lA}x + \left(1 - \frac{lB}{lB-lA}\right) \quad \text{si} \quad lA < x < lB \\ 1 \quad \text{si} \quad lB \leq x \leq lC \\ \frac{-1}{lD-lC}x + \left(\frac{lD}{lD-lC}\right) \quad \text{si} \quad lC < x < lD \\ 0 \quad \text{si} \quad x \geq lD \end{cases}$$

donde lA, lB, lC y lD son números reales que cumplen la siguiente relación lA <lB <lC <lD. La representación gráfica de la función la podemos ver en la figura 4.17.

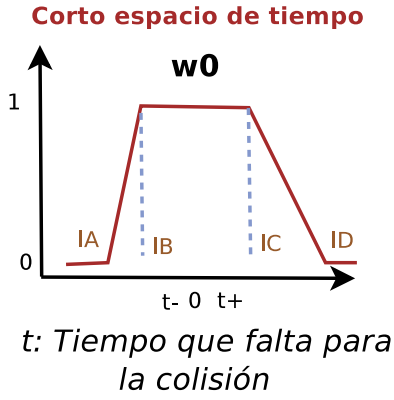


Figura 4.17: Función de pertenencia al conjunto difuso “corto espacio de tiempo”

- **Evaluación de la segunda condición. Obtención de** w_1.

  Dentro del contexto en el que estamos, no es adecuado decir que dos posiciones están cerca y en el instante siguiente decir que no lo

están. La cercanía es un concepto que varía de forma gradual. No se debe discretizar porque perdemos información. "Estar cerca" es un concepto borroso. De esta forma podemos saber si dos posiciones están "muy cerca", sólo "cerca" o "poco cerca". Por esta razón, aplicamos una función de pertenencia (ver figura 4.18) que nos permite saber cómo de cerca están dos posiciones y nos devuelve un grado de creencia sobre dicha cercanía. Nos basamos en una distancia d de referencia que indica que dos objetos están cerca con grado 0.5:

$$f(x) = \begin{cases} 1 & \text{si} \quad x \leq \frac{d}{2} \\ \frac{-1}{d}x + \left(\frac{d+\frac{d}{2}}{d}\right) & \text{si} \quad \frac{d}{2} < x < d + \frac{d}{2} \\ 0 & \text{si} \quad x \geq d + \frac{d}{2} \end{cases}$$

Con esta función, suponemos que cualquier pareja de posiciones cuya distancia de separación 'x' sea menor a la mitad de la distancia de referencia ($x < 0{,}5d$), están cerca. En cambio, si su distancia supera la suma de la distancia de referencia más la mitad de ésta ($x > d + 0{,}5d$), probablemente no lo estén. En casos intermedios $x \in (0{,}5d \;..\; d + 0{,}5d)$, se irá decrementando de forma gradual la certeza de que estén cerca, es decir, aquellos que estén menos separados que la distancia de referencia estarán más cerca que aquellos que su separación supere la distancia de referencia.

Tras conocer el parámetro $dist$, se aplica esta función que indica el grado de pertenencia w_1 de $dist$ al conjunto difuso 'cerca', (ver figura 4.18).

- **Evaluación de la tercera condición. Obtención de w_2.**

  El concepto de ir a gran velocidad o ir rápido también es un concepto borroso. Proponemos otra función que indica el grado de pertenencia de una velocidad 'x' con respecto a este concepto difuso 'rápido'. Esta función es definida como:

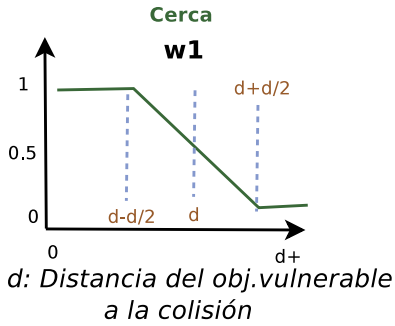


Figura 4.18: Función de pertenencia al conjunto difuso “estar cerca”

$$f(x) = \begin{cases} 0 \quad \text{si} \quad x \leq lInf \\ \frac{1}{lSup - lInf} x + \left(1 - \frac{lSup}{lSup - lInf}\right) \quad \text{si} \quad lInf < x < lSup \\ 1 \quad \text{si} \quad x \geq lSup \end{cases}$$

donde $lInf$ y $lSup$ son dos números reales que indican los límites inferior y superior de la función, respectivamente. Un objeto que se mueve a una velocidad inferior a $lInf$ no se considera que sea rápido. Sin embargo, si su velocidad fuese mayor que $lSup$, sí se considera que se mueve rápido o a gran velocidad. Con una velocidad intermedia, el objeto se moverá rápido con un grado de certeza, el que indique la función.

Su representación gráfica puede verse en la figura 4.19.

Esta función es aplicada para obtener la importancia w_2 de cómo de rápido va el ‘objeto amenazador’ tras conocer su velocidad V_a.

- **Evaluación del consecuente.**.

  Una vez evaluadas de forma independiente las tres condiciones de la regla, se evalúa el antecedente completo, lo que conlleva aplicar el operador AND difuso sobre los resultados obtenidos al evaluar las distintas condiciones (w_0, w_1 y w_2).

  Usamos la función mínimo como operador AND porque pretendemos que la existencia de las tres condiciones influyan en

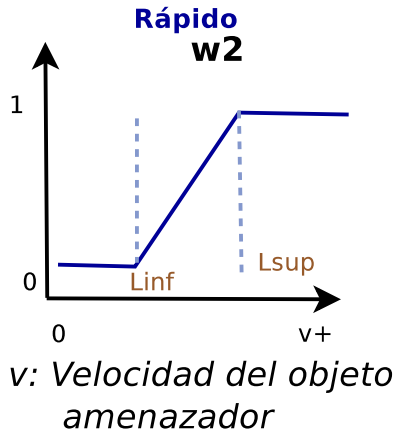


Figura 4.19: Función de pertenencia al conjunto difuso “rápido”

el nivel de peligro detectado para la pareja ‘objeto amenazador’-‘objeto vulnerable’ analizada. De este modo, obtenemos el grado de cumplimiento de la regla para un determinado par de objetos:

$$wR = AND(w_0, w_1, w_2) = w_0 \wedge w_1 \wedge w_2 = min(w_0, w_1, w_2)$$

$$wR \in [0, 1]$$

El resultado que obtenemos es el nivel de posibilidad de que exista una colisión futura entre los dos objetos estudiados. Este nivel es un valor de grado que pertenece al intervalo [0,1].

Como podemos ver, al aplicar el $minimo$ es necesario que se cumplan las tres condiciones (w_0, w_1, w_2) para que exista riesgo de colisión. El hecho de que una de ellas no se cumpla implica que el grado de predicción de colisión sea cero.

- **Cálculo del nivel de alerta**.

  Como se señaló anteriormente, la regla que constituye el controlador difuso es evaluada para cada pareja de objetos ‘objeto amenazador’-‘objeto vulnerable’ existente en la escena en el momento actual. El modelo propuesto se basa en la idea de que existe peligro de colisión si **existe** al menos una pareja de objetos donde

el objeto vulnerable esté en peligro porque el objeto amenazador pueda chocar con él. De este modo, el nivel de alerta final será:

$$Nivel\ de\ Colision = wR_{p0}\ OR\ wR_{p1}\ OR \cdots OR\ wR_{pN}$$

donde N es el número de parejas 'objeto amenazador'-'objeto vulnerable' existente en la escena en el momento actual y wR_{pi} es el grado de cumplimiento de la regla tras ser evaluada con una pareja i. Como operador OR, empleamos la función *máximo* para obtener el mayor nivel de peligro detectado.

El resultado que obtenemos, tras aplicar las pautas de este modelo, es un valor de grado que pertenece al intervalo [0,1] y que indica el máximo nivel de riesgo detectado sobre la posible existencia de una futura colisión entre dos objetos del escenario de estudio.

Los parámetros de referencia de las funciones de pertenencia a los conceptos difusos estudiados serán parámetros de configuración del modelo propuesto. Nos referimos a las variables: lA, lB, lC y lD para el concepto *'corto espacio de tiempo'*; d para el concepto *'cerca'*; y $lInf$ y $lSup$ para el conjunto difuso *'rápido'*. Estos parámetros podrán ser ajustados en función del escenario de estudio donde se aplicará el Sistema.

Aplicación al caso de la colisión vehículo-peatón

El Modelo Abstracto presentado es fácilmente adaptable a cualquier tipo de colisión. En este caso, se desea identificar la presencia de riesgo de que un peatón pueda ser atropellado por un vehículo. El objetivo es detectar, con adelanto, una colisión vehículo-peatón, para así predecir un posible atropello y evitar que ocurra un hecho desagradable.

En esta situación, el peatón es el 'objeto vulnerable' y el vehículo el 'objeto amenazador'. Si particularizamos el Modelo Abstracto para la detección de colisiones a este caso concreto, tendremos en cuenta los siguientes factores:

1. El tiempo que falta para que el vehículo colisione con la persona.

2. La distancia que falta para que la persona llegue al punto donde puede tener lugar la colisión.

3. La velocidad a la que va el vehículo.

De esta forma, la regla difusa descrita en el modelo general pasa a definirse de la siguiente forma concreta:

Si existe un vehículo que puede colisionar con un peatón en un 'corto espacio de tiempo' (w_0)*, y ese peatón está 'cerca' del punto de colisión entre ambos objetos* (w_1) *y el vehículo corre a 'gran velocidad'* (w_2)*, entonces, existe riesgo de que el vehículo colisione con el peatón.*

Se ha diseñado un algoritmo como motor de inferencia para la detección del peligro de atropello, basado en el modelo experto para detección de futuras colisiones. Ha sido diseñado bajo el paradigma de la programación imperativa, donde se describen un conjunto de instrucciones que le indican al computador cómo detectar la situación de alerta, basándonos siempre en el sistema de reglas definido. No se ha usado una programación declarativa porque ésta sería menos eficiente cuando hay muchos objetos que evaluar, y nosotros pretendemos que la situación de alerta sea detectada en tiempo real, con adelanto, y de la forma más eficiente posible.

Este algoritmo puede verse en el Algoritmo 1 [1].

El estado de la situación estudiada, en este caso, el peligro de atropello, se refleja mediante un objeto *'alerta'* definido en la ontología de salida del Sistema (ver sección 3.5). La alerta almacena siempre el nivel de presencia detectado. Pero, también, es importante destacar que sería de gran valor guardar la información referente sobre cualquier cambio

[1] $objetosBC$ es el conjunto de objetos existentes en la BC; lA, lB, lC y lD son los limites de referencia de la función trapezoidal que indica la pertenencia al conjunto difuso "poco espacio de tiempo"; $distRef$ es la distancia de referencia de la función de pertenencia de otra distancia 'x' al concepto difuso "cerca"; $lRefVelInf$, $lRefVelSup$ son los valores de referencia para la función de pertenencia al conjunto difuso 'rápido'. Otros: $Nivel\_de\_Alerta$ es el nivel de presencia de peligro una posible colisión futura; $w0$ es el grado de cumplimiento de la 1ª condición de la regla. $w1$ es el grado de cumplimiento de la 2ª condición; $w2$ es el grado de cumplimiento de la 3ª condición

Algorithm 1 peligroAtropello($objetosBC$, lA, lB, lC, lD, $distRef$, $lRefVelInf$, $lRefVelInf$)

Require: $objetosBC, lA, lB, lC, lD, distRef, lRefVelInf, lRefVelInf$
  $Nivel\_de\_Alerta$ = 0
  $evaluados$ = $\{\phi\}$
  for $\forall\ obj\_1 \in objetosBC$ **do**
    $evaluados$ = $evaluados \cup obj\_1$
    for $\forall\ obj\_2 \in objetosBC - evaluados$ **do**
      if $esPersona(obj\_1)$ && $esVehiculo(obj\_2)$ **then**
        $obj\_persona$ = obj_1
        $obj\_vehiculo$ = obj_2
      else
        if $esPersona(obj\_2)$ && $esVehiculo(obj\_1)$ **then**
          $obj\_persona$ = obj_2
          $obj\_vehiculo$ = obj_1
        else
          continue
        end if
      end if
      $w0 = 0$; $w1 = 0$; $w2 = 0$;
      $t_{colision} = calcularTiempoDeColision()$
      $dist = Distancia(posicionFutura(obj\_persona, t_{colision}), PuntoColision)$
      $V_a = velocidad(obj\_vehiculo)$
      $w0$=gradoPertenenciaCORTOESPACIOTIEMPO($t_{colision}$, lA, lB, lC, lD)
      $w1$=gradoPertenenciaCERCANIA($dist$,$distRef$)
      $w2$=gradoPertenenciaRAPIDO(V_a,$lRefVelInf$, $lRefVelInf$)
      wR=AND_{min}($w0$,$w1$,$w2$)
      $Nivel\_de\_Alerta$=OR_{max}($Nivel\_de\_Alerta$,wR)
    end for
  end for

que exista en la situación de estudio. De esta forma, mientras se realiza la evaluación y el análisis de la situación, se recoge información sobre los objetos implicados en la actualización del nivel de alerta, tanto en modo texto (porque se conoce la semántica del problema) como en formato de lista de Objetos de la BC que estén implicados.

Como se puede observar, si en un mismo escenario existe más de una situación que indique un posible peligro de atropello, es decir, hay más de un peatón en peligro por accidente, el Sistema indica el máximo nivel de alerta detectado para las distintas situaciones. Es importante destacar, que, en este caso, aunque se recoja el máximo nivel de alerta detectado, correspondiente a una única situación (pareja vehículo-peatón con mayor riesgo de accidente), en la explicación se recoge toda la información de los demás objetos que están en peligro por accidente.

4.1.8. Plugins

Proponemos 3 plugins en este Sistema. Uno de ellos es el plugin básico que debe tener toda aplicación, según el Modelo propuesto en el capítulo 3, el *Eliminador de Objetos*.

Eliminador de Objetos (Plugin EO)

La función de este componente es eliminar aquellos Objetos que ya no formen parte del escenario porque hayan salido de la escena. Se basa en la idea de que si un objeto no ha sido actualizado durante un tiempo t, entonces se debe de decrementar el grado de creencia que indica la vigencia del objeto con una cantidad c, hasta llegar a cero. Lo que indicará que se debe eliminar. Su funcionamiento completo ha sido descrito en la sección 3.4.2.

Este plugin es importante para mantener la coherencia en la Base de Conocimiento y evitar la identificación de falsas alertas. Para su buen funcionamiento, es necesario configurarlo en función del escenario de estudio. Como parámetros de configuración tenemos:

- t, es el tiempo máximo que el Eliminador de Objetos espera, ante un objeto que no es actualizado, para empezar a disminuir el grado de

creencia que indica la vigencia del objeto en la escena. Este tiempo es medido en milisegundos.

- c, es un valor entre 0 y 1. Este valor indica la cantidad a restar al grado de creencia que indica la vigencia del objeto en la escena.

Visualización de la Base de Conocimiento (Plugin VBC)

La función principal de este plugin será recoger toda la información residente en la Base de Conocimiento en el momento actual y representarla de forma legible por una persona, en formato texto. Lo que se persigue con este plugin es tener una ventana que dé acceso a la información conocida de cada objeto de la escena. Este componente será útil cuando se desarrolle una herramienta software para la monitorización del Sistema. De este modo, el usuario podrá acceder, en todo momento, a los objetos vigentes en el escenario y conocer todos sus datos.

Identificación de Objetos Rápidos (Plugin IOR)

Este componente se encargará de clasificar los objetos del escenario como *rápidos*, con un índice de certeza. Una vez conocida la velocidad del objeto en la Base de Conocimiento, este plugin podrá conocer el grado de pertenencia de esta velocidad al concepto *rápido*. Para ello, se empleará la función mostrada en la figura 4.19 y definida anteriormente en la sección 4.1.7 como:

$$f(x) = \begin{cases} 0 \quad \text{si} \quad x \leq lInf \\ \frac{1}{lSup - lInf} x + \left(1 - \frac{lSup}{lSup - lInf}\right) \quad \text{si} \quad lInf < x < lSup \\ 1 \quad \text{si} \quad x \geq lSup \end{cases}$$

donde $lInf$ y $lSup$ son dos números reales que indican los límites inferior y superior de la función, respectivamente. Un objeto que se mueve a una velocidad inferior a $lInf$, no se considera que sea rápido. Sin embargo, si su velocidad fuese mayor que $lSup$, sí se considera que se

mueve rápido o a gran velocidad. Con una velocidad intermedia, el objeto se moverá rápido con un grado de certeza, el que indique la función.

Evidentemente, es importante diferenciar qué es rápido para un vehículo y qué es rápido para un peatón. Por lo que este plugin emplea esta función con parámetros diferentes para $lInf$ y $lSup$, en función de que un objeto sea clasificado como vehículo o como persona. Para obtener una buena clasificación sobre la rapidez de un objeto, es necesario configurar y ajustar estos parámetros en función del escenario de estudio.

Los datos obtenidos en este plugin actualizarán la Base de Conocimiento. Este dato será representado como una nueva cualidad *(c,v,g)* del Objeto según la ORHC (sección 3.5) donde: (c,v,g), donde:

- **c** es "velocidad".
- **v** "rápido".
- **g** es un grado de creencia entre 0 y 1, que indica cómo de rápido va el objeto en función de su velocidad dentro de la escena.

Esta información puede ser usada para obtener el grado de cumplimiento de la tercera condición de la regla del modelo propuesto para la detección de futuras colisiones (ver sección 4.1.8).

4.1.9. Desarrollo del sistema

El Sistema de Vigilancia Inteligente, propuesto en esta sección (4.1) para el análisis de la presencia de posibles atropellos, se ha implementado usando como lenguaje de programación *JAVA*.

El middleware empleado es *ICE ZeroC* [58], ya que es orientado a objetos y se puede usar en entornos heterogéneos, donde los clientes y los servidores pueden escribirse en diferentes lenguajes de programación y pueden ejecutarse en distintos sistemas operativos y en distintas arquitecturas. Este aspecto es fundamental para el Sistema, ya que uno de nuestros objetivos es la reutilización de otros Sistemas de análisis

previos sobre fuentes de información y, por otro lado, la reutilización de los datos generados en el Sistema actual para otros fines.

Se han desarrollado dos herramientas: una herramienta que permite la adquisición de conocimiento del entorno compuesta por el Traductor y otra herramienta para la monitorización del mismo, que incluye una aplicación para dispositivos móviles.

Herramienta de adquisición de conocimiento sobre el entorno: Traductores

En esta aplicación, desarrollada en JavaSE, se lleva a cabo la capa de *Fusión e Integración de los datos de entrada al Sistema*, acompañada de un pre-procesamiento de la información.

Como se ha mencionado anteriormente, el Sistema recibe vídeo anotado como entrada. Concretamente los resultados de un trabajo descrito en [136], donde se realiza un proceso de detección y tracking-2D de objetos móviles. En esta herramienta, se ha implementado el *Traductor* para dicha fuente de información (el cual se ha descrito en la sección 4.1.4). Este traductor se encarga de representar la información de entrada bajo el esquema definido en la Ontología ORHC (sección 3.5) e integrarla en la Base de Conocimiento del Sistema, creando nuevos objetos o actualizando los anteriores. Además, este componente añade nuevo conocimiento sobre las velocidades y el posicionamiento real de los objetos en el escenario.

La herramienta desarrollada permite definir el escenario de estudio y agregar de forma dinámica componentes *Traductores*. Se agregan tantos Traductores como cámaras de vigilancia existan en el entorno. En cada uno de esos Traductores, se añade la imagen que captura la cámara y se ajustan de forma adecuada los parámetros referentes a la calibración de la cámara. Esta información es necesaria para poder realizar el cálculo del posicionamiento 3D.

En la figura 4.20, podemos ver la aplicación referente a los Traductores. Sobre la imagen asociada a la cámara, el Sistema pinta la información de entrada recibida, en este caso, se pintan las elipses asociadas a los objetos. El color de la elipse indica la clasificación

con creencia máxima asociada al objeto, se usa el color azul para los vehículos, el rojo para las personas y el negro para otros casos. El pequeño punto verde indica la posición 2D del objeto sobre la imagen.

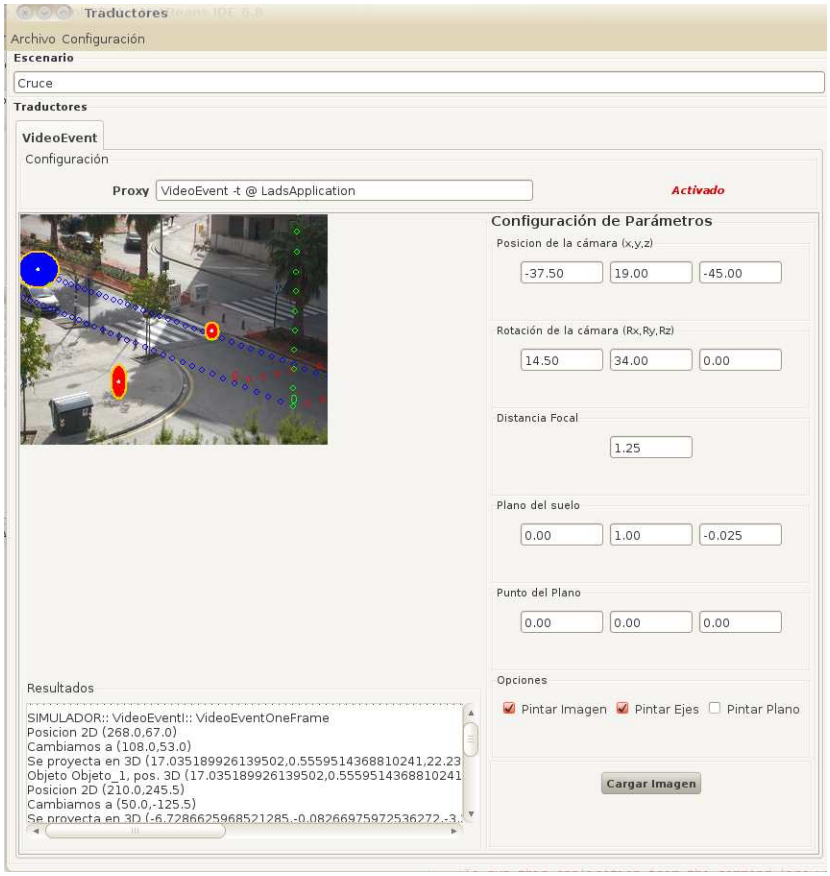


Figura 4.20: Traductor

Herramienta de monitorización de la Alerta Peligro de Atropello

Existe otra herramienta, también implementada bajo JavaSE, que ha sido desarrollada para observar el estado de la situación analizada en el escenario de estudio. En esta herramienta se llevan a cabo todas las tareas descritas en la capa de *Análisis*.

En esta aplicación diferenciamos dos partes, las cuales están separadas en pestañas diferentes de la interfaz de usuario. Por un lado, tenemos la configuración y la visualización de los plugins y, por otro lado está la configuración y visualización de las alertas, en este caso, de la alerta Peligro de Atropello. Esta herramienta permite configurar y ajustar desde la interfaz de usuario aquellos parámetros que dependen del escenario a vigilar, tanto para los plugins como para la alerta.

Siempre que el Sistema esté recibiendo eventos de entrada, la situación de alerta será evaluada obteniendo como resultado el nivel de presencia de una futura colisión entre un objeto-peatón y un objeto-vehículo. Como se ha dicho anteriormente, este nivel de alerta es un valor perteneciente al intervalo [0,1]. Nosotros hemos dividido este intervalo en subintervalos, a los cuales se le ha asignado una etiqueta. Por lo tanto, el nivel de predicción de una colisión es reinterpretado para determinar el nivel de riesgo de peligro de atropello como *nulo*, *bajo*, *medio* o *alto*.

Estos estados están en función del valor *umbral*, que también pertenece al intervalo [0,1]. Además, se ha asociando a cada etiqueta un color (ver figura 4.21):

- *Estado Blanco: riesgo nulo,* cuando el nivel de predicción de colisión es cero. No hay ningún signo que indique riesgo de colisión entre objetos.
- *Estado Verde: riesgo bajo,* cuando el nivel de predicción de colisión pertenece a (0, umbral/2].
- *Estado Amarillo: riesgo medio,* cuando el nivel de predicción de colisión pertenece a (umbral/2, umbral). Cuando el color es amarillo, el Sistema emite una alarma sonora que atrae la atención del operador.
- *Estado Rojo: riesgo alto,* cuando el nivel de predicción de colisión pertenece a [umbral,1], es muy probable que tenga lugar una colisión. En este caso, la alarma sonora emitida es más intensa.

Para que el nivel de alerta pueda ser fácilmente observado e interpretado por un humano, la visualización del estado de las alertas se

ha realizado gráficamente en forma de barras verticales. De esta manera, se puede conocer, en un solo vistazo, el estado de las distintas situaciones estudiadas en el entorno vigilado. El grado de creencia de una alerta vendrá reflejado en la longitud de la barra. Ésta irá cambiando de color en función de su tamaño. Si el nivel de certeza supera el umbral de activación de la alerta, la barra adquirirá el color rojo. Si el nivel es muy bajo, la barra tendrá color verde. Un nivel intermedio se corresponderá con un color amarillo.

A parte de la visualización gráfica, se muestra, en modo texto, los motivos que justifican el valor actual de la alerta. Además, el Sistema guarda un historial donde aparecen todas las veces que la alerta ha superado el umbral y los motivos que han influido en dicha activación.

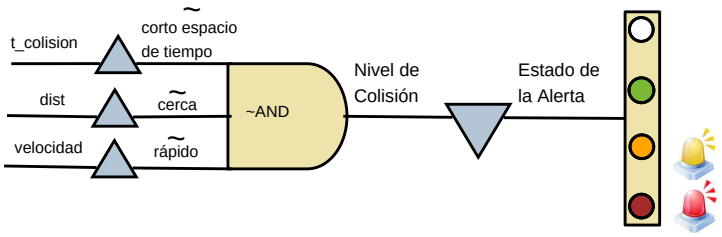


Figura 4.21: Proceso Difuso

En la figura 4.22 puede verse la interfaz de usuario para esta aplicación, concretamente, la pestaña que permite al usuario observar el estado de la situación de estudio que analiza el Sistema.

Como se observa, se trata de un Sistema que puede ser portado fácilmente a cualquier entorno donde se pretenda estudiar la presencia de posibles atropellos. Actualmente, solo existen las aplicaciones descritas en esta sección. Se pretende que estas herramientas den soporte al operador humano en su proceso de vigilancia, avisándole de forma automática de la existencia de un posible riesgo.

Sin embargo, si este Sistema se llevase a la práctica real, el objetivo sería alertar, en tiempo real, a los objetos implicados en la situación estudiada. Para ello, sería necesario llamar su atención de forma visual o sonora en escenario donde tiene lugar el riesgo de atropello detectado,

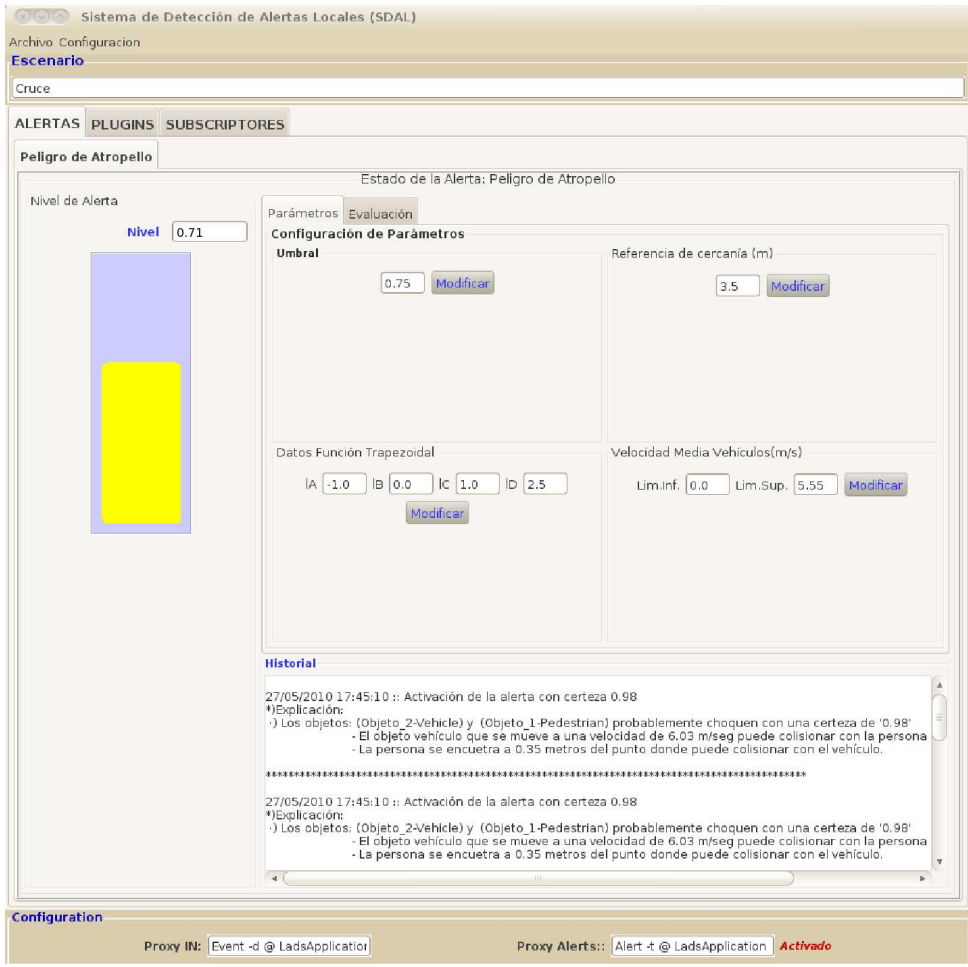


Figura 4.22: Aplicación de Escritorio: Sistema de Vigilancia para la detección de Peligro de Atropello

con el fin de evitar un hecho desagradable.

Aplicación móvil

Con el objetivo de ofrecer cierta movilidad al usuario del Sistema, se ha desarrollado una aplicación móvil. Esta herramienta ha sido implementada en JavaME con la configuración CLDC (Connected Limited Device Configuration). Puede ser instalada en cualquier teléfono móvil con wifi o incluso hasta en PDAs.

Se trata de una aplicación estándar y sencilla que puede suscribirse a cualquier alerta. En este caso, el cliente de esta aplicación debe suscribirse a la alerta de *Peligro de Atropello* e indicar cada cuánto tiempo quiere ser informado y a partir de qué umbral de alerta.

El Notificador de alertas se ha implementado para ser transparente al tipo de cliente que se suscribe. Se permiten dos tipos de clientes, el cliente normal, a través de middleware ICE ZeroC, y el cliente móvil, que realiza la suscripción a través del middleware pero que recibe los eventos mediante datagramas UDP (ya que no se puede transmitir en UDP con Ice ZeroC para móviles). En el sistema, hay un servidor a la escucha de suscripciones. Cuando se recibe una suscripción móvil, el servidor notifica al cliente del puerto donde se debe escuchar. Y la aplicación móvil leerá los datos desde ese puerto.

Una vez realizada esta suscripción, el usuario recibirá notificaciones sobre el estado de la alerta. Se podrá ver el nivel de alerta de forma gráfica y la explicación de la misma en formato texto. Esta aplicación permite visualizar el estado de las alertas de forma similar a la aplicación de monitorización para un entorno de escritorio (usando una barra gráfica que cambia de color y tamaño en función del nivel de riesgo detectado).

4.1.10. Fase experimental

Un paso muy importante, tras la propuesta de un nuevo modelo y el desarrollo de su aplicación, es llevar a cabo un proceso de evaluación que permita validar dicha propuesta. De este modo, se han realizado dos tipos de experimentos con el fin de evaluar el Sistema de Vigilancia para

la detección del Peligro de Atropello:

1. Un primer experimento enfocado para medir la *fiabilidad del Sistema*. Con este estudio se ha pretendido evaluar cómo de buenas son las respuestas del Sistema ante determinadas circunstancias.
2. Un segundo experimento para evaluar los *tiempos de procesamiento* del Sistema. Este aspecto es de vital importancia ya que se desea que el Sistema sea eficiente y detecte riesgos en tiempo real. Concretamente, en este caso, se desea que el riesgo sea predicho con antelación, ya que el objetivo es evitar un accidente y no detectarlo cuando ya ha pasado.

El Sistema ha sido testado con situaciones reales. Para ello, hemos analizado vídeo capturado de una zona urbana.

El escenario escogido para las pruebas ha sido un cruce de peatones sin semáforos (ver figura 4.23), el cual es muy apropiado porque existe tráfico de vehículos y peatones.

Figura 4.23: Escenario de prueba para evaluar el Sistema de Vigilancia Inteligente para la Detección del Peligro de Atropello

El primer paso para realizar las pruebas, fue calibrar la cámara usada para poder obtener las posiciones 3D de los objetos detectados a

partir del vídeo.

Con el fin de evaluar el Sistema para ambos experimentos, hemos confeccionado un conjunto de ejemplos que cubren diferentes situaciones. Estos ejemplos han sido simulados, ya que es necesario evaluar circunstnacias críticas donde haya riesgo o peligro de que suceda un accidente de peatones y, también, situaciones donde un peatón sea realmente atropellado.

De forma concreta, se han diseñado 24 ejemplos canónicos de situaciones que pueden ocurrir en la vida real para el escenario seleccionado y se han simulado como eventos de vídeo anotado que recibe como entrada el Sistema. Entre estos ejemplos tenemos: 4 situaciones donde está claro que no hay riesgo de accidente, 7 situaciones en las que el riesgo o el peligro no es fácil de detectar, 5 situaciones donde peatones son atropellados por vehículos, 4 situaciones en las que los vehículos se detienen para permitir el cruce de los peatones y 4 situaciones donde los vehículos no paran y los peatones tienen que esperar. En estas simulaciones aparecen desde 2 a 5 objetos.

Los parámetros usados en las funciones de pertenencia a los conjuntos difusos (w0)poco tiempo; w1)cerca; w2)rápido) (ver figura 4.24) para testear el Sistema son:

- Límites usados en la función trapezoidal que muestra la importancia del tiempo que queda para la colisión: lA=-1seg. lB=0seg., lC=1seg., lD=2.5seg.
- Distancia de referencia usada en la función que muestra la importancia de la distancia que existe desde la persona al punto de colisión: d=4 metros como valor 0.5. Para este caso, 6 metros corresponden a un valor 0 y, 2 metros corresponden a un valor 1.
- Límites de la función que muestra la importancia de la velocidad del vehículo: lInf=0km/h, como valor 0, y lSup=20km/h, como valor 1.
- El umbral de alarma elegido es 0.75.
- Los eventos simulados son enviados a 8 frames por segundo.

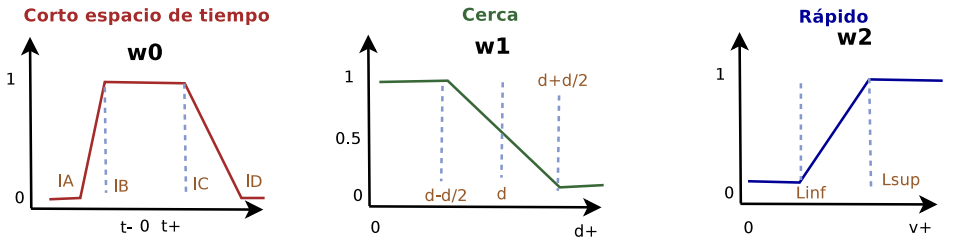


Figura 4.24: Funciones de pertenencia a los conceptos difusos: w0)corto espacio de tiempo; w1)cerca; w2)rápido.

Experimento 1: Estudio de la fiabilidad del Sistema

Este experimento ha sido llevado a cabo en 3 partes:

- Primero, hemos testado el Sistema con los 24 ejemplos, anotando los resultados obtenidos para cada simulación. En este proceso, se ha almacenando el nivel de alerta más alto que se ha obtenido durante el transcurso de cada escena.
- Segundo, las mismas situaciones han sido analizadas por ocho personas, las cuales han anotado sus observaciones. Para ello, les mostramos los vídeos simulados de las distintas situaciones y dichas personas anotaron el máximo grado de presencia de riesgo de colisión peatón-vehículo que percibieron al ver las escenas. Los datos obtenidos en este test son mostrados en la tabla 4.25.
- Tercero, hemos comparado ambos resultados, los del Sistema con los de las personas. Los datos obtenidos pueden verse en la tabla 4.26. Si comparamos estos resultados, podemos ver que encontramos 20 coincidencias, 2 sobre-estimaciones y 2 infra-estimaciones. Estos datos muestran un alto rendimiento del Sistema.

Los dos casos donde el Sistema infraestima la predicción humana no son tan críticos, ya que en el caso en el que el nivel baja de verde a

Studied situations	Collision detection in advance				Average Value	Estate
	Choice carried out by the eigh people					
	White	Green	Yellow	Red		
1	7	1			0,04166667	Green
2	6	2			0,08333333	Green
3	8				0	White
4	1	3	3	1	0,5	Yellow
5			5	3	0,79166667	Red
6		1	7		0,625	Yellow
7	3	3	1		0,20833333	Green
8	4	3	1		0,20833333	Green
9				8	1	Red
10				8	1	Red
11				8	1	Red
12				8	1	Red
13	1	3	3	1	0,5	Yellow
14	1	5	2		0,375	Green
15	1	2	3	2	0,58333333	Yellow
16		1	4	2	0,625	Yellow
17	3	2	2	1	0,375	Yellow
18	3	4	1		0,25	Green
19	6	2			0,08333333	Green
20	2	6			0,25	Green
21	7	1			0,04166667	Green
22			6	2	0,75	Red
23				8	1	Red
24		2	5	1	0,625	Yellow

Figura 4.25: Observaciones humanas

Studied situations	Human	Our System					Real Collision
	Obtained Estate	Obtenited Value	Obtained Estate	coincidences	over-estimated	under-estimated	
1	Green	0,22	Green	1			
2	*Green*	0	*White*			1	
3	White	0	White	1			
4	Yellow	0,64	Yellow	1			
5	Red	0,87	Red	1			
6	Yellow	0,47	Yellow	1			
7	Green	0,14	Green	1			
8	Green	0,72	Yellow		1		
9	Red	1	Red	1			X
10	Red	1	Red	1			X
11	Red	1	Red	1			X
12	*Red*	0,69	*Yellow*			1	X
13	Yellow	0,65	Yellow	1			
14	Green	0,24	Green	1			
15	Yellow	0,54	Yellow	1			
16	Yellow	0,45	Yellow	1			
17	Yellow	0,65	Yellow	1			
18	Green	0,31	Green	1			
19	Green	0,65	Yellow		1		
20	Green	0,34	Green	1			
21	Green	0,06	Green	1			
22	Red	0,75	Red	1			
23	Red	1	Red	1			X
24	Yellow	0,7	Yellow	1			
				20	2	2	

Figura 4.26: Rendimiento del Sistema de Vigilancia Inteligente para la Detección del Peligro de Atropello

blanco, la alarma es silenciosa y en el caso que baja de rojo a amarillo, la alarma auditiva es activada.

A su vez, los dos casos de sobreestimación, donde los verdes son estimados como amarillos, se deben a un ajuste del Sistema, ya que se prefiere evitar situaciones peligrosas, en lugar de hacer caso omiso de las colisiones donde los peatones pueden ser heridos.

Experimento 2: Estudio de los tiempos de procesamiento en Sistema

En este experimento, se han medido los tiempos de procesamiento del Sistema. El equipo utilizado para realizar estas pruebas es un Pentium (R) Dual-Core CPU E5200@2.50GHz 2.51GHz, 2,75 GB de RAM.

Para llevar a cabo este estudio hemos realizado dos tests diferentes. En el primero, hemos medido los tiempos de procesamiento en escenas reales simuladas, usando el conjunto de ejemplos empleado en el experimento anterior. Para ello, hemos distinguido dos partes que implican procesamientos independientes en el Sistema:

- Por un lado, se ha estudiado el tiempo que emplea el Traductor en procesar los datos de entrada. Para ello, se ha medido el tiempo que transcurre desde que el Sistema recibe un evento de entrada hasta que el Traductor actualiza la Base de Conocimiento con dicha información. En esta fase, se realizan diversos procesamientos: cálculo del posicionamiento 3D, obtención de velocidades y aplicación del algoritmo de seguimiento a alto nivel.
- Por otro lado, se ha evaluado el tiempo que el Sistema emplea en evaluar la situación de alerta tras detectar un cambio en la Base de Conocimiento.

Para ambos casos, se han obtenido los tiempos de los 24 ejemplos de situaciones simuladas. En la tabla 4.1 pueden verse los datos estadísticos obtenidos en esta fase. Estos tiempos son medidos en milisegundos.

Tabla 4.1: Estudio de Tiempos (en milisegundos) del Sistema de Vigilancia que detecta el Peligro de Atropello

Tiempos	**Procesamiento Traductor**	**Procesamiento Alerta**
Máximo	115	27
Mínimo	4	2
Medio	63.84	6.15
Desv.Típica	12.68	0.76

Podemos apreciar que los tiempos medios obtenidos son bastantes satisfactorios. El Sistema emplea más tiempo en tratar la información de entrada (calculando el posicionamiento 3D, las velocidades y aplicando el algoritmo de seguimiento de alto nivel), que ejecutar el Módulo de Detección de la Alerta. El tiempo máximo empleado por el Sistema en ambos procesos es inferior a 200 milisegundos, por lo que podemos afirmar que, en el peor de los casos, los resultados se generan en tiempo real, sin apreciar un ligero retraso para el ojo humano.

En el segundo test, hemos estudiado cuánto tarda en evaluarse la alerta en escenas complejas en las que el número de objetos es elevado. Para ello, hemos generado conjuntos de objetos aleatorios en la Base de Conocimiento y hemos ejecutado el Modulo de Detección de la Alerta Peligro de Atropello propuesto. En cada iteración, hemos aumentado el número de objetos evaluados por el Sistema para observar como se comporta el tiempo de procesamiento del mismo. Los objetos se han creado con una probabilidad de que el 50 % sean vehículos y el otro 50 % peatones. Los resultados obtenidos en este test pueden verse en la figura 4.27.

Podemos observar que el tiempo de procesamiento aumenta conforme se incrementa el número de objetos, aún así, los resultados son bastante eficientes. Con 50 objetos en la BC, el modelo de detección de colisiones emplea 56 milisegundos en ejecutarse, lo que implica que es un buen método como aplicación sensible al tiempo real.

Figura 4.27: Tiempos de procesamiento del Módulo de Detección de la Alerta peligro de atropello

En general, nuestra propuesta presenta unos tiempos de procesamiento muy bajos. Este hecho hace que el Sistema sea eficiente y pueda llevarse a la práctica para su aplicación en tiempo real. Sin embargo, dependemos del tracking de objetos, tanto para la fiabilidad de los datos evaluados como para los tiempos de delay. Por un lado, la calidad de las entradas influye notablemente en el análisis de nuestro Sistema. Por otro lado, si el sistema de detección y tracking de objetos presenta un retardo, este hecho se acumulará en el Sistema final. Por ello, para mantener la fiabilidad y la eficiencia demostrada en nuestra propuesta, es necesario contar con una fuente de información fiable y eficiente.

4.2. Vigilancia inteligente para la identificación de peligro por niños en zona de tráfico

Los niños son elementos vulnerables en una situación de riesgo. Si nos centramos en el ámbito del tráfico, los niños viandantes pueden convertirse en víctimas fáciles de accidentes. En muchos casos, son los causantes del peligro, ya que pueden realizar imprudencias sin ser conscientes.

Por ejemplo, un niño puede jugar y correr en una zona de tráfico, o bien cruzar una vía en un momento no oportuno, o por un lugar inadecuado para ello. Con esa conducta, el niño puede no ser visto a tiempo por los vehículos, causando un posible atropello u otro tipo de accidente (p.ej. provocar un frenazo fuerte o el desvío de dirección de un vehículo, que pueda desembocar en un choque con otro objeto de la vía).

Realmente, podemos pensar que cualquier peatón puede realizar una imprudencia. Sin embargo, los niños son puntos débiles ya que existe una mayor probabilidad de que desconozcan las normas viales y los riesgos que conlleva la realización de determinadas acciones y, en muchos casos, actúan instintivamente, sin reflexionar o sin prestar atención al entorno.

La presencia de niños en un espacio donde hay tránsito de vehículos, origina unos ciertos riesgos de los que deben ser advertidos los conductores para evitar situaciones desagradables. En el caso de

colegios, zonas de juegos o residenciales, las vías disponen de una señal de tráfico que avisa del peligro por la proximidad de un lugar frecuentado por niños. Sin embargo, existen otras áreas urbanas que también pueden ser frecuentadas por niños y que no disponen de ese tipo de aviso.

El peligro de la existencia de niños en zonas de tráfico, se acentúa si no están acompañados de adultos que puedan guiarles o avisarles de ciertos riesgos. En la vida real, se dan muchas ocasiones donde los adultos actúan de protectores de niños en las zonas donde hay circulación de vehículos. Basándonos en esta idea, analizaremos la detección de niños que se encuentran lejos de personas adultas, en zonas de tráfico.

De esta forma, desarrollaremos un Sistema que sea capaz de alertar de la presencia de niños que puedan estar en peligro por el tráfico, y a la vez, del riesgo de accidente que tienen los vehículos por la presencia de niños.

4.2.1. Objetivo. Estructura General.

El propósito es desarrollar un Sistema de Vigilancia Inteligente (aplicando el Modelo propuesto en el capítulo 3) para la detección de riesgo debido la existencia de niños en zona de tráfico. El escenario de estudio será cualquier área donde exista tráfico de vehículos y niños simultáneamente.

De forma similar al Sistema inteligente anterior, destinado a detectar el peligro de atropello, la información básica sobre cuántos vehículos y peatones hay en el entorno, cuáles son sus posiciones y velocidades, si existen niños en el área vigilada, etc., se puede obtener a partir de un análisis del contenido del vídeo y constituye la información necesaria para analizar esta nueva situación de riesgo.

Tampoco es necesario que haya micrófonos u otro tipo de sensores que monitoricen el entorno, ya que la información obtenida de estas fuentes no aportarían nada nuevo para el estudio de la presencia de niños en zonas de tráfico.

De este modo, la única fuente de información será el vídeo. Por lo tanto, el escenario de estudio estará únicamente monitorizado por, al

menos, una cámara de vigilancia.

El **objetivo** del Sistema inteligente que vamos a construir será *identificar la presencia de niños que no van acompañados de adultos en un entorno monitorizado por cámaras y donde existen vehículos en circulación*.

A grandes rasgos, diferenciamos tres partes en la estructura general del Sistema, que serán detalladas más adelante (ver figura 4.28):

- La entrada del Sistema será un flujo de eventos sobre la detección y el tracking 2D de objetos a partir de un análisis de vídeo.
- El Sistema procesará la información de entrada, complementándola con nueva información para ser estudiada por un módulo experto que analice el peligro por la existencia de niños en el entorno vigilado.
- La salida del Sistema será el nivel de riesgo debido la presencia de niños que no están bajo la tutela de adultos, mientras hay vehículos en movimiento.

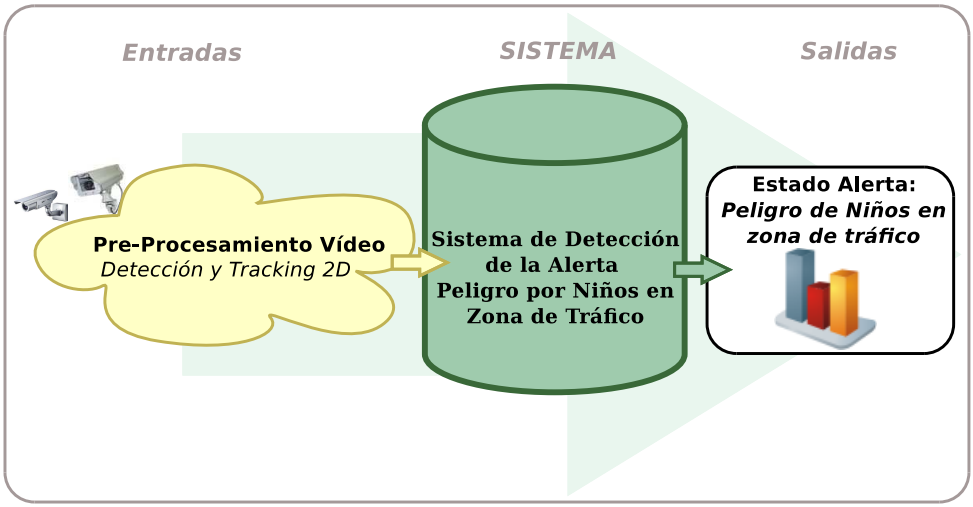


Figura 4.28: Estructura general del Sistema Inteligente para la Detección del Peligro por niños en zona de tráfico

A continuación, describimos la arquitectura del Sistema para el estudio de esta nueva situación de vigilancia.

4.2.2. Arquitectura

El Sistema de Vigilancia Inteligente para la detección de peligro por niños en una zona donde hay tránsito de vehículos, basado en el Modelo propuesto en el capítulo 3, presenta una arquitectura basada en componentes que puede verse en la figura 4.29.

Como entrada, el Sistema recibe información sobre análisis de vídeo (ver sección 4.2.3). Esta información es integrada, completada y analizada con el fin de obtener el nivel de presencia de la situación de estudio.

La arquitectura está compuesta por:

- Un **Traductor**, que es el encargado de procesar la información de entrada para complementarla e integrarla en la Base de Conocimiento del Sistema. (Ver sección 4.2.4)
- La **Base de Conocimiento, BC**,que tiene toda la información referente a todos los Objetos del Sistema. En este componente se integra la información de entrada con la información obtenida tras el procesamiento del Traductor y el análisis interno del Sistema. Este conocimiento está representado bajo la *OHRC* (ver sección 3.5).
- Tres **Plugins**: (ver sección 4.1.8)
  - *Eliminador de Objetos, EO*. Es el encargado de eliminar los objetos cuando estos hayan desaparecido del escenario.
  - *Visualización de la Base de Conocimiento, VBC*. Este plugin obtiene toda la información actual residente en la Base de Conocimiento en modo texto.
  - *Identificación de Niños, IN*. Este plugin estudia los objetos de la Base de Conocimiento que pueden ser posibles personas y los clasifica como niños en función de su altura.

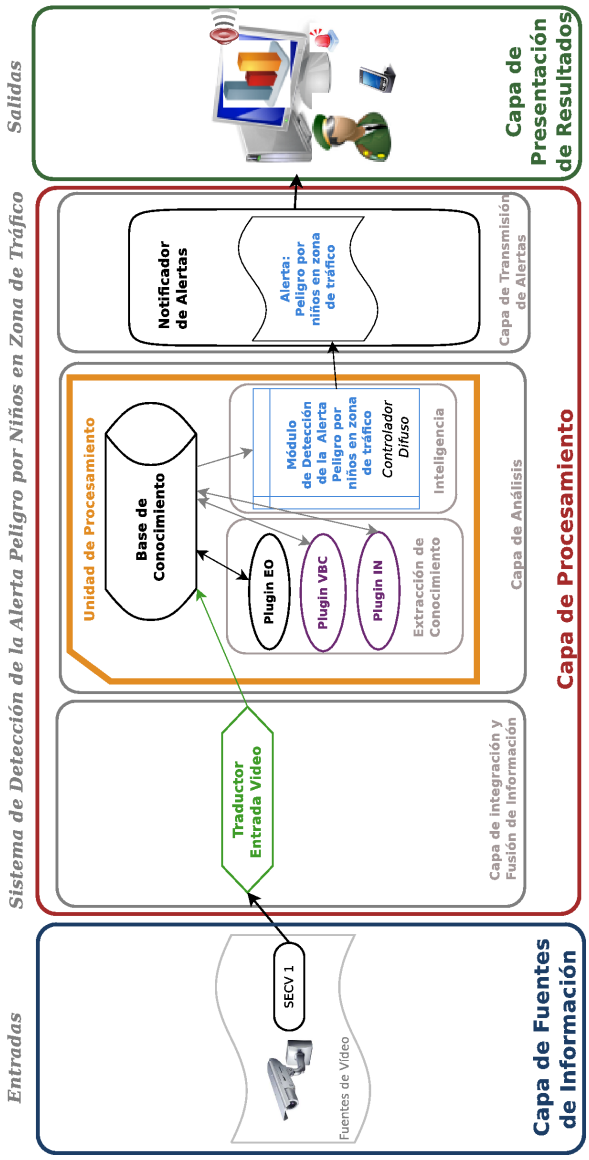


Figura 4.29: Arquitectura del Sistema de Vigilancia Inteligente para la peligro por niños en zonas de tráfico.

- Un **Módulo de Detección de la Alerta peligro por niños en zona de tráfico**, consiste en un Sistema Experto Basado en Reglas Difusas que será el encargado de analizar todo el conocimiento del Sistema, con el fin de detectar circunstancias que impliquen la presencia de niños que están, sin la tutela de adultos, en una zona donde puede haber vehículos en movimiento. (Ver sección 4.2.5)
- Un **Notificador de la Alerta**, en tiempo real y sensible al contexto. Este Notificador es un componente básico definido en el Modelo propuesto en el capítulo 3 que ha sido la base del Sistema. Este componente fue descrito en la sección 3.4.3.

Es importante resaltar que en este caso, tanto los Plugins como el Módulo de Detección de la Alerta, se ejecutan cada vez que se detecte un cambio en la Base de Conocimiento.

A continuación, en los siguientes apartados, se definirán los distintos componentes de la arquitectura de forma detallada.

4.2.3. Fuentes de información

La fuente de información de este Sistema es la misma fuente de información que se usa en el Sistema de vigilancia inteligente para la detección del peligro de atropello, la cual es descrita con mayor detalle en la sección 4.1.3.

En esta sección recordaremos que el flujo de eventos de vídeo, usado como fuente, se define con el siguiente cuádruple **(c,f,t,o)** donde:

- **c** denota el identificador de la cámara.
- **f** es el número del frame.
- **t** es el tiempo en el que es anotado el frame de vídeo.
- **o** es una la lista de objetos detectados en el frame actual. Estos objetos son representados con la 5-tupla (i,p,t,c,a), donde:
  - **i** es el identificador del objeto,

- **p** es la posición 2D del objeto.
- **t** representa el tamaño 2D.
- **c** es un triple (cp,cv,co), donde cp es el grado de creencia (entre 0 y 1) de que el objeto i sea una persona, cv es el grado de creencia (entre 0 y 1) de que el objeto i sea un vehículo y cp es el grado de creencia (entre 0 y 1) de que el objeto i sea otra cosa.
- **a** es el conjunto de identificadores de los objetos (en el frame anterior) de los que proviene el objeto de estudio.

4.2.4. Traductores

El hecho de usar en este Sistema la misma fuente de información que el Sistema Inteligente para la detección del Peligro de Atropello, implica que pueda usarse el mismo Traductor (descrito en detalle en la sección 4.1.4). El diseño de la arquitectura basada en componentes permite la reutilización de los mismos en otros sistemas, siendo una de las grandes ventajas de la arquitectura.

Recordemos que el Traductor es el encargado de procesar el flujo de información sobre de vídeo que se recibe como entrada. Esta información es complementada e integrada en la Base de Conocimiento del Sistema. Para ello, este Traductor está dotado de un procedimiento adicional que permite calcular información 3D de los objetos a partir de la información 2D recibida. De forma concreta, se calcula la posición de los objetos en el mundo real, el tamaño real aproximado, la velocidad real de los objetos... Todos estos cálculos son descritos en la sección 4.1.5.

Además, se aplica un algoritmo de seguimiento a alto nivel, basado en la posición 3D de los objetos y en su clasificación (descrito en la sección 4.1.6). Este seguimiento permite realizar una creación y una actualización adecuada de los Objetos en la Base de Conocimiento.

4.2.5. Módulo de detección de la alerta peligro por niños en zona de tráfico

El objetivo de este punto es desarrollar un *sistema experto* que analice la presencia de niños en una zona de tráfico y genere una alerta sobre el nivel de peligro detectado.

En este estudio, nos basaremos en la idea de que si existen niños que circulan por un área de tráfico sin la compañía de adultos, éstos son más propensos a poder realizar imprudencias que pongan en peligro su vida y la de posibles conductores. En cambio, si los niños se encuentran bajo la tutela de adultos, éstos podrán protegerles, guiarles y recomendarles sobre cómo actuar cuando hay tráfico de vehículos en la zona.

El Modelo que presentamos para el análisis de la presente situación de alerta se centra principalmente en tres aspectos:

- **La detección de niños en el entorno vigilado**. Este aspecto es el más importante, ya que es el eje central en el que se basa este estudio: la presencia de niños en una zona de tráfico. Si no se detectan niños en el escenario monitorizado, no hay motivo de alerta.

  Como veremos a continuación, para llevar a cabo el proceso de clasificación de niños, se estudiará la altura real de los objetos que previamente han sido clasificados como personas.

- **La identificación de personas adultas que estén cerca de los niños**. Una vez que uno o más niños hayan sido identificados en el escenario vigilado, es importante analizar la presencia de adultos cercanos a ellos.

  En este estudio, partimos de la idea de que si los niños van acompañados de personas mayores, éstos suelen estar cerca y pendientes de los niños si hay tránsito de vehículos. Por lo tanto, supondremos que cuando el Sistema detecte que hay al menos un adulto cerca de los niños es porque se trata de una persona que está a cargo de ellos. En estas situaciones, el peligro de que un niño

pueda causar una imprudencia que ocasione un riesgo de accidente disminuye, ya que suponemos que los adultos pueden indicarle o aconsejarle sobre sus movimientos.

- **La existencia de vehículos en movimiento que estén cerca de los niños**. Evidentemente, el peligro es más relevante si hay niños y tránsito de vehículos simultáneamente en el escenario de estudio.

  En el caso de que se identifique uno o más niños, el nivel de alerta se ve incrementado por la existencia de vehículos en movimiento cercanos a los niños. Si se detectan niños, pero no hay tráfico circulando, el nivel de alerta dependerá únicamente de si hay niños que están solos, lo cual puede suponer un peligro en el caso de que pueda aparecer de inmediato un vehículo en la escena.

  La detección de vehículos en movimiento debe ser un factor que incremente el nivel de alerta. De este modo, el nivel de peligro de esta situación se verá influenciado por la existencia o no de tráfico.

Como podemos observar, los diferentes aspectos a estudiar usan conceptos difusos: *estar cerca*, *ser un niño*, *ser un adulto*... Son conceptos difusos porque no es posible valorarlos con precisión. Una de las formas más apropiadas, para analizar este tipo de problemas y para razonar con incertidumbre, es el desarrollo de un ***Sistema Experto Basado en Reglas Difusas, (SBRD)***. Este es el tipo de sistema experto difuso que aquí presentamos.

Diferenciamos tres partes importantes en el controlador difuso:

- El *fuzzificador*: Se encargará en fuzzificar aquellos datos de entrada que no sean difusos.
- El *Marco de Conocimiento*: que contiene todos los datos relativos al Sistema. En este módulo distinguimos 2 partes: 1) la información que se va a analizar, que en este caso coincide con el conjunto de objetos de la *Base de Conocimiento (BC)* del Sistema de Vigilancia; 2) La *Base de Reglas*, que integra todas las reglas difusas que van a regir el Sistema.

- El *Motor de Inferencia*: que se encarga de extraer las conclusiones partiendo de los datos analizados, aplicando las reglas que rigen el Sistema. Una modificación del conocimiento conllevará una nueva evaluación de las reglas que puede dar como resultado unas conclusiones distintas.

Fuzzificador

El Sistema de vigilancia que proponemos en esta sección, al igual que el anterior, recibe información en la que existen datos crips y datos con cierto grado de incertidumbre. Por lo tanto, ya disponemos de algunos valores difusos con los que se van a operar y que no necesitan un proceso de fuzzificación. Por ejemplo, la clasificación de los objetos detectados a partir del vídeo (un objeto puede ser clasificado como *persona*, *vehículo* u *otro*, con un grado de posibilidad).

Sólo existirá un proceso de fuzzificación para los datos de entrada crisp que necesiten ser convertidos a datos difusos. Como veremos a continuación, en este controlador, el proceso de fuzzificación no se realiza directamente a partir de los datos de entrada. En primer lugar, los datos de entrada son procesados para obtener nuevo conocimiento requerido en el sistema. De forma concreta, nos referimos a: la altura real de los objetos y la distancia entre dos objetos.

De este modo, las variables de control principales que vamos a analizar, junto con los términos lingüísticos que vamos a usar son: la altura de las personas para clasificarlos como niños o adultos y, la distancia entre objetos, como cerca o lejos.

Un conjunto difuso A queda definido por su función de pertenencia μ (también conocida como función de membresía). Recordamos que esta función $\mu_A(x)$ varía de valor en el intervalo [0,1]. El valor de 1 corresponde a la máxima pertenencia. A continuación pasamos a definir cada una de las funciones de membresía de los conjuntos difusos estudiados:

- **Ser niño** ($n\tilde{i}\tilde{n}o$).

Actualmente, el único dato que conocemos para poder analizar un objeto e identificarlo como niño es su altura. Este dato se obtiene gracias al cálculo del posicionamiento 3D, que nos permite conocer la aproximación de la altura real de un objeto en el mundo (ver sección 4.1.5).

En este estudio, suponemos que solo serán evaluados para ser clasificados como niños aquellos objetos que sean previamente clasificados como personas con un grado de creencia superior a un umbral α. De esta forma evitamos clasificar a vehículos como posibles niños.

Como podemos observar, estamos suponiendo que el conjunto de niños es definido como el conjunto de personas de baja estatura. De esta forma, el concepto borroso "niño" está caracterizado por una función de pertenencia que obtiene un grado de creencia sobre cómo de niño podría ser el objeto que tenga una altura 'x'. Esta función toma un valor de referencia 'h', que indica una media sobre la altura máxima de los niños. Evidentemente, al tratar valores difusos, nosotros no vamos a afirmar que todo el que tenga una altura mayor a 'h' no va a ser un niño, sino que suavizaremos los valores. La función (ver figura 4.30) quedaría de la siguiente manera:

$$f(x) = \begin{cases} 1 & \text{si} \quad x \leq \frac{h}{2} \\ \frac{-1}{h}x + \left(\frac{h+\frac{h}{2}}{h}\right) & \text{si} \quad \frac{h}{2} < x < h + \frac{h}{2} \\ 0 & \text{si} \quad x \geq h + \frac{h}{2} \end{cases}$$

De esta forma, suponemos que cualquier objeto que su altura sea menor a la mitad de la altura de referencia, es un niño. En cambio si la altura supera la suma de la altura de referencia más la mitad de ésta, probablemente no sea un niño. En casos intermedios, se irá decrementando la posibilidad de que sea niño, es decir, aquellos que sean un poco más bajitos de la altura de referencia tienen más posibilidades de ser niños que aquellos que sean un poco más altos.

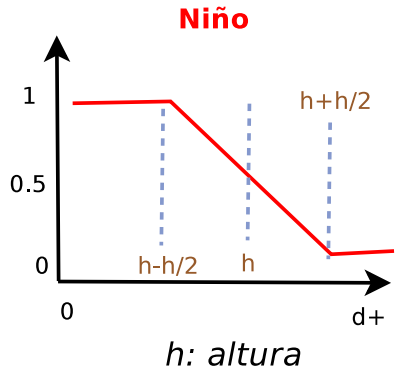


Figura 4.30: Función de pertenencia al conjunto difuso "ser niño"

- **Ser adulto** ($\tilde{adulto}$).

  Ser adulto también es un concepto borroso que depende de la altura del objeto. Nosotros definimos el conjunto difuso $\tilde{adulto}$ como el complemento del conjunto $\tilde{niño}$.

$$\mu_{\tilde{adulto}} = \mu_{\overline{\tilde{niño}}} = 1 - \mu_{\tilde{niño}}$$

- **Estar cerca** ($\tilde{cerca}$).

  Para conocer cómo de cerca están dos objetos (en función de la distancia entre sus posiciones), hemos definido una función de pertenencia al concepto difuso "cerca" (ver figura 4.31). Esta función permite clasificar una distancia como cercana o no. Para ello, usamos dos distancias de referencia d_{inf} y d_{sup}:

$$f(x) = \begin{cases} 1 & \text{si} \quad x \leq d_{inf} \\ \frac{1}{d_{sup}-d_{inf}}x + \left(1 - \frac{d_{sup}}{d_{sup}-d_{inf}}\right) & \text{si} \quad d_{inf} < x < d_{sup} \\ 0 & \text{si} \quad x \geq d_{sup} \end{cases}$$

Con esta función, suponemos que cualquier pareja de posiciones cuya distancia de separación 'x' sea menor la distancia de referencia

d_{inf}, están cerca. En cambio, si su distancia supera d_{sup}, probablemente no lo estén. En casos intermedios $x \in (d_{inf}..d_{sup})$, se irá decrementando de forma gradual la certeza de que estén cerca.

En este estudio se analizará la cercanía entre personas, y, entre personas y vehículos. En ambos casos, la distancias d_{inf} y d_{sup} de referencia puede ser diferentes. Por lo tanto, se usará la misma función de pertenencia al concepto $ceñca$ pero se instanciará dos veces. Una, para hallar la pertenencia al conjunto difuso estar cerca una persona y un vehículo. Otra, para calcular la pertenencia al conjunto difuso estar cerca dos personas. Cada una tendrá sus propios parámetros de referencia.

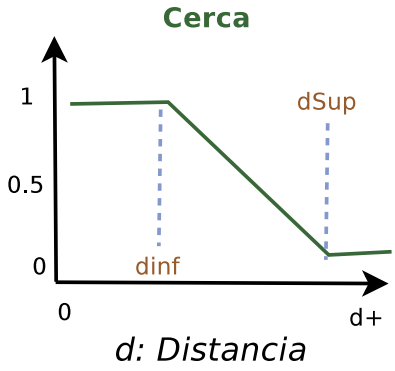


Figura 4.31: Función de pertenencia al conjunto difuso “estar cerca”

Base de Reglas

Basándonos en las ideas definidas al comienzo de esta sección y que constituyen la base para el análisis de la situación, las proposiciones de estudio que analizaremos son: si existe al menos un niño en el escenario de estudio, si no hay adultos cerca, si hay vehículos que estén cerca... Las reglas nos permiten expresar semánticamente las relaciones entre dichas proposiciones.

A continuación, de manera informal, proponemos dos ideas principales que darán lugar a las dos reglas que constituirán la base de estudio

del módulo experto que presentamos:

1. Si existe al menos un niño en el escenario de estudio y no hay adultos cerca, entonces el nivel de alerta se incrementa. (Este incremento debe hacerse en función de lo lejos que se encuentren los niños respecto a los adultos).
2. Si existe al menos un niño en el escenario de estudio y hay vehículos que estén cerca, entonces el nivel de alerta se incrementa (en función de lo cerca que se encuentren los vehículos respecto a los niños).

De esta forma, estudiamos que si existe un niño en la zona de tráfico, éste está en peligro si se dan cualquiera de las siguientes dos situaciones:

1. El adulto más cercano a él se encuentra lejos. (= no hay adultos cerca)
2. El vehículo más cercano a él está cerca. (= hay vehículos que están cerca)

Se trata de dos situaciones que implican un análisis complejo. A continuación, vamos a definir formalmente las dos reglas que constituyen el controlador difuso. Cada una de estas reglas será evaluada respecto a cada niño que sea detectado en la escena.

Regla 1. *Si no existe una persona adulta que esté cerca del niño, entonces incrementamos el nivel de alerta con un grado* α_1 *en función del grado de cumplimiento del antecedente de la regla* (w_{R1}).

$$\text{IF } \nexists\, obj\_persona\ (obj\_persona \in \tilde{adulto}\ AND\ distancia(obj\_niño, obj\_persona) \in \tilde{cerca}), \text{ THEN } F_{Incrementar\_Nivel\_Alerta}(w_{R1},\alpha_1).$$

Regla 2. *Si existe un vehículo que esté cerca del niño, entonces incrementamos el nivel de alerta con un grado* α_2 *en función del grado de cumplimiento del antecedente de la regla* (w_{R2}).

IF $\exists$ $obj\_vehiculo$ ($distancia(obj\_niño, obj\_vehiculo) \in cer\tilde{c}a$), THEN $F_{Incrementar\_Nivel\_Alerta}(w_{R2}, \alpha_2)$.

Las reglas anteriores se basan en la siguiente idea:

$$IF\ antecedente\ THENy = F(grado\ de\ cumplimiento\ del\ antecedente)$$

Motor de Inferencia

El motor de inferencia modela el proceso de razonamiento humano, usando las reglas descritas anteriormente.

Cuando un evento de entrada es detectado en el Sistema, el Módulo de Detección de la Alerta se activa y es cuando entra en funcionamiento el motor de inferencia del controlador difuso que proponemos. Los pasos a seguir por dicho motor son los siguientes:

1. *Se inicializa el nivel de alerta a cero*. De esta forma, cada vez que se ejecuta el módulo de detección de la alerta, se analizan las circunstancias actuales estudiando los objetos existentes en la Base de Conocimiento en el momento actual, sin tener en cuenta las circunstancias anteriores.

2. *Se procesan uno a uno los objetos pertenecientes a la Base de Conocimiento y se analiza si ese objeto puede ser un niño*. En este caso, solo se consideran aquellos objetos que son clasificados previamente como personas. Posteriormente se estudia si el objeto procesado puede pertenecer al conjunto difuso $n\tilde{i}\tilde{n}o$ (definido anteriormente con una función de membresía). En el caso de que el grado de posibilidad de que sea niño sea superior a un umbral de referencia, consideraremos que el objeto analizado es un niño ($obj\_niño$). El hecho de fijar un umbral permite despreciar aquellos objetos de los que apenas se tiene creencia de que sean niños.

   Posteriormente, se procede a estudiar si ese niño está en peligro. Para ello, se aplican las dos reglas definidas anteriormente:

a) *Cálculo del grado de aplicabilidad de la Regla 1.* Se estudia si no existe una persona adulta que esté cerca del niño, lo cual implicaría el aumento de nivel de peligro.

El antecedente de la regla es: $\nexists\ obj\_persona$ ($obj\_persona \in \tilde{adulto}\ AND\ distancia(obj\_niño, obj\_persona) \in \tilde{cerca}$). Esto es lo mismo que:

$$\neg\ (\ \{obj\_persona_1 \in \tilde{adulto} \wedge distancia(obj\_niño, obj\_persona_1) \in \tilde{cerca}\} \vee \{obj\_persona_2 \in \tilde{adulto} \wedge distancia(obj\_niño, obj\_persona_2) \in \tilde{cerca}\} \vee \ldots \vee \{obj\_persona_n \in \tilde{adulto} \wedge distancia(obj\_niño, obj\_persona_n) \in \tilde{cerca}\}\ )$$

donde aplicaremos el mínimo como operador AND ($\wedge = \wedge_{min}$) y el máximo como operador OR ($\vee = \vee_{max}$). El grado de cumplimiento final para el antecedente de la regla se obtendrá tras aplicar el complemento al resultado obtenido de la expresión entre paréntesis. De esta forma, se obtiene el grado de posibilidad de que no exista una persona que sea adulta y esté cerca del niño. A continuación veremos este proceso de forma detallada:

Si hay varios objetos-persona en la escena (aparte del objeto-niño evaluado), para cada uno de ellos:

1) Se analiza si ese objeto-persona es un adulto, o lo que es lo mismo, que no sea un niño. Para ello, se aplica el operador de complemento a la función de pertenencia al conjunto "ser niño" y obtenemos un grado de creencia, wA, que indica la pertenencia al concepto $\tilde{adulto}$.
2) Se estudia cómo de cerca está respecto al objeto-niño evaluado. Para ello, se calcula la distancia existente entre los dos objetos y se aplica a dicho resultado la función de pertenencia al conjunto "estar cerca". De esta forma, se obtiene wC, que es el grado de pertenencia al concepto difuso estar $\tilde{cerca}$ (entre dos personas).
3) Y posteriormente, para obtener el grado de cumplimiento de que dicho objeto sea adulto y esté cerca del niño

estudiado, se aplica el operador AND difuso sobre ambos conjuntos y obtenemos el grado wAC:

$$(\mu_{\tilde{adulto}}(altura)) \wedge (\mu_{\tilde{cerca}}(distancia)) = wA \wedge wC = min(wA, wC) = wAC$$

Una vez que se hayan evaluado todos los objetos-personas, calculamos el máximo grado obtenido en wAC (wAC_{max}). Esto quiere decir, usando términos difusos, que nos quedamos con aquella "persona que más adulto es y que más cerca está" del objeto evaluado. Este hecho influye inversamente en el nivel de alerta, cuanto mayor sea el valor wAC, el nivel de peligro detectado es menor (el niño tiene un adulto cercano). Por esta razón estudiamos el hecho de que no exista ningún adulto cerca aplicando el operador NOT sobre wAC, obteniendo el grado de cumplimiento de la regla 1: $w_{R1} = \neg wAC_{max} = 1 - wAC_{max}$.

De esta modo, el nivel de alerta es menor si hay adultos cerca y mayor en caso contrario.

b) *Cálculo del grado de aplicabilidad de la Regla 2.* Se comprueba si existe algún vehículo en la escena que esté cercano al niño.

El antecedente de la regla es: $\exists$ $obj\_vehiculo$ ($distancia(obj\_niño, obj\_vehiculo) \in \tilde{cerca}$). En otras palabras:

$$(\ distancia(obj\_niño, obj\_vehiculo_1) \in \tilde{cerca} \vee distancia(obj\_niño, obj\_vehiculo_2) \in \tilde{cerca} \vee ... \vee distancia(obj\_niño, obj\_vehiculo_n) \in \tilde{cerca}\ )$$

donde aplicaremos el máximo como operador OR ($\vee = \vee_{max}$) para calcular el grado de cumplimiento del antecedente, obteniendo un grado de posibilidad sobre la existencia de un vehículo cercano al niño. A continuación describimos este proceso:

En el caso de que haya varios objetos vehículos en la Base de Conocimiento, para cada uno de ellos calculamos la distancia existente entre el vehículo y el objeto-niño estudiado. Posteriormente, analizamos cómo de cerca es clasificada dicha

distancia y obtenemos un grado de cercanía wVC. wVC será el grado de pertenencia (entre 0 y 1) al concepto cerca de la distancia existente entre el objeto-vehículo y el objeto-niño evaluado. Este grado se obtiene aplicando la función de pertenencia al conjunto difuso "estar cerca", descrita anteriormente.

Sin embargo, el propósito es encontrar aquel vehículo que esté más cerca del objeto de estudio y obtener el grado de creencia sobre la cercanía entre ambos objetos. Para ello, nos quedamos en el máximo de los grados wVC obtenidos, que constituirá el grado de cumplimiento de la regla w_{R2}. Este hecho influirá incrementando el nivel de alerta. Una mayor proximidad entre el vehículo más cercano y el niño repercutirá en un mayor nivel de peligro.

3. *Se calcula el Nivel de Alerta detectado para el objeto evaluado.*

Para cada objeto-niño que se evalúe, se obtendrá un nivel de presencia de peligro por niño en zona de tráfico que actualizará el nivel de alerta.

El nivel de alerta final se obtiene aplicando las dos reglas consideradas. Este dato viene dado por el hecho de que alguna de las dos reglas se cumpla. $Nivel\_de\_Alerta$ = $R1 \vee R2$

En este caso lo que se pretende es que ambas reglas influyan en el nivel de alerta final. Por este motivo, aplicaremos como operador OR el método de Lukasievicz.

$$\vee_{Lukasievicz} = min[1, R1 + R2]$$

Una vez obtenido el grado de cumplimiento de ambas reglas: w_{R1}, el nivel de presencia detectado de que el niño no tenga un adulto cerca, y w_{R2}, el máximo nivel de presencia detectado de que haya un vehículo cerca del niño. Aplicamos el consecuente:

- Para la regla 1: $F_{Incrementar\_Nivel\_Alerta}(w_{R1},\alpha_1)$=$w_{R1}$*$\alpha_1$
- Para la regla 2: $F_{Incrementar\_Nivel\_Alerta}(w_{R2},\alpha_2)$=$w_{R2}$*$\alpha_2$

siendo α_1 y α_2 dos pesos que indican el grado de incremento de ambas reglas respectivamente. Gracias a estos pesos, podemos otorgar una mayor o menor importancia a los dos aspectos principales estudiados.

Por consiguiente, el nivel de presencia de peligro por niños en zonas de tráfico es:

$$Nivel\_de\_Alerta = R1 \vee_{Lukasievicz} R2 = min\ [1, w_{R1} * \alpha_1 + w_{R2} * \alpha_2]$$

En este estudio, pretendemos que la situación de alerta sea detectada en tiempo real y de la forma más eficiente posible. Por esta razón, el motor de inferencia será diseñado bajo el paradigma de la programación imperativa.

La programación imperativa, en contraposición a la programación declarativa, es un paradigma de programación que describe la programación en términos del estado del programa y de sentencias que cambian dicho estado. Los programas imperativos son un conjunto de instrucciones que le indican al computador cómo realizar una tarea.

La programación declarativa se suele aplicar a muchos sistemas basados en reglas. Sin embargo, llevar a la práctica un Sistema como éste con un lenguaje declarativo (Prolog, Lisp...) puede perder eficiencia cuando hay muchos objetos que evaluar. Esto es debido a que es necesaria una fase de interpretación extra, en la cual se deben evaluar todas las consecuencias de todas las declaraciones realizadas. El proceso es relativamente más lento que en la programación imperativa, donde los cambios de estado del Sistema están dados por instrucciones particulares y no por un conjunto de condiciones arbitrariamente grande.

Hemos diseñado un *algoritmo* que permite evaluar todas las condiciones establecidas en las reglas y aplicar las acciones definidas en los consecuentes de las mismas. El procedimiento propuesto es definido formalmente en el Algoritmo 2 [2].

[2] $objetosBC$ es el conjunto de objetos existentes en la Base de Conocimiento; $valorRefNino$ es un valor entre 0 y 1 que se toma como referencia para filtrar (no considerar) aquellos objetos que son identificados como niños con un grado de creencia menor que dicho valor de referencia; $distRefPersona$ es la distancia de referencia para

Algorithm 2 peligroNiños ($objetosBC$, $valorRefNino$, $distRefPersona$, $distRefVehiculo$,α_1,α_2)

Require: $objetosBC$, $valorRefNino$, $distRefPersona$, $distRefVehiculo$,α_1,α_2
  $Max\_Nivel\_de\_Alerta = 0$
  for $\forall\ obj \in objetosBC$ **do**
    if $\neg$ esNiño(obj,$valorRefNino$) **then**
      continue
    else
      $w_{R1} = 0$
      $w_{R2} = 0$
      for $\forall\ otro\_obj \in \{objetosBC\ -\ obj\}$ **do**
        if $esPersona(otro\_obj)$ **then**
          wA=gradoCreenciaADULTO($otro\_obj$)
          $distP$ = Distancia (obj,$otro\_obj$)
          wC=gradoCreenciaCERCANIA($distP$,$distRefPersona$)
          wAC=AND(wA,wC)
          wAC_{max}=OR(w_{R1},wAC)
        else
          if $esVehiculo(otro\_obj)$ **then**
            $distV$ = Distancia (obj,$otro\_obj$)
            wVC=gradoCreenciaCERCANIA($distV$,$distRefVehiculo$)
            w_{R2}=OR(w_{R2},wVC)
          end if
        end if
      end for
      w_{R1} = (1 - $wACmax$)
      $Nivel\_de\_Alerta$ = min [1,w_{R1}*α_1+w_{R2}*α_2]
      $Max\_Nivel\_de\_Alerta$ = max ($Max\_Nivel\_de\_Alerta$,$Nivel\_de\_Alerta$)
    end if
  end for

Como podemos observar, el modelo que proponemos, calcula el nivel de peligro para cada niño existente en la escena ($Nivel\_de\_Alerta$). El nivel de alerta global será el máximo peligro detectado ($Max\_Nivel\_de\_Alerta$) entre todos los niños identificados. Aun así, el sistema recoge en la explicación los motivos de peligro de todos los niños que se encuentren afectados, tanto en modo texto, como en lista de objetos de la OHRC, gracias a que se conoce la semántica del problema.

Los parámetros de referencia de las funciones de membresía ($valorRefNino$, $distRefPersona$, $distRefVehiculo$) de los conjuntos difusos junto con las variables α_1 y α_2 serán los parámetros de configuración del controlador difuso que proponemos. Estos parámetros pueden ser ajustados en función del escenario de estudio donde de aplique el sistema, ya que este es portable a cualquier entorno.

4.2.6. Plugins

Proponemos 3 *plugins* para este Sistema:

Eliminador de Objetos (Plugin EO)

el cálculo del grado de pertenencia de otra distancia 'x' al conjunto difuso "cerca". Se usa para analizar la cercanía entre un niño y otra persona; $distRefVehiculo$ es la distancia de referencia para el cálculo del grado de pertenencia de otra distancia 'x' al concepto difuso "cerca". Esta distancia de referencia se usa para analizar la cercanía entre dos personas; α_1 valor de peso entre 0 y 1 que indica el grado de incremento del nivel de alerta respecto a la regla 1; α_2 peso (entre 0 y 1) que indica el grado e incremento del nivel de alerta respecto a la regla 2. Otros parámetros: $Max\_Nivel\_de\_Alerta$ es el nivel de presencia de peligro por niños en zona de tráfico;$Nivel\_de\_Alerta$ es el nivel de presencia de peligro para un niño en concreto. wA es el grado de creencia obtenido sobre que una persona sea adulta. wC es es grado de posibilidad de que una persona esté cerca del niño estudiado. wAC es el grado de creencia sobre que una persona sea un adulto y además esté cerca del objeto niño evaluado. wAC_{max} es un valor entre 0 y 1 que indica el máximo grado de creencia encontrado (tras evaluar los distintos objetos) sobre si existe una persona adulta que esté cerca del objeto estudiado; w_{R1} es el grado de cumplimiento de la regla 1, consiste en un valor entre 0 y 1 que indica la posibilidad de que no exista un adulto cerca del niño; wVC es el grado de creencia que indica cómo de cerca está un vehículo respecto al objeto niño estudiado; w_{R2} es el grado de cumplimiento de la regla 2, consiste en un valor entre 0 y 1 que indica el máximo grado de creencia encontrado (tras evaluar los distintos objetos) sobre si existe un vehículo en movimiento que esté cerca del objeto estudiado.

Es el plugin básico que debe tener todo sistema que se base en el Modelo propuesto (capítulo 3). La función de este componente es eliminar aquellos Objetos que hayan salido de la escena. Su funcionamiento completo ha sido descrito en la sección 3.4.2 y complementado en la sección 4.1.8, por lo que en esta sección no volveremos a describirlo.

Visualización de la Base de Conocimiento (Plugin VBC)

Lo que se persique con este plugin es tener una ventana que dé acceso a la información conocida de cada objeto de la escena. Su función será recoger toda la información residente en la Base de Conocimiento en el momento actual y representarla de forma legible por una persona en formato texto. Este componente será útil cuando se desarrolle una herramienta software para la monitorización del Sistema. De este modo, el usuario podrá acceder en todo momento a los objetos vigentes en el escenario y conocer todos sus datos.

Identificación de Niños (Plugin IN)

Este plugin es necesario para generar nuevo conocimiento en la *BC* que pueda ser utilizado por el Módulo de Detección de la Alerta propuesto en la sección 4.2.5. Su función principal consiste en identificar niños en la escena. Para ello, se estudiarán aquellos objetos que sean clasificados como personas con un grado de certeza que supere un umbral determinado (α), y se analizará la posibilidad de que puedan ser niños.

Gracias al cálculo del posicionamiento 3D, podemos conocer la aproximación de la altura real de un objeto en el mundo (ver sección 4.1.5). A partir de este dato podríamos derivar información acerca de las propiedades del objeto, por ejemplo, *ser niño*. El hecho de identificar a niños puede ser importante en un proceso de vigilancia, ya que los niños requieren una atención especial. Por ejemplo, como en este caso, en una zona donde hay tránsito de vehículos, los niños son personas vulnerables que pueden ser causa o víctimas de accidentes, si no tiene cerca a un adulto, o corren hacia vehículos en movimiento...

Como se ha visto anteriormente, "ser un niño" es una propiedad difusa. No es adecuado decir este objeto que mide 1.50 es un niño, y

este que mide 1.51 no lo es. ¿Dónde está el margen de si lo es o no? Por este motivo, creamos el concepto borroso "niño", caracterizado por una función de pertenencia que devuelve un grado de creencia sobre cómo de niño podría ser el objeto que tenga una altura 'x'. Esta función ha sido definida en el apartado 4.2.5 (ver figura 4.30). Recordemos que se usa un valor de referencia 'h' que indica una media sobre la altura máxima de los niños. Evidentemente, al tratar valores difusos, nosotros no vamos a afirmar que todo el que tenga una altura mayor a 'h' no va a ser un niño, sino que suavizaremos los valores.

Este plugin está en función de dos parámetros: α y h.

- α es un valor comprendido entre 0 y 1. Este valor que indica el umbral de referencia que filtrará el conjunto de objetos de la BC, clasificados como personas, que se van a estudiar en este componente. En otras palabras, este plugin solo analizará si son niños (o no) aquellos objetos de la BC que son clasificados como personas con un grado de certeza superior a α. Lo que se pretende con esto es evaluar sólo aquellos objetos que fuertemente sean clasificados como personas, ya que no tendría sentido evaluar objetos altamente clasificados como vehículos, pero que tienen una leve certeza de que pueden ser personas.

- h un parámetro de referencia, medido en metros, que alude a la media de la altura máxima de los niños.

Para obtener una buena clasificación sobre la identificación de niños, es necesario configurar y ajustar estos parámetros en función del escenario de estudio.

Los datos obtenidos en este plugin actualizarán la Base de Conocimiento. Este dato será representado como una nueva cualidad *(c,v,g)* del Objeto según la ORHC (sección 3.5) donde: (c,v,g), donde:

- **c** es "ser niño".

- **v** es "niño".

- **g** es un grado de certeza entre 0 y 1 que indica cómo de cierto puede ser que este objeto sea niño en función de su altura.

El módulo de detección de la alerta usará esta información, accediendo a las cualidades del objeto, en su análisis.

4.2.7. Desarrollo del sistema

Se ha desarrollado una aplicación JAVA para llevar a la práctica el Sistema de Vigilancia Inteligente para la detección de riesgo o peligro por la existencia de niños en zona de tráfico. El middleware empleado es ICE ZeroC [58].

Se han desarrollado dos herramientas: una herramienta para la adquisición de conocimiento del entorno y otra para la monitorización del mismo, que incluye una aplicación para dispositivos móviles.

Herramienta de adquisición de conocimiento sobre el entorno: Traductores

Esta herramienta es la misma que la desarrollada para el Sistema de detección del peligro de atropello 4.1.9, ya que se emplea la misma fuente de información y el mismo Traductor. La reutilización de componentes es una de las ventajas de la arquitectura del Modelo propuesto.

La figura 4.32 muestra la interfaz gráfica de la aplicación, mediante la cual se pueden agregar de forma dinámica tantos *Traductores* como cámaras de vigilancia existan en el entorno vigilado. Para cada uno de esos Traductores, se ajustan los parámetros de calibración de la cámara.

Herramienta de monitorización de la Alerta peligro por niños en zona de tráfico

Existe otra herramienta, similar a la herramienta de monitorización del Peligro de Atropello, que ha sido desarrollada para observar el estado de la situación analizada en el escenario de estudio. En esta

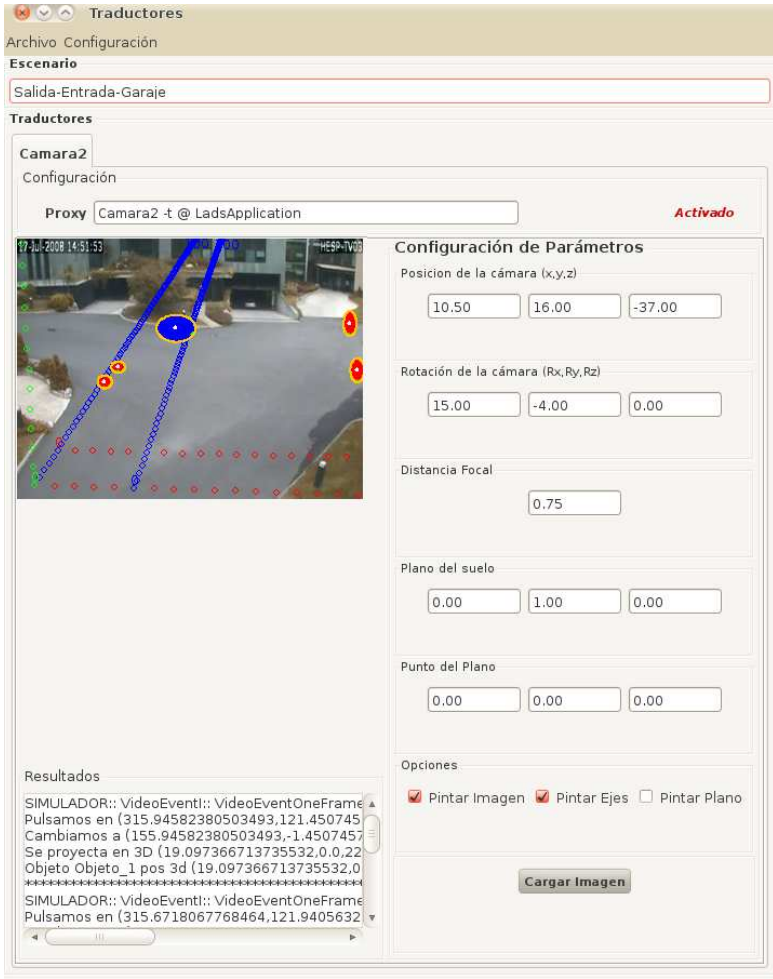


Figura 4.32: Traductor: Sistema de Vigilancia para la detección de peligro por niños en zona de tráfico

herramienta se llevan a cabo todas las tareas descritas en la capa de *Análisis*.

En la aplicación se diferencian dos partes: Plugins y Alertas, las cuales están separadas en pestañas diferentes de la interfaz de usuario. Por un lado, tenemos la configuración y la visualización de los plugins, y por otro lado está la configuración y visualización de las alertas, concretamente de la alerta Peligro por niños en zona de tráfico. Esta herramienta permite configurar y ajustar desde la interfaz de usuario aquellos parámetros que dependen del escenario a vigilar, tanto para los plugins como para la alerta.

Cada vez que exista una actualización de los datos de la Base de Conocimiento, el Módulo de Detección de la Alerta (compuesto por el SBRD propuesto) se evaluará obteniendo un resultado distinto sobre el nivel de presencia del peligro estudiado. Como se ha dicho anteriormente, este nivel de alerta es un valor perteneciente al intervalo [0,1].

Con el fin de mostrar el estado del nivel de alerta de forma fácilmente legible y comprensible, el resultado obtenido, es clasificado como riesgo alto, medio, bajo o nulo. Para ello, de igual forma que en el sistema anterior, el intervalo [0,1] correspondiente a los valores de salida obtenidos se ha dividido en subintervalos en función de un valor umbral y se les ha asignado una etiqueta y un color:

- *Estado Blanco: riesgo nulo,* cuando el nivel de alerta es cero. En este caso, no hay niños en peligro en el escenario de estudio.
- *Estado Verde: riesgo bajo,* cuando el nivel de peligro detectado por la existencia de niños en el entorno vigilado pertenece al intervalo (0, umbral/2].
- *Estado Amarillo: riesgo medio,* cuando el nivel de riesgo detectado pertenece al intervalo (umbral/2, umbral). Cuando el color es amarillo, el Sistema emite una alarma sonora que atrae la atención del operador.
- *Estado Rojo: riesgo alto,* cuando el nivel de peligro por niños y tráfico en la zona vigilada pertenece al intervalo [umbral,1]. En este

caso es muy probable que existan niños que anden solos mientras los vehículos circulan por la zona. En este caso, la alarma sonora emitida es más intensa.

Para que el nivel de alerta pueda ser fácilmente observado e interpretado, la aplicación dispone de una barra vertical que varía de color y tamaño en función de dicho nivel.

Simultáneamente a la visualización gráfica, existe un panel de texto en la aplicación donde se detallan los motivos que justifican la alteración del nivel de alerta.

En la figura 4.33 puede verse la interfaz de usuario para esta aplicación, concretamente, la pestaña que permite al usuario observar el estado de la situación de estudio que analiza el Sistema.

Actualmente, el Sistema atrae la atención del operador cuando el peligro estudiado es detectado. Posteriormente, el vigilante debe acudir a la zona o de alguna manera llamar la atención del niño(s) para que tengan cuidado con el tráfico, o bien llamar la atención del vehículo para que extremen la precaución por la presencia de niños. Si se llevase a la práctica real, lo ideal sería poder alertar en tiempo real, en el mismo escenario, a los objetos implicados.

Este Sistema puede ser portado fácilmente a cualquier entorno donde se pretenda estudiar la presencia niños que no están acompañados de adultos, en zonas donde hay o puede hacer movimiento de vehículos.

Aplicación móvil

Esta herramienta es la misma que la construida para el Sistema anterior. Se trata de una aplicación móvil (desarrollada en JavaME CLDC) que permite suscribirse a cualquier alerta. Para ello, el usuario introduce el identificador de la alerta sobre la que desea ser informado y el umbral a partir del cual desea recibir información. También puede indicar cada cuánto tiempo quiere ser informado, por ejemplo, cada segundo.

Una vez realizada esta suscripción, el usuario recibirá notificaciones sobre el estado de la alerta. Se podrá ver el nivel de alerta de for-

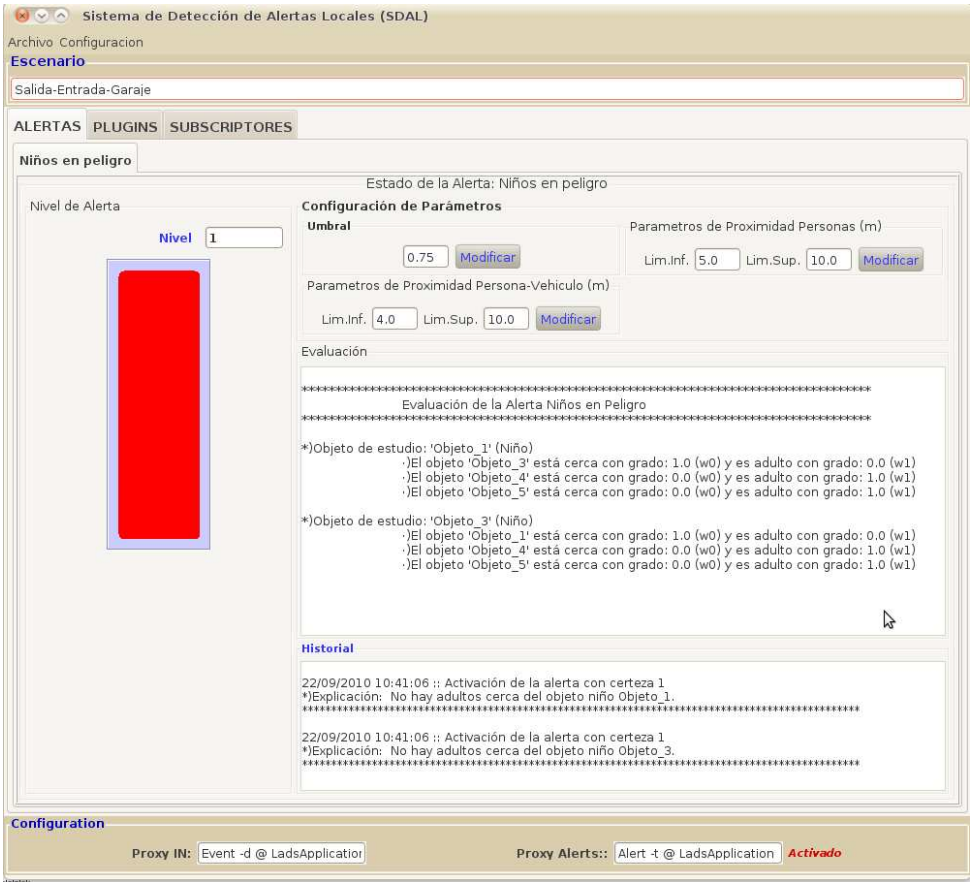


Figura 4.33: Aplicación de Escritorio: Sistema de Vigilancia para la detección del peligro por niños en zona de tráfico.

ma gráfica y la explicación de la misma en formato texto. Esta aplicación permite visualizar el estado de las alertas, usando una barra gráfica que cambia de color y tamaño en función del nivel de riesgo detectado.

4.2.8. Fase experimental

El Sistema propuesto ha sido sometido a una fase experimental. El escenario de prueba seleccionado consiste una zona de entrada y salida de vehículos a un garaje, donde cabe la posibilidad que que existan niños circulando o jugando.

Al igual que en el Sistema anterior, se ha confeccionado un conjunto de ejemplos, que muestran diferentes situaciones reales sobre el entorno vigilado en este caso. Concretamente, se han simulado datos de vídeo anotado sobre 10 situaciones diferentes, en las que aparecen entre 3 y 7 objetos.

El sistema experto basado en reglas difusas ha sido implementado y probado de forma satisfactoria. La lógica implementada se corresponde con la definición de niños en peligro que se definió:*"niños lejos de adultos en zonas de tráfico"*. Sin embargo, hemos de decir que este modelo es sensible a la precisión de los datos de entrada. La sensibilidad radica sobretodo en la identificación de niños, ya que depende totalmente del tamaño 2D identificado en el proceso de detección y seguimiento de objetos. Cuando existe una distancia considerable de la cámara a la escena, unos pocos de píxeles pueden suponer varios decímetros en la realidad.

En la mayoría de los tracking desarrollados en la actualidad, el tamaño de los objetos no es detectado con exactitud, a veces es sobre-estimado y otras infra-estimado. Estos errores en la capa inferior se transmiten a la capa superior donde tiene lugar la identificación de niños en función de la altura que se ha recibido desde el tracking. En caso de que exista una sobre-estimación o infra-estimación se produce una identificación de niños con falsos negativos o falsos positivos; esto es, que existan niños que no sean identificados o adultos que sean identificados como niños. Por ello, estos errores ocurrirán derivados del tracking usado y no del proceso en sí que hemos detallado previamente.

Para los distintos ejemplos, se han medido los tiempos de procesamiento. El equipo utilizado para realizar estas pruebas es un Pentium (R) Dual-Core CPU E5200@2.50GHz 2.51GHz, 2,75 GB de RAM. En este estudio, diferenciamos dos aspectos del Sistema que implican procesamientos diferentes:

- Por un lado, se ha estudiado el tiempo que emplea el Traductor en procesar los datos de entrada. Para ello, hemos medido el tiempo que transcurre desde que el Sistema recibe un evento de entrada hasta que el Traductor actualiza la Base de Conocimiento con dicha información. En esta fase, se realizan diversos procesamientos: cálculo del posicionamiento 3D, obtención de velocidades y aplicación del algoritmo de seguimiento a alto nivel.
- Por otro lado, se ha evaluado el tiempo que el Sistema emplea en evaluar la situación de alerta tras detectar un cambio en la Base de Conocimiento.

En la tabla 4.2 pueden verse los datos estadísticos obtenidos en esta fase experimental. Estos tiempos son medidos en milisegundos.

Tabla 4.2: Estudio de Tiempos (en milisegundos) del Sistema de Vigilancia que detecta el Peligro de Niños en zonas de tráfico

Tiempos	**Procesamiento Traductor**	**Procesamiento Alerta**
Máximo	327	51
Mínimo	2	3
Medio	140.31	17,51
Desv.Típica	61.65	10,66

Los tiempos medios obtenidos muestran que el Sistema es eficiente. Podemos observar que el Módulo de Detección de la Alerta emplea menos tiempo en procesarse que el Traductor, ya que éste realiza procesos algo más costosos en tiempos, como el cálculo del posicionamiento 3D, de las velocidades y la aplicación del algoritmo de seguimiento de alto nivel. Aún así, estamos hablando de pocas centenas de milisegundos.

El tiempo máximo empleado por el Sistema en ambos procesos no llega a ser ni medio segundo, por lo que podemos afirmar que, en el peor de los casos, los resultados se generan en tiempo real, sin apenas apreciar un ligero retraso.

Para probar cuánto tardaría en evaluarse la alerta en escenas complejas, en las que el número de objetos es mayor que 5, hemos generado conjuntos de objetos aleatorios en la Base de Conocimiento y hemos ejecutado el Modulo de Detección de la Alerta Peligro por niños propuesto. En cada iteración, hemos aumentado el número de objetos evaluados por el Sistema para observar como se comporta el tiempo de procesamiento del mismo. Los objetos se han creado con una probabilidad de que el 30 % sean vehículos, otro 30 % sean personas y el otro 40 % restante sean niños. Los resultados obtenidos en este test pueden verse en la figura 4.34.

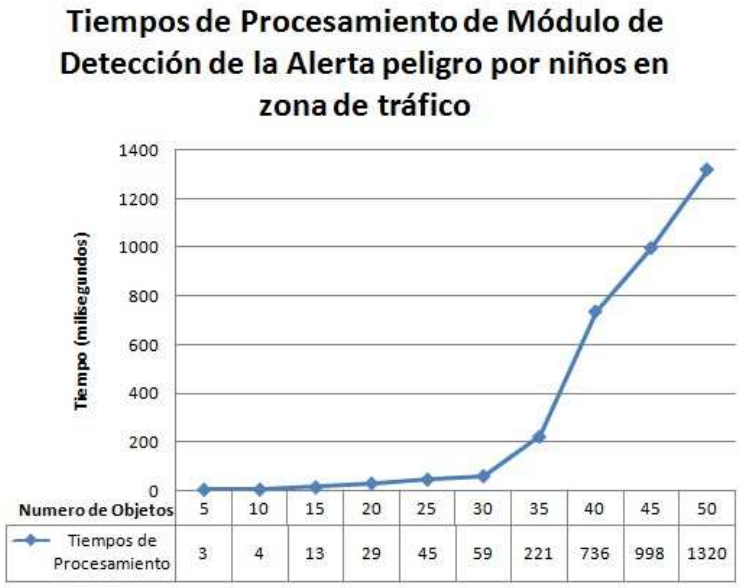


Figura 4.34: Tiempos de procesamiento del Módulo de Detección de la Alerta peligro por niños en escenas de tráfico

Podemos observar que, en escenas donde el número de objetos es

menor a 40, el tiempo es inferior a un segundo, lo que es prácticamente inapreciable. Un mayor número de objetos conlleva un mayor tiempo de procesamiento. Vemos que para unos 50 objetos, el tiempo empleado es algo más de un segundo. Realmente es difícil que en circunstancias reales se de este número considerable de objetos. Aún así, el Sistema no deja de ser apto para su aplicación en tiempo real.

Recordemos que, el Sistema propuesto depende de la fuente de pre-procesamiento que lo alimente, tanto para la fiabilidad de los datos evaluados como para los tiempos de delay. Si la detección y el tracking de objetos presenta un retardo, éste se acumulará en el Sistema final. Por ello, como se ha dicho anteriormente, para mantener la fiabilidad y la eficiencia demostrada en nuestra propuesta, es necesario contar con una fuente de información fiable y eficiente.

4.3. Vigilancia inteligente para la identificación de intrusiones

La intrusión de individuos a espacios físicos donde está restringido el acceso supone un riesgo para las infraestructuras y las personas. Muchas de estas intrusiones acaban en robos, asaltos o incluso agresiones a ciudadanos. Cuando se necesita defender una infraestructura y a las personas que residen en ella, la sociedad recurre a medios tradicionales que puedan alertar de posibles intrusiones, como la existencia de guardias de seguridad y la implantación de cámaras de vigilancia y, en raros casos, se implantan también sensores de movimiento. Como se ha dicho en capítulos anteriores, es difícil encontrar sistemas de seguridad inteligentes que sean capaces de integrar información proveniente de distintos tipos de sensores.

Con el fin mantener la seguridad en las infraestructuras, proponemos un Sistema de Vigilancia Inteligente que sea capaz de alertar, en tiempo real, de circunstancias que supongan un riesgo de intrusión en un espacio protegido y restringido.

Es muy importante destacar que una situación de intrusión es totalmente dependiente de contexto, ya que va ligada al escenario y

a las normas impuestas sobre el entorno vigilado. Unos determinados acontecimientos pueden ser considerados como una intrusión o no, en función del contexto del escenario de estudio. Por lo tanto, para cada entorno, el concepto de intrusión será distinto. Incluso, para un mismo escenario, dependiendo de si es un día festivo o no, la intrusión puede ser definida de forma diferente.

Nuestra propuesta se basa en crear un componente (Módulo de Detección de Alertas) para la detección de intrusiones, válido sobre cualquier tipo de escenario. De esta manera, se logra que el Sistema sea fácilmente portable de un entorno a otro y mantenemos las características del modelo definido en el capítulo 3.

4.3.1. Objetivo. Estructura General.

El propósito es desarrollar un Sistema de Vigilancia Inteligente para la detección de intrusiones, basándonos en Modelo propuesto en el capítulo 3 para desarrollar sistemas de seguridad. El escenario de estudio podrá ser cualquier zona donde se pretenda estudiar las intrusiones en cualquier área de acceso restringido.

Este Sistema, a diferencia de los anteriores, se alimentará de información proveniente de diversas fuentes: vídeo, audio y sensores. Ya que, cuanta más información haya disponible sobre los acontecimientos que tienen lugar en el espacio vigilado, más potente será el Sistema. Será necesario conocer: ¿Cuántos objetos hay en la escena? ¿Qué tipos de objetos son? ¿Qué trayectorias realizan? ¿Qué ruidos han sido detectados en el entorno? ¿Qué sensores han sido activados?... Para así poder detectar anormalidades que ocurren en un espacio con acceso restringido. Por lo tanto, el escenario de estudio estará monitorizado por al menos una cámara de vigilancia, un micrófono y algún tipo de sensor.

El objetivo de este Sistema inteligente será identificar la presencia de circunstancias anormales que impliquen un cierto grado de intrusión en una determinada zona.

A grandes rasgos, diferenciamos tres partes en la estructura general del Sistema, que serán detalladas más adelante (ver figura 4.35):

- Las entradas del Sistema serán varios flujos de información procedente de diversas fuentes: vídeo, audio y sensores.
- El Sistema procesará la información de entrada, complementándola con nueva información para ser estudiada por un módulo experto que analice el riesgo de intrusiones en el entorno vigilado.
- La salida del Sistema será el nivel de riesgo o presencia de intrusiones en el escenario de estudio.

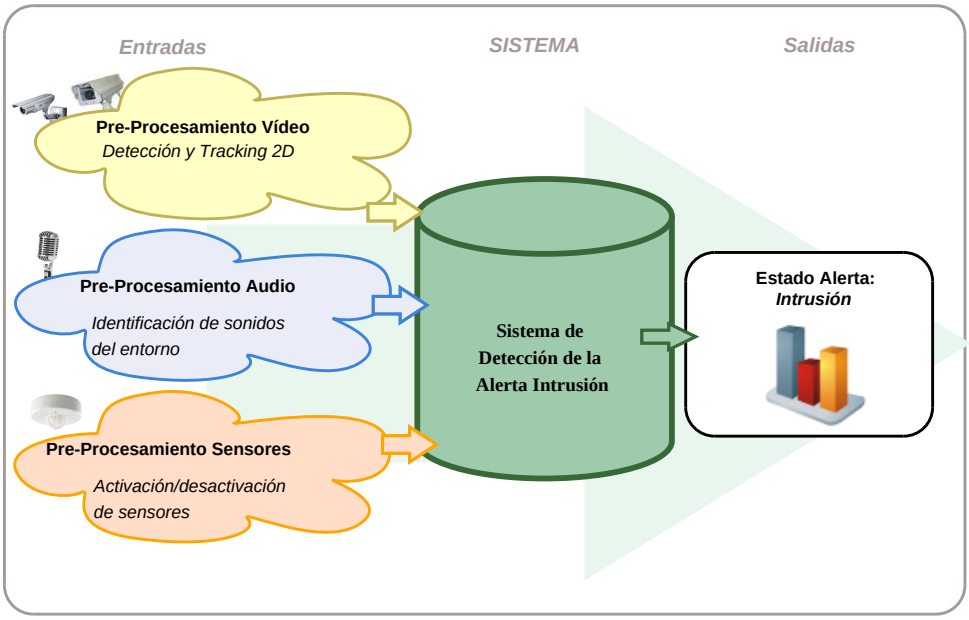


Figura 4.35: Estructura general del Sistema Inteligente para la Detección de Intrusiones

A continuación, describimos la arquitectura del Sistema de Vigilancia para la detección del Peligro de Intrusión.

4.3.2. Arquitectura

El Sistema de Vigilancia Inteligente para la Detección de Intrusiones, según el Modelo propuesto en el capítulo 3, presenta una arqui-

tectura basada en componentes que puede verse en la figura 4.36.

El Sistema recibe información de diversas fuentes: vídeo, audio y sensores, (ver sección 4.3.3). Como salida, obtiene el nivel de presencia de una intrusión en el escenario de estudio. Esa arquitectura está compuesta por:

- Cuatro **Traductores**: dos Traductores para la fuente de vídeo, uno para la fuente de sensores y otro para la fuente de audio. Estos componentes son los encargados de procesar la información de entrada para complementarla e integrarla en la Base de Conocimiento del Sistema. (Ver sección 4.3.4)
- La **Base de Conocimiento, BC**,que tiene toda la información referente a todos los Objetos del Sistema. En este componente se integra la información de entrada con la información obtenida tras el procesamiento y el análisis interno del Sistema.
- Tres **Plugins**: (ver sección 4.3.6)
  - *Eliminador de Objetos, EO*. Es el encargado de eliminar los objetos cuando estos hayan desaparecido del escenario. Este plugin se evalúa cada vez que hay un cambio en la Base de Conocimiento.
  - *Visualización de la Base de Conocimiento, VBC*. Este plugin obtiene toda la información actual residente en la Base de Conocimiento en modo texto. Este plugin se evalúa cada vez que hay un cambio en la Base de Conocimiento.
  - *Relación de Eventos de Audio y Sensores con Objetos Móviles (REASOM)*. Se encarga de relacionar los objetos detectados a partir del vídeo (personas, vehículos), con los ruidos detectados del entorno o la activación de sensores. De esta forma, el Sistema identifica a los posibles causantes de los eventos auditivos o de sensorización. Este plugin se evalúa cada vez que el Sistema recibe un evento sobre audio o sensores.
- Un **Módulo de Detección de la Alerta Intrusión**, que será en encargado de analizar todo el conocimiento del Sistema con el fin

de detectar circunstancias que impliquen la presencia intrusiones. (Ver sección 4.3.5). Este módulo se evalúa cada vez que hay un cambio en las acciones de los Objetos del Sistema.

- Un **Notificador de la Alerta**, en tiempo real y sensible al contexto. Este Notificador es un componente básico, definido en el Modelo propuesto en el capitulo 3, que ha sido la base del Sistema. Este componente fue descrito en la sección 3.4.3.

Cada uno de los componentes de la arquitectura, son descritos con detalle en las secciones sucesivas.

4.3.3. Fuentes de información

La identificación de intrusiones en un entorno vigilado, siguiendo el Modelo propuesto, necesita disponer del conocimiento sobre qué acciones tienen lugar en el escenario de estudio. Estas acciones pueden derivarse a partir de información sobre vídeo. Como hemos visto anteriormente, las técnicas de detección y seguimiento 2D de objetos a partir del análisis de imágenes nos han permitido obtener información acerca de la clasificación, la posición, la velocidad y el tamaño de los objetos en el mundo real. Sin embargo, en este caso, sería necesario avanzar en otros aspectos que nos permitan conocer dónde se encuentra los objetos en el escenario y que la contestación a la pregunta no sea un punto en el espacio. Buscamos una respuesta más semántica (por ejemplo, el objeto está en la puerta, en la plaza...).

Una solución pasaría por dividir el espacio en zonas con etiquetas semánticas y en función de la posición del objeto comprobar en qué zona se encuentra. Estas zonas se representarían como acciones asociadas a los objetos, en función de su movimiento por el escenario. Sin embargo, existen muchos estudios que realizan tareas similares, por ejemplo, la identificación de las circulaciones que siguen los objetos a partir del vídeo. Usar este conocimiento experto nos resolvería nuestro requerimiento sin detenernos y nos permitiría seguir avanzando en nuestro estudio.

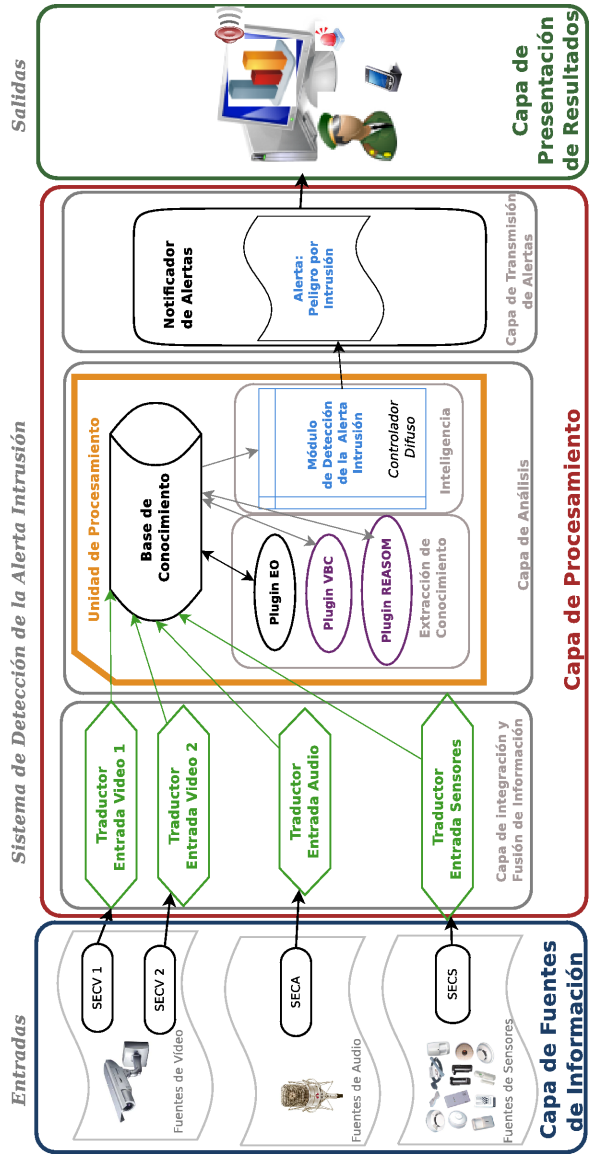


Figura 4.36: Arquitectura del Sistema de Vigilancia Inteligente para la Detección de Intrusiones.

Si además de disponer del conocimiento sobre las trayectorias que realizan los objetos, complementamos esos datos con información procedente de otros estudios, como la clasificación de ruidos en el entorno o la activación y desactivación de sensores que nos informen de otras acciones que tiene lugar en la escena, estaríamos enriqueciendo la Base de Conocimiento del Sistema. Este hecho hace que el Sistema sea más potente en su etapa de análisis y razonamiento ya que dispone de mayor información para poder interpretar la escena.

Lo ideal sería que el Sistema se alimentarse de cierto conocimiento que dé información sobre la escena, concretamente sería necesario conocer al menos:

1. ¿Qué identificador tienen los objetos móviles de la escena?
2. ¿Qué tipo de objeto es? ¿Es un vehículo o una persona?
3. ¿Cuál es su posición dentro del escenario?
4. ¿A qué velocidad se mueven?
5. ¿Qué trayectoria siguen?
6. Si hay un ruido extraño ¿de qué se trata? ¿Es un disparo? ¿Es una rotura de cristales? ¿Es que alguien está hablando?
7. Si existen otro tipo de sensores (de incendio, de movimiento...) sería útil saber cuándo son activados o desactivados.

Para conocer dicha información es necesario usar resultados sobre análisis de vídeo, audio y sensores, los cuales constituirían las fuentes de información de nuestro Sistema.

Análisis de Vídeo.

En este caso, también se puede emplear el mismo análisis de vídeo que se usó en el Sistema de Vigilancia Inteligente para la detección del Peligro de Atropello, donde se realiza un pre-procesamiento de señales de vídeo llevando a cabo una detección, clasificación y seguimiento 2D de objetos.

El flujo de vídeo genera este sistema [136] fue descrito en 4.1.3. Además, ya tenemos diseñado y desarrollado el Traductor para dicha fuente.

Esta información debe ser completada con la identificación de trayectorias que realizan los objetos. Necesitaríamos otro flujo de análisis de vídeo que informe de ello. Este nuevo flujo de datos de entrada debe tener, al menos, los siguientes datos:

- **i** es el identificador del objeto, que debe corresponder con el identificador detectado en el análisis [136].
- **c** es el conjunto de circulaciones asociadas con el objeto en el momento actual, es decir, de posibles circulaciones que el objeto puede estar realizando actualmente. Sería apropiado que cada circulación vaya acompañada de un grado de creencia (entre 0 y 1) que indique el nivel de certeza con el que se detecta que el objeto i esté realizando dicha circulación.

Un trabajo que realiza un procesamiento sobre vídeo para analizar las trayectorias que siguen los objetos es descrito en [7]. El objetivo principal de este estudio es estudiar la normalidad de un entorno vigilado, centrándose en el análisis de trayectorias y velocidades de los objetos. Basándonos en nuestra idea de reutilizar código experto, podemos usar sus resultados porque se asemejan bastante a nuestros requerimientos. El Sistema desarrollado en este trabajo [7] se alimenta también de un pre-procesamiento inicial que realice una detección y un tracking 2D de objetos a partir de vídeo. Para llevar a cabo su análisis el escenario es dividido en zonas. Y las circulaciones son definidas como trayectorias que se inician desde una zona de origen a una zona de destino, pasando por un conjunto de zonas intermedias. Parte del flujo de información que ellos generan tras su análisis, y que es interesante para nuestro estudio, es representado por una triple *(i,z,c)*, donde:

- **i** es el identificador del objeto.
- **z** es el conjunto de zonas donde actualmente puede encontrarse el objeto. Cada zona asociada tiene un grado de creencia que indica la posibilidad detectada de que el objeto se encuentre en dicha zona.

- **c** es el conjunto de circulaciones asociadas con el objeto en el momento actual, es decir, de posibles circulaciones que el objeto puede estar realizando actualmente. Cada circulación tiene un grado de creencia que indica cómo de normal es que el objeto i esté realizando dicha circulación (en función de la clasificación del objeto y del horario en que se esté realizando, entre otros factores).

Este análisis es generado frame a frame, por lo para cada frame dispondríamos de la información (i,z,c) para cada objeto detectado.

El inconveniente encontrado para nuestro estudio es que no conocemos el índice de certeza con el que una circulación se está realizando, pero este nuevo conocimiento es generado en el traductor con los datos que hasta ahora tenemos (ver sección 4.3.4).

Análisis de Audio.

La información acerca de ruidos en el entorno puede ser muy útil en el Sistema propuesto. Un trabajo muy interesante que realiza un análisis cognitivo sobre audio para conseguir una clasificación en tiempo real de ruidos en el entorno diferenciando entre: *'rotura de cristal', 'disparo', 'sirenas' y 'dialogo'*, es descrito en [92]. Como este trabajo obtiene resultados muy apropiados que nuestro Sistema puede usar, proponemos usarlo como fuente de información.

El flujo de información obtenido tras el análisis de este preprocesamiento de las señales de audio es representado por el cuádruple **(i,m,g,p)**, donde:

- **i** es el identificador del evento detectado, en este caso: *'rotura de cristal', 'disparo', 'sirenas' y 'dialogo'*.
- **m** es el identificador del micrófono que ha detectado el evento.
- **g** es el grado de creencia con el que ha sido clasificado como i.
- **p** es la posición aproximada relativa al micrófono que lo ha captado.

El hecho de obtener la posición donde ha tenido lugar el evento de sonido nos puede dar más información, como veremos a continuación en la sección 4.3.6.

Pre-procesamiento de sensores.

EL flujo de entrada que necesita el Sistema tras un pre-procesamiento de la señal de sensores es representado mediante una dupla **(s,(x,g))**, donde:

- **s** es el identificador del sensor.
- **(x,g)**, es un par donde x es la acción que ha detectado el sensor, (*activación* o *desactivación*) y g es el índice de creencia que indica el nivel de certeza con el que ha sido detectada dicha acción.

4.3.4. Traductores

Actualmente, el Sistema recibe 4 flujos de información diferente: dos flujos sobre análisis cognitivo de vídeo, un flujo sobre análisis cognitivo de audio y un flujo sobre el pre-procesamiento de sensores. Es necesario desarrollar un traductor para cada uno de los flujos heterogéneos de entrada.

Traductor para la fuente de Vídeo.

El Sistema recibe como entrada dos flujos de información sobre análisis de vídeo. Uno de ellos se corresponde con los resultados del análisis descrito en [136]. El traductor para esta fuente ya ha sido descrito en la sección 4.1.4. Este traductor procesa el flujo de datos recibido y lo complementa obteniendo las velocidades de los objetos, sus características en el mundo real (posicionamiento 3D, velocidad 3D, tamaño real...) (ver sección 4.1.5) y realiza un seguimiento para solventar algunas situaciones que el tracking 2D no abarca (ver sección 4.1.6).

Para integrar en el Sistema el nuevo flujo de información sobre un análisis de vídeo [7] es necesaria la existencia de un nuevo Traductor.

Recordamos que este flujo de vídeo muestra para cada objeto i un conjunto de zonas donde puede estar (junto con un grado de creencia que indica la posibilidad de que el objeto esté allí) y un conjunto de circulaciones que puede estar haciendo (junto con un grado de creencia sobre la normalidad de que dicho objeto realice dicha trayectoria). Este flujo de información es descrito con detalle en 4.3.3

Con el fin de que la información de vídeo sea coherente, el sistema descrito en [7] debe procesar los resultados del análisis de detección y tracking descrito en [136] (correspondiente a nuestra primera fuente de vídeo). De esta forma, ambos pre-procesamientos evaluarán los mismos objetos y los identificadores concordarán.

El traductor, para llevar a cabo la fusión e integración de esta nueva información realizará los siguientes pasos:

1. Buscar en la *Base de Conocimiento* el Objeto con identificador i.
2. Añadir el conjunto de circulaciones c asociadas al objeto. Cada circulación será representada como una ***acción*** en el objeto, siguiendo el esquema conceptual definido en la *ORHC*(ver sección 3.5)).

Como veremos a continuación, el segundo paso no es trivial. Recordemos que una *acción* es definida en la ORHC como un cuádruple *(i,g,*t_i*,*t_f*)* donde, a grandes rasgos:

- **i** es el identificador de la acción.
- **g** es un grado de creencia sobre la posibilidad de que el objeto esté realizando o haya realizado la acción i.
- t_i representa el tiempo cuando se inició la acción.
- t_f representa el tiempo de finalización de la acción.

Con los datos que tenemos, solo podemos cubrir el identificador i de la acción que se correspondería con el identificador de la circulación. Los demás datos deben ser calculados en el traductor:

- *Cálculo del grado de creencia que indica la certeza de que se esté realizando la acción*. En este caso, coincide con el grado de certeza con el que el objeto está realizando la circulación. Este valor no es conocido con la información de entrada, pero se puede obtener a partir de la información referente a las zonas actuales donde se encuentra el objeto (conjunto z, ver sección 4.3.3). El proceso sería el siguiente:

  1. Se seleccionan las zonas que formen parte de la circulación estudiada, (bien porque sean el origen, el destino o las zonas intermedias), desde el conjunto detectado de posibles zonas (z) donde actualmente se encuentra el objeto.
  2. Se calcula la media de los grados de certeza de las zonas seleccionadas.
  3. El grado de creencia de la acción será la media calculada en el paso 2.

- *Tiempo de inicio de la acción*. El tiempo de inicio solo se actualiza la primera vez que se crea la acción y se añade al objeto. En el caso de que la acción ya exista, el tiempo de inicio no se actualiza.

- *Tiempo de finalización de la acción*. El tiempo de finalización de la acción solo se actualiza si ésta existe previamente (la primera vez no). Mientras lleguen eventos que indiquen que el objeto está realizando la misma circulación, el Traductor solo actualizará el grado de creencia y el tiempo de fin de la acción en el Objeto. Cuando dejen de llegar eventos sobre esa circulación, se dejará de actualizar el tiempo de finalización.

Es importante tener en cuenta la posibilidad de que se vuelva a repetir la acción después de un tiempo. Para ello se debe de controlar que: si llega un evento en el que se asocia una determinada circulación a un objeto, primero hay que comprobar si en el evento anterior también se había notificado que dicha circulación se había asociado a dicho objeto. En caso afirmativo, como se ha dicho anteriormente, se actualiza el grado de creencia y el tiempo de fin de la acción en el Objeto. En caso negativo, se crea una nueva acción con su grado de creencia y su tiempo de inicio.

En este caso, puede ser que se trate de la primera vez o bien que la acción ya se haya realizado previamente.

Es conveniente que el mismo traductor esté dotado de un proceso que compruebe la existencia de acciones que han sido iniciadas una vez y no han sufrido actualizaciones durante tiempo. En este caso, se podrían eliminar esas acciones porque el hecho de no haber habido ningún evento más sobre ellas supone que probablemente no se hayan realizado.

Traductor para la fuente de Audio.

Este componente será el encargado de recibir los eventos de audio e integrar la información recibida en la Base de Conocimiento. El flujo de información sobre audio fue descrito en la sección 4.3.3. Recordemos que contiene el identificador del micrófono (m), el tipo de evento detectado (i) y el grado de creencia (g) con el que ha sido clasificado, y la posición aproximada donde ha sido identificado (relativa al micrófono).

El proceso que sigue este traductor cada vez que recibe un evento sobre audio consiste en los siguientes pasos:

1. Integra la información sobre el tipo de evento recibido sobre los datos del micrófono. Para ello, se comprueba si existen un objeto fijo en la Base de Conocimiento con el mismo identificador m del micrófono.

    - En caso afirmativo, se sabe que ya existe el micrófono creado como objeto de la BC y por lo tanto se le asocia una nueva acción a dicho objeto fijo. Esta acción va a reflejar el nuevo evento de sonido detectado.

      Recordamos que una acción se define con cuatro variables: identificador, grado de creencia, tiempo de inicio y tiempo de fin. El identificador i del evento de sonido constituye el identificador de la nueva acción. El grado de creencia de la misma será el grado de creencia g que indica el nivel de certeza de que el sonido detectado sea del tipo i. En este caso, los tiempos de inicio y fin de la acción coinciden con el tiempo actual en el que se ha recibido el evento.

- En caso negativo, se crea un nuevo objeto fijo en la Base de Conocimiento que simboliza el micrófono. Esto solo se realiza la primera vez que se detecta un sonido desde dicho micrófono. Los objetos fijos se caracterizan porque no pueden ser borrados por el Eliminador de Objetos. Los campos de este nuevo objeto, respetando el esquema descrito en la ORHC, quedarían de la siguiente manera:
  - El *identificador del Objeto* es m, el identificador del micrófono.
  - El *grado de actividad* o vigencia del objeto es 1.
  - La lista de identificadores contiene solo el identificador m.
  - El *tiempo* de la última actualización del objeto en el Sistema es el tiempo actual en el que se ha recibido el evento.
  - Las *cualidades* iniciales del objeto representadas como un triple *(clase, valor, grado de creencia)* son:
    - ($tipo$, $evento\ audio$, 1). Esta cualidad indica que es un objeto creado a partir de la recepción de un evento de sonido.
    - ($clase$, $microfono$, 1). Esta cualidad alude a la clasificación del objeto como micrófono.
    - ($objeto$, $fijo$, 1). Esta cualidad indica que es un objeto fijo en la BC y no puede ser borrado por el Eliminador de Objetos.
  - La *acción* asociada al objeto consiste en la información obtenida del nuevo evento de sonido detectado. El identificador i del evento de sonido constituye el identificador de la nueva acción. El grado de creencia de la misma será el grado de creencia g que indica el nivel de certeza de que el sonido detectado sea del tipo i. Los tiempos de inicio y fin de la acción coinciden con el tiempo actual en el que se ha recibido el evento.
  - Como *localización* del objeto, solo rellenamos el parámetro referente a la posición del micrófono en el escenario. Evidentemente, los valores de velocidad, tamaño, tiempo e

incremento de tiempo quedan inicializados a cero porque no son necesarios en este tipo de objetos.

La posición donde se encuentra el micrófono no se obtiene a partir del evento de sonido. El Sistema debe de disponer de un dominio de información, donde queden registrados cada uno de los micrófonos que se usen en el espacio vigilado. De cada micrófono, se debe de conocer un identificador que lo identifique unívocamente y la posición donde ha sido instalado dentro del escenario. Esta posición debe ser relativa al mismo eje de referencia usado para la calibración de las cámaras. De esta manera, el Traductor de la fuente de audio consulta la posición del micrófono sobre el dominio de información a partir del identificador del micrófono (dato que recibe en el evento).

2. Se crea un nuevo Objeto en la BC que simbolice el evento de sonido. Por ejemplo, si se detecta una rotura de cristales o un disparo..., se crea un Objeto dinámico en el Sistema que dure un tiempo hasta que el Eliminador de Objetos lo borre. Durante ese tiempo, el Sistema podrá razonar con él. Una vez que sea eliminado, supondremos que evento ya ha pasado. Sin embargo, dicho evento queda reflejado y guardado en el historial de acciones del micrófono.

Los campos de este nuevo objeto que refleja el evento de sonido, siguiendo el esquema descrito en la ORHC, son:

- El *identificador del Objeto* es i, el identificador del evento de sonido.
- El *grado de actividad* o vigencia del objeto inicialmente es 1.
- La lista de identificadores contiene solo el identificador i.
- El *tiempo* de la última actualización del objeto en el Sistema es el tiempo actual en el que se ha recibido el evento.
- La *cualidad* inicial del objeto representada como un triple *(clase, valor, grado de creencia)* es:
  - **(***tipo***,** $evento\ audio$**,** $1{,}0$**)**. Esta cualidad indica que es un objeto creado a partir de la recepción de un evento de sonido.

- La *acción* asociada al objeto consiste en la información obtenida del nuevo evento de sonido detectado. El identificador i del evento de sonido constituye el identificador de la nueva acción. El grado de creencia de la misma será el grado de creencia g que indica el nivel de certeza de que el sonido detectado sea del tipo i. Los tiempos de inicio y fin de la acción coinciden con el tiempo actual en el que se ha recibido el evento.
- Como *localización* del objeto, solo rellenamos el parámetro referente a la posición donde se tiene creencia que ha podido ocurrir el evento. Uno de los datos facilitados por el análisis de audio es la posición del evento de sonido respecto al micrófono que lo ha detectado. Como el Sistema dispone de la posición del micrófono en el entorno, puede calcular la posición del evento.

  Los valores de velocidad, tamaño, e incremento de tiempo quedan inicializados a cero porque no son necesarios en este tipo de objetos.

Pongamos un ***ejemplo*** de Traducción si se recibe un evento de audio. Supongamos que el Sistema recibe en el instante "10:00 10/01/10" el siguiente evento de sonido:

- **i**:*'rotura de cristal'*.
- **m**: *'micro_1'*.
- **g**: *'0.88'*.
- **p**: *'(1,0,0.5)'*.

1. Si el Sistema no dispone de información asociada al micrófono $micro\_1$, se crea un nuevo objeto en la Base de Conocimiento que refleje la existencia de esa fuente de audio.
   - Identificador: $micro\_1$
   - Grado de creencia: 1.0
   - Lista de identificadores: $micro\_1$

- Tiempo de la última actualización: “10:00 10/01/10” .
- Cualidades: (*tipo*, *evento audio*, 1), (*clase*, *microfono*, 1), (*objeto*, *fijo*, 1).
- Acciones: (*rotura de cristal*,0.88,“10:00 10/01/10”)
- Localización:
  - Posición: (111.0,0,25.5)
  - Velocidad: (0,0,0)
  - Tamaño: (0,0)
  - Tiempo: 10:00 10/01/10
  - Incremento de tiempo: 0

2. Si el Sistema ya dispone de un Objeto que refleja al micrófono, ya que tienen el mismo identificador, simplemente añade a dicho Objeto la nueva acción que refleja la detección de un ruido en el entorno. Esta acción sería: (*rotura de cristal*,0,88,“10:00 10/01/10”)

3. Se crea en la Base de Conocimiento un nuevo objeto que refleja el ruido detectado. Este objeto sería:

   - Identificador: *rotura de cristal*
   - Grado de creencia: 1.0
   - Lista de identificadores: *rotura de cristal*
   - Tiempo de la última actualización: “10:00 10/01/10” .
   - Cualidades: (*tipo*, *evento audio*, 1).
   - Acciones: (*rotura de cristal*,0.88,“10:00 10/01/10”)
   - Localización:
     - Posición: (112.0,0,26.0)
     - Velocidad: (0,0,0)
     - Tamaño: (0,0)
     - Tiempo: 10:00 10/01/10
     - Incremento de tiempo: 0

Traductor para la fuente de Sensores.

Este componente será el encargado de recibir los eventos que detecten los sensores instalados en el entorno. Su función es integrar la información recibida en la Base de Conocimiento.

Como hemos visto en la sección anterior, el flujo de información que recibimos desde los sensores es contiene el identificador del sensor (i), la acción (x) que ha detectado el sensor (*activación* o *desactivación*) y el índice de creencia que indica el nivel de posibilidad (g) con el que ha sido detectada dicha acción.

El proceso a seguir por este Traductor es similar al del Traductor de la fuente de audio. Se siguen los siguientes pasos,

1. Primero, se consulta si existe un Objeto del Sistema que refleje al sensor.
    - En el caso de que no exista, se crea como un nuevo Objeto fijo del Sistema. Este nuevo objeto tendrá las siguientes características:
        - El *identificador del Objeto* es s, el identificador del sensor.
        - El *grado de actividad* o vigencia del objeto inicialmente es 1.
        - La lista de identificadores contiene solo el identificador s.
        - El *tiempo* de la última actualización del objeto en el Sistema es el tiempo actual en el que se ha recibido el evento.
        - Las *cualidades* iniciales del objeto representadas como un triple *(clase, valor, grado de creencia)* son:
            - **(**$tipo$**,** $evento\ sensor$**,** 1**)**. Esta cualidad indica que es un objeto creado a partir de la recepción de un evento de un sensor.
            - ($clase$, $sensor$, 1). Esta cualidad alude a la clasificación del objeto como sensor.
            - ($objeto$, $fijo$, 1). Esta cualidad representa que el objeto es fijo en el escenario, y por lo tanto no puede ser eliminado.

- La *acción* asociada al objeto consiste en la información obtenida del nuevo evento detectado. El identificador x, que indica el estado que el sensor ha detectado (activación o desactivación), constituye el identificador de la nueva acción. El grado de creencia de la misma será el grado de creencia g que indica el nivel de certeza con el que el sensor ha detectado el evento. Los tiempos de inicio y fin de la acción coinciden con el tiempo actual en el que se ha recibido el evento.
- Como *localización* del objeto, solo rellenamos el parámetro referente a la posición. Este dato será la posición que ocupa el sensor s dentro del entorno. Al definir el escenario, se describen los sensores que van a participar en el espacio monitorizado, indicado su identificador y su posición dentro del escenario. Dicha información puede ser consultada por el traductor en el momento de crear este Objeto.

  Los valores de velocidad, tamaño, e incremento de tiempo quedan inicializados a cero porque no son necesarios en este tipo de objetos.

- En el caso de que si exista un Objeto que refleje ha dicho sensor en el Sistema, simplemente se asocia una nueva acción a dicho Objeto que simbolice el evento de activación o desactivación que ha sido detectado. El identificador de la nueva acción se corresponde con el identificador x que indica el estado que el sensor ha detectado (activación o desactivación). El grado de creencia de la misma será el grado de creencia g que indica el nivel de certeza con el que el sensor ha detectado el evento. Los tiempos de inicio y fin de la acción coinciden con el tiempo actual en el que se ha recibido el evento.

2. Finalmente, se crea un nuevo objeto que simbolice el evento temporal del sensor. Este objeto se define de la siguiente manera:

   - El *identificador del Objeto* es s_x, consiste en la unión del identificador del sensor y el identificador del evento detectado.
   - El *grado de actividad* o vigencia del objeto inicialmente es 1.

- La lista de identificadores contiene solo el identificador s_x.
- El *tiempo* de la última actualización del objeto en el Sistema es el tiempo actual en el que se ha recibido el evento.
- La *cualidad* inicial del objeto representada como un triple *(clase, valor, grado de creencia)* es:
  - **(***tipo***,** *evento sensor***,** 1,0**)**. Esta cualidad indica que es un objeto creado a partir de la recepción de un evento de un sensor.
- La *acción* asociada al objeto consiste en la información obtenida del nuevo evento detectado. El identificador x que indica el estado que el sensor ha detectado (activación o desactivación), constituye el identificador de la nueva acción. El grado de creencia de la misma será el grado de creencia g que indica el nivel de certeza con el que el sensor ha detectado el evento. Los tiempos de inicio y fin de la acción coinciden con el tiempo actual en el que se ha recibido el evento.
- Como *localización* del objeto, solo rellenamos el parámetro referente a la posición. Este dato será la posición que ocupa el sensor s dentro del entorno. Al definir el escenario, se describen los sensores que van a participar en el espacio monitorizado, indicado su identificador y su posición dentro del escenario. Dicha información puede ser consultada por el traductor en el momento de crear este Objeto.

  Los valores de velocidad, tamaño, e incremento de tiempo quedan inicializados a cero porque no son necesarios en este tipo de objetos.

Ilustraremos con un ejemplo que indique cómo actúa este Traductor al recibir un evento procedente de un sensor. Su pongamos que el Sistema recibe en el instante “10:30 10/01/10” el siguiente evento de sonido:

- **s**: ‘sensorMovimiento1’
- **(x,g)**: *(activación,1.0)*

1. El Traductor comprueba si el Sistema dispone de información asociada al micrófono $sensorMovimiento1$. Si no es así, se crea un nuevo objeto en la Base de Conocimiento que refleje la existencia de esa fuente de sensorización.

   - Identificador: $sensorMovimiento1$
   - Grado de creencia: 1.0
   - Lista de identificadores: $sensorMovimiento1$
   - Tiempo de la última actualización: "10:30 10/01/10" .
   - Cualidades: ($tipo$, $evento\ sensor$, $1{,}0$), ($clase$, $sensor$, $1{,}0$), ($objeto$, $fijo$, 1).
   - Acciones: (*activación*,$1{,}0$,"10:00 10/01/10")
   - Localización:
     - Posición: (10.0,0,10.5)
     - Velocidad: (0,0,0)
     - Tamaño: (0,0)
     - Tiempo: 10:30 10/01/10
     - Incremento de tiempo: 0

2. Si el Sistema ya dispone de un Objeto que refleja al micrófono, ya que tienen el mismo identificador, simplemente añade a dicho Objeto la nueva acción que refleja la detección movimiento captada por el sensor. Esta acción sería: (*activación*,$1{,}0$,"10:00 10/01/10")

3. Además, el Traductor crea otro nuevo Objeto en la Base de Conocimiento refleja la activación detectada. Este objeto sería:

   - Identificador: $sensorMovimiento1\_activacion$
   - Grado de creencia: 1.0
   - Lista de identificadores: $sensorMovimiento1\_activacion$
   - Tiempo de la última actualización: "10:00 10/01/10" .
   - Cualidades: ($tipo$, $evento\ sensor$, 1).
   - Acciones: (*activación*,1.0,"10:30 10/01/10")
   - Localización:

- Posición: (10.0,0,10.5)
- Velocidad: (0,0,0)
- Tamaño: (0,0)
- Tiempo: 10:30 10/01/10
- Incremento de tiempo: 0

4.3.5. Módulo de detección de la alerta intrusión

Con el fin de analizar la presencia de intrusiones, hemos desarrollado un nuevo Módulo de Detección de Alertas que consiste en un *Sistema Experto Basado en Reglas Difusas*. Se trata de un Sistema cuya característica principal es que es *fácilmente configurable*, es decir, no tiene un conjunto de reglas fijas, sino que el conjunto de reglas se puede definir y ajustar fácilmente para cada escenario de estudio.

Este Sistema tiene una estructura general similar a los anteriores sistemas expertos difusos propuestos, donde diferenciamos tres partes importantes:

- El Fuzzificador, que será el encargado de fuzzificar aquellos datos que no sean difusos.
- El *Marco de Conocimiento*, el cual integra todos los datos relativos al Sistema. En este módulo distinguimos entre:
  - la información que se va a analizar, que en este caso coincide con el conjunto de objetos de la *Base de Conocimiento (BC)* del Sistema de Vigilancia,
  - la *Base de Reglas*, que contiene todas las reglas difusas que van a regir el Sistema.
- El *Motor de Inferencia*, que se encarga de extraer las conclusiones partiendo de los datos analizados y aplicando las reglas que rigen el Sistema. Una modificación del conocimiento conllevará una nueva evaluación de las reglas que puede dar como resultado unas conclusiones distintas.

En este caso, como veremos a continuación, solo se definirá un conjunto difuso "ser reciente". El resto de información borrosa que el sistema usará en su análisis ya es conocida en los datos de entrada. Dicha información está almacenada como cualidades o acciones de los objetos en la BC (ver sección 3.5).

Un ejemplo de variables difusas que considerará el Sistema son las referentes a la clasificación y a las acciones de los objetos. La clasificación de los objetos puede ser como vehículo, persona, otro y sensor, con un grado de creencia que indica la pertenencia a cada uno de los conjuntos. Las acciones están compuestas por las trayectorias que siguen los objetos, por la clasificación de sonidos en el entorno... también con un grado de incertidumbre asociado que indica la pertenencia a dichos conjuntos (identificadores de acción).

Base de Reglas

Las reglas definen acontecimientos que son considerados como intrusiones o bien circunstancias que constituyen parte de una intrusión. Es importante destacar que estas reglas, son evaluadas con los Objetos del Sistema que se encuentran actualmente en la Base de Conocimiento, representados bajo el esquema definido en la ORHC (ver sección 3.5).

En cada regla, se define qué tipo de objetos, al realizar una determinada acción, incrementarán el nivel de alerta con un grado de relevancia. Informalmente, una regla expresa:

IF existe un objeto de tipo A que realiza o ha realizado recientemente una acción X, THEN el nivel de alerta se incrementa en un grado α en función del grado de cumplimiento del antecedente

En este caso, al igual que con el sistema experto para la detección del peligro de atropello, nos encontramos ante la variante de reglas Takagi-Sugeno, propuestas en dicho sistema, donde el consecuente de la regla está en función del grado de cumplimiento del antecedente (que a su vez está en función de las entradas del sistema). Estas reglas son del tipo:

$IF\ x_0\ is\ A\ and\ x_1\ is\ B\ THEN\ F(w_0, w_1)$,

donde:

- w_0 es el grado de cumplimiento de $x_0\ is\ A$,
- w_1 es el grado de cumplimiento de $x_1\ is\ B$,
- y F(w_0,w_1)= $\alpha \cdot AND_{min}$(w_0,w_1).

La variable x_0 alude al tipo de objeto que se esta estudiando, y x_1 a la acción que se está evaluando. Al ser reglas estándar, el conjunto A puede ser $\{vehiculos,\ personas\ o\ sensores\}$ y el conjunto B es ser $reciente$ y $real$. En este último aspecto, es importante estudiar acciones *recientes* de las que se tenga una gran certeza de que el objeto las esté realizando (*reales*).

Vamos a definir un patrón más formal que debe seguir cada regla, desde el punto de vista del usuario que vaya ha configurar las reglas del sistema para conocer los parámetros de configuración de las mismas:

IF $\exists$ *obj* (*obj* es un **'Tipo_Objeto'** AND *obj* realiza la **'Acción_X'** teniendo en cuenta un valor de **'TiempoRef'**), THEN el nivel de alarma es incrementado en función del grado de cumplimiento del antecedente y de un valor de peso 'α'

Donde 'Tipo_Objeto', 'Accion_X', 'TiempoRef' y 'α' son cuatro variables que deben ser definidas para cada regla:

- **Tipo_Objeto**: es un parámetro que indica qué tipo de objetos son afectados por la regla. Por ejemplo, en nuestro caso pueden ser: vehículos, personas o sensores (teniendo en cuenta los micrófonos y otros sensores como "sensores"). Cuando este parámetro está vacío, se considera que la regla afecta a todos los tipos de objetos, ya sean vehículos, personas o sensores.
- **Acción_X**: es un parámetro que indica el identificador de la acción evaluada por la regla.

- **TiempoRef**: es un valor de tiempo de referencia, que se mide en milisegundos. Este valor es usado para obtener un grado de relevancia (entre 0 y 1) acerca de la importancia de la acción que se está evaluando para un objeto. Esta relevancia se obtiene en función del tiempo que haya transcurrido desde que finalizó la acción estudiada, tomando como referencia el valor 'TiempoRef'. Este aspecto será explicado con mayor detalle a continuación.

- α: es el parámetro introducido en el consecuente de la regla. Consiste en un valor que indica el factor de incremento del nivel de alerta. Cuando un objeto del Sistema es un 'Tipo_Objeto", y, además, está realizando o recientemente ha realizado la acción con identificador 'Acción_X', entonces el nivel de alerta se aumenta en función de este valor.

  Este dato pertenece al intervalo [-1,1] y permite otorgar a la regla una importancia. Si α es positivo, indica que la acción de la regla es peligrosa porque puede suponer una intrusión y se aumenta el nivel de alerta. Sin embargo, si es negativo, el nivel de alerta se decrementa. Este último caso correspondería a acciones que pueden suavizar una situación peligrosa.

De aquí en adelante, nos referimos a *"obj es un Tipo_Objeto"* como la primera condición de la regla o condición 1; y *"obj realiza la 'Acción_X' teniendo en cuenta un valor de 'TiempoRef'"*, como la segunda condición o condición 2. La segunda condición lo que hace es comprobar si la acción 'Acción_X' es reciente y se tiene gran certeza de su realización.

Motor de inferencia

La evaluación del Sistema basado en reglas, o lo que es lo mismo, el motor de inferencia, se lleva a cabo cada vez que un cambio en las acciones de los objetos de la Base de Conocimiento es detectado. Las reglas son evaluadas, con todos los objetos, una a una.

A continuación, vamos a explicar cómo una regla es evaluada por los Objetos de la Base de Conocimiento.

Evaluación de una regla.

Para cada regla, se estudian todos los objetos que existen en la Base de Conocimiento en el momento actual con el fin de encontrar aquellos que cumplen la regla y cambian el nivel de alerta.

Cuando una regla es evaluada por un objeto concreto, se comprueba que se cumplen las dos condiciones para dicho objeto (condición 1 $\wedge$ Condición 2):

1. **Condición 1**. En primer lugar, comprobamos que el objeto tiene el mismo tipo, que el tipo especificado en el parámetro 'Tipo_Objeto' de la condición 1 de la regla. Usaremos el parámetro $w0$ para referirnos al grado de creencia con el que se cumple la primera condición.

   El tipo de objeto se almacena como una cualidad *(c, v, g)* con el parámetro *c* igual a *'tipo'*, el parámetro *v* es igual a la *clasificación del objeto* (persona, vehículo o sensor) y el parámetro *g* es igual al *grado de creencia*. Este esquema se definió en la ORHC (ver sección 3.5). Es importante considerar que los objetos de la Base de Conocimiento pueden tener diferentes tipos o clasificaciones asociados. En este caso, cada clasificación viene reflejada en una cualidad diferente en el Objeto. Por ejemplo, las cualidades de un objeto con varios tipos asociados pueden ser: ("tipo", "vehículo", 0.2), ("tipo", "Persona", 0.8).

   El grado de creencia (w0), con el que se cumple la primera condición de la regla, es el grado de creencia que tiene el objeto para el tipo especificado en la regla; es decir, es el valor g de la cualidad, donde *c* es igual a *'tipo'* y *v* es igual al contenido de la variable **Tipo_Objeto**.

   Se consideran dos casos especiales: 1) Si el objeto evaluado no tiene ninguna cualidad que contemple el tipo especificado en esta condición, w0 es cero. 2) Si la variable **Tipo_Objeto** es vacía, simboliza que esta regla se cumple para todas las clasificaciones del objeto. En este caso, w0 será el valor de creencia g máximo entre las cualidades con la clase *c* igual a *'tipo'*.

En los casos en los que w0 sea 0, la primera condición no se cumple. Este hecho implica la no activación de la regla, por lo que esta deja de evaluarse para este objeto y pasa a ser evaluada con el siguiente objeto de la Base de Conocimiento.

2. **Condición 2**. Usaremos el parámetro $w1$ para referirnos al grado de creencia con el que se cumple la segunda condición.

   En este proceso, se estudian todas las acciones con el mismo identificador que el identificador de la acción especificada en la regla, ya que el objeto puede haber realizado varias veces una misma acción.

   La evaluación de esta condición consiste en comprobar si la acción especificada en la regla está siendo realizada o ha sido realizada recientemente por el objeto. Para ello, estudiamos dos factores: 1) el grado de creencia que indica la posibilidad de que el objeto haya realizado o esté realizando dicha acción y 2) cómo de reciente es la acción. El objetivo es estudiar acciones de las que tenemos una mayor certeza de que se están realizando y que sean recientes a la vez.

   Por consiguiente, para cada acción, se evalúa el cumplimiento de dos parámetros: la importancia que tiene la acción (wt) y el grado de creencia de la acción ($a.GradoDeCreencia$). Ambos pertenecen al intervalo [0,1]. El segundo de ellos se obtiene desde los datos del Objeto en la Base de Conocimiento. Sin embargo, wt necesita un proceso de cálculo.

   $$\text{Condición 2= } wt \wedge a.GradoDeCreencia$$

   Evidentemente, para la detección en tiempo real de una intrusión es más importante conocer si un objeto está realizando o hace poco que ha realizado una determinada acción conflictiva, que conocer que hace dos horas que realizó esa acción. Por esta razón, si la acción 'Acción_X' se está desarrollando actualmente por el objeto evaluado, entonces la importancia de dicha acción es 1. Sin embargo, si la acción ya está finalizada, porque ya había sido realizada antes por el objeto 'obj', entonces en este caso el grado

de importancia (wt) se obtiene en función del tiempo transcurrido desde que acabó la acción y un valor de referencia, el valor del parámetro 'TiempoRef' de la regla. En este último caso, se aplica una función (ver figura 4.37) para obtener un grado de creencia acerca de *"Cómo de importante es una acción en función del tiempo transcurrido desde que acabó"*, o en otras palabras, si la acción es reciente.

El valor 'TiempoRef' se utiliza en esta función como valor de referencia para indicar si la acción es reciente o no. Si el tiempo transcurrido desde que la acción acabó, supera el valor de referencia, se le considera que la acción no es reciente y la relevancia es 0. En cambio, si el tiempo transcurrido no supera dicho valor, el grado de importancia asociado a la acción varía entre 0 y 1 (otorgando un valor inversamente proporcional al tiempo transcurrido, a mayor tiempo menor importancia).

Una vez conocidos los valores $a.GradoDeCreencia$ y wt, se aplica el operador AND utilizando la función *mínimo*, evaluando así cada acción objeto (ver figura 4.38). De esta forma, obtenemos el valor w como el $min(a.GradoDeCreencia, Wt)$. Después de evaluar todas las acciones del objeto con el mismo identificador que la acción especificada en la regla, obtenemos un valor w para cada una. Finalmente nos quedamos con el valor máximo de todos los grados de creencia que será el valor final con el que la segunda condición es cumplida ($w2 = Max(w_{a1}...w_{an})$).

En el algoritmo 3 [3] puede verse el proceso descrito para evaluar la segunda condición.

Después de verificar que ambas condiciones se cumplen para un

[3] obj es el objeto de estudio; $Accion\_X$ es el identificador de la acción a evaluar y $TiempoRef$ es el valor de tiempo (en milisegundos) usado como referencia para obtener un grado de creencia sobre cómo de reciente es la acción. Otros parámetros: w_t es el grado de creencia que indica cómo de reciente es la acción evaluada para un determinado objeto. w es el grado de relevancia final de una acción en función de w_t y el último grado de creencia de la acción almacenado en la BC ($a.GradoDeCreencia$). $w2$ es el máximo grado de relevancia de todos los grados w obtenidos, simboliza el grado de cumplimiento de la condición dos. $TiempoTranscurrido$ es el tiempo que ha transcurrido desde que la acción $Accion\_X$ acabó

Algorithm 3 cumplimientoCondicion2 ($obj, Accion\_X, TiempoRef$)

Require: $obj, Accion\_X, TiempoRef$
  $w2 = 0$
  for ∀ a ∈ acciones(obj) **do**
    $w_t = 0$
    if $a.id = Action\_X$ **then**
      if $a.isFinished()$ **then**
        $TiempoTranscurrido = TiempoActual - a.TiempoFinAccion()$
        $w_t = esReciente(TiempoTranscurrido, TiempoRef)$(ver Fig.4.37)
      else
        $w_t = 1$
      end if
    end if
    w=AND_{min}(w_t,$a.GradoDeCreencia$)
    $w2$=OR_{max}(w,$w2$)
  end for
  return $w2$

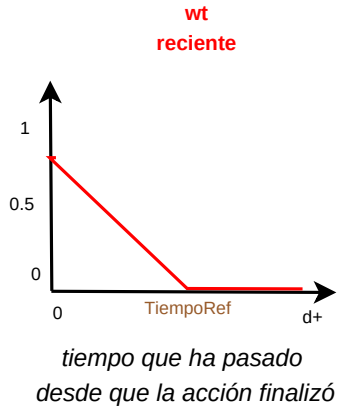


Figura 4.37: Función que expresa el grado de importancia de una acción en función del tiempo transcurrido desde que acabó de realizarse

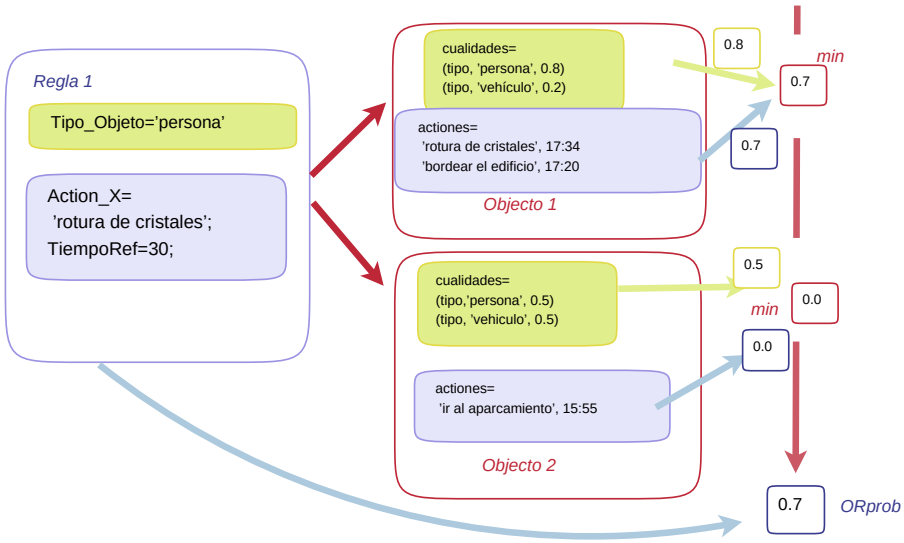


Figura 4.38: Evaluación de una regla con un objeto concreto

objeto 'obj' concreto, se aplica el operador AND utilizando la función *mínimo*. De esta forma, conocemos el grado de creencia asociado a la regla i cuando es evaluada para un objeto concreto (wRi_{obj}).

$$wRi_{obj} = Condicion_1 \text{ AND } Condicion_2 = min(w0, w1)$$

El fin es evaluar la regla con todos los objetos existentes en la escena, para conocer si al menos existe algún objeto que cumple las condiciones expuestas en el antecedente. De este modo, para obtener el valor final del cumplimiento de una regla (WR_i), tras realizar el matching con los distintos objetos, aplicamos el operador OR.

$$WR_i = EvaluacionRegla_i(obj1) \text{ OR } ... \text{ OR}$$
$$EvaluacionRegla_i(objk) = \ WRi_{obj1}\ OR\ ...OR\ WRi_{objk}$$

En este caso no es adecuado emplear la función *máximo*, ya que, por ejemplo, para la detección de una intrusión no es lo mismo que existan 3 personas merodeando un edificio protegido a que solo haya

una. Por ello, en este punto emplearemos un *OR probabilístico* ya que es acumulativo y permite incluir el tipo de situación anterior.

$$OR_{prob}(a,b) = SumaPromedio(a,b) = a + b - a \cdot b$$

Incremento del nivel de alerta.

El nivel de alerta final se obtiene tras evaluar todas las reglas con todos los objetos. En un sistema de m reglas el nivel obtenido es:

$Nivel\_Alerta$= WR_1 OR WR_2 OR ... OR WR_m

Todas las reglas cambian el nivel de alerta de forma independiente (ver figura 4.39). Como se ha descrito anteriormente, el motor de inferencia propuesto sigue el algoritmo 4. [4]

En este caso, también pretendemos que el nivel de alerta sea acumulativo. No es lo mismo que un coche merodee una zona restringida que, a parte de dicho evento, una persona rompa una ventana. En este ejemplo, hay dos eventos peligrosos y no solo uno, por lo que el nivel de peligro es mayor y deben influir los dos hechos (y no solo aquel del que tenemos más certeza, por ejemplo). Por consiguiente, también empleamos un *OR probabilístico*.

De este modo, conseguimos el siguiente efecto: el nivel de alerta tras evaluar la regla i-ésima es igual al nivel de alerta tras evaluar la regla anterior más las alteración que indica el consecuente de la regla i. Esta alteración lo que pretende es cambiar el nivel de alerta en función del grado de cumplimiento de las condiciones del antecedente y de un valor α_i, como se explicó anteriormente. De esta forma, tenemos que:

[4] $objetosBC$ es el conjunto de objetos existentes en la Base de Conocimiento en el momento actual y R es el conjunto de reglas del Sistema. Otros parámetros: $w0$ es el grado de creencia asociado con el cumplimiento de la primera condición de una regla cuando es evaluada para un objeto determinado, $w1$ es el grado de creencia asociado con el cumplimiento de la segunda condición de una regla cuando es evaluada para un objeto determinado, wRi_{obj} es el grado de creencia asociado con el cumplimiento de una regla i cuando es evaluada para un objeto determinado, wR_i es el máximo de los grados de creencia con los que la regla se cumple cuando es evaluada por diferentes objetos, $Nivel\_Alerta$ es el nivel de presencia de intrusión detectado

$$Nivel\_Alerta_{Ri} = Nivel\_Alerta_{Ri-1} + \text{alteración}$$
$$(WR_i, \alpha_i, Nivel\_Alerta_{Ri-1})$$

$$\text{alteración}$$
$$(WR_i, \alpha_i, Nivel\_Alerta_{Ri-1}) = (1 - Nivel\_Alerta_{i-1}) \cdot (WR_i \cdot \alpha_i)$$

que es lo mismo que:

$Nivel\_Alerta_{Ri} = Nivel\_Alerta_{Ri-1} + (WR_i \cdot \alpha_i) - Nivel\_Alerta_{Ri-1} \cdot (WR_i \cdot \alpha_i)$

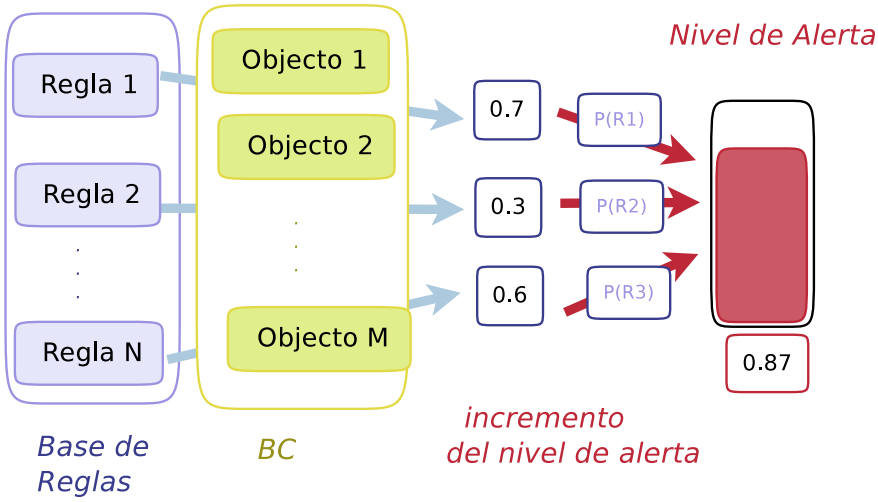


Figura 4.39: Evaluación de las reglas y actualización del nivel de alerta

El sistema experto disminuye de forma automática el nivel de alerta cuando la situación cambia, es decir, cuando las acciones asociadas a los objetos varían o dejan de hacerse. Este hecho es posible gracias al valor 'TiempoRef', que es un tiempo de referencia que permite otorgar o restar importancia a las acciones una vez que han sido realizadas. La duración de la alerta activa depende de este parámetro, el cual puede ser ajustado en función de las necesidades o preferencias.

Si se pretende que la activación de la alerta, debido a una

Algorithm 4 EvaluacionSistemaReglas ($objetosBC, R$)

Require: $objetosBC, R$
  $Nivel\_Alerta = 0$
  for ($from\ i = 1 \rightarrow\ i = numero(R)$) **do**
    $wR_i = 0$
    for $\forall obj \in objetosBC$ **do**
      $w0 = gradoCreenciaCumplimientoTipoDeObjeto(obj, r.Tipo\_Objeto)$
      $w1 = cumplimientoCondicion2(obj, r.The\_X\_action, TiempoRef)$
      (ver Algoritmo 3)
      wRi_{obj}=AND_{max}($w0$,$w1$)
      wR_i=OR_{prob}(wR_{i-1},wRi_{obj})
    end for
    $Nivel\_Alerta = Nivel\_Alerta + alteracion(wR_i, \alpha_i, Nivel\_Alerta)$
  end for

determinada acción, dure un tiempo considerado, se usará un valor más alto de 'TiempoRef', que si queremos que no dure tanto y que se vuelva actualizar cuando cese dicha acción. Un valor más alto de tiempo de referencia implica que una acción sigue siendo importante a pesar de que haya pasado más tiempo desde que se realizó. Cuando transcurre mucho tiempo (en base al tiempo de referencia usado) desde que termina una acción vigilada, esta acción deja de ser relevante. Esto significa que la segunda condición de la regla a penas se cumple o directamente no se cumple, y por lo tanto el valor de cumplimiento de la regla es muy bajo y no afecta el nivel de alerta.

Otro aspecto importante es que el parámetro 'α' puede ser positivo o negativo. Si es positivo, se aumenta el nivel de alerta. Sin embargo, si es negativo, el nivel de alerta se decrementa. Este último caso correspondería a acciones que pueden suavizar una situación peligrosa.

4.3.6. Plugins

Proponemos 3 *plugins* en este Sistema:

Eliminador de Objetos (Plugin EO)

Es el plugin básico que debe tener todo sistema que se base en el Modelo propuesto en esta tesis para el desarrollo de sistemas de vigilancia (véase capítulo 3). La función de este componente es eliminar aquellos Objetos que hayan salido de la escena. Su funcionamiento completo ha sido descrito en la sección 3.4.2 y complementado en la sección 4.1.8, por lo que en esta sección no volveremos a describirlo.

Visualización de la Base de Conocimiento (Plugin VBC)

Este plugin permite acceder a toda la información residente en la Base de Conocimiento en el momento actual y, representarla de forma legible por una persona. Este componente fue definido en la sección 4.1.8.

Relación de Eventos de Audio y Sensores con Objetos Móviles (Plugin REASOM)

Es importante dotar al Sistema de un procedimiento que permita asociar eventos de sensorización o de audio con posibles objetos detectados a partir del vídeo, para obtener un conocimiento a más alto nivel.

Lo que se pretende con este plugin es asociar a personas o a vehículos como posibles causantes de los distintos eventos de sonido o de sensores detectados. Por ejemplo, si se detecta una rotura de cristales, o un disparo, identificar (en el caso que sea posible) aquellos objetos dinámicos que puedan haber ocasionado dicho evento. Lo mismo ocurriría en el caso de que se active un sensor de movimiento.

De este modo, este plugin consiste en un proceso adicional que relaciona los eventos de sonido o los eventos de sensores a las personas o vehículos.

Cuando un sonido es detectado e identificado, o bien un sensor es activado, este plugin comprueba si existe algún objeto móvil en el Sistema que esté cercano a la posición donde el evento ha sido detectado. Este plugin se evalúa cada vez que se recibe un evento de audio o de otro sensor.

Para cada Objeto del Sistema detectado a partir de vídeo, se calcula la distancia $dist$ entre dicho Objeto y el evento auditivo o

de sensorización. Posteriormente, se comprueba si esa distancia $dist$ obtenida indica que el objeto y el evento se encuentran cerca.

Para ello, como hemos visto anteriormente, usamos una función que indica el grado de pertenencia (w) de $dist$ con respecto con el concepto "cerca" (ver figura 4.40). Nos basamos en una distancia de referencia d. Con esta función, se supone que si la distancia entre dos posiciones es menor que d/2, es que están cerca. En cambio, si la distancia fuera superior a la cantidad de d + d/2, probablemente no van a estar cerca. En los casos intermedios, irá disminuyendo la certeza de que están cerca.

Aquellos objetos donde se haya obtenido un $w > 0$, serán los posibles causantes del evento evaluado y w indicará el grado de posibilidad de que este hecho sea cierto. Para indicar este nuevo dato, se añade a dichos objetos una nueva $accion$ que simbolice que han realizado dicha acción. El identificador del evento (de sonido o de sensor), constituye el identificador de la nueva acción. El grado de creencia de la misma será el grado de creencia w obtenido que indica el nivel de creencia de que el objeto haya sido el causante del evento de sonido o se sensorización. Los tiempos de inicio y fin de la acción coinciden con el tiempo actual en el que se ha recibido el evento.

Por ejemplo, si se detecta la rotura de un cristal cuando es capturada por el micrófono, el Sistema analiza si hay personas o vehículos que están cerca de la posición de dicho evento. En caso afirmativo, (si el grado de cercanía es mayor que cero) estas personas o vehículos tendrán asociada una nueva acción, "la rotura de cristal" cuyo grado de certeza es el grado de cercanía obtenido respecto al evento.

4.3.7. Desarrollo del sistema

Este Sistema se ha implementado en JAVA, usando como middleware de comunicación ICE ZeroC [58]. Al igual que los sistemas anteriores, se han desarrollado dos herramientas: una herramienta que permite la adquisición de conocimiento del entorno compuesta por los distintos Traductores y otra herramienta para la monitorización de la *alerta intrusión*, que incluye una aplicación para dispositivos móviles.

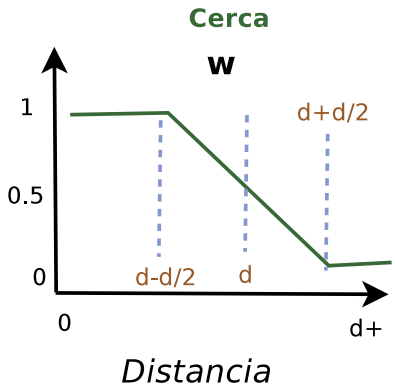


Figura 4.40: Función de pertenencia al conjunto difuso 'cerca'

Herramienta de adquisición de conocimiento sobre el entorno: Traductores

En esta aplicación, se han implementado los distintos Traductores propuestos en la sección 4.3.4. Esta herramienta permite definir el escenario de estudio y agregar de forma dinámica componentes *Traductores*, en función del número de cámaras, micrófonos o sensores.

En la figura 4.41 muestra la interfaz de usuario correspondiente a esta aplicación. Podemos apreciar que la pestaña seleccionada es la correspondiente a Traductor que recibe los eventos relacionados con las trayectorias a partir del análisis de vídeo.

Herramienta de monitorización de la Alerta Peligro de Intrusión

Esta otra herramienta permite observar el estado de la situación analizada en el escenario de estudio: la intrusión. Se trata de la misma herramienta desarrollada para los sistemas anteriores, aunque en este caso se han cargado los plugins específicos para este Sistema y el módulo de detección de la alerta intrusión (ver figura 4.42).

Mediante esta aplicación, el usuario experto puede confeccionar el conjunto de reglas específico que conlleve la detección de circunstancias

Figura 4.41: Traductor: Sistema de monitorización de intrusiones.

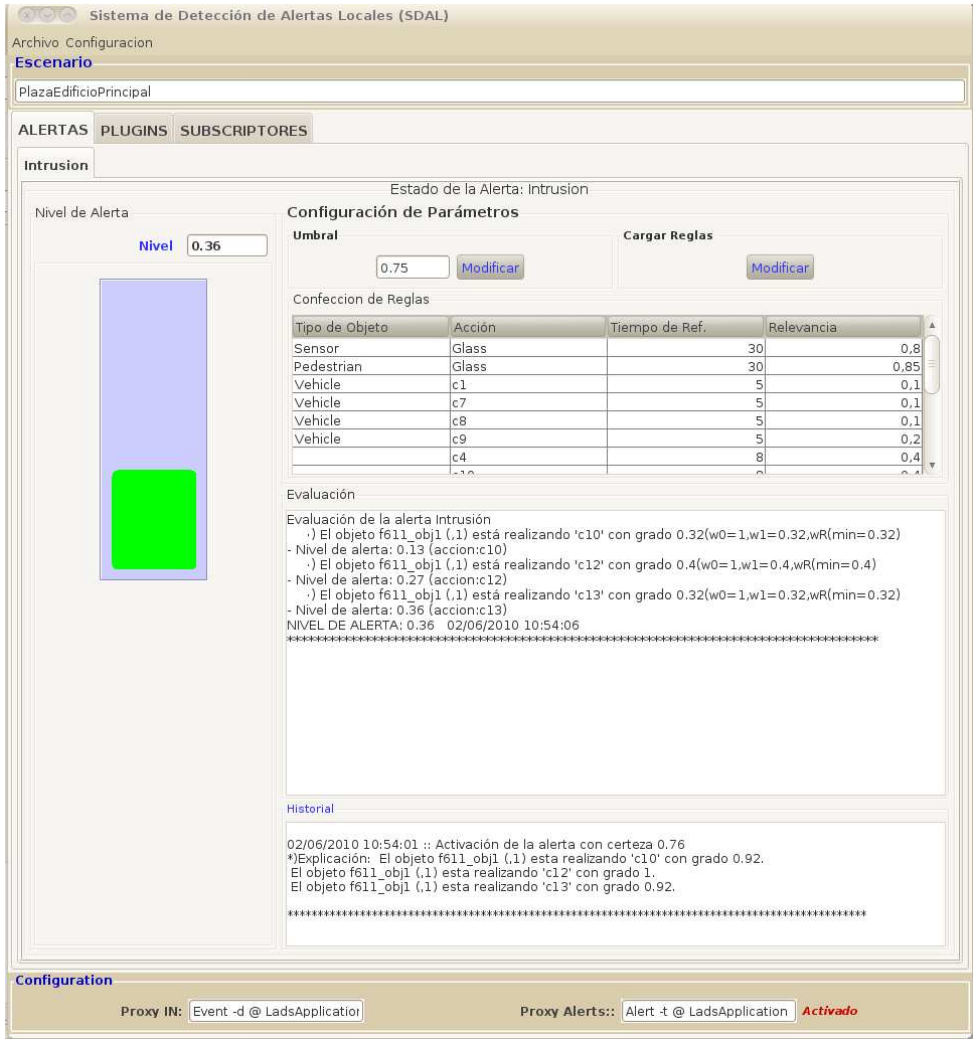


Figura 4.42: Aplicación de Escritorio: Sistema de monitorización de intrusiones.

que impliquen una intrusión.

El estado de la alerta viene reflejado por una barra gráfica que cambia de color y tamaño en función del nivel de peligro detectado y un umbral. Éste puede ser:

- *Estado Blanco: riesgo nulo,* cuando no hay ningún signo que indique riesgo de intrusiones.
- *Estado Verde: riesgo bajo,* cuando el nivel de peligro de intrusiones pertenece a (0, umbral/2].
- *Estado Amarillo: riesgo medio,* cuando el nivel de predicción de colisión pertenece a (umbral/2, umbral). Cuando el color es amarillo, el Sistema emite una alarma sonora que atrae la atención del operador.
- *Estado Rojo: riesgo alto,* cuando el nivel de peligro de intrusiones pertenece a [umbral,1]. En este caso, la alarma sonora emitida es más intensa.

A parte de la visualización gráfica, se muestran en modo texto los motivos que justifican el valor actual de la alerta. Además, el Sistema guarda un historial donde aparecen todas las veces que la alerta ha superado el umbral y los motivos que han influido en dicha activación.

Una de las grandes ventajas del Modelo propuesto es que genera automáticamente una explicación a través de las acciones que han cambiado la alerta de nivel. Así, el Sistema construye una descripción de la situación, comprensible para las personas.

Como se puede observar, al tener que basarse en un conjunto de reglas configurable, el Sistema puede ser portado fácilmente a cualquier entorno, donde se desee estudiar la presencia de posibles intrusiones.

Aplicación móvil

Este Sistema también incluye la aplicación móvil desarrollada para los sistemas anteriores (ver sección 4.1.9). De esta forma, se ofrece cierta movilidad al usuario. En la figura 4.43 podemos ver la interfaz de

usuario correspondiente a la aplicación móvil, donde se puede visualizar de forma gráfica el estado del nivel de alerta y leer los motivos de activación de la misma.

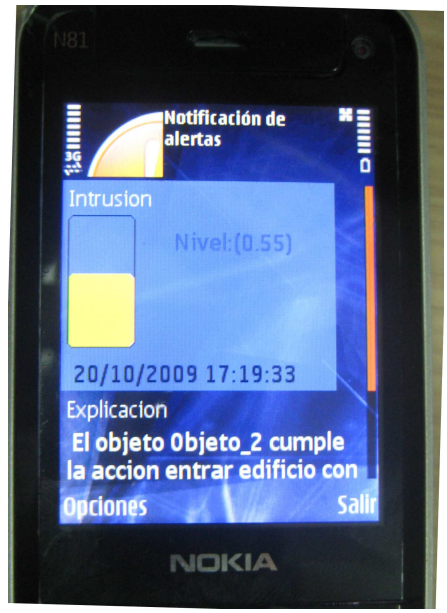


Figura 4.43: Aplicación Móvil: Sistema de Detección de Intrusiones

4.3.8. Fase experimental

El sistema experto para la detección de intrusiones ha sido probado en un escenario específico. Este escenario (ver figura 4.44)corresponde a la plaza central que da acceso al edificio principal de una sede empresarial.

En este caso, el objetivo es detectar intrusiones en el edificio principal durante un día festivo, donde el personal de la empresa no trabaja en dicho edificio. Otro de los edificios de la Sede es un museo y puede ser visitado en días festivos. Este hecho ocasiona el acceso de personas y vehículos al recinto. Las personas y vehículos pueden acceder al garaje y al museo y por lo tanto pasar por la calle principal, pero tienen el acceso restringido a la plaza central que da acceso al edificio principal,

Figura 4.44: Escenario de pruebas para la evaluación del Sistema Inteligente para la Detección de Intrusiones

escenario local que se pretende vigilar.

Suponemos que el escenario seleccionado para testear es controlado por una cámara, un micrófono y un sensor de movimiento, distribuidos estratégicamente en el entorno.

Para este estudio, se considera que una intrusión puede ser:

- Cualquier persona o vehículo que accede a la plaza central.
- El merodeo de cualquier persona que bordea el edificio principal.
- La identificación de movimiento en la puerta de entrada.
- La identificación de un arma de fuego o una rotura de cristales.

Las reglas usadas en este análisis y que describen esta situación se pueden ver en la tabla 4.3. Estas reglas permiten describir acciones con un mayor o menor grado de importancia, lo que se refleja en un mayor o menor riesgo de intrusión. Por ejemplo, no es lo mismo que un vehículo acceda a la plaza para aparcar por error, que si una persona se dirige a la parte trasera del edificio (donde no hay cámaras).

Hemos simulado 15 ejemplos de situaciones reales que pueden ocurrir en el escenario de estudio. Entre estos ejemplos, existen 12 intrusiones y 3 sin intrusiones. Para ello, hemos generado eventos de vídeo anotado, eventos de sonido y la activación del sensor de movimiento

Tabla 4.3: Conjunto de reglas estudiado

	'Tipo_Obj'	**'Accion_X'**	**'TiempoRef'**	α
R1	Persona	ir a la plaza	10	0.7
R2	Vehículo	ir a la plaza	10	0.6
R3	Persona	bordear edificio	30	0.8
R4	Sensor	rotura de cristal	30	0.8
R5	Persona	rotura de cristal	30	0.8
R6	Persona	ir a la puerta principal	20	0.85
R7		salir de la plaza	5	-0.80
R8	Sensor	activación movimiento	5	0.70

como entradas del Sistema. Las reglas que se han definido, han sido suficientes para detectar las intrusiones en la mayoría de los casos, lo que muestra un alto rendimiento del Sistema (ver Tabla 4.4). Como ocurre en la vida real, un gran número de datos conocidos hacen que la detección de la presencia de intrusiones sea más notable.

Hemos realizado otro estudio, referente a los tiempos de ejecución del Sistema en los ejemplos de prueba anteriores. El equipo utilizado es un Pentium (R) Dual-Core CPU E5200@2.50GHz 2.51GHz, 2,75 GB de RAM. En este caso, hemos llevado a cabo tres experimentos.

En primer lugar, se midió el tiempo que el middleware gasta desde que envía el evento hasta que éste es capturado por los Traductores del Sistema. El máximo tiempo transcurrido detectado es de 16 milisegundos. El promedio de tiempos es de 0,62 milisegundos, con una desviación estándar de 3.04 milisegundos. Con estos datos, conocemos que el middleware puede procesar hasta 62,5 eventos por segundo, sin causar retrasos.

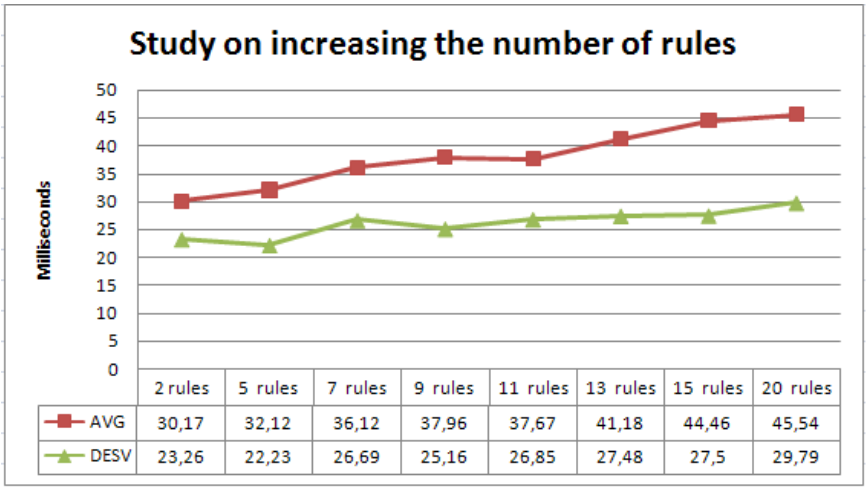
En segundo lugar, medimos los tiempos de procesamiento del Sistema, desde que un evento es recogido por los Traductores hasta que el Sistema obtiene el nivel de alerta generado por dicho evento. El tiempo máximo empleado en el procesamiento (traductores + MDA) es de 288 milisegundos. La media es de 36.11 milisegundos, con una desviación estándar de 28.60 milisegundos. Observamos que el tiempo empleado es

Tabla 4.4: Resultados obtenidos por el Sistema inteligente para la detección de intrusiones

Número de Ejemplo	Intrusión	Máximo nivel detectado
1	Si	0.8
2	Si	0.9
3	Si	1.0
4	Si	0.7
5	Si	0.83
6	Si	0.8
7	Si	0.76
8	Si	0.32
9	Si	1.0
10	Si	0.8
11	Si	0.8
12	Si	0.9
13	No	0.3
14	No	0.4
15	No	0.1

breve, lo que indica que el Sistema es bastante eficiente.

Por último, hemos llevado a cabo otro estudio sobre los tiempos obtenidos si el número de reglas varía. En este caso, solo se ha medido el tiempo empleado en el procesamiento del Módulo de Detección de la Alerta. Estos resultados se muestran en la figura 4.45. Podemos ver que el tiempo medio se incrementa con el aumento del número de reglas. Sin embargo, observamos que los datos obtenidos mediante el cambio en el número de reglas no son significativos. El hecho de añadir más reglas afecta en pocos milisegundos el tiempo de procesamiento del controlador difuso. Esto significa que el Sistema es escalable, ya que la introducción de nuevas reglas tiene un efecto leve sobre el tiempo de evaluación del Sistema.



	2 rules	5 rules	7 rules	9 rules	11 rules	13 rules	15 rules	20 rules
AVG	30,17	32,12	36,12	37,96	37,67	41,18	44,46	45,54
DESV	23,26	22,23	26,69	25,16	26,85	27,48	27,5	29,79

Figura 4.45: Estudio del Incremento del número de reglas en el módulo de detección de intrusiones

Capítulo 5

Conclusiones y Trabajo Futuro

El eje principal de esta tesis ha sido la definición de un Modelo que permite el desarrollo de sistemas de vigilancia escalables, flexibles y portables, los cuales pueden alimentarse de información procedente de fuentes heterogéneas. Este Modelo ha sido aplicado para la detección inteligente de tres situaciones de riesgo diferentes. De este modo, hemos conseguido abordar los objetivos que habíamos marcado en este estudio.

En este capítulo, exponemos las conclusiones obtenidas tras el trabajo realizado y los principales puntos que consideramos para continuar la línea de investigación que hemos dejado abierta.

5.1. Conclusiones

Los objetivos que marcamos al inicio de este trabajo, centrados en mejorar algunas de las carencias encontradas en el estado actual de los sistemas de vigilancia inteligentes, consistieron principalmente en:

1. **Diseñar una tecnología para el desarrollo de sistemas de vigilancia inteligentes** que se caractericen por ser escalables, flexibles y portables a cualquier entorno, que reciban información

ya procesada procedente del nivel sensorial y ofrezcan como salida la identificación de situaciones de interés. Incluyendo en dicha tecnología:

- El tratamiento e integración de la información heterogénea.
- La extracción de nuevo conocimiento.
- La robustez ante información imprecisa o borrosa.
- Razonamiento inteligente.

2. **Aplicar dicha propuesta para la creación de sistemas de seguridad inteligentes**, desarrollando sistemas de vigilancia que analicen distintas situaciones reales.

3. Realizar un **estudio experimental** de las diversas tecnologías desarrolladas.

En la memoria de esta tesis, podemos observar que los objetivos anteriores han sido alcanzados. El cumplimiento del primer objetivo puede verse en el capítulo 3, en el que **hemos presentado un Modelo para el desarrollo de Sistemas de Detección de Alertas en tiempo real**, que permite la integración de fuentes heterogéneas.

Este Modelo ha sido definido para cubrir las 4 características citadas anteriormente: integración de información heterogénea (usando la Ontología para la Representación Homogénea del Conocimiento), extracción de nuevo conocimiento (que tiene lugar en los Traductores, Plugins, y Modulos de Detección de Alertas), robustez ante información imprecisa (al hacer uso de técnicas formales de lógica difusa) y razonamiento inteligente (en los Módulos de Detección de Alertas). En realidad, estas características se cubren en su totalidad al aplicar el Modelo a un caso concreto de sistema de vigilancia.

En el capítulo 4, se han descrito tres aplicaciones del Modelo propuesto, abarcando el segundo objetivo y completando el primero. También en dicho capítulo se han expuesto los resultados obtenidos en la etapa experimental de los distintos sistemas desarrollados, cumpliendo el tercero de los objetivos.

A modo de resumen y conclusión, el Modelo propuesto en esta tesis, se compone principalmente de una *ontología* y una *arquitectura*. Los puntos fuertes de esta propuesta son:

- Cualquier tipo de información obtenida por diversas fuentes (audio, vídeo, y otros sensores) en un entorno vigilado puede ser representada bajo el mismo esquema conceptual definido en la *Ontología Homogénea para la Representación del Conocimiento* que hemos definido.

  De esta forma, la diversa *información heterogénea* de entrada (obtenida de diferentes análisis y bajo diferentes contextos) *es integrada de forma homogénea*. Este procedimiento se lleva a cabo en los Traductores.

  Gracias a esta integración de los datos, se consigue que el sistema estudie la información de forma transparente a su procedencia.

- La arquitectura definida se basa en capas y componentes, lo que permite:

  - Desarrollar aplicaciones de forma *eficiente*, ya que pueden realizarse desarrollos paralelos, en las distintas capas y componentes.
  - Crear aplicaciones más *robustas* debido al encapsulamiento, tanto en las capas como en los componentes.
  - Ofrecer un *mantenimiento y un soporte sencillo*, ya que es más fácil cambiar un componente que modificar una aplicación monolítica.
  - Tener mayor *flexibilidad*, porque se pueden añadir nuevos módulos para incluir nuevas funcionalidades.
  - Alta *escalabilidad*.
  - *Agilizar* el estudio inteligente de la información, debido a que el razonamiento está dividido en módulos que realizan análisis paralelos e independientes.

- *Reutilización* de los componentes, ya que son unidades independientes de desarrollo y tienen múltiples usos en diferentes sistemas de seguridad.

Este Modelo se caracteriza por ser abstracto. La Ontología ha sido descrita formalmente de forma concreta. En cambio, la arquitectura se ha definido de forma general y, sólo algunos de sus componentes, de forma específica. Esto se debe a que se trata de una arquitectura dinámica y adaptable a cada problema. Cada situación de alerta o cada entorno de monitorización exigirá unos determinados módulos de razonamiento o de extracción de conocimiento. Por ello, *este Modelo se ha definido de forma abierta para ser concretado y especificado en cada aplicación*.

El diseño definido permite que cada situación de estudio pueda ser resuelta de forma diferente. La integración de una nueva alerta puede conllevar una técnica distinta (una red bayesiana, un modelo de Markov . . .). Según las características de cada situación de estudio, se empleará la técnica más adecuada para su análisis.

El Modelo propuesto se ha aplicado en el desarrollo de tres aplicaciones, en las que se estudian tres tipos de alerta, de interés social. Se trata de situaciones reales y difíciles de detectar de forma automática. Las tres aplicaciones desarrolladas son:

1. **Sistema de vigilancia inteligente para la detección del peligro de atropello.**

   Hemos desarrollado un sistema de vigilancia inteligente capaz de identificar la presencia de peligro de atropello. Esta aplicación detecta peatones que puedan estar en riesgo porque un vehículo pueda atropellarlos. De esta manera, se puede predecir un atropello a tiempo y evitar una fatalidad. Este sistema ha sido publicado en [133].

   Para ello, hemos diseñado un modelo abstracto que permite avisar sobre posibles colisiones futuras entre dos objetos cualesquiera. Este modelo es un controlador difuso y ha sido adaptado para detectar colisiones peatón-vehículo.

Gran parte del peso de esta aplicación reside principalmente en un conjunto de procesos matemáticos y geométricos complejos que calculan características de los objetos en el mundo real, como el posicionamiento 3D, la velocidad, las trayectorias de los objetos, el tiempo estimado para que pueda tener lugar una colisión vehículo-peatón, las distancias entre distintas posiciones... Además, establecemos cálculos para minimizar los errores locales.

Una vez obtenidas las distintas variables ya suavizadas, todo este nuevo conocimiento es utilizado para resolver de forma sencilla esta situación tan compleja, empleando un controlador difuso, que usa una variante de reglas del modelo Takagi-Sugeno. En este caso, la salida de cada regla está en función del grado de cumplimiento del antecedente, y a su vez está en función de las entradas del sistema. Actualmente, el controlador consta de una sola regla, ya que esta es apta para la detección del peligro estudiado. Gracias a la elección de un SBRD, en trabajos futuros se podrá ampliar el conocimiento con nuevas reglas fácilmente.

Este sistema es una buena herramienta para la mejora de la seguridad de las personas en las zonas de tráfico. Actualmente, se muestra la alarma de forma visual y sonora en un equipo. Sin embargo, a la hora de llevar esta aplicación a la práctica, se estudiaría la manera de situar la alarma en el mismo lugar para avisar a los peatones y vehículos que están involucrados.

Los resultados de la etapa experimental demuestran una alta fiabilidad de las respuestas del Sistema. El número y el tipo de falsos negativos y falsos positivos es insignificante. Esto es posible a que el sistema indica el nivel de presencia de peligro de forma gradual (ofreciendo un valor entre 0 y 1). Los dos casos en los que se sobre-estimó la presencia de riesgo de atropello, no hubo ningún tipo de alarma sonora. Y en los dos casos en los que se infra-estimó el nivel de alerta, se emitió una alarma sonora que avisó del peligro, lo que no es tan grave, ya que el sistema avisó de un riesgo (aunque no lo clasificase como alto).

Con respecto a los tiempos de procesamiento, hemos de enfatizar que el sistema es eficiente y analiza la información ofreciendo

resultados en tiempo real, incluso adelantándose a la colisión. En el peor de los casos, con 5 objetos presentes en la escena, el sistema emplea un tiempo de procesamiento final (tiempo de Traductor + tiempo de MDA) inferior a 200 milisegundos, despreciable para el ojo humano.

Podemos observar que el Módulo de Detección de la Alerta emplea menos tiempo en procesarse que el Traductor, ya que éste realiza procesos algo más costosas en tiempos, como el cálculo del posicionamiento 3D, de las velocidades y la aplicación del algoritmo de seguimiento de alto nivel.

También hemos comprobado que, como era de esperar, un mayor número de objetos implica un mayor tiempo de procesamiento, pero esto no es un problema, ya que con 50 objetos en la Base de Conocimiento, el Sistema emplea un tiempo para el análisis de la alerta de 66 milisegundos, tiempo insignificante.

2. **Sistema de vigilancia inteligente para la detección de peligro por niños en zona de tráfico**

Se ha construido una herramienta de vigilancia inteligente que identifica el peligro por niños que no estén acompañados de adultos en una zona donde pueden transitar vehículos. De este modo, se podría avisar tanto a niños como a conductores para que tomen precaución y así evitar incidencias.

El módulo para la detección de esta alerta consiste en un sistema basado en reglas difusas. Las reglas permiten describir semánticamente la situación a detectar y facilitan la resolución del problema en cuestión.

Los tiempos medios obtenidos, en la etapa experimental, muestran que el Sistema es eficiente. Al igual que el sistema anterior, el Traductor emplea más tiempo de procesamiento, ya que su carga computacional es más grande. Aún así, estamos hablando de pocas centenas de milisegundos. El tiempo máximo final empleado por el Sistema no llega a ser ni medio segundo, en el peor de los casos, por lo que podemos afirmar que los resultados se generan en tiempo real y de forma eficiente.

3. **Sistema de vigilancia inteligente para la detección de intrusiones.**

   Hemos creado un Sistema inteligente para la detección de intrusiones en un escenario específico y vigilado. Una alerta de intrusión puede evitar, en muchos casos, robos, agresiones (tanto a los edificios como a las personas), ...

   Al igual que los anteriores, se trata de un controlador difuso. Sin embargo, en este caso, las reglas no son fijas, sino que son fácilmente configurables por un experto sin necesidad de reprogramar el Sistema.

   La herramienta implementada permite la definición de las reglas desde la interfaz de usuario. Este hecho permite modificar el concepto de intrusión, en función de las circunstancias que se desean detectar, o bien, si cambia el escenario de estudio.

   De este modo, para cada escenario, se definen un conjunto de reglas diferentes que indiquen cuáles son las circunstancias que suponen una intrusión. Incluso, dentro de un mismo escenario, se puede analizar un conjunto de reglas diferente en función del día.

   En esta aplicación, al existir tres traductores, consideramos interesante medir el tiempo que el middleware consume desde que envía el evento hasta que éste es capturado por los Traductores del Sistema. El promedio de tiempos obtenido es de de 0,62 milisegundos, con lo que podemos afirmar que el middleware puede procesar hasta 62,5 eventos por segundo, sin causar retrasos.

   Con respecto a los tiempos de procesamiento, en el peor de los casos, el tiempo máximo empleado por el Sistema es de 288 milisegundos, lo que indica que el Sistema es bastante eficiente.

   El tiempo medio se incrementa con el aumento del número de reglas. Sin embargo, en los resultados experimentales, observamos que los datos obtenidos mediante el cambio en el número de reglas no son significativos. El hecho de añadir más reglas afecta en pocos milisegundos el tiempo de procesamiento del Sistema. Esto significa que el Sistema es escalable, ya que la introducción de nuevas reglas tiene un efecto leve sobre el tiempo de evaluación

del Sistema.

El Modelo general propuesto permite crear *sistemas que se alimenten de fuentes heterogéneas*. Un ejemplo de ello es la aplicación para la detección de intrusiones, la cual se alimenta de información proveniente de análisis de vídeo, audio y otros sensores (de movimiento). El punto fuerte de este sistema es la unión de análisis de vídeo, audio y sensores. En este caso, se lleva a cabo un proceso de integración de la información y se construye un sistema de vigilancia más potente, ya que dispone de más información en la etapa de toma de decisiones.

Nuestra propuesta se ha centrado en avanzar en aspectos de integración de información heterogénea y análisis de situaciones de alerta, centrándonos más en la etapa de razonamiento sobre los acontecimientos de una escena. Para ello, en las distintas aplicaciones, hemos empleado sistemas de extracción de conocimiento, donde se realiza un análisis cognitivo preliminar de señales de audio, vídeo u otros sensores.

Uno de los puntos débiles que encontramos es que la calidad de las salidas de los Sistemas de Detección de Alertas está condicionada por la precisión y calidad de la información de los sistemas de extracción de conocimiento de audio, vídeo y sensores que se empleen como entrada. Una mala detección, clasificación o seguimiento de objetos, introduce ruido en el sistema final, ya que no muestra la verdadera realidad de la escena. Estos hechos podrían provocar la detección de falsas alertas.

Basándonos en esta idea, nos propusimos que nuestros sistemas sean robustos ante análisis de fuentes reales que usemos como entrada. Como hemos señalado anteriormente, en las distintas herramientas hemos partido de los datos obtenidos de sistemas de extracción de conocimiento existentes en la literatura. De esta forma, conseguimos hacer uso de conocimiento experto, haciendo que los sistemas desarrollados se basen en resultados de trabajos reales, y sean robustos ante este tipo de información.

El Modelo posibilita que los sistemas estén dotados de componentes que procesen la información de entrada, generando nuevo conocimiento y enriqueciendo la inteligencia de los sistemas. Este es el

caso de los Traductores y los Plugins.

Por ejemplo, al integrar los resultados de análisis de vídeo, el Traductor específico para la fuente procesa dicha información, generando *nuevo conocimiento*:

- *Se calcula la posición de real de los Objetos en el mundo (posicionamiento 3D)*. Para ello, se ha diseñado un proceso matemático y geométrico, que se basa en la calibración de la cámara.
- Obtenemos datos acerca de las *velocidades de los Objetos*. Por consiguiente, podemos conocer la posible posición futura de un Objeto dentro de un instante de tiempo 't'.
- *Se ha desarrollado un algoritmo de Seguimiento más potente y robusto*, basado en la clasificación de los objetos y en el posicionamiento 3D de los mismos. El algoritmo recoge situaciones que el seguimiento 2D no puede abarcar. Además, este método gestiona de forma adecuada la información que se recibe del análisis de vídeo, manteniendo la coherencia y evitando réplicas de objetos dentro de un escenario. Este método fue creado para robustecer al sistema ante los datos de entrada, ya que no siempre se recibe la información que se espera.

Además, hemos desarrollado Plugins que permiten extraer nuevo conocimiento, por ejemplo, obtener: una aproximación del tamaño real de un objeto, la posibilidad de que un objeto sea un niño, la posibilidad de que un objeto se mueva rápido, relacionar sonidos con otros objetos que puedan haber sido los causantes de dicho evento, ...

Otro de los puntos fuertes de los sistemas es la tolerancia a la *visualización con múltiples cámaras*. Los algoritmos diseñados permiten trabajar con diferentes vistas de un escenario concreto, integrando los datos de las diferentes fuentes de vídeo. Para ello, es necesario que éstas hayan sido calibradas usando el mismo eje de referencia sobre la escena.

Todas las aplicaciones, al estar basadas en el Modelo de desarrollo de sistemas de detección de alertas propuesto en este estudio, presentan las mismas cualidades enumeradas anteriormente (al tener una arqui-

tectura basada en capas y componentes). Destaca el hecho de ser *escalables y flexibles*:

- Si se quiere que el sistema se alimente de una nueva fuente de información, se añade un nuevo componente *Traductor* para integrar el conocimiento obtenido desde esa fuente.
- Si es necesario calcular nueva información del entorno acerca de los objetos, sólo hay que desarrollar y añadir un nuevo *Plugin* que analice y obtenga dicho conocimiento.
- Si se requiere analizar una nueva situación de riesgo, basta con añadir otro Módulo de Detección de Alertas para estudiar dicha situación.
- Además, se pueden reutilizar componentes entre las distintas aplicaciones.

Gracias a su diseño general, y al de cada uno de sus módulos, las tres aplicaciones desarrolladas son **fácilmente portables** a cualquier escenario de estudio. Para ello, sólo habría que ajustar los parámetros de configuración de cada sistema.

Los sistemas están diseñados para que sean **robustos ante información imprecisa o con vaguedad**. La lógica difusa ha jugado un papel fundamental en los módulos expertos de los sistemas. La definición de conjuntos difusos ha permitido representar el conocimiento experto con una gran expresividad. Este hecho ha dotado de interpretabilidad al sistema y se ha facilitado la etapa de toma de decisiones.

La Ontología propuesta recoge la definición de conceptos difusos y permite una **correcta representación del conocimiento analizado** para así obtener una buena interpretabilidad de los resultados. Cada sistema informa, en todo momento, y en tiempo real, del estado de su respectiva situación de estudio.

La correcta representación del conocimiento permite generar, de forma automática, explicaciones (comprensibles por el usuario) sobre los distintos acontecimientos analizados en la escena, que generan una

alteración del nivel de alerta. Consecuentemente, se proporciona una buena comprensión de la información obtenida tras analizar las distintas situaciones de riesgo. La explicación se podrá obtener tanto en formato de texto, como en formato de una lista de objetos con las acciones y cualidades que han participado en la alteración de la alerta.

También hemos desarrollado una aplicación ad hoc dispositivos móviles que puede subscribirse a cualquier alerta. De esta forma, le otorgamos **movilidad al sistema**, ya que el aviso de las alertas se puede realizar en tiempo real mediante dispositivos móviles. En este caso, cualquier operador puede ser informado, sin necesidad de estar permanentemente frente a la aplicación. Este hecho es bastante novedoso, ya que pocos sistemas de vigilancia lo incluyen.

Existe **un notificador de alertas dependiente de contexto y en tiempo real**. Este componente se encarga de informar mediante el middleware a todo cliente (móvil o no) interesado sobre el estado de las distintas situaciones de alerta estudiadas. De esta manera, hacemos que los sistemas publiquen sus resultados con el fin de que puedan ser empleados en una etapa posterior de vigilancia, por ejemplo, para la resolución de una crisis detectada.

Una de las ventajas, es que la suscripción de un cliente a una alerta, se realiza en función de sus características. Cada cliente indica cada cuánto tiempo quiere ser informado y a partir de qué umbral de alerta. Otra ventaja importante que presenta esta transmisión dependiente de contexto es la *reducción del tráfico en la red*.

Los resultados obtenidos en las etapas experimentales de cada sistema nos demuestran que son **aplicaciones eficientes**, cuyos tiempos de respuesta son casi despreciables. Este factor es fundamental en aplicaciones de tiempo real. Por otro lado, en los tests, también se ha demostrado una **alta fiabilidad** de las respuestas de cada sistema ante el análisis de situaciones reales.

Para mantener la fiabilidad y la eficiencia demostrada en las aplicaciones propuestas, es necesario contar con una fuente de información fiable y eficiente, que no conlleve retardos considerables.

5.2. Trabajo Futuro

Los objetivos marcados en esta tesis han sido alcanzados. Sin embargo, es un trabajo que no ha acabado y aún queda bastante por avanzar. Este estudio se continuará con distintos trabajos futuros:

- Queremos fortalecer las capacidades de las aplicaciones desarrolladas, por ejemplo:
  - Ampliar el sistema de detección de peligro de atropello, para detectar también otros peligros en zonas de tráfico, concretamente la detección de una nueva alerta: el peligro de choque entre vehículos. También, introduciremos nueva información acerca de la sensorización de los semáforos, para aquellos escenarios que los incluyan, con el fin de disponer de un mayor conocimiento de la escena que pueda influir en una colisión.
  - Robustecer los controladores difusos propuestos, ampliando el espacio de reglas, para complementar el análisis realizado en el caso que sea posible.
  - Adaptar los sistemas para su integración en dispositivos empotrados, con el objetivo de disponer de pequeñas máquinas capaces de realizar los cálculos en el ambiente. De esta forma, conseguiríamos llevar los sistemas al escenario vigilado con el fin de alertar en tiempo real a los individuos involucrados en el tipo de riesgo estudiado.
  - Dotar a los sistemas con procesos de aprendizaje, logrando herramientas adaptativas al ambiente, que puedan aprender de los eventos que tiene lugar en el escenario de estudio.
- Desarrollar nuevos Sistemas de Detección de Alertas, integrando nuevas fuentes de conocimiento y definiendo nuevas situaciones de interés o riesgo.

Bibliografía

[1] *5th International Symposium on Visual Computing (ISCV09)*, November 30-December 2 2009.

[2] 13th IEEE Conference on Image Processing (ICIP07). September 2007. San Antonio. Texas. USA.

[3] 14th IEEE Conference on Image Processing (ICIP08). October 10-12, 2008. san diego. california. usa.

[4] 2th ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC08). September 7-11. 2008. Stanford University. USA.

[5] A. Amato; V. D. Lecce and V. Piuri. Neural Network Based Video Surveillance System. In *CIHSPS 2005. IEEE International Conference on Computational Intelligence for Homeland Security and Personal Safety*, pages 85–89, 2005.

[6] A. Teschioni and C. S. Regazzoni. Performance evaluation strategies of an image processing system for surveillance applications. *in Advanced Video–Based Surveillance Systems, C. S. Regazzoni, G. Fabri, and G. Vernazza, Eds. Norwell, MA: Kluwer*, 2:76–90, 1999.

[7] J. Albusac, D. Vallejo, L. Jimenez-Linares, J. Castro-Sanchez, and L. Rodriguez-Benitez. Intelligent surveillance based on normality analysis to detect abnormal behaviour. *in International Journal of Pattern Recognition and Artificial Intelligence. Special issue on*

Visual Analysis and Understanding for Surveillance Applications, 23(7):1223–1244, 2009.

[8] J.A. Albusac. *Modelo para el Análisis de la Normalidad de Eventos y Conductas en Entornos Monitorizados: Aplicación a la Vídeo Vigilancia*. PhD thesis, Universidad de Castilla La Mancha, 2009.

[9] L. Alvarez, F. Guichard, P.L. Lions, and J.M. Morel. Axioms and fundamental equations of image processing. *Arch. Rat. Mech. and Anal. 16, IX*, pages 200–257, 1993.

[10] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for on–line non–linear/non–gaussian bayesian tracking. *IEEE Trans. on Signal Process.*, 50(2):174–188, 2002.

[11] P. K. Atrey, M. S. Kankanhalli, and R. Jain. Information assimilation framework for event detection in multimedia surveillance systems. *Multimedia Systems*, 13(3):239–252, December 2006. 10.1007/s00530-006-0063-8.

[12] P. K. Atrey, N. C. Maddage, and M. S. Kankanhalli. Audio based event detection for multimedia surveillance. *ICASSP06*, 2006.

[13] J.L. Barron, D.J. Fleet, and S.S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12:43–77, 1994.

[14] J.L. Barron, D.J. Fleet, and S.S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–47, 1994.

[15] A. Behrad, A. Shahrokni, and M. Ahmad. A robust vision-based moving target detection and tracking system. *In Proceeding of Image and Vision Computing Conference*, 2001.

[16] D. Beymer, P. McLauchlan, B. Coifman, and J. Malik. A real-time computer vision system for measuring traffic parameters. *Conf. on Computer Vision and Pattern Recognition, IEEE Computer Society*, pages 495–501, 1997.

[17] D. Bloisi, L. Iocchi, P. Remagnino, and N.D. Monekoso. Argos - a video surveillance system for a boat traffic monitoring in venice. *To appear in International Journal of Pattern Recognition and Artificial Intelligence*, 2009.

[18] L. Bo, C. Qimei, and G. Fan. Freeway auto-surveillance from traffic video. *In 6th International Conference on ITS Telecommunications Proceedings*, pages 167–170, 2006.

[19] Li Bo and Chen Qi-mei. Framework for freeway auto-surveillance from traffic video. *WRI World Congress on Computer Science and Information Engineering*, 6:360–365, 2009.

[20] M. Bogaert, N. Chleq, P. Cornez, C. S. Regazzoni, A. Teschioni, and M.Thonnat. The password project. In *IEEE International Conference on Image Processing*, pages 675–678. Swizerland, 1996.

[21] M. Borg, D. Thirde, J. Ferryman, F. Fusier, V. Valentin, F. Bremond, M. Thonnat, O. Team, and I. Sophia-Antipolis. Video surveillance for aircraft activity monitoring. *Proc. of IEEE Conf. on AVSS*, 1(16):16–21, 2005.

[22] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal. *A System of Patterns: Pattern-Oriented Software Architecture*, 1, 1996.

[23] H. Buxton. Learning and understanding dynamic scene activity: a review. *Image and Vision Computing*, 21(1):125–136, 2003.

[24] C. Micheloni and G.L. Foresti and L. Snidaro. A co–operative multicamera system for video–surveillance of parking lots. In *Intelligent Distributed Surveillance Systems Symp. by the IEE*, pages 21–24. London, 2003.

[25] C. Motamed. Motion detection and tracking using belief indicators for an automatic visual–surveillance system. *Image and Vision Computing*, 24(3):1192–1201, 2006.

[26] C. Sacchi and C. S. Regazzoni. A Distributed Surveillance System for Detection of Abandoned Objects in Unmanned Railway

Environments. *IEEE Transactions on vehicular technology*, 49(5):2013–2026, September 2000.

[27] C. Sacchi and C. S. Regazzoni and G. Vernazza. A Neural Network–Based Image Processing System for Detection of Vandal Acts in Unmanned Railway Environments. *IEEE*, ():529–534, 2001.

[28] C. Stauffer. Automatic hierarchical classification using time-based co-occurrences. *In Proceeding of IEEE Conf. Computer Vision and Pattern Recognition*, 2:335–339, 1999.

[29] Bao Rong Chang, Hsiu Fen Tsai, and Chung-Ping Young. Intelligent data fusion system for predicting vehicle collision warning using vision/gps sensing. *Expert Systems with Applications*, 37(3):2439 – 2450, 2010.

[30] Siu-Yeung Cho, Chai Quek, Shao-Xiong Seah, and Chin-Hui Chong. Hebbr2-taffic: A novel application of neuro-fuzzy network for visual based traffic monitoring system. *Expert Systems with Applications*, 36(3, Part 2):6343 – 6356, 2009.

[31] C. Clavel, T. Ehrette, and G. Richard. Events detection for an audio-based surveillance system. *Multimedia and Expo, IEEE International Conference on*, 0:1306–1309, 2005.

[32] R.T. Collins, A.J. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, D. Tolliver, N. Enomoto, O. Hasegawa, P. Burt, and L. Wixson. A system for video surveillance and monitoring. *Technical report, CMU-RI-TR-00-12, Carnegie Mellon University*, 2000.

[33] IEEE conference on Advanced Video and Signal Based Surveillance. July 2003.

[34] D. Corral. View: Computer vision for surveillance applications. *Colloquium Active and Passive Techniques for 3D Vision, IEE*, 8(1-3):192–204, 1991.

[35] M. Cristani, M. Bicego, and V. Murino. On–line adaptive background modelling for audio surveillance. *Pattern Recognition, International Conference on*, 2:399–402, 2004.

[36] R. Cucchiara, A. Prati, and R. Vezzani. A multi–camera vision system for fall detection and alarm generation. *Expert Systems*, 24(5):334–345, November 2007.

[37] D. Abrams and S. McDowall. Video Content Analysis with Effective Response. *IEEE*, ( ):57–63, 2007.

[38] D. L. Hall and J. Llinas. An introduction to multisensor data fusion. In *Proceedings of the IEEE*, volume 85, pages 6–23, January 1997.

[39] D. Tao and X. Li and X. Wu and S. Maybank. ELAPSED TIME IN HUMAN GAIT RECOGNITION: A NEW APPROACH. *General Tensor Discriminant Analysis and Gabor Features for Gait Recognition*, 29(10):1700–1715, 2007.

[40] D. Thirde and M. Borg and J. Aguilera and J. Ferryman and K. Baker and M Kampel. Evaluation of motion segmentation quality for Aircraft Activity Surveillance. In *In IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 293–300, 2005.

[41] D. Xu and S. Yan and D. Tao and L. Zhang, X. Li and H.J. Zhang. Human gait recognition with matrix representation. *IEEE Transactions on Circuits and Systems for Video Technology*, 16(7):896–903, 2006.

[42] L.S. Davis, I. Haritaoglu, and D. Harwood. w^4: Real-time surveillance of people and their activities. *Proc. 2000 Int. Conf. Pattern Recognition*, pages 809–830, 2000.

[43] J.P. Deparis, S.A. Velastin, and A.C. Davies. Cromatica project: A collection of telematic tools for improvement of safety in public transport. *in Advanced Video Based Surveillance Systems, C. S. Regazzoni, G. Fabri, and G. Vernazza, Eds. Norwell, MA: Kluwer*, 5:201–214, 1999.

[44] R. J. Evans and .R G. Porge. Video motion anomaly detection for military applications. In *in 3rd EMRS DTC Technical Conference*, 2006.

[45] F. Bremond. Scene Understanding: perception, multi-sensor fusion, spatio-temporal reasoning and activity recognition. *Habilitation a diriger der recherches presented at the University of Nice. Sophia Antipolis*, 2007.

[46] H. Feng, C. Chen, and C. Wu. Vision-based fuzzy 8051 surveillance systems design. *13th International Conference on Parallel and Distributed Systems*, pages 1–2, 2007.

[47] A. Fernández-Caballero, F. J. Gómez, and J. López-López. Road–traffic monitoring by knowledge–driven static and dynamic image analysis. *Expert Systems with Applications*, 35:701–719, 2008.

[48] J.M. Ferryman, S.J. Maybank, and A.D. Worrall. Visual surveillance for moving vehicles. *Int. J. Comput. Vis.(Kluwer Academic Publishers)*, 37(2):187–197, 2000.

[49] G.L. Foresti. A real–time system for video surveillance of unattended outdoor environments. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(6):697–704, October 1998.

[50] N. Friedman and S. Russell. Image segmentation in video sequences: A probabilistic approach. *13th Conf. Uncertainty in Artificial Intelligence*, pages 1–3, 1997.

[51] L. Fuentes, J.M. Troya, and A. Vallecillo. Desarrollo de software basado en componentes.

[52] F. Fusier, V. Valentin, F. Brémond, M. Thonnat, M. Borg, D. Thirde, and J. Ferryman. Video understanding for complex activity recognition. *Machine Vision and Applications*, 18(3):167–188, 2007.

[53] G.L. Foresti and L. Snidaro. A distributed sensor network for video surveillace of outdoor environments. *IEEE International Conference on Image Processing (ICIP 2002)*, 1:522–528, September 2002.

[54] I. Haritaoglu, D. Harwood, and L.S. Davis. Active outdoor surveillance. 1999.

[55] J. Heikkila and O. Silven. A real–time system for monitoring of cyclists and pedestrians. In *2nd IEEE Int. Workshop on Visual Surveillance*, pages 74–81. Colorado, 1999.

[56] G. Van Heist, A. Schreiber, and B. Wielinga. Using explicit ontologies in kbs development. *Int. J. Human-Computer Studies*, 45:138–292, 1997.

[57] E. E. Hemayed. A survey of self–camera calibration. In *Proc. of the IEEE Conf. on Advanced Video and Signal based Surveillance*, pages 351–358, Florida, 2003.

[58] M. Henning and M. Spruiell. Distributed programming with ice. technical report, zeroc ice. 2006.

[59] http://84.14.57.154/co friend.

[60] http://dilnxsvr.king.ac.uk/cromatica/.

[61] https://www.proyecto hesperia.org/hesperia/index.jsp.

[62] http://w3.racc.es/index.php?mod=fundacion&mem=EPDetalle&relmenu=31

[63] http://www.cieffe.com.

[64] http://www.detect.com.ar/.

[65] http://www.eurotestmobility.net/eurotest.php?itemno=342&lang=EN.

[66] http://www.gotchanow.com.

[67] http://www.ipsotek.com.

[68] http://www.neurodynamics.com.

[69] http://www.objectvideo.com.

[70] http://www.sivicam.com/rootss/index.php.

[71] I. Paulidis and V. Morellas. Two examples of indoor and outdoor surveillance systems. *in Remagnino, P., Jones, G.A., Paragios, N., and Regazzoni, C.S. (Eds.): 'Video–based Surveillance Systems' (Kluwer Academic Publishers*, pages 39–51, 2002.

[72] K. A. Ismail, T. Sayed, N. Saunier, and C. Lim. Automated analysis of pedestrian-vehicle conflicts using video data. *Transportation Research Board 88th Annual Meeting*, 2009.

[73] Special issue on third generation surveillance systems. Proc. IEEE. 2001.

[74] Special issue on Visual Analysis and Understanding for Surveillance Aplications.

[75] Special issue on visual surveillance. IEEE Trans. Pattern Anal. Mach. Intell. 2000.

[76] Special issue on visual surveillance. Int. J. Comput. Vis. 2000.

[77] Y. Ivanov, C. Stauffer, A. Bobick, and W.E.L. Grimson. Video surveillance of interactions. In *2nd IEEE Int. Workshop on Visual Surveillance*. Colorado.

[78] J. Krumm and S. Harris and B. Meyers and B. Brumit and M. Hale and S. Shafer. Multi–camera multi–person tracking for easy living. In *Third IEEE Int. Workshop on Visual Surveillance*, pages 8–11. Ireland, 2000.

[79] K.P. Karmann and A. von Brandt. Moving object recognition using an adaptive background memory. *Time-Varying Image Processing and Moving Object Recognition*, 2:289–296, 1990.

[80] M. Hsiao Ko, G. West, S. Venkatesh, and M. Kumar. Using dynamic time warping for online temporal fusion in multisensor systems. *Information Fusion*, 9(3):370 – 388, 2008. Special Issue on Distributed Sensor Networks.

[81] L. Bo and C. Qimei and G. Fan. Freeway Auto–surveillance From Traffic Video. In *6th International Conference on ITS Telecommunications Proceedings*, pages 167–170, 2006.

[82] L. Jian Guang and L. Qi Feing and T. Tie Niu and H. Wei Ming. 3D–model based visual traffic surveillance. *Acta Automatica Sinica*, 29(3):434–449, 2003.

[83] L. Snidaro and G. L. Foresti and R. Niu and P. K. Varshney. Sensor Fusion for Video Surveillance. In *7th Int. Conf. on Information Fusion*, pages 739–746, 2004.

[84] L. Wang and T. Tan and H. Ning and W. Hu. Silhouette Analysis-Based Gait Recognition for Human Identification. *IEEE Transactions on pattern analysis and machine intelligence*, 25(12):1505–1518, 2003.

[85] C. Larman. Capítulo 30. *UML y Patrones: Introducción al análisis y diseño orientado a objetos y proceso unificado*, 2008.

[86] C. Lee, B. Hellinga, and F. Saccomanno. Real-time crash prediction model for application to crash prevention in freeway traffic. *Transportation Research Record: Journal of the Transportation Research Board*, pages 67–77, 2007.

[87] P. Li, L. Ding, and J. Liu. A video-based traffic information extraction system. *In Proceedings of the IEEE intelligent vehicles symposium*, 2003.

[88] A.J. Lipton, H. Fujiyoshi, and R.S. Patil. Moving target classification and tracking from real-time video. *Fourth IEEE Workshop on Applications of Computer*, 1998.

[89] M. Christensen and R. Alblas. V^2 design issues in distributed video surveillance systems. In , pages 1–86, 2000.

[90] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *In Proceedings of the 1996 European Conference on Computer Vision*, pages 343–356, 1996.

[91] M. Xu and L. Lowey and J. Orwell. Architecture and algorithms for tracking football players with multiple cameras. In *Proc. IEE Workshop on Intelligent Distributed Surveillance Systems*, pages 51–56. London, 2004.

[92] T. Machmer, J. Moragues, A. Swerdlow, L. Vergara, J. Gosálbez, and K. Kroschel. Robust impulsive sound source localization by means of an energy detector for temporal alignment and pre-classification. *EUSIPCO 2009*, 2009.

[93] Nicholas A. Mandellos, Iphigenia Keramitsoglou, and Chris T. Kiranoudis. A background subtraction algorithm for detecting and tracking vehicles. *Expert Systems with Applications*, In Press, Corrected Proof:–, 2010.

[94] L. Marchesotti, A. Messina, L. Marcenaro, and C.S. Regazzoni. A cooperative multisensor system for face detection in video surveillance applications. *Acta Automatica Sinica*, 29(3):423–433, 2003.

[95] S.J. McKenna, S. Jabri, Z. Duric, A. Rosenfeld, and H. Wechsler. Tracking groups of people. *Computer Vision and Image Understanding*, 80(1):42 – 56, 2000.

[96] D. Meyer and J. Denzler. Model based extraction of articulated objects in image sequences for gait analysis. *Image Processing, International Conference on*, 3:78, 1997.

[97] P. G. Michalopoulos. Vehicle detection video through image processing: The autoscope system. *IEEE Transactions on vehicular technology*, 40:21–29, 1991.

[98] M. Mohammadian. Multi-agents systems for intelligent control of traffic signals. *In International conference on computational intelligence for modelling control and automation and international conference on intelligent agents web technologies and international commerce (CIMCA-06)*, pages 270–276, 2006.

[99] N.T. Nguyen and H.H. Bui and S. Venkatesh and G. West. Recognising and monitoring high–level behaviour in complex spatial environments. *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 1–6, 2003.

[100] C. Nwagboso. User focused surveillance systems integration for intelligent transport systems. *in Regazzoni, C.S., Fabri, G., and*

Vernazza, G. (Eds.): Advanced Video–based Surveillance Systems (Kluwer Academic Publishers), 1(1):8–12, 1998.

[101] IEEE Conference on Advanced Video and Santa Fe. NM. USA. Signal Based Surveillance (AVSS08). September. 1-3. 2008.

[102] First IEE Workshop on Intelligent Distributed Surveillance Systems. February 2003. London.

[103] Second IEE Workshop on Intelligent Distributed Surveillance Systems. February 2004. London.

[104] First IEEE Workshop on Visual Surveillance. January 1998. Bombay. India.

[105] Second IEEE Workshop on Visual Surveillance. January 1999. Fort Collins. Colorado.

[106] Third IEEE International Workshop on Visual Surveillance. July 2000. Dublin. Ireland.

[107] Seventh IEEE International Workshop on Visual Surveillance. June 2007. Minneapolis. USA.

[108] Ninth International Workshop on Visual Surveillance. Kyoto. Japan. 2009.

[109] Sixth IEEE International Workshop on Visual Surveillance. May 2006. Graz. Austria.

[110] B. Ugur Toreyin; E. Birey Soyer; I. Onaran and A. Enis Cetin. Falling person detection using multisensor signal processing. *EURASIP Journal on Advances in Signal Processing*, 2008.

[111] P. Blauensteiner and M. Kampel. Visual Surveillance of an Airport Apron - An Overview of the AVITRACK Project. In *In Proceedings of the Workshop of the Austrian Association for Pattern recognition and Artificial Intelligence*, pages 213–220, 2004.

[112] P. Gil–Jiménez and R. López–Sastre and P. Siegmann and J. Acevedo–Rodríguez and S. Maldonado–Bascón. Automatic Control

of Video Surveillance Camera Sabotage. *IWINAC 2007*, pages 222–231, 2007.

[113] P. Lai Lo and J. Sun and S.A. Velastin. Fusing visual and audio information in a distributed intelligent surveillance system for public transport systems. *Acta Automatica Sinica*, 29(3):393–407, 2003.

[114] G. Park and S.S. Park. Moving objects spatiotemporal reasoning model for battlefield analysis. In *ICS'08: Proceedings of the 12th WSEAS international conference on Systems*, pages 722–733, Stevens Point, Wisconsin, USA, 2008. World Scientific and Engineering Academy and Society (WSEAS).

[115] M. A. Patricio, J. Carbó, O. Pérez, J. García, and J. M. Molina. Multi-agent framework in visual sensor networks. *EURASIP J. Appl. Signal Process*, pages 226–226, 2007.

[116] I. Paulidis and V. Morellas. Two examples of indoor and outdoor surveillance systems. *in Remagnino, P., Jones, G.A., Paragios, N., and Regazzoni, C.S. (Eds.): 'Video–based Surveillance Systems' (Kluwer Academic Publishers, Boston)*, pages 39–51, 2002.

[117] M. Pellegrini and P. Tonani. Highway traffic monitoring. *in Regazzoni, C.S., Fabri, G., and Vernazza, G. (Eds.): Advanced Videobased Surveillance Systems (Kluwer Academic Publishers)*, 1998.

[118] G. Pieri and D. Moroni. Active video surveillance based on stereo and infrared imaging. *EURASIP Journal on Advances in Signal Processing*, 2008.

[119] F. Porikli, Y. Ivanov, and T. Haga. Robust abandoned object detection using dual foregrounds. *EURASIP Journal on Advances in Signal Processing*, 2008.

[120] A. Pozzobon, G. Sciutto, and V. Recagno. Security in ports: the user requirements for surveillance system. *in Regazzoni, C.S., Fabri, G., and Vernazza, G. (Eds.): Advanced Video–based Surveillance Systems (Kluwer Academic Publishers)*, 1998.

[121] A. Prati, R. Cucchiara, and R. Vezzani. A multi-camera vision system for fall detection and alarm generation. *Expert Systems*, 24(5):334–345, 2007.

[122] R. Bodor and B. Jackson and N. Papanikolopoulos. Vision–Based Human Tracking and Activity Recognition. In *Proc. of the 11th Mediterranean Conf. on Control and Automation*, pages 18–20, June 2003.

[123] R. Cutle and L.S. Davis. Robust Real-Time Periodic Motion Detection, Analysis, and Applications. *IEEE Computer Society*, 22(8):781–796, 2000.

[124] R. Radhakrishnan and A. Divakaran and P. Smaragdis. Audio analysis for surveillance applications. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New Paltz, October 2005.

[125] P. Remagnino, A. Baumberg, T. Grove, D. Hogg, T. Tan, A. Worral, and K. Baker. An integrated traffic and pedestrian model–based vision system. In *BMVC Proc.*, pages 380–389. Israel, 1997.

[126] H. A. Richards and R. T. Bartoskewitz. The intelligent highway–rail intersection: Integrating its and acts for improved crossing operations and safety. In *Int. Conf. Road Vehicle Automation*, pages 510–519. Bolton, U.K., 1995.

[127] N. Ronetti and C. Dambra. Railway station surveillance: the italian case. *in Foresti, G.L., Mahonen, P., and Regazzoni, C.S. (Eds.): Multimedia Video–Based Surveillance Systems (Kluwer Academic Publishers*, pages 13–20, 2000.

[128] N. Rota and M. Thonnat. Video sequence interpretation for visual surveillance. *3rd IEEE Int. Workshop on Visual Surveillance*, pages 59–68, 2000.

[129] M.D. Ruiz and J. Medina. Medea, sistema para la informatización del proceso de enfermería. In Proyecto Sur. Industrias Gráficas S.L., editor, *VI International Simposium of Nursing Diagnoses*, pages 231–235, Granada, July 2006.

[130] M.D. Ruiz, J. Medina, M. Delgado, and A.Vila. Architecture for databases access and consultation through handheld devices. In *International Symposium on Ubiquitous Computing and Ambient Intelligence*, pages 301–308, Zaragoza, September 2007.

[131] M.D. Ruiz, J. Medina, M. Delgado, and A.Vila. Integrating signals from different cameras for video surveillance mobilization. In *International Symposium on Ubiquitous Computing and Ambient Intelligence*, pages 301–308, Zaragoza, September 2007.

[132] M.D. Ruiz, J. Medina, M.Ros, M. Delgado, and A.Vila. Guía turística móvil basada en localización mediante rfid. In *2as Jornadas Científicas sobre RFID*, Cuenca, November 2008.

[133] M.D. Ruiz-Lozano, J.L. Castro, M. Delgado, and J. Medina and. An expert fuzzy system for predicting object collisions. its application for avoiding pedestrian accidents. *Expert Systems with Applications*, 38(1):486 – 494, 2011.

[134] S.A. Velastin and L. Khoudour and B.P.L. Lo and J. Sun and M. Vicencio–Silva. PRISMATICA: a multi–sensor surveillance system for public transport networks. In *Institute of Electrical Engineers (IEE)*, pages 19–25, 2004.

[135] N.T. Siebel and S. Maybank. The advisor visual surveillance system. *Proceedings of the ECCV workshop Applications of Computer Vision*, pages 103–111, 2004.

[136] M. J. Silla-Martínez. Sistema de videovigilancia inteligente utilizando anotaciones mpeg-7. Master's thesis, Grupo de Procesado de Imagen y Video, Universidad Politécnica de Valencia, 2008.

[137] G.J.D. Smith. Behind the screens: Examining constructions of deviance and informal practices among cctv control room operators in the uk. *Surveillance & Society*, 2((2/3)):377–396, 2004.

[138] J.G. Stell. Part and complement: Fundamental concepts in spatial relations. *Anals of Mathematics and Artificial Intelligence*, 41(1):1–17, 2004.

[139] E. Stringa and C.S. Regazzoni. Content–based retrieval and real time detection from videosequences acquired by surveillance systems. In *International Conference on Image Processing. ICIP*, volume 3, pages 138–142, October 1998.

[140] H. Susama and M. Ukay. Application of image processing for railways. In *Int. Rep. Railway Technological Res. Inst. (RTRI)*, volume 30, pages 74–81, 1989.

[141] C. Szyperski. Component software: Beyond object-oriented programming. *Addison-Wesley*, 2002.

[142] A.M. Sánchez, M.A. Patricio, J. García, and J.M. Molina. A context model and reasoning system to improve object trackingin complex scenarios. *Expert Systems with Applications*, 36(8):10995 – 11005, 2009.

[143] T. Takagi and M. Sugeno. Fuzzy identification of systems and its application to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, 15(1):116–132, 1985.

[144] T. N. Tan, G. D. Sullivan, and K. D. Baker. Pose determination and recognition of vehicles in traffic scenes. In *Proc. European Conf. Comput. Vision*, pages 272–276. Sweden, 1994.

[145] F. Terroso-Saenz, M. Valdes-Vela, C. Sotomayor-Martínez, and A. Gomez-Skarmeta. Detecting traffic risk situations from vanet messages. an event stream processing approach. *IV International Symposium of Ubiquitous Computing & Ambient Intelligence, UCACmI*, pages 149–158, 2010.

[146] Thou–Ho (Chao–Ho) Chen and Cheng–Liang Kao and Sju–Mo Chang. An Intelligent Real–Time Fire–Detection Method Based on Video Processing. *IEEE 37th Annual 2003 Internat. Carnahan Conf. on Security Technology*, 14-16:104–111, October 2003.

[147] V. R. Tomas and L.A. Garcia. A cooperative multiagent system for traffic management and control. *In Proceedings of the fourth international joint conference on autonomous agents and multiagent systems*, pages 52–59, 2005.

[148] T.Schreck and J. Bernard and T. Von Landesberger and J. Kolhhammer. Visual cluster analysis of trajectory data with interactive Kohonen maps. *Information Visualization*, 8(1):14–29, 2009.

[149] I. Paulidis; V. Morellas; V. Tsiamyrtzis and S. Harp. Urban surveillance systems: from the laboratory to the commercial world. *Proc. IEEE*, 89(10):1478–1495, 2001.

[150] V. Kastrinaki and M. Zervakis and K. Kalaitzakis. A survey of video processing techniques for traffic applications. *Image and Vision Computing*, 21(4):359–381, 2003.

[151] M. Valera and S.A. Velastin. Intelligent distributed surveillance system: a review. *IEE Proc. Vis. Image Signal Process*, 152(2):192–204, April 2005.

[152] D. Vallejo, J. Albusac, L. Jimenez, C. Gonzalez, and J. Moreno. A cognitive surveillance system for detecting incorrect traffic behaviors. *Expert Systems with Applications*, 36(7):10503 – 10511, 2009.

[153] S.A. Velastin. Getting the best use out of cctv in the railways. *Rail Safety and Standards Board*, pages 1–17, July 2003.

[154] W. Hu and T. Tan and L. Wang and S. Maybank. A survey on Visual Surveillance of Object Motion and Behaviors. *IEEE Transactions on Systems, Man and Cibernetics*, 34(3):334–352, 2004.

[155] Wen–Bing Horng and Jian–Wen Peng. Real–Time Fire Detection From Video: A Preliminary Report. In *14th IPPR, Computer fission. Graphics and Image Processing*, pages 1–5, 2001.

[156] X.Yuan, Z. Sun, Y. Varol, and G. Bebis. A distributed visual surveillance system. *Proceedings. IEEE Conference on Advanced Video and Signal Based Surveillance*, pages 199–205, 2003.

[157] Y. Ivanov and A. Bobick. Recognition of visual activities and interaction by stochastic parsing. *IEEE Trans. Pattern Recognit. Mach. Intell.*, 22(8):852–872, 2000.

[158] Z. Zhi-Hong. Lane detection and car tracking on the highway. *Acta Automatica Sinica*, 29(3):450–456, 2003.