Universidad Carlos III de Madrid
Escuela Politécnica Superior

Departamento de Informática
Doctorado en Ciencia y Tecnología Informática

Tesis Doctoral

Fusión de Datos Distribuida en Redes de Sensores Visuales Utilizando Sistemas Multi-Agente

Federico Castanedo Sotela

Dirigida por

Jesús García Herrero

Miguel Ángel Patricio Guisado

Mayo de 2010

Página web: http://fcastanedo.com
http://www.giaa.inf.uc3m.es

Correo electrónico: castanedofede@gmail.com
federico.castanedo@uc3m.es

Teléfono: +34 619 77 55 45

Dirección:

Grupo de Inteligencia Artificial Aplicada
Departamento de Informática
Universidad Carlos III de Madrid
Av. de la Universidad Carlos III, 22
Colmenarejo 28270 — Spain

A Ilaria,
por toda su paciencia en estos años

Resumen

Las técnicas de fusión de datos han surgido como una solución al problema de combinar de forma óptima múltiples fuentes de información redundantes; de esta manera, representan una alternativa que mejora (cuando se utiliza correctamente) los sistemas basados en una sola fuente de información. Esta mejora viene proporcionada por la redundancia inherente en los sistemas que utilizan múltiples fuentes de información. En esta tesis, se estudian las técnicas de fusión datos existentes en la actualidad y su aplicación al dominio de las redes de sensores visuales. Una red de sensores visuales consiste en un conjunto de cámaras, distribuidas de forma lógica, espacial o geográfica en un entorno y conectadas por una red de transmisión. La información del entorno, obtenida por las redes de sensores visuales, se encuentra distribuida y debe ser combinada o integrada utilizando técnicas de fusión de datos. Para que la fusión de los datos del entorno, obtenidos por la red de sensores visuales, pueda ser llevada a cabo de una forma óptima, es necesario disponer de una arquitectura software que proporcione la comunicación entre los nodos implicados y la suficiente flexibilidad al proceso de fusión.

El paradigma de los sistemas multi-agente, se ha establecido dentro de la comunidad de inteligencia artificial, como el conjunto de tecnologías que pretenden proporcionar los principios para construir sistemas complejos que involucran múltiples agentes y los mecanismos para coordinar los comportamientos individuales de los agentes y las comunicaciones entre ellos. En este contexto, se propone el uso de los sistemas multi-agentes para construir redes de sensores visuales que utilicen técnicas de fusión de datos para mejorar el proceso visual.

La primera parte de este documento, se dedica a revisar el estado actual de las técnicas, métodos y algoritmos de fusión de datos clásicos (en su modalidad centralizada y distribuida) y de los sistemas multi-agente, haciendo hincapié en aquellos que han sido aplicados en el dominio de las redes de sensores visuales. Debido a la gran extensión de los trabajos existentes en estas tres áreas (fusión de datos, sistemas multi-agente y visión artificial), esta sección no pretende ser una revisión exhaustiva, sino un marco para establecer el contexto de este trabajo y una forma de resaltar los trabajos más destacados en cada área.

La segunda parte, proporciona una descripción de la propuesta realizada dentro de las redes de sensores visuales, que consiste en el diseño e implementación de una arquitectura multi-agente que realiza fusión de datos dentro de una red de sensores distribuidos. Entre todos los tipos de arquitecturas multi-agente se ha optado por el uso de una arquitectura híbrida (de tipo Belief-Desire-Intention), ya que representa de forma explícita el conocimiento frente a otras arquitecturas puramente reactivas. La principal aportación de la arquitectura propuesta es el uso de la tecnología de sistemas multi-agente para modelar las redes de sensores visuales y realizar la fusión de los datos. En concreto se propone utilizar una *fusión activa*, la cual aporta la ventaja de que los agentes involucrados en el proceso de fusión se auto-corrijan; además, se establecen las bases de protocolos para la formación de coaliciones dinámicas entre los agentes involucrados en el proceso.

El sistema desarrollado se ha probado de forma experimental, por un lado utilizando información obtenida por varias cámaras desplegadas dentro del laboratorio de investigación y por otro lado utilizando conjuntos de datos disponibles dentro de la comunidad de visión artificial para la investigación.

Abstract

Data fusion techniques have been proposed as a solution to the problem of combining, in an optimal way, multiple redundant sources. Therefore, they are an alternative which, when are correctly employed, improve traditional systems that are based on only one source. This improvement is provided by the inherent redundancy of the systems that use multiple sources. In this work, data fusion state-of-art techniques are explored and their applications to the visual sensor domain are analyzed.

A visual sensor network consists of a network of cameras, logically, spatially or geographically distributed in an environment and connected by a transmission network. Environmental information, obtained by the visual sensor network is distributed and should be combined or integrated using data fusion techniques.

With the aim of fuse in an optimal way, the monitoring data, obtained by the visual sensor network it is required to have a software architecture which provides the communication between the nodes and the necessary flexibility to the data fusion process.

The multi-agent paradigm has been established by the artificial intelligence community, as the technology which provides the principles for building complex systems that involve multiple agents and which provide the mechanism to coordinate individual agent's behaviors and their communications. In this scenario, the application of multi-agent systems with the aim of building visual sensor networks that use data fusion techniques to improve the visual process, is proposed.

The first part of this document, reviews the classic data fusion state-of-art techniques, methods and algorithms (in their centralized and distributed versions) and the multi-agent systems, providing special emphasis in those systems applied to visual sensor networks. Due to the enormous amount of works in these three areas (data fusion, multi-agent systems and computer vision), this section does not aim to be an exhaustive reviewing, instead aims to be a framework for this work and a way to highlight the most relevant works.

The second part of the document, provides a detailed description of the developed proposed in the area of visual sensor networks. The work consists in the design and implementation of a hybrid multi-agent architecture (Belief-Desire-Intention model) which performs data fusion in a visual sensor network. The main contribution of this proposed architecture is the use of multi-agent systems for modeling visual sensor networks and fuse data. In particular, is proposed an active data fusion, which provides the advantage of auto-correct the state of the agents involved in the data fusion process. Moreover, the basis of protocols for dynamic coalition formation between the agents involved in the data fusion process, are proposed.

The developed system has been experimental tested, on one hand using the information obtained from several cameras deployed in our indoor laboratory and on other hand using available datasets for the computer vision community.

Agradecimientos

Me gustaría agradecer a todas aquellas personas que han hecho este trabajo posible. Sin embargo, han sido tantas las que de una forma directa o indirecta han contribuido al mismo, que enumerarlas a todas ellas sin olvidarme de alguna, sería un problema imposible de resolver con las restricciones de tiempo y tamaño (y mi mala memoria) que tengo para escribir estos agradecimientos.

En primer lugar, quiero agradecer a mis padres Celedonio y Lizbeth, por su constante apoyo a la hora de enfrentarme con la difícil tarea que tiene un investigador predoctoral, la cual requiere de la paciencia necesaria para sacar adelante una tesis doctoral. A mis hermanas, Natalia y Cristina; a Luis de la Flor, y a mis familiares y amigos de Costa Rica e Italianos (Simona, Marco y Onorato), gracias también por aguantarme en algunos momentos. A mis amigos: Dany, Gema, Nacho, Ana, Pedro, Carmen, Tomás, Héctor y muchos más... A Alberto Roncero por echarme una mano más de una vez con la escritura de los artículos en inglés.

También quiero agradecer a mis primeros compañeros de aventuras en la aplicación de técnicas de optimización a problemas de ingeniería reales (allá por el año 2002) de la empresa Meteológica: Manuel, Enrique y Rafa. Y a mis antiguos compañeros del Puño de Tecsidel (Alvaro, Fernando, Carlos, Javi, Luis, Chelu, deivid, Gato,... y muchos más) con los cuales compartí buenos ratos con el p**o gdb. Todos ellos me ayudaron aportándome el suficiente conocimiento técnico para afrontar con éxito este trabajo.

Este trabajo nunca hubiera sido posible sin el apoyo científico y técnico recibido por parte de mis dos directores de tesis: Jesús García Herrero y Miguel Ángel Patricio Guisado, que han podido dedicar parte de su tiempo y de sus conocimientos para contribuir al mismo. Quiero agradecer también a todos los miembros del Grupo de Inteligencia Artificial Aplicada (GIAA) del departamento de Informática (los que están y los que se fueron) y a los profesores de otros departamentos/grupos del campus de Colmenarejo, que de una forma u otra han hecho más llevadero el trabajo diario. A Ramón J. Flores por recordarme que hay que comer de vez en cuando para coger fuerzas y poder darle a la raqueta, a Luis Martí por sus consultorías sobre como darle al Latex con el batex y por el hacking de esta gran plantilla, sin la cual este documento no tendría este aspecto. A Cesár de Pablo por sus discusiones sobre la web 2.0 y conversaciones frikis. A Rodri y Ana por sus ayudas en el desarrollo de parte de las ideas de este trabajo y por las interesantes discusiones. A todos los demás profesores de la UC3M con los que he compartido agradables discusiones durante la hora de la comida y los cafés.

Finalmente, quiero agradecer las subvenciones recibidas por parte de los proyectos de investigación, sin los cuales no hubiera podido estar al día y tener interesantes discusiones con investigadores en congresos internacionales. Los comentarios y las discusiones intercambiadas con José Miguel Canino, James Llinas y Hamid Aghajan también me han resultado de mucha utilidad para el desarrollo de este trabajo.

A todos vosotros, de nuevo, gracias por dedicarme parte de vuestro preciado tiempo.

Fede

Índice general

Índice de figuras

Índice de tablas

1
Introducción

Vale más actuar exponiéndose a arrepentirse de ello, que arrepentirse de no haber hecho nada.
Giovanni Boccaccio.

En este primer capítulo se presenta el contexto del problema de este trabajo y la motivación del mismo en la sección 1.1. A continuación, en la sección 1.2 se presenta el planteamiento general. El planteamiento del problema y las técnicas de fusión de datos presentadas en este trabajo se enfocan desde una visión general, sin embargo, la experimentación realizada es aplicada a la fusión de posiciones durante el seguimiento de objetivos en una red de sensores visuales.

En la sección 1.3, se proporciona una descripción formal del problema a tratar. Finalmente, en el apartado 1.4, se describe la estructura del resto del documento.

1.1. Contexto del problema y motivación

Los primeros trabajos relacionados con los métodos de fusión de datos se remontan al año 1786 con el método de Condorcet[1] para las votaciones en los modelos de la democracia. Los métodos de fusión de datos se siguieron aplicando a una serie de disciplinas diversas como la fiabilidad (Von Neumann, J., 1956), el reconocimiento de patrones (Chow, 1965), las redes de neuronas (Hashem, 1993, 1997), la toma de decisiones (Dasarathy, 1994) (Varshney, 1997), la estimación estadística (Breiman, 1996) (Juditsky & Nemirovski, 2000) y la predicción del tiempo (Granger, 2001).

En los sistemas que constan de diversos sensores, los métodos de fusión de datos han demostrado ser especialmente útiles, debido a que proporcionan al sistema la posibilidad de utilizar los datos de múltiples sensores frente a los sistemas mono-sensores. La fusión de los datos es un aspecto clave y crítico en sistemas con varios sensores. El objetivo de la fusión de datos de múltiples sensores es similar al de otras disciplinas, y puede definirse como: *la acción de fusionar o combinar de una forma eficiente y precisa los datos de múltiples sensores para superar las limitaciones de utilizar un solo sensor.*

Una red de sensores distribuidos, es un conjunto de sensores, distribuidos de forma lógica, espacial o geográfica en un entorno y conectados por una red de transmisión. Los sensores pueden ser visuales (cámaras), auditivos (micrófonos), infrarrojos, sensores de humedad, sensores de temperatura, etc. Existen muchas ventajas derivadas de utilizar sistemas de múltiples sensores frente a sistemas mono-sensores tradicionales (Luo et al., 2002), entre las cuales se pueden enumerar las siguientes:

[1] El método de Condorcet, proviene del Marqués de Condorcet, matemático y filósofo del siglo XVIII. Básicamente definió un sistema de voto para elegir al candidato que los votantes prefieren, comparando cada candidato con los demás.

1. La mejora de la robustez y de la precisión en caso de fallos en los sensores.
2. La posibilidad de obtener información de características independientes en el sistema.
3. La posibilidad de cubrir un área más amplia, lo que proporciona una visión más completa del entorno.
4. El poder incrementar la dimensionalidad de las medidas.
5. La mejora de la resolución con la que se obtienen las medidas.
6. La reducción de la incertidumbre y del ruido y el incremento de la confianza.
7. Mejorar la discriminación de las hipótesis con la ayuda de información más completa obtenida por los diversos sensores.

Los sensores, por lo general, obtienen continuamente datos de mediciones del entorno, los cuales son procesados y transmitidos por la red. La datos del entorno, obtenidos de forma distribuida, deben ser integrados o combinados utilizando técnicas de fusión de datos. Estas técnicas pueden variar desde simples operaciones aritméticas hasta técnicas bastante más complejas. El reto clave en las redes de sensores distribuidos, es desarrollar modelos de red y enfoques computacionales óptimos para realizar el procesado y la fusión de los datos de los sensores.

Desplegar redes con múltiples sensores donde, cada uno de ellos captura las mismas características del entorno con diferente precisión, proporciona una redundancia inherente. Los sensores, por tanto, se pueden complementar entre ellos proporcionando información que sería muy difícil de obtener utilizando un solo sensor. En todos los sistemas de redes de sensores, un aspecto crítico es la fusión de los datos obtenidos por los sensores para obtener una mejor descripción.

En el caso de utilizar únicamente sensores visuales (cámaras), las redes de sensores se conocen como redes de sensores visuales y emplean técnicas de visión artificial para obtener los datos. La visión artificial es un campo muy amplio, entendiéndose como: *la transformación de los datos obtenidos por un sensor visual en una decisión o una nueva representación*. La tendencia actual es evolucionar desde los sistemas centralizados a los sistemas distribuidos basados en múltiples plataformas de bajo coste, las cuales como un grupo son capaces de realizar tareas más complejas que las que realizaría un único súper nodo[2]. Una de las características diferenciadoras de los sensores visuales frente a otros tipos de sensores, como los sensores de temperatura o de presión, es la gran cantidad de información que generan. Esto conlleva que sea imprescindible un procesado previo de los datos, para convertir la gran cantidad de datos obtenidos del entorno (imágenes en bruto), en un nivel conceptual que implique un menor ancho de banda a la hora de ser transmitidos por la red.

Los nuevos sistemas de vigilancia, conocidos como sistemas de tercera generación (Regazzoni et al., 2001) (Valera & Velastin, 2005), están basados en redes de sensores visuales y tienen como finalidad detectar, reconocer y seguir objetivos[3] a partir de secuencias de imágenes obtenidas por una red de sensores visuales. El fin primordial de estos sistemas de

[2]Se entiende por nodo cada una de las máquinas que conforman una red distribuida. Un sensor con capacidad de proceso también es considerado un nodo.

[3]Por objetivo (en inglés target) se define a la persona, entidad u objeto sobre el cual se aplican las distintas técnicas. La definición de lo que consiste un objetivo depende de la aplicación específica que se considera.

vigilancia es desarrollar una monitorización inteligente para reemplazar a los sistemas actuales, los cuales no son del todo efectivos, debido entre otras razones a la incapacidad de los operadores para gestionar un gran número de cámaras. La percepción humana y el razonamiento están restringidos por los límites de las capacidades humanas, por ejemplo, solo la información de un área espacial limitada puede ser observada al mismo tiempo. Los sistemas de vigilancia de tercera generación pretenden automatizar el proceso de vigilancia, partiendo del nivel de sensor y llegando hasta la información visual y simbólica proporcionada a los operadores. La monitorización de grandes áreas, solo puede ser llevada a cabo por el uso coordinado de múltiples cámaras y la información obtenida debe ser mostrada al usuario de una forma unificada, haciendo indispensable el uso de los métodos y técnicas de fusión de datos. Por tanto, este tipo de sistemas se puede considerar como un ejemplo particular donde se pueden aplicar las técnicas de fusión de datos para proporcionar mejores resultados.

Incluso en condiciones ambientales ideales, los sensores de los sistemas de vigilancia, pueden proporcionar medidas incorrectas. Esencialmente un sensor es un dispositivo que proporciona medidas con una incertidumbre asociada. En el caso de los sensores visuales, la información puede verse afectada por las oclusiones de los elementos o personas en seguimiento. Por tanto, los errores generados en una red de sensores visuales pueden ser debidos (entre otras) a las siguientes causas:

1. Las imperfecciones de la tecnología utilizada para obtener la información visual.
2. Los algoritmos utilizados para procesar los datos.
3. El entorno físico donde se obtiene la información visual.

Los fallos en los sensores pueden ocurrir con cierta frecuencia proporcionando medidas erróneas al proceso de seguimiento. Estos tipos de errores, generados por los sensores, pueden ser corregidos o reducidos utilizando las técnicas y métodos de fusión de datos. Finalmente, para poder realizar una fusión de los datos de una forma coherente, los distintos sensores involucrados deben ser coordinados y gestionados, esto se conoce como *sensor management* (Manyika & Durrant-Whyte, 1995).

Dentro del contexto de los sistemas distribuidos, los sistemas multi-agente han surgido como una evolución de las técnicas de inteligencia artificial distribuida. Un sistema multi-agente esta compuesto por varios agentes software que se ejecutan de forma autónoma. Esencialmente un agente software es un proceso computacional que se ejecuta en un nodo que tiene capacidad de proceso y de transmisión. En el caso de los sistemas multi-agente lo que se tiene son diversos agentes cada uno de ellos ejecutándose de forma distribuida en uno o varios nodos de los que se compone el sistema. La comunicación de los datos entre los agentes del sistema multi-agente, por lo general, se realiza por medio del paso de mensajes de una forma asíncrona. Un intercambio de mensajes asíncrono implica que la finalización de una operación de envío de un mensaje no conlleva su inmediata recepción y tratamiento. Por lo general, el tiempo que transcurre desde que se envía un mensaje, hasta que se recibe y se procesa, es impredecible y puede variar entre los diferentes mensajes. En el apartado 4 se proporcionan más detalles de las características de los sistemas multi-agente.

Todos los motivos expuestos anteriormente, justifican el interés en el estudio y mejora de las técnicas y métodos de fusión de datos existentes con el fin de ser aplicados en las redes de sensores visuales; lo cual se plasma en la principal motivación y aportación de este trabajo, que consiste en la aplicación de los sistemas multi-agente en una red de sensores visuales para realizar la fusión de datos.

1.2. Planteamiento general

La arquitectura clásica para la fusión de datos es centralizada, esto quiere decir que los datos en bruto[4] de los diferentes sensores son enviados al nodo central en el cual son fusionados y los resultados son presentados a los usuarios. No obstante, existen distintas arquitecturas posibles para realizar una fusión distribuida (Liggins et al., 1997). En el caso particular de las redes de sensores visuales, una arquitectura centralizada, en la cual las imágenes obtenidas son transmitidas en bruto por la red a un nodo central, puede ser muy poco práctico, debido al coste necesario del ancho de banda para transmitir por la red todas las imágenes. Por tanto, en las redes de sensores visuales es necesario realizar un procesado en el nodo o sensor que captura las imágenes y utilizar técnicas de fusión de datos distribuidas para agrupar la información.

Para solucionar los fallos en los sensores, las limitaciones tecnológicas y los problemas de cobertura espacial y temporal, es necesario asegurar tres propiedades en el sistema (Whyte, 1988) (Luo et al., 2002): (1) cooperación, (2) redundancia y (3) complementariedad. Las cuales se pueden definir como:

1. **Cooperación:** Normalmente una región amplia de interés solo puede ser cubierta de forma completa con el uso de varios sensores, cada uno de ellos cooperando con una vista parcial de la escena. La fusión de los datos puede ser utilizada para componer una visión completa de los datos locales proporcionados por cada sensor. En el caso de los sistemas multi-agente la cooperación conlleva importantes retos teóricos y prácticos (Ren et al., 2005), debido a que los datos necesarios para la cooperación pueden ser compartidos de diversas formas.

2. **Redundancia y** 3. **Complementariedad:** Intuitivamente se puede pensar que en el proceso de fusión, cuantos más datos se obtengan mejor, debido a que datos adicionales proporcionaran un mayor conocimiento. Sin embargo, Dasarathy (2000) demostró, que obviamente, cuando la cantidad de datos incorrectos es mayor que la cantidad de datos correctos el rendimiento del proceso de fusión se ve reducido. Esto se conoce como la propiedad de *no divergencia*. Un método de fusión se define como no divergente, si y solo si, nunca produce una estimación fusionada que es peor (de acuerdo a algún criterio fijo) que cualquiera de las estimaciones que se utilizan para producirla. Por ello, se hace necesario disponer de las técnicas de fusión de datos apropiadas para comprobar la coherencia de los datos antes de llevar a cabo el proceso de fusión.

La fusión de datos se enmarca dentro de una serie de disciplinas relacionadas, algunas de las cuales se encuentran más estudiadas que otras. Por un lado, las disciplinas más estudiadas como el seguimiento de múltiples objetivos proporcionan importantes bases teóricas al desarrollo de las aplicaciones. Por otro lado, las disciplinas más novedosas presentan una deficiencia de metodologías. El precio que se ha pagado, debido a la falta de una metodología unificada, es que los subsistemas son muy difíciles de integrar (Goodman et al., 1997). Esto conlleva, que cuestiones como la optimización de los sistemas sean una tarea muy difícil, ya que una mejora en un subsistema puede producir una degradación en otro (Goodman et al.,

[4] Entendiendo por datos en bruto, los obtenidos directamente por el sensor sin haber realizado ninguna transformación.

1997); por tanto, se hace necesario la existencia de una mayor facilidad de integración en los sistemas de fusión de datos.

El paradigma de agentes software y sistemas multi-agente se ha establecido, dentro de la comunidad internacional de inteligencia artificial, como una metodología para desarrollar entidades software autónomas y flexibles capaces de realizar un alto grado de razonamiento. Los sistemas multi-agente permiten el desarrollo de redes de sensores visuales de una forma natural. Estos sistemas, al consistir en un marco de trabajo para realizar sistemas distribuidos y al ser una evolución de las técnicas de inteligencia artificial distribuida, permiten modelar fácilmente los sistemas de fusión de datos distribuidos. Además los sistemas multi-agente proporcionan una mayor autonomía, flexibilidad y un mayor grado de razonamiento que los sistemas de fusión tradicionales. Debido a ello, se propone el uso de los sistemas multi-agente para obtener una mayor facilidad de integración, flexibilidad y grado de razonamiento dentro de las redes de sensores visuales que utilizan técnicas de fusión de datos. Esto implica que un sistema de fusión distribuido puede verse como un sistema multi-agente donde cada agente es un nodo de la red de sensores distribuida.

Para realizar una fusión de datos distribuida utilizando sistemas multi-agente y demostrar su aplicación en las redes de sensores visuales, en este trabajo, se propone:

- Establecer un lenguaje de comunicación e intercambio de datos común entre los sensores de la red. Esto se consigue definiendo el contenido de los mensajes FIPA ACL[5] que se intercambian los distintos procesos del sistema.
- Establecer un marco de coordenadas global y común a los datos proporcionados por todos los sensores. Esto se consigue por medio de las técnicas de calibración o alineación espacial, las cuales se presentan en el apartado 3.3.
- Disponer de mecanismos de comunicación e interacción entre los sensores de la red. El problema de determinar que se debe comunicar, es más importante, que el como se realiza la comunicación. El mecanismo de comunicación utilizado está basado en el intercambio asíncrono de mensajes entre los diferentes nodos involucrados.

Una red de sensores visuales consiste en un sistema de diversas cámaras, geográficamente dispersas, conectadas entre sí y cooperando. La tarea más importante en una red de sensores es procesar los datos, posiblemente ruidosos, adquiridos por varios sensores e integrarlos reduciendo la incertidumbre y produciendo interpretaciones abstractas de ellos. Existen dos aspectos importantes en este marco de trabajo:

1. La red debe poseer inteligencia en cada nodo. Es decir, los sensores o los sistemas que obtienen los datos deben tener capacidad de procesado de los mismos y de esta forma contribuir al proceso cooperativo de la red. Es necesario el procesado de los datos obtenidos (imágenes) para establecer los datos que van a ser intercambiados para la fusión.
2. El sistema debería permitir añadir más sensores, con el fin de incrementar el área a monitorizar, sin que esto conlleve una degradación en el rendimiento.

[5] FIPA son las siglas de The Foundation for Intelligent Physical Agents, una organización del IEEE que se encarga de promover y estandarizar la comunicación entre las tecnologías de agentes y otras tecnologías. ACL son las siglas de Agent Communication Language, el lenguaje de comunicación de agentes propuesto por FIPA.

Una red de sensores visuales es un ejemplo claro de un sistema distribuido y debería funcionar bajo las siguientes limitaciones:

1. Cada sensor en la red puede ver algunas de las actividades que tienen lugar dentro de su área de visión de una forma completa (pero no necesariamente todas las actividades).

2. El valor de los datos depende de forma crítica del tiempo requerido para obtenerlos y procesarlos. Es decir, si el tiempo que transcurre desde que se obtienen los datos hasta que se procesan es bastante mayor que la frecuencia con la que el entorno evoluciona, los datos obtenidos pierden valor.

3. La comunicación entre los nodos de la red debe ser limitada. Esto es debido a que una sobrecarga de comunicación conlleva un exceso de carga computacional y de ancho de banda necesario, lo cual es deseable evitar.

4. Debe existir suficiente información en el sistema para superar las condiciones anómalas (fallos de los nodos) y continuar proporcionando una solución al problema. Esto se puede conseguir por medio del uso de datos redundantes.

Con el objetivo de llevar a cabo con éxito la integración de los datos proporcionados por múltiples sensores, son necesarios los siguientes requisitos:

1. El desarrollo de métodos para representar de forma abstracta los datos de los sensores y que por tanto estos datos puedan ser integrados con facilidad.

2. El desarrollo de métodos que tengan en cuenta las diferencias en los datos obtenidos por los distintos sensores. Por ejemplo, en el caso de los sensores visuales, es necesario establecer un marco común a los diferentes campos de visión de cada sensor, y de esta forma establecer de forma cuantitativa las diferencias en los datos obtenidos.

3. El desarrollo de técnicas que permitan reducir la incertidumbre de las señales del sensor.

4. El análisis de la coherencia de los datos proporcionados por cada uno de los sensores, el cual es necesario como fase previa para realizar la fusión de datos.

5. Modelar el comportamiento de los sensores y los objetos así como la incertidumbre asociada.

La mayoría de los métodos de fusión de datos se basan en la independencia estadística de los errores de los sensores, lo cual simplifica mucho el diseño de las técnicas de fusión. Otra solución clásica, consiste en utilizar los métodos Bayesianos que minimizan el error esperado. Sin embargo, la solución práctica de estos métodos, requiere de expresiones analíticas de las distribuciones de los sensores bien definidas. En las redes de sensores visuales, obtener las distribuciones de error de los sensores puede ser muy complicado, además obtener expresiones analíticas de los sensores bien definidas es una tarea difícil y costosa que requiere la aplicación del conocimiento de muchas áreas, como la ingeniería eléctrica y el modelado estadístico. Por tanto, con el objetivo de simplificar su diseño, es necesario utilizar enfoques diferentes y más flexibles que las técnicas estadísticas clásicas.

1.3. Formalización del problema

Supongamos que se tiene un sistema compuesto por n sensores, $S = \{S_1, S_2, ...S_n\}$, los cuales se encuentran obteniendo imágenes en bruto del entorno con una cierta resolución y a una cierta frecuencia propias de cada sensor y su hardware asociado. Estos sensores se encuentran establecidos de forma que existen áreas solapadas entre ellos. Cada uno de estos sensores, S_i, está conectado a un procesador independiente por medio de una tarjeta capturadora, lo que constituye cada uno de los agentes sensores del sistema[6]. Los agentes sensores procesan las imágenes obtenidas generando los datos con la información del estado de cada uno de los objetivos detectados con una cierta frecuencia (*FreqProcesado*). Por otro lado, se define, $O = \{O_1, O_2, ...O_m\}$ como el conjunto de los m objetivos que se encuentran dentro del alcance de los sensores en un intervalo de tiempo determinado y sobre los que se proporciona información de su estado a partir de las observaciones de los sensores.

Cada uno de los sensores S_i, obtienen una serie de medidas con una frecuencia fc_i. Estas medidas, dan lugar al conjunto de medidas del objetivo O_j proporcionadas por el sensor S_i en un intervalo de tiempo t:
$X_{S_i}(t) = \{x_{S_i}^{O_j}(t_1), x_{S_i}^{O_j}(t_2), ..., x_{S_i}^{O_j}(t_t)\}$
siendo $t_1 < t_2 < ... < t_t$ y asumiendo que el instante de tiempo en el que se obtiene cada una de las observacionales es conocido. Además los relojes de todos los agentes involucrados en el proceso deben estar sincronizados entre sí.

Se asume que los agentes sensores tienen la capacidad de comunicación del estado de los objetivos entre ellos y con otros agentes pertenecientes a la red por medio de mensajes asíncronos. En particular, el agente o los agentes encargados de realizar la fusión tiene la posibilidad de recibir/enviar información de/a cada uno de los agentes sensores en forma de mensajes. El tiempo que transcurre desde que se envía un mensaje hasta que se recibe es impredecible y puede variar entre diferentes mensajes.

Los agentes sensores comunican la información procesada por medio de mensajes a los agentes de tipo fusión. A continuación, el proceso de fusión que se ejecuta en un agente de tipo fusión, debe realizar una cadena de decisiones y cálculos entre los datos comunes de los objetivos de cada sensor. En primer lugar, es necesario asociar los datos que corresponden al mismo objetivo O_j entre los diferentes agentes sensores S_i.

Posteriormente, es necesario realizar un análisis acerca de la calidad de la información proporcionada por cada sensor, dando lugar a un proceso cooperativo el cual tiene como finalidad limitar los posibles errores introducidos por las observaciones de los sensores. Este análisis se lleva a cabo en el agente de fusión, realizando una comprobación de la consistencia de los datos de los diferentes sensores.

Finalmente, el proceso de fusión, proporciona para un intervalo de tiempo k y con una frecuencia ff_i, la descripción fusionada que mejor se corresponde con los objetivos detectados en el área solapada de los sensores, por tanto proporcionaría:

$$\hat{X}_F(t) = \{\hat{x}_F^{O_j}(t_1), \hat{x}_F^{O_j}(t_2), ..., \hat{x}_F^{O_j}(t_k)\} \text{ para el objetivo } O_j, \text{ siendo } t_1 < t_2 ... < t_k.$$

Además, es posible realizar una fase de realimentación de la información desde el proce-

[6] En este documento, de cara a simplificar su lectura, no se realizan diferencias entre el sensor visual, la tarjeta capturadora y el procesador. Entiendo todo el conjunto como un sensor con capacidad de proceso o un agente sensor.

so de fusión al proceso de obtención de información de los agentes sensores en cada uno de los nodos. Es decir, la información de $\hat{X}_F(t)$ es enviada a los nodos de los agentes sensores S_i para corregir las posibles desviaciones en el estado de los objetos de cada uno de los agentes sensores. Este proceso, que utiliza los datos fusionados como realimentación, se ha denominado fusión activa.

El fin último, es que para cada uno de los valores fusionados del objetivo $\forall i \in \hat{X}_F(t)$, el resultado fusionado: $\hat{x}_F^{O_j}(t_i)$ proporcione una descripción más precisa o igual que la de cada uno de los sensores por separado. Esta mejora en la descripción, se debe precisamente al proceso de fusión de los mismos datos proporcionados por distintas fuentes. Para ello, es necesario realizar acciones de forma coordinada entre los nodos, con el objetivo de que la fusión sea más coherente y mejore el resultado.

1.4. Estructura del documento

Una vez presentada una introducción al contexto del trabajo y su motivación, el planteamiento y la definición del problema que se pretende resolver, se expone ahora la estructura del resto del documento.

El documento se encuentra dividido en dos partes. La primera de ellas consiste en el estado actual de las técnicas de fusión de datos, las redes de sensores visuales y los sistemas multi-agente.

Dentro de esta primera parte, en primer lugar en el capítulo 2, se expone la clasificación de las técnicas de fusión de datos. A continuación, se presentan los métodos, técnicas y algoritmos de fusión de datos centralizada (sección 2.7). Estos métodos se presentan clasificados en función del objetivo de los mismos, dando lugar a los siguientes 3 grupos:

1. Métodos de asociación.
2. Métodos de estimación.
3. Métodos de fusión de decisiones.

En esta sección, se detallan las características principales de cada uno de los diferentes métodos para el caso de la fusión de datos centralizada y se describen sus ventajas e inconvenientes. La misma división y descripción se realiza con los métodos utilizados en la fusión de datos distribuida, los cuales se presentan en la sección 2.8. A continuación, en el capítulo 3 se introducen las redes de sensores visuales y sus particularidades para realizar un seguimiento distribuido de forma coherente. Posteriormente, en el capítulo 4 se presenta el estado actual de los sistemas multi-agente y como estos se pueden aplicar en las redes de sensores visuales para facilitar y mejorar las técnicas de fusión de datos.

La segunda parte del documento, consiste en las aportaciones realizadas y las soluciones que se han aplicado para realizar una fusión de datos en las redes de sensores visuales utilizando sistemas multi-agente. En primer lugar, en el capítulo 5, se describe el sistema multi-agente desarrollado para solucionar los problemas que existen dentro de las redes de sensores visuales. Posteriormente, el capítulo 6 presenta la experimentación realizada para validar los objetivos del sistema. Finalmente, el capítulo 7, proporciona las conclusiones derivadas de la investigación, la lista de publicaciones generadas durante el desarrollo de la misma y las líneas de trabajo futuro.

I

Estado actual

2
Fusión de datos

Essentially, all models are wrong, but some are useful.
George Box.

2.1. Introducción

Diversos sinónimos se utilizan para referirse a la materia de fusión de datos, tales como: fusión de información, fusión de sensores, integración de datos, integración de información, etc...A pesar de las posibles diferencias filosóficas de la palabra datos o información, en este trabajo los términos fusión de datos o fusión de información se consideran equivalentes.

La definición de fusión de datos más aceptada data de 1991, cuando el grupo de trabajo de fusión de datos del "Joint Directors of Laboratories", JDL (1991), proporcionó la siguiente definición: *"Un proceso multinivel que trata con la detección automática, asociación, correlación, estimación y combinación de los datos e información desde múltiples fuentes".*

Hall & Llinas (1997) definieron la fusión de datos como: "*la combinación de datos de distintos sensores, y su información relacionada proporcionada por las bases de datos asociadas, para conseguir una mejor precisión e inferencias más especificas de las que se pueden obtener con un solo sensor*" .

La fusión de datos es una materia multi-disciplinar, en el sentido de que las técnicas y métodos que se utilizan provienen de diversas disciplinas. Es posible realizar una clasificación de los diferentes métodos que realizan fusión de datos atendiendo a los siguientes cinco criterios:

1. Dependiendo de las relaciones existentes entre los datos de entrada. Estas pueden consistir en datos complementarios, redundantes o cooperativos.

2. En base a los tipos de datos de entrada y salida utilizados en el proceso de fusión de datos.

3. Utilizando el nivel de abstracción de los datos manipulados durante el proceso de fusión, es decir, si se utilizan medidas, señales, características, o decisiones.

4. Basándose en el criterio de los niveles de fusión definidos por el "Joint Directors of Laboratories" (JDL).

5. Atendiendo a los distintos tipos de arquitecturas posibles.

El resto del capítulo está organizado para describir cada una de las clasificaciones que se acaban de enumerar en las secciones 2.2, 2.3, 2.4, 2.5 y 2.6.

Posteriormente en la sección 2.7 se describen las técnicas de fusión de datos centralizada más relevantes, divididas en 3 grupos: (1) asociación de datos en la sección 2.7.1, (2) estimación de estado en la sección 2.7.2 y (3) fusión de decisiones en 2.7.3. La fusión distribuida y las técnicas asociadas se presentan en la sección 2.8. Finalmente se presenta un resumen crítico en la sección 2.9.

2.2. Clasificación basada en las relaciones de los datos de entrada

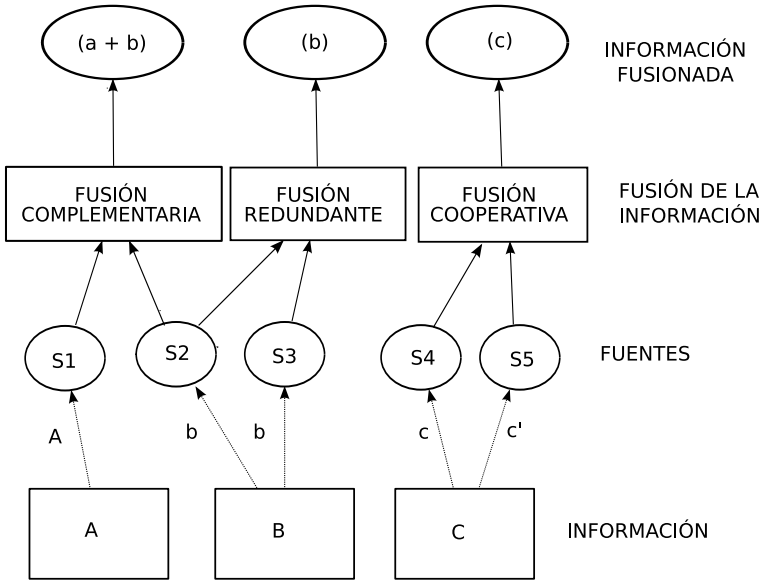

Figura 2.1: Esquema conceptual de los tipos de fusión de datos atendiendo a las relaciones de los datos de entrada.

Dependiendo de las relaciones existentes entre los datos de entrada (Whyte, 1988), la fusión de información[7] se puede clasificar como:

1. **Fusión de información complementaria:** Si la información proporcionada por los datos de entrada representa diferentes partes de la escena, y por tanto la fusión de información se puede utilizar para obtener una información más completa; por ejemplo, en el campo de las redes de sensores visuales la información de un mismo objetivo proporcionada por cámaras que no comparten el campo de visión se considera información complementaria.

2. **Fusión de información redundante:** Si dos o más fuentes independientes proporcionan información acerca del mismo objeto, pueden ser fusionadas para incrementar la confianza; por ejemplo, en el campo de los sensores visuales la información de áreas solapadas se considera información redundante.

[7] A lo largo del documento y en el contexto de la fusión de datos, se utiliza el termino datos o información como sinónimos.

3. **Fusión de información cooperativa:** Dos fuentes de información son cooperativas cuando la información proporcionada por ellas es fusionada en una nueva información (normalmente más compleja que los datos originales). La fusión cooperativa debe ser aplicada con cuidado debido a que los datos resultantes están sujetos a las imperfecciones de las fuentes participantes. En el ejemplo de los sistemas de vigilancia, es posible realizar fusión de información cooperativa si se utilizan distintos tipos de sensores (visuales y auditivos) (Brooks & Iyengar, 1998).

En la figura 2.1 se muestra un esquema de los tipos de fusión de información atendiendo las relaciones existentes entre los datos de entrada.

2.3. Clasificación basada en la entrada y salida

Una de las clasificaciones de fusión más conocida es la proporcionada por Dasarathy (1997), en la cual los procesos de fusión son clasificados basándose en la abstracción de entrada y salida de la información. Dasarathy define cinco posibles categorías:

1. **Data In - Data Out (DAI-DAO):** Esta es la forma más elemental o de bajo nivel de fusión que se considera en esta clasificación. El proceso de fusión trata con datos en bruto y el resultado son datos en bruto, normalmente más precisos o fiables. La fusión de datos en este nivel se lleva a cabo inmediatamente después de la adquisición de los datos por los diferentes sensores. Los paradigmas de fusión en esta categoría generalmente están basados en técnicas desarrolladas en los dominios del tratamiento de la señal y del procesado de imágenes.

2. **Data In - Feature Out (DAI-FEO):** En este siguiente paso de la clasificación, el proceso de fusión utiliza datos en bruto de las fuentes para extraer características o atributos que describan una entidad en el entorno. La fusión en esta fase se conoce como fusión de características, fusión simbólica, fusión de información, fusión a nivel intermedio, etc. La forma en la cual la percepción humana distingue la profundidad de los objetos por medio de la combinación de la información de los dos ojos, es un claro ejemplo de esta categoría de fusión.

3. **Feature In - Feature Out (FEI-FEO):** En esta etapa de la jerarquía tanto la entrada al proceso de fusión como la salida del mismo son características. Por tanto, el proceso de fusión trabaja con un conjunto de características para mejorar, refinar o extraer nuevas características. Esto es lo que se conoce normalmente como fusión de características. En este tipo de fusión en lugar de fusionar medidas en bruto se fusionan las características extraídas de estas medidas y eso dentro del dominio de las redes de sensores visuales, sucede cuando cada sensor en el entorno tiene su propio conjunto único de estructuras de datos diferentes y donde no se pueden obtener los datos internos de cada estructura. Este es el enfoque utilizado en este trabajo.

4. **Feature In - Decision Out (FEI-DEO):** Este tipo de fusión es uno de los enfoques más comunes utilizados en la literatura de fusión de datos. Se toman una serie de características de entrada y se genera como salida una decisión. Dependiendo del enfoque se puede llamar fusión de características o fusión de decisiones. La mayoría de los sistemas

de reconocimiento de patrones que toman una decisión en función del valor de las entradas de los diferentes sensores, están realizando este tipo de fusión.

5. **Decision In - Decision Out (DEI-DEO):** Esta es la última clasificación de la jerarquía establecida por Dasarathy y normalmente se conoce como fusión de decisiones. Consiste en fusionar las decisiones para obtener mejores o nuevas decisiones, es decir, tanto la entrada al proceso de fusión como la salida son decisiones. Dependiendo del tipo de sensores, la fusión a nivel de datos de entrada no siempre puede ser realizada; debido a que el proceso de fusión debe alinear la información obtenida de los sensores para poder realizar la fusión. En el caso de la fusión de decisiones el problema se simplifica debido a que el proceso de fusión trabaja directamente con las decisiones obtenidas de los sensores.

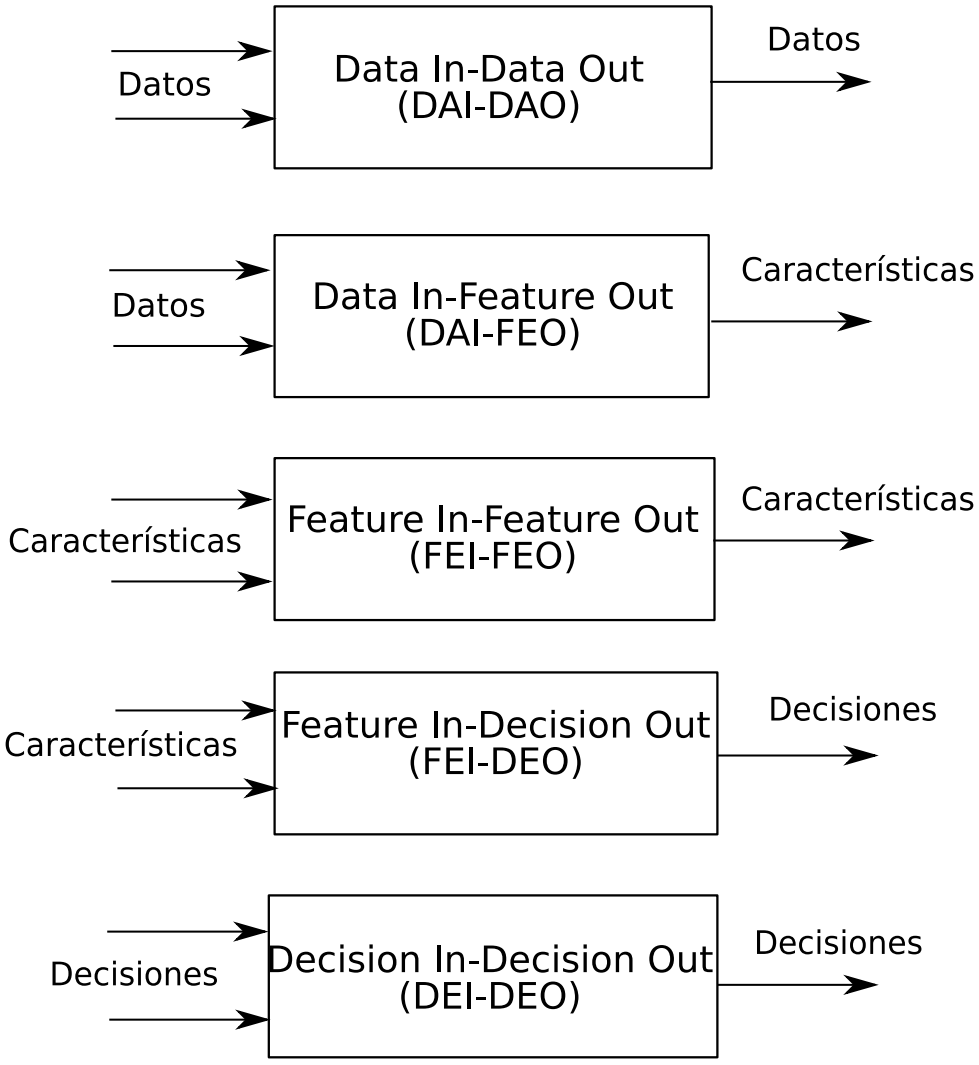

Figura 2.2: Clasificación de los tipos de fusión de datos en función de la entrada y salida.

La principal contribución de la clasificación de Dasarathy (1997) es que específica el nivel de abstracción tanto para la entrada como para la salida del proceso de fusión. Sin embargo, no tiene en cuenta en su clasificación la fusión de señales y características, por ejemplo, para refinar el proceso de fusión. En la figura 2.2 se muestra de forma esquemática las cinco divisiones de esta clasificación.

2.4. Clasificación basada en los niveles de abstracción

Luo et al. (2002) utilizan los siguientes cuatro niveles de abstracción para clasificar la fusión de información:

1. **Señal**: La fusión a nivel de señal trata con las señales de los sensores.
2. **Píxel**: La fusión a nivel de píxel opera sobre las imágenes y puede utilizarse para mejorar tareas de procesamiento de la señal.
3. **Característica**: La fusión de características trata con atributos extraídos de las señales o imágenes, como la forma o la velocidad.
4. **Símbolo**: En la fusión a nivel de símbolos, la información es un símbolo que representa una decisión; también es llamado nivel de decisión.

En realidad la fusión de información trata con tres niveles de abstracción (Dasarathy, 1997) (Iyengar et al., 2001): (1) medidas, (2) características y (3) decisiones; y por tanto puede ser clasificada en las siguientes categorías:

1. **Fusión de bajo nivel:** Los datos en bruto son proporcionados como entradas al proceso de fusión, el cual obtiene datos más precisos (reducción del ruido) que las entradas individuales. También es conocido como *signal-level* fusión.
2. **Fusión de nivel medio:** Los atributos o características de una entidad (forma, textura, posición) son fusionados para obtener características que pueden ser usadas para otras tareas. Este tipo de fusión es conocido como fusión a nivel de atributos o características.
3. **Fusión de alto nivel** También conocida como fusión de decisión. Obtiene decisiones o representaciones simbólicas como entrada y las combina para obtener una decisión más precisa. Las aplicaciones de fusión a este nivel suelen utilizar enfoques Bayesianos.
4. **Fusión múltiple:** Cuando el proceso de fusión se lleva a cabo con datos de diferentes niveles de abstracción, por ejemplo, una medida es fusionada con una característica para proporcionar una decisión. La inferencia Dempster-Shafer se suele utilizar para realizar fusión a este nivel.

Los diferentes niveles de fusión descritos se pueden utilizar para proporcionar información a un sistema con diferentes propósitos. La fusión a nivel de señal se puede utilizar en aplicaciones de tiempo real y puede considerarse como un paso adicional en el procesado de la señal. La fusión a nivel de píxel se puede usar para mejorar el rendimiento de muchas tareas de procesado de imágenes como la segmentación y la fusión a nivel de características y de símbolos se puede utilizar para proporcionar características adicionales a un sistema de reconocimiento de objetos para mejorar sus capacidades.

2.5. Clasificación basada en los niveles de fusión del JDL

El modelo JDL es uno de los más populares y conocidos modelos conceptuales dentro de la comunidad de fusión de datos. Fue propuesto originalmente por el U.S. Joint Directors of Laboratory (JDL) y el Departamento de Defensa Americano (DoD) (Hall & Llinas, 1997). Este organismo ha clasificado el proceso de fusión de datos en 5 niveles de procesado, una base de datos asociada y un bus de información que conecta todos los componentes (ver figura 2.3). Los 5 niveles de procesamiento se pueden agrupar en dos grupos: (1) fusión de bajo nivel y (2) fusión de alto nivel. Los 5 niveles de procesamiento definidos por el JDL y los componentes consisten en:

1. **Fuentes:** Las fuentes son las responsables de proporcionar la información de entrada y pueden ser: sensores, información a priori (referencias o información geográfica), bases de datos, entradas humanas, etc..

2. **Interacción persona-ordenador:** Consiste en la interfaz que permite las entradas de los operadores al sistema y del sistema a los operadores; tales como: consultas, comandos, información del resultado de fusión, alarmas, etc.

3. **Sistema de gestión de BBDD:** Este sistema se encarga de almacenar la información proporcionada y el resultado obtenido de la fusión. Es una parte crítica, ya que se encarga de almacenar una gran cantidad de información y de diversa índole.

4. **Nivel 0 - Preprocesado de las fuentes:** Es el nivel más bajo del proceso de fusión. Este nivel de fusión de datos incluye la fusión a nivel de señal y la fusión a nivel de píxel. Si se trata de fuentes de texto, esta etapa además incluye la extracción de información. Este nivel de fusión sirve para reducir la cantidad de los datos y mantener la información útil para los procesos de fusión de alto nivel.

5. **Nivel 1 - Valoración de objetos:** Este nivel de fusión utiliza los datos procesados del nivel anterior. Los procedimientos que son habitualmente necesarios en este nivel son: alineación espacio-temporal, asociación, correlación, técnicas de agrupamiento o clustering, estimación de estado, eliminación de falsos positivos, fusión de la identidad y combinación de las características extraídas de las imágenes (también conocido como fusión a nivel de características). Los resultados finales de esta fase de fusión son la discriminación de objetos (la clasificación y la identificación) y el seguimiento de objetos (estado del objeto y orientación). Esta fase se encarga de transformar la información de entrada en estructuras de datos consistentes. Todos los algoritmos de estimación que se presentan en las secciones 2.7.2 y 2.8.4, se clasifican dentro de esta fase.

6. **Nivel 2 - Valoración de la situación:** El procesado a nivel 2 busca un nivel más alto de inferencia sobre el procesado a nivel 1. De acuerdo con el JDL, la valoración de la situación identifica las situaciones probables dados los datos y eventos observados. Básicamente, se trata de establecer relaciones entre los objetos. Las relaciones (como proximidad, comunicación, etc) son valoradas para determinar el significado de las entidades u objetos en un entorno específico. El objetivo de esta fase incluye realizar inferencia a alto nivel e identificar actividades y eventos significativos (patrones en general). La salida de esta etapa es una colección de inferencias de alto nivel en orden cronográfico que proporcionan una vista de lo que está ocurriendo en el entorno. Es

decir, proporciona una descripción contextual de las relaciones entre los eventos observados y los objetos detectados. En esta fase, es común utilizar información del entorno y conocimiento a priori para identificar situaciones.

7. **Nivel 3 - Valoración del impacto:** Como su nombre indica, en este nivel se valora el impacto de las actividades detectadas en el nivel 2 para obtener una perspectiva propia. Se evalúa la situación actual y se proyecta al futuro para identificar posibles riesgos, vulnerabilidades y oportunidades operacionales. Este nivel incluye: (1) valorar el nivel de peligro o amenaza y (2) predecir el posible resultado lógico. En el caso de que no sea posible predecir las intenciones, se debería proporcionar las diferentes posibilidades. La fase de predicción de los algoritmos de estimación presentados en la sección 2.7.2 y 2.8.4 entran dentro del nivel 3, puesto que una vez identificado un objetivo y prediciendo su posición futura se puede identificar si este representa o no una amenaza. En este nivel principalmente se utilizan técnicas basadas en la teoría de juegos.

8. **Nivel 4 - Refinamiento del proceso de fusión:** El proceso de refinamiento consiste en mejorar los niveles del 0 al 3. En este nivel se lleva a cabo la gestión de recursos, en concreto de los recursos de los sensores. Este nivel conocido también como gestión de sensores tiene como objetivo común conseguir una gestión de recursos eficiente, llevando a cabo: prioridad de tareas, planificación (scheduling), reserva y control de recursos. Es el responsable de monitorizar el rendimiento del sistema y reservar los recursos necesarios de acuerdo a los objetivos específicos.

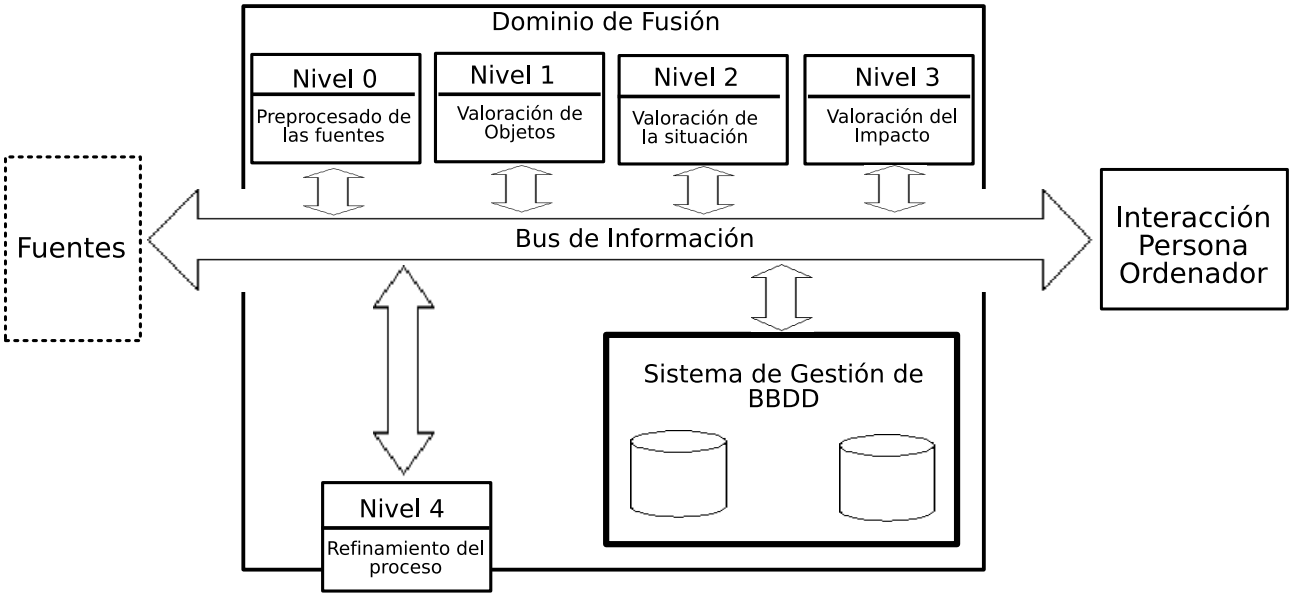

Figura 2.3: Clasificación basada en el modelo JDL [Adaptada de (Hall & Llinas, 1997)].

La fusión de alto nivel (high level fusion) normalmente empieza en el nivel 2, lugar donde los objetos son conocidos, es decir, se conoce el tipo, localización, movimiento y la cantidad de los objetos. La fusión de decisiones, término normalmente utilizado dentro de la comunidad

de fusión, se puede decir que pertenece a la fusión de alto nivel. En la mayoría de los casos la fusión de los niveles 2, 3 y 4 contribuye al proceso de toma de decisiones.

Una de las limitaciones del modelo JDL es que no específica de forma clara el uso de la retroalimentación en el proceso de fusión, es decir, no específica como los resultados actuales o pasados se pueden utilizar para mejorar iteraciones futuras.

Históricamente, el modelo del JDL representa el primer esfuerzo serio de proporcionar un modelo detallado y una terminología común para el dominio de fusión de datos. Sin embargo, al surgir a raíz de las aplicaciones militares, la terminología adoptada está orientada a los riesgos que se producen en ese contexto. El modelo de Dasarathy difiere del modelo JDL en la terminología adoptada y en el enfoque usado. El modelo JDL está orientado principalmente a aplicaciones militares y las tareas de fusión identificadas en el modelo reflejan las peculiaridades de ese dominio. Por otro lado, el modelo de Dasarathy está orientado a las diferencias entre la entrada y la salida independientemente del dominio de fusión. La principal diferencia entre el modelo JDL y el de Dasarathy, es que el primero presenta una perspectiva de fusión adecuada para el diseño de sistemas que incorporan tareas de fusión; mientras que el segundo proporciona una perspectiva de entrada-salida del proceso de fusión, el cual es adecuado para entender las relaciones entre las tareas de fusión y los datos utilizados.

Cada uno de los niveles de procesado de fusión del JDL se puede dividir en subprocesos y para cada uno de los subprocesos existen una serie de técnicas y métodos que se pueden aplicar. En la tabla 2.1 se muestra una posible clasificación de estas técnicas para cada uno de los procesos a partir del nivel 1 y estas técnicas se presentan con más detalle en la sección 2.7 y 2.8.

Tabla 2.1: División de los procesos de cada uno de los niveles del JDL y algunas de las técnicas asociadas. [Adaptada de (Hall & Llinas, 1997)]

Nivel del JDL	Subproceso	Técnicas asociadas
Nivel 1.	Alineación de datos	Transformación de coordenadas Ajustes de unidades
	Correlación de datos/objetos	Vecinos cercanos Técnicas de enventanado MHT, PDA, JPDA
	Estimación de posición y atributos	Filtro de kalman ML, MHT
	Estimación de la identidad del objeto	Modelos físicos Técnicas basadas en características Redes de neuronas Reconocimiento de patrones Clustering
Nivel 2.	Agregación de Objetos Interpretación eventos/actividades Interpretación del contexto	Sistemas basados en el conocimiento Sistemas de Reglas Lógica difusa Redes de neuronas Sistemas blackboard
Nivel 3.	Estimación agregada Predicción de intento Evaluación múltiples perspectivas	Redes de neuronas Sistemas blackboard
Nivel 4.	Evaluación rendimiento Control de Proceso Requisitos de las fuentes Gestión de la misión	Teoría de utilidad Optimización multi-objetivo Modelos de sensores Sistemas basados en conocimiento

2.6. Clasificación basada en los tipos de arquitecturas

Una de las principales decisiones que se deben tomar en los sistemas de fusión de datos es donde realizar la fusión de la información y dependiendo de como se realice el proceso de fusión, las arquitecturas de los sistemas se pueden dividir (ver figura 2.4) en los siguientes 4 grupos:

1. **Arquitectura de fusión centralizada:** En una arquitectura de fusión centralizada, la unidad de fusión se localiza en el procesador central que obtiene toda la información de las distintas fuentes. Por tanto, todos los procesos se ejecutan en el procesador central a partir de los datos en bruto (sin que se realice ninguna modificación) proporcionados por las fuentes. Es decir, las fuentes de datos solo se encargan de obtener los datos en bruto y transmitirlos sin ninguna modificación al procesador central donde se ejecuta el proceso de fusión. En el esquema de fusión centralizada de la figura 2.4 se observa como cada uno de los sensores obtienen la información del entorno. A continuación, la transmiten al nodo de fusión, el cual se encarga de realizar el alineamiento de los datos, su posterior asociación y la estimación. Finalmente, se obtiene como resultado, la fusión del estado estimado del objeto. El enfoque centralizado es teóricamente el óptimo para fusionar datos, asumiendo que la alineación y la asociación se llevan a cabo de forma correcta y que el tiempo para transmitir los datos es despreciable. Sin embargo, estas situaciones no se suelen producir en los sistemas reales.

   Otro problema de este enfoque es que la transmisión por la red de la información en bruto obtenida por los sensores puede requerir de un enorme ancho de banda, siendo un inconveniente para trabajar con imágenes dentro de las redes de sensores visuales. Además los tiempos de propagación de los datos de cada sensor visual son variables y esta diferencia en el retardo de la llegada de las medidas afecta principalmente a la arquitectura de fusión centralizada.

2. **Arquitectura de fusión descentralizada:** Una arquitectura de fusión descentralizada consiste en una red de nodos, donde cada nodo tiene sus propias capacidades de procesamiento. No existe un punto central donde se realiza la fusión, debido a que se realiza en cada nodo basándose en la información propia y en la de los nodos vecinos (Durrant-Whyte & Stevens, 2001). Es decir, los nodos comunican entre ellos la información obtenida y cada uno de ellos a su vez realiza su propia fusión de la información propia y de la información recibida. Por tanto, la fusión se lleva a cabo en cada nodo utilizando la información local y la información de los nodos vecinos. Por lo general, los algoritmos de fusión de datos descentralizados comunican información, utilizando las medidas de Fisher y Shannon en lugar del estado (Durrant-Whyte & Stevens, 2001).

   El mayor inconveniente que tienen este tipo de arquitecturas es el coste de comunicación, que es del orden de $O(n^2)$ siendo n el número de nodos que se comunican, en el caso extremo de que todos los nodos realicen la fusión de las observaciones de todos los sensores; lo cual hace que estos algoritmos no sean fácilmente escalables cuando crece el número de nodos.

3. **Arquitectura de fusión distribuida:** En una arquitectura de fusión distribuida, las medidas de cada sensor son procesadas de forma independiente (por ejemplo, se lleva a cabo la asociación de datos y la estimación del estado) antes de enviar el estado del

objetivo[8] a un procesador central para una fusión posterior con otras fuentes de entrada distribuidas. Es decir, las fuentes obtienen la información y realizan un procesado de la misma antes de que sea enviada, posteriormente esta información se fusiona, en el nodo encargado de realizar la fusión, con la información recibida de las otras fuentes. Por tanto, cada sensor proporciona una estimación del objeto basándose solo en su información local y esta información es la entrada para el proceso de fusión multisensor que proporciona la salida a nivel global. Este tipo de arquitectura proporciona diferentes posibilidades para situar los nodos de fusión, desde un solo nodo que realiza la fusión de los datos a varios nodos de fusión intermedios.

4. **Arquitectura de fusión jerárquica:** Las otras arquitecturas posibles son una combinación de arquitecturas centralizadas, descentralizadas y distribuidas, dando lugar a esquemas de arquitecturas jerárquicas donde la fusión de los datos se lleva a cabo en distintas capas de nodos relacionados jerárquicamente.

De las posibles arquitecturas de fusión descritas anteriormente no existe una única arquitectura que sea mejor que las otras, debido a que la selección de una arquitectura apropiada es una función de la necesidad, de la demanda, de las conexiones existentes, de la disponibilidad de los datos y de la organización del sistema de fusión de datos.

En principio, una arquitectura completamente descentralizada es más difícil de implementar debido a las restricciones de comunicación y computación y a las características espaciales y temporales. Además, los algoritmos robustos de fusión descentralizada que pretenden solucionar estos problemas están siendo investigados hoy en día.

[8] El estado del objetivo enviado por un sensor, es la estimación óptima del estado del objetivo a partir de las observaciones obtenidas por ese sensor.

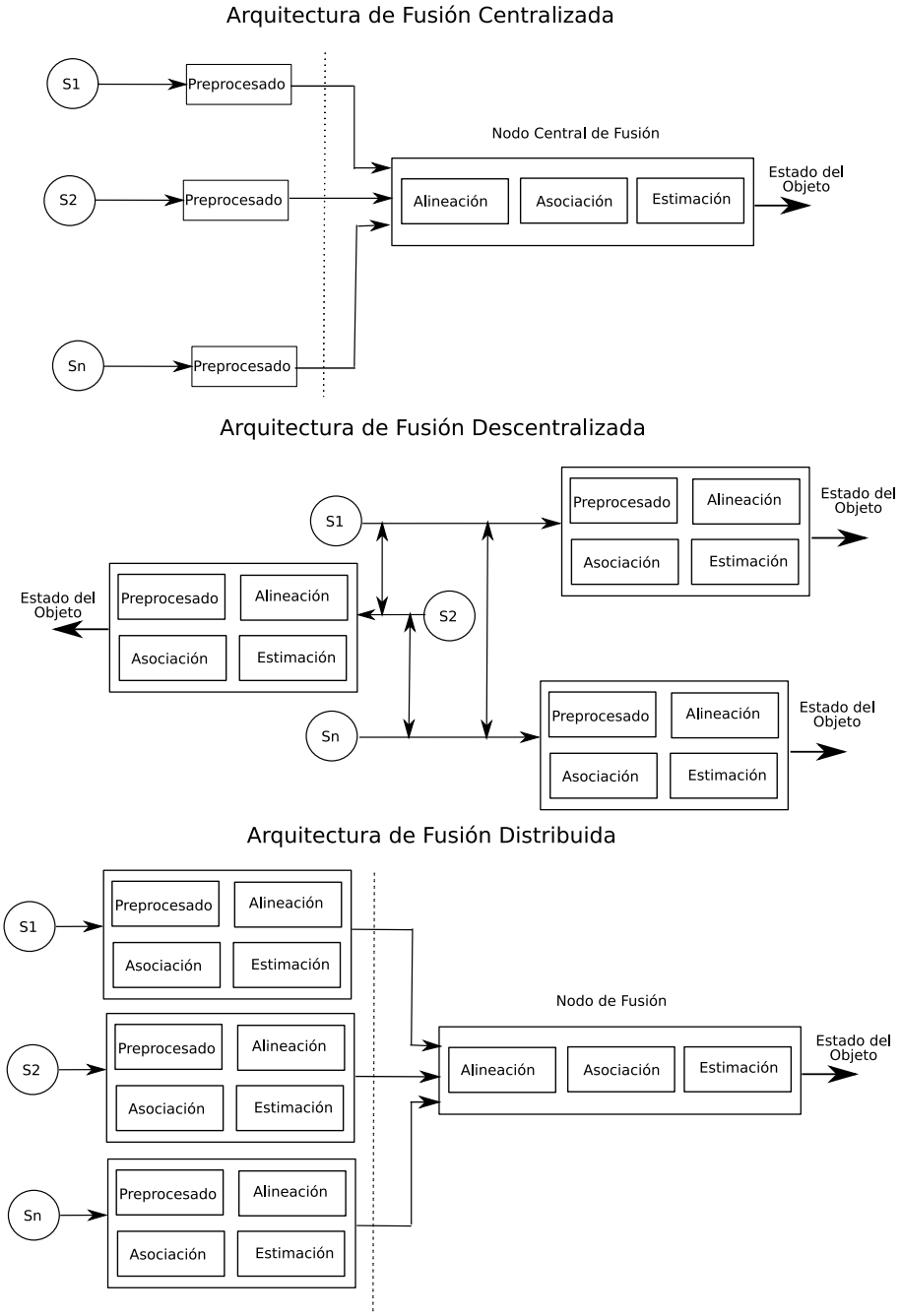

Figura 2.4: Clasificación de los métodos de fusión de datos basada en las diferentes arquitecturas. En la arquitectura centralizada, existe un único nodo que realiza la alineación, la asociación de los datos y la estimación. En la versión descentralizada, cada nodo tiene capacidad de proceso, realizando la alineación, la asociación de los datos y la estimación; además cada nodo produce su propia estimación del estado del objeto utilizando la información de otros nodos. En la versión distribuida se realiza un preprocesado de los datos en cada nodo, pero la fusión de los datos se lleva a cabo en único nodo.

2.6.1. Fusión descentralizada vs Fusión distribuida

La división de las arquitecturas de fusión en arquitectura descentralizada y arquitectura distribuida puede resultar similar y dar lugar a confusión. Es por ello que se dedica esta sub-sección a clarificar sus principales diferencias.

La principal diferencia en los esquemas de fusión distribuida y descentralizada es que en los primeros se realiza un pre-procesado de la información que da lugar a un vector de características, mientras que en la fusión descentralizada se lleva a cabo el proceso de fusión completo en todos y cada uno de los nodos y cada uno de ellos produce una estimación del objeto utilizando la información global. Es decir, en la fusión descentralizada, todos los nodos realizan la fusión con los datos de otras fuentes; mientras que en la fusión distribuida lo que se realiza en cada una de las fuentes es un pre-procesado de los datos en bruto obtenidos y la fusión se realiza en los nodos específicos de fusión, que pueden ser uno o varios.

Los algoritmos de fusión descentralizados suelen comunicar información, definida con las medidas de Fisher o Shannon para estados continuos o discretos respectivamente; mientras que los algoritmos de fusión distribuida intercambian la noción común de estado (posición, velocidad, identidad, etc) junto con sus probabilidades asociadas para fusionar los datos (Manyika & Durrant-Whyte, 1995).

Los algoritmos de fusión descentralizada, al comunicar información (medidas de Fisher y Shannon) en lugar de estados y probabilidades, tienen la ventaja de separar fácilmente el conocimiento nuevo del conocimiento previo. Esto es debido a que la fusión de información es aditiva y por tanto el proceso de fusión es asociativo (no importa el orden en que se realiza) lo que implica que no es tan importante cuando se envía la información y cuando se realiza la fusión. Sin embargo, en los algoritmos de fusión convencionales (como el filtro de Kalman) el estado que se va a fusionar no es asociativo y por tanto importa, cuando y como, se construyen las estimaciones fusionadas. Los sistemas de fusión distribuidos reducen la comunicación y la computación necesaria distribuyendo el trabajo en los diferentes nodos antes de producirse la fusión en el nodo de fusión.

Por otro lado, ya se han comentado las desventajas de la fusión centralizada. Como consecuencia, en este trabajo, se opta por utilizar un enfoque de fusión distribuido y realimentado, el cual se describe detalladamente en el capítulo 5.

2.7. Fusión de datos centralizada: métodos, técnicas y algoritmos

Las diferentes técnicas, métodos y algoritmos de fusión pueden aplicarse a los datos de entrada de forma centralizada o distribuida. En la forma centralizada, todos los datos de las fuentes son recibidos en un único nodo donde se produce la asociación, estimación o fusión de decisiones. Por otro lado, en la forma distribuida existen procesos de fusión que se ejecutan en cada uno de los nodos. Es decir, la información transmitida por los nodos, que obtienen la información directamente de los sensores, se modifica respecto a su estado original.

En esta sección, se presenta el estado actual de los métodos, técnicas y algoritmos de la versión centralizada, dejando la versión distribuida para la sección 2.8. En primer lugar, en el apartado 2.7.1, se presenta el problema de la asociación y el conjunto de técnicas existentes para resolverlo. En segundo lugar, en el apartado 2.7.2 se presenta el problema de la estimación del estado de los objetos y el conjunto de métodos existentes para resolverlo.

Finalmente, en el apartado 2.7.3 se presenta el problema de la fusión de decisiones y las técnicas asociadas.

Debido a la enorme cantidad de trabajos relacionados con fusión de datos, estos apartados no pretenden ser una revisión exhaustiva de todos los trabajos sino solo resaltar los fases más importantes de la fusión de datos y las técnicas y métodos más representativos de cada una de las diferentes fases.

2.7.1. Asociación de datos

En el caso de una arquitectura centralizada, todas las medidas se reciben en el nodo de fusión antes de ser asociadas y posteriormente fusionadas. El problema de la asociación de datos, que consiste en determinar que observaciones han sido generadas por cada objetivo, se puede definir de la siguiente forma:

- Supongamos la situación en la que existen O objetivos que están siendo seguidos por un solo sensor en un entorno denso[9].
- Las observaciones de los sensores se reciben en el nodo de fusión en intervalos de tiempo discretos.
- Cada una de las observaciones está originada por al menos un objetivo.
- Los sensores pueden no proporcionar observaciones en algún intervalo.
- Algunas de las observaciones provienen de los objetivos y otras del ruido.
- Para un objetivo concreto, no se sabe en cada intervalo de tiempo que observaciones han sido generadas por él.

Por tanto, la asociación de datos (ver figura 2.5) consiste en solucionar el problema de determinar el conjunto de datos u observaciones que se refieren al mismo objetivo o evento a lo largo del tiempo. Hall & Llinas (1997), proporcionaron la siguiente definición de asociación de datos: *"El proceso de asignar y calcular el peso que relaciona las observaciones o pistas*[10] *de un conjunto de datos a las observaciones o pistas de otro conjunto de datos"*. En el caso particular de la asociación de datos aplicada a sistemas visuales, dependiendo de donde se realice, puede ser: (1) asociación fotograma[11] a fotograma, (2) observación a observación, (3) observación a pistas o bien (4) pistas a pistas. Como ejemplo de la complejidad de la asociación de datos, en el caso de la asociación de fotograma a fotograma, si asumimos que se detectan M puntos en todos los n fotogramas, el número de posibles conjuntos de pistas es $(M!)^{n-1}$. Entre todas estas posibles soluciones, existe un único conjunto que establece el verdadero movimiento de los M puntos.

La asociación de datos es un paso previo a la estimación del estado de los objetivos detectados. Además es un paso clave, ya que independientemente del algoritmo de estimación

[9] Por entorno denso nos referimos a situaciones donde existen múltiples objetivos dentro del área que se monitoriza, los cuales están muy cerca unos de otros.

[10] Una pista es un conjunto ordenado de puntos que siguen un camino y que han sido generados por el mismo objetivo.

[11] Se denomina fotograma, a una imagen particular dentro de una secuencia de imágenes que componen una animación.

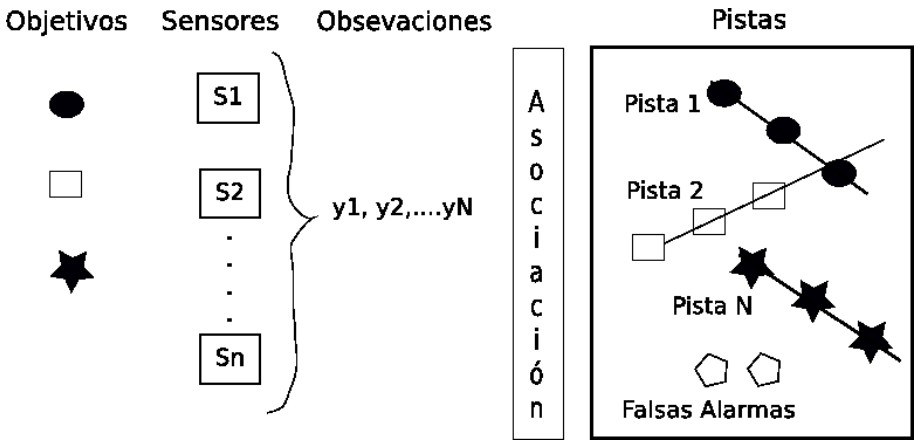

Figura 2.5: Esquema conceptual del proceso de asociación de datos de múltiples sensores y múltiples objetivos. Es necesario establecer las pistas, que consisten en una serie de observaciones en el tiempo, generadas por un mismo objetivo.

o clasificación utilizado, este se comportará de forma errónea si el proceso de asociación no proporciona los datos correctamente asociados. El proceso de asociación está presente a lo largo de todos los niveles de fusión, sin embargo, es distinto dependiendo de las necesidades de cada uno de los niveles de fusión.

En el caso de la asociación de observaciones a pistas, determinar que medidas pertenecen a que pistas es el problema que debe resolver la asociación de datos. Una búsqueda exhaustiva de todas las posibles combinaciones crece exponencialmente con el número de objetivos, haciendo que el problema de asociación sea generalmente NP completo.

A continuación se presenta el estado actual de las técnicas más representativas para la asociación de datos.

Vecinos cercanos

Vecinos cercanos (*nearest-neighbour*), es la técnica de asociación de datos más simple. Este método, consiste en agrupar o seleccionar los datos que están más cercanos unos con otros. Como de cercano está un valor es algo que depende de la métrica de la distancia utilizada y la mayoría de las veces es decidido por el valor de un umbral que está implementado por el diseñador. Este criterio puede estar basado en: (1) una distancia absoluta, (2) una distancia euclídea o (3) una función estadística de distancia.

Es un algoritmo muy simple, pero capaz de encontrar una solución viable en poco tiempo de cálculo; no obstante, en un entorno denso, puede dar lugar a muchos pares con la misma probabilidad y por tanto en una propagación de errores (Blackman, 1990). Además, esta técnica tiene un rendimiento muy pobre en un entorno en el que las medidas falsas ocurren con mucha frecuencia, es decir, en situaciones con mucho ruido.

Todos los vecinos, es una técnica similar, en la cual todas las medidas dentro de una región son incluidas en las pistas. Existen algunas mejoras a los vecinos cercanos como el filtro de vecinos cercanos generalizado (Alouani et al., 1990). También es muy popular la variante de los k-vecinos (conocida también como k-medias) que consiste en agrupar el conjunto de

datos en *k* clusters distintos. El algoritmo de los k-vecinos busca la mejor localización de los centroides de los clusters, donde mejor indica que el centroide se encuentra en el centro del conjunto de datos a los que representa. El algoritmo de los k-vecinos es un algoritmo iterativo (en la figura 2.6 se muestra un ejemplo de ejecución) y se divide en las siguientes fases:

1. Obtiene como entrada el conjunto de datos y el número de agrupaciones o clusters deseados (valor k).
2. De forma aleatoria asigna el valor central de los clusters.
3. Asocia cada dato con el centroide del cluster más cercano.
4. Mueve los centros de los clusters al centroide del conjunto de datos.
5. Vuelve al paso 3 hasta que el algoritmo converge (el centroide no se mueve).

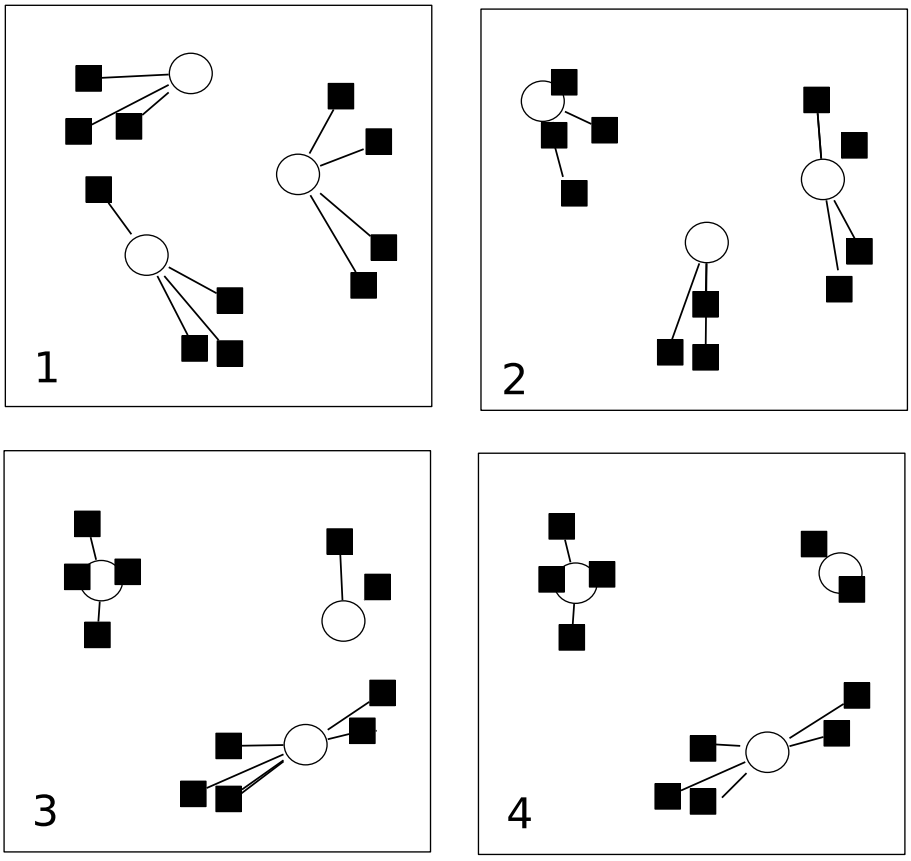

Figura 2.6: Ejemplo de ejecución del algoritmo k-vecinos. (1) Los centroidess de los clusters se establecen de forma aleatoria y cada punto de datos es asignado al centroide del cluster más cercano. (2) Los centroides de los clusters se desplazan a la posición de los centroides de sus puntos. (3) Los puntos de datos se asignan a los centroides de los clusters más cercanos. (4) Los centroides de los clusters se desplazan a la nueva posición que les corresponde.

El algoritmo de los k-vecinos es muy popular y ha sido ampliamente utilizado, pero tiene los siguientes 3 problemas o inconvenientes:

1. El algoritmo no garantiza que encuentra la mejor solución a la localización de los centros de los clusters.

2. Es necesario conocer a-priori el número de clusters y no garantiza que el valor de k elegido por el usuario sea el correcto.

3. El algoritmo presupone que la covarianza en el espacio del conjunto de datos no importa o ya se ha normalizado previamente.

Para solucionar estos problemas existen distintas aproximaciones al algoritmo clásico. Para el primer problema, se puede ejecutar el algoritmo varias veces y después elegir la ejecución que proporcione una menor varianza. En el caso del segundo problema, se suele resolver empezando con un valor de $k = 1$ e incrementandolo hasta que se consiga un buen resultado. Finalmente para solucionar el tercer problema, basta con multiplicar los datos por la inversa de la matriz de covarianza.

Dentro del contexto particular de las redes de sensores visuales, este algoritmo se puede utilizar en los escenarios donde el número de objetivos es conocido de antemano y es siempre el mismo.

Asociación de datos probabilística (PDA)

Propuesto por Bar-Shalom & Tse (1975), el algoritmo, también conocido como filtro modificado de todos los vecinos, asigna una probabilidad (probabilidad de asociación), a cada hipótesis asociada a una medida valida de un objetivo. Las medidas validas hacen referencia a las medidas que caen en la ventana de validación de un objetivo en el instante actual. La ventana de validación, centrada alrededor de la medida predicha del objetivo, para seleccionar el conjunto de medidas válidas se define como:

$$(Z(k) - \hat{z}(k|k-1))^T S^{-1}(k)(z(k) - z(k|k-1)) \leq \gamma \tag{2.1}$$

donde $S(k)$ es la covarianza de la innovación y γ determina el tamaño de la ventana. El conjunto de medidas válidas en el instante k, se define como:

$$Z(k) = z_i(k), i = 1, ..., m_k \tag{2.2}$$

donde $z_i(k)$ es la medida i-esima en la región de validación en el instante k. Las ecuaciones del algoritmo PDA estándar son las siguientes:

Predicción de estado:

$$\hat{x}(k|k-1) = F\hat{x}(k-1|k-1) \tag{2.3}$$

Predicción de medida:

$$\hat{z}(k|k-1) = H\hat{x}(k|k-1) \tag{2.4}$$

Innovación de la medida i-esima:

$$v_i(k) = z_i(k) - \hat{z}(k|k-1) \tag{2.5}$$

Predicción de la covarianza:

$$P(k|k-1) = FP(k-1|k-1)F^T + GQG^T \tag{2.6}$$

Innovación de la covarianza y ganancia de kalman:

$$S(k) = HP(k|k-1)H^T + R \tag{2.7}$$

$$K(k) = P(k|k-1)H^T S(k)^{-1} \tag{2.8}$$

Actualización de la covarianza si las medidas originadas por el objetivo son conocidas:

$$P^o(k|k) = p(k|k-1) - K(k)S(k)K(k)^T \tag{2.9}$$

Actualización total de la covarianza

$$v(k) = \sum_{i=1}^{m_k} \beta_i(k) v_i(k) \tag{2.10}$$

$$P(k|k) = P^o(k|k) + K(k)[\beta_o(k)S(k) + \sum_{i=1}^{m_k}[\beta_i(k)v_i(k)v_i(k)^T] - v(k)v(k)^T]K^T(k) \tag{2.11}$$

donde m_k es el número de devoluciones validas en el instante k.
Actualización de la estimación de estado:

$$\hat{x}(k|k) = \hat{x}(k|k+1) + k(k)v(k) \tag{2.12}$$

Las probabilidades de asociación de PDA son:

$$\beta_i(k) = \frac{p_i(k)}{\sum_{i=0}^{m(k)} p_i(k)} \tag{2.13}$$

donde

$p_i(k) = \lambda(1 - P_d P_g)$ si $i = 0$

$p_i(k) = \frac{P_d}{(2\pi)^{M/2}[S(k)]^{\frac{1}{2}}} exp[-\frac{1}{2} r_i(k)^2]$ si $[\Omega(k)] = 1; i \neq 0$

$p_i(k) = 0$ en otro caso

y

$\lambda = \frac{m_k}{V(k)}, V(k) = \frac{\Pi^{\frac{M}{2}}}{\Gamma(\frac{M}{2+1})}\gamma^M[S(k)]^{\frac{1}{2}}$

$\Omega(k) = 1$ si el valor pertenece a la ventana de validación del objetivo
0 en otro caso

M es la dimensión del vector de estados y λ es la densidad del entorno denso. P_d es la probabilidad de detectar el valor correcto y P_g es la probabilidad de validar un valor detectado.

Es decir, en el algoritmo PDA, una suma ponderada del estado estimado, bajo todas las hipótesis, asociando diferentes medidas a un objetivo particular sirve como la estimación del estado del objetivo. Es decir, la asociación de las diferentes observaciones a un objetivo concreto, le sirve a PDA para estimar el estado del objetivo. Por tanto, las probabilidades de asociación se utilizan como pesos.
Las principales desventajas de PDA son:

- Pérdida de pistas: Debido a que PDA ignora las interferencias de otros objetivos, a veces puede clasificar erróneamente pistas de objetivos muy cercanos. Por tanto, presenta un rendimiento pobre cuando los objetivos se cruzan o cuando los objetivos están cerca entre ellos.
- Aproximación Bayesiana Subóptima: Cuando la fuente de medición de los datos es incierta, debido a las multitudes o a las perdidas de datos, PDA es una aproximación Bayesiana subóptima al problema de asociación.
- Un único objetivo: PDA está diseñado para la asociación de un único objetivo en un entorno denso. El número de falsas alarmas normalmente se modela utilizando una distribución de Poisson y las falsas alarmas se presupone que se distribuyen de manera uniforme en el espacio. Cuando hay múltiples objetivos en la zona bajo vigilancia, PDA no se comporta bien debido a que el modelo de falsas alarmas no funciona bien.
- Necesidad de proporcionar de forma separada algoritmos de inicialización y eliminación de pistas: El algoritmo PDA asume que una pista se ha establecido y por tanto no proporciona un algoritmo de inicialización y eliminación.
- Principalmente bueno para el seguimiento de objetivos que no realizan cambios bruscos de movimiento en entornos densos. Si el objetivo realiza cambios bruscos de movimiento, PDA seguramente perderá al objetivo.

Asociación de datos probabilística conjunta (JPDA)

JPDA fue diseñado por Fortmann et al. (1980) como un enfoque "subóptimo" para el seguimiento de múltiples objetivos en entornos densos. JPDA es similar a PDA, con la diferencia de que las probabilidades de asociación se calculan usando todas las observaciones y todas las pistas. Es decir, a diferencia de PDA, considera simultáneamete varias hipótesis y las combina, busca determinar la probabilidad $\beta_i^t(k)$ de que la medida i es originada por el objetivo t, teniendo en cuenta que bajo esa hipótesis la medida no puede ser originada por los otros objetivos, afectando a los pesos calculados de cada pista. Por tanto, para un número de objetivos conocido evalúa las probabilidades de asociación medida-objetivo (para el último conjunto de medidas) y las combina en la estimación de estado correspondiente.

Si la probabilidad de asociación $\beta_i^t(k)$ es conocida, la ecuación del filtro de kalman de actualización de la pista t se puede escribir como:

$$\hat{x}^t(k|k) = \hat{x}^t(k|k-1) + K(k)\bar{v}^t(k) \tag{2.14}$$

donde $\hat{x}^t(k|k)$ y $\hat{x}^t(k|k-1)$ son respectivamente la estimación y la predicción del estado del objetivo t. $K(k)$ es la ganancia del filtro y una suma ponderada de los residuos asociados con la observación $m(k)$ del objetivo t es:

$$\bar{v}^t(k) = \sum_{i=1}^{m(k)} \beta_i^t(k) v_i^t(k) \tag{2.15}$$

donde $v_i^t(k) = z_i(k) - Hx^t(k|k-1)$. Por tanto, incorpora todas las observaciones dentro del vecindario de la posición predicha del objetivo para actualizar la posición estimada, utilizando

una probabilidad a posteriori, que es una suma ponderada de los residuos. Igualmente que PDA, JPDA lleva a cabo la asociación solo en el último fotograma utilizando pistas establecidas y por tanto no es válido para realizar una inicialización de las pistas, haciendo necesario su utilización de forma conjunta con un algoritmo de inicialización de pistas.
Por un lado, las principales restricciones de JPDA son:

- Una medida no puede provenir de dos objetivos.
- No pueden provenir dos medidas del mismo objetivo.
- $\sum_{i=0}^{m(k)} \beta_i^t(k) = 1$. La suma de las probabilidades de todas las medidas asignadas a un objetivo debe ser 1.

Por otro lado, las mayores desventajas de JPDA son:

- La falta de un mecanismo explícito de inicialización de pistas. Al igual que PDA, JPDA es incapaz de inicializar nuevas pistas o eliminar pistas que dejan de estar en el campo visual.
- Se trata de un algoritmo computacionalmente intensivo en entornos con multitud de objetivos, puesto que se incrementa de forma exponencial el número de posibles hipótesis cuando se incrementan los objetivos.

Además, debido a que JPDA lleva a cabo la asociación de datos utilizando un número fijo de objetivos entre dos fotogramas, pueden producirse errores si existe un cambio en el número de objetivos existentes. JPDA es más apropiado que MHT para situaciones en las cuales las densidades de los falsos objetivos son altas (como en el caso de las aplicaciones de sonar).

Test de múltiples hipótesis (MHT)

La idea que subyace bajo el test de múltiples hipótesis (MHT) está basada en el uso de más de 2 fotograma para realizar la correspondencia con unos mejores resultados puesto que si la correspondencia se establece utilizando solo dos fotogramas consecutivos, existe una gran probabilidad de cometer un error en la asociación. A diferencia de PDA y JPDA, MHT evalúa en cada iteración todas las posibles hipótesis y las nuevas hipótesis se mantienen.

MHT fue desarrollado para realizar el seguimiento de múltiples objetivos en entornos densos, por tanto combina el problema de la asociación y seguimiento en un marco de trabajo unificado, siendo también una técnica de estimación (a diferencia de PDA y JPDA). Para el cálculo de las hipótesis de MHT normalmente se usa la regla de Bayes o las redes Bayesianas. Generalmente los investigadores han reivindicado el mejor comportamiento de MHT frente a JPDA para densidades bajas de falsos positivos. El principal inconveniente de MHT es la sobrecarga computacional cuando se incrementan los objetivos o los falsos positivos. Esta limitación se intenta solventar realizando la poda de las hipótesis por medio de una ventana de tamaño fijo.

El algoritmo de seguimiento de Reid (1979) está considerado como el algoritmo MHT estándar. MHT es un algoritmo iterativo, una iteración comienza con un conjunto de hipótesis de correspondencia. Cada hipótesis es una colección de pistas disjuntas, y para cada hipótesis, se realiza la predicción del objetivo en el siguiente instante. Las predicciones a continuación

son comparadas con las nuevas observaciones evaluando una medida de distancia. El conjunto de asociaciones establecida para cada hipótesis, las cuales están basadas en la distancia, introduce nuevas hipótesis en la siguiente iteración. Cada nueva hipótesis representa un nuevo conjunto de pistas basadas en las observaciones actuales.

Hay que resaltar que cada medida puede venir de: (1) la entrada de un nuevo objetivo en el campo visual, (2) un objetivo en seguimiento o (3) del ruido en la toma de la medida. Además, es posible que una medida no se asigne a un objetivo, porque el objetivo haya desaparecido, o la medida correspondiente a ese objetivo no se obtenga. Esto puede ocurrir, bien porque el objetivo presente una oclusión, o bien porque se produzca un error en la detección.

MHT mantiene varias hipótesis de correspondencia para cada objetivo en cada fotograma. El algoritmo genera una serie de hipótesis candidatas basadas en las medidas recibidas y evalúa esas hipótesis según va recibiendo más datos.

Si la hipótesis en el instante k se representa por $H(k) = [h_l(k), l = 1, ..., n]$, entonces la probabilidad de una hipótesis $h_l(k)$ se puede representar de forma recursiva utilizando la regla de Bayes:

$$\begin{aligned} P(h_l(k)|Z(k)) &= P(h_g(k-1), a_i(k)|Z(k)) \\ &= \frac{1}{c} P(Z(k)|h_g(k-1), a_i(k)) \\ &* P(a_i(k)|h_g(k-1)) * P(h_g(k-1)) \end{aligned} \tag{2.16}$$

donde, $h_g(k-1)$ es la g-esima hipótesis del conjunto de hipótesis hasta el instante $k-1$; $a_i(k)$ es la i-esima posible asociación de detección a pista del objeto; $Z(k)$ es el conjunto de detecciones del fotograma actual y c es una constante normalizadora.

El primer término de la parte derecha de la ecuación, es la función de verosimilitud del conjunto de medidas $Z(k)$, dada la prioridad conjunta y las hipótesis actuales. El segundo término, es la probabilidad de las hipótesis de asociación de datos actuales dadas las hipótesis anteriores $h_g(k-1)$. El tercer término, es la probabilidad de las hipótesis previas a partir de las cuales la hipótesis actual es calculada.

El algoritmo MHT, tiene la característica de detectar una nueva pista manteniendo la estructura del árbol de las hipótesis. La probabilidad de una verdadera pista se proporciona por medio del modelo de decisión de Bayes como:

$$P(\lambda|Z) = \frac{P(Z|\lambda) * P_o(\lambda)}{P(Z)} \tag{2.17}$$

donde $P(Z|\lambda)$ es la probabilidad de recibir el conjunto de datos de medidas Z, dado que es verdad que la señal origen λ está presente. $P_o(\lambda)$ es la probabilidad a priori de que la señal origen se produzca y $P(Z)$ es la probabilidad de recibir el conjunto de detecciones Z.

MHT considera todas las posibilidades, no solo el mantenimiento de la pista, sino también la inicialización de las pistas y la eliminación en un marco integrado. Evalúa las posibilidades de que haya un objetivo a partir de la generación de una secuencia de medidas de una forma exhaustiva y además, el algoritmo no asume un número de objetivos conocido.

El principal inconveniente de este algoritmo, es que a medida que aumenta el número de pistas y observaciones la complejidad computacional crece exponencialmente. Por tanto,

su implementación práctica es limitada ya que es exponencial computacionalmente tanto en tiempo como en memoria.

Para reducir la sobrecarga computacional, Streit & Luginbuhl (1994) propusieron un MHT probabilístico donde las asociaciones son consideradas variables aleatorias estadísticamente independientes y por tanto no existiría la necesidad de una enumeración exhaustiva, el algoritmo se conoce como PMHT. El método PMHT asume que el número de objetivos y las medidas son conocidas.

Con el mismo objetivo de reducir la sobrecarga computacional, Cox & Hingorani (1996) presentaron una implementación eficiente del algoritmo MHT. Esta implementación, además es la primera propuesta de uso del algoritmo MHT en el seguimiento visual. Utilizaron el algoritmo de Murty (1968) para determinar las mejores k hipótesis en tiempo polinómico con el objetivo realizar el seguimiento de puntos de interés.

MHT normalmente realiza el seguimiento basándose en una sola característica, que suele ser la posición. Una combinación Bayesiana para utilizar múltiples características fue propuesta por Liggins et al. (1997).

Recientemente, Joo & Chellappa (2007), han propuesto un algoritmo de asociación para el seguimiento visual de múltiples objetivos. Su algoritmo se basa en una modificación de MHT en el que no imponen la restricción de asociar una medida con un único objetivo, sino que permiten que una medida sea asociada con varios objetivos y varios objetivos con con una sola medida. Además, proponen un algoritmo de optimización combinatoria para generar solo la mejor hipótesis de asociación. A diferencia de otros métodos de asociación, que son aproximados, su algoritmo encuentra la mejor hipótesis.

Otras técnicas

Grafos de asociación. La teoría de grafos es una materia antigua dentro las ciencias de la computación y se utiliza en numerosas aplicaciones. Uno de los campos en los que se ha usado la teoría de grafos es el problema de la asignación de tareas, el cual está relacionado con los problemas de asociación de datos. La solución al problema de asociación de datos utilizando grafos implica trazar el grafo previamente. Los algoritmos de asociación basados en grafos se resumen en los siguientes pasos:

1. Construir el grafo de asociación. Un grafo de asociación consiste en una serie de vértices o puntos y un conjunto de enlaces entre los vértices.

2. Podar los enlaces por medio de una asociación fuerte. Una asociación fuerte implica eliminar los valores que distan de una forma significativa (utilizando por ejemplo técnicas como vecinos cercanos).

3. Calcular el valor de la asociación local de los enlaces que quedan. El valor de la asociación local consiste en el valor del enlace entre dos vértices.

4. Cortar los enlaces por medio del valor del umbral de la asociación local. Todos aquellos enlaces por debajo del umbral definido, son eliminados.

5. Calcular el valor de asociación global. El valor de asociación global se calcula como la diferencia entre el máximo de todos los enlaces y el valor local del enlace.

6. Cortar los enlaces que se encuentren por debajo del valor del umbral de asociación global.

Un enfoque para resolver el problema de asociación utilizando grafos es tener en cuenta una secuencia finita de conjuntos de medida, crear un grafo que represente las distintas asociaciones de un paso a a otro y finalmente extraer el conjunto óptimo de caminos del grafo. Medioni et al. (2001) utilizan un grafo donde el coste del camino es una función de la similitud entre los nodos y la longitud del camino. Una solución similar es proporcionada por Chia & Huang (2006), donde la representación del grafo implícitamente mantiene múltiples hipótesis teniendo en cuenta los posibles caminos. Chen et al. (2001) describen un método que utiliza asociación de grafos bipartitos. Un grafo se define como bipartito, si puede ser coloreado sin conflictos utilizando solo dos colores. El método propuesto por Chen et al. (2001) busca la asociación conjunta óptima bajo la consideración uno-a-uno. El método propuesto por Han et al. (2004) también puede verse como un enfoque basado en grafos, ya que múltiples grafos que representan diferentes conjuntos de posibles caminos se mantienen a lo largo del tiempo.

Los técnicas basadas en grafos consideran múltiples hipótesis de asociación, pero se basan en técnicas ad-hoc de búsqueda de las trayectorias óptimas.

Redes de neuronas. El principal motivo para utilizar redes de neuronas es el hecho de que la complejidad de calcular las probabilidades de asociación ($\beta_i^t(k)$) se reduce. Se utilizan principalmente dos tipos de redes de neuronas para resolver la asociación de datos: (1) las redes de Boltzmann (Iltis & Ting, 1991, 1993) y (2) las redes de Hopfield (Sengupta & Iltis, 1989) (Leung, 1996) .

Iltis & Ting (1991) demostraron que el uso de una red de Boltzmann, con elementos booleanos simples, puede estimar las probabilidades de asociación. La calidad de sus resultados son similares a los del algoritmo JPDA pero con una menor carga computacional. Iltis & Ting (1993), también demostraron que usando suficientes maquinas Boltzmann paralelas las probabilidades de asociación se pueden calcular con menores errores.

El uso de una red de neuronas de Hopfield para calcular las probabilidades de asociación ($\beta_i^t(k)$) puede verse como un problema de optimización con restricciones (Leung, 1996). Las restricciones se obtienen con una evaluación cuidadosa de las propiedades de las reglas de asociación del algoritmo JPDA. El problema se formula de forma parecida al problema del viajante (TSP), es decir, la función de energía de la asociación de datos es similar a TSP. En la asociación de datos, minimizando la función de energía se espera que las probabilidades $\beta_i^t(k)$ se calculen de forma más precisa y con una menor carga computacional.

Winter et al. (1999), también han propuesto el uso de redes neuronales aplicadas al problema de asociación de datos. En su red de neuronas basada en Hopfield, siempre se encuentra una solución óptima el 17.4 % del tiempo y el resto del tiempo se encuentra una solución que se aproxima.

Cuando se utilizan redes de neuronas para realizar la asociación de datos y se incrementa el número de objetivos, es necesario incrementar también el número de neuronas de entrada.

Las ventajas de las redes de neuronas es que son capaces de aproximar cualquier función. Las principales desventajas son que: (1) son lentas para entrenar algunas funciones, (2) tienden al sobre ajuste de los pesos a los datos de entrada y (3) es imposible determinar los criterios que han utilizado para realizar el aprendizaje.

Lógica difusa. La lógica difusa (Novák et al., 1999) también ha sido utilizada en la fase de asociación de datos. La desventaja de usar la asociación de datos probabilística conjunta (JPDA) y la asociación de datos probabilística (PDA), puesto que el tiempo computacional aumenta exponencialmente a medida que aumentan los objetivos, ha llevado al uso de la lógica difusa para esta tarea (Aziz et al., 1999).

Stover et al. (1996), desarrollaron una arquitectura general basada en lógica difusa para la fusión autónoma de datos. También, García et al. (2005) utilizaron la lógica difusa para un sistema de seguimiento con vídeo en entornos aeroportuarios. El método propuesto calcula los intervalos de confianza que se utilizan posteriormente para ponderar la contribución de cada detección a la actualización de la pista. El conocimiento del dominio se representa en forma de reglas para calcular los pesos y las reglas se extraen previamente de situaciones predefinidas (ejemplos) y de esta forma se lleva a cabo un proceso inductivo de generalización. El inconveniente de este método es que necesita suficientes ejemplos para caracterizar todas las posibles situaciones que pueden darse. Por ello es útil cuando se utiliza en entornos controlados y con reglas definidas como los aeroportuarios.

Lefebvre & Helleur (2002) proponen el uso de un sistema basado en lógica difusa para resolver problemas de asociación en condiciones de baja observabilidad y alta densidad de blancos para una aplicación de vigilancia marítima.

La ventaja del uso de un sistema difuso, en el problema de asociación de datos, es que permite ponderar la asignación a las pistas de forma que sea posible asignar parcialmente a varias pistas. Se trata de un método de asociación de datos suave, el cual considera todas las posibilidades de asociación de una pista sin tomar decisiones de asociación excluyentes. Otra ventaja de utilizar un enfoque de lógica difusa para fusión de datos es una mejora en el coste computacional y una mayor flexibilidad de diseño.

Resumen de las técnicas de asociación de datos

Tabla 2.2: Resumen de las técnicas de asociación de datos más utilizadas, mostrando las ventajas e inconvenientes de cada una de ellas

Técnica	**Ventajas**	**Inconvenientes**
Vecinos cercanos (NN, k-NN)	Simple Solución en poco tiempo	Rendimiento bajo en entornos densos y con mucho ruido
PDA	Objetivos sin cambios bruscos en entornos densos	Necesita algoritmos de gestión de pistas No considera el cruce de los objetivos
JPDA	Entornos densos con tasas altas de falsos objetivos	Necesita algoritmos de gestión de pistas Computacionalmente intensivo
MHT (PMHT)	Múltiples objetivos en entornos densos y densidades bajas de falsos objetivos	Computacionalmente intensivo
Grafos de asociación	Consideran múltiples hipótesis de asociación	Se basan en técnicas ad-hoc Hay que construir el grafo
Redes de neuronas	Menor carga computacional	Es necesario construir la red
Lógica difusa	Flexibilidad Menor carga computacional	Crear las funciones de pertenencia

2.7.2. Estimación del estado

Las técnicas de estimación del estado o seguimiento pretenden solucionar el problema de determinar el valor del estado (por lo general es la posición de un objetivo), dadas las observaciones o medidas de un objetivo en movimiento. No se garantiza que las medidas sean relevantes, en el sentido en que algunas pueden venir del propio objetivo y otras de otros objetivos o de ruido. La estimación del estado es una componente de cualquier sistema de fusión desde el momento en que las observaciones pueden venir de sensores o fuentes distintas.

El problema de estimación implica buscar los valores del vector de estado (posición, velocidad, etc.) que mejor encajen, en un sentido matemático, con los datos observados. Desde un punto de vista matemático, se tienen un conjunto de observaciones redundantes y se intenta buscar el valor del conjunto de parámetros que proporcionan un mejor ajuste con respecto a los datos observados. En general, las observaciones están corrompidas por errores de medida, propagación del ruido y otros factores, además, es posible disponer o no de una distribución estadística apriori del error de las fuentes de observación. Los métodos de estimación del estado se encuadran dentro del nivel 1 (valoración de objetos) de la jerarquía del JDL y pueden dividirse en dos grandes grupos:

1. **Medidas y dinámica lineal:** En donde el problema de estimación tiene una solución estándar. En concreto, cuando las ecuaciones del estado del objetivo y las medidas son

lineales, el ruido es Gaussiano y no se trata de entornos densos, la solución analítica óptima está basada en el filtro de Kalman.

2. **Dinámica no lineal:** El problema de estimación se vuelve difícil y una existe una solución analítica única para resolverlo de una forma general. En principio, no existe una solución satisfactoria para este problema.

Los métodos de estimación provienen de la teoría de control y utilizan las leyes de la probabilidad para calcular un vector de estados a partir de un vector de medidas o una secuencia de vectores de medidas.
A continuación se presentan los métodos de estimación más conocidos y utilizados: Máxima verosimilitud (2.7.2), Máximo a posteriori (2.7.2), Filtro de Kalman (2.7.2), Filtros de partículas (2.7.2) y por último, diferentes técnicas (sistemas basados en reglas, redes de neuronas y lógica difusa) que han sido agrupadas bajo la denominación de técnicas de inteligencia artificial (2.7.2).

Máxima verosimilitud (ML)

La técnica de estimación basada en la máxima verosimilitud (o máxima probabilidad) consiste en un método de estimación basado en probabilidades. Los métodos de estimación basados en probabilidades son apropiados cuando el estado que se va a estimar no es el resultado de una variable aleatoria (Brown et al., 1992).

En el contexto de fusión de información, siendo x el estado que se está estimando y $z = (z(1), ...z(k)))$ una secuencia de k observaciones de x, la función verosimilitud $\lambda(x)$ se define como la función de densidad de probabilidad de la secuencia de las z observaciones dado el verdadero valor del estado x:

$$\lambda(x) = p(\mathbf{z}|x) \tag{2.18}$$

El estimador de máxima verosimilitud busca el valor de x que maximiza la función de verosimilitud:

$$\hat{x}(k) = argmax_x p(\mathbf{z}|x) \tag{2.19}$$

el cual puede ser obtenido de modelos analíticos o empíricos de los sensores. El principal inconveniente de este método para su aplicación práctica, es que requiere el conocimiento de modelos analíticos o empíricos de los sensores para obtener la distribución de los datos.

Máximo a posteriori (MAP)

El método de máximo a posteriori está basado en la teoría Bayesiana. Se utiliza cuando el parámetro x a ser estimado es la salida de una variable aleatoria con una función de densidad de probabilidad $p(x)$ conocida.

En el contexto de fusión de información, siendo x el estado que se está estimando y $z = (z(1), ..., z(k))$ una secuencia de k observaciones de x, el estimador Máximo a posteriori busca el valor de x que maximiza la función de distribución posterior:

$$\hat{x}(k) = argmax_x p(x|\mathbf{z}) \tag{2.20}$$

Ambos métodos (MAP y ML) intentan encontrar el valor más probable para el estado x. Sin embargo, el primer método asume que x es un punto fijo aunque desconocido del espacio de

parámetros, mientras que el último, considera x como la salida de una variable aleatoria con función de densidad de probabilidad conocida a priori. Estos dos métodos son equivalentes cuando no hay información a priori acerca de x, solo las medidas.

Filtro de Kalman

El filtro de Kalman, es sin lugar a dudas, uno de los métodos de estimación más populares. Fue propuesto originalmente por Kalman (1960) y ha sido muy estudiado y utilizado desde entonces. Dentro del contexto de la fusión de varios sensores proporciona un enfoque de estimación clásico.

El filtro de Kalman estima el estado x de un proceso en tiempo discreto gobernado por el siguiente modelo de espacio-tiempo:

$$x(k+1) = \Phi(k)x(k) + \mathbf{G}(k)u(k) + w(k), \tag{2.21}$$

con las medidas u observaciones z representadas por

$$z(k) = \mathbf{H}(k)x(k) + v(k) \tag{2.22}$$

donde: $\Phi(k)$ es la matriz de transición de estados, $\mathbf{G}(k)$ es la matriz de transición de entradas, $u(k)$ es el vector de entrada, $\mathbf{H}(k)$ es la matriz de medidas, w y v son variables aleatorias que obedecen a leyes Gaussianas de media cero con matrices de covarianza $\mathbf{Q}(k)$ y $\mathbf{R}(k)$ respectivamente.

Basándose en las medidas $z(k)$ y en los parámetros del sistema, la estimación de $x(k)$, representada por $\hat{x}(k)$, y la predicción de $x(k+1)$ representada por $\hat{x}(k+1|k)$ son

$$\hat{x}(k) = \hat{x}(k|k-1) + \mathbf{K}(k)[z(k) - \mathbf{H}(k)\hat{x}(k|k-1)] \tag{2.23}$$

$$\hat{x}(k+1|k) = \Phi(k)\hat{x}(t|t) + \mathbf{G}(k)u(k) \tag{2.24}$$

respectivamente, donde $\mathbf{K}$ es la ganancia del filtro determinada por

$$\mathbf{K}(k) = Pr(k|k-1)\mathbf{H}^T(k)[\mathbf{H}(k)P(k|k-1)\mathbf{H}^T(k) + \mathbf{R}(k)]^{-1} \tag{2.25}$$

donde: $P(k|k-1)$ es la matriz de covarianzas de la predicción que puede ser determinada por

$$P(k+1|k) = \Phi(k)Pr(k)\Phi^T(k) + \mathbf{Q}(k) \tag{2.26}$$

con

$$P(k) = P(k|k-1) - \mathbf{K}(k)\mathbf{H}(k)P(k|k-1) \tag{2.27}$$

El filtro de Kalman principalmente se utiliza para fusionar datos redundantes de bajo nivel. Si el sistema se puede describir como un modelo lineal y el error se puede modelar como ruido Gaussiano, el filtro de Kalman recursivo obtiene estimaciones estadísticas óptimas (Luo & Kay, 1992). Sin embargo, para tratar con modelos dinámicos no lineales y medidas no lineales, son necesarios otros métodos. La modificación del filtro de Kalman, conocida como Extended Kalman Filter (EKF) es un enfoque conocido para implementar filtros no lineales recursivos (Welch & Bishop, 2001). El filtro EKF es una de las herramientas más utilizadas para fusionar datos en la navegación de robots. Sin embargo, no está exento de problemas debido a que

calcular los jacobianos es computacionalmente muy costoso. Algunos intentos para suavizar el coste computacional, como la linearización, introducen errores en el filtro y lo pueden hacer inestable.

Más recientemente, el Unscented Kalman Filter (UKF) (Julier & Uhlmann, 1997a) ha ido ganando popularidad debido a que no tiene un paso de linearización y los errores asociados (Wan & Van Der Merwe, 2000) del filtro EKF. El filtro UKF utiliza una técnica de muestreo determinista para escoger el conjunto mínimo de puntos alrededor de la media. Estos puntos capturan de forma completa la verdadera media y covarianza. A continuación, estos puntos se propagan a través de funciones no lineales y la covarianza de la estimación se puede recuperar. Además, otra ventaja del filtro UKF es que es muy apropiado para implementaciones paralelas.

Filtros de partículas

Los filtros de partículas son implementaciones recursivas de los métodos secuenciales de Monte Carlo (Crisan & Doucet, 2002). Los filtros de partículas construyen la función de densidad de probabilidad posterior basándose en un gran número de muestras aleatorias llamadas partículas. Las partículas se van propagando en el tiempo, combinando secuencialmente los pasos de muestreo y remuestreo. En cada paso, el remuestreo se utiliza para descartar algunas partículas, incrementando la relevancia de regiones con una mayor probabilidad posterior. En el proceso de filtrado, múltiples partículas de la misma variable de estado son utilizadas y cada partícula tiene un peso asociado que indica la calidad de la partícula. Por tanto, la estimación es el resultado de la suma ponderada de todas las partículas. El algoritmo del filtro de partículas tiene dos fases: (1) predicción y (2) actualización. En la fase de predicción, cada partícula es modificada conforme al modelo existente incluyendo la suma de ruido aleatorio para simular el efecto del ruido. Después, en la fase de actualización, el peso de cada partícula es reevaluada basándose en la última información de los sensores disponible, por tanto las partículas con menores pesos son eliminadas.

Un filtro de partículas genérico consta de las siguientes fases:

1. Inicialización de las partículas:
   - N es igual al número de partículas
   - $X^{(i)}(1) = [x(1), y(1), 0, 0]^T$ para $i = 1, ..., N$.
2. Fase de predicción:
   - Para cada partícula $i = 1, ..., N$ evaluar el estado $(k+1|k)$ del sistema usando el estado en el instante k con el ruido del sistema en el instante k.

     $$\hat{X}^{(i)}(k+1|k) = F(k)\hat{X}^{(i)}(k) + (ruido - distribucion - cauchy)_{(k)} \quad (2.28)$$

     $F(k)$ es la matriz de transiciones del sistema
3. Evaluación del peso de las partículas:
   Para cada partícula $i = 1, ..., N$
   - Calcular la observación predicha del estado del sistema usando la predicción del estado actual del sistema y el ruido en el instante k.

     $$\hat{z}^{(i)}(k+1|k) = H(k+1)\hat{X}^{(i)}(k+1|k) + (ruido - medida - gaussiano)_{(k+1)} \quad (2.29)$$

- Calcular la verosimilitud (pesos) de acuerdo a la distribución proporcionada.

$$likelihood^{(i)} = N(\hat{z}^{(i)}(k+1|k); z^{(i)}(k+1), var) \tag{2.30}$$

- Normalizar los pesos

$$\tilde{w}^{(i)} = \frac{likelihood^{(i)}}{\sum_{j=1}^{N} likelihood^{(j)}} \tag{2.31}$$

4. Remuestreo/Selección: Multiplicar las partículas con mayor peso y suprimir las partículas con menor peso. Es necesario modificar el estado actual usando los pesos de las nuevas partículas calculadas.

   - Calcular los pesos acumulados

$$CumWt^{(i)} = \sum_{j=1}^{i} \tilde{w}^{(j)} \tag{2.32}$$

   - Generar variables aleatorias distribuidas uniformemente entre $U^{(i)} \sim U(0,1)$ con el número de pasos igual al número de partículas
   - Determinar que partícula multiplicar y cual eliminar.

5. Fase de propagación:

   - Incorporar los nuevos valores del estado después del remuestreo del instante k para calcular el instante k+1.

$$\hat{x}^{(1:N)}(k+1|k+1) = \hat{x}(k+1|k) \tag{2.33}$$

   - Calcular la media posterior

$$\hat{x}(k+1) = mean[x^{i}(k+1|k+1)], i = 1, ..., N \tag{2.34}$$

   - Repetir los pasos del 2 al 5 para cada instante de tiempo

Los filtros de partículas son más flexibles que los filtros de kalman y son capaces de tratar con dependencias no lineales y densidades no Gaussianas en el modelo dinámico y en el error de los datos. A pesar de que los filtros de partículas han demostrado ser muy útiles en aplicaciones de estimación, no están exentos de problemas. Una cuestión importante tiene que ver con el número de partículas. Es necesario un gran número de partículas para que la varianza del estimador sea lo suficientemente pequeña como para ser aceptable. Por tanto, es muy difícil, decidir cuantas partículas se requieren para producir estimaciones útiles y se suele resolver de forma experimental. Las primeras versiones del filtro de partículas utilizaban un número fijo de partículas, sin embargo, recientemente se han publicado trabajos con modificaciones del filtro que se basan en un número de partículas variable (Martinez-del Rincon et al., 2007).

Técnicas de inteligencia artificial

La fusión de datos con métodos estadísticos se basa en técnicas conocidas como los filtros de kalman o las estadística Bayesiana. Cuando no existe un modelo específico de la incertidumbre, pueden ser más apropiadas otras técnicas basadas en inteligencia artificial (Smith & Singh, 2006). Dentro de estas técnicas, las más utilizadas son (1) los sistemas basados en reglas, (2) las redes neuronales y (3) la lógica difusa.

Sistemas basados en reglas. Son una de las formas más simples que existen para que un sistema actúe de forma inteligente. Básicamente consisten en un conjunto de reglas de tipo [si...entonces]. Por ejemplo, si la velocidad del viento es mayor que 100 km/h, entonces activa la alarma. Son sistemas que funcionan bien si consiguen anticipar todas las situaciones, sin embargo son sistemas que requieren de un proceso de ingeniería costoso. En algunos casos, es imposible tener todas las soluciones pre-programadas. Flynn (1988) propuso un sistema de reglas para la fusión de datos, que proporciona un conjunto simple de reglas heurísticas usadas con frecuencia en los robots móviles para fusionar datos de dos tipos de sensores. A pesar de ser un conjunto de reglas simple, es bastante efectivo.

Redes de neuronas. Las redes de neuronas también han sido utilizadas para problemas de estimación. Un ejemplo es el algoritmo Neurally Inspired Contact Estimator (NICE), un algoritmo de análisis del movimiento desarrollado por DeAngelis et al. (1998). El algoritmo NICE es equivalente al estimador de máxima verosimilitud (ML) pero es un orden de magnitud más rápido.

Uno de los problemas relacionados con las redes de neuronas es determinar el número de capas ocultas y sus neuronas. Ash (1989) propuso un sistema de creación de nodos dinámico que empieza con una pequeña red y va incrementando los nodos hasta que la red cumple su tarea. Este método fue aplicado a la fusión de datos por Ghosh & Holmberg (1990).

El uso de las redes de neuronas como una técnica genérica de fusión de datos de múltiples parámetros fue propuesto por Taylor & MacIntyre (1998), utilizando redes de Kohonen.

Recientemente, los algoritmos genéticos han sido utilizados para diseñar redes de neuronas de cara a fusionar datos. Abdel-Aty-Zohdy & Ewing (2000) proponen esta técnica para el diseño electrónico de un sistema de fusión de datos de una nariz electrónica.

Lógica difusa. El filtro de Kalman asume un conocimiento a-priori del proceso y de la medida del ruido. Esta información generalmente no está disponible en la mayoría de los sistemas, por lo que se suele estimar, con una estimación, que puede dar lugar a una degradación en el rendimiento del filtro y puede proporcionar una divergencia (Smith & Singh, 2006). Un filtro difuso adaptativo puede proporcionar un mejor rendimiento que un filtro de kalman si los problemas de estimación son solucionados. Escamilla-Ambrosio & Mort (2001, 2002) propusieron el Fuzzy Logic Adaptive Kalman Filter (FL-AKF). Su propuesta modifica los valores del filtro de kalman utilizando lógica difusa para ajustarlos lo mejor posible a los valores estimados de covarianza. Este método parece que funciona bien detectando los fallos de los sensores, rechazando valores raros y cuando los errores cambian con el tiempo. El trabajo de Escamilla-Ambrosio ha sido ampliado por Sasiadek & Hartana (2000) con tres nuevas técnicas: Fuzzy-Logic-based Adaptive C KF (FL-ACKF), Fuzzy-Logic-based Adaptive D KF (FL-ADKF) y Fuzzy-Logic-based Adaptive F KF (FL-AFKF). Sus resultados muestran

que estas técnicas son efectivas en situaciones donde hay sensores heterogéneos midiendo los mismos parámetros pero con diferentes estadísticos de ruido y dinámica. Borotschnig et al. (1997) propusieron utilizar grafos relacionales basados en lógica difusa para afrontar el problema de agrupamiento de atributos y estimación entre fotogramas sucesivos.

2.7.3. Fusión de decisiones

Una decisión es tomada basándose en el conocimiento de la situación percibida, la cual es proporcionada por distintas fuentes. El objetivo de estos métodos es realizar una inferencia de alto nivel sobre las actividades y los eventos que se producen entre los objetivos detectados. Suelen utilizar información simbólica y su fusión conlleva el razonamiento y la inferencia teniendo en cuenta la incertidumbre y las restricciones. Estos métodos se encuadran dentro del nivel 2 (valoración de la situación) y nivel 4 (valoración del impacto) de la jerarquía de niveles del JDL.

Atendiendo a su origen, los métodos más utilizados para realizar fusión de decisiones sobre los objetivos o eventos percibidos pueden dividirse en tres grupos:

1. **Métodos probabilísticos:** Inferencia Bayesiana, redes Bayesianas e inferencia Dempster-Shafer.
2. **Métodos basados en técnicas de inteligencia artificial:** Razonamiento abductivo y fusión semántica.
3. **Métodos basados en la teoría de la información**: Métodos de votación.

Métodos probabilísticos

Inferencia Bayesiana. La fusión de información basada en la inferencia Bayesiana proporciona un formalismo para combinar evidencia de acuerdo a las reglas de la teoría de probabilidades. La incertidumbre es representada en términos de probabilidades condicionadas que describen las creencias y que pueden tener valores en el intervalo [0,1] donde 0 indica falta total de creencia y 1 creencia absoluta. La inferencia Bayesiana, está basada en la popular regla de Bayes:

$$Pr(Y|X) = \frac{Pr(X|Y)Pr(Y)}{Pr(X)} \tag{2.35}$$

donde la probabilidad a posteriori, $Pr(Y|X)$, representa la creencia de la hipótesis Y dada la información X. Esta probabilidad es obtenida multiplicando la probabilidad a-priori de la hipótesis, $Pr(Y)$, por la probabilidad de recibir X dado que Y es verdad, $Pr(X|Y)$. El valor de $Pr(X)$ se utiliza como una constante normalizadora. El mayor problema relacionado con la inferencia Bayesiana es que las probabilidades $Pr(x)$ y $Pr(X|Y)$ deben conocerse. Algunos autores proponen el uso de redes neuronales para estimar las probabilidades condicionales (Pan et al., 1998). Otros como Cou et al. (2002) utilizan programación Bayesiana. En el libro de Hall & Llinas (2001), se describen una serie de problemas relacionados con la inferencia Bayesiana:

- Dificultad en definir el valor de las probabilidades a priori.
- Complejidad cuando hay muchas hipótesis potenciales y muchos eventos que dependen de condiciones.

- Las hipótesis deben ser mutuamente exclusivas.
- La imposibilidad de describir la incertidumbre en las decisiones.

Redes Bayesianas. Una red Bayesiana es una herramienta gráfica de modelado que tiene forma de grafo directo acíclico y que se utiliza para modelar la distribución conjunta de una serie de variables aleatorias. El grafo consta de una serie de nodos, donde cada nodo representa una variable aleatoria y de una serie de arcos que representan influencia causal entre los nodos conectados (Pearl, 1988). Kumar & Desai (1996), utilizaron las redes Bayesianas para solucionar el problema de interpretación en imágenes.

Inferencia Dempster-Shafer. La inferencia Dempster Shafer está basada en la teoría matemática introducida por Dempster (1968) y Shafer (1976) que generaliza la teoría Bayesiana. La teoría Dempster-Shafer proporciona un formalismo que puede ser utilizado para representar conocimiento incompleto, actualización de creencias y combinación de evidencias (Provan, 1992) y permite la representación explícita de la incertidumbre.

Un concepto fundamental en el sistema de razonamiento Dempster-Shafer, es el marco de discernimiento (frame of discernment), que se define de la siguiente forma: Sea $\Theta = \{\theta_1, \theta_2, ..., \theta_N\}$ el conjunto de todos los posibles estados que definen el sistema, Θ es exhaustivo y mutuamente exclusivo en el sentido de que el sistema está realmente en un y solo un estado, el estado $\theta_i \in \Theta$, donde $1 \leqslant i \leqslant N$. Al conjunto Θ se le llama marco de discernimiento, debido a que sus elementos se utilizan para discernir el estado actual del sistema.

Los elementos del conjunto 2^{Θ} son llamados hipótesis. En la teoría Dempster-Shafer, basándose en la evidencia E, una probabilidad es asignada a cada hipótesis $H \in 2^{\theta}$, de acuerdo a la asignación básica de probabilidades o función de masas, $m : 2^{\Theta} \rightarrow [0, 1]$ que satisface:

$$m(\emptyset) = 0 \tag{2.36}$$

Es decir, la función de masas del conjunto vacío es cero.

$$m(H) \geq 0, \forall H \in 2^{\Theta} \tag{2.37}$$

Se define que la función de masas de una hipótesis es mayor o igual que cero para todas las hipótesis.

$$\sum_{H \in 2^{\Theta}} m(H) = 1 \tag{2.38}$$

La suma de la función de masas de todas las hipótesis es 1.

Para expresar las creencias completas en una hipótesis H, Demspter-Shafer define la función de creencia $bel : 2^{\Theta} \rightarrow [0, 1]$ sobre Θ como:

$$bel(H) = \sum_{A \subseteq H} m(A) \tag{2.39}$$

donde $bel(\emptyset) = 0$, y $bel(\Theta) = 1$. El nivel de duda en H puede expresarse intuitivamente en términos de la función de creencia $bel : 2^{\Theta} \rightarrow [0, 1]$ como:

$$dou(H) = bel(\neg H) = \sum_{A \subseteq \neg H} m(A) \tag{2.40}$$

Para expresar la plausibilidad de cada hipótesis , la función $pl : 2^{\Theta} \rightarrow [0, 1]$ sobre Θ se define como:

$$pl(H) = 1 - dou(H) = \sum_{A \cap H = \emptyset} m(A) \tag{2.41}$$

La plausibilidad intuitivamente indica que existe menos duda en la hipótesis H cuanto mayor plausible sea. En este contexto, el intervalo de confianza $[bel(H), pl(H)]$ define la verdadera creencia de la hipótesis H.
Para combinar los efectos de dos funciones de masas m_1 y m_2, la teoría Demspter-Shafer define una regla de combinación, $m_1 \oplus m_2$, que viene dada por:

$$m_1 \oplus m_2(\emptyset) = 0, \tag{2.42}$$

$$m_1 \oplus m_2(H) = \frac{\sum_{X \cap Y = H} m_1(X) m_2(Y)}{1 - \sum_{X \cap Y = \emptyset} m_1(X) m_2(Y)} \tag{2.43}$$

De acuerdo con Luo & Kay (1992), el uso de la teoría de Dempster-Shafer para fusión de datos de sensores se empezó a utilizar por Garvey et al. (1995). La teoría de Dempster-Shafer, es más flexible que la inferencia Bayesiana, en el sentido que permite que cada fuente contribuya con información de diferentes niveles de detalle. A diferencia de la inferencia Bayesiana, Dempster-Shafer permite fusionar datos proporcionados por diferentes tipos de sensores. Además en la inferencia Dempster-Shafer, no es necesario proporcionar probabilidades apriori a proposiciones desconocidas, sino que las probabilidades son asignadas en el momento en que la información está disponible. Existen varios artículos que realizan comparaciones entre el uso de la inferencia Bayesiana y la inferencia Dempster-Shafer como los de Buede et al. (1988), Cheng & Kashyap (1988) y Cobb & Shenoy (2003). Wu et al. (2002) publicaron un artículo en el que utilizan la teoría de Dempster-Shafer para realizar fusión de información en entornos de computación dependiente del contexto. Los mismos autores (Wu et al., 2003) extienden el trabajo anterior para dinámicamente ir modificando los pesos asociados a las medidas de los sensores; de esta forma, el mecanismo de fusión se calibra de acuerdo a las medidas recientes de los sensores cuando se dispone de información de ground-truth. En el entorno militar, Bossé et al. (2006) utilizan el razonamiento Dempster-Shafer junto con información a priori almacenada en una base de datos para clasificar barcos militares. La clasificación la realizan utilizando redes de neuronas. En un trabajo reciente, Basir & Yuan (2007) proponen el uso del razonamiento Dempster-Shafer para el diagnostico de los motores, el cual puede verse como un problema multi-sensor.

Métodos basados en técnicas de inteligencia artificial

Razonamiento abductivo. Abducción, o inferir la mejor explicación, es un método de razonamiento en el cual se elige aquella hipótesis, que de ser cierta, explica mejor la evidencia observada (Peirce, 1955). Es decir, cuando un hecho es observado, la abducción busca la mejor explicación para ese hecho.

En el contexto del razonamiento probabilístico, la inferencia abductiva consiste en encontrar la máxima probabilidad a posteriori de las variables del sistema, dadas algunas variables observadas. De hecho, la abducción es realmente un patrón de razonamiento en lugar de un método de fusión. Por tanto, se pueden usar diferentes métodos de inferencia, como redes neuronales (Abdelbar et al., 2003) o lógica difusa (Romero Agüero & Vargas, 2005) para realizar abducción.

Fusión semántica de información. La fusión semántica de información es esencialmente un esquema en el cual los datos en bruto de los sensores son procesados de forma que los nodos intercambian únicamente la información semántica resultante. La fusión semántica de la información, normalmente comprende dos fases: (1) construcción del conocimiento y (2) concordancia de patrones (inferencia). La primera fase (normalmente offline) incorpora el conocimiento más apropiado en información semántica, que después es usada en la segunda fase (normalmente online) para fusionar atributos relevantes y proporcionar una interpretación semántica de los datos de los sensores (Friedlander & Phoha, 2002) (Friedlander, 2005) (Whitehouse et al., 2006).

La idea de la fusión semántica es integrar y convertir los datos de los sensores en lenguajes formales. Por tanto, el lenguaje resultante obtenido de las observaciones del entorno es comparado con los lenguajes con comportamiento similar almacenados en la base de datos. La base de esta estrategia es que los comportamientos similares, representados por lenguajes formales, son semánticamente similares. Este método proporciona un ahorro en la transmisión debido a que los sensores transmiten solo los lenguajes formales de los datos percibidos en lugar de los datos en bruto. No obstante, es necesario tener un conjunto de comportamientos conocidos y almacenados previamente en una base de datos, que en algunos casos es difícil de obtener.

Métodos basados en la teoría de la información. Los métodos basados en la teoría de la información más utilizados, son los métodos de votación, que combinan las clasificaciones de cada uno de los sensores involucrados en la fusión, tratando cada una de las decisiones como un voto en donde las decisiones apoyadas por la mayoría son utilizadas. Se trata de los primeros métodos históricamente utilizados para realizar fusión de datos (modelos de Condorcet). Los métodos de fusión basados en modelos de votación simplifican bastante el proceso de fusión; sin embargo, suelen ser apropiados únicamente para problemas con decisiones booleanas, debido a que su razonamiento no es suficientemente potente para razonar acerca de múltiples estados. Los métodos de votación han sido usados en el seguimiento de objetivos (Klein et al., 1993) y en la detección de minas (Kacalenga et al., 2003).

2.8. Fusión de datos distribuida: métodos, técnicas y algoritmos

Un sistema distribuido consta de una red de nodos de proceso conectados entre sí, los cuales generalmente comparten un objetivo común. La principal diferencia frente a un sistema centralizado, es que, en los sistemas centralizados existe un único nodo que procesa la información. En los sistemas centralizados, este único nodo maneja su propio reloj interno y no es necesario establecer comunicación con otros nodos. Sin embargo, en los sistemas distribuidos, los datos de los sensores se adquieren por medio de múltiples nodos y la comunicación entre estos nodos es el mayor cuello de botella. Además, en cada nodo de procesado de un sistema distribuido, se asume que existe un sistema operativo que soporta el concepto de proceso y que proporciona primitivas para la comunicación de red. En un sistema distribuido, los diferentes pasos de procesado del sistema están divididos entre los diferentes nodos. Los sistemas de vigilancia multi-cámara o las redes de sensores visuales, donde cada una de las imágenes capturadas son procesadas por un nodo, son un claro ejemplo de los sistemas distribuidos.

En las arquitecturas distribuidas, el problema clave es como combinar los resultados de varios nodos (Liggins et al., 1997). Las técnicas clásicas de fusión de datos centralizadas, como las presentadas en el apartado anterior, deben ser utilizadas con cuidado en los sistemas distribuidos, debido a que asumen que los errores en los datos que van a ser fusionados son independientes.

En una arquitectura completamente distribuida, no existe una relación de superior subordinado, sino que cada nodo se comunica con los otros nodos. La información se puede adaptar dependiendo de su contenido y de las necesidades de los nodos. En el caso extremo de los sistemas descentralizados, cada sensor tiene su propio procesador para fusionar los datos locales y cooperar con otros sensores.

La arquitectura de fusión centralizada, es teóricamente óptima, si el ancho de banda es suficientemente grande para transmitir toda la información al nodo central y este tiene suficientes recursos computacionales para procesar toda la información recibida (Liggins et al., 1997). En los años 80, Tenney & Sandell (1981) publicaron un trabajo en el que afirmrona, que debido a la sobrecarga computacional (de un sistema distribuido), un sistema de fusión centralizado, teóricamente se comporta mejor que un sistema de fusión distribuido; puesto que un sistema centralizado tiene un conocimiento global en el sentido de que todos los datos medidos están disponibles, mientras que en un sistema de fusión distribuido el conocimiento debe ser propagado por los nodos.
Sin embargo, el enfoque centralizado tiene (Rao et al., 1993) dos grandes problemas:

1. Implica un solo lugar donde existe el conocimiento de la información fusionada. En caso de fallo de este nodo el sistema se ve completamente perjudicado.

2. La enorme sobrecarga de comunicación del sistema centralizado, lo hace prohibitivo en situaciones con muchos nodos. Además, en sistemas específicos, como las redes de sensores visuales, el coste que supone el envío por la red de las imágenes obtenidas hace computacionalmente costoso el sistema, debido al gran ancho de banda necesario.

Frente a un sistema centralizado, una arquitectura de fusión distribuida, tiene las siguientes ventajas (Liggins et al., 1997):

1. Una menor carga de procesamiento en el nodo de fusión, debido a que en el nodo de fusión la información se recibe una vez procesada.

2. No existe la necesidad de mantener una gran base de datos centralizada.

3. Existe una menor carga de comunicación por la red, ya que la información que se transmite ocupa menos ancho de banda que los datos originales.

4. Acceso más rápido a los datos fusionados debido a un menor retardo en la comunicación.

5. Menor probabilidad de fallos, al no existir un único punto crítico y al poder tener varios nodos que realizan la fusión de datos.

Además, una arquitectura de fusión distribuida, puede ser una necesidad, porque muchos sistemas de fusión necesitan ser construidos a partir de componentes existentes. En cuanto a los principales retos científicos y técnicos (Liggins et al., 1997) de un sistema de fusión distribuido, se pueden mencionar los siguientes:

- **Arquitectura**: Como se comparte la responsabilidad de la fusión entre los nodos. Es decir, que fuentes o sensores deben comunicarse con que nodos y de que objetivos es responsable cada nodo. En este trabajo, se utiliza una arquitectura distribuida basada en un sistema multi-agente.

- **Comunicación**: Es necesario establecer como se deben comunicar los nodos, que conectividad y ancho de banda de la red es necesario, determinar si se comparte información de tipo push o pull[12] y si se comunican datos brutos o procesados. El mecanismo de comunicación utilizado en este trabajo está basado en el intercambio asíncrono de mensajes (de tipo push).

- **Algoritmos**: Como los nodos deben procesar los datos para obtener resultados de alto rendimiento y seleccionar las acciones de comunicación (a quien comunicar, cuando comunicar, que comunicar y como comunicar). En los sistemas distribuidos, cada proceso tiene su propio espacio de direcciones, dentro del espacio de memoria de la maquina donde se ejecuta y los datos no se pueden transferir de forma sencilla de un proceso a otro.

A continuación, se presentan los principales problemas de las técnicas de fusión de datos distribuida: las medidas que llegan fuera de secuencia (2.8.1) y los datos correlados (2.8.2). Posteriormente, se presentan los métodos de asociación de datos distribuida (2.8.3), los métodos de estimación de estado distribuida (2.8.4) y la fusión distribuida de decisiones (2.8.5) más importantes.

[12] Es decir si los nodos envían la información (push) o si la solicitan (pull)

2.8.1. Medidas fuera de secuencia

Una suposición, que se realiza frecuentemente en los sistemas de fusión de datos distribuidos consiste en que las medidas llegan al nodo de fusión sin retardos, sin embargo, esta suposición normalmente no se corresponde a una situación realista puesto que en la práctica, existen retardos entre el instante en el cual una medida está disponible para el sensor y el instante en el cual es utilizada para ser fusionada. Por tanto, la suposición de no retardo no se puede asumir en sistemas reales. Las medidas fuera de secuencia, consisten en datos que llegan desordenados a los nodos que los necesitan, porque en una red de sensores visuales existen distintos tiempos de propagación de la información entre los nodos. Cuando los retardos de las medidas son conocidos y fijos, el filtro de kalman se puede extender fácilmente para considerar el tiempo de latencia. Sin embargo, en el caso de que los retardos de las medidas sean variables el problema se acentúa y las soluciones no son tan sencillas.

Cada medida, que es necesario fusionar, debería llegar al proceso donde se ejecuta el algoritmo de fusión con una marca de tiempo (timestamp), generada por el reloj del sistema del nodo donde se obtiene la medida. Para poder realizar una comprobación temporal de la información transmitida por todos los nodos sensores en el nodo de fusión, los relojes de los distintos nodos deben estar alineados. Por tanto, los relojes de todos los nodos involucrados en el sistema deberían estar sincronizados para poder realizar las marcas de tiempo de forma correcta.

En un sistema de fusión de datos, pueden existir distintos tiempos de propagación para cada una de las fuentes, lo cual se acentúa en los sistemas de fusión distribuidos. Existe una alta probabilidad, en la mayoría de los sistemas multi-sensor, de que los datos lleguen desordenados o fuera de secuencia. Por lo general, los sistemas de fusión de datos, solo almacenan un histórico de las estadísticas más importantes como el estado estimado y la matriz de covarianzas. Por tanto, las medidas fuera de secuencia introducen el problema de encontrar una forma de actualizar la estimación actual utilizando datos fuera de secuencia.

Las medidas fuera de secuencia, se han reconocido como un problema para los sistemas de fusión desde los años 80 (Thomopoulos & Zhang, 1988). Los primeros intentos para solucionar el problema de las medidas fuera de secuencia, consistieron en descartar la información recibida entre el instante en que los datos se generaron y el instante en el que se está fusionando la información (Larsen et al., 1998) (Alexander, 2005). El uso de esta técnica solo permite solucionar el problema de los datos que llegan retrasados, no el hecho de que los datos lleguen desordenados.

Una solución intuitiva y perfecta en términos de precisión, es almacenar todos los datos siguiendo un orden temporal y cuando llegue un dato fuera de secuencia, recalcular todos los datos. Aunque sería una solución óptima, es prohibitiva en términos computacionales y requisitos de memoria y no se puede aplicar en sistemas en tiempo real.

Blackman & Popoli (1999) y Hilton et al. (1993) propusieron una solución aproximada al problema, aplicado a sistemas de seguimiento, llamada algoritmo B. Bar-Shalom (2002), amplió la propuesta proporcionando un algoritmo con salida óptima (algoritmo A) y demostrando que el algoritmo B está cerca del óptimo. Todas estas soluciones, sin embargo, asumen que el retardo en las medidas fuera de secuencia, es menor que un instante de tiempo. A partir de la publicación de los artículos anteriores, ha habido varios intentos de extender los algoritmos de fusión de datos para trabajar con retardos arbitrarios, como los de Mallick et al. (2001), Nettleton & Durrant-Whyte (2001) y Zhang et al. (2002). La solución proporcionada por Nettleton & Durrant-Whyte (2001) la hace inviable para sistemas en tiempo real, puesto que

el sistema se tiene que ejecutar con una latencia igual a la máxima latencia de los sensores. El algoritmo MHT, también se ha extendido para funcionar frente a medidas fuera de secuencia por Mallick et al. (2002).

En el caso particular de la fusión pista-a-pista, Challa & Legg (2002) desarrollaron un algoritmo basado en un filtro de Kalman. Este filtro de Kalman procesa no solo el estado actual, sino también simultáneamente, los estados anteriores. Demostraron que existe una mejora significativa frente a la opción de ignorar las medidas fuera de secuencia. En otro trabajo, Challa et al. (2003) realizaron un análisis Bayesiano detallado para la fusión de datos con medidas fuera de secuencia. Demostraron que es necesario realizar una estimación, que tenga en cuenta la densidad conjunta del estado actual del objetivo estimado y del estado del objetivo, considerando las medidas fuera de secuencia.

Sin embargo, todos los métodos anteriores asumen, que aunque las medidas lleguen fuera de secuencia, la cantidad de retardo (con un ruido aleatorio), es conocida. Existen varias situaciones en las cuales no es posible conocer con exactitud el retardo que poseen las observaciones, por ejemplo, en los casos en los que es posible asignar una marca de tiempo a la medida, en el momento de fusionarla pero no el momento de obtenerla.

En este trabajo, el problema de las medidas fuera de secuencia aparece debido a que los agentes se comunican por medio de mensajes utilizando una red de transmisión con el protocolo TCP/IP. El tiempo de trasmisión de los mensajes de cada agente sensor y el que existe entre los mensajes de los diferentes agentes sensores es variable. Para solucionarlo, se ha optado por una solución simple y práctica que consiste en descartar aquellos mensajes cuya diferencia temporal sea mayor que un determinado umbral (parámetro denominado temporal difference). Por lo general, en los experimentos realizados con el sistema, el valor de este parámetro se establece mayor o igual que el valor del parámetro de la frecuencia de fusión en el agente de fusión.

Finalmente, Julier & Uhlmann (2005) han propuesto el uso de las técnicas de unión de covarianzas para realizar la fusión distribuida de medidas fuera de secuencia. Su trabajo se basa en una combinación del filtro de kalman y de la unión de covarianzas y se explicará más en detalle en el apartado 2.8.4.

2.8.2. Datos correlados o incesto de datos

Las técnicas de fusión de datos distribuida, deben tener en cuenta el problema de los datos correlados, también conocido como *incesto de datos* o *propagación de rumores*, el cual puede producir resultados fusionados erróneos y muy sesgados. El incesto de datos, es una situación en la cual las medidas en bruto son usadas, sin saberlo, múltiples veces como si fueran independientes. El incesto de datos es una posible causa de la correlación entre las estimaciones en un entorno de fusión de datos distribuido y las inconsistencias con la incertidumbre real.

El incesto de datos es el resultado del uso repetitivo de la misma información sin ser conscientes de ello. En un escenario de fusión de datos en el cual las estimaciones basadas en las medidas de todos los nodos son intercambiadas, el problema del incesto de datos crece exponencialmente. El problema del incesto de datos se ilustra de forma sencilla en la figura 2.7, donde existen tres nodos (A, B y C) que obtienen información del entorno y producen sus propias estimaciones. En el caso de que el nodo C no tenga la posibilidad de comunicar sus estimaciones directamente al nodo de fusión, las envía a los nodos A y

B, que a continuación las combinan internamente y las envían al nodo de fusión de forma independiente, produciendo por tanto el incesto de datos, ya que la estimación enviada por C es utilizada varias veces.

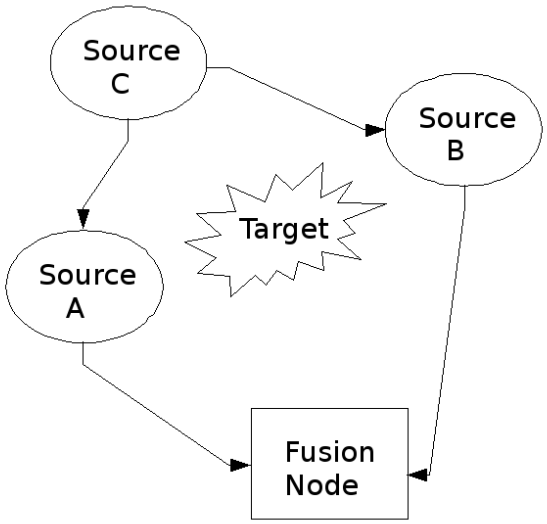

Figura 2.7: Ejemplo de una situación de incesto de datos. Si las tres fuentes obtienen datos del entorno, pero la fuente C no se puede comunicar con el nodo de fusión y envía la información simultáneamente a la fuente A y B se produce el problema de incesto de datos en el nodo de fusión.

En un esquema de fusión descentralizado donde las estimaciones producidas por cada uno de los nodos de fusión son enviadas a todos los nodos, el problema del incesto de datos crece de forma exponencial.

Uno de los mayores problemas a la hora de llevar a cabo la fusión de datos utilizando los filtros de kalman distribuidos, es la necesidad de requerir que las medidas sean independientes o las covarianzas cruzadas conocidas. Una simplificación que se suele utilizar, es asumir que la covarianza cruzada es cero, entonces en este caso el filtro de Kalman produce covarianzas no conservativas. Esto genera artificialmente un alto valor de confianza, lo que puede provocar la divergencia del filtro. La solución óptima del filtro de Kalman distribuido, es aquella que mantiene las covarianzas cruzadas entre las actualizaciones; sin embargo esta solución, escala de forma cuadrática con el número de actualizaciones, haciéndola inviable para su utilización en la práctica.

McLaughlin et al. (2003) han desarrollado una estrategia para eliminar el incesto de datos en las arquitecturas distribuidas. El algoritmo obtiene las estimaciones de estado de los otros nodos y resuelve las medidas remotas de ellos. Almacena estas medidas remotas y las usa para actualizar la estimación de su propio estado; de esta forma el incesto es eliminado antes de que los datos sean usados.

En el esquema utilizado en este trabajo (arquitectura distribuida jerárquica con feedback) el incesto de datos no afecta de forma significativa a las estimaciones producidas por cada uno de los nodos, ya que la estimaciones que pueden ser re-utilizadas solo se dan en el caso de la fusión activa cuando se utiliza en las fuentes el resultado del nodo de fusión.

2.8.3. Asociación de datos distribuida

El problema de la asociación de datos presentado en 2.7.1, se vuelve más difícil de resolver cuando se trata de una red de sensores distribuida, en la cual diversos sensores pueden observar a los mismos objetivos. Cuando, una asociación de datos, distribuida en M sensores se realiza en un nodo central utilizando solo las medidas recibidas en un instante k, esta tiene la misma complejidad que la asociación de datos con M medidas en un solo nodo. En una asociación de datos distribuida, si un sensor tiene n posibles combinaciones y hay M sensores, existen n^M posibles combinaciones que se pueden llevar a cabo en el nodo que se encarga de realizar la asociación. Esta situación hace el problema intratable para realizar una búsqueda exhaustiva con grandes valores de M.

Asociación de datos probabilística conjunta distribuida (JPDA-D)

La versión distribuida del algoritmo de asociación de datos probabilística conjunta (JPDA-D) fue presentada por Chang et al. (1986). En el algoritmo JPDA-D, el estado estimado para un objetivo (utilizando dos sensores) después de ser asociado, viene dado por:

$$E\{x|Z^1, Z^2\} = \sum_{j=0}^{m_1} \sum_{l=0}^{m_2} E\{x|\chi_j^1, \chi_l^2, Z^1, Z^2\} \cdot P\{\chi_j^1, \chi_l^2|Z^1, Z^2\} \tag{2.44}$$

donde $m_i, i = 1, 2$ son los últimos conjuntos de medidas de los sensores uno y dos, $Z^i, i = 1, 2$ son los conjuntos de datos acumulados y χ_j^i es la asociación (hipótesis) de que Z_j^i es la medida correcta (es decir del objetivo correcto). El primer término de la parte derecha de la ecuación se calcula a partir de las asociaciones realizadas previamente. El segundo término se obtiene a partir de las probabilidades de asociación individuales de la siguiente forma:

$$P(\chi_j^1, \chi_l^2|Z^1, Z^2) = \sum_{x^1} \sum_{x^2} P(\chi^1, \chi^2|Z^1, Z^2)\hat{\omega}_j^1(\chi^1)\hat{\omega}_l^2(\chi^2) \tag{2.45}$$

$$P(\chi^1, \chi^2|Z^1, Z^2) = \frac{1}{c} P(\chi^1|Z^1)P(\chi^2|Z^2)\gamma(\chi^1, \chi^2) \tag{2.46}$$

donde χ^i son las hipótesis conjuntas, las cuales involucran todas las medidas y los objetivos y $\hat{\omega_j^i}(\chi^i)$ son los indicadores binarios de asociación medidas-objetivos. El término adicional $\gamma(\chi^1, \chi^2)$ depende de la correlación de las hipótesis individuales y refleja la influencia de la localización de las actuales medidas en las hipótesis combinadas conjuntas.

Estas ecuaciones se obtienen asumiendo que existe comunicación después de cualquier observación y pasan a ser solo aproximaciones cuando existe comunicación intermitente y cuando existe ruido. Por tanto, este algoritmo consiste en un modelo teórico que tiene sus limitaciones cuando se aplica en la práctica.

Test de múltiples hipótesis distribuido (MHT-D)

La versión distribuida del algoritmo MHT (Chong et al., 1985) (Chong et al., 1990) sigue una estructura similar a la del algoritmo JPDA-D. Consideremos el caso en el que un nodo de fusión necesita fusionar dos conjuntos de hipótesis y pistas. Si los conjuntos de hipótesis y pistas están representados por $H^i(Z^i)$ y $T^i(Z^i)$ con $i = 1, 2$, las probabilidades de las hipótesis por λ_j^i y las distribuciones de estado de las pistas τ_j^i por $P(\lambda_j^i|Z^i)$ y $p(x|Z^i, \tau_j^i)$, la

máxima información disponible en el nodo de fusión es $Z = Z^1 \cup Z^2$. El objetivo de la fusión de datos con MHT-D, es obtener el conjunto de hipótesis $H(Z)$, el conjunto de pistas $T(Z)$, las probabilidades de las hipótesis $P(\lambda|Z)$ y el seguimiento de las distribuciones de estado $p(x|Z, \tau)$.

El algoritmo de fusión distribuida MHT-D consta de los siguientes pasos:

1. Formación de Hipótesis: Para cada par de hipótesis λ_j^1 y λ_k^2 que pueden ser fusionadas, una pista τ es formada, mediante la asociación de los pares de pistas τ_j^1 y τ_k^2 cada una de un nodo, y que han podido ser originadas por el mismo objetivo. El resultado final de esta fase es un conjunto de hipótesis $H(Z)$ y de pistas fusionadas $T(Z)$.

2. Evaluación de Hipótesis: En esta fase, se calcula la probabilidad de asociación de cada hipótesis y el estado estimado para cada pista fusionada. El algoritmo distribuido de estimación se utiliza para calcular las verosimilitudes de las posibles asociaciones y las estimaciones resultantes de una determinada asociación. Usando el modelo de información, la probabilidad de cada hipótesis fusionada viene dada por:

$$P(\lambda|Z) = C^{-1} \prod_{j \in J} P(\lambda^{(j)}|Z^{(j)})^{\alpha(j)} \prod_{\tau \in \lambda} L(\tau|Z) \tag{2.47}$$

   donde C es una constante normalizadora.

El problema del método MHT-D es su gran coste computacional, que es del orden de $O(n^M)$, siendo n el número de posibles configuraciones de asociación y M el número de variables a ser estimadas.

Modelos gráficos

Una solución al problema de asociación de datos distribuido, en redes de sensores con áreas solapadas, donde los sensores están sincronizados y cada sensor recibe medidas ruidosas fue propuesto por Chen et al. (2005, 2006) utilizado un marco de trabajo de modelos gráficos (Jordan, 1998).

Su propuesta, se basa en utilizar un enfoque gráfico para diseñar la asociación de datos, en el cual se puede representar la estructura de las dependencias estadísticas de las variables aleatorias. De esta forma, es posible obtener un problema de inferencia que se puede solucionar de forma eficiente con el algoritmo max-product (Weiss & Freeman, 2001). Los modelos gráficos representan las dependencias estadísticas de las variables en forma de grafo, el algoritmo max-product de propagación de creencias converge en un punto cuando el grafo es un árbol.

Además, el algoritmo resultante puede implementarse de una forma distribuida por medio del intercambio de mensajes entre los nodos de forma paralela, lo que proporciona una solución al problema de asociación de datos distribuido. Con este método, si cada sensor tiene n posibles configuraciones de asociación y existen M variables a ser estimadas, la complejidad es $O(n^2 M)$, que es menor que la complejidad $O(n^M)$ del método MHT distribuido. Sin embargo, el inconveniente es que es necesario construir el modelo gráfico prestando atención a las variables correladas.

2.8.4. Estimación del estado distribuida

Filtro de Kalman distribuido

Las ecuaciones del filtro de kalman en una arquitectura distribuida y jerárquica pueden consultarse en el artículo de Liggins et al. (1997). La necesidad de una sincronización apropiada de relojes entre las fuentes es una característica propia del uso de los filtros de Kalman en los sistemas distribuidos y está demostrada por (Ganeriwal et al., 2003). Por tanto, para aplicar de una forma coherente un filtro de kalman distribuido, los relojes de los sistemas implicados deben estar sincronizados. La forma de obtener este sincronismo viene dada por protocolos que se apoyan en un reloj global, como el protocolo Network Time Protocol (NTP). Por otro lado, Manzo et al. (2005), han demostrado en un trabajo, como los problemas de sincronización causados por un ataque en la sincronización de los relojes, afecta al rendimiento de las estimaciones del filtro de Kalman, proporcionando estimaciones incorrectas. Si las estimaciones son consistentes entre sí y la covarianza cruzada de las mismas es conocida (o bien las estimaciones no están correladas), entonces se pueden utilizar los filtros de kalman distribuidos (Uhlmann, 2003), teniendo en cuenta que la covarianza cruzada debe ser determinada de forma exacta y las observaciones deben de ser consistentes entre sí.

Filtro de partículas distribuido

Los filtros de partículas distribuidos han sido objeto de estudio recientemente (Bashi et al., 2003) (Coates, 2004) (Gu, 2007) . Coates (2004), presentó un filtro de partículas distribuido, aplicado a un problema en el que el entorno monitorizado puede ser capturado por un modelo espacio-estado markoviano, que involucra dinámica y observaciones no lineales y ruido no Gaussiano. Por otro lado, los primeros intentos en solucionar las medidas fuera de secuencia, usando filtros de partículas, se han basado en regenerar la función de densidad de probabilidad al instante de la medida fuera de secuencia (Bar-Shalom, 2002). En un filtro de partículas, esto requiere una carga computacional muy elevada, además del espacio necesario para almacenar el histórico de las partículas. Para evitar este problema, Orton & Marrs (2001) propusieron almacenar la distribución de las partículas en cada instante y así evitar tener que recalcularlas. Utilizando esta técnica, a medida que el retardo aumenta el resultado se degrada levemente estando bastante cerca del óptimo (Hernandez et al., 2002). El problema de esta técnica, es que aunque reduce el coste computacional, sigue necesitando una cantidad muy grande de almacenamiento para mantener el estado de las partículas en cada instante.

Métodos de consistencia de covarianzas (CI y CU)

Los métodos de consistencia de covarianzas (intersección y unión de covarianzas) consisten en un marco general, riguroso y tolerante a fallos para mantener la media y la estimación de las covarianzas, en una red distribuida y han sido propuestos por Uhlmann (2003). No consisten en una técnica de estimación, sino más bien, en un método de fusión de estimaciones. Como ya se ha comentado, uno de los principales inconvenientes del filtro de kalman distribuido, es que necesita que las medidas sean independientes o bien que las covarianzas cruzadas sean conocidas. Este inconveniente no se produce en los métodos de consistencia de covarianzas.

A continuación se describe el método de intersección de covarianzas (CI) y posteriormente el método de unión de covarianzas (CU).

Método de intersección de covarianzas (CI). Se asume, que cuando se utiliza el filtro de Kalman para combinar dos estimaciones $(\mathbf{a}_1, \mathbf{A}_1)$ y $(\mathbf{a}_2, \mathbf{A}_2)$ la covarianza conjunta es de la forma:

$$\begin{bmatrix} \mathbf{A}_1 & \mathbf{X} \\ \mathbf{X}^\top & \mathbf{A}_2 \end{bmatrix} \tag{2.48}$$

donde la covarianza cruzada $\mathbf{X}$ debe ser conocida de forma exacta para que las ecuaciones del filtro de Kalman se puedan aplicar sin problemas. Debido a que estimar las covarianzas cruzadas es computacionalmente intensivo, Julier & Uhlmann (1997b) propusieron el algoritmo de intersección de covarianzas (CI). Si es posible definir una covarianza conjunta $\mathbf{M}$ con bloques diagonales $M_{A_1} > A_1$ y $M_{A_2} > A_2$ tal que:

$$\mathbf{M} \geq \begin{bmatrix} \mathbf{A}_1 & \mathbf{X} \\ \mathbf{X}^\top & \mathbf{A}_2 \end{bmatrix} \tag{2.49}$$

para cualquier instancia posible de la covarianza cruzada desconocida $\mathbf{X}$, entonces los componentes de la matriz $\mathbf{M}$ se pueden utilizar en las ecuaciones del filtro de Kalman para proporcionar una estimación fusionada $(\mathbf{c}, \mathbf{C})$ que se considera consistente.

La clave de este método, consiste en generar una matriz de covarianzas conjunta $\mathbf{M}$, que sea capaz de construir una estimación fusionada útil[13]. Es decir, el método CI obtiene la matriz de covarianza conjunta $\mathbf{M}$, para la cual las ecuaciones del filtro de Kalman proporcionan la mejor estimación fusionada $(\mathbf{c}, \mathbf{C})$, con respecto a una medida fija del tamaño de covarianza (por ejemplo, mínimo determinante). Es necesario establecer una medida de covarianza concreta, ya que no existe una única covarianza conjunta que sea menor, en el orden de las matrices definidas positivas. Aunque la estructura de covarianza conjunta sea la base del análisis formal del algoritmo CI, el resultado real del algoritmo de fusión de CI es una mezcla no lineal de la información contenida en las estimaciones que se están fusionando, de la siguiente forma:

$$\begin{aligned} \mathbf{C} &= (\mathbf{w}_1\mathbf{H}_1^\top\mathbf{A}_1^{-1}\mathbf{H}_1 + \mathbf{w}_2\mathbf{H}_2^\top\mathbf{A}_2^{-1}\mathbf{H}_2 + \cdots + \mathbf{w}_n\mathbf{H}_n^\top\mathbf{A}_n^{-1}\mathbf{H}_n)^{-1} \\ \mathbf{c} &= \mathbf{C}\,(\mathbf{w}_1\mathbf{H}_1^\top\mathbf{A}_1^{-1}\mathbf{a}_1 + \mathbf{w}_2\mathbf{H}_2^\top\mathbf{A}_2^{-1}\mathbf{a}_2 + \cdots + \mathbf{w}_n\mathbf{H}_n^\top\mathbf{A}_n^{-1}\mathbf{a}_n) \end{aligned} \tag{2.50}$$

donde $\mathbf{H}_i$ es la transformación del espacio de estado de la estimación fusionada al espacio de estado de la estimación i. Los valores de w, se pueden calcular para minimizar el determinante de la covarianza utilizando paquetes de software de optimización convexa y programación de matrices semi-definidas. El resultado del algoritmo CI tiene características distintas que el proporcionado por el filtro de kalman. Por ejemplo, si se proporcionan dos estimaciones: $(\mathbf{a}, \mathbf{A})$ y $(\mathbf{b}, \mathbf{B})$ y las covarianzas son iguales $\mathbf{A} = \mathbf{B}$, el filtro de Kalman se basa en la presunción de independencia estadística y produce una estimación fusionada con covarianza $\mathbf{C} = \frac{1}{2}\mathbf{A}$. Debido a que el método CI no asume independencia y por tanto debe ser consistente incluso en el caso en el que las estimaciones estén completamente correladas, la covarianza estimada de fusión es $\mathbf{C} = \mathbf{A}$. Este hecho, es precisamente lo que asegura consistencia en un contexto de fusión de datos distribuida donde un nodo puede recibir copias redundantes de la misma estimación. En el caso de estimaciones donde $\mathbf{A} < \mathbf{B}$, CI no proporciona información acerca de la estimación $(\mathbf{b}, \mathbf{B})$, por tanto el resultado fusionado es $(\mathbf{a}, \mathbf{A})$.

Cualquier covarianza conjunta consistente, es suficiente para producir una estimación fusionada que garantice consistencia, pero además es necesario garantizar la no divergencia.

[13] Por útil, en este contexto, se entiende que tenga una incertidumbre asociada menor.

La no divergencia se consigue en el método CI, eligiendo una medida particular (por ejemplo el determinante) que es minimizada en cada operación de fusión. Esta medida representa un criterio de no divergencia, debido a que el tamaño de la covarianza estimada de acuerdo a este criterio no debe incrementarse.

La aplicación del método de intersección de covarianzas, garantiza consistencia y no divergencia para cualquier secuencia de estimaciones consistentes de media y covarianza. Sin embargo, no funciona bien cuando las medidas que se van a fusionar son inconsistentes entre sí, lo cual se pretende solucionar con el siguiente método de unión de covarianzas.

Método de unión de covarianzas (CU). Como se ha descrito anteriormente, la intersección de covarianzas soluciona el problema de las entradas correladas, pero no el de los valores inconsistentes[14], por eso, para solucionarlo Uhlmann (2003) propuso la unión de covarianzas (CU).

La unión de covarianzas (CU) se enfrenta a un problema importante: dos estimaciones $(\mathbf{a}_1, \mathbf{A}_1)$ y $(\mathbf{a}_2, \mathbf{A}_2)$ que relacionan el estado del mismo objeto de forma mutuamente inconsistente la una con la otra. Esto se produce si la diferencia entre las medias de la estimación del estado del objeto es mayor que la covarianza proporcionada.

Las entradas inconsistentes, se pueden detectar calculando la distancia de Mahalanobis (1936) entre las mismas, que se define por:

$$(\mathbf{a}_1 - \mathbf{a}_2)^\top (\mathbf{A}_1 + \mathbf{A}_2)^{-1} (\mathbf{a}_1 - \mathbf{a}_2) \tag{2.51}$$

y calculando si la distancia supera un determinado umbral. La distancia de Mahalanobis tiene en cuenta los valores de las covarianzas para medir la distancia. Si la diferencia entre las estimaciones es grande y las covarianzas de las mismas son muy grandes, la distancia de Mahalanobis proporciona un valor pequeño. Al contrario, si la diferencia entre las estimaciones es más pequeña y las covarianzas de las estimaciones son pequeñas se puede producir un valor de distancia grande. Un valor alto de la distancia de Mahalanobis puede indicar que las estimaciones no son consistentes entre ellas, sin embargo, es necesario un umbral definido por el usuario o aprendido por el sistema para establecer el punto de corte.

La unión de covarianzas afronta el siguiente problema: supongamos que un algoritmo de filtrado obtiene dos observaciones con medias y covarianzas $(\mathbf{a}_1, \mathbf{A}_1)$ y $(\mathbf{a}_2, \mathbf{A}_2)$ respectivamente. Se sabe que una de las observaciones es la correcta y la otra la errónea. Sin embargo, la identidad de la estimación correcta es desconocida y no puede ser determinada. En esta situación, si estas observaciones son utilizadas como entrada a un filtro de kalman, solo se puede garantizar que el filtro proporcionará una salida consistente, si se actualiza la observación con una medida que es consistente a las dos anteriores. En el caso de que las medidas sean las estimaciones del mismo objeto pero de diferentes sensores, igualmente solo se puede garantizar que la salida será consistente, si lo es la de las dos por separado. Como no es posible conocer que estimación es la buena, la única forma de combinar las dos estimaciones rigurosamente, es proporcionar una estimación $(\mathbf{u}, \mathbf{U})$ que sea consistente con las dos estimaciones y que obedece a las siguientes propiedades:

$$\begin{aligned} \mathbf{U} &\geqslant \mathbf{A}_1 + (\mathbf{u} - \mathbf{a}_1)(\mathbf{u} - \mathbf{A}_1)^\top \\ \mathbf{U} &\geqslant \mathbf{A}_2 + (\mathbf{u} - \mathbf{a}_2)(\mathbf{u} - \mathbf{A}_2)^\top \end{aligned} \tag{2.52}$$

[14] Por valores inconsistentes, nos referimos a distintas estimaciones, cada una de ellas con gran precisión (pequeña varianza) pero con una gran diferencia en el valor del estado de cada una de ellas.

donde alguna medida del tamaño de la matriz $\mathbf{U}$, como por ejemplo el determinante, es minimizada.

En otras palabras, las ecuaciones anteriores indican que si la estimación $(\mathbf{a}_1, \mathbf{A}_1)$ es consistente, entonces la traslación del vector $\mathbf{a}_1$ a $\mathbf{u}$ requiere que su covarianza se aumente por la suma de una matriz al menos tan grande como el producto de $(\mathbf{u} - \mathbf{a}_1)$ para poder ser consistente. La misma situación se debe cumplir para que la medida $(\mathbf{a}_2, \mathbf{A}_2)$ sea consistente.

Un estrategia simple, es elegir que la media de la estimación sea el valor de la medida de una de las entradas $(\mathbf{u} = \mathbf{a}_1)$. En este caso, se debe elegir el valor de $\mathbf{U}$ de forma que la estimación sea consistente con el peor caso (es decir que la medida correcta sea $\mathbf{a}_2$). Sin embargo, se puede asignar a $\mathbf{u}$ un valor intermedio entre $\mathbf{a}_1$ y $\mathbf{a}_2$ de forma que el valor de $\mathbf{U}$ sea menor. La unión de covarianzas, establece por tanto, el valor fusionado estimado medio $\mathbf{u}$ que tiene la menor covarianza $\mathbf{U}$, pero que es suficientemente grande para que las dos medidas ($\mathbf{a}_1$ y $\mathbf{a}_2$) sean consistentes.

Debido a que las desigualdades de las matrices de (2.52) son convexas, se pueden utilizar varios algoritmos de optimización convexa, para resolverlas. El valor de $\mathbf{U}$ puede calcularse con el método iterativo descrito por Julier et al. (2004). La covarianza resultante, puede ser significativamente más grande que cualquiera de las covarianzas iniciales, lo cual es un preciso reflejo de la incertidumbre que existe entre las dos estimaciones iniciales.

Una de las ventajas del método de unión de covarianzas, es que el mismo razonamiento que se ha realizado sobre dos entradas, se puede extender a N entradas.

2.8.5. Fusión distribuida de decisiones

El estudio de las técnicas y métodos de la fusión distribuida de decisiones queda fuera del alcance de este documento. No obstante, para seguir una coherencia en la estructura del estado actual de las técnicas de fusión de datos, se mencionan los siguientes trabajos.

En primer lugar, Varshney (1997) describe ampliamente el problema de la fusión distribuida para realizar decisiones. En segundo lugar, una arquitectura distribuida para realizar inferencia de forma robusta frente a fallos de los sensores, es presentada por Paskin et al. (2005). El núcleo de la arquitectura es una estructura de datos llamada "junction tree" (Aji & McEliece, 2000) que permite que los problemas de inferencia sean solucionados con algoritmos de paso de mensajes.

2.9. Resumen crítico

Existe una amplia gama de métodos, técnicas y algoritmos para realizar fusión de datos, tanto en su vertiente centralizada como distribuida. Sin embargo, a pesar del gran número de técnicas existentes, estas no están exentas de problemas, por tanto, la elección de la técnica apropiada depende en gran medida del tipo de problema a resolver y del cumplimiento de las precondiciones que establecen cada una de ellas.

Los métodos estadísticos de fusión de datos (PDA, JPDA, MHT, Kalman, etc...) solo son apropiados en determinadas circunstancias por diversas razones (Cox, 1993).

En primer lugar, la precondición de que los objetivos se mueven independientemente y sobre todo que las medidas están distribuidas de forma normal alrededor de la posición predicha, no se suelen cumplir.

En segundo lugar, debido a que las técnicas estadísticas modelan todos los eventos como probabilidades, suelen tener bastantes parámetros (como los del filtro de Kalman) y probabilidades a priori para medidas falsas y detecciones erróneas. En general, no es sencillo determinar el valor óptimo de estos parámetros. Por ejemplo, en el caso de MHT, se requiere establecer el valor de parámetros específicos que no son triviales de determinar, y que además son bastante sensibles (Veenman et al., 2001).

Además, el conocimiento a priori utilizado en los modelos estadísticos, no realiza diferencias entre los distintos objetivos, como consecuencia de eso, la inicialización puede verse en dificultad si las velocidades iniciales de los objetivos son muy divergentes, debido a que el estado de los modelos de movimiento solo se adaptaría de forma gradual.

Por último, los métodos estadísticos que optimizan a lo largo de varios fotogramas son, por sus aproximaciones, computacionalmente intensivos, ya que la complejidad crece exponencialmente con el número de objetivos. Por ejemplo, en el caso de los filtros de partículas, el seguimiento de múltiples objetivos se puede realizar de forma individual para cada uno de ellos o de forma conjunta. El seguimiento conjunto de múltiples objetivos tiene el problema de que conforme aumentan los objetivos, el número de partículas necesarias crece exponencialmente, por tanto, es preferible realizar el seguimiento de cada uno de los objetivos de forma independiente con la suposición básica de que los objetivos solo interactúan en muy pocas ocasiones.

Por otro lado, la familia de técnicas cualitativas y basadas en heurísticas, como por ejemplo las técnicas de inteligencia artificial, aportan una mayor flexibilidad al modelo y además no tienen el inconveniente de requerir modelos analíticos de los sensores. Estas técnicas en lugar de utilizar funciones de densidad de probabilidad, utilizan heurísticas y razonamiento para restringir posibles situaciones. Sin embargo, existen pocos trabajos para utilizar estas técnicas dentro de sistemas de fusión distribuidos en redes de sensores visuales.

Los sistemas de fusión distribuidos implican nuevos retos en el proceso de fusión frente a los sistemas centralizados, siendo los más importantes: (1) la alineación espacial y temporal de la información, (2) las medidas fuera de secuencia y (3) la correlación de los datos.

Los sistemas de fusión distribuidos deberían ser capaces de detectar inconsistencias en los distintos modelos y con el desconocimiento de los detalles de error. La redundancia inherente de los sistemas distribuidos se puede aprovechar con técnicas de razonamiento distribuido y cooperación para mejorar las estimaciones individuales de los nodos. En el caso particular de los sistemas multi-camara, las técnicas de fusión de datos proporcionan una forma de solucionar los problemas de las oclusiones y de los errores en la obtención de las medidas.

3

Redes de sensores visuales

Cuatro ojos ven más que dos.
Refrán popular.

3.1. Introducción

En este capítulo se describen las características y particularidades de las redes de sensores visuales frente a otro tipo de sensores, destacando las publicaciones que se han considerado más relevantes en el contexto de este trabajo.

En la sección 1.1 se definieron las redes de sensores visuales como: *un conjunto de cámaras, distribuidas de forma lógica, espacial o geográfica en un entorno y conectadas por una red de transmisión*. Las redes de sensores visuales (Patricio et al., 2007), están directamente relacionadas con la distribución espacial que existe en los entornos multi-cámara. En una red visual, los sensores deberían ser capaces de procesar los datos de las imágenes de forma local y extraer la información relevante, colaborar con otros sensores para tareas concretas y proporcionar al usuario del sistema una información detallada de los eventos capturados. Las redes de sensores visuales, son extremadamente útiles debido a que proporcionan una mayor cobertura geográfica y la posibilidad de explotar la redundancia de la información percibida. Las redes de sensores visuales, establecen la posibilidad de crear un gran número de aplicaciones novedosas basadas en la visión artificial (ver tabla 3.1). Una de las principales aplicaciones, es la vídeo vigilancia, que hoy en día, es sin lugar a dudas, una potente herramienta de seguridad pública. Con la creciente demanda de la seguridad en los aeropuertos, entornos marítimos y otros entornos críticos, la necesidad de desarrollos rápidos y flexibles en los sistemas de vigilancia está creciendo. Prácticamente todos estos sistemas requieren de una amplia distribución geográfica que hace necesaria la gestión y comunicación de los datos obtenidos por los diferentes sensores. El desarrollo e implementación de las redes de sensores visuales, genera varios problemas que son necesarios resolver. Algunos de estos problemas están relacionados con las técnicas de fusión de datos para combinar la información compartida por los diferentes sensores (Collins et al., 2001), mientras que otros están relacionados con los aspectos de tratamiento de la información visual, comunicación y de gestión de sensores (Regazzoni et al., 2001). En el caso particular de las redes de sensores visuales aplicadas a vigilancia, conocidas como sistemas de vigilancia de tercera generación (Regazzoni et al., 2001) (Valera & Velastin, 2005), su principal meta es conseguir que los objetos o personas existentes dentro del campo visual sean detectados, seguidos y reconocidos correctamente. Las redes de sensores visuales, también pueden ser vistas como sistemas empotrados distribuidos con restricciones de energía, recursos de ancho de banda y capacidades de procesado limitadas. Sin embargo, este no es el enfoque utilizado en este trabajo, donde no se tienen en cuenta de forma estricta las restricciones de energía.

Tabla 3.1: Algunas de las aplicaciones de las redes de sensores visuales

Objetivo General	Aplicación concreta
Vigilancia	Sitios públicos Autopistas, aeropuertos Aparcamientos
Monitorización del Entorno	Ríos, embalses Animales Edificios
Casas inteligentes	Cuidado de ancianos Cuidado de niños
Reuniones inteligentes	Teleconferencia Estudios virtuales
Realidad aumentada	Sistemas de telepresencia Juegos con avatares

Uno de los sistemas de redes de sensores visuales más conocido, es el sistema W4 (Haritaoglu et al., 2000), el cual fue diseñado para realizar una vigilancia exterior y un análisis de la actividad con cámaras monocromo y estacionarias. En el sistema W4, los objetos detectados deben estar separados y no ocluidos para el buen funcionamiento del sistema. El proyecto VSAM (Collins et al., 2001), es otro sistema que se basa en el seguimiento de múltiples objetivos en un entorno multi-cámara. Sin embargo, a diferencia de la propuesta realizada en este trabajo, el énfasis del proyecto VSAM es proporcionar una interfaz unificada para un operador humano en lugar de un seguimiento cooperativo completamente automático y realimentado. A continuación se presentan las problemas específicos de las redes de sensores visuales que aparecen cuando se desea realizar un seguimiento coherente de los objetivos detectados. En concreto se comentan los problemas de la instalación de los sensores (3.2.1), de la estimación del estado en redes de sensores visuales (3.2.2), del cambio de contexto (3.2.3), de la fusión de datos en redes de sensores visuales (3.2.4) y del manejo de las oclusiones (3.2.5). Posteriormente, se dedica una sección al problema de la calibración del entorno (3.3) donde se detalla el modelo Tsai de calibración (3.3.1) utilizado en este trabajo y las métricas correspondientes para evaluar la calidad de la calibración realizada (3.3.2).

3.2. Particularidades de las redes de sensores visuales

Una de las principales diferencias de las redes de sensores visuales frente a otros sistemas distribuidos, es el tipo de información que obtienen. La información proporcionada por los sensores visuales es en forma de imágenes, por tanto cada medida obtenida por el sensor visual proporciona un conjunto de puntos en dos dimensiones; esto implica que proporcionen resultados con un contenido más rico (es decir, con más información) que otro tipo de redes, así como una mayor complejidad en el procesado de los datos obtenidos y su análisis posterior. A continuación se detallan los principales problemas que hay que solucionar para desplegar una red de sensores visuales y poder realizar de forma coherente la detección y el seguimiento de los objetivos.

3.2.1. Instalación de los sensores visuales

La forma en que se realiza el despliegue de los sensores visuales tiene una gran influencia en el rendimiento y en el coste del sistema; además, para su despliegue es necesario tener en cuenta las características del terreno. Por un lado, cámaras redundantes proporcionan más información pero incrementan el coste de instalación, por otro lado, la falta de cámaras puede provocar que algunos sitios de interés se queden sin cubrir. Por tanto, el problema de como cubrir el entorno con el número mínimo de cámaras es importante y ha sido objeto de estudio por varios investigadores. El número, tipo, localización y densidad de los sensores determinan la configuración de la red de sensores. El número de los sensores utilizados es un factor importante en términos del tiempo, dinero y esfuerzo empleado y debería ser limitado por la ganancia de información que se obtiene. Un poco de redundancia suele ser necesaria para la detección de errores y la tolerancia de fallos. Sin embargo, la definición y el cálculo óptimo del número de sensores necesarios en un sistema es complejo.

Pavlidis et al. (2001) presentaron un algoritmo óptimo para el problema de la instalación de múltiples cámaras en un aparcamiento. La idea de los autores, es ir colocando cámaras una a una, manteniendo la restricción de que se solapen visualmente en un 25 %-50 %. Básicamente se puede concluir que el despliegue de las cámaras depende del objetivo que vaya a tener la red de sensores visuales y es necesario que exista un balance entre la disponibilidad de los sensores, la facilidad de despliegue, el rendimiento y el coste de instalación.

3.2.2. Estimación del estado en redes de sensores visuales

El seguimiento visual de un objetivo puede verse como un problema de inferencia en el cual, el objetivo en movimiento tiene un estado interno que es medido/observado en cada uno de los fotogramas capturados por el sensor visual. La disponibilidad en las redes de sensores visuales de múltiples vistas del objetivo puede proporcionar una mejora en el seguimiento, a costa de incrementar la comunicación entre los sensores. Por tanto, es necesario combinar las medidas/observaciones del objetivo de la forma más efectiva posible para estimar el estado del mismo. Las medidas/observaciones del objetivo pueden ser la posición de algunos puntos de la imagen, la posición o momentos de algunas regiones de la imagen o cualquier otra información obtenida a partir de los fotogramas. En las secciones 2.7.2 y 2.8.4 se describieron los algoritmos de seguimiento de forma general, que cuando son aplicados en redes visuales, deben combinar elementos de la detección y reconocimiento de los objetos con la componente temporal. Es decir, los objetos en primer lugar deben ser detectados en cada uno de los fotogramas, a continuación, se deben establecer correspondencias entre ellos, fotograma a fotograma, para poder ser estimados. Esto implica que la asociación de datos y la detección de los objetos sean la mayor fuente de dificultades en las aplicaciones de seguimiento visuales, ya que una asociación/detección errónea afecta enormemente al seguimiento. En la tabla 3.2, se muestra un resumen de los aspectos que deben gestionar los algoritmos de estimación cuando son aplicados a las redes de sensores visuales. Dentro del contexto de las redes de sensores visuales, se puede realizar la siguiente clasificación de los algoritmos de seguimiento:

1. **Seguimiento basado en características:** Se realiza el seguimiento de partes de los objetos (características), como por ejemplo, las esquinas o puntos específicos de los objetos. Los problemas de estos algoritmos aparecen cuando los objetivos tienen poca

resolución y cuando se produce el agrupamiento de las características que se encuentran bajo seguimiento.

2. **Seguimiento basado en contornos:** Se realiza el seguimiento de los contornos de los objetos. El gran inconveniente de estos algoritmos son las oclusiones parciales de los objetos.

3. **Seguimiento basado en regiones:** Estos algoritmos realizan el seguimiento de blobs[15] que corresponden de manera aproximada a los objetos. El principal inconveniente vuelven a ser las oclusiones parciales. Este es el tipo de algoritmo de seguimiento utilizado en este trabajo.

4. **Seguimiento basado en modelos:** Existe un conocimiento a priori de la forma del objeto que es modelado y utilizado. Sus principales problemas son el coste computacional y la necesidad de un gran conjunto de modelos para llevar a cabo del seguimiento de una forma apropiada.

Los problemas de estimación del estado han sido resueltos matemáticamente hace bastante tiempo; sin embargo, su aplicación en las redes de sensores visuales genera varios problemas añadidos.

En primer lugar, cuando se aplican los problemas de estimación a fuentes de información basadas en imágenes, es necesario realizar el modelado del objeto. Es decir, determinar dentro de la imagen que es lo que constituye el objeto que se desea seguir en términos que puedan ser computables. El modelado del objeto consiste en encontrar una buena descripción visual del mismo y por lo general se utilizan características similares a las que utilizamos las personas; tales como características de forma, características de color, texturas, características de movimiento, etc..

En segundo lugar, el cambio de apariencia de los objetos es un gran problema en el proceso de estimación visual. Aparte de algunos casos concretos, como el seguimiento de objetos que permanecen sin modificar su aspecto (por ejemplo, balones de baloncesto), la gran mayoría de los objetivos bajo seguimiento cambian de apariencia cuando son vistos desde distintos ángulos. Además los objetivos bajo seguimiento pueden modificar su apariencia en el tiempo, como en el caso del seguimiento de personas, las cuales modifican su aspecto al mover los brazos.

En tercer lugar, los cambios de iluminación presentan otro conjunto de problemas que afectan a la calidad del seguimiento visual de objetos. Los cambios de iluminación consisten en el principal problema para los algoritmos que utilizan el modelado del color para definir la apariencia del objeto.

En cuarto lugar, el problema clásico en el seguimiento de personas viene dado por la proyección de la sombra de las mismas en el suelo. Este problema deriva en considerar que la sombra forma parte de la persona, dando lugar a grandes modificaciones en el tamaño del objeto bajo seguimiento.

La información temporal presenta otro problema, ya que el contenido extraído de una imagen deja de tener sentido si no se tiene una referencia temporal del instante al que se refiere. Por tanto, la gran mayoría de los algoritmos de estimación en redes de sensores

[15] Un blob se define como un conjunto de píxeles en movimiento que pertenecen al mismo objeto.

Tabla 3.2: Aspectos que deben gestionar los algoritmos de estimación cuando son aplicados para realizar el seguimiento de objetivos en redes de sensores visuales.

- Inicialización de las pistas
- Terminación de las pistas
- Gestión de las oclusiones (parciales o totales)
- Separación y unión de pistas
- Actualización del modelo del objetivo

visuales, requieren que los relojes de los sensores, donde se obtienen las imágenes, estén correctamente sincronizados entre ellos.

Finalmente, las imperfecciones en la tecnología de las cámaras y el uso de una red distribuida presentan retos y problemas que deben ser solucionados con las técnicas de fusión de datos apropiadas.

3.2.3. Cambio de contexto

Uno de los problemas que aparecen en las redes de sensores visuales o en los entornos multi-cámara es el cambio de contexto. El cambio de contexto se produce cuando un objeto se acerca al límite del campo de visión de una cámara, y por tanto es necesario establecer cual es la siguiente cámara que va a obtener la visión del objeto, de forma que se pueda continuar el seguimiento utilizando la información proporcionada por la nueva cámara. Los problemas clave aquí, son: como asegurar la continuidad en el seguimiento y como minimizar los cambios de contexto durante el seguimiento. El problema del cambio de contexto aparece sobre todo en las redes visuales con áreas sin solapar. Para afrontar el problema, Cai & Aggarwal (1996, 1999) propusieron establecer una confianza de seguimiento a cada objeto, cuando la confianza cae por debajo de un umbral, el sistema empieza una búsqueda para encontrar la siguiente cámara activa con la mayor confianza. La confianza en el seguimiento es una medida robusta del seguimiento entre fotogramas consecutivos basada en la distancia de Mahanalobis; su limitación consiste en que el sistema está restringido a una solo persona y no contempla el seguimiento de múltiples objetivos.

3.2.4. Fusión de datos en redes de sensores visuales

Los métodos y técnicas de fusión de datos comentados en el capítulo 2, cobran especial relevancia dentro de las redes de sensores visuales. La principal aportación de este trabajo es el uso de técnicas de fusión de datos en las redes de sensores visuales por medio de los sistemas multi-agente (los cuales se describen en el capítulo 4). Este apartado, es solo una pequeña muestra de algunos trabajos que utilizan técnicas de fusión de datos en el dominio de las redes de sensores visuales. Además de las ventajas comentadas, la fusión de datos es de gran utilidad para el solucionar el problema de las oclusiones y asegurar la continuidad en el seguimiento de los objetos.

Cox & Hingorani (1996) fueron los primeros en utilizar el enfoque de fusión de datos probabilístico, de la comunidad de radar, para aplicarlo en tareas de visión artificial. Siguiendo un enfoque probabilístico, Kettnaker & Zabih (1999) reconstruyen las trayectorias de los

objetos en un entorno de múltiples cámaras con áreas solapadas. Para reducir el número de hipótesis, utilizaron la restricción de que un objeto solo puede ser visto por una cámara. El enfoque de fusión probabilístico también es usado por Dockstader & Tekalp (2001) que utilizan una red Bayesiana para fusionar vectores de estado en dos dimensiones adquiridos por varias cámaras y obtener un vector de estado en tres dimensiones. La red Bayesiana utilizada, no tiene en consideración la correlación entre las vistas proporcionadas, y asume los datos de cada vista como variables independientes. Por otro lado, en el trabajo de Collins et al. (2001) se realiza la fusión de datos mapeando los datos de cada sensor por medio de un proceso de geo-localización utilizando las coordenadas geodésicas. La fusión se lleva a cabo por medio de una función de concordancia que tiene en cuenta la localización, el color y el tipo de objeto.

El trabajo de Snidaro et al. (2004, 2003), es similar a esta propuesta, ya que consiste en mejorar la precisión del seguimiento por medio de una fusión de datos que explota la redundancia proporcionada por múltiples cámaras. Se utiliza una métrica denominada *Aspect Ratio* (AR) para controlar el proceso de fusión de acuerdo a una medida de confianza de cada sensor visual. La medida de AR se aplica a cada objeto extraído de cada cámara, de esta forma la fusión se lleva a cabo ponderando la información de cada una de las cámaras. Una medida similar se puede asociar al nivel de creencia que tienen los agentes sobre la posición de los objetos observados.

3.2.5. Manejo de las oclusiones

En la práctica, las oclusiones propias u oclusiones entre objetos son inevitables. Los sistemas de redes de sensores visuales o sistemas multi-cámara pueden proporcionar los métodos necesarios para evitar los problemas de las oclusiones. De los trabajos publicados relacionados con las redes de sensores visuales, se deduce que la solución más práctica para manejar las oclusiones es utilizar varias cámaras. Esta es la idea que subyace bajo el trabajo de Utsumi et al. (1998), donde se utilizan múltiples cámaras para el seguimiento de personas y se resuelven los problemas de las oclusiones seleccionando la mejor vista de cada cámara. El trabajo de Tsutsui et al. (2001) también utiliza un sistema multi cámara, pero se basa en el flujo óptico. El seguimiento se lleva a cabo mediante un intercambio de la información entre las cámaras para evitar las oclusiones. El enfoque de utilizar múltiples cámaras para evitar las oclusiones en entornos densos también es propuesto por Mittal & Davis (2003); también en entornos densos, pero con el objetivo de contar personas, el trabajo de Yang et al. (2003) utiliza 8 cámaras. La información obtenida por cada una de las cámaras proporciona la silueta de la persona detectada y es enviada a un nodo central. En el nodo central se calcula la posición de las personas por medio de una proyección de la intersección de las siluetas de las distintas cámaras. El trabajo de Yang et al. (2003) escala bien y es robusto frente a fallos en las cámaras.

(Khan & Shah, 2009) también proponen el uso de un sistema multi-cámara para resolver las oclusiones que se pueden producir cuando se realiza el seguimiento de las personas. En su trabajo utilizan las homografías entre las diferentes vistas de cada una las cámaras y el plano del suelo para generar restricciones en las cuales si una persona es vista por una cámara, debe también ser observada desde las demás, ya que realizan la suposición de que todas las cámaras deben compartir el área de visión.

3.3. Calibración del entorno

La calibración del entorno se debe realizar en una fase previa para poder realizar de forma adecuada la fusión de los datos de los diferentes sensores visuales, es el primer paso para poder realizar correctamente el proceso de fusión de datos y además uno de los más importantes. El problema de la calibración geométrica de los sensores visuales se basa en estimar los parámetros extrínsecos e intrínsecos de la cámaras. Por tanto, la calibración, consiste en la alineación espacial de la información de los diferentes sensores, que implica utilizar un marco de coordenadas común para representar los objetos, proyectando las coordenadas del plano local de cada sensor al plano global del conjunto.

La calibración visual de los sensores, requiere de un cierto número de puntos de correspondencia para realizar la transformación geométrica entre las diferentes vistas y poder relacionar las medidas de los sensores en dos dimensiones, con las medidas en el mundo real en tres dimensiones. La relación entre las unidades de los sensores visuales (píxeles) y las unidades en el mundo físico (por ejemplo, metros) es un componente crítico en cualquier aplicación de visión artificial.

El proceso de calibración del sensor, proporciona un modelo de la geometría de la cámara y de la distorsión del lente, es decir, define lo que se conoce como los parámetros intrínsecos de la cámara.

Los algoritmos que reconstruyen la estructura 3D a partir de una imagen, o calculan la posición de los objetos en el espacio, necesitan ecuaciones para hacer corresponder los puntos en 3D con sus correspondientes proyecciones en el plano 2D. Para ello, se utiliza una colección de puntos con posiciones conocidas en un sistema fijo de coordenadas. Por tanto, el problema de calibración puede modelarse como un proceso de optimización, donde la diferencia entre la posición del plano de la imagen y la posición en el plano global, es minimizada con respecto a los parámetros intrínsecos e extrínsecos. La tarea de obtener el mejor valor posible de estos parámetros es lo que se conoce, en visión artificial, como calibración.

Varias técnicas de calibración han sido propuestas, siendo las de Tsai (1987), Heikkila (2000) y Zhang (2000) las más utilizadas e importantes. En este trabajo se utiliza el modelo de cámara de Tsai (1987) para llevar a cabo la calibración de los sensores. De acuerdo con Salvi et al. (2002), el método Tsai es el que proporciona el mejor rendimiento.

La calibración se vuelve más compleja cuando es necesario tener en cuenta múltiples cámaras; la calibración puede llevarse a cabo como un paso previo a la ejecución del sistema y, por tanto realizarse solo una vez si se trata de cámaras estáticas, o bien se puede realizar de forma online durante la ejecución del sistema. Algunos de los métodos de calibración online utilizan información temporal de las posiciones de los diferentes sensores visuales para realizar la correspondencia. Stein (1999) y Lee et al. (2000) utilizan la trayectoria del movimiento y la restricción del plano del suelo para determinar la matriz de transformación de la proyección, y luego, esta matriz se descompone para obtener los parámetros extrínsecos de la cámara. Lee et al. (2000) restringen el movimiento de los objetos en el plano del suelo y hacen concordar los objetos entre las diferentes vistas de las cámaras; a continuación obtienen la matriz de transformación de la proyección a partir de la cual obtienen las posiciones en el plano del suelo. La principal aportación de su trabajo es demostrar que, utilizando solo las pistas de los objetos en movimiento, es posible obtener las posiciones en 3D de los objetos y de las cámaras. Utilizan los centroides de los objetos bajo seguimiento como las características para

establecer una marco de coordenadas común entre todos los sensores visuales.

Intille & Bobick (1995) utilizan las medidas de las líneas de un campo de fútbol americano para transformar las posiciones de los objetos en un sistema de coordenadas global y único. La diferencia respecto al trabajo de Lee et al. (2000) es que estos últimos no emplean ninguna marca en las imágenes como Intille y Bobick, sino que se apoyan en un sistema multi-cámara para obtener un sistema de coordenadas global.

3.3.1. Método Tsai de calibración

El modelo de cámara Tsai (1986, 1987), está basado en el modelo pinhole de proyección de la perspectiva 3D-2D, con lentes de distorsión radial de primer orden. Para llevar a cabo la calibración, el modelo necesita la correspondencia entre una serie de puntos en 3D y sus correspondientes puntos en el plano de la imagen en 2D. El modelo Tsai consta de 11 parámetros, 5 de ellos son parámetros intrínsecos o internos (los cuales relacionan el sistema de referencia de la cámara a la imagen) y 6 son parámetros extrínsecos o externos (que relacionan los sistemas de referencias en 3D). Los 5 parámetros intrínsecos se definen como:

1. **f:** distancia focal efectiva de la cámara pinhole.
2. **kappa1:** coeficiente de la distorsión radial del lente de primer orden.
3. **sx:** factor de escala, para tener en cuenta cualquier incertidumbre en el muestreado por la tarjeta capturadora de las lineas horizontales.
4. **Cx, Cy:** coordenadas del centro de la distorsión radial y el punto de perforación del marco de coordenadas del eje Z de la cámara con el plano del sensor.

Por otro lado, los 6 parámetros extrínsecos, consisten en:

1. **Rx, Ry, Rz:** ángulos de rotación de la transformación entre el sistema de coordenadas del mundo y el sistema de coordenadas del plano de la cámara.
2. **Tx, Ry, Tz:** componentes de traslación de la transformación entre las coordenadas del mundo y las coordenadas del plano de la cámara.

El algoritmo utiliza una técnica en dos pasadas, para calcular, en primer lugar, los parámetros de la posición y la orientación, y a continuación los parámetros intrínsecos.

Una vez calculados los parámetros, la transformación de las coordenadas del mundo (X_w, Y_w, Z_w) a las coordenadas del plano de la imagen (X_i, Y_i, Z_i) se realiza utilizado los parámetros extrínsecos de la cámara (la rotación R y traslación T) con la siguiente ecuación:

$$\begin{bmatrix} X_i \\ Y_i \\ Z_i \end{bmatrix} = R \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} + T$$

donde las matrices R y T caracterizan la transformación de las coordenadas 3D del mundo al sistema de coordenadas de la cámara y se definen de la siguiente forma:

$$R = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix}$$

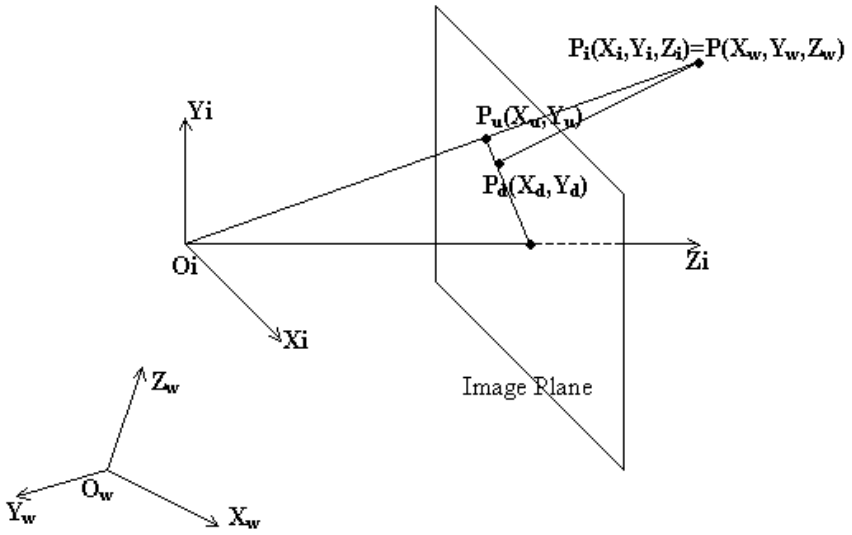

Figura 3.1: Esquema del modelo Tsai con la proyección de la perspectiva y la distorsión radial.

$$T = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix}$$

donde:

$r_1 = cos(R_y)cos(R_z)$
$r_2 = cos(R_z)sin(R_x)sin(R_y) - cos(R_x)sin(R_z)$
$r_3 = sin(R_x)sin(R_z) + cos(R_x)cos(R_z)sin(R_y)$
$r_4 = cos(R_y)sin(R_z)$
$r_5 = sin(R_x)sin(R_y)sin(R_z) + cos(R_x)cos(R_z)$
$r_6 = cos(R_x)sin(R_y)sin(R_z) - cos(R_x)sin(R_x)$
$r_7 = -sin(R_y)$
$r_8 = cos(R_y)sin(R_x)$
$r_9 = cos(R_x)cos(R_y)$
(R_x, R_y, R_z) son los ángulos de rotación alrededor de los tres ejes
(T_x, T_y, T_z) son los parámetros de traslación 3D de las coordenadas del mundo a la imagen

La transformación de las coordenadas en 3D al plano de la imagen (ver imagen 3.1), se calcula, llevando a cabo los siguientes pasos:

1. Transformación a las coordenadas del plano de la imagen no distorsionada (X_u, Y_u):

   $X_u = f\frac{X_i}{Z_i}$
   $Y_u = f\frac{Y_i}{Z_i}$

2. Transformación de las coordenadas no distorsionadas (X_u, Y_u) a las coordenadas de la imagen distorsionadas (X_d, Y_d):

$X_u = X_d(1 + kr^2)$
$Y_u = Y_d(1 + kr^2)$

donde:

$r = \sqrt{{X^2}_d + {Y^2}_d}$ y k es el coeficiente de distorsión del lente.

3. Transformación de las coordenadas distorsionadas del plano de la imagen (X_d, Y_d) a las coordenadas finales de la imagen (X_f, Y_f):

$X_f = \frac{S_x X_d}{d_x} + C_x$
$Y_f = \frac{Y_d}{d_y} + C_y$

donde:

d_x y d_y son parámetros fijos de la cámara que dependen del lente y de la resolución de la imagen.
X_f y Y_f son las posiciones finales de los píxeles en la imagen.

Existen aproximaciones que simplifican el modelo pinhole añadiendo algunas restricciones basadas en el conocimiento inicial del entorno. En este trabajo se ha utilizado un modelo de calibración simple basado en la técnica Tsai, usando la vista de cada una de las cámaras y añadiendo la restricción de que todos los puntos están en el mismo plano (ver figura Fig. 3.2), normalmente llamado plano del suelo o *ground-plane*.

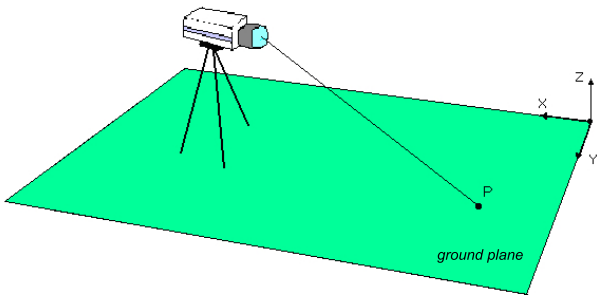

Figura 3.2: Ilustración de la proyección en el plano del suelo.

En el modelo Tsai, las coordenadas del mundo en 3D se deben establecer utilizando un sistema de coordenadas en el cual el origen $(0, 0)$ se encuentre en la esquina superior izquierda. Este modelo de calibración tiene dos variantes: datos no coplanares y datos coplanares. Para los datos coplanares, el algoritmo Tsai requiere que la componente Z sea 0. Dado que lo que se desea es realizar el seguimiento en el plano del suelo, este requisito se puede cumplir fácilmente. Todos los sensores visuales se van a calibrar con respecto a la misma plantilla de calibración. Es decir, se obtiene una homografía del plano de la imagen al plano del mundo 3D (ground plane) que consiste en la relación entre las coordenadas de los puntos en 3D (X, Y, Z) y sus correspondientes coordenadas en el plano de la imagen (X_i, Y_{pi}) en píxeles. La calibración coplanar básica necesita por lo menos 5 puntos para estimar con precisión

la distorsión radial y los parámetros del centro de la imagen. Además es deseable que el conjunto de puntos estén distribuidos a lo largo del campo de visión.

Por tanto, la restricción de movimiento en el plano se utiliza para estimar la homografía entre diferentes vistas y evitar el uso de mecanismos de visión estereoscópica más complejos.

3.3.2. Evaluación de la fase de calibración

La calidad de la calibración realizada depende de varios factores, como por ejemplo, lo bien distribuidos que se encuentran los puntos dentro de la zona que se va a calibrar o los errores en la obtención de las medidas. Con el propósito de evaluar la calidad de la calibración realizada, se van a utilizar las siguientes métricas:

1. **Error con distorsión en las coordenadas de los píxeles (E_d):** este indicador, se obtiene calculando la diferencia entre las coordenadas de los píxeles estimadas $(\hat{x}_{pi}, \hat{y}_{pi})$, obtenidas a partir de las coordenadas del mundo que se han medido y aplicando el modelo de cámara con la distorsión del lente, y las coordenadas de los píxeles observadas (x_{pi}, y_{pi}), las cuales se han obtenido de forma manual, de la siguiente forma:

$$E_d = \frac{1}{n}\sum_{i=1}^{n}\sqrt{(\hat{x}_{pi} - x_{pi})^2 + (\hat{y}_{pi} - y_{pi})^2} \tag{3.1}$$

   donde n es el número de puntos de entrada del algoritmo de calibración.

2. **Error sin distorsión de las coordenadas de lo píxeles (E_u):** este indicador se calcula de forma similar que E_d, pero sin utilizar la distorsión del lente en las coordenadas estimadas de los píxeles y eliminando la distorsión del lente en las coordenadas observadas de los píxeles, de la siguiente forma:

$$E_u = \frac{1}{n}\sum_{i=1}^{n}\sqrt{(\hat{x}_{upi} - x_{upi})^2 + (\hat{y}_{upi} - y_{upi})^2} \tag{3.2}$$

3. **El error espacial del objeto o la distancia respecto al rayo óptico (E_o):** se calcula utilizando los puntos en 3D de las coordenadas de la cámara (X_{ci}, Y_{ci}, Z_{ci}) y los rayos ópticos proyectados de los puntos sin distorsión en el plano de la imagen (x_{ui}, y_{ui}), de la siguiente forma:

$$E_o = \frac{1}{n}\sum_{i=1}^{n}\sqrt{(X_{ci} - x_{ui}\cdot t)^2 + (Y_{ci} - y_{ui}\cdot t)^2 + (Z_{ci} - t)^2}, \tag{3.3}$$

   donde $t = (X_{ci}x_{ui} + Y_{ci}y_{ui} + Z_{ci})/(x_{ui}^2 + y^2 ui + 1)$

4. **Error de calibración normalizado (NCE):** este valor, se obtiene normalizando la diferencia entre los puntos en 3D estimados y observados, con respecto al área que cubre cada píxel a una cierta distancia de la cámara, de la siguiente forma:

$$NCE = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{(\hat{X}_{ci} - X_{ci})^2 + (\hat{Y}_{ci} - Y_{ci})^2}{Z_{ci}^2(\alpha^{-2} + \beta^{-2})/12}\right)^{1/2} \tag{3.4}$$

Los primeros tres valores de precisión (E_d, E_u, E_o), son sensibles a la resolución digital de las imágenes, al campo de visión de la cámara y a la distancia del objeto a la cámara. Sin embargo, la métrica NCE propuesta por Weng et al. (1992) es insensible respecto a los anteriores parámetros y proporciona resultados de precisión menos sesgados. Estos indicadores se utilizan para evaluar la calidad de la calibración de los experimentos en el capítulo 6 y se presentan los valores que se han obtenido, después de aplicar la calibración para cada uno de los escenarios utilizados en la parte experimental.

4

Sistemas multi-agente

There are two important problems in AI. What is A and what is I?.
Donald Knuth.

4.1. Introducción

Para la correcta integración de la información proporcionada por los sensores visuales en una red cooperativa, es necesario utilizar técnicas de fusión de datos, las cuales necesitan una arquitectura software para abstraer los dispositivos físicos en un modelo lógico y proporcionar un conjunto de servicios a través de una serie de APIs estándar sobre diferentes plataformas. En este trabajo, se propone el uso de los sistemas multi-agente como el paradigma de arquitectura software que proporciona los mecanismos de comunicación y razonamiento entre los diferentes nodos de la red. Por tanto, la red de sensores visuales es modelada como un sistema multi-agente que consta de varios agentes software.

Un agente software es un proceso computacional que opera en un entorno y basa sus decisiones en lo que razona acerca de la información que obtiene del entorno. El problema de estimación, presentado anteriormente en el apartado 2.7.2, se puede relacionar con la percepción interna de un agente, debido a que ambos pretenden reconstruir un modelo del entorno a partir de las observaciones percibidas, que sea lo más semejante posible a la realidad. Una de las decisiones más importantes, a la hora de diseñar sistemas multi-agente, es decidir la información que intercambian los agentes. Algunos autores proponen el intercambio de información en bruto, mientras que otros proponen el intercambio de las creencias (beliefs) de los agentes. Independientemente del tipo de información que deben intercambiar los agentes, este intercambio debe realizarse de una forma coordinada. La coordinación se entiende como el proceso de gestionar las inter-dependencias entre diferentes agentes. Por tanto, la coordinación de los diferentes agentes del sistema es una tarea imprescindible (Lesser, 1999). La coordinación puede llevarse a cabo de forma implícita (sin comunicación) o de forma explícita (utilizando comunicación). La coordinación implícita de los agentes se basa en utilizar reglas conformes a políticas o normas que deben seguir los agentes. La ventaja de una coordinación implícita es que no requiere de comunicación y es más rápida, sin embargo, requiere de una planificación y un diseño previo completo. La coordinación se convierte en una cooperación cuando se trata de agentes que tienen un objetivo común y en una negociación cuando se trata de agentes con objetivos individuales, es decir, competitivos.

En la siguiente sección se explica más en detalle el concepto de los sistemas multi-agente. Posteriormente, en la sección 4.3 se presentan las diferentes arquitecturas de los sistemas multi-agente. A continuación, en la sección 4.4, se describe el modelo de agentes utilizado en este trabajo. En la sección 4.5 se describen las publicaciones relacionadas con sistemas

multi-agente aplicados a las redes de sensores de sensores visuales. Finalmente, en la sección 4.6 se proporciona un resumen crítico.

4.2. Descripción de los sistemas multi-agente

El paradigma de agentes define la forma de crear sistemas software concurrentes y distribuidos, los cuales presentan un bajo acoplamiento. Para conseguirlo, se asume que un agente es una entidad autónoma que actúa conforme a sus objetivos y se comunica con el resto de agentes por medio de mensajes asíncronos. Este paradigma de desarrollo de orientación a agentes, representa un nuevo nivel conceptual de abstracción, que también puede utilizar las prácticas del paradigma orientado a objetos.

Un agente software, se define como un proceso computacional que es capaz de llevar a cabo decisiones flexibles de forma autónoma para obtener sus objetivos (Wooldridge & Jennings, 1995). Por decisiones flexibles, se entienden decisiones que poseen las siguientes características:

1. **Reactividad:** Permitir a los agentes percibir y responder a entornos cambiantes para satisfacer sus objetivos, lo cual implica que los agentes deben responder con rapidez a los cambios en el entorno.

2. **Habilidad social:** Por la cual los agentes interactúan con otro agentes. Esta habilidad social se lleva a cabo por medio del uso de pizarras compartidas o el paso de mensajes entre los agentes.

3. **Proactividad:** Por la cual los agentes se comportan de una forma dirigida por objetivos y toman iniciativa para conseguir sus objetivos de diseño. La proactividad se refiere a la capacidad de anticipación de planes futuros de los agentes.

Un agente es un proceso computacional que tiene control sobre su estado interno y su propio comportamiento. Obtiene la información del entorno por medio de sensores y actúa conforme a sus objetivos, establecidos en la fase de diseño del mismo.

Un sistema multi-agente, se puede definir, como una red de agentes software que colaboran de forma conjunta y que comparten la habilidad social. Esta habilidad social, conlleva que los agentes no solo realicen comunicaciones intra-agentes (internas de los agentes), sino además, inter-agente (entre los agentes). Los sistemas multi-agente, pertenecen al campo de la inteligencia artificial, que pretende proporcionar los principios para construir sistemas complejos que involucran múltiples agentes y además los mecanismos necesarios para coordinar los comportamientos individuales de los agentes (Stone & Veloso, 2000). Un sistema multi-agente, también se puede definir como un conjunto de agentes software que trabajan de forma conjunta y que tienen las siguientes características:

1. **Falta de visión global:** Cada uno de los agentes tiene acceso solo a una parte de la información que existe en el sistema a nivel global.

2. **Control descentralizado:** El control de la información esta descentralizado y cada uno de los agentes es responsable de su propio control.

3. **Autonomía:** Los agentes se consideran procesos autónomos; de manera que eligen en que momento toman las decisiones en función de los datos que obtienen del entorno.

4. **Subsistemas asíncronos:** Cada uno de los agentes se comporta de forma asíncrona y con un bajo acoplamiento entre los demás, no siendo necesario utilizar o conocer el sistema operativo, lenguaje de programación y plataforma hardware de cada uno de ellos.

5. **Necesidad de cooperación:** Existe una necesidad explícita de cooperación entre los agentes del sistema.

La necesidad de cooperación entre los agentes de un sistema multi-agente puede venir dada por las siguientes inter-dependencias de los problemas:

1. Los subproblemas son los mismos o se solapan, pero cada uno de los agentes utilizan distintos métodos o datos para generar las soluciones individuales que se combinan para obtener la solución conjunta.

2. Los subproblemas son parte de problemas más grandes, en donde la solución general requiere satisfacer ciertas restricciones entre las soluciones de los subproblemas.

3. No es posible descomponer el problema en un conjunto de subproblemas para los cuales existe una solución perfecta, en términos de donde se localiza la información y el procesado y en términos de necesidades de comunicación necesarias para solucionar de forma efectiva cada subproblema.

En el campo de los sistemas multi-agente, los subproblemas pueden venir por la distribución espacial, temporal o funcional de la información, por los recursos o por el conocimiento.

Por tanto, una de las principales características de los agentes, es la cooperación entre ellos, para obtener un mejor rendimiento o para utilizar capacidades adicionales que no posean. Para una correcta cooperación entre los agentes es necesario definir qué se debe comunicar, cuando se debe realizar la comunicación y a quien se debe comunicar. Desde el punto de vista de la cooperación, pueden darse tres tipos distintos de interacciones: (1) interacciones de objetivos conjuntos entre los agentes, (2) interacciones de objetivos complementarios entre los agentes y (3) interacciones de objetivos en conflicto. En este trabajo, se aborda una cooperación donde los agentes comparten objetivos conjuntos, con el fin de realizar una adecuada fusión de datos para interpretar bien la situación. Esta cooperación puede surgir de forma predefinida o por medio de coaliciones dinámicas o equipos entre los agentes. El concepto de coalición, aparece cuando un grupo de agentes decide trabajar conjuntamente, de forma temporal, para obtener un objetivo común (Wooldridge, 2000). El concepto de coalición ha sido estudiado ampliamente (Kahan & Rapoport., 1984) (Raiffa, 1984) (Ketchpel, 1994) (Shehory & Kraus, 1995), aunque existen pocos trabajos relacionados con las redes de sensores visuales, destacando principalmente el de Dang et al. (2006).

Numerosos investigadores han reivindicado el uso de los sistemas multi-agente para desarrollar sistemas que obtengan los datos de una forma distribuida, donde el control y/o los recursos también pueden estar distribuidos. En este tipo de sistemas, los agentes aportan los beneficios de la escalabilidad y la modularidad para llevar a cabo un razonamiento distribuido de los datos obtenidos.

4.3. Arquitecturas de sistemas multi-agente

La arquitectura multi-agente, determina los mecanismos que utiliza un agente para reaccionar a los estímulos, actuar y comunicarse (Mas et al., 2005). Es posible realizar la siguiente clasificación (ver tabla 4.1) de las arquitecturas en las que están basados los sistemas multi-agente:

1. **Arquitecturas de agentes reactivos:** Se caracterizan por no tener como elemento central de razonamiento un modelo simbólico. Son sistemas que responden a los cambios del entorno de una forma inmediata a partir de una serie de reglas básicas. El objetivo de estos sistemas es realizar el razonamiento sin modelar de forma explicíta el conocimiento. Un agente reactivo por tanto, no tiene ningún tipo de modelo simbólico del mundo y no utiliza un razonamiento lógico complejo. Sin embargo, intenta llevar a cabo una serie de objetivos en un entorno complejo y dinámico. En este tipo de arquitecturas, el comportamiento inteligente emerge a partir de reglas simples de interacción.

2. **Arquitecturas deliberativas:** Son aquellas que utilizan modelos de representación simbólica del conocimiento, parten de un estado inicial y son capaces de generar planes para alcanzar sus objetivos (Maes, 1990). Los agentes deliberativos explícitamente representan el modelo del mundo de una forma simbólica y las decisiones se llevan a cabo por medio de razonamientos de tipo lógico. Estos tipos de agentes deben solucionar dos problemas. En primer lugar, lo que se conoce como *transduction problem*, que consiste en trasladar la información del mundo real a una descripción simbólica adecuada y precisa para poder ser utilizada a tiempo. En segundo lugar, el problema de la representación o razonamiento, que consiste en representar simbólicamente, la información acerca de entidades complejas del mundo real y sus procesos y como hacer que los agentes razonen con esta información de forma eficiente para obtener resultados útiles. Los principales inconvenientes de las arquitecturas puramente deliberativas (y en general de cualquier modelo lógico) es que al incrementar el conocimiento los problemas se vuelven muy difíciles de resolver siguiendo el esquema lógico. Algunos trabajos combinan estas arquitecturas con otros razonamientos como el de Corchado & Laza (2003), que han propuesto la construcción de agentes deliberativos utilizando la tecnología de razonamiento basado en casos y su aplicación en un entorno de comercio electrónico.

3. **Arquitecturas híbridas:** Conjugan elementos de las arquitecturas reactivas y las deliberativas. Están basadas en los modelos de razonamiento práctico y en la necesidad de buscar tanto reacciones rápidas como en razonar acerca de los siguientes objetivos a alcanzar. El primer modelo basado en una arquitectura híbrida fue el sistema PRS (Georgeff & Lansky, 1987) (Ingrand et al., 1992) y a continuación surgieron IRMA, dMARS (D'Inverno et al., 2004), JACK, Agentis, etc.

Además de las clasificaciones existentes atendiendo al tipo de arquitectura multi-agente (deliberativa, reactiva o híbrida), es posible realizar una división de los sistemas multi-agente atendiendo a estas cuatro dimensiones:

1. Agentes homogéneos no comunicativos.

2. Agentes heterogéneos no comunicativos.

3. Agentes homogéneos comunicativos.

4. Agentes heterogéneos comunicativos.

La diferencia entre los agentes homogéneos y los heterogéneos, esta basada en que los agentes homogéneos tienen todos la misma estructura interna, incluyendo sus objetivos, el conocimiento del dominio y las posibles acciones y además tienen el mismo procedimiento para seleccionar las acciones. La única diferencia entre ellos, son las entradas de la información del entorno (los sensores) y las acciones que llevan a cabo en función de esas entradas. En cambio, los agentes heterogéneos tienen objetivos distintos entre ellos y generalmente contrapuestos, lo que normalmente se traduce en un comportamiento basado en una competición y no en una colaboración. Por otro lado, tanto los agentes homogéneos como los heterogéneos, pueden no tener capacidad de comunicación, entendiendo por capacidad de comunicación el envío y recepción de mensajes.

En este trabajo, para realizar el desarrollo de redes de sensores visuales con sistemas multi-agente se ha optado por utilizar agentes homogéneos comunicativos, basados en una arquitectura híbrida de tipo Belief-Desire-Intention (BDI).

Tabla 4.1: Resumen de las diferentes arquitecturas de sistemas multi-agente

Tipo	**Características**
Reactivas	No tienen un modelo interno Los objetivos se representan implícitamente La inteligencia emerge de interacciones simples
Deliberativas	Representan de forma explícita un modelo simbólico del mundo Las decisiones se llevan a cabo por medio de razonamientos lógicos Al incrementar el conocimiento se complica el razonamiento
Híbridas	Basadas en el modelo del sentido común Reaccionan rápido y deliberan sobre los planes Ejemplos: PRS, IRMA, dMARS, Agentis, JACK

4.4. Modelo de agentes Belief-Desire-Intention (BDI)

Las arquitecturas de agentes basadas en el modelo *Belief-Desire-Intention* (BDI) se originaron con el proyecto *Rational Agency* de la Universidad de Stanford a mediados de 1980 y este modelo se corresponde con una arquitectura deliberativa híbrida. Tiene sus orígenes en el modelo conceptual descrito por Bratman et al. (1988), el cual radica en el modelo filosófico del razonamiento humano, desarrollado originalmente por Bratman (1987), que a su vez reduce la explicación del complejo comportamiento humano a una *postura de motivación* (Dennett, 1987). La postura de motivación, implica que las causas para las acciones están siempre relacionadas con los deseos humanos, ignorando otras facetas de las posibles motivaciones humanas para actuar. Además, propone usar, de una forma consistente, conceptos psicológicos que se corresponden bastante con los términos que los seres humanos utilizamos para explicar el comportamiento.

En un modelo de agentes BDI, cada agente tiene su conjunto de Creencias (Beliefs), Deseos (Desires) e Intenciones (Intentions), las cuales representan internamente el estado mental de cada uno de los agentes. Las creencias modelan los estados del mundo, los deseos hacen referencia a las elecciones entre los posibles estados y las intenciones son compromisos que se adquieren para conseguir estados concretos. Las creencias definen el conocimiento parcial que un agente tiene acerca del mundo en forma de hechos observados. Los deseos representan los estados que el agente le gustaría obtener o alcanzar. Finalmente, las intenciones representan los deseos que el agente se ha comprometido a alcanzar. En un instante determinado un agente puede no ser capaz de satisfacer todos sus deseos y estos pueden no ser consistentes entre sí.

El modelo de razonamiento práctico, sobre el cual están basadas las arquitecturas BDI, involucra dos procesos importantes: (1) decidir que objetivos alcanzar y (2) como alcanzar esos objetivos. El primer proceso se conoce como *deliberación* y el último como *razonamiento final o acción*. El proceso de deliberación implica seleccionar objetivos, ponderando los diferentes deseos, y generar intenciones. El razonamiento final implica llevar a cabo las acciones oportunas, considerando los recursos disponibles, además de generar planes y convertirlos en acciones. Las intenciones de los agentes BDI tienen las siguientes propiedades:

- Una vez que una intención se ha adoptado el agente intenta llevarla a cabo.
- Una vez que una intención se ha adoptado, el agente persiste hasta que se lleve a cabo o se considere que es imposible de completar.
- Las intenciones actuales pueden excluir otras intenciones u opciones.
- Un agente solo puede adoptar una intención si se dan las precondiciones para conseguirla.

Para hacer la teoría de agentes BDI computacionalmente tratable, Rao y Georgeff simplificaron la teoría y solo representaron explícitamente las creencias, mientras que los deseos fueron reducidos a eventos que disparan plantillas de planes predefinidas y las intenciones fueron representadas explícitamente en la pila de ejecución de los planes que se ejecutan. En la teoría del modelo de agentes BDI de Rao & Georgeff (1995), se definen las Creencias (Beliefs), Deseos (Desires) e Intenciones (Intentions) como actitudes mentales que representan posibles estados del mundo. A continuación se presenta el ciclo de ejecución básico del interprete de razonamiento de un agente BDI (ver también figura 4.1).

```
Inicializar-estado()
Repetir
 Opciones := generador-opción(cola-eventos, S);
 Opciones-seleccionadas := deliberar(Opciones, S);
 Modificar-intenciones(Opciones-seleccionadas, S);
 Ejecutar(S);
 Cola-eventos := Tomar-nuevos-eventos-externos();
Fin-repetir
```

Por tanto, un agente BDI tiene sus propias creencias (acerca de sí mismo y del entorno), sus deseos (son los estados que quiere obtener) e intenciones (que consisten en los planes adoptados). Además, cada agente BDI mantiene un repositorio de planes disponibles, conocido como librería de planes, que se ejecutan en base a las intenciones seleccionadas. Los

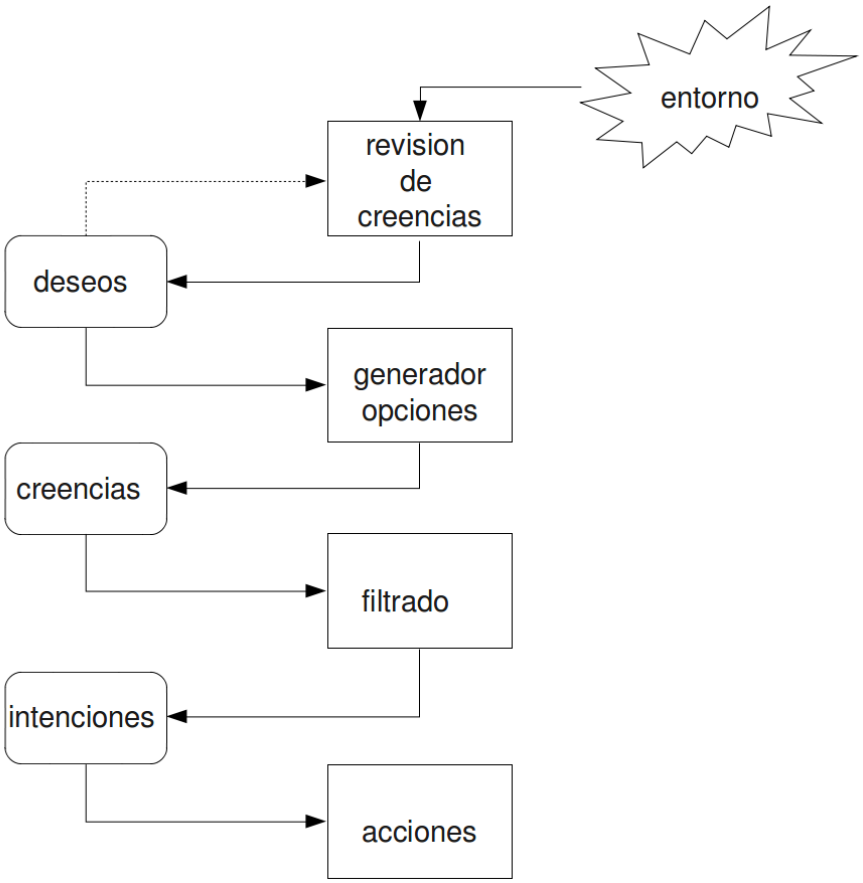

Figura 4.1: Esquema simplificado de la arquitectura BDI.

agentes BDI responden a cambios en sus deseos y creencias que resultan de la percepción del entorno y que son empaquetados en estructuras de datos, las cuales representan o bien nuevas creencias o bien nuevos deseos. Los agentes responden a estos cambios seleccionando planes de la librería de planes para cada uno de los cambios e instanciando estos planes como intenciones. Las intenciones por tanto comprenden planes, los cuales a su vez comprenden acciones y objetivos concretos para ser llevados a cabo con la posibilidad de que se adopten nuevos planes.

Jo et al. (2004) proporcionaron una explicación más detallada de los 3 términos usados en el modelo de agentes BDI:

- **Creencias**: Las creencias de un agente corresponden a la información que se adquiere del entorno. Representan el conocimiento del mundo o el estado del mundo como el valor de una variable, una base de datos relacional o como expresiones simbólicas en lógica de predicados. Este conocimiento puede ser incompleto o incorrecto.

- **Deseos**: Los deseos representan los estados que al agente le gustaría alcanzar. Los deseos u objetivos forman un componente esencial del estado del sistema, representan el estado final deseado. Un deseo u objetivo puede ser un valor de una variable, la

estructura de un registro o una expresión simbólica en alguna lógica. La situación común es que los agentes no son capaces de alcanzar todos sus deseos, por tanto, deben centrarse en algún subconjunto de sus deseos y reservar recursos para alcanzarlos. Los deseos u objetivos elegidos que el agente intenta conseguir, son conocidos como intenciones.

- **Intenciones**: Las intenciones representan los deseos que el agente intenta llevar a cabo. Las intenciones son un conjunto de planes o procedimientos y pueden también ser un conjunto de hilos de ejecución de un proceso que busca los objetivos del sistema. Las intenciones obligan a utilizar el razonamiento interno del agente para seleccionar la acción a llevar a cabo. Los deseos y las intenciones se llevan a cabo por medio de la ejecución de una serie de acciones y la serie de acciones llevadas a cabo definen un plan.

Las creencias que puede almacenar un agente pueden ser de los siguientes tipos:

- **Dinámicas**: Corresponden a información que varia en el tiempo, por ejemplo: el balance actual de mi cuenta bancaria es de 300 euros.
- **Estáticas**: Son creencias que no modifican su valor durante el ciclo de vida de ejecución del agente, por ejemplo: el nombre de la persona con el número de cuenta 20381874713000303179 es Federico Castanedo.
- **Propias**: Corresponden a información obtenida por los datos de entrada del agente, por ejemplo: (Cajero-1) mi saldo disponible es de 3000 euros.
- **De otros agentes**: Son creencias que poseen sobre las creencias de otros agentes, por ejemplo: (Banco) el saldo del cajero-1 es de 3000 euros.

Por otro lado, Georgeff et al. (1999), realizaron la siguiente división interna de los trabajos basados en el modelo de agentes BDI:

1. Aquellos que usan modelos generales de razonamiento práctico basados en conceptos BDI.
2. Trabajos de modelos computacionales basados en la arquitectura IRMA (Intelligent Resource-Bounded Machine).
3. Los que están basados en el modelo computacional usado en los sistemas PRS (Georgeff & Lansky, 1987) (Ingrand et al., 1992), ampliamente utilizado en la práctica e implementado en soluciones como JACK (Busetta et al., 1999), JAM (Huber, 1999), Jadex (Pokahr et al., 2003) y Agentis (Kinny & Phillip, 2004).

Uno de los lenguajes de programación orientado a agentes BDI más conocido es AgentSpeak, propuesto originalmente por Rao (1996). AgentSpeak esta basado en los lenguajes lógicos de programación e inspirado por el sistema PRS (Georgeff & Lansky, 1987) (Ingrand et al., 1992), el sistema dMARS (D'Inverno et al., 2004) y la lógica BDI. Su objetivo es ser un punto intermedio entre la teoría BDI y los sistemas prácticos de agentes BDI como PRS (Georgeff & Lansky, 1987) (Ingrand et al., 1992). Al ser un lenguaje basado en el modelo BDI, las construcciones del lenguaje están basadas en Beliefs, Goals y Plans. La arquitectura

de un agente en AgentSpeak tiene 4 componentes: (1) una base de creencias, (2) una librería de planes, (3) un conjunto de eventos y (4) un conjunto de intenciones. En AgentSpeak los agentes reaccionan a los eventos ejecutando planes y los eventos se producen como consecuencia de cambios en las creencias del agente o en los objetivos del mismo. El interprete de AgentSpeak más conocido es Jason (JASON, [Online 03/2010]), que está implementado utilizando el lenguaje de programación Java bajo licencia GPL.

El sistema dMARS (D'Inverno et al., 2004) es un sucesor del sistema PRS basado en el modelo de agentes BDI. Es el sistema basado en agentes que más se ha utilizado desde un punto de vista práctico. El sistema parte de la especificación formal hecha por Mark d'Inverno y representa el modelo operacional del sistema mediante planes. Al igual que Jadex también es un sistema basado en eventos. Las creencias son literales de la lógica de primer orden, por tanto una creencia se expresa como una fórmula donde el número de variables es cero. El sistema dMARS distingue solo entre dos tipos de objetivos: achieve y query. Las intenciones son secuencias ordenadas de planes que se han activado y se crean como respuesta a un evento externo. Finalmente, un agente dMARS consta de los siguientes componentes:

- Una biblioteca de planes.
- Una función de selección de las intenciones.
- Una función de selección de los planes.
- Una función de selección de los eventos.
- Una función de selección de las sustituciones.
- Una función para determinar la siguiente rama de un plan.

Aunque el modelo BDI ha sido ampliamente estudiado, cuestiones como cuanto se debe comprometer un agente con sus intenciones, continúan siendo un área de investigación hoy en día. El dilema, es esencialmente, el problema de balancear el comportamiento pro-activo (dirigido por objetivos) y el reactivo (dirigido por eventos). Debe existir un balance entre el nivel de compromiso de un agente y su reconsideración de intenciones (Kinny, 1991). Es obvio que un agente a veces debe descartar intenciones debido a las siguientes razones: (1) el agente llega a la conclusión que la intención no va a ser conseguida, (2) la intención ya ha sido conseguida o (3) la razón (la precondición) para alcanzar la intención ya no está presente. En este contexto, Wooldridge (2000) propuso tres niveles de compromisos para los agentes: *blind commitment*, *single-minded commitment* y *open-minded commitment*. Para Wooldridge (2000), el nivel de compromiso depende del último proceso de deliberación llevado a cabo; por lo tanto, es importante que un agente reconsidere sus intenciones cada cierto tiempo, no obstante, esto conlleva un coste computacional asociado.

Los agentes autónomos cooperan con el objetivo de mejorar su información y esta cooperación se lleva a cabo por medio del intercambio de mensajes. En los sistemas multiagente de tipo BDI, la cooperación se lleva a cabo por medio de asumir las creencias (beliefs) e intenciones (intentions) de unos agentes en otros. Por ejemplo, en las redes de sensores visuales la percepción del agente en un determinado momento (sus creencias) debe ser compartida, haciendo uso del intercambio de mensajes, con las de los otros agentes para, de esta forma, obtener un razonamiento distribuido.

4.5. Trabajos relacionados

Existen numerosos trabajos publicados en congresos y revistas científicas relacionados con los sistemas multi-agente y en menor medida con su aplicación en las redes de sensores visuales, lo que demuestra el reciente interés en esta línea de investigación.

Uno de los organismos que más ha utilizado el modelo de agentes BDI para simulaciones militares es el Australia's Defence Science and Technology Organisation (DSTO) (Murray et al., 1995) (Lucas & Goss, 1999). También se han realizado trabajos para establecer la utilización del modelo BDI en entornos de tiempo real (Urlings et al., 2006) y para usar el modelo BDI en la detección de intrusos en el campo de la seguridad informática (Shajari & Ghorbani, 2004). El proyecto RETSINA (Sycara & Pannu, 1998), es una infraestructura multi-agente basada en el modelo BDI, en la cual los agentes se coordinan para obtener información y solucionar problemas al usuario. Cada agente en RETSINA es un agente BDI que integra planificación, scheduling y coordinación con otros agentes.

El modelo de agentes BDI también es interesante porque se ha dedicado mucho esfuerzo en desarrollar los formalismos asociados. En concreto, Rao y Georgeff han desarrollado una serie de lógicas BDI (Rao, 1993) (Rao & Georgeff, 1991) (Rao & Georgeff, 1992) (Rao & Georgeff, 1993) (Rao et al., 1992) (Rao et al., 1991) que se utilizan para definir las propiedades de los agentes BDI racionales.

Debido a que existen aspectos del comportamiento y razonamiento humano que no son capturados en el modelo BDI, Norling (2004) propuso una extensión a dicho modelo para tenerlos en consideración. En su trabajo, se indica que la noción de capturar el razonamiento humano utilizando deseos, creencias e intenciones, como lo propone el modelo BDI, consiste en una abstracción de alto nivel muy extrema del razonamiento humano. La autora, propone una extensión del modelo BDI utilizando un enfoque basado también en Folk Psychology, pero distinto, en el cual se tengan en cuenta otros aspectos como las emociones. En el trabajo, se utiliza el framework JACK (Busetta et al., 1999) para implementar el modelo *recognition-primed decision* (RPD) propuesto por Klein.

En lo que respecta a los sistemas multi-agente utilizados para realizar tareas de fusión de datos, el trabajo de Berge-Cherfaoui & Vachon (1991) fue uno de los primeros en publicarse. Los autores proponen un enfoque multi-agente basado en un sistema blackboard, uno de los primeros modelos de comunicación multi-agente. El sistema blackboard es menos flexible que los modelos actuales de comunicación basados en FIPA ACL (FIPA, 2002). Un sistema blackboard consiste en un sistema donde se almacenan datos de forma distribuida, los diferentes nodos del sistema tiene acceso a la pizarra para leer y escribir datos que son el resultado de su percepción y/o acciones. El problema de este tipo de sistemas es que el conocimiento está centralizado en un único sitio, la pizarra. Esto hace al sistema vulnerable a fallar de forma completa si se produce una caída del nodo que almacena la pizarra.

Detmold et al. (2006) también han propuesto el uso de un middleware basado en los sistemas blackboard para aplicarlo en redes de video-vigilancia, en su modelo los resultados de procesar la información de entrada se publican en el sistema blackboard compartido. El sistema blackboard actúa como un repositorio de información donde determinados cálculos se disparan en respuesta a los resultados públicados. El sistema propuesto tiene diferentes niveles de interacción; por un lado, el nivel de análisis individual de la escena proporciona información de la detección de objetos y análisis de la actividad. A continuación, el nivel de análisis de múltiples escenas realiza conclusiones de los objetos que han estado bajo

seguimiento. Finalmente, el nivel de razonamiento proporciona hipótesis de alto nivel para considerar el comportamiento anormal. Dentro del sistema, la información es propagada hacia arriba y compartida entre los niveles de la pizarra.

Por otro lado, el sistema MAVI, consiste en una arquitectura multi-agente para construir sistemas de visión integrados y fue presentado por Boissier & Demazeau (1994). En ella, se utilizan los sistemas multi-agente como una forma de integrar los sistemas de visión y gestionar su control. Demazeau et al. (1994) propusieron un lenguaje de interacción, basado en los protocolos de interacción de Speech Act Theory, para controlar módulos de visión. Su trabajo no está basado directamente en sistemas multi-agente, pero si en los protocolos de interacción de la inteligencia artificial distribuida que han sido la base para el posterior desarrollo de los sistemas multi-agente. Remagnino et al. (2001, 2004, 1998) presentaron una arquitectura de agentes flexible, donde los agentes son creados cuando se necesitan, con el objetivo de construir un sistema de vigilancia automático.

Un enfoque de agentes, basado en el modelo de información-subscripción, es utilizado en el proyecto Europeo MODEST (1998). En el proyecto MODEST, se propone extender el lenguaje semántico de FIPA (SL) para tener en cuenta la incertidumbre y los descriptores MPEG-7. Dentro del proyecto MODEST, se desarrolló Monitorix, un sistema multi-agente de vigilancia de tráfico con áreas no solapadas (Abreu et al., 2000). En este sistema, los agentes están divididos en cuatro capas: (1) la capa de los sensores, (2) la capa de descripción de objetivos, (3) la capa de asistente de aplicación y (4) la de asistente de usuario. Los agentes de Monitorix son conformes al estándar FIPA y utilizan el lenguaje de comunicación FIPA ACL (FIPA, 2002), sin embargo no son de tipo BDI.

Una arquitectura de agentes, donde un agente es creado por cada objeto detectado en la escena, fue presentada por Orwell et al. (1999). El objetivo del sistema es obtener una descripción de alto nivel de los eventos observados, a diferencia de la principal contribución del trabajo presentado en este documento, que consiste en fusionar la información de seguimiento de los objetos detectados por medio del sistema multi-agente. En su trabajo, cada cámara de vigilancia es controlada por un agente y el seguimiento se lleva a cabo en el plano del suelo.

Un sistema de coordinación, basado en el protocolo de la red de contratos (contract-net protocol), fue propuesto por Graf & Knoll (2000). Una red de contratos (Smith, 1980) es un mecanismo de coordinación, basado en el modelo de funcionamiento de los mercados, para realizar asignación de tareas entre agentes comunicativos. El mecanismo de coordinación consiste en que el agente gestor anuncie una oferta, se deje a los agentes pujar por ella y se asigne la oferta al agente que el gestor considere más oportuno, por tanto es un mecanismo de coordinación basado en subastas. En su trabajo, se distingue entre dos tipos de agentes: maestros y esclavos, los cuales están conectados por medio de una red de contratos. Uno de los inconvenientes de la red de contratos, es que antes de que los sub-problemas se puedan distribuir, es necesario realizar la descomposición de los mismos y estos problemas deben tener la granularidad adecuada. Además, la fase de reconocimiento por parte del agente que solicita colaboración no se describe de forma explícita dentro del protocolo y debe ser implementada de forma ad-hoc para cada problema.

Hongeng & Nevatia (2001) presentaron un sistema multi-agente, para reconocer eventos, utilizando descripciones temporales complejas. Su enfoque está basado en un método Bayesiano, donde cada evento está representado por una serie de acciones relacionadas por sus restricciones temporales.

Marchesotti et al. (2003) proponen el uso de un sistema dividido en 3 capas funcionales y basado en agentes para combinar los datos de los sensores visuales. Se trata de una propuesta similar a la realizada en este documento, sin embargo, con el solo abordaje de la fusión de los datos de dos cámaras. Además es un sistema no compatible con el estándar FIPA y basado en la comunicación de mensajes multicast en lugar de utilizar FIPA ACL (FIPA, 2002).

Zhou et al. (2004), presentan una arquitectura multi-agente que se basa en la optimización de funciones de utilidad[16] para garantizar que las tareas de visión se realicen bajo determinadas restricciones; se definen múltiples comportamientos posibles, entre los cuales, se elige uno basándose en el valor de la función de utilidad definida. Se argumenta que para modelar tareas de visión, los sistemas multi-agente son apropiados, debido a la gran flexibilidad inherente y se ajustan los tiempos de proceso de cada agente mediante la asignación de diferentes recursos a cada uno de ellos. A diferencia de esta propuesta, su trabajo está más enfocado hacia la gestión de recursos.

En el trabajo de Matsuyama (1999), se monitoriza el entorno, por medio de múltiples cámaras estacionarias o robots móviles, con el objetivo de construir de forma dinámica un modelo en 3D de la escena. En este mismo proyecto, Ukita & Matsuyama (2005) combinan el *active sensing*, que consiste en la capacidad de mover la cámara a un área de mayor incertidumbre o interés con la comunicación entre los múltiples agentes visuales que componen el sistema.

Recientemente, Aguilar-Ponce et al. (2007) han presentado un sistema multi-agente para realizar vigilancia visual de forma automática. El objetivo del trabajo, similar a esta propuesta, es realizar la detección y el seguimiento de las personas por medio de un sistema multi-agente; sin embargo, el protocolo de comunicación de agentes utilizado está basado en KQML y no en FIPA.

El trabajo de Patricio et al. (2007), cuyo objetivo principal es realizar seguimiento en interiores minimizando los cambios de contexto por medio de la comunicación de los agentes, está también basado en un sistema de agentes BDI.

Con el mismo objetivo de solucionar los problemas de cambio de contexto, Quaritsch et al. (2007) proponen el uso del paradigma orientado a agentes, en concreto agentes móviles [17]. En su trabajo, las diferentes tareas de procesado de las imágenes son llevadas a cabo por distintos agentes. Los agentes son los responsables de la ejecución de las tareas en la unidad de procesamiento y pueden crear nuevos agentes locales o remotos en otras unidades de procesamiento. Los agentes encargados de realizar el seguimiento, continúan su ejecución en otra cámara cuando el objetivo se va a perder del campo de visión de la cámara. Recientemente, Biswas et al. (2008) han propuesto también un enfoque de agentes móviles, donde la estación base despliega agentes móviles que migran de un nodo a otro y van realizando la fusión de datos en cada nodo.

Un sistema multi-agente jerárquico para realizar fusión de datos y gestión eficiente de múltiples sensores fue propuesto por Zhu et al. (2007). Los autores, propusieron el uso del mecanismo de subscripción/publicación para gestionar la información de los sensores.

Mastrogiovanni et al. (2007) presentaron un sistema distribuido para realizar las fusión de datos simbólica en un entorno de inteligencia ambiental. Su sistema es modelado como un conjunto de agentes dentro de un ecosistema. Se trata de un sistema en el que existen

[16] Función que representa de forma cuantitativa el valor de las acciones para el sistema.

[17] Entendiendo por agentes móviles, los que tienen capacidad de continuar su ejecución en otro nodo manteniendo el estado de la ejecución anterior.

dos tipos de agentes: agentes cognitivos y agentes de dispositivo. Los agentes dispositivo se encargan de obtener la información del entorno y los agentes cognitivos realizan la fusión simbólica de los datos. La fusión simbólica de datos es llevada a cabo por el mecanismo de "subsumption", el principal esquema de razonamiento para los lenguajes basados en lógica descriptiva (Brachman & Levesque, 1984). En su trabajo hacen un recorrido por los distintos niveles de fusión propuestos por el JDL y los enlazan con modelos lógicos de razonamiento.

El trabajo de Iwaki et al. (2008), consiste en un sistema multi cámara para la detección y seguimiento de personas, que consta de una red de agentes distribuidos, cada uno de los cuales tiene diferentes funcionalidades. La incertidumbre de las medidas proporcionadas por cada cámara se reduce por medio de un sistema de acumulación de evidencias que utiliza una arquitectura multi-agente distribuida jerárquica, similar a la que se propone en este trabajo. En cuanto a la técnica de acumulación de evidencia, siguen un enfoque basado en la distancia de Mahalanobis y una técnica ponderada de mínima varianza. La principal diferencia con este trabajo, es que se realiza el seguimiento de las caras de las personas en tres dimensiones y no el de las trayectorias en el plano del suelo, y además, no se utiliza un sistema multi-agente conforme al estándar FIPA.

Por otro lado, desde el punto de vista metodológico de la ingeniería del software, Pavón et al. (2007) han presentado el diseño de un sistema multi-agente para vigilancia, que a diferencia de este trabajo, se basa en la aplicación de una metodología de software al problema particular de la vigilancia, sin llegar a dar detalles concretos de implementación y resultados experimentales.

Además, los sistemas multi-agente también han sido aplicados en redes de sensores (no visuales) distribuidas, destacando principalmente el trabajo de Lesser et al. (2003), dentro del proyecto DVMT. En los entornos radar, Molina et al. (2003) propusieron un sistema multi-agente para la gestión cooperativa de una red de sensores radar, combinando el uso de sistemas multi-agente junto con la lógica difusa para gestionar la red de sensores. También utilizando la lógica difusa, Molina et al. (2004) propusieron un sistema multi-agente para solucionar de forma cooperativa el problema de la asignación de las tareas en los sensores. El sistema proporciona un razonamiento similar al que es llevado a cabo por los operadores humanos, permitiendo adaptarse a situaciones cambiantes.

Finalmente, en el trabajo de Cicirelli et al. (2007) se utiliza el paradigma multi-agente para aplicarlo en redes de sensores con el objetivo de obtener la configuración de los sensores mínima necesaria para cubrir una región.

4.6. Resumen crítico

En este capítulo, se han presentado los sistemas multi-agente. Siempre que se presenta una tecnología o paradigma nuevo surgen dos cuestiones obvias: (1) ¿que ventajas ofrece esta tecnología frente a las alternativas? y (2) ¿en que circunstancias resulta útil utilizarla?. En este apartado se pretende dar respuesta a estas dos preguntas.

La idea principal del trabajo que se presenta en este documento, es explotar las posibilidades que proporcionan los sistemas multi-agente para modelar las redes de sensores visuales y realizar una fusión de datos más flexible. A diferencia de propuestas anteriores, que utilizan los sistemas multi-agente como una técnica para gestionar los sensores, se propone involucrar a los agentes en el proceso de seguimiento y fusión. Para ello, se va a presentar una fusión activa, que se basa en utilizar la realimentación proporcionada por el resultado final del sistema en los agentes que se encargan de obtener la información visual. De esta forma, es posible corregir situaciones individuales basándose en la información colectiva que proporciona la realimentación. Un seguimiento colaborativo, a diferencia de un seguimiento estéreo, proporciona la ventaja del coste computacional. Esto es debido a que, el seguimiento estéreo es computacionalmente costoso, ya que requiere la concordancia a nivel de píxel.

Una de las ventajas del paradigma orientado a agentes y en concreto del modelo BDI, es que permite el desarrollo de sistemas complejos de una forma natural, porque utiliza, en la fase de modelado del sistema, conceptos similares al comportamiento de las personas. Existen algunos dominios en los cuales resulta especialmente útil el uso de los sistemas multi-agente; en concreto, si existen diferentes personas u organizaciones con diferentes objetivos (posiblemente contradictorios) e información propia, un sistema multi-agente es muy útil para modelar sus interacciones. Aunque cada organismo quiera modelar sus deseos propios con un sistema interno, es necesario establecer una interacción con el resto de organismos, de forma que cada uno de los distintos organismos necesita sus propios sistemas para reflejar sus capacidades y sus prioridades. Incluso en dominios en los cuales los agentes no tengan objetivos contradictorios, pero que se puedan dividir fácilmente en componentes (los cuales son procesados de forma individual por los agentes), un sistema multi-agente puede ser muy útil, ya que aporta una modularidad inherente. También en los sistemas distribuidos (de forma geográfica y/o lógica) como es el caso de las redes de sensores visuales, los sistemas multi-agente proporcionan varias ventajas, ya que permiten modelar de una forma natural la distribución de las tareas. Por otro lado, es posible obtener robustez, por medio de la redundancia, en los sistemas multi-agente de forma sencilla, simplemente replicando el número de agentes que realizan las mismas tareas, debido a que si el control y las responsabilidades están adecuadamente compartidas entre los diferentes agentes, el sistema puede tolerar los fallos de uno o más agentes, al obtener la información de los agentes que no fallan.

Otro de los beneficios de un sistema multi-agente es su escalabilidad. Teniendo en cuenta de que se trata de sistemas que son inherentemente modulares, es más fácil añadir nuevos agentes en un sistema multi-agente que añadir nuevas funcionalidades en un sistema monolítico; por lo tanto aquellos sistemas en que las funcionalidades y los parámetros pueden variar con el tiempo, se pueden beneficiar de esta característica, ya que se pueden crear nuevos agentes en función de las necesidades.

Desde del punto de vista del programador o del desarrollador del sistema, la modularidad inherente de los sistemas multi-agente le proporciona una mayor facilidad de desarrollo ya que los programadores pueden identificar sub-tareas y asignar el control de esas sub-tareas

a determinados agentes.

Mientras que las razones anteriores consisten en un punto de vista general de los sistemas multi-agente, existen además, argumentos a favor de su uso en las redes de sensores visuales. Un solo sensor visual únicamente puede monitorizar el entorno desde un punto de vista limitado, sin embargo el uso de un sistema multi-cámara implementado con sistemas multi-agente, proporciona una visión global y desde distintos puntos de vista del entorno. La ventaja que proporcionan los sistemas multi-agente frente a los sistemas multi-cámara tradicionales es que la información del entorno, obtenida por cada uno de los agentes de forma individual, puede ser fácilmente compartida con otros agentes por medio del intercambio de mensajes y de los protocolos propios de los sistemas multi-agente.

II

Propuesta, experimentación y conclusiones

5

Cooperative sensor agents (CSA)

Data fusion is deceptively simple in concept but enormously complex in implementation.
US Department of Defense,1990.

5.1. Introducción

En esta parte del documento, se presenta el sistema desarrollado para cumplir con los objetivos descritos en el capítulo 1, los cuales consisten en una arquitectura software capaz de proporcionar soporte a la comunicación de los nodos y a los algoritmos que se ejecutan, lo que puede resumirse en:

- Disponer de los algoritmos necesarios para soportar la captura de la información visual.
- Realizar la transformación de la información visual a datos estructurados (información) para poder ser intercambiados entre los distintos nodos de la red (agentes) y realizar una adecuada fusión de datos.
- Proporcionar funciones para llevar a cabo la fusión de los datos locales de cada uno de los sensores.
- Proporcionar a los agentes que forman el sistema la posibilidad de razonar acerca de la calidad de la información local basándose en la información global.

En definitiva, se utiliza una arquitectura multi-agente basada en el modelo BDI para construir redes de sensores visuales que sean más robustas, flexibles y adaptables que las desarrolladas con una arquitectura clásica. El principal objetivo del sistema desarrollado es realizar la fusión de la información de seguimiento proporcionada por cada uno de los sensores; no obstante, el mismo esquema de arquitectura propuesto puede utilizarse con diferentes objetivos como: contar personas, clasificar objetos, reconocimiento de actividades, etc. Además, el sistema multi-agente propuesto proporciona de forma inherente la gestión de los sensores (*sensor management*), es decir, por medio de los deseos e intenciones de los agentes se establece también la forma de gestionar la información con la ventaja de que no es necesario desarrollar un sub-sistema de gestión de los sensores independiente.

El resto del capítulo está organizado de la siguiente forma: en primer lugar, en el apartado 5.2, se describe el modelo de agentes BDI aplicado en las redes de sensores visuales. A continuación, en el apartado 5.3 se describe formalmente la arquitectura del sistema y la forma de interacción de los agentes. Posteriormente, en los apartados 5.4 y 5.5, se detallan los procesos internos de los agentes sensores y los agentes de fusión. A continuación, en el apartado 5.6, se realiza una breve descripción de otros posibles agentes que pueden

formar parte del sistema. Finalmente, en el apartado 5.7, se presenta una descripción de la implementación concreta del sistema que se ha realizado.

5.2. Modelo de agentes BDI aplicado en redes de sensores visuales: CSA

Como ya se ha indicado en el apartado 4.4, los agentes racionales tienen una representación explícita de su entorno[18] y de los objetivos que intentan alcanzar. La racionalidad implica que el agente siempre lleva a cabo aquellas acciones (dependiendo de su propio conocimiento y del entorno) que son más prometedoras para alcanzar sus objetivos. Normalmente, no se conocen los efectos de todas las acciones a priori, por tanto, el agente debe deliberar acerca de las opciones disponibles para elegir la más apropiada. En los sistemas distribuidos complejos, es imposible conocer, en el momento de diseño del sistema, todas las posibles interacciones que se van a producir entre los componentes, debido a que la comunicación ocurre en instantes impredecibles y por razones impredecibles. Desde un punto de vista de diseño, es mejor dejar que los agentes decidan en tiempo de ejecución, que información comunicar y cuando comunicarla, en lugar de modelar de forma explícita todas las posibles situaciones. Además, los entornos complejos poseen cierta incertidumbre que debe ser considerada si se quiere que el sistema se pueda aplicar a algo más que a un problema trivial. Por tanto, en estos entornos inciertos y dinámicos, los agentes deben ser autónomos, ya que un agente no puede conocer con exactitud el efecto de sus acciones o de las acciones de otros agentes. Esto conlleva que un agente deba ser diseñado con la flexibilidad necesaria que le permita responder a entornos dinámicos, evaluándolos y respondiendo a ellos de una forma adecuada. En este sentido, la autonomía del agente permite la ejecución de comportamientos que no se basan en reglas definidas de forma previa.
Por regla general, se considera que un entorno es dinámico cuando se producen las siguientes circunstancias:

- En cualquier instante de tiempo, puede haber múltiples formas distintas en las cuales el entorno puede evolucionar. Esto se conoce formalmente como un entorno no determinista.
- En cualquier instante de tiempo, existen múltiples acciones o procedimientos que el sistema puede ejecutar. Es decir, el sistema propiamente es no determinista.
- En cualquier instante de tiempo, pueden haber varios objetivos distintos que el sistema debe llevar a cabo, los cuales no pueden ser ejecutados todos a la vez
- Las acciones o procedimientos, que mejor pueden conseguir los objetivos, dependen del estado del entorno (contexto) y son independientes del estado interno del sistema.
- El entorno solo se puede monitorizar de forma local y además una sola acción no es suficiente para determinar el estado completo del entorno.
- El ratio por el cual las acciones y los cálculos se llevan a cabo está dentro de los límites razonables del ratio al que el sistema evoluciona.

[18] El entorno de un agente también es denominado modelo del mundo y consiste en la información percibida por los sensores del agente y por los mensajes recibidos por otros agentes.

Las redes de sensores visuales suelen monitorizar entornos dinámicos que cumplen las características descritas anteriormente y los sistemas de agentes deben ser capaces de gestionarlos.

Como se ha comentado en el apartado 4.4, en un modelo BDI, cada agente tiene sus propias creencias (beliefs), deseos (desires) e intenciones (intentions). La ventaja de utilizar actitudes mentales en el diseño y realización de los sistemas multi-agente conlleva que se pueda modelar el sistema de una forma más natural y que se pueda proporcionar un alto nivel de abstracción.

En el caso de las redes de sensores visuales, estas se componen de diversos sensores para obtener la información de cada objetivo en el entorno. Estos sistemas deben resolver principalmente dos tipos de problemas (Manyika & Durrant-Whyte, 1995):

1. **Fusión de datos**: Consiste en la combinación de datos de diferentes fuentes de una forma óptima (Waltz & Llinas, 1990). La información fusionada, representa una entidad con mayor detalle y menor incertidumbre de lo que sería posible obtener por medio de cada una de las fuentes de forma individual. La combinación adicional de datos independientes y redundantes puede proporcionar una mejora en los resultados si se lleva a cabo de forma adecuada.

2. **Gestión multi-sensor**: Asume que el problema anterior está resuelto y se encarga de optimizar la gestión global del sistema completo por medio de la aplicación de operaciones individuales en cada sensor.

En lo referente a la cooperación o componente social de los agentes, dentro de una red de sensores visuales, los agentes autónomos pueden necesitar cooperar con otros agentes por las siguientes razones:

- Para obtener un mejor rendimiento o una mayor precisión en una tarea de vigilancia específica. De esta forma, incorporan información complementaria que se combina por medio de técnicas de fusión de datos. Por ejemplo, en el caso del seguimiento de personas, los agentes pueden cooperar para intercambiar información acerca de la posición de la persona en cada instante, con el objetivo de realizar una estimación más precisa.

- Para usar capacidades de otros agentes con el objetivo de ampliar la cobertura y llevar a cabo tareas que por si solos no pueden realizar. La cooperación puede proporcionar una visión global de la escena que se está monitorizando, más amplia de la que pueden proporcionar cada uno de los sensores por separado.

En particular, las redes de sensores visuales presentan una serie de características, que las hacen apropiadas para poder ser modeladas con los sistemas multi-agente, como son:

1. Se trata de sistemas, que están por definición, geográficamente distribuidos, lo cual se puede plasmar en una distribución lógica de los procesos que ejecutan la información obtenida de cada sensor. Los procesos que se deben ejecutar en cada sensor, pueden formar parte del ciclo de ejecución de un agente, facilitando de esta forma la gestión y la comunicación con otros nodos del sistema.

2. Cada sensor visual puede tener unas características propias, tanto inherentes, como de la situación física en la que se encuentra, lo cual se puede modelar de una forma natural utilizando el concepto de las capacidades de los agentes.

3. En una red de sensores visuales, la información debe ser comunicada entre los distintos sensores y procesos software que forman parte del sistema. Los protocolos de comunicación estándar de los sistemas multi-agente consisten en una forma definida y estructurada de llevar a cabo el intercambio de información de los sensores.

4. El uso de una arquitectura abierta y fácilmente escalable, en la cual, es muy sencillo incorporar nuevos agentes en el sistema con los mismos o diferentes objetivos. Esto facilita la labor de aumentar la capacidad de la red de sensores visuales.

5. Los sistemas multi-agente, al estar basados en estándares bien definidos, proporcionan al sistema la capacidad de interactuar más fácilmente con otros sistemas.

6. La posibilidad de mejorar el proceso visual por medio del razonamiento y cooperación de los agentes involucrados y de la información procesada por sus sensores.

A la hora de realizar una labor cooperativa y utilizar las capacidades de otros agentes, el concepto de *coalición* aparece cuando un grupo de agentes autónomos decide trabajar juntos para alcanzar un objetivo común (Wooldridge, 2000). Dentro de una red de sensores visuales, un agente puede necesitar cooperar con otros agentes para obtener un rendimiento mejor y más preciso. El proceso de hacer una coalición se conoce como *formación de coalición* (Coalition Formation), y ha sido ampliamente estudiado (Kahan & Rapoport., 1984) (Raiffa, 1984) (Ketchpel, 1994) (Shehory & Kraus, 1995). Sin embargo, prácticamente ninguno de los trabajos de redes de sensores visuales modeladas con sistemas multi-agente consideran la posibilidad de formar grupos temporales de 'sensores' para cooperar entre ellos en tareas específicas.

La formación de la coalición se puede llevar a cabo dinámicamente en tiempo de ejecución, estableciendo los agentes que van a participar en la coalición, o por otro lado, se puede definir de forma previa en la etapa de desarrollo del sistema, dando lugar al determinismo de los agentes que forman parte de la coalición. En el primer caso, nos referimos a ellas como *coaliciones dinámicas* y en el segundo caso como *coaliciones estáticas*. El modelo de coalición de Wooldridge (2000) consta de las siguientes 4 fases:

1. **Reconocimiento**. El proceso de cooperación empieza cuando un agente dentro del sistema multi-agente tiene que llevar a cabo un objetivo y reconoce el potencial para realizar una acción cooperativa. Es decir, una solución se puede llevar a cabo más rápidamente o de forma más precisa si se realiza de forma colaborativa. El agente además debe tener la creencia de que existe algún agente que puede ayudarle a llevar a cabo el objetivo.

2. **Formación del equipo**. En esta fase, el agente solicita crear el equipo. En principio, un agente no puede garantizar que el equipo se vaya a crear con éxito, solo puede intentarlo. El hecho de que una agente puede rechazar formar parte de una coalición, es consistente y coherente con la autonomía de los propios agentes, los cuales deciden formar parte del equipo o no.

3. **Plan de formación del equipo**. Los agentes negocian un plan conjunto que consideran que pueda satisfacer el objetivo deseado. Un grupo de agentes solo forman un colectivo si consideran que se puede alcanzar el objetivo.

4. **Acción del equipo**. El plan negociado se lleva a cabo por el equipo de agentes de la coalición de una forma conjunta. Una coalición finaliza cuando se obtiene el objetivo común. Sin embargo, debido a la autonomía de los agentes, estos pueden decidir abandonar la coalición (por ejemplo, para atender a otros objetivos más prioritarios).

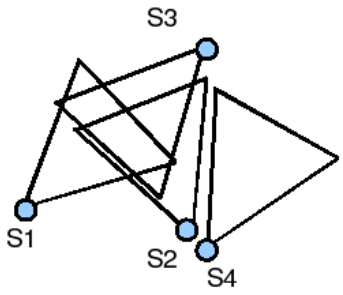

Figura 5.1: Ejemplo abstracto de una red de sensores visuales desplegada con áreas solapadas. Los sensores S1,S2 y S3 pueden formar una coalición para monitorizar el área solapada.

Cuando se inicia el proceso de la coalición, la cooperación existe solo en el estado mental del agente que lo inicia. Un corolario de que los agentes son autónomos es que la cooperación puede fallar. La cooperación que tiene lugar cuando la coalición, que es iniciada por el agente fusión, está completamente formada, implica la compartición de los datos entre los agentes involucrados y la resolución de conflictos. Una coalición se establece en un instante particular del tiempo para conseguir un objetivo común y cuando este objetivo se ha conseguido, la coalición se termina. La figura 5.1 muestra, de forma abstracta, un sistema de redes de sensores visuales desplegado con áreas solapadas, que pueden ser explotadas por una coalición para obtener resultados más precisos y garantizar una monitorización coherente en el área global. Los algoritmos de seguimiento implementados en los agentes sensores deben considerar errores en la detección del movimiento e interacciones complejas de objetos (unión, oclusiones, fragmentaciones, etc.). Los agentes responsables de la fusión de los datos se encargan de combinar la información inferida por cada uno de los agentes sensores para maximizar la calidad de la información final del área vigilada. En el esquema de la figura 5.1, una coalición se puede crear por el agente responsable de la fusión de los datos cuando predice que un objeto va a entrar en el área común a los sensores S1, S2 y S3. El agente encargado de la fusión de los datos, combina de forma inteligente los datos de los agentes sensores que forman parte de la coalición.

Usar un modelo de agentes en un entorno de redes de sensores visuales o vídeo vigilancia proporciona varias ventajas. En primer lugar, el bajo acoplamiento del modelo multi-agente proporciona una mayor flexibilidad para la comunicación de los datos entre los procesos. En segundo lugar, los sistemas multi-agente establecen un marco para modelar de forma natural, la forma de razonar con la información percibida por cada agente frente a la información global fusionada. Además, la habilidad de asignar responsabilidades a cada agente es ideal para resolver tareas complejas en un sistema de vigilancia. Estas tareas complejas involucran el uso de mecanismos como coordinación, configuración dinámica y cooperación, los cuales han sido ampliamente estudiados por la comunidad investigadora de sistemas multi-agente.

Según Durfee (1988), si los agentes deben coordinar sus actividades, deben conocer no solo la existencia de los otros agentes y su conocimiento interno, sino además, deben tener el conocimiento que otros agentes tienen de ellos y el conocimiento que tienen los otros agentes de los demás. Este nivel de razonamiento es muy difícil de conseguir, en las redes de sensores visuales, con una comunicación a bajo nivel sin utilizar una arquitectura multi-agente, por lo que se hace necesario el uso del paradigma orientado a agentes.

El sistema de agentes que se ha desarrollado para proporcionar una mayor flexibilidad, capacidad de razonamiento y autonomía en las redes de sensores visuales y que se ha aplicado para realizar un seguimiento distribuido, se ha llamado: **Cooperative Sensor Agents (CSA)** y consiste en una arquitectura multi-agente BDI donde los agentes son homogéneos y comunicativos. En la figura 5.2 se ilustra una visión de alto nivel de la arquitectura multi-agente, donde se observan los siguientes agentes:

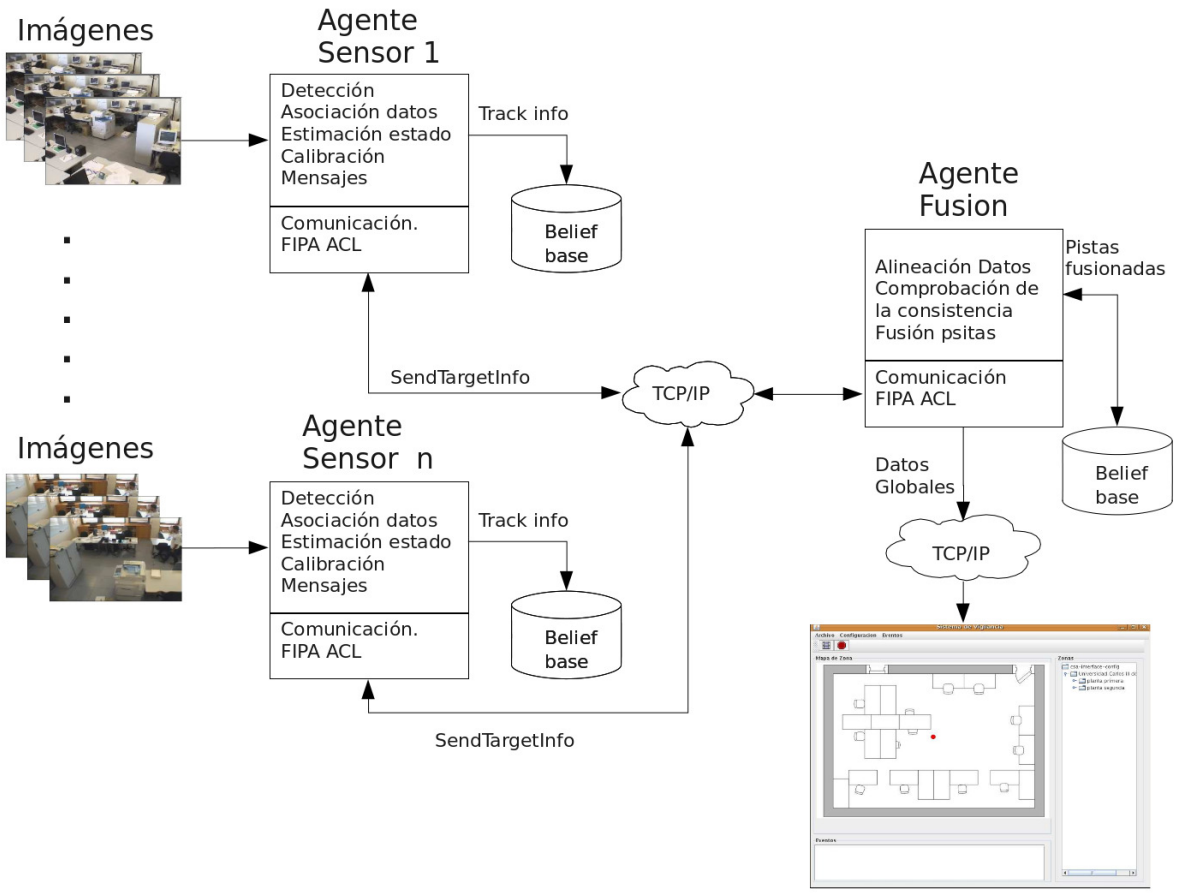

Figura 5.2: Esquema de alto nivel de la arquitectura multi-agente.

- **Agente Sensor:** Realizan el seguimiento de las trayectorias de todos los objetivos detectados dentro de su campo de visión, además envían la información de los objetivos detectados al agente de fusión correspondiente y se coordinan con otros agentes sensores para mejorar la calidad de la información. En el sistema CSA puede haber de 1 a N agentes de este tipo.

- **Agente Fusión:** Son los agentes fusión, que integran la información enviada por los agentes sensores asociados y analizan la situación para gestionar los recursos y coordinar a los agentes sensores. Estos agentes tienen la visión global del entorno que está siendo monitorizado por los agentes sensores. Son los responsables de crear las coaliciones dinámicas de los agentes sensores usando la información de contexto y la predicción de situaciones que requieren de un proceso de fusión cooperativo. Una de las ventajas de utilizar un agente de fusión de datos, es que la arquitectura es más flexible frente a las arquitecturas clásicas que son más rígidas ya que utilizando un protocolo de negociación la arquitectura es reconfigurable en función de las necesidades. En el sistema multi-agente CSA puede haber de 1 a M agentes de este tipo, siendo $M < N$.

Una de las ventajas de una arquitectura multi-agente que se encuentre estandarizada bajo FIPA es que es muy sencillo su inter-operabilidad con nuevos agentes que se deseen integrar en el sistema. Para ello, solo es necesario establecer las direcciones de la plataforma donde se ejecutaran estos agentes y los mensajes que pueden enviar/recibir entre ellos. Los siguientes agentes se presentan como posibilidades para extender el sistema:

- **Agente de planificación:** Este agente tiene una visión global del entorno y realiza inferencias de alto nivel sobre la situación.
- **Agente de contexto:** Este tipo de agente proporciona información dependiente del contexto e indica la distancia semántica entre los diferentes agentes sensores. Por distancia semántica se entiende la distancia de la información proporcionada por los sensores, independientemente de la distancia física de los sensores que están monitorizando la misma escena y posiblemente los mismos objetivos. Es decir, dos sensores físicamente alejados pueden tener una distancia semántica pequeña pues hacen referencia a la misma información. El agente de contexto, almacena la información acerca de los objetos estáticos que pueden provocar oclusiones parciales de los objetivos bajo seguimiento, pero además almacena información dinámica de la escena, por ejemplo, oclusiones debidas a la ocultación de objetivos por un camión que aparque, ocultando la escena.

5.3. Descripción formal de la arquitectura multi-agente CSA

CSA es un framework de agentes BDI autónomos, homogéneos y comunicativos, que operan en un entorno de redes de sensores visuales; en este apartado se realiza una descripción formal de su arquitectura.

La comunicación es un aspecto fundamental en los sistemas sociales y también en los sistemas distribuidos y por tanto en los sistemas multi-agente. La comunicación dentro de un sistema distribuido o de un sistema multi-agente se puede ver como el acto de transformar el conocimiento del estado local de los nodos en el conocimiento del sistema en su conjunto.

En el ámbito del sistema CSA, cada agente es capaz de cooperar con otros agentes para conseguir los objetivos comunes que se establecen en la fase de diseño, gracias a la comunicación. Los agentes pueden intercambiar información libremente para mejorar el proceso de seguimiento; sin embargo, la comunicación entre los agentes también introduce varios retos como: qué información comunicar, cuando comunicar la información y a quien se debe comunicar.

Los siguientes requisitos se consideran necesarios para el correcto funcionamiento de la arquitectura multi-agente CSA:

- Decidir de forma autónoma, es decir, por los procesos internos de cada uno de los agentes que información se envía a los otros agentes.
- Realizar un tratamiento asíncrono de los eventos con la información que reciben. Esto debe permitir que cada nodo sea capaz de realizar otras tareas de forma concurrente con la gestión de los eventos recibidos.
- Cada nodo debe ser capaz de llevar a cabo de forma autónoma la corrección de su propio estado, basándose en la información global recibida, y además, debe ser capaz de conseguir de forma activa la máxima coherencia entre la información obtenida (local) y la información global (fusionada).
- Mejorar la calidad de la información del área monitorizada por medio de las tareas cooperativas.
- Explotar la redundancia visual que proporciona la configuración espacial de los sensores visuales y de esta forma, obtener coherencia en la fusión de datos y detectar inconsistencias entre la información de cada nodo.

Los principales problemas que hay que resolver, relativos a la información transmitida y utilizada, para cumplir con los requisitos previos son:

1. Cual es la información que se debe comunicar entre los agentes. En el dominio de aplicación de las redes de sensores visuales, los sensores generan una gran cantidad de información que depende de forma directa de la resolución de las imágenes obtenidas. Dependiendo del tipo de aplicación deseada es conveniente comunicar por la red solo la información realmente necesaria.
2. En que momento se transmite la información. En función del tipo de aplicación la información transmitida puede dejar de tener valor y ser obsoleta para el proceso de fusión. Además puede ser necesario disponer del conocimiento del instante de tiempo en el que se obtiene la información.
3. Como es compartida y fusionada la información. Es necesario establecer protocolos de intercambio de los datos entre los diferentes agentes del sistema los cuales deben contener la información necesaria para realizar adecuadamente el proceso de fusión.

En la figura 5.2 se ha mostrado una visión de los agentes que pueden ejecutarse dentro del sistema multi-agente CSA. En el sistema CSA pueden ejecutarse varios agentes de tipo sensor, los cuales realizan el procesamiento de la información local obtenida del sensor visual y envían los datos a los agentes fusión. Tanto el número de agentes de fusión, como el número de agentes sensores es variable y depende de las necesidades particulares. Sin embargo, debe existir al menos un agente sensor para cada uno de los sensores visuales y al menos un agente fusión en el sistema. Sobre la configuración física de los sensores visuales, en el sistema CSA, se considera que las cámaras están colocadas de forma que el campo de visión de las mismas se encuentra parcialmente solapado (ver figura 5.3).

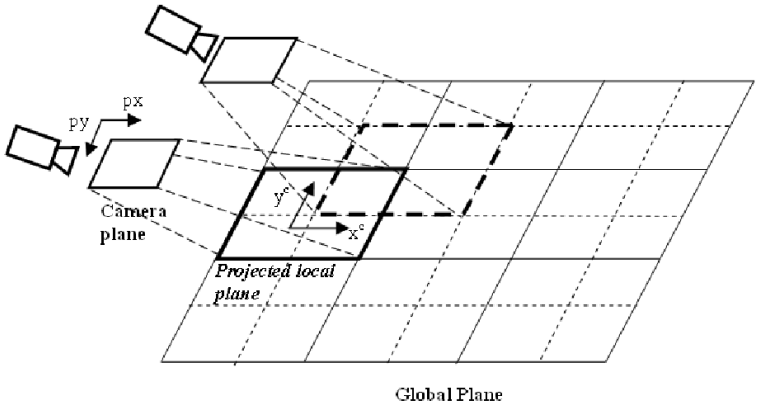

Figura 5.3: Ejemplo de despliegue de un sistema multi-cámara con áreas solapadas.

El principal objetivo de una red de sensores visuales, es monitorizar una gran área geográfica y proporcionar información precisa de lo que está ocurriendo en ella. Para poder llevar a cabo la monitorización de una gran área, un solo sensor visual no es suficiente; por tanto, es necesario disponer de una red de sensores visuales y además, es deseable proporcionar un cierto grado de redundancia, de forma que se puedan solventar errores de obtención de medidas y errores de funcionamiento.

Por otro lado, el modelo de razonamiento práctico sobre el cual está basado el modelo BDI, involucra dos procesos importantes:

1. Decidir los objetivos que se van a alcanzar. Es decir, hay que establecer los objetivos que cada uno de los agentes intenta conseguir en función de sus creencias y llevando a cabo sus intenciones por medio de la ejecución de planes concretos.

2. Como alcanzar los objetivos a los que el agente se ha comprometido. Para esto, es necesario establecer los planes concretos que un agente va a ejecutar en un instante determinado en función de sus creencias y de sus objetivos.

En el sistema propuesto, las intenciones dejan de tener sentido cuando se alcanzan los objetivos o bien cuando evoluciona el entorno y la intención deja de tener sentido, debido a que no se cumplen las precondiciones (creencias) para llevarla a cabo. Por ejemplo, si un objetivo que está bajo seguimiento abandona el campo de visión de un sensor visual, la intención de seguimiento deja de tener sentido y es descartada. Para poder actuar de forma racional, el modelo BDI representa internamente la situación y el estado mental de cada agente en forma de Creencias (Beliefs), Deseos (Desires) e Intenciones (Intentions). El estado de un agente en cualquier instante es un tripleta (B, D, I), donde $B \subseteq$ *Creencias*, $D \subseteq$ *Deseos* y $I \subseteq$ *Intenciones*.

En la teoría de agentes BDI, cualquier decisión llevada a cabo por un agente, viene definida por las motivaciones propias del modelo del agente y por las motivaciones de los otros agentes. Las motivaciones o creencias de los otros agentes se reciben por medio del paso de mensajes. El termino creencia o belief normalmente tiene dos significados distintos en el campo de la inteligencia artificial: (1) en robótica y visión artificial, una creencia es normalmente una distribución de probabilidad; (2) en inteligencia artificial lógica, una creencia

es normalmente una proposición lógica. En el contexto del modelo de agentes BDI, una creencia es un conocimiento específico del estado del mundo, que se almacena internamente dentro de la base de creencias del agente y que el agente considera como un hecho cierto. Las *creencias* de un agente representan el conocimiento de sus propias capacidades, el conocimiento de las capacidades de los otros agentes con los que puede colaborar y el conocimiento de la información del entorno obtenida por el sensor visual.

Sea n igual al número total de agentes sensores en el sistema multi-agente, S el conjunto de agentes sensores $S = \{S_1, S_2, \dots, S_n\}$ y S_i cualquier agente sensor, se pueden representar los *creencias* de los agentes sensores como:

- Siendo, $Env(t)$ el entorno que percibe un agente en el instante t, todos los agentes sensores almacenan la creencia de la información percibida en su entorno (por el sensor visual) en el instante de tiempo t. Definido formalmente como: $\forall i \in S \cdot (Bel\ S_i\ Env(t))$.
- Sea C_i el conjunto de capacidades del agente S_i, todos los agentes sensores tienen conocimiento de sus propias capacidades. Definido de manera formal como: $\forall i \in S \cdot (Bel\ S_i\ C_i)$.
- Sea Θ_i el vecindario[19] del agente sensor S_i, donde $(\Theta_i \subseteq S) \wedge (\Theta_i \neq \emptyset)$, todos los agentes sensores S_i, conocen las capacidades $(Bel\ S_j\ C_j)$ de todos los agentes de su vecindario $(\forall j \in \Theta_i)$. Definido formalmente como: $\forall i \in S \cdot (Bel\ S_i\ \forall j \in \Theta_i \cdot (Bel\ S_j\ C_j)$.

Por otro lado, supongamos que el número total de agentes de fusión es igual a k y F es el conjunto de agentes de fusión $F = \{F_1, F_2, \dots, F_k\}$, se pueden representar las *creencias* de cada uno de los agentes fusión como:

- Sea Δ_i el subgrupo de agentes sensores que están coordinados por el agente fusión F_i, donde $(\Delta_i \subseteq S) \wedge (\Delta_i \neq \emptyset)$, entonces, todos los agentes de fusión deben conocer el entorno de los agentes sensores que coordinan. Definido formalmente como: $\forall i \in F \cdot (Bel\ F_i\ \forall j \in \Delta i \cdot (Bel\ S_j\ Env(t))$.
- Todos los agentes fusión F_i, conocen las capacidades de los agentes sensores que coordinan. Definido formalmente como: $\forall i \in F \cdot (Bel\ F_i\ \forall j \in \Delta i \cdot (Bel\ S_j\ C_j))$.

Cada agente i, tiene una serie de k capacidades $C = \{C_1, C_2, ..., C_k\}$. Se entiende por capacidad de un agente la posibilidad de ejecutar planes determinados en función de las intenciones del mismo. En las redes de sensores visuales, estas capacidades pueden ser: la capacidad de seguimiento, la capacidad de reconocer eventos, la capacidad de grabación, la capacidad de calibración, etc. Sea O, el conjunto de objetivos detectados en el entorno de un agente en un instante de tiempo t: $O = \{O_1^t, O_2^t, ..., O_j^t\}$ y considerando A_i, como cualquier agente dentro del sistema multi-agente independientemente del tipo que sea, se pueden dar las siguientes definiciones:

Definición 1. Apply. *Apply* (A_i, C_k, O_j^t). Es una función que aplica la capacidad k del agente i en el objetivo j en el instante t.

$$Apply : A_i \times C_k \times O_j^t \rightarrow Boolean \tag{5.1}$$

[19] Se entiende por vecindario de un agente a aquellos agentes que potencialmente pueden cooperar para una tarea determinada, es decir, agentes separados por una distancia semántica pequeña. Por ejemplo, el vecindario de una agente sensor está formado por otros agentes sensores que comparten áreas de visión.

Por ejemplo, es posible aplicar la capacidad de seguimiento (tracking) utilizando la información visual obtenida por el sensor en forma de imágenes, para un objetivo determinado en un instante de tiempo. Durante el tiempo en que se está realizando el seguimiento de un objetivo por un agente, la función *Apply* proporcionaría un valor verdadero.

Definición 2. Coalición. Una coalición en el instante t, es una tripleta $\Psi_i = <Co_i,\ C_l,\ O_k^t>$, formada por un conjunto de agentes, una capacidad y un objetivo concreto. Donde $Co_i \subseteq \Delta_i$ es un conjunto de agentes autónomos tales que en el instante $t \rightarrow \forall A_j \in Co_i\ Apply\ (A_j, C_l, O_k^t)$ *es verdadero*. Por tanto, una coalición es un grupo temporal de agentes autónomos realizando de forma conjunta una acción para un objetivo concreto.

Dentro del modelo de agentes BDI, lo que se define como *deseos*, son las funciones que capturan la motivación de los agentes.

- El deseo de todos los agentes sensores es la vigilancia permanente de su entorno, siendo $Surveillance(O)$ la detección y seguimiento de todos los objetivos. Se define formalmente como: $\forall i \in S \cdot (Des\ S_i\ Surveillance(O))$.

El deseo de todos los agentes fusión es:

- Fusionar la información recibida acerca de los objetivos por todos los agentes sensores proporcionando la mejor estimación posible dadas las estimaciones de cada uno de los agentes sensor. Definido formalmente como: $\forall i \in F \cdot (Des\ F_i\ FuseData(O))$.

La ultima característica dentro del modelo BDI, son las *intenciones*, que se definen como los pasos básicos que un agente decide llevar a cabo para alcanzar sus *deseos*. Las intenciones de los agentes sensores son:

- Todos los agentes sensores ($\forall i \in S$), tienen la intención de hacer una coalición con otros agentes sensores (S_j) de su vecindario (Θ_i), que involucre el objetivo O_k, a partir del instante t y aplicando la capacidad C_l. Lo que se define formalmente como: $\forall i \in S \cdot (Int\ S_i\ \forall j \in \Theta_i \cdot (MakeCoalition(S_j,\ O_k^t,\ C_l)))$.

- Todos los agentes sensores ($\forall i \in S$), tienen la intención de aceptar formar una coalición con otros agentes sensores (S_j) de su vecindario (Θ_i), que involucre el objetivo O_k, a partir del instante t, aplicando la capacidad C_l. Lo que se define como: $\forall i \in S \cdot (Int\ S_i\ \exists j \in \Theta_i \cdot (AcceptCoalition(S_j,\ O_k^t,\ C_l)))$.

- Todos los agentes sensores ($\forall i \in S$), tienen la intención de rechazar una coalición con otros agentes sensores (S_j) de su vecindario (Θ_i), que involucre el objetivo O_k, a partir del instante t, para aplicar la capacidad C_l. Lo que se define como: $\forall i \in S \cdot (Int\ S_i\ \exists j \in \Theta_i \cdot (DenyCoalition(S_j,\ O_k^t,\ C_l)))$.

- Todos los agentes sensores ($\forall i \in S$) tienen la intención de abandonar la coalición que involucra al objetivo O_k, a partir del instante t, usando la capacidad C_l. Lo que se define como: $\forall i \in S \cdot (Int\ S_i\ LeaveCoalition(S_i,\ O_k^t,\ C_l)))$.

- En el sistema multi-agente CSA existe al menos un agente sensor ($\exists i \in S$) con la capacidad de seguimiento. Lo que se define como: $\exists i \in S \cdot (Int\ S_i\ Tracking(O_k))$. La capacidad de seguimiento da lugar a los procesos de detección de objetos (5.4.2), asociación de datos (5.4.3) y estimación de los objetivos (5.4.4).

- En el sistema multi-agente CSA existe al menos un agente sensor ($\exists i \in S$) con la capacidad de reconocimiento. Lo que se define como: $\exists i \in S \cdot (Int\ S_i\ Recognition(O_k))$.
- En el sistema multi-agente CSA existe al menos un agente sensor ($\exists i \in S$) con la capacidad de calibración. Lo que se define formalmente como: $\exists i \in S \cdot (Int\ S_i\ Calibration)$.
- Todos los agentes sensores ($\forall i \in S$) pueden comunicar a otros agentes (A_j), de su coalición Ψ_i, la información de un nuevo objetivo O_k detectado en el instante t. Lo que se define formalmente como:
  $\forall i \in S \cdot (Int\ S_i\ \forall j \in \Psi_i \cdot (SendNewTrackInfo(A_j,\ O_k^t,\ C_l)))$.
- Todos los agentes sensores ($\forall i \in S$) pueden comunicar a otros agentes (A_j), de su coalición Ψ_i, la información actualizada de un objetivo O_k en el instante t. Lo que se define formalmente como:
  $\forall i \in S \cdot (Int\ S_i\ \forall j \in \Psi_i \cdot (SendUpdateTrackInfo(A_j,\ O_k^t,\ C_l)))$.
- Todos los agentes sensores ($\forall i \in S$) pueden comunicar a otros agentes (A_j), de su coalición Ψ_i la información de la desaparición de un objetivo O_k en el instante t. Lo que se define formalmente como:
  $\forall i \in S \cdot (Int\ S_i\ \forall j \in \Psi_i \cdot (SendDeleteTrackInfo(A_j,\ O_k^t,\ C_l)))$.

Las intenciones de los agentes fusión son similares, entre las que cabe destacar:

- Todos los agentes de fusión ($\forall i \in F$) tienen la intención de recibir y fusionar información de otro agente A_j, de su coalición Ψ_i, que corresponde a la detección de un nuevo objetivo O_k en el instante t. Lo que se define formalmente como: $\forall i \in F \cdot (Int\ F_i\ \forall j \in \Psi_i \cdot (FuseNewTrackInfo(A_j,\ O_k^t,\ C_l)))$.
- Todos los agentes de fusión ($\forall i \in F$) tienen la intención de recibir y fusionar información de otro agente A_j, de su coalición Ψ_i, que corresponde a la actualización de un objetivo O_k en el instante t. Lo que se define formalmente como: $\forall i \in F \cdot (Int\ F_i\ \forall j \in \Psi_i \cdot (FuseUpdateTrackInfo(A_j,\ O_k^t,\ C_l)))$.
- Todos los agentes de fusión ($\forall i \in F$) tienen la intención de recibir y fusionar información de otro agente A_j, de su coalición Ψ_i, que corresponde a la eliminación de un objetivo O_k en el instante t. Lo que se define formalmente como: $\forall i \in F \cdot (Int\ F_i\ \forall j \in \Psi_i \cdot (FuseDeleteTrackInfo(A_j,\ O_k^t,\ C_l)))$.

En la sección 4.4 se describió el interprete BDI básico, que consiste en el ciclo de ejecución interno de cada uno de los agentes en el sistema, por el cual se deciden las objetivos a ejecutar en cada instante. A continuación se describe de una forma más detallada el interprete BDI utilizado por los agentes en CSA:

```
BDI Interpreter:
  Initialize_State (B, D, I);
  do
    get_external_events (e);
    new_options := trigger_plans (e, B, D);
    selected_option := select_option (new_options);
    update_intentions (selected_option, I);
    selected_intention := select_intention (I);
    execute (selected_intention);
    update_intention (selected_intention, I);
    get_observation (O);
    update_beliefs (O, B);
    update_desires (B, I)
    drop_successful_plans (B, I);
    drop_impossible_plans (B, I);
  until finish
```

El primer paso del interprete BDI es inicializar el estado del agente, es decir, el conjunto de creencias, deseos e intenciones, a continuación, el agente empieza la ejecución del ciclo BDI. En primer lugar, obtiene los eventos externos, que en este caso particular están basados en mensajes externos e imágenes de entrada obtenidas por la cámara, después, se disparan los posibles planes dependiendo de los eventos obtenidos, las creencias actuales y los deseos. Estos planes se almacenan en la variable *new_options*, después se selecciona un plan de entre todos los posibles planes invocando al método *select_option*(*new_options*) y se actualizan las intenciones actuales del agente, utilizando las intenciones anteriores y las del plan elegido. Se selecciona una intención y se ejecuta con el método *execute*(*selected_intention*) y una vez ejecutada, se actualiza el estado de las intenciones y el conjunto de las creencias, con el resultado de lo observado por la ejecución de la intención con el método *update_beliefs*(O, B). Los deseos también se actualizan con el método *update_desires*(B, I). Finalmente, antes de terminar el ciclo, los planes terminados con éxito son eliminados de la cola de planes a ejecutar con el método *drop_successful_plans*(B, I) y también los planes imposibles con el método *drop_impossible_plans*(B, I). Este interprete BDI se ejecuta dentro de cada uno de los agentes de forma concurrente con la gestión de los mensajes y de los eventos externos. Las funciones *trigger_plan* que se encargan de disparar los planes seleccionados y *select_option*() que selecciona el siguiente plan a ejecutar, basan su razonamiento en un motor de reglas, el cual se detalla en el apartado 5.7.6.

5.3.1. Protocolo para la formación de coaliciones

A la hora de diseñar un protocolo, es necesario describir las capacidades de interacción de los agentes involucrados y definir el tipo de mensajes utilizado para el intercambio de datos. A continuación hay que determinar las posibles secuencias de intercambio de mensajes que se pueden producir, así como el estado interno que deben tener los agentes cuando se producen estos intercambios. Finalmente, es bastante útil describir el protocolo con un diagrama de secuencia en UML.

Wooldridge (2000), estableció que el primer paso en un proceso cooperativo de solución de problemas, empieza cuando un agente reconoce el potencial para la acción cooperativa. En el sistema CSA, este proceso empieza cuando el agente de fusión detecta la necesidad de formar una coalición; por tanto, el agente fusión es el iniciador de la coalición. La formación de la coalición y las necesidades de fusión de datos son centralizadas en el agente fusión, que es el responsable de reconocer el potencial de formar coaliciones para: controlar las desviaciones de los agentes sensores, controlar el hand-over entre áreas adyacentes y los eventos que dependen de situaciones de contexto.

Como ya se ha indicado al principio, la cooperación solo existe en el estado mental del agente de fusión, que es el que inicia el proceso, llamado reconocimiento de cooperación y que se define formalmente como:

$$\exists F_c \cdot (Bel\ F_c\ \exists j \in \Theta_j\ \cdot (Bel\ A_j\ MakeCoalition(A_i, O_k^t, C_l))).$$

Esta definición indica que el agente fusión F_c, tiene el conocimiento de que otro agente A_j, existe y puede querer formar una coalición (con otro agente A_i) para el grupo de objetivos $\{O_k\}$ en el instante t y aplicando la capacidad C_l.

A continuación, el agente fusión F_c, manda una serie de mensajes de tipo *Call-for-Coalition* a los agentes sensores pertenecientes al vecindario, en este caso al agente A_j con la intención de formar una coalición con éxito. Cada mensaje es enviado por la intención *MakeCoalition* y usando el concepto de vecindario.

- **Call-for-Coalition**.

  $<F_c,\ cfp\ (A_j,\ <A_j, MC>, Ref\ A_j O_k^t C_l,\ \phi(A_j, O_k^t, C_l))>$
  $\equiv <F_c,\ query\text{-}ref\ (A_j, Ref\ A_j O_k^t C_l (I_{Fc}\ Done\ (<A_j, MC>, \phi(A_j, O_k^t, C_l)) \Rightarrow$
  $(I_{Fc}\ Done\ (<A_j, MC>,\ \phi\ (A_j, O_k^t, C_l)))))>$

  donde MC se refiere a la acción *MakeCoalition*, *cfp* (Call-for-Proposal) se utiliza para comprobar la disponibilidad de un agente para llevar a cabo la acción de *MakeCoalition* (FIPA Communicative Act Library Specification (FIPA, 2002)) ,*query-ref* es la acción para preguntar a otro agente por el objeto referido por la expresión referencial, *Ref* es la expresión referencial, en este caso se trata de una capacidad específica aplicada en un objetivo concreto en el instante t ($A_j O_k^t C_l$), *Done* indica que la acción, en este caso la petición de formar la coalición, se ha llevado a cabo e I indica la intención de llevar a cabo la acción.

Se asume que todos los agentes sensores a los que se les solicita formar una coalición son consistentes entre ellos. Por ejemplo, si F_c pregunta para formar una coalición entre S_1, S_2 y S_3, los tres deben tener la posibilidad de formar una coalición; es decir, deben formar parte del mismo vecindario. Por tanto, el agente de fusión debe tener una tabla donde se almacenen las posibles opciones para formar una coalición entre todos los agentes sensores. El mensaje *Call-for-Coalition* cumple con el estándar FIPA que añade un performative a cada acto de comunicación.

Cuando el agente A_j recibe el mensaje, tiene dos posibilidades: aceptar o rechazar la propuesta de coalición. Esta decisión está basada en su conocimiento interno de la situación percibida por medio del sensor visual y de otros mensajes recibidos. El mensaje *Accept-Coalition* se

envía por medio de la intención *AcceptCoalition* e indica la aceptación de formar la coalición, mientras que el mensaje *Reject-Coalition* se envía por la intención *DenyCoalition* e indica la negación de formar la coalición. A continuación se definen formalmente cada uno de estos mensajes.

- **Accept-Coalition**

  $<A_j$, *accept-proposal* $(F_c<F_c, MC>, \phi(A_j, O_k^t, C_l))>$
  $\equiv <A_j, inform(F_c, I_{F_c}\ Done(<F_c, MC>,\ \phi(A_j, O_k^t, C_l)))>$

  donde MC se refiere a la acción *MakeCoalition*, *accept-proposal* es una aceptación general de la propuesta que ha sido previamente enviada (normalmente por medio de un mensaje de tipo propose act), en este caso la acción *MakeCoalition*.

Con este mensaje, el agente que envía su aceptación (A_j) informa al receptor (F_c) que tiene la intención de llevar a cabo (en algún momento en el futuro) la acción (MC), tan pronto como la precondición dada ($\phi(A_j, O_k^t, C_l)$) sea o llegue a ser verdad.

- **Reject-Coalition**

  $<A_j$, *reject-proposal* $(F_c,\ <F_c, MC>,\ \phi(A_j, O_k^t, C_l)\ ,\ \Gamma)>$
  $\equiv <A_j, inform(F_c,\ \neg I_{A_j} Done(<A_j,\ MC>, \phi(A_j, O_k^t, C_l))\ \wedge \Gamma)>$

  Con este mensaje, el agente A_j informa al agente F_c que, debido a la preposición Γ, A_j no tiene la intención de llevar a cabo la acción *MakeCoalition* con la precondición $\phi(A_j, O_k^t, C_l)$ para F_c.

Una consecuencia de la autonomía de los agentes es que el proceso de formación de la coalición puede fallar. Esto puede ocurrir si todos los agentes sensores involucrados envían mensajes de tipo *Reject-Coalition* y además debe haber al menos dos agentes disponibles para formar una coalición. Si el proceso de formar la coalición falla, el agente de fusión (el iniciador del proceso) espera por un periodo aleatorio de tiempo y vuelve a iniciar el proceso. Los procesos de formación de coalición fallidos son reiniciados un número limitado de veces, es decir, existe un número limitado de reintentos. Por otro lado, si la coalición se ha formado con éxito, los agentes que pertenecen a la misma coalición deben enviar mensajes de tipo *Inform-Coalition*, los cuales son enviados por la intención *SendTargetInfo*. Estos mensajes transmiten la información del entorno percibida por cada uno de los agentes sensores.

- **Inform-Coalition**
  $<A_j, inform(F_c, \phi(A_j, O_k^t, C_l))>$

  Con este mensaje, el remitente (A_j) informa al receptor (F_c) acerca del valor de la proposición dada ($\phi(A_j, O_k^t, C_l)$).

Cualquier agente perteneciente a una coalición puede abandonarla, solo necesita mandar un mensaje al agente de fusión que gestiona la coalición y dejar de enviar mensajes de tipo *Inform-Coalition*. El mensaje de abandono llamado *Cancel-Coalition* es enviado por la intención *LeaveCoalition*:

- **Cancel-Coalition**
  $<A_j, cancel(F_c, MC)> \equiv$
  $<A_j, disconfirm(F_c, I_{A_j} Done(MC))>$

  Con este mensaje, el agente A_j informa al agente F_c que ya no tiene la intención de llevar a cabo la acción MC.

Si el número de agentes sensores involucrados en la coalición desciende hasta uno solo, el agente de fusión informa al agente de sensor que la coalición se ha roto. Además la coalición se puede romper por los agentes de fusión en el caso de que detecten grandes desviaciones en el seguimiento. La acción de informar el fin de una coalición se lleva a cabo con la intención *InformBrokenCoalition*, que dispara los siguientes mensajes:

- **Inform-Broken-Coalition**
  $<F_c, inform(A_j, Broken\text{-}Coalition\ (\Psi_i))>$

  Con este mensaje, el agente F_c informa al agente A_j que la coalición Ψ_i se ha roto.

Cuando los agentes sensores rompen una coalición para un objetivo específico, la información acerca de ese objetivo se sigue enviando al agente de fusión, pero sin formar parte de una coalición. En el caso del agente fusión, el mensaje *Inform-Broken-Coalition* activa la intención *DestroyCoalition*. La figura 5.4 muestra un ejemplo de uso del protocolo de formación de coaliciones utilizando un diagrama de secuencia UML.

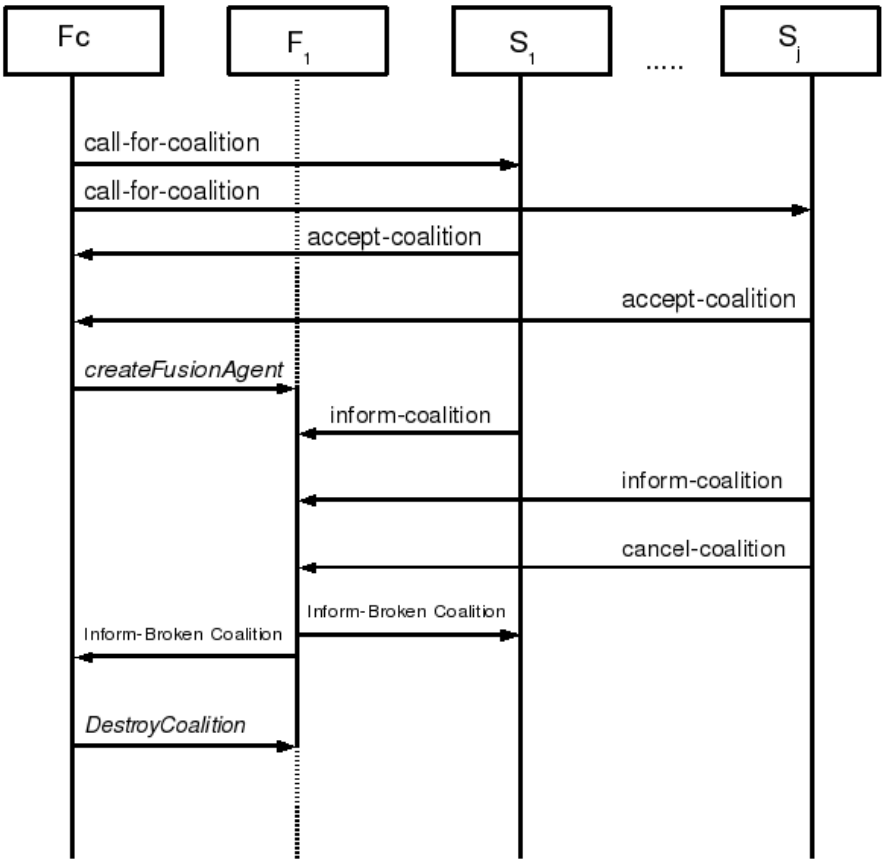

Figura 5.4: Diagrama de secuencia que muestra un ejemplo de los mensajes intercambiados durante el proceso de formación de una coalición dinámica. Las palabras en cursiva equivalen a las intenciones. Los parámetros de los mensajes se han omitido por claridad.

5.4. Agentes sensores

Este apartado describe en detalle los procesos que deben ejecutarse en los agentes sensores y su función dentro del sistema CSA para llevar a cabo las creencias, deseos e intenciones definidas anteriormente y obtener la información relevante del entorno. Como se observa en las figuras 5.10 y 5.11, la principales funciones de los agentes sensores se pueden dividir en: calibración, detección de objetivos, asociación de datos, seguimiento y comunicación de la información.

En una red de sensores visuales, las imágenes generadas y capturadas por cada sensor visual producen unas enormes cantidades de datos, generalmente poco útiles. Por tanto, estos datos, deben ser procesados utilizando algoritmos de análisis de imágenes y de vídeo para obtener solo la información relevante. El procesado de grandes cantidades de imágenes, obtenidas en la red visual, se lleva a cabo en los agentes sensores. El razonamiento producido por el modelo simbólico, interno de los agentes sensores, está basado en una deliberación acerca del estado del mundo (incluyendo su evolución pasada) y de las acciones que son más probables que tengan lugar en el futuro. El agente sensor se encarga de realizar el seguimiento de los objetivos dentro de su campo local de visión y de enviar los datos al agente de fusión que le corresponde. Cada uno de los agentes sensor, se encuentra coordinado y en cooperación con otros agentes sensores para mejorar el proceso de vigilancia

A continuación, se describen cada uno de las fases que tienen lugar internamente en el proceso de los agentes sensores: calibración (5.4.1), detección de objetos (5.4.2), asociación de datos (5.4.3), estimación de los objetivos detectados (5.4.4) y comunicación de la información (5.4.5).

5.4.1. Calibración

La funcionalidad de calibración se corresponde con la intención de *Calibration* definida anteriormente. Tal y como se ha explicado y detallado en el apartado 3.3, en una red distribuida de sensores visuales, cada sensor tiene su propio campo de visión y la correspondencia entre las múltiples cámaras, conlleva buscar asociaciones entre los objetos de las diferentes cámaras en el mismo instante de tiempo. El proceso de combinar distintos campos de visión y generar una visión global del área que se está observando se conoce como alineación espacial u calibración. El proceso de calibración comprende la calibración geométrica de cada cámara, que consiste en estimar el conjunto de parámetros intrínsecos y extrínsecos de la cámara.

Se asume, que la cámaras, por medio de las cuales los agentes sensores obtienen la información, son fijas durante la ejecución del sistema; por tanto, la imagen de referencia utilizada para la calibración puede ser la misma y el sistema solo necesita ser calibrado una vez, utilizando posteriormente la información de calibración de cada agente sensor durante la ejecución del mismo. Es decir, la fase de calibración se puede dividir en dos etapas:

1. Obtención de los parámetros intrínsecos e extrínsecos del campo de visión de la cámara con respecto al plano global y de esta forma poder aplicar la homografía correspondiente a los datos de la cámara para transformarlos a coordenadas globales.

2. Aplicar la transformación del plano de la imagen al plano global, tantas veces como sea necesario durante la ejecución del sistema.

Cada agente sensor, almacena internamente la información de calibración correspondiente al campo de visión de la cámara por la cual obtiene información del entorno. La calibración se realiza con respecto al plano del suelo de la escena, sobre la cual se desea realizar el seguimiento, asumiendo por tanto que todos los objetos se mueven en el plano ($Z = 0$). El método utilizado (ver figura 5.5) para obtener la posición del objeto en el plano, consiste en proyectar el centroide del blob que corresponde al objeto en el plano del suelo y aplicar la homografía en ese punto, para transformar las coordenadas locales en las coordenadas globales de representación.

Figura 5.5: Proyección en el suelo del punto del centroide del blob detectado para aplicar a continuación la transformación de la calibración y obtener la posición del punto en coordenadas globales.

El trabajo de Khan & Shan. (2003); S. Khan & Shah (2001), también utiliza los puntos localizados en los pies de las personas para hacer concordar a las personas en múltiples vistas, utilizando la restricción de la homografía basada en el plano del suelo.

5.4.2. Detección de objetos

La detección de objetos es una fase clave y crítica dentro de las redes de sensores visuales, tanto si lo que se desea es simplemente realizar la detección o si además se desea realizar un seguimiento de los objetivos. En este último caso, la fase de detección de objetos viene acompañada de la fase de asociación de datos y de la estimación de los objetivos. La importancia de esta fase, viene dada porque afecta a todas las demás fases del sistema, ya que todos los errores que se producen en la fase de detección de objetos se propagan por las distintas fases.

Una vez se ha realizado la fase de calibración descrita anteriormente, la cual normalmente se lleva a cabo en una fase previa a la puesta en marcha del sistema, el primer proceso que se ejecuta es la detección de objetos o blobs. Por medio de una correcta detección, una red de sensores visuales puede gestionar un número variable y dinámico de objetos en instantes de tiempo impredecibles.

En la figura 5.6, se muestra una imagen para clarificar la información de entrada de los agentes sensores y la salida obtenida en la fase de detección de objetos. Las imágenes de la izquierda muestran los fotogramas de entrada en un instante determinado, mientras que las

de la derecha muestran los píxeles de la imagen (en color blanco) que han sido detectados como píxeles en movimiento.

Figura 5.6: Ejemplo de detección de píxeles en movimiento de dos imágenes capturadas en instantes distintos por una misma cámara. Las imágenes de la derecha son el resultado del algoritmo de detección de movimiento sobre la imagen de la izquierda.

De entre todos los métodos posibles para la detección de objetos en movimiento, los agentes sensores utilizan una implementación del algoritmo descrito por Li et al. (2003). El proceso de detección de objetos en movimiento se lleva a cabo en dos fases. En primer lugar se realiza la detección de píxeles en movimiento, a continuación aquellos píxeles, que se considera que están en movimiento, se utilizan para realizar su agrupamiento en blobs. La detección de píxeles en movimiento se lleva a cabo modelando de forma estadística los píxeles del fondo (background) de las imágenes donde se quiere realizar la detección de los objetos en movimiento. Dada una secuencia de vídeo como entrada, el algoritmo de Li et al. (2003) utiliza el color de cada píxel y las co-ocurrencias estadísticas del color. El color de cada píxel y las distribuciones de co-ocurrencias del color están representadas por histogramas. Se utiliza la regla de decisión de Bayes para clasificar si un píxel forma parte del fondo o no y el fondo es actualizado después de cada clasificación de los píxeles.

El algoritmo es capaz de soportar con éxito cambios graduales o bruscos en el fondo, así como objetos presentes en el fondo de la imagen, que están en movimiento o estacionarios. Los objetos del fondo estacionarios pueden ser paredes, puertas y muebles en una escena interior, así como edificios, vegetación y terreno en una escena exterior. Los objetos de fondo en movimiento pueden ser las ramas de los arboles, las superficies de agua, las pantallas de ordenador, los ventiladores, las escaleras mecánicas, etc... Los objetos estacionarios del fondo se representan por las características de color de los píxeles mientras que los objetos en movimiento se representan por las co-ocurrencias de color. Además, el fondo puede sufrir dos tipos de variaciones en el tiempo: la primera consiste en cambios graduales que se puede deber a los cambios de iluminación del día a la noche; la segunda consiste en cambios repentinos que se deben a modificaciones como encender o apagar las luces, dejar objetos nuevos o mover mesas y sillas. Finalmente, hay objetos en movimiento que pueden pasar a formar

parte del fondo al cabo de un tiempo, por ejemplo un coche en movimiento que es aparcado.

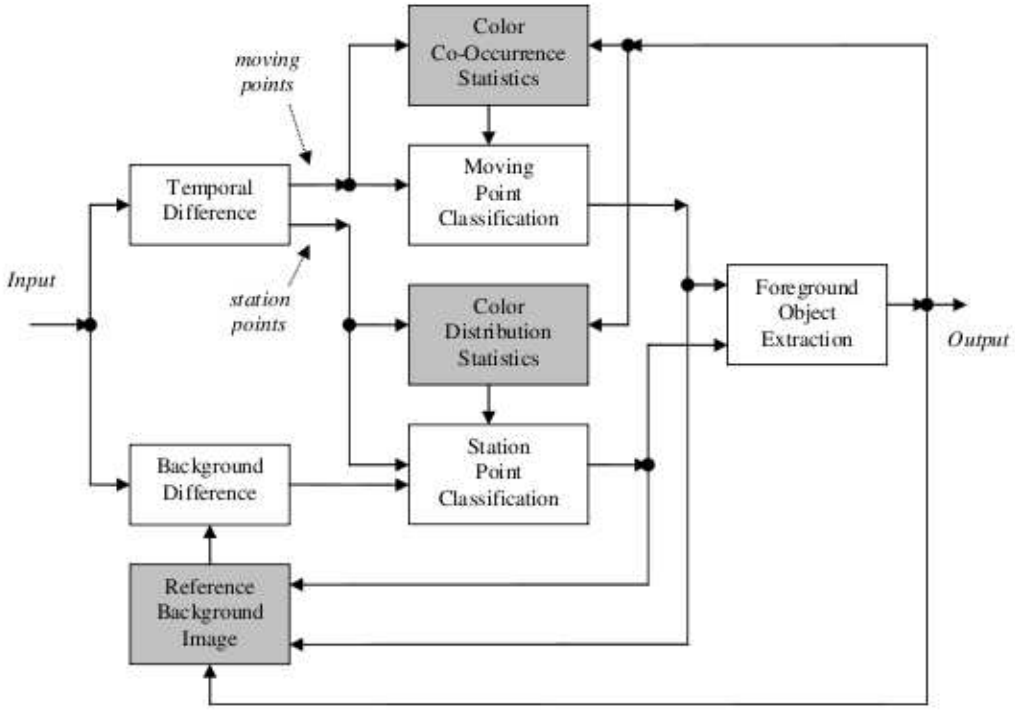

Figura 5.7: Diagrama de bloques de los componentes del algoritmo de detección de objetos en movimiento. [Adaptada de Li et al. (2003).]

El algoritmo consta de los bloques mostrados en la figura 5.7, que dividen al algoritmo en 4 fases: (1) detección de los cambios, (2) clasificación de los píxeles de primer plano y del fondo, (3) segmentación de los objetos del primer plano y (4) aprendizaje y mantenimiento del fondo. A continuación se describen brevemente estas 4 fases.

1. **Detección de los cambios**. En este primer paso, los píxeles con cambios no significativos son descartados utilizando la diferencia temporal y la diferencia con la imagen del fondo que se utiliza como referencia. En primer lugar se lleva a cabo una diferencia de la imagen para la componente de cada color, utilizando un umbral adaptativo con el método descrito en (Rosin, 1998). Los resultados de las tres componentes se combinan para generar la imagen de referencia del fondo.

2. **Clasificación de los píxeles de primer plano y de fondo**. En este segundo paso, los píxeles asociados con objetos estacionarios o en movimiento son clasificados bien como fondo o bien como píxeles de primer plano. Para la clasificación, se utilizan las estadísticas de los colores y las co-ocurrencias por medio de la regla de Bayes.

3. **Segmentación de los objetos**. En esta fase, se aplica una operación morfológica para eliminar los errores dispersos y conectar los píxeles de primer plano.

4. **Aprendizaje y mantenimiento del fondo**. El mantenimiento del fondo, adapta el modelo del fondo (imagen utilizada como referencia) a los cambios que ocurren en el tiempo. El mantenimiento se lleva a cabo en dos partes, la primera consiste en actualizar las tablas de las estadísticas y la segunda en actualizar la imagen de referencia. La actualización de las tablas de las estadísticas conlleva mantener dos tablas para cada píxel (color y co-ocurrencia de color). De esta forma se consigue tener en consideración los cambios graduales y los cambios repentinos.

En definitiva, el algoritmo para la detección de los objetos en movimiento está basado en el modelado de una imagen utilizada como referencia (imagen del fondo) y en una clasificación Bayesiana de los píxeles. Este algoritmo, es capaz de segmentar los objetos de primer plano en situaciones donde los píxeles del fondo de la imagen pueden sufrir variaciones.

5.4.3. Asociación de datos (paso de blobs a objetivos)

Una vez realizada la detección de los blobs[20], en cada uno de los campos de visión de las cámaras, el siguiente paso que se ejecuta internamente en cada agente sensor, es la asociación de datos. El problema de la asociación de datos y las técnicas más utilizadas para resolverlo se ha descrito en profundidad en la sección 2.7.1 para las técnicas de asociación centralizadas y en la sección 2.8.3 para las técnicas de asociación distribuidas. En este caso, se trata de un problema de asociación centralizado, ya que la asociación tiene lugar internamente en cada uno de los agentes sensores, para relacionar la información visual obtenida fotograma a fotograma.

La asociación de datos en vídeo es un problema de correspondencia blob a pista. Varios o ningún blob pueden ser asignados a la misma pista y simultáneamente varias pistas se pueden solapar y compartir blobs. Básicamente, el problema consiste en calcular y asignar el peso que relaciona el conjunto de los nuevos blobs detectados (donde puede ocurrir que se hayan detectado nuevos objetivos) con el conjunto de objetivos previamente detectados. Dentro de cada uno de los agentes sensores, el proceso de asociación de datos se lleva a cabo en cada instante de muestreo, para relacionar las nuevas observaciones obtenidas (en forma de blobs) con los objetivos previamente detectados. La salida de la fase de asociación de datos, es un conjunto de blobs donde se presupone que cada blob pertenece a uno de los objetos detectados. Esta fase, también se conoce con el nombre de detección de blobs (blob detection) e implica la asociación de regiones de píxeles en movimiento a cada uno de los objetivos previamente detectados o bien la creación de un nuevo objetivo.

El módulo de detección de blobs o asociación de blobs a objetivos de los agentes sensores, está basado en una implementación del algoritmo de componentes conectadas de Senior et al. (2006) y consta de los siguientes pasos:

1. Calcular las componentes conectadas de la mascara del fondo obtenida en la fase de detección de píxeles en movimiento. Cada componente es considerada un blob.

2. Buscar los blobs de los objetivos obtenidos en el instante anterior que coinciden con los actuales.

[20] Recordemos que un blob se define como un conjunto de píxeles en movimiento que pertenecen al mismo objeto.

3. Añadir un nuevo blob en la lista de blobs si se observa que puede ser seguido de un fotograma a otro. Es decir, la continuación de un blob existente viene dado por un posible movimiento uniforme de la máscara de píxeles obtenida en el algoritmo de detección de movimiento.

La estructura del algoritmo de una forma más detallada es la siguiente. En primer lugar, las regiones de los píxeles en movimiento de cada fotograma se agrupan para formar las componentes conectadas; se utiliza un filtro de tamaño, para eliminar el ruido que aportan los grupos pequeños de píxeles y a continuación, cada componente se almacena en un bounding box o caja envolvente. En los fotogramas siguientes, el proceso asocia las cajas envolventes detectadas previamente con las nuevas cajas envolventes obtenidas por el proceso en el instante actual. La asociación se lleva a cabo construyendo una matriz de distancias que muestra la distancia entre las cajas envolventes. La distancia de cada caja envolvente (ver figura 5.8), se calcula como la menor distancia del centroide C_a de la caja A al punto más cercano de la caja en B o del centroide C_B de la caja B al punto más cercano en A. Si cualquiera de los centroides queda dentro de la caja envolvente del otro, su distancia es cero (imagen de la derecha de la figura 5.8). La razón del uso de la distancia de las centroides de las cajas envolventes de esta forma y no de la distancia euclídea entre los centroides, viene dada por el gran salto que existe en la distancia euclídea cuando dos cajas se juntan o se rompen. Además, se establece una distancia temporal entre las observaciones, para penalizar aquellas que no se ha recibido información durante un tiempo. Se utiliza una matriz de correspondencia para indicar los blobs asociados a pistas y viceversa (ver figura 5.9). Esta matriz $A[k]$, se define como $A_{ij} = 1$ si el blob $B_i[k]$ está asignado a la pista P_j; en otro caso $A_{ij} = 0$. Si los blobs extraídos en el fotograma k son $B[k] = B_1[k], ..., B_N[k]$ y las pistas creadas hasta ese instante son $P[k-1] = P_1[k-1], ..., P_M[k-1]$, el tamaño de la matriz NxM cambia en el tiempo, debido a que el número de blobs detectados depende de los efectos variables en el procesado de la señal y el número de objetos bajo el área de visión también es variable. El algoritmo de asociación debe considerar las siguientes cuatro posibilidades:

1. **Detección de un nuevo objetivo y creación de una nueva pista:** Esta situación se corresponde con la creación de una nueva pista. Esto ocurre cuando en la matriz de correspondencia (donde las filas corresponden a pistas existentes y las columnas a objetivos detectados), existen columnas con todos sus elementos cero, debido a que no se han asociado con ninguna pista. En este caso se crea una nueva pista asignando el nuevo objetivo detectado.

2. **Actualización de un objetivo existente y de su pista correspondiente:** Consiste en la actualización de una pista existente. Esto se puede producir porque el objetivo ya no existe y es necesario eliminarlo de la matriz. Las filas de la matriz con todos sus elementos a cero corresponden a esta situación, es decir no existe ningún objetivo que se puede asignar a esa pista.

3. **Unión de dos objetivos:** Esta situación corresponde a la unión de dos pistas y se produce cuando dos o más pistas corresponden a la misma región o caja envolvente. Dentro de la matriz de distancias la situación es detectada porque una columna en la matriz tiene más de una entrada, es decir, un objetivo se asocia a más de una pista.

4. **División de dos objetivos:** Consiste en la separación o división de una pista en 2 o más pistas. Si dos objetivos que han sido detectados juntos (como si fueran uno solo) se empiezan a separar, las partes se siguen tratando como una pista hasta que se separen lo suficiente y las dos cajas envolventes sean consideradas dos pistas distintas.

Debido a los posibles errores del algoritmo de detección de objetivos en movimiento, los cuales se deben principalmente a los cambios de iluminación, algunas regiones de la escena pueden ser detectadas erróneamente como pistas. Sin embargo, muchas de estas falsas pistas se pueden filtrar y descartar utilizando umbrales en los siguientes parámetros: el número de fotogramas seguidos en los que tiene que aparecer la pista candidata, el tamaño de la pista candidata y un factor de olvido para indicar que la pista candidata deja de existir.

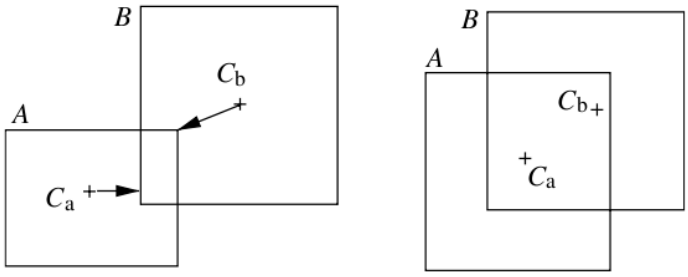

Figura 5.8: Medida de distancia de dos cajas envolventes. En la imagen de la izquierda, la distancia entre la caja A y B, es la distancia del centroide de A (C_a) al borde de B, mientras que la distancia de B a A es la distancia del centroide de B (C_b) al borde de A. En la imagen de la derecha la distancia entre las dos cajas envolventes es cero.

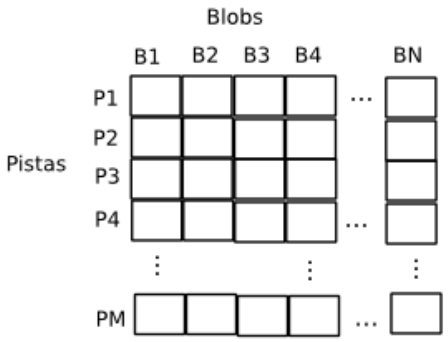

Figura 5.9: Matriz de correspondencia utilizada para asignar los blobs detectados a pistas y viceversa.

5.4.4. Estimación de los objetivos

Una vez detectados los objetivos y realizada la fase de asociación, el agente sensor conoce en ese instante concreto, la posición en su campo de visión de los objetivos detectados y a continuación, se produce la fase de estimación de seguimiento de los objetivos detectados. Las técnicas de estimación actuales han sido discutidas y comentadas en la sección 2.7.2 para la versión centralizada y en la sección 2.8.4 para la versión distribuida. El método de estimación utilizado por los agentes sensores no es único, pudiendo utilizar diferentes algoritmos, los cuales pueden ser más o menos apropiados para cada una de las circunstancias. En cualquier caso se trata de algoritmos que pertenecen a la clasificación de las técnicas de estimación centralizada. Para utilizar distintos algoritmos, es necesario proporcionar la implementación del algoritmo de estimación utilizado y una referencia al mismo en el fichero de definición del agente sensor, conocido como *agent definition file*.

Para cada agente sensor i, el objetivo j detectado, O_j^i, se representa con un vector (pista o track) $\hat{x}_j^i[n]$, que contiene la descripción numérica de sus atributos y estado: localización, velocidad, dimensiones; y por otro lado también las correspondientes matrices de error de covarianza $R_j^i[n]$.

En este proceso, la localización del objetivo y el seguimiento están expresadas en coordenadas locales (píxeles) a cada uno de los campos de visión de las cámaras, siendo n el índice temporal asociado con el instante t_n. En cada instante de tiempo, la proyección del centroide del objetivo en el plano del suelo es transformado a coordenadas globales utilizando el proceso de calibración previamente realizado. Esta información es enviada al agente de fusión con el objetivo de llevar a cabo el seguimiento de forma global.

Debido a que solo la información de seguimiento de cada agente sensor es enviada por la red, el ancho de banda necesario es menor que el requerido si se enviaran las imágenes en bruto. La información de cada uno de los objetos detectados, O_j^i, contiene, entre otros datos (ver tabla 5.1), una marca de tiempo que se establece tan pronto se detecta el objetivo, con el propósito de proporcionar información temporal al proceso de fusión, debido a los posibles problemas de sincronización de relojes que se pueden producir en un sistema distribuido. Por tanto, cada una de las maquinas donde se obtiene la información debe estar lo más sincronizada posible con el resto de los relojes de las otras maquinas. Para solucionar este inconveniente, se ha optado por instalar un software de sincronización que se basa en el uso del protocolo NTP (Mills, 1991), el cual se encarga de mantener los relojes sincronizados con respecto a un servidor global.

5.4.5. Comunicación de la información de los agentes sensores

La información de seguimiento, de los objetivos detectados por los agentes sensores, se envía a los agentes de fusión. Como se ha comentado, una de las ventajas de un sistema multi-agente, es que proporciona una infraestructura para el desarrollo de sistemas basados en redes de sensores visuales y específica los protocolos de comunicación e interacción necesarios. Los agentes se comunican para obtener sus propios deseos por medio de la ejecución de intenciones o los deseos del sistema completo. La comunicación se lleva a cabo por medio del intercambio de mensajes FIPA ACL. Los mensajes FIPA ACL definen la sintaxis, semántica y comportamiento del intercambio de mensajes llevado a cabo por los agentes. Los protocolos de intercambio definidos por FIPA ACL describen como se organizan los mensajes

de forma coherente en una conversación; por tanto, definen los patrones de intercambio de los mensajes. Estos protocolos se suelen formalizar utilizando diagramas de secuencia UML.

La comunicación es necesaria para que se produzca de forma más productiva la cooperación entre los agentes. Sin embargo, es necesario definir, que información se debe comunicar, cuando y con que frecuencia se debe comunicar y a quien se debe comunicar. En un sistema distribuido de redes de sensores visuales, los agentes se comunican entre ellos para avisar de las modificaciones en el estado de los objetivos que están bajo seguimiento. En otros entornos como el Robo Soccer, además se ha demostrado que la comunicación del estado interno de los agentes puede ser muy efectiva (Stone & Veloso, 1999). En el sistema CSA, los agentes comunican la información estimada de las posiciones de los objetivos detectados, es decir, la proyección en el plano de los objetivos detectados en el campo de visión de los agentes en coordenadas globales. Sin embargo, la misma arquitectura software se puede utilizar para comunicar otro tipo de información, como por ejemplo el resultado de la clasificación de actividades, el número de personas detectadas u otro tipo de información de más alto nivel. Para ello sería necesario definir el contenido de los nuevos mensajes intercambiados y especificar su tratamiento dentro de las intenciones actuales o bien creando nuevas intenciones.

Todos los mensajes FIPA ACL consisten en actos de información basados en speech act theory. La teoría speech act es la base de comunicación de muchas plataformas de agentes y puede ser modelada utilizando operadores de acción en un entorno de planificación entre un agente interlocutor y un agente que escucha. La teoría de speech act describe formas en las cuales el lenguaje se utiliza no solo para transmitir información, sino también para lograr cambios u objetivos. Se trata de una teoría basada en la filosofía y la lingüística que se aplica específicamente en situaciones cooperativas. La teoría de speech act se basa en el trabajo del filósofo J.L Austin, que establece que algunos actos de comunicación se pueden ver como acciones que intentan conseguir un objetivo en lugar de simplemente transmitir información. Los actos de información de speech act pueden ser de distintos tipos, como: informing, requesting, accepting, rejecting, believing, etc..

La comunicación de los mensajes FIPA ACL se lleva a cabo de una forma asíncrona utilizando las estructuras de datos de colas de mensajes internas de cada agente, por tanto el receptor y el comunicador del mensaje no necesitan estar sincronizados para que la comunicación se lleve a cabo. Los mensajes enviados de un agente a otro se almacenan en la cola de mensajes del agente receptor, el cual accede al contenido del mensaje y lo procesa una vez que su ciclo de ejecución haya accedido a la cola de mensajes. La comunicación y la interacción de los agentes proporcionan una forma natural por la cual la información es intercambiada dentro de una red de sensores visuales. La información enviada por los agentes sensores a los agentes de fusión, consiste en la proyección en el suelo del centroide del blob detectado (ver figura 5.5) junto con la covarianza asociada. Esta posición en el plano local se transforma utilizando la calibración y es enviada a los agentes de fusión. En el apartado 5.7.1 se explica con más detalle la información transmitida.

5.5. Agentes de fusión de datos

Los sistemas de vigilancia, que fusionan datos de varias fuentes de información, pueden ser más robustos que aquellos que dependen de una única fuente de información, si utilizan la información redundante de forma adecuada. Además de extender la cobertura espacial, algunas de las ventajas que aporta la fusión de datos son: mejorar la precisión con la combinación por medio de la reducción de covarianza, mejorar la robustez detectando sensores con errores de funcionamiento y mejorar la continuidad de los objetivos detectados por medio del uso de información complementaria. Para poder conseguir estos objetivos, los aspectos básicos a tener en cuenta para fusionar datos de diversas cámaras son los siguientes:

- Establecer un sistema de coordenadas global y un sistema temporal sincronizado.
- Realizar el alineamiento espacio-temporal (o recalibración) de la información para garantizar que se está fusionando información sin sesgar.
- Eliminar objetivos erróneos por medio de tests de análisis.
- Asociar los datos al nivel adecuado.
- Combinar las estimaciones obtenidas por los diferentes sensores y procesadores locales.

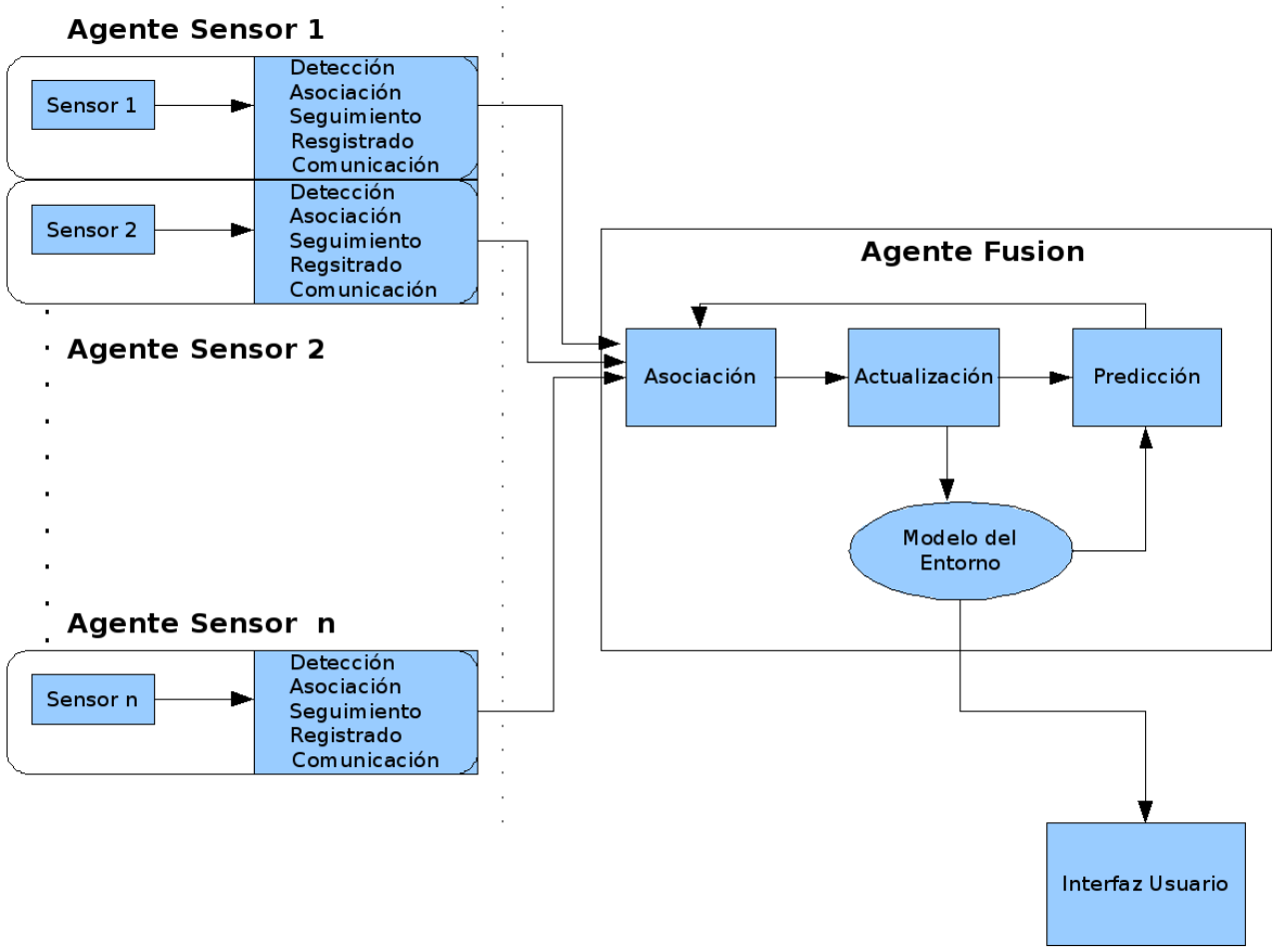

Figura 5.10: Esquema de la arquitectura del sistema en el modo de fusión pasiva

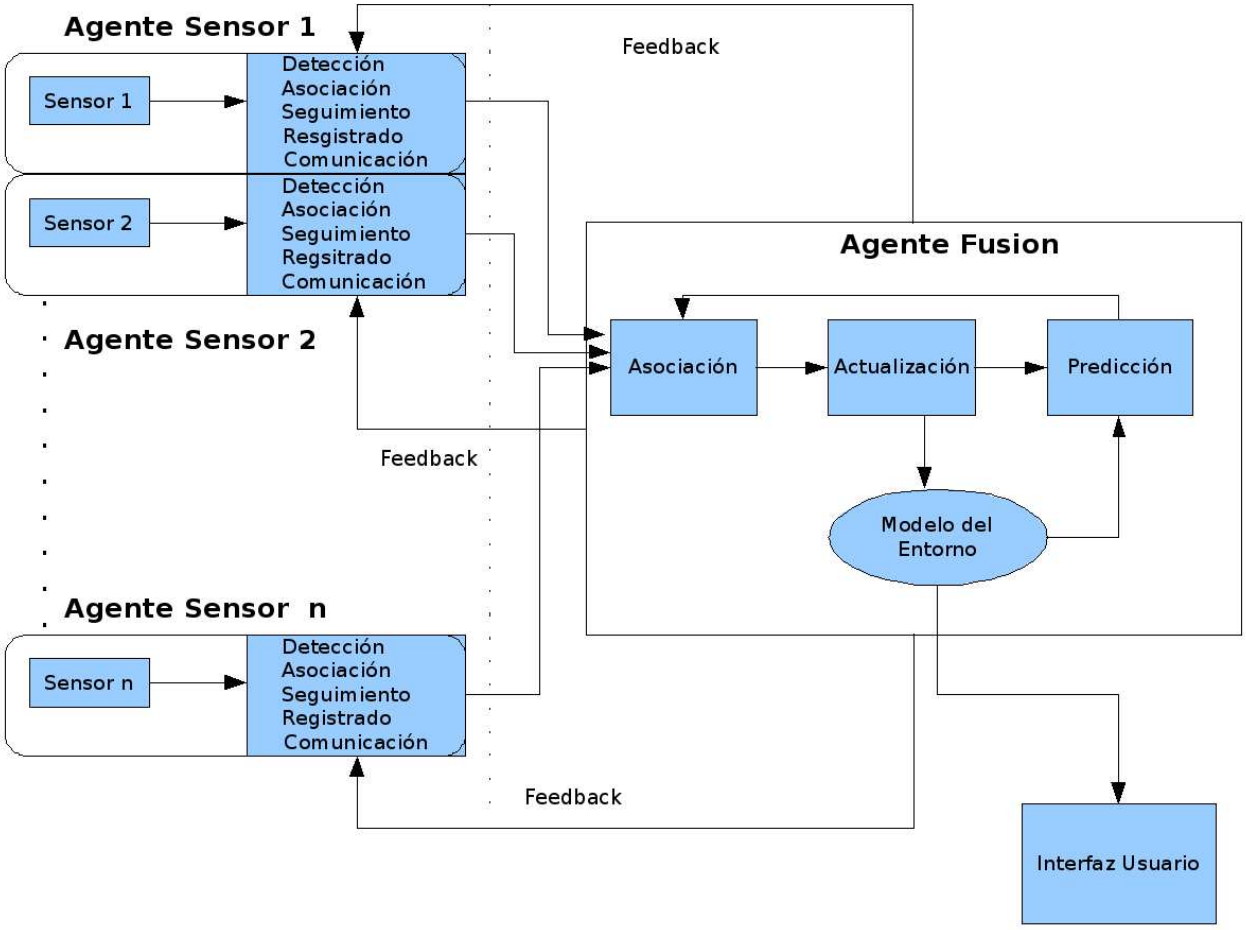

Figura 5.11: Esquema de la arquitectura del sistema en el modo de fusión activa

Dentro del sistema CSA, los agentes de fusión, son los encargados de realizar la fusión de los datos de las posiciones de los objetivos, enviadas por cada uno de los agentes sensor. Además son los responsables de enviar la información de realimentación a cada uno de los agentes sensor. Los agentes de fusión poseen la visión completa del entorno que está siendo monitorizado por los agentes sensores y también, son los encargados de iniciar el proceso de formación de coaliciones. Los criterios para la creación de coaliciones son: la predicción de la perdida de objetivos entre los agentes sensores, el disparo de eventos con información de contexto y la detección de pistas cercanas.

El sistema CSA consiste en una arquitectura distribuida jerárquica, en la cual cada agente de fusión interactúa con varios agentes sensor. Este tipo de arquitectura permite que sea más fácil la escalabilidad, puesto que si se añaden más agentes sensores, estos tienen que estar en colaboración y enviar mensajes únicamente con sus correspondientes agentes fusión. A su vez, el resultado de la fusión de la información que se obtiene en los agentes fusión se transmite a la interfaz de usuario. En la figura 5.10 se muestra la arquitectura del sistema para la versión de fusión pasiva y en la figura 5.11 la arquitectura de la fusión activa. Las diferencias entre una arquitectura y otra, consisten en el canal de realimentación que permite a los agentes sensores individuales tener una medida de la calidad de información que envían de acuerdo al resultado global fusionado.

La principal función del agente fusión es realizar de forma precisa y robusta, el proceso de fusión de pistas de cada uno de los agentes sensor, para realizar el seguimiento de los objetivos en un sistema multi-cámara de la forma más coherente posible. Las regiones con campos de visión solapadas y el establecimiento de coordenadas globales permite al agente de fusión corregir las medidas para mejorar la calidad del seguimiento y evitar los posibles errores e inconsistencias.

5.5.1. Actualización del modelo

Cuando varios agentes sensor con diferentes campos de visión realizan el seguimiento del mismo objeto, es posible, que alguno de los agentes sensor proporcione información errónea acerca de la posición del objeto. La información errónea se puede deber a varias razones: el objeto se puede ver afectado por sombras, los sensores pueden estar afectados por cambios de iluminación, pueden aparecer errores de hardware, se pueden producir oclusiones por objetos en algunos sensores, etc...

El algoritmo de fusión desarrollado, está basado en un algoritmo en dos fases. En la primera fase se seleccionan las pistas consistentes para ser fusionadas y en la segunda fase, se realiza la fusión entre las pistas que se consideran consistentes entre si.

Sea N el número total de agentes en el sistema y $\{S\}$ el subconjunto de agentes sensor que están monitorizando la misma escena con $3 \leqslant i \leqslant N$ (es decir, suponemos que existen al menos 3 agentes sensor en el subconjunto $\{S\}$ que están monitorizando la misma área) y sea j el número total de objetivos comunes en $\{S\}$ y T el conjunto de los objetivos comunes $T = \{T_1, T_2, \dots, T_j\}$, para cada uno de los objetivos, el agente sensor obtiene un vector de características. Cada característica es un atributo observable que puede ser tenido en cuenta para fusionar la información. Algunos ejemplos de características pueden ser: la posición, la velocidad, el color, el tamaño, la forma, etc....Por tanto, existe un vector de características $\hat{X}^{S_i}_{T_j}[n]$ para cada objetivo j obtenido por cada agente sensor i.

Se define F como el vector de características $F = \{\{\hat{X}^{S_1}_{T_1}[n],\ \hat{X}^{S_1}_{T_2}[n],\ \dots \hat{X}^{S_1}_{T_j}[n]\},\ \{\hat{X}^{S_2}_{T_1}[n],$

$\hat{X}^{S_2}_{T_1}[n]$, ... , $\hat{X}^{S_2}_{T_j}[n]\}$, ... , $\{\hat{X}^{S_i}_{T_1}[n]$, $\hat{X}^{S_i}_{T_2}[n]$, , ... , $\hat{X}^{S_i}_{T_j}[n]\}\}$
y P como el conjunto de matrices de covarianza asociadas $P = \{\{\hat{P}^{S_1}_{T_1}[n]$, $\hat{P}^{S_1}_{T_2}[n]$, ... $\hat{P}^{S_1}_{T_j}[n]\}$, $\{\hat{P}^{S_2}_{T_1}[n]$, $\hat{P}^{S_2}_{T_1}[n]$, ... , $\hat{P}^{S_2}_{T_j}[n]\}$, ... , $\{\hat{P}^{S_i}_{T_1}[n]$, $\hat{P}^{S_i}_{T_2}[n]$, , ... , $\hat{P}^{S_i}_{T_j}[n]\}\}$.

En esta fase del proceso de fusión, se asume que el problema de la asociación del mismo objetivo en diferentes agentes sensor está solucionado. A continuación se describe el algoritmo utilizado para la primera fase, que consiste en selección de pistas consistentes.

Fase 1. Pistas consistentes

En el entorno de las redes de sensores visuales, es necesario mantener la consistencia entre la información del mismo objetivo, obtenida por los diferentes campos de visión de las cámaras. La consistencia de la información de los diferentes sensores solo se puede mantener si existe una coherencia espacial y temporal. De otra forma, la información de seguimiento sesgada podría dar lugar a alarmas erróneas o a efectos no deseados en las trayectorias estimadas. La coherencia espacial se ha conseguido por medio de la calibración del entorno y la temporal utilizando las marcas de tiempo.

SelectConsistentTracks: $(\{S\}, \{T\}, \{F\})$.
for each $S_i \in \{S\}$
  for each common target $T_j \in \{T\}$
    $\hat{M}^{S_i}_{T_j}[n] \leftarrow$ CalculateMean$(S_i, T_j, \hat{X}^{S_i}_{T_j}[n])$
  end for
end for
Initialize fusion set: $\{S^F\} \leftarrow \{S\}$
  for each $S_i \in \{S\}$
    for each common target $T_j \in \{T\}$
      if (MD $(\hat{X}^{S_i}_{T_j}[n], \hat{M}^{S_i}_{T_j}[n])) \leq \lambda$ or IsOutOfContext $(\hat{X}^{S_i}_{T_j}[n])$
        $\{S^F\} = \{S^F\} - S_i$
          for each common target $T_j \in \{T\}$
            $\hat{M}^{S_i}_{T_j}[n] \leftarrow$CalculateMean$(S_i, T_j, \hat{X}^{S_i}_{T_j}[n])$
          end for
    end for
  end for
  if $\{S^F\} = \emptyset$
    $\{S^F\} \leftarrow$ take S_i with Min$(\hat{P}^{S_i}_{T_j}[n])$
return $(\{S^F\})$

Figura 5.12: Algoritmo Fase 1: Seleccionar las pistas consistentes de cada cámara

La primera fase del algoritmo de fusión se describe en la figura 5.12 y consiste en eliminar las pistas inconsistentes y seleccionar las consistentes. Por pistas inconsistentes, nos referimos a aquellas que nos proporcionan información con gran precisión (pequeña varianza) sobre el mismo objetivo, pero que lo sitúan en puntos del espacio muy distintos. Para detectar pistas inconsistentes se pueden utilizar dos métodos:

- Calcular la distancia de Mahalanobis (MD) (Mahalanobis, 1936) entre las características de cada agente sensor (S_i) y la media (M) para cada una de las características

candidatas.

$$MD_{S_i} = (\hat{X}^{S_i}_{T_j}[n] - \hat{M}^{S_i}_{T_j}[n])(\hat{P}^{S_i}_{T_j}[n])^{-1}(\hat{X}^{S_i}_{T_j}[n] - \hat{M}^{S_i}_{T_j}[n])^T \quad (5.2)$$

Si la distancia de Mahalanobis supera un umbral λ, la pista no es seleccionada para ser tenida en cuenta en la segunda fase.

- Teniendo en cuenta la información de contexto. La idea es establecer información de contexto a priori en la cual las medidas de seguimiento que no tienen sentido (restricciones de seguimiento espaciales) son descartadas. Por ejemplo, si un sistema de vigilancia está siguiendo a una persona dentro de una oficina, la información de contexto puede ser la posición que ocupan las mesas dentro de una oficina.

Una posibilidad para mejorar la eficiencia, a la hora de buscar las pistas consistentes entre sí, es utilizar una estructura de datos basada en KD-Trees (Bentley, 1975) para almacenar las pistas enviadas por cada uno de los agentes sensores y de esta forma el coste computacional puede estar en el orden de $O(logN)$ siendo N el número de pistas. Una estructura de datos basada en KD-Tree proporciona una forma rápida de acceder a cualquier objeto por su posición ya se navega por los datos utilizando la estructura arbórea.

Análisis de la complejidad

Un análisis de la complejidad computacional del tiempo de ejecución (en el peor caso) del algoritmo **SelectConsistentTracks** nos proporciona la siguiente información. El bucle de la línea 2 tiene una complejidad de $O(S)$, el de la línea 3 está en el orden de $O(T)$, el calculo de la media de la línea 4 está en el orden de $O(S)$, por tanto la complejidad de esta primera parte es de $O(S^2T)$. La segunda parte proporciona una complejidad de$O(S^2T^2)$, que proviene de una complejidad de $O(S)$ de la línea 6, $O(T)$ de la líneas 7, las líneas 8 y 9 son comparaciones que tienen un tiempo de ejecución constante y las líneas 10 y 11 proporcionan una complejidad de $O(ST)$. Por tanto, podemos concluir que la complejidad total de este algoritmo es de $O(S^2T + S^2T^2)$.

Fase 2. Fusionar entre pistas consistentes

Una vez que las pistas consistentes se han seleccionado, la fusión de datos se lleva a cabo teniendo en cuenta la confianza de cada pista. Se utiliza una técnica de fusión simple que consiste en ponderar cada fuente de acuerdo con su nivel de confianza ($\alpha^{S_i}_{T_j}[n]$). Por tanto, es necesario calcular el nivel de confianza $\alpha^{S_i}_{T_j}[n]$ de cada objetivo T_j para todos los agentes sensor S_i en el conjunto de sensores consistente $\{S^F\}$.

En el algoritmo previo (Figura 5.13), se calcula el nivel de confianza de cada uno de los sensores consistentes para cada objetivo común. Este valor está basado en la inversa del valor de covarianza de cada sensor para ese objetivo más un valor heurístico h^{S_i} por sensor, el cual puede ser establecido por un operador humano. A continuación, la función *Normalize*() normaliza los valores para satisfacer la siguiente ecuación 5.3.

$$\forall j \in \{T_j\} \rightarrow 1 = \sum_{i=1}^{\sharp\{S^F\}} \alpha^{S_i}_{T_j}[n] \quad (5.3)$$

CalculateWeigthValues $(\{S^F\}, \{T\}, \{F\})$.
for each consistent camera $S_i \in \{S^F\}$
  for each common target $T_j \in \{T\}$
    $\alpha_{T_j}^{S_i}[n] \leftarrow (\sharp\{S^F\})^{-1} * (\hat{P}_{T_j}^{S_i}[n])^{-1} + h^{S_i}$
  end for
end for
$\alpha_{T_j}^{S_i} Norm[n] \leftarrow$ Normalize $(\alpha_{T_j}^{S_i}[n])$
return $(\alpha_{T_j}^{S_i} Norm[n])$

Figura 5.13: Algoritmo: Calcular los valores de los pesos para la fusión de las pistas

Análisis de la complejidad

La complejidad computacional del algoritmo de **CalculateWeigthValues** es del orden de $O(S^2{}_f\, T)$, que proviene del coste computacional de ejecución de los bucles de la línea 2 ($O(S_f)$), de la línea 4 ($O(T)$) y de la normalización ($O(S_f)$).
A continuación, el vector obtenido, es usado en la siguiente fase del algoritmo.

FusionConsistentTracks $(\{S^F\}, \{T\}, \{F\}, \alpha_{T_j}^{S_i}[n])$
for each consistent camera $S_i \in \{S^F\}$
  for each common target $T_j \in \{T\}$
    $\hat{XF}_{T_j}^{S_i}[n] = \alpha_{T_j}^{S_i}[n] * value(\hat{X}_{T_j}^{S_i}[n])$
  end for
end for
return $(\hat{XF}_{T_j}^{S_i}[n])$

Figura 5.14: Algoritmo Fase 2: Fusión entre pistas consistentes

Con el algoritmo anterior, obtenemos los valores fusionados de las características de cada objetivo del conjunto de sensores consistentes.

Análisis de la complejidad

La complejidad en tiempo de ejecución del algoritmo **FusionConsistentTracks** es de $O(S_f\, T)$. Por tanto, la complejidad total del algoritmo que se lleva a cabo para realizar la fusión, tiene una complejidad de $O(S^2 T + S^2 T^2 + S^2{}_f\, T + S_f\, T)$, siendo cuadrática en el número de sensores y en el número de pistas.

5.5.2. Envío de realimentación a los agentes sensores

Una de las aportaciones del trabajo que se presenta en este documento es el uso de la fusión activa dentro de las redes de sensores visuales. Para llevar a cabo la fusión activa (ver figura 5.11) debe existir una comunicación de la información de realimentación (feed-back) entre los agentes de fusión y los agentes sensores. El objetivo de la información de realimentación, es permitir un proceso de razonamiento interno en cada agente sensor, para

realizar las correcciones oportunas en la información local basándose en el resultado fusionado. Esto permite obtener una medida de disparidad acerca de la calidad de información que se está enviando por cada uno de los agentes sensores. Si la información difiere en un valor superior a un umbral definido, el estado del objeto percibido por el agente sensor se modifica de acuerdo con el estado fusionado.

La información de realimentación, proporciona un mecanismo para mejorar la calidad de información de cada sensor local, basándose en la información global obtenida. Este proceso está relacionado con la visión local/global del entorno que posee cada uno de los agentes sensores.

5.6. Otros posibles agentes y el interfaz de usuario

En cuanto a los agentes principales del sistema (los agentes sensores y los agentes de fusión), es posible crear el número de agentes necesarios y de esta forma hacer que el sistema escale cuanto se desee. Además, también se pueden incluir otros tipos de agentes, como por ejemplo el agente de contexto y el de planificación.

Además es de utilidad disponer de una interfaz de usuario para visualizar el resultado del agente fusión. La interfaz de usuario muestra al usuario final el resultado del proceso de vigilancia obtenido por medio de la red de sensores visuales, que es el resultado de la información enviada por el agente fusión. Esta información, consiste en las trayectorias globales y fusionadas de cada uno de los objetivos detectados. La forma de mostrar la información al usuario puede variar ampliamente. Las formas de representación más comunes son utilizar modelos en tres dimensiones del entorno que se está observando o bien utilizar una representación en el plano global en dos dimensiones. El interfaz desarrollado, dentro del sistema CSA, muestra la información en el plano global en dos dimensiones, como se observa en la figura 5.15. El desarrollo de la interfaz se ha realizado leyendo ficheros de configuración de las zonas que se deben mostrar y de los planos de las zonas de un fichero xml. De esta forma es muy fácil el poder adaptar la aplicación a diferentes entornos, simplemente modificando el fichero xml de configuración.

Por otro lado, una de las ventajas del uso de una arquitectura multi-agente es su facilidad de integración con nuevos desarrollos, debido a la estandarización de la forma en la que los agentes se comunican (por medio de mensajes FIPA ACL). Por tanto, es posible incluir otros agentes como los agentes de contexto, agentes de planificación y agentes grabadores (ver figura 5.2) de forma muy sencilla.

Los agentes de contexto, se pueden encargar de almacenar el contexto específico de un entorno. El contexto, puede ser información estática, como la posición de objetos en una situación interior o puede ser una definición de situaciones dinámicas. Los agentes de planificación, pueden ayudar a mejorar el proceso global, por medio de la aplicación de algoritmos de planificación automática en el sistema. Pueden ser útiles en situaciones donde los sensores visuales se vean restringidos en cuanto a su capacidad de ejecución o de uso de energía.

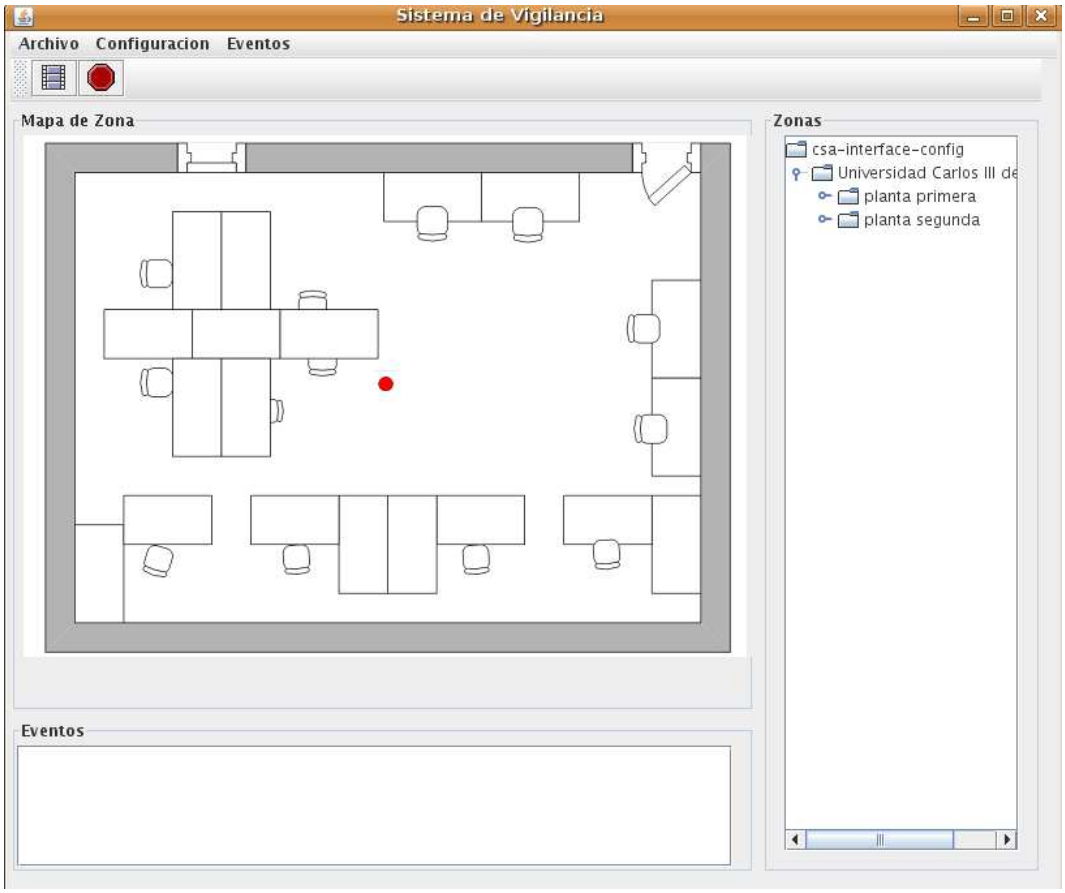

Figura 5.15: Captura de pantalla del interfaz del sistema.

5.7. Implementación del sistema CSA

Entre todas las plataformas de sistemas multi-agente actualmente disponibles, se ha optado por utilizar el framework Jadex (Pokahr et al., 2003) para desarrollar el sistema CSA. La plataforma Jadex está basada en el modelo computacional del sistema PRS (Georgeff & Lansky, 1987) (Ingrand et al., 1992) y ha sido desarrollada en la Universidad de Hamburgo. Han sido varios los motivos de la elección de Jadex, en primer lugar se trata de una implementación de una plataforma de agentes que sigue una arquitectura deliberativa basada en el modelo BDI; en segundo lugar, es conforme al estándar FIPA, lo que facilita la inter-operabilidad con otros sistemas. En tercer lugar, es de código fuente abierto, lo que nos proporciona el poder modificar y adaptar la plataforma a las necesidades específicas del sistema y finalmente, está desarrollada en Java lo que proporciona independencia de la plataforma y facilidad de desarrollo aunque se sacrifique el rendimiento que se obtendría con otras plataformas basadas en C/C++. El uso de un framework de agentes existente, proporciona la implementación de todas las tareas comunes de los agentes como la gestión de los mensajes, la codificación/de-codificación de los mensajes, la creación y destrucción de los agentes, el ciclo de vida del interprete BDI básico, etc...

Como ya se ha comentado, un agente BDI tiene creencias (acerca de sí mismo y de su

entorno), deseos (estados que el agente desea alcanzar) e intenciones (que se plasman en planes adoptados por el agente para alcanzar sus deseos). Además, mantiene un repositorio de planes disponibles conocido como librería de planes, los cuales son utilizados para llevar a cabo las intenciones. Los agentes responden a cambios en sus objetivos y creencias, originados por su percepción, y que son almacenados dentro de estructuras de datos llamadas eventos, los cuales dan lugar a nuevas creencias u objetivos. Los agentes responden a estos cambios seleccionando planes de la librería de planes e instanciando el plan elegido como una intención. El ciclo de razonamiento de cada agente dentro de la plataforma multi-agente se puede resumir como:

1. Si hay eventos que procesar, seleccionar uno de entre todos ellos.
2. Obtener de la librería de planes todos los planes disparados por el evento elegido que se pueden ejecutar.
3. Seleccionar para ejecutar uno de los planes disparados y generar una instancia de ese plan.
4. Ejecutar el plan y los posibles subplanes que genere.
5. Si el plan se termina, considerar ejecutar el siguiente plan.

El motor de razonamiento de Jadex (ver figura 5.16) está basado en el modelo BDI y permite la programación de agentes inteligentes utilizando el meta-lenguaje XML[21] y Java. Concretamente el motor de razonamiento utilizado en Jadex, para seleccionar el plan a ejecutar, está basado en un motor de reglas. Jadex representa un enfoque de desarrollo de agentes de tipo conservador por varias razones. En primer lugar, utiliza un lenguaje de programación ya existente: Java, haciendo que el desarrollador pueda utilizar los entornos de desarrollo existentes y en segundo lugar, la independencia del midleware[22] utilizado por Jadex, facilita el uso de Jadex por encima de distintas plataformas existentes que ya tengan sus propias primitivas de comunicación.

El razonamiento en Jadex es un proceso que consiste en dos componentes entrelazadas. Por un lado, el agente reacciona a los mensajes que recibe, a los eventos internos y sus objetivos por medio de seleccionar y ejecutar planes y por otro lado, el agente está continuamente razonando acerca de sus objetivos actuales para alcanzar un subconjunto consistente de los mismos.

Además, para las tareas de visión artificial, se ha utilizado la librería de visión artificial y de código fuente abierto OpenCV ([Online 04/2010]), sobre la cual se han realizado algunas modificaciones. La librería OpenCV es una colección de funciones en C y una serie de clases en C++ que implementan los algoritmos de procesado de imágenes y visión artificial más populares. OpenCV está implementada en el lenguaje de programación C/C++ y Jadex está implementado en Java. Por tanto, se ha utilizado Java Native Invocation (JNI) (JNI, 1997) para comunicar el código desarrollado dentro del framework Jadex con el código desarrollado en OpenCv.

A continuación se describen las creencias de los agentes y su implementación (5.7.1), posteriormente los deseos (5.7.2), las intenciones (5.7.3), el fichero de definición de los agentes

[21] XML se define como un meta-lenguaje porque es un lenguaje que permite definir nuevos lenguajes.

[22] Un middleware proporciona un conjunto de primitivas de comunicación de alto nivel que son requeridas para el desarrollo de aplicaciones distribuidas en red.

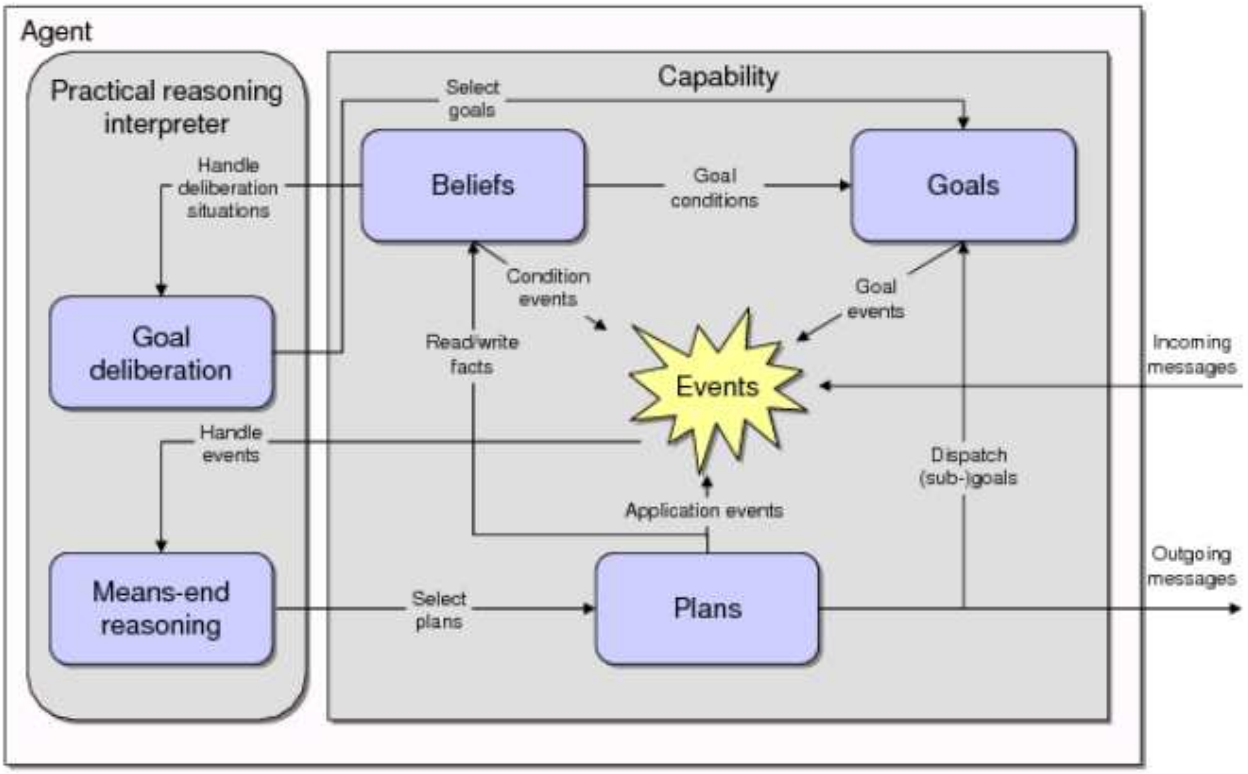

Figura 5.16: Arquitectura abstracta de Jadex y su motor de razonamiento. [Obtenida de Jadex UserGuide]

(ADF) (5.7.4), los eventos (5.7.5) y finalmente el razonamiento utilizado para seleccionar el plan a ejecutar (5.7.6).

5.7.1. Creencias de los agentes

La información interna de cada agente está compuesta de las creencias del agente, las cuales se almacenan en la base de creencias. Como ya se ha comentado, las creencias corresponden a la información que el agente posee del entorno, a la información estática definida previamente y a la información que recibe de otros agentes por medio del paso de mensajes. Por tanto, el sistema multi-agente está compuesto por una red de agentes con capacidades visuales que monitorizan el entorno y proporcionan información de su punto de vista local a otros agentes.

En la red de sensores visuales desarrollada, la información del entorno se proporciona en forma de las imágenes adquiridas por cada cámara dentro de la función *get_external_events*() del ciclo del interprete BDI presentado anteriormente. Una vez que la información visual es obtenida, es necesario almacenarla en las creencias de una forma consistente para permitir el intercambio de información entre los diferentes agentes.

En Jadex las creencias (beliefs) de los agentes, pueden consistir en cualquier tipo de objeto Java y son almacenados en una base de creencias (beliefbase); es decir, dentro de la base de creencias se pueden almacenar objetos Java que sean instancias de una clase previamente definida. La base de creencias almacena los hechos que el agente cree (considera verdad) y es el punto de acceso de los datos que contiene el agente. Las creencias, proporcionan una mayor abstracción que los atributos en el paradigma de orientación a objetos y representan un punto de vista unificado del conocimiento del agente. Las creencias pueden verse como la información que el agente almacena acerca del entorno y del propio proceso y son la forma de almacenar el conocimiento y los datos que dirigen la ejecución de los planes futuros. Cada creencia, dentro de la base de creencias, tiene una cadena que representa un identificador de la creencia. Estos identificadores mapean hechos (facts) que pueden ser cualquier objeto Java posible. Las creencias son fuertemente tipadas y en tipo de ejecución se comprueba que solo se almacenan objetos con el tipo adecuado.

Por encima del modelo de datos de las creencias, Jadex proporciona un lenguaje de consultas OQL[23], utilizando por tanto conceptos de las bases de datos relacionales orientadas a objetos. Es posible establecer condiciones que pueden disparar planes o crear nuevas intenciones cuando se produce algún cambio en las creencias y también es posible establecer creencias que se almacenan como expresiones y se evalúan dinámicamente bajo demanda.
Las creencias de los agentes se pueden establecer de las dos formas siguientes:

1. **Utilizando el Agent Definition File (ADF):** Son ficheros en xml que definen de forma estática las características de los agentes. Cuando un agente comienza su ejecución dentro del framework utiliza los elementos definidos en este fichero para iniciar y dirigir su ejecución.

2. **Dinámicamente en tiempo de ejecución:** Por medio de los métodos que proporciona el framework Jadex, es posible incluir o eliminar creencias de la base de creencias en tiempo de ejecución.

Las creencias de los agentes, en cada momento, determinan el proceso de deliberación del agente y los planes pueden modificar las creencias actuales mientras se están ejecutando.

[23]Object Query Language (OQL) es un lenguaje de consultas estándar para bases de datos orientadas a objetos.

Tabla 5.1: Información almacenada en las creencias de los agentes sensores para cada uno de los objetos detectados.

Elemento	Tipo de Dato	Descripción
id	String	Identificador único
sensor-id	Int	Identificador del sensor visual
timestamp	Date	Marca de tiempo de la información
global-x	Float	Valor de la coordenada X en el plano global
global-y	Float	Valor de la coordenada Y en el plano global
local-x	Float	Valor de la coordenada X en el plano local
local-y	Float	Valor de la coordenada Y en el plano local
w	Float	Anchura del objeto (píxeles)
h	Float	Altura del objeto (píxeles)
dx	Float	Velocidad en el eje x del plano local (píxeles/seg)
dy	Float	Velocidad en el eje y del plano local (píxeles/seg)
R	Matrix Float	Matriz 6x6 de covarianza la información local
R-global	matrix Float	Matriz 2x2 de covarianza de la información global
q	Float	Valor que indica la calidad de la información

Los cambios en las creencias, pueden a su vez, provocar que se disparen eventos internos que pueden dar lugar a la adopción de nuevos objetivos y a la ejecución de nuevos planes.

Dentro del sistema CSA, cada uno de los agentes sensores S_i almacena, en cada instante de tiempo t y para cada uno de los objetos detectados, la información que se muestra en la tabla 5.1 como creencias dentro de su base de creencias.

El valor de q es un indicador de la calidad de la segmentación del algoritmo de detección de objetos para ese instante determinado. Como ya se ha comentado anteriormente, la fase de segmentación es una fase crítica en las aplicaciones de seguimiento y es la fuente de muchos errores. Debido a que, en este trabajo, el seguimiento se realiza en el plano del suelo, el hecho de una mala segmentación, entendiendo por tal aquella en la cual no se detectan los pies de las personas, conlleva que se produzcan grandes errores en la estimación de la posición de los objetos (ver figura 5.6). Siendo h_{max} la máxima altura de una persona en píxeles de un sensor visual S_i y h_{cur} la altura de la persona detectada, el valor de q se calcula fácilmente como $q = 1$ si $\frac{h_{cur}}{h_{max}} \geq 1$ en otro caso $q = \frac{h_{cur}}{h_{max}}$. h_{max} es un valor específico para cada agente sensor y para el objeto de interés sobre el que se realiza el seguimiento. Es decir, si el sensor se utiliza para realizar el seguimiento de personas, h_{max} es el valor medio de la altura que suelen tener las personas. En otros escenarios, por ejemplo en el seguimiento de coches, h_{max} debe ser ajustada de acuerdo a la máxima altura esperada. Por supuesto, el valor de h_{cur} depende de la profundidad de la persona en la imagen. Es decir, si se hace seguimiento de una persona que está bastante alejada del sensor visual, h_{cur} es menor que si la persona se encuentra cerca del sensor visual.
Además, los agentes sensores deben tener las siguientes creencias:

- *Frecuencia de actualización del entorno*: Se trata de un parámetro interno del agente sensor, que específica la frecuencia de actualización en milisegundos de los objetos que se perciben en el entorno. Por tanto, consiste en la frecuencia en la que el agente percibe el entorno, es decir, la frecuencia en la que se obtienen las imágenes del sensor visual.

El agente sensor debe tener el balance adecuado entre la frecuencia de actualización del entorno y el coste computacional que se produce al obtener esta información y la ejecución de otras actividades.

- *Frecuencia de comunicación*: Los agentes sensores envían la información de los objetivos detectados a los agentes de fusión con una cierta frecuencia. Esta creencia específica la frecuencia del envío de estos mensajes. También debe existir un balance adecuado en este parámetro para evitar la congestión de la red y asegurar la comunicación correcta.

- *Frecuencia de recepción de realimentación*: Los agentes sensores, en el esquema de funcionamiento de fusión activa, deben consultar los mensajes de realimentación recibidos con cierta frecuencia. Esta creencia específica la frecuencia en el uso de los mensajes de realimentación recibidos.

- *Agente fusión*: Cada uno de los agentes sensores necesitan almacenar la información necesaria para enviar los mensajes a los agentes fusión de forma correcta. En esta creencia almacenan el nombre y la dirección del agente de fusión al que envían la información.

- *Algoritmo de detección*: Los agentes sensores pueden utilizar diferentes algoritmos de detección, en esta creencia se establece el nombre del algoritmo de detección que utilizan. El algoritmo de detección se implementa en C/C++ utilizando las interfaces definidas en la librería OpenCV y se accede a su información por medio de las pasarelas definidas en JNI.

- *Algoritmo de seguimiento*: Al igual que los algoritmos de detección, es posible utilizar diferentes algoritmos de estimación de trayectorias o seguimiento dentro del agente sensor. El valor de esta creencia específica el algoritmo de seguimiento concreto que utilizará el agente durante su ejecución.

- *Tipo de fuente*: Esta creencia indica la fuente de información que el agente sensor utilizará en su ejecución para obtener la información de los objetivos detectados. Los valores pueden ser el sensor visual (especificando el canal de la tarjeta capturadora donde se encuentra conectado), el nombre de un fichero de vídeo o una serie de imágenes de entrada.

Por otro lado, las creencias de los agentes de fusión consisten en:

- *Conocimiento de los agentes sensores*: Los agentes de fusión deben conocer las direcciones de los agentes sensores que le informan acerca del estado del entorno. En esta creencia se almacena también la información necesaria para que los agentes de fusión puedan enviar la información de realimentación a los agentes sensores, que consiste en el nombre del agente y la dirección de la plataforma donde se ejecuta.

- *Grado de consistencia*: Esta creencia, almacena la información necesaria para que el agente fusión pueda descartar la información de seguimiento recibida de los agentes sensores debido a inconsistencias entre ellas. Concretamente almacena los umbrales espacio-temporales (diferencia espacial y diferencia temporal) por los cuales la información es descartada y no utilizada para la fusión.

- *Frecuencia de fusión*: Esta creencia almacena un valor en milisegundos con la frecuencia en la que se lleva a cabo el algoritmo de fusión. Con esta frecuencia el agente de fusión comprueba las colas de entrada de cada agente sensor y en función de la información recibida lleva a cabo el algoritmo de fusión.

- *Frecuencia de envío de realimentación*: Este valor de creencia almacena el tiempo en milisegundos por el cual el agente de fusión envía la información de realimentación a los agentes sensores.

- *Tipo de fusión*: Establece si el mecanismo de fusión utilizado consiste en fusión activa o fusión pasiva.

- *Algoritmo de fusión*: Indica el nombre del algoritmo de fusión de utilizado, de esta forma es posible establecer diferentes implementaciones posibles de los algoritmos de fusión.

5.7.2. Deseos de los agentes

En Jadex, los deseos de los agentes se implementan como objetivos (Goals). El objetivo de un agente representa un determinado estado que el agente desea obtener. En la programación orientada a objetivos, los objetivos representan los estados que se tienen que alcanzar e influyen en las acciones que se llevan a cabo para alcanzar esos estados. Los objetivos representan las motivaciones concretas que influyen en el comportamiento del agente. En Jadex se pueden establecer tres distintos tipos de objetivos:

1. **Conseguir (Achieve):** Es el objetivo más básico, simplemente define el estado objetivo que se quiere alcanzar sin decir como se va a alcanzar.

2. **Mantener (Mantain):** En este tipo de objetivo, los agentes siguen el histórico del estado y ejecutan determinados planes cuando el estado no es el deseado.

3. **Llevar a cabo (Perform):** Es un tipo de objetivo que establece los planes concretos que se deben ejecutar cuando se trata de acciones determinadas en lugar de estados que se desean alcanzar.

Los objetivos se representan internamente como objetos con atributos. El estado deseado de los objetivos de tipo *conseguir* se establece poniendo condiciones en los valores de las creencias, las cuales son evaluadas para determinar si el estado deseado se ha alcanzado. Los objetivos son relevantes porque pueden restringir lo que se almacena en la estructura interna.

La semántica de la ejecución del objetivo activo es capturada en los flags BDI, que sirven de guía para la selección de los planes y ejecución.
Los agentes sensores tienen los siguientes deseos:

- *Seguimiento*: Este deseo se corresponde con la intención de realizar el seguimiento de los objetos que se ejecuta de forma continua.

- *Mirar*: Este deseo permite al agente sensor obtener la información que se observa del entorno.

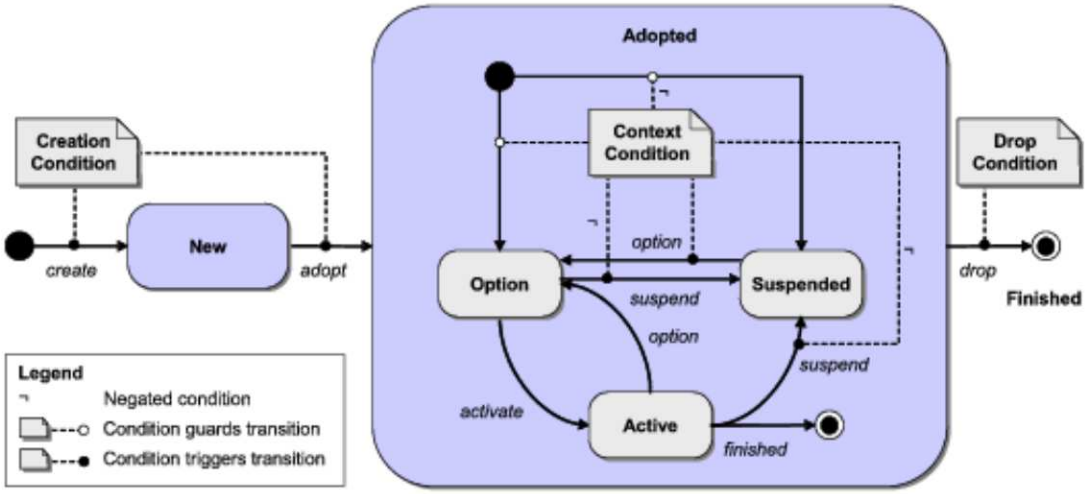

Figura 5.17: Arquitectura abstracta de Jadex. [Obtenida de Jadex UserGuide]

- *Comunicación*: Este tipo de agente también tiene el deseo de comunicarse y realiza la comunicación del vector de características y la matriz de covarianzas asociada de cada objetivo detectado. Por tanto, la granularidad de la información transmitida es pequeña si la comparamos con el envío de información de más alto nivel como la tendencia de las trayectorias. Una frecuencia de envío constante de la información de las características, en lugar de las tendencias, proporciona un mayor realismo cuando se está intentando conseguir comportamientos de tiempo real.

Por otro lado, los agentes de fusión, tienen los siguientes deseos:

- *Fusionar datos*: Los agentes de fusión tienen el deseo de fusionar la información recibida de los diferentes sensores.
- *Recibir datos*: Los agentes de fusión tiene el deseo de recibir la información de los distintos agentes sensores.
- *Informar*: Los agentes de fusión tienen el deseo de comunicarse con los agentes sensores y/o con el interfaz para informarles del resultado de la fusión.

5.7.3. Intenciones de los agentes

Dentro del modelo BDI y en concreto en Jadex, para conseguir sus objetivos un agente ejecuta planes, que consisten en código Java. Las intenciones de los agentes se representan en forma de planes concretos implementados en clases Java. Un plan, no es solo una secuencia de acciones, también puede tener subobjetivos. Otros planes se pueden ejecutar para cumplir los subobjetivos, dando lugar a una jerarquía de planes. Por tanto, la funcionalidad de los agentes se descompone en planes separados que se implementan como clases en Java. Los planes se pueden reutilizar en diferentes agentes y también se puede utilizar código Java existente.

El modelo de ejecución de Jadex está basado en eventos, todo lo que ocurre dentro de un agente Jadex se representa en forma de eventos. Los eventos de mensaje, indican la llegada de un mensaje ACL, los eventos de objetivos, indican la obtención de los mismos y los eventos internos, indican cambios en las creencias, timeouts o condiciones que se satisfacen. Los eventos pueden disparar determinados planes cuando se reciben y los planes tienen filtros para esperar determinados eventos que los disparan. El modelo de eventos permite que determinados planes que se están ejecutando esperen a que ocurran determinados objetivos (goals). Esta es una diferencia con respecto al modelo PRS inicial donde los planes se crean para un objetivo (goal) y se eliminan cuando se consiguen o fallan.

Cada plan en ejecución tiene su propia cola de eventos interna, la cual almacena los eventos del plan para procesarlos posteriormente.
Los agentes sensores tienen implementadas las siguientes intenciones:

- *Seguimiento*: Utilizando el algoritmo indicado por la creencia correspondiente, cada agente sensor S_i obtiene imágenes $I(x, y)$ con una cierta frecuencia, V_i. El proceso interno de seguimiento proporciona para cada objetivo X_{T_j}, un vector asociado de características $\hat{X}_{T_j}^{S_i}[n]$, que contiene la descripción numérica de su localización, velocidad, dimensiones, etc así como la matriz de covarianza asociada, $\hat{P}_{T_j}^{S_i}[n]$.

- *Observar*: Esta intención se encarga de obtener la imagen correspondiente al instante de tiempo t.

- *Crear una nueva pista*: La intención de crear una nueva pista y almacenarla como creencia, debido a que, según la información obtenida del entorno, se ha detectado un nuevo objeto.

- *Actualizar una pista*: La intención de actualizar una pista existente (es decir, un objeto ya detectado) y almacenar la correspondiente información en la base de creencias.

- *Eliminar una pista*: La intención de eliminar una pista previamente existente, debido al transcurso de un tiempo sin obtener información de la misma o porque desaparece del área de visión del agente sensor.

- *Informar nueva pista*: La intención de comunicar al agente fusión la información de la creación de una nueva pista en el instante de tiempo t.

- *Informar actualización pista*: La intención de comunicar al agente fusión la actualización de una pista existente en el instante de tiempo t.

- *Informar eliminación pista*: La intención de comunicar al agente fusión la eliminación de una pista existente en el instante de tiempo t.

- *Calcular calidad seguimiento*: La intención de calcular la calidad de la información obtenida por el agente sensor con respecto a la información recibida del agente fusión.

- *Corregir su estado*: La intención de corregir el estado de los objetos detectados debido a desviaciones con respecto al resultado fusionado.

Por otro lado, los agentes de fusión, tienen implementadas las siguientes intenciones:

- *Información de nueva pista*: La intención de recibir información de nuevas pistas por parte de los los agentes sensores por medio de mensajes FIPA ACL.
- *Información de actualización de pista*: La intención de recibir actualizaciones de pistas por parte de los agentes sensores por medio de mensajes FIPA ACL.
- *Información de eliminación de pista*: La intención de recibir información de pistas eliminadas por parte de los agentes sensores por medio de mensajes FIPA ACL.
- *Fusionar datos*: La intención de fusionar los datos recibidos de los diferentes agentes sensores.
- *Enviar realimentación*: La intención de enviar a los agentes sensores información de realimentación sobre el resultado fusionado por medio de mensajes FIPA ACL.
- *Enviar resultado fusión*: La intención de enviar al interfaz de usuario los mensajes con información del resultado fusionado.

5.7.4. Agent Definition File (ADF)

El Agent Definition File (ADF), es un fichero en formato XML para cada posible tipo de agente, que consiste en la definición de las creencias, deseos e intenciones del agente. Para definir un agente en Jadex, se utiliza un enfoque declarativo y procedimental, haciendo por tanto necesario crear el fichero ADF y las clases Java que se referencian en el fichero. La implementación concreta de los planes que el agente puede ejecutar tiene que estar en clases Java. Los otros conceptos: creencias, objetivos, filtros y condiciones se utilizan mediante un lenguaje que permite crear objetos Jadex de una forma declarativa. Jadex utiliza internamente JiBX[24] para instanciar el código Java necesario a partir de las declaraciones en formato XML que se proporcionan en el fichero ADF. En el fichero de definición del agente se definen las creencias y los objetivos iniciales, es decir, el estado inicial que tiene el agente en el momento de su ejecución y los mensajes que puede enviar y recibir el agente. Por tanto, los agentes se inician dentro de la plataforma Jadex por medio del fichero ADF en el cual se establecen las creencias, deseos e intenciones iniciales del agente. Además de los componentes propios del modelo BDI, también es posible almacenar otro tipo de información en el fichero de definición del agente. Los elementos que se pueden establecer dentro del fichero ADF (definidos por su schema), son los siguientes:

- *imports*: indica las clases y los paquetes que pueden ser utilizados por las expresiones dentro del fichero ADF.
- *capabilities*: con el propósito de modularizar la funcionalidad, los agentes se pueden organizar en capacidades.
- *beliefs*: la definición de las creencias iniciales.
- *goals*: la definición de los objetivos iniciales del agente.
- *plans*: los planes iniciales del agente.

[24] JiBX es una librería open source para instanciar clases Java a partir de XML. http://jibx.sourceforge.net/.

- *events*: los eventos (internos y externos) que el agente puede enviar y/o recibir.
- *expressions*: permite definir expresiones y condiciones que se pueden utilizar en los planes.
- *properties*: se utilizan para información específica de depuración y logs.
- *configuration*: sirve para indicar las creencias, objetivos y deseos iniciales cuando el agente comienza su ciclo de vida.

5.7.5. Eventos internos y eventos de mensaje

Una propiedad importante de los agentes es la capacidad de reacción ante los diferentes tipos de eventos. Jadex soporta dos tipos de eventos a nivel de aplicación, que pueden ser definidos por el desarrollador en el fichero ADF: eventos internos y eventos de mensaje.

Los eventos internos se pueden usar para indicar la existencia de conceptos internos al agente, mientras que los eventos de mensaje representan la comunicación entre dos o más agentes. Los eventos pueden ser definidos por el desarrollador en el fichero ADF (Agent Definition File). Los tipos de mensaje que un agente puede enviar o recibir deben ser descritos en el fichero ADF.

El mecanismo de paso de mensajes entre los agentes está basado en el estándar FIPA de identificación de agentes. Un identificador de agente, es un nombre que lo identifica globalmente, el cual consiste de una parte local seguida de @ y del nombre de la plataforma (nombre_agente@nombre_plataforma). Al lector que desee más información se le refiere a la especificación FIPA[25] de identificación de agentes.

Las plantillas para los mensajes se definen en la sección <*events*> del fichero ADF del agente. El atributo de dirección se utiliza para indicar si el agente quiere enviar, recibir el mensaje determinado. Los parámetros que se pueden establecer en el mensaje vienen dados por la especificación FIPA ACL Message Structure Specification [26]. En la tabla 5.2 se muestran los parámetros reservados de los mensajes FIPA, algunos de los cuales son obligatorios y otros opcionales.

Cada vez que un agente recibe un mensaje, debe procesarlo en el plan que se ha definido para ese mensaje. Para decidir que tipo de mensaje ha sido recibido, de entre todos los posibles, el framework utiliza un proceso de concordancia simple. Es decir, cada vez que se recibe un mensaje este es comparado con todos los tipos de mensaje posibles para ver con cual coincide. Además se pueden poner comprobaciones en el contenido de los mensajes para saber si cumplen determinadas condiciones.

Los posibles tipos de mensajes que envían cada uno de los agentes también se tienen que definir en el fichero ADF. Dentro del sistema CSA, el envío del mensaje se produce dentro de un plan que instancia un mensaje de un tipo determinado, establece el valor de sus parámetros y envía el mensaje usando uno de los métodos de sendMessage...()

Los posibles mensajes que puede enviar o recibir un agente se definen dentro de la etiqueta *events* del fichero ADF. A continuación se muestran los mensajes enviados y recibidos más importantes por los agentes sensores.

[25] http://www.fipa.org/specs/fipa00023/SC00023J.html
[26] http://www.fipa.org/specs/fipa00061/SC00061G.html

Tabla 5.2: Parámetros reservados de los mensajes FIPA ACL

Nombre	Clase	Significado
performative	String	Speech act del mensaje que indica que expresa la intención del contenido del mensaje
sender	AgentIdentifier	El identificador del agente que envía el mensaje
reply-to	AgentIdentifier	Identificador del agente al cual se debe responder
receivers [set]	AgentIdentifier	La dirección (al menos una) de los receptores del mensaje
content	Object	El contenido (String u objeto) del mensaje. Si el contenido es un objeto se debe especificar como se realiza el marshalling
language	String	El lenguaje en el cual el contenido del mensaje se debe codificar
encoding	String	La codificación del mensaje
ontology	String	La ontología que se debe utilizar para entender el mensaje
protocol	String	El protocolo de interacción del mensaje
reply-with	String	Para asignar una respuesta al mensaje original
in-reply-to	String	En los mensajes de respuesta contiene el contenido de reply-with
conversation-id	String	Se usa en las interacciones para identificar mensajes que pertenecen a una conversación
reply-by	Date	Contiene el instante de la última respuesta del mensaje

El mensaje de *NewTarget* (ver figura 5.18) lo envía el agente sensor al agente de fusión cuando se detecta un nuevo objetivo dentro de su campo de visión. Los datos enviados corresponden a la información definida en la tabla 5.1. Si asumimos que no se produce ninguna perdida en el seguimiento de un objetivo, este mensaje se envía una sola vez.

```
<messageevent name="NewTarget" direction="send" type="fipa">
    <parameter name="performative" class="String" direction="fixed">
        <value> SFipa.INFORM </value>
    </parameter>
    <parameter name="conversation-id" class="String" direction="fixed">
        <value> new String("NewTarget")</value>
    </parameter>
    <parameter name="language" class="String" direction="fixed">
        <value>SFipa.JAVA_XML</value>
    </parameter>
    <parameter name="content-class" class="Class" direction="fixed">
        <value>Target.class</value>
    </parameter>
</messageevent>
```

Figura 5.18: Definición del tipo de los mensajes enviados por el agente sensor para informar de la detección de un nuevo objetivo.

A continuación, la información nueva sobre un objetivo ya detectado se envía con la frecuencia de envío correspondiente utilizando el mensaje de la figura 5.19.

```
<messageevent name="UpdateTarget" direction="send" type="fipa">
    <parameter name="performative" class="String" direction="fixed">
        <value>SFipa.INFORM</value>
    </parameter>
    <parameter name="conversation-id" class="String" direction="fixed">
        <value>new String("UpdateTarget")</value>
    </parameter>
    <parametername="language" class="String" direction="fixed">
        <value>SFipa.JAVA_XML</value>
    </parameter>
    <parametername="content- class" class="Class" direction="fixed">
        <value>UpdateTarget.classs</value>
    </parameter>
</messageevent>
```

Figura 5.19: Definición del tipo de los mensajes enviados por el agente sensor para informar de la actualización de un nuevo objetivo.

Finalmente, si el objetivo se ha dejado de detectar durante los fotogramas establecidos, se envía el mensaje *DeleteTarget* (ver figura 5.20) informando al agente de fusión de la desaparición del objetivo.

```
<messageevent name="DeleteTarget" direction="send" type="fipa">
    <parametername="performative" class="String" direction="fixed">
        <value>SFipa.INFORM</value>
    </parameter>
    <parametername="conversation-id" class="String" direction="fixed">
        <value>newString("DeleteTarget")</value>
    </parameter>
    <parametername="language" class="String" direction="fixed">
        <value>SFipa.JAVA_XML</value>
    </parameter>
    <parametername="content-class" class="Class" direction="fixed">
        <value>DeleteTarget.classs</value>
    </parameter>
</messageevent>
```

Figura 5.20: Definición del tipo de los mensajes enviados por el agente sensor para informar de la desaparición de un objetivo.

En el marco de trabajo de la fusión activa, el agente de fusión envía con cierta frecuencia el mensaje de realimentación *FeedBackTargetInfo* (ver figura 5.21), que contiene la información fusionada de los objetivos de forma global. En el esquema de la fusión activa, el agente sensor utiliza la información de este mensaje para poder corregir sus desviaciones.

```
<messageevent name="FeedBackTargetInfo" direction="receive" type="fipa">
    <parametername="performative" class="String" direction="fixed">
        <value>SFipa.INFORM</value>
    </parameter>
    <parameter name="conversation-id" class="String" direction="fixed">
        <value>newString("FeedbackTargetInfo")</value>
    </parameter>
    <parameter name="language" class="String" direction="fixed">
        <value>SFipa.JAVA_XML</value>
    </parameter>
    <parameter name="content-class" class="Class" direction="fixed">
        <value>FeedbackTarget.classs</value>
    </parameter>
</messageevent>
```

Figura 5.21: Definición del tipo de los mensajes recibidos por el agente sensor con información de realimentación (feedback) del agente de fusión.

Si se desea utilizar coaliciones dinámicas dentro del sistema CSA, es necesario indicar los mensajes correspondientes al protocolo de coaliciones definido en 5.3.1. Por tanto, el agente sensor debe poder recibir el mensaje *CallForCoalition* (ver figura 5.22) por parte del agente fusión, con la intención de crear una coalición.

```
<messageevent name="CallForCoalition" direction="receive" type="fipa">
    <parameter name="performative" class="String" direction="fixed">
        <value>SFipa.INFORM</value>
    </parameter>
    <parameter name="conversation-id" class="String" direction="fixed">
        <value>newString("CallForCoalition")</value>
    </parameter>
    <parameter name="language" class="String" direction="fixed">
        <value>SFipa.JAVA_XML</value>
    </parameter>
    <parameter name="content-class" class="Class" direction="fixed">
        <value>CallForCoalition.classs</value>
    </parameter>
</messageevent>
```

Figura 5.22: Definición del tipo de los mensajes recibidos por los agentes sensores donde se les solicita crear un equipo para un objetivo determinado.

A su vez, el agente sensor, si acepta la petición de formación de coalición envía un mensaje de tipo *AcceptCoalition* (ver figura 5.23).

```
<messageevent name="AcceptCoalition" direction="send" type="fipa">
    <parameter name="performative" class="String" direction="fixed">
        <value>SFipa.INFORM</value>
    </parameter>
    <parameter name="conversation-id" class="String" direction="fixed">
        <value>newString("AcceptCoalition")</value>
    </parameter>
    <parameter name="language" class="String" direction="fixed">
        <value>SFipa.JAVA_XML</value>
    </parameter>
    <parameter name="content-class" class="Class" direction="fixed">
        <value>AcceptCoalition.classs</value>
    </parameter>
</messageevent>
```

Figura 5.23: Definición del tipo de los mensajes enviados por los agentes sensores indicando que aceptan formar una coalición para un objetivo determinado.

En caso de no aceptar la formación del equipo, el tipo de mensaje enviado para informar al agente de fusión es *RejectCoalition* (ver figura 5.24).

```
<messageevent name="RejectCoalition" direction="send" type="fipa">
    <parameter name="performative" class="String" direction="fixed">
        <value>SFipa.INFORM</value>
    </parameter>
    <parameter name="conversation-id" class="String" direction="fixed">
        <value>newString("RejectCoalition")</value>
    </parameter>
    <parameter name="language" class="String" direction="fixed">
        <value>SFipa.JAVA_XML</value>
    </parameter>
    <parameter name="content-class" class="Class" direction="fixed">
        <value>RejectCoalition.classs</value>
    </parameter>
</messageevent>
```

Figura 5.24: Definición del tipo de los los mensajes enviados por los agentes sensor para informar del rechazo de formar una coalición.

En el caso de los agentes de fusión, los mensajes de entrada y salida se definen de forma similar. A modo de ejemplo en la figura 5.25 se muestra el mensaje de entrada que recibe el agente de fusión cuando se ha detectado un nuevo objetivo en un agente sensor.

Dentro del sistema CSA, los mensajes de entrada se utilizan para disparar planes que se encarguen de gestionar la información recibida. Además también se pueden generar eventos internos en el agente (a diferencia de los mensajes que son eventos externos) para disparar determinados planes. La creación de estos eventos internos se puede establecer en tiempo de ejecución por medio del código Java de los planes. A modo de ejemplo, en la figura 5.26

```
<messageevent name="NewTarget" direction="send_receive" type="fipa">
    <parameter name="performative" class="String" direction="fixed">
        <value>SFipa.INFORM</value>
    </parameter>
    <parameter name="conversation-id" class="String" direction="fixed">
        <value>new String("NewTarget")</value>
    </parameter>
    <parameter name="language" class="String" direction="fixed">
        <value>SFipa.JAVA_XML</value>
    </parameter>
    <parameter name="content-class" class="Class" direction="fixed">
        <value>Target.classs</value>
    </parameter>
</messageevent>
```

Figura 5.25: Definición del tipo de mensaje recibido en el agente de fusión indicando la detección de un nuevo objetivo por parte del agente sensor.

se describe la parte del fichero ADF del agente fusión que define la ejecución de planes concretos ante la recepción de determinados mensajes, esta sección se define dentro de la etiqueta <*plans*> del fichero ADF de cada agente.

```
<plans>
<plan name="FusionNewTargetPlan">
    <body>new FusionNewTargetPlan ()</body>
    <trigger><messageevent ref="NewTarget" /></trigger>
</plan>
<plan name="FusionUpdateTargetPlan">
    <body>new FusionUpdateTargetPlan ()</body>
    <trigger><messageevent ref="UpdateTarget" /></trigger>
</plan>
<plan name="DeleteTargetPlan">
    <body>new FusionDeleteTargetPlan ()</body>
    <trigger><messageevent ref="DeleteTarget" /></trigger>
</plan>
<plan name="FusionDataFusionPlan">
    <body>new FusionDataFusionPlan ()</body>
    <trigger><internalevent ref="DataFusion" /></trigger>
</plan>
</plans>
```

Figura 5.26: Definición de los planes que se disparan en los agentes de fusión en función de los eventos recibidos.

5.7.6. Selección del plan a ejecutar

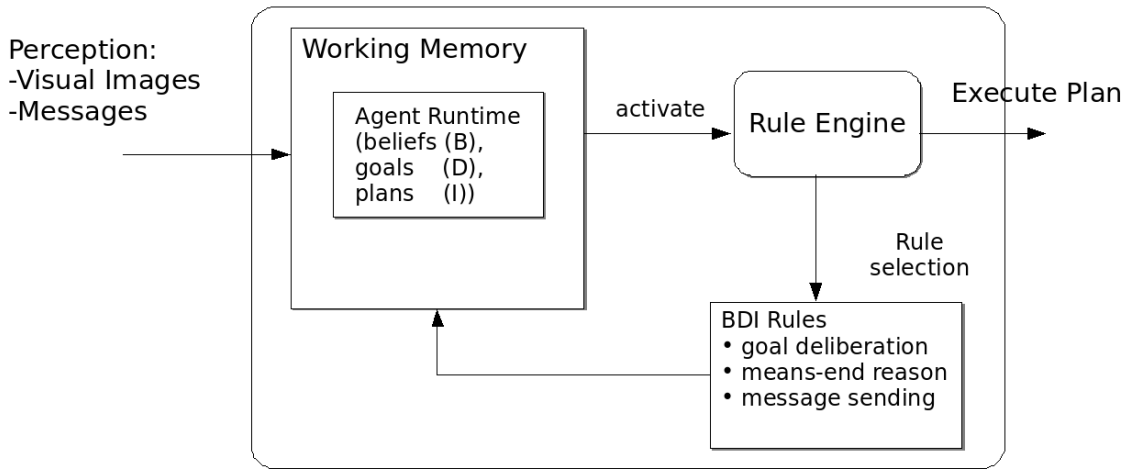

Figura 5.27: Interprete abstracto de razonamiento de Jadex.

En Jadex, el motor de razonamiento utilizado para seleccionar el siguiente plan a ejecutar por cada uno de los agentes está basado en un motor de reglas. En la figura 5.27 se muestra un esquema de alto nivel de la selección de los planes por parte del motor de reglas. Existen varios motores de reglas disponibles, como por ejemplo: Jess, Drools, Jamocha, Jrules and CLIPS. Jadex tiene su propio motor de reglas y está basado en el algoritmo Rete (Forgy, 1991), para realizar la concordancia de reglas de una forma eficiente. Sin embargo, debido a que las reglas Jadex tienen claramente separado la representación de la regla y del estado, es posible utilizar otros algoritmos como TREAT (Miranker, 1987).

En los sistemas expertos, o en aquellos sistemas en los que hay que aplicar un conjunto de reglas (base de conocimiento o knowledge base) a partir de una serie de hechos (memoria de trabajo o working knowledge), la base de conocimiento suele cambiar con menos frecuencia que los hechos. Cuando se está llevando a cabo el proceso de razonamiento, una regla de la base de conocimiento debe dispararse de acuerdo al conjunto de hechos (facts) existentes. Las implementaciones menos eficientes de los motores de reglas comprueban la parte izquierda (LHS) de todas las posibles reglas y ejecutan la parte derecha (RHS) de las reglas que cumplen la condición. Este algoritmo es muy ineficiente, ya que suponiendo que el número de reglas sea R, y F el número de hechos y P el número medio de patrones por cada regla, este algoritmo tiene una complejidad de $O(RF^P)$, lo que implica que escala bastante mal conforme el número de patrones por regla crece.

En el algoritmo Rete (ver figura 5.28) la ineficiencia de comprobar todas las reglas se soluciona recordando los resultados previos del proceso de razonamiento. Por tanto, solo los nuevos hechos (facts) se comprueban contra las reglas. Esta modificación permite que el algoritmo reduzca su complejidad a $O(RFP)$, siendo lineal con el tamaño de la base de hechos. Rete construye un árbol a partir de las reglas, similar a una maquina de estados. Cada regla es un grafo acíclico dirigido de nodos con el nodo Rete como inicial y un nodo terminal como final. El grafo se compone de nodos alfa y beta. Los nodos alfa realizan restricciones de comprobación en los objetos y los nodos betas unen dos nodos de entrada.

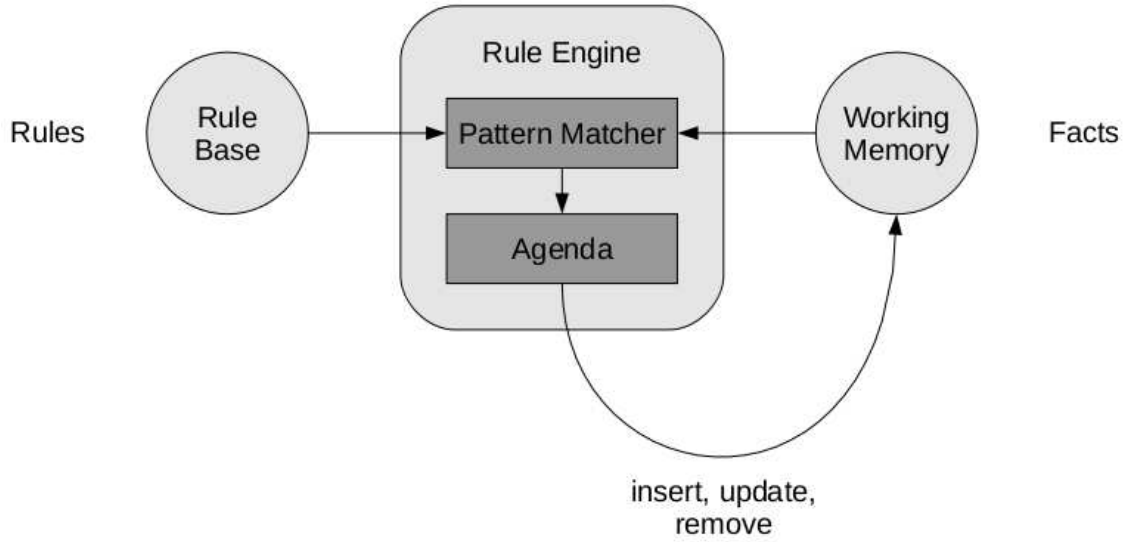

Figura 5.28: Motor de producción de reglas de Jadex usando el algoritmo Rete.

Los hechos (facts) entran en el árbol por los nodos superiores como parámetros de las reglas y van bajando por el árbol si coinciden con las condiciones hasta que llegan a los nodos hoja (consecuencias de las reglas). El motor de reglas estándar de encadenamiento hacia adelante (forward chining) (ver figure 5.28), utiliza la concordancia de patrones para buscar activaciones de forma incremental, después encola las activaciones en la agenda y finalmente selecciona una activación y ejecuta su acción.

La base sobre la que trabajan los motores de reglas son los tipos de objeto, los cuales consisten en la representación del estado interno en cada momento. En Jadex, estos tipos de objetos se pueden definir usando objetos Java (Java Beans[27]) o tripletas Objeto-Atributo-Valor (OAV); donde *Objeto* se refiere al id del estado del objeto, *Atributo* indica el nombre del parámetro y *Valor* es el valor concreto del objeto Java. Tripletas con el mismo valor corresponden al mismo objeto.

Recientemente se han realizado algunas mejoras al algoritmo Rete estándar (Berstel, 2002) (Walzer et al., 2008) para extenderlo a eventos en tiempo real, que pueden resultar interesante utilizar en el entorno de las redes de sensores visuales.

[27] Un Java Bean es una clase Java que cumple las condiciones de: tener un constructor sin argumentos, métodos get() y set() para todas las propiedades y debe ser serializable.

6

Experimentación y validación

It's not what you know, it's what you can prove.
Denzel Washington (Alonzo Harris) en Training Day.

6.1. Introducción

En este capítulo, se presentan los resultados experimentales que se han llevado a cabo para validar el sistema propuesto, con el objetivo final de demostrar su mejora y aportación frente a los métodos multi-cámara clásicos. Las principales características que se pretende demostrar con los experimentos, son:

- Robustez: Como se comporta el sistema ante la presencia de fallos. Para ello se han hecho pruebas utilizado el conjunto de datos de APIDIS e incrementando el número de sensores visuales utilizado.
- Estabilidad: El comportamiento del sistema a lo largo del tiempo utilizando secuencias de varios fotogramas.
- Capacidad de corrección: Para ello se han realizado pruebas utilizando el esquema de fusión activa con el objetivo de corregir los errores obtenidos en el procesado de imágenes.

Se han realizado experimentos utilizando sensores visuales propios en entornos exteriores e interiores y utilizando conjuntos de grabaciones de imágenes almacenadas en ficheros de vídeo disponibles para la comunidad investigadora de visión artificial. A continuación se describen los resultados de las pruebas y los experimentos realizados.

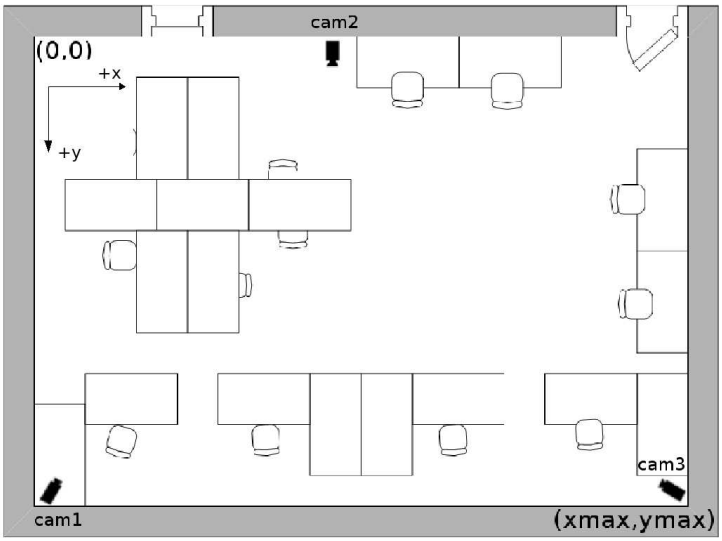

Figura 6.1: Esquema de la colocación de las cámaras dentro del laboratorio de investigación.

6.2. Experimentación preliminar con el dataset PETS 2006

La primera prueba de funcionamiento del sistema (Castanedo et al., 2007a) (Castanedo et al., 2007b) se realizó con el conjunto de datos de visión artificial del *Performance Evaluation of Tracking and Surveillance* (PETS 2006) (PETS2006, [Online 03/2010]). Los conjuntos de datos del PETS son muy conocidos dentro de la comunidad de visión artificial y se han creado anualmente de forma regular desde el año 2001; tienen como finalidad permitir una evaluación objetiva de los sistemas de vigilancia implementados.

La resolución de las imágenes de todas las secuencias del conjunto PETS 2006 son PAL estándar de 768x576 píxeles, grabadas a 25 frames por segundo y comprimidas como secuencias de imágenes JPEG. Las imágenes grabadas se han obtenido con 4 cámaras distintas, desde diferentes puntos de vista de la misma escena (ver figura 6.2). La cámara 1 y 3 son de tipo Canon MV-1 1xCCD w/progresiva y la cámara 2 y 4 son de tipo Sony DCR-PC1000E 3xCMOS. Las grabaciones de los escenarios se llevaron a cabo en una estación de trenes pública. Los datos de calibración de cada cámara individual se han obtenido por medio de los puntos de localización de los mosaicos del suelo (ver figura 6.3).

Para las pruebas realizadas se han utilizado los 200 primeros fotogramas de las cámaras 1, 3 y 4 y se han descartado los de la cámara 2 debido a que tienen una calidad de imagen bastante pobre por la localización de la cámara.

La prueba realizada ha consistido en observar el comportamiento del seguimiento en el sistema para los fotogramas de entrada de cada una de las cámaras, es decir la calidad en el seguimiento de la persona detectada. En la figura 6.4 se muestran las trayectorias de cada una de las cámaras en los fotogramas 91, 140 y 199. La cámara 1 (imagen de la izquierda) muestra un seguimiento bastante inestable, la cámara 3 (imagen en el centro) presenta los mejores resultados de seguimiento, debido a su localización y finalmente la cámara 4 (imagen de la derecha) presenta un seguimiento estable pero muy impreciso debido al error introducido por la sombra de la persona. En la figura 6.5 se muestran las imágenes

Figura 6.2: Imágenes de ejemplo de entrada de cada una de las 4 cámaras del conjunto de datos de PETS 2006.

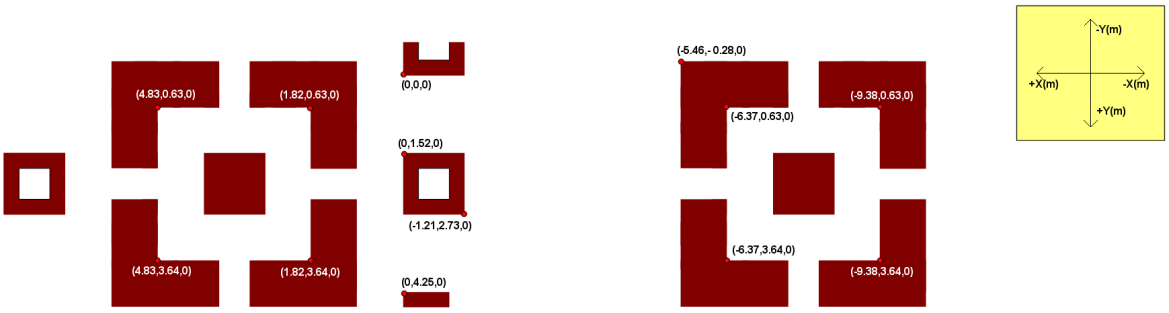

Figura 6.3: Puntos de calibración en coordenadas del mundo utilizados para realizar la calibración en PETS 2006. Se han utilizado como referencia los mosaicos del suelo.

con los píxeles en blanco de los blobs detectados para los anteriores fotogramas de las tres cámaras. Las trayectorias obtenidas en el plano del suelo, de cada uno de los sensores en coordenadas locales, se transformaron a coordenadas globales utilizando los valores de calibración obtenidos previamente. En las figuras 6.6, 6.7, 6.8 se muestran las posiciones obtenidas en coordenadas globales para la cámara 1, 3 y 4 respectivamente. En los gráficos se observa claramente como las posiciones de la cámara 1 (ver figura 6.6) están bastante dispersas, debido a los problemas de ruido existentes en la segmentación de la persona por las imágenes obtenidas por esa cámara. Por otro lado, las posiciones de la cámara 3 (ver figura 6.7) presentan un seguimiento bastante preciso gracias a la buena localización de la misma. Finalmente, las posiciones de la cámara 4 (ver figura 6.8) muestran un seguimiento estable, pero desplazado de la realidad, debido a los problemas de sombras y de mala segmentación que conllevan a que el sistema tenga un salto brusco al conectarse con varias personas (ver imagen abajo a la derecha de la figura 6.5).

Figura 6.4: Trayectorias obtenidas por las imágenes de cada uno de los sensores visuales (cámara 1, cámara 2 y cámara 3) para los fotogramas 91, 140 y 199 del conjunto de datos PETS 2006.

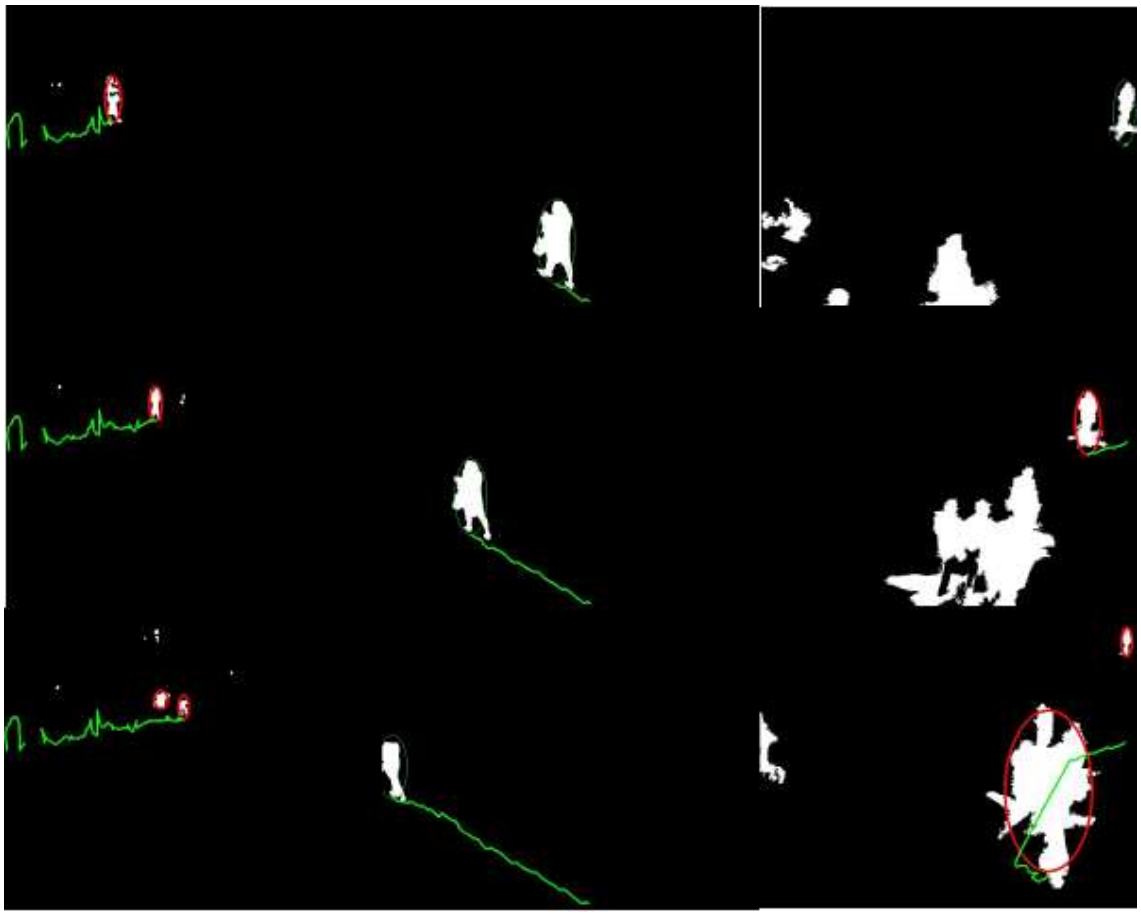

Figura 6.5: Píxeles de las imágenes que han sido detectados como blobs y las trayectorias en el plano del suelo para cada una de las cámaras en los fotogramas 91, 140 y 199 del conjunto de datos PETS 2006.

Estas pruebas iniciales proporcionan un ejemplo de la motivación que existe para mejorar el proceso de seguimiento visual en un sistema multi-cámara, por medio de técnicas de fusión de datos adecuadas para obtener un resultado coherente, y que sean capaces de corregir las desviaciones individuales de los sensores, utilizando el resultado fusionado. Además, el sistema debería poder ejecutarse de forma distribuida y proporcionar la flexibilidad suficiente para añadir más sensores visuales y comunicar la información dentro del sistema. La aproximación seguida para mejorar la información de seguimiento proporcionada por cada sensor, no consiste en mejorar de forma individual cada proceso (por ejemplo eliminado las sombras o utilizando otros modelos de detección de fondo); sino en detectar los datos inconsistentes entre sí, por medio del uso de múltiples fuentes de información redundantes. Las pruebas realizadas con este conjunto de datos, muestran las principales causas de los errores (ruido, sombras, oclusiones) que existen, al obtener la información en los agentes sensores y su dificultad para fusionar la información adecuadamente.

En una red de sensores visuales la integración de los resultados proporcionados por múltiples sensores visuales puede ser más precisa que la que se obtiene con un solo sensor. Sin embargo, la fusión de los datos debe aplicarse con cuidado, debido a que aunque múltiples

sensores pueden proporcionar más información acerca de un objeto, esta información puede resultar inconsistente entre sí.

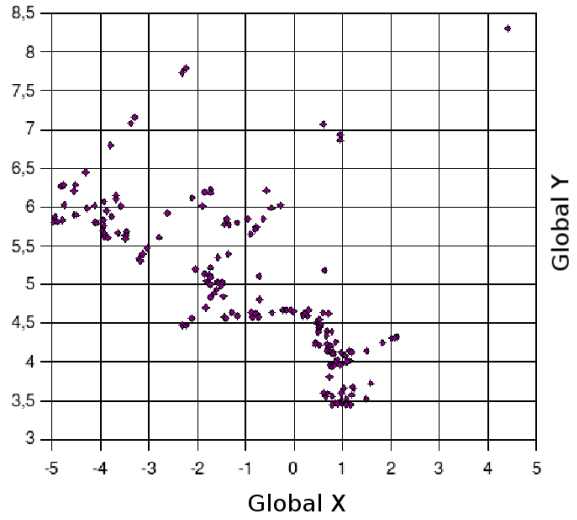

Figura 6.6: Posiciones de seguimiento (en coordenadas globales) en el plano del suelo de la cámara 1 para los 200 primeros fotogramas de PETS 2006. Los valores de los ejes están expresados en metros respecto del punto (0,0).

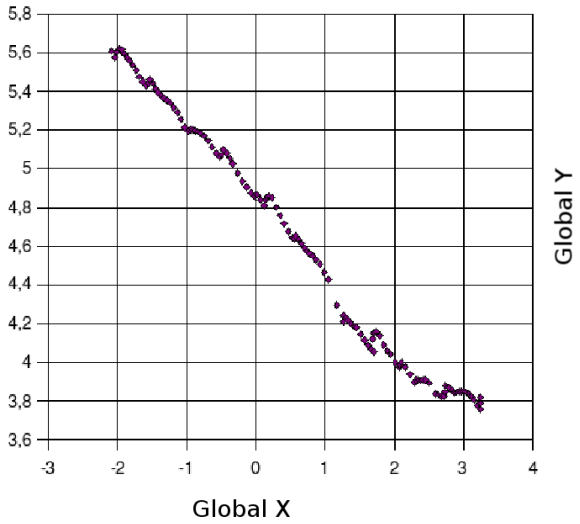

Figura 6.7: Posiciones de seguimiento (en coordenadas globales) en el plano del suelo de la cámara 3 para los 200 primeros fotogramas de PETS 2006. Los valores de los ejes están expresados en metros respecto del punto (0,0).

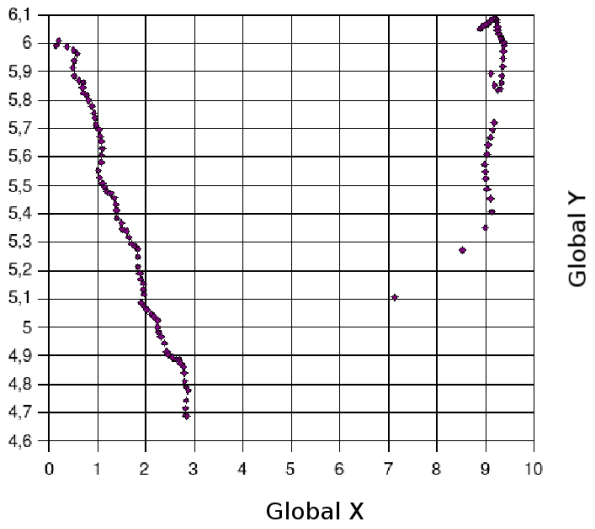

Figura 6.8: Posiciones de seguimiento (en coordenadas globales) en el plano del suelo de la cámara 4 para los 200 primeros fotogramas de PETS 2006. Los valores de los ejes están expresados en metros respecto del punto (0,0).

6.3. Experimentación realizada en entornos exteriores

Con estas pruebas (Castanedo et al., 2010) se demuestra, como una vez establecidas las coaliciones de agentes, se garantiza que los objetos de interés son seguidos con éxito a lo largo de todo el campo visual, asegurando la continuidad de las pistas y las transiciones suaves entre las áreas. En concreto, se realiza el proceso de fusión para combinar los datos de seguimiento de los objetos de interés, mientras la coalición se encuentra activa, utilizando sensores desplegados con áreas parcialmente solapadas. La coalición está formada por dos agentes sensores y un agente de fusión. Los agentes sensores se han desplegado con áreas visuales solapadas, lo que proporciona una redundancia en el área monitorizada por los sensores. Las fases que se ejecutan en el proceso de fusión, una vez que la coalición está establecida, son las siguientes:

1. Se lleva a cabo el seguimiento de los objetos de forma local en cada agente sensor S_i y el resultado, expresado en coordenadas globales, se envía al agente de fusión F_j.

2. El agente de fusión F_j ejecuta los algoritmos de fusión para realizar asociación pista a pista, comprobación de consistencia, registrado inter-sensor y combinación.

3. El resultado del proceso de fusión sirve para estimar los sesgos entre los sensores por medio del algoritmo de registrado. Esta información se envía de nuevo a los agentes sensores (por medio de la intención *SendFeedbackInfo*()) para asegurar la coherencia global de la información.

La primera intención llevada a cabo por cada uno de los agentes sensores es la intención de seguimiento. Cada agente sensor es capaz de medir la localización de los objetos en movimiento dentro de su campo de visión con respecto a un sistema de coordenadas común.

Una vez que la coalición está formada, se utilizan técnicas de fusión de datos para combinar la información local de cada uno de los sensores.

Cada uno de los agentes sensores envían mensajes con la información temporal, el vector de seguimiento que contiene las descripciones numéricas de los atributos y el estado (localización, velocidad, dimensiones) y la matriz de covarianza asociada: $\left\langle t[n], \hat{x}_l^i[n], R_l^i[n] \right\rangle$.

A continuación se describe el proceso de fusión realizado en el agente de fusión para estas pruebas específicas. En primer lugar, se realiza una comprobación de la consistencia, a continuación se realiza un registrado multi-cámara y finalmente se fusionan las pistas consistentes.

6.3.1. Comprobación de la consistencia

La primera fase del algoritmo de fusión es la comprobación de la consistencia entre las pistas. La comprobación de consistencia se realiza sobre los pares de pistas recibidas por los agentes sensores (S_i, S_j) utilizando la distancia de Mahanalobis:

$$MD_{S_i,S_j} = \left(\hat{x}_l^i[n] - \hat{x}_l^j[n]\right)^t \left(R_l^i[n] + R_l^j[n]\right)^{-1} \left(\hat{x}_l^i[n] - \hat{x}_l^j[n]\right) \leq \lambda \qquad (6.1)$$

Si la distancia de Mahalanobis excede el umbral λ, el par de pistas es marcado como inconsistente indicando que una de las pistas debe ser descartada del proceso de fusión.

6.3.2. Registrado multi-cámara

La transformación de coordenadas locales a globales se lleva a cabo como resultado de un proceso de calibración previo, de forma que la salida se expresa en un marco de coordenadas cartesianas comunes. En estas pruebas se ha utilizado un refinamiento on-line adicional para eliminar posibles errores sistemáticos entre las fuentes de información y garantizar la estabilidad del proceso de fusión. Este proceso de alineamiento multi-sensor dinámico se conoce como registrado multi-sensor. La diferencia entre el proceso de calibración y el de registrado multi-sensor, es que el primero se lleva a cabo utilizando datos estáticos (normalmente marcas fijas en el plano) mientras que el proceso de registrado utiliza las trayectorias de los objetos que se van a fusionar.

Sea $\hat{x}_j^i[n]$, $R_j^i[m]$ el resultado en coordenadas globales para el agente sensor i, en el fotograma m y sobre el objetivo j; en el proceso de registrado on-line se calculan los vectores estimados de sesgo, $\hat{b}^i[n]$, para refinar el alineamiento con respecto al resultado fusionado $\hat{x}_j^F[n]$, $R_j^F[n]$.

Se ha aplicado una técnica de batch de registrado (Besada et al., 2004) para estimar y eliminar los errores sistemáticos de forma on-line y se almacena en las creencias del agente de fusión, la información enviada por los agentes sensores que corresponde a los mismos objetivos. La corrección de las transformación se ha estimado utilizando un enfoque de mínimos cuadrados: calculando el vector de sesgo, que aplicado a la salida de cada cámara, minimiza las magnitudes ponderadas de la diferencia del vector de medidas expresado en coordenadas globales (resultado fusionado).

Sea $\Delta x_c^i[m]$ el vector de diferencia, entre las estimaciones proporcionadas por el agente sensor S_i y la pista central proporcionada por el agente de fusión después de aplicar la corrección del sesgo $f_i(.)$ y sea está diferencia ponderada con $R_c^i[m]$, (la matriz de error de covarianza del vector de la diferencia entre las medidas proporcionadas por el agente sensor

S_i y el agente fusión F_j), la función objetivo que se debe minimizar, es la raíz cuadrada del módulo del vector de diferencias corregido, ponderado con las respectivas matrices de covarianza, dando lugar a las siguientes componentes del vector: $(\Delta x_c^i)^T (R_c^i)^{-1} \Delta x_c^i$.

El vector de corrección, $\hat{b}_i$, contiene: la corrección absoluta en la localización $[b_x \ b_y]^t$, los parámetros de la homografía entre la proyección 2D de la cámara y las coordenadas de fusión (asumiendo movimiento en el plano del suelo), $[a_{xx} \ a_{xy} \ a_{yx} \ a_{yy}]$, y la corrección local del reloj, b_t. Con la información previa, el vector de corrección $\hat{b}$ tiene una relación lineal con la localización fusionada, $\left[\ x^F[k] \quad y^F[k] \ \right]$ y la estimación local en el sensor S_i, $\left[\ x^i[k] \quad y^i[k] \ \right]$, considerando también la velocidad $\left[\ v_x^F[k] \quad v_y^F[k] \ \right]$ para dar lugar a la medida de diferencia:

$$\begin{aligned} \begin{bmatrix} \Delta x_c^i[m] \\ \Delta y_c^i[m] \end{bmatrix} &= \begin{bmatrix} x^F[m] \\ y^F[m] \end{bmatrix} - \begin{bmatrix} 1 & 0 & x_l^i[m] & y_l^i[m] & 0 & 0 & v_x^F[m] \\ 0 & 1 & 0 & 0 & x_l^i[m] & y_l^i[m] & v_y^F[m] \end{bmatrix} \begin{bmatrix} b_x \\ b_y \\ a_{xx} \\ a_{xy} \\ a_{yx} \\ a_{yy} \\ b_t \end{bmatrix} \\ &= \overline{x}^F[m] - F^i[m]\overline{b} \end{aligned} \tag{6.2}$$

Teniendo en cuenta un conjunto de N medidas de diferencia, $\Delta x_i^c[m], m = 1, \ldots, N$ correspondientes al intervalo de tiempo de la transición solapada y teniendo en cuenta las matrices de covarianza de las diferencias estimadas, $R_i[k]$, la solución de mínimos cuadrados (Herrero et al., 2007) es:

$$\left(\hat{b}\right)_*^i = \left(\sum_{m=1}^{N} (F^i[m])^t (R^i[m])^{-1} (F^i[m]) \right)^{-1} \left(\sum_{m=1}^{N} (F^i[m])^t (R^i)^{-1} \overline{x}^F[m] \right) \tag{6.3}$$

La corrección previa se aplica a todas las pistas enviadas por cada uno de los agentes sensor S_i.

6.3.3. Fusión entre pistas consistentes

Una vez se han seleccionado las pistas consistentes, la fusión se lleva a cabo ponderando la fiabilidad de cada pista. Para ello se ha utilizado un enfoque federado simple (Hall & Llinas, 2001), basado en ponderar cada fuente de información de acuerdo a su matriz de error de covarianza y modificando la ponderación en función de una puntuación adicional que establece el nivel de confianza asignado al seguimiento de ese agente sensor. Por tanto, para cada objeto j, la combinación se obtiene por:

$$\left(R_j^F\right)^{-1} = \sum_{k \in C} \left(\alpha_j^i R_j^i\right)^{-1}; \ \hat{x}_j^F = R_j^F \sum_{k \in C} \left(\alpha_j^i R_j^i\right)^{-1} \hat{x}_j^i \tag{6.4}$$

.

El nivel de confianza de cada cámara consistente y para cada objetivo está basado en la inversa de la covarianza de cada sensor y obejtivo y multiplicado por la puntuación de la función

heurística α_j^i. La función heurística $\alpha_j^i \in [1, \text{inf})$ es un escalar que caracteriza el rendimiento del sensor el cual puede estar basado en una combinación de métricas de la calidad del seguimiento (combinación de color, consistencia de forma, estabilidad del movimiento, etc).

6.3.4. Descripción del escenario y resultados

El escenario analizado con estas pruebas (ver figura 6.9) es una escena exterior, donde las vídeo-cámaras tienen áreas de visión solapadas y cubren el camino que siguen los peatones. Los dos agentes sensores y el agente de fusión, establecen una coalición con el objetivo de seguir al mismo objeto de forma cooperativa. Las regiones solapadas se muestran en la figura 6.10 y las líneas de ground-truth para identificar el camino de los peatones se muestra en la figura 6.11. Esta configuración sirve como ejemplo para llevar a cabo pruebas en las cuales los agentes sensores proporcionan diferentes vistas del mismo escenario y sus pistas locales se fusionan en una representación global.

Por tanto, los dos agentes sensores monitorizan la misma escena y aplican la intención de *SendTargetInfo*() para enviar la información de las pistas. Estos dos agentes sensores, utilizan distintas tecnologías para adquirir las imágenes, utilizando uno la tecnología analógica y estando conectado a una tarjeta capturadora y basándose el otro en la tecnología firewire (ver tabla 6.1).

Antes de llevar a cabo la ejecución de las pruebas, se ha realizado un proceso de calibración en cada una de las escenas. Por tanto, una vez aplicada la calibración, cada agente sensor lleva a cabo el seguimiento de los objetos dentro de su campo de visión respecto a un sistema de referencias común. Se ha utilizado el sistema Global Position System (GPS) para representar las localizaciones del objeto, utilizando un equipo portatil (GarminTM GPS-18 USB) para obtener las medidas. A continuación se han obtenido las correspondencias entre la imagen (en píxeles) y el sistema de referencia común (posición en coordenadas GPS). El área de las regiones solapadas permite a los agentes sensores seguir a los objetivos de forma conjunta. Cuando el agente de la derecha (agente sensor digital) detecta una persona, calcula su tamaño, su localización y su velocidad. Utilizando la información recibida el agente de fusión puede alinear y corregir las desviaciones en las pistas del otro agente de la coalición.

Se han analizado 10 vídeos de personas andando a diferentes velocidades de derecha a izquierda en las dos escenas. En la figura 6.12, se muestran las posiciones de las personas en coordenadas de la imagen locales (sin aplicar la calibración) para el primero de los vídeos grabados, cada punto se encuentra dentro de la región de calibración descrita por las marcas de calibración a cada lado del camino (asteriscos). Por medio de la calibración es posible realizar el mapeo de las coordenadas locales a las coordenadas globales. La figura 6.13 muestra el resultado de la transformación en coordenadas GPS (latitud, longitud y altitud) para las dos pistas. En realidad, están expresadas como desplazamientos relativos, en miles de minutos, del punto de referencia: Norte 40 32 min, Oeste 4 0 min. A continuación las coordenadas geodésicas se proyectan directamente a coordenadas cartesianas utilizando como referencia el punto tangente (coordenada 0,0). La figura 6.14 muestra el efecto de la inicialización de la pista del agente digital con el agente analógico. La figura 6.15 muestra el resultado fusionado, realizado por el agente de fusión después del alineamiento, donde se observa que la transición es más suave que un salto directo entre las dos pistas. Además, la corrección de las pistas del agente analógico proporciona un resultado fusionado más coherente. La figura 6.16 muestra el resultado de todos los vídeos después de la fusión.

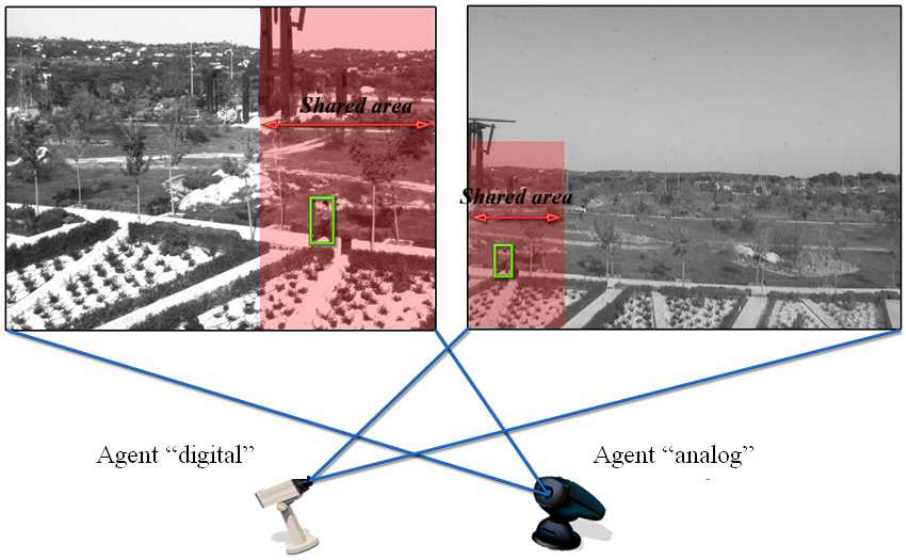

Figura 6.9: Escenario de las pruebas exteriores realizadas, donde se observa el área solapada.

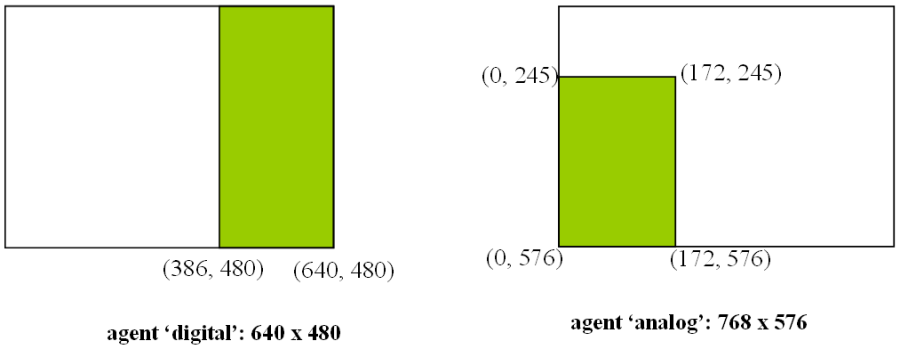

Figura 6.10: Tamaño de las áreas solapadas dentro de cada una de las imágenes de cada agente sensor.

Tabla 6.1: Modelos de cámaras utilizadas para las pruebas realizadas en entornos exteriores.

	Resolución	Fotogramas/sg
Sony EVI-D100P + Data Translation 3130 Frame Grabber	768 × 576	24
Basler A312fc/c + Matrox Meteor2 - 1394	640 × 480	30

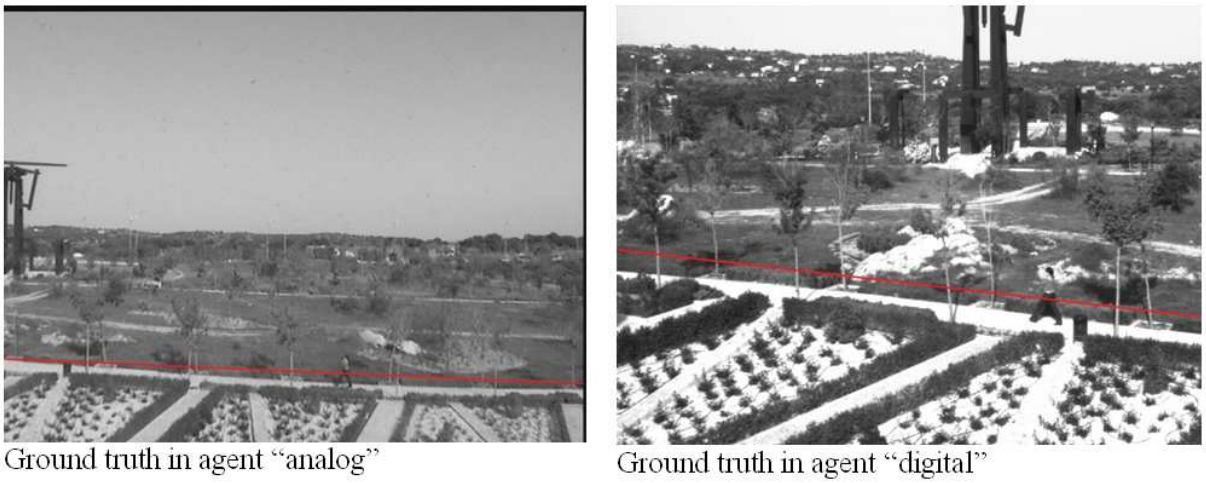

Figura 6.11: Lineas de ground-truth de los dos agentes sensores involucrados en el seguimiento (analógico a la izquierda y digital a la derecha).

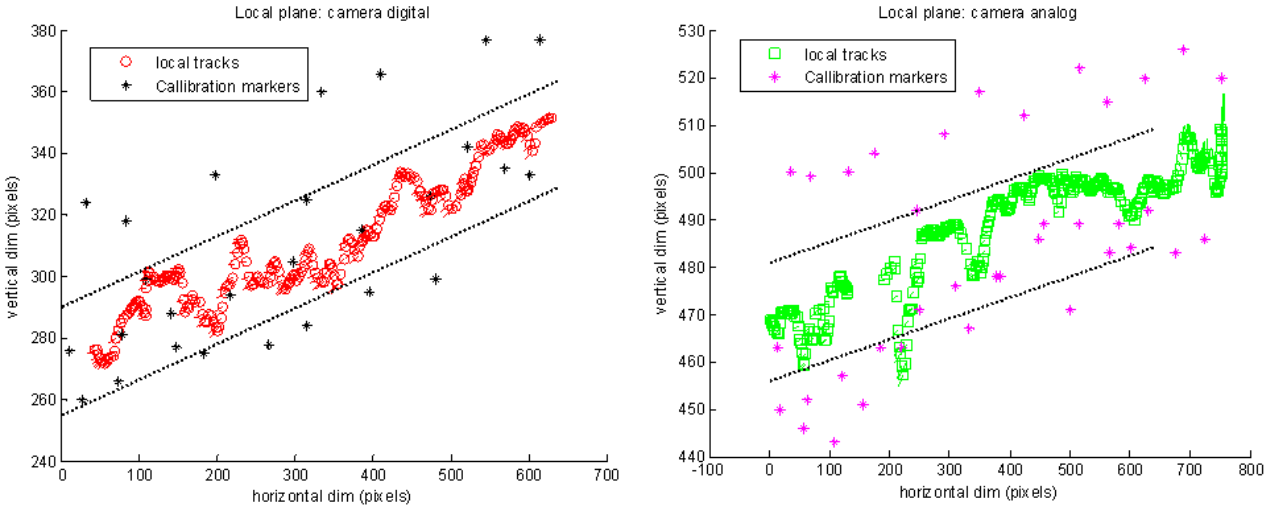

Figura 6.12: Posiciones en el plano de la imagen (coordenadas locales) obtenidas por los dos agentes sensores involucrados.

6.4. Experimentación realizada en entornos interiores

Para las pruebas realizadas en entornos interiores (Castanedo et al., 2008a)(Castanedo et al., 2008b), se han generado una serie de vídeos, los cuales han sido grabados en el interior del laboratorio del grupo de inteligencia artificial aplicada (GIAA). Este laboratorio tiene unas dimensiones de 660cm por 880cm (ver figura 6.1) donde se encuentran colocadas de forma fija 3 vídeo-cámaras (modelo SONY EVI-100 PTZ). A modo de ejemplo, en la figura 6.27 se muestran una serie de imágenes de entrada donde se observan las diferentes vistas de cada una de las cámaras. La figura 6.28 muestra una captura del seguimiento llevado a cabo en

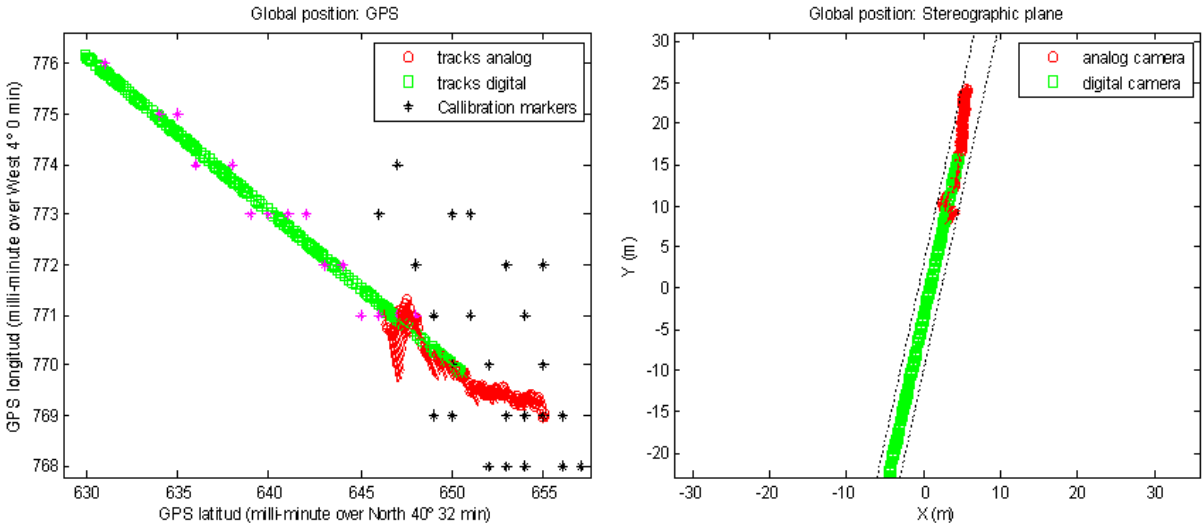

Figura 6.13: Proyección de las pistas en coordenadas globales: GPS y plano estereográfico.

un instante determinado y en la figura 6.29 se muestra la detección de píxeles en movimiento para el mismo instante.

Para llevar a cabo las grabaciones, se han utilizado las tres vídeo-cámaras, las cuales están conectadas, cada una de ellas, a una tarjeta capturadora modelo matrox morphis. Se ha realizado una grabación de 1200 fotogramas, a una resolución de 768x576 píxeles y con una velocidad de captura de 25 fotogramas/segundo, que corresponde a una duración total de 48 segundos. La obtención de las imágenes, de las tres vídeo cámaras, se ha llevado a cabo utilizando un solo ordenador y por medio de un programa multi-hilo, en el cual los fotogramas de cada vídeo cámara, se capturan en instantes de tiempo lo más similares posibles, dando lugar a tres ficheros de imágenes que están sincronizados fotograma a fotograma.

La secuencia de imágenes, obtenidas por cada cámara, se ha almacenado en un fichero de vídeo y posteriormente durante la ejecución del sistema cada una de las secuencias de entrada son analizadas por un agente sensor. Se ha optado por realizar las pruebas a partir de los ficheros grabados, debido a su facilidad para repetir los experimentos. No obstante, el sistema se encuentra preparado para funcionar también en tiempo real, simplemente cambiando el origen de la fuente de información. En la ejecución de los experimentos realizados se despliegan: 3 agentes sensores, cada uno de los cuales analiza la información de una secuencia de entrada; un agente de tipo fusión y la interfaz de usuario para mostrar el resultado. Todos los agentes del sistema se encuentran conectados entre sí por una red de transmisión TCP/IP ethernet a 100 MB.

Cada uno de los 3 agentes sensores, S_i, obtienen imágenes $I(i,j)$ con una frecuencia determinada, V_i. El objetivo detectado O_j se representa con una pista $\hat{x}_j^i[n]$, que contiene la descripción numérica de sus atributos y estado: localización, velocidad, dimensiones y la matriz de error de covarianza asociada, $R_j^i[n]$ (ver figura 6.17).

La grabación de los vídeos ha consistido en el movimiento de una persona siguiendo un patrón predefinido dentro del entorno monitorizado por las cámaras, por tanto ha sido posible obtener el ground-truth de una forma sencilla interpolando las posiciones conocidas. En la

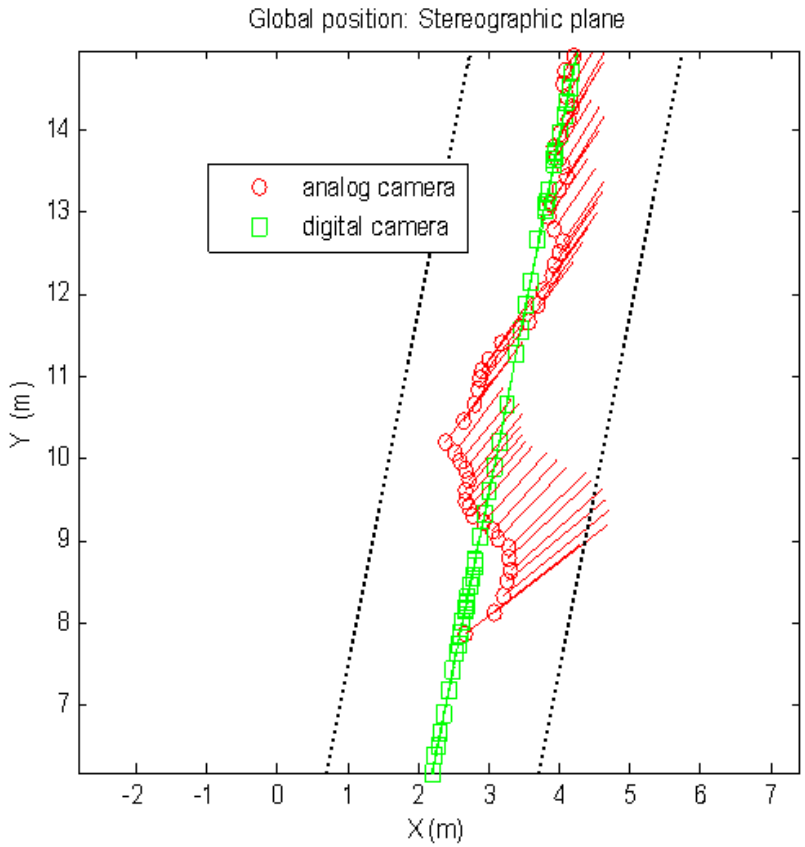

Figura 6.14: Proyección de las pistas en el plano estereográfico cartesiano.

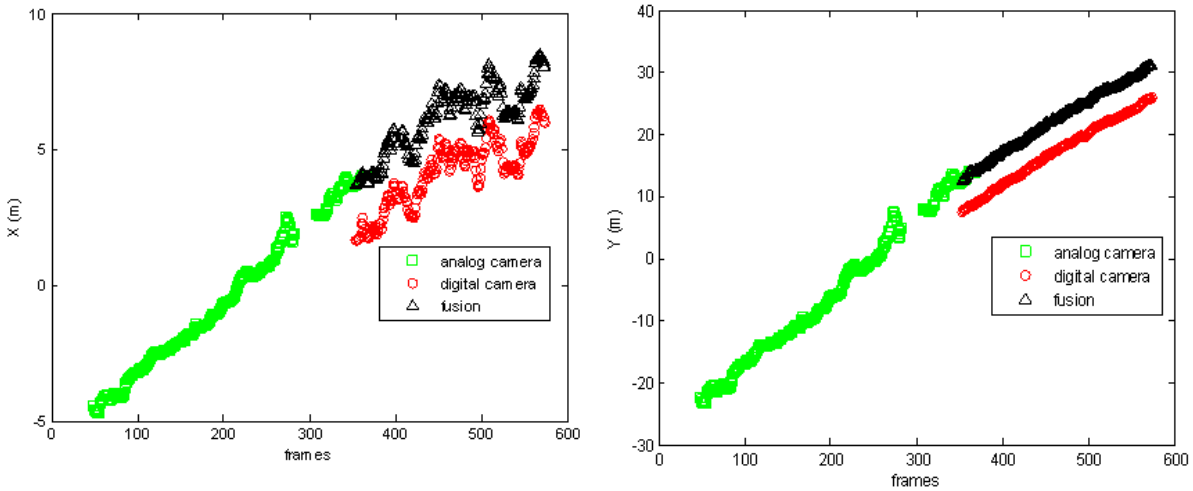

Figura 6.15: Coordenadas X e Y de las pistas originales y resultado fusionado.

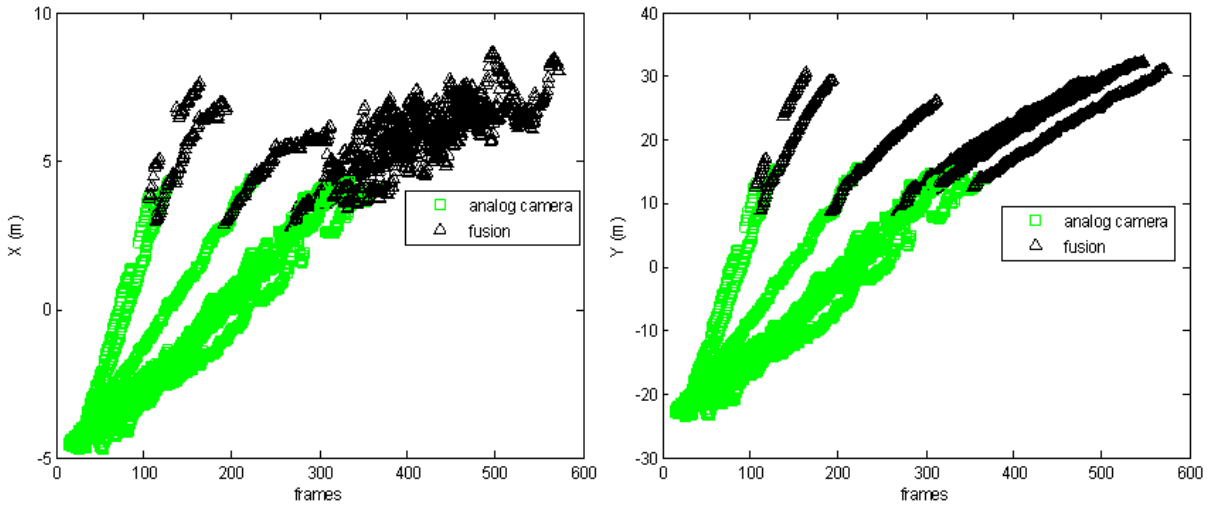

Figura 6.16: Coordenadas X e Y de las pistas fusionadas para todos los vídeos.

figura 6.18 se muestran las posiciones del movimiento de la persona proyectadas en el plano del suelo. Estas posiciones se han considerado como el ground-truth del sistema para estas pruebas y de esta forma se ha podido evaluar su funcionamiento de una forma cuantitativa. Los vídeos de los tres cámaras y el resultado del seguimiento en coordenadas locales se pueden ver en `http://www.youtube.com/fcastanedo`. Las pruebas experimentales se han llevado a cabo utilizando dos configuraciones distintas: Fusión pasiva y Fusión activa. Los fotogramas de entrada de cada uno de los 3 vídeos se han analizado con un agente sensor, ejecutándose cada uno de ellos en una máquina distinta. Los fotogramas de entrada se han procesado a una media de 5 fotogramas por segundo y el resultado es enviado a un agente de fusión que se ejecuta en otra máquina distinta.

6.4.1. Fusión pasiva

En este esquema de funcionamiento, cada agente sensor procesa la información del entorno en su campo de visión y envía las pistas al agente de fusión (ver figura 5.10).

En esta primera ejecución, el agente de fusión simplemente obtiene las posiciones en coordenadas globales de las pistas recibidas y son comparadas frente a los valores reales. Esta información se muestra en las figuras 6.19, 6.20 y 6.21 para el agente sensor 1,2 y 3 respectivamente. Con las pruebas anteriores se obtiene información del comportamiento individual de cada sensor.

A continuación se ha probado el sistema utilizando los 3 agentes sensores y el agente de fusión. En esta configuración, el agente de fusión se encarga de llevar a cabo la combinación de los datos, utilizando las pistas recibidas y disparando, el plan encargado de combinar los datos, con la frecuencia especificada en la frecuencia de fusión. El resultado obtenido se muestra en la figura 6.22, donde se observa el resultado de los valores fusionados frente a la información real.

Figura 6.17: Ejemplo de la persona detectada con su caja envolvente.

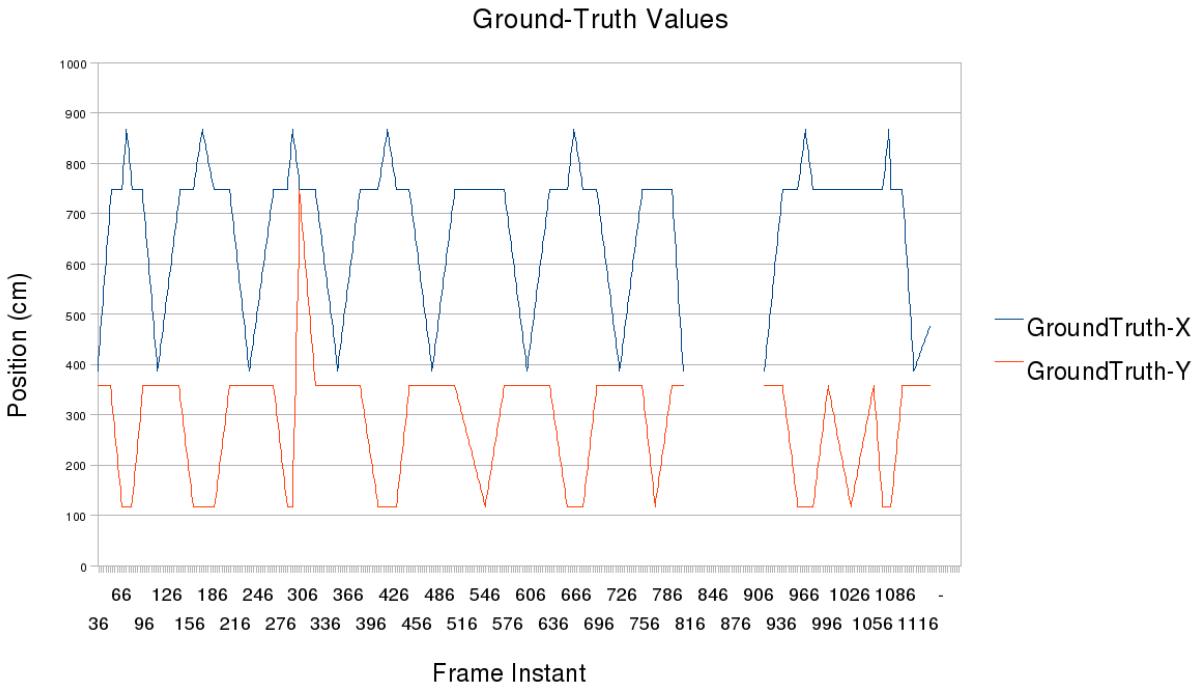

Figura 6.18: Posiciones en el plano del suelo del movimiento seguido durante la grabación del vídeo (ground-truth).

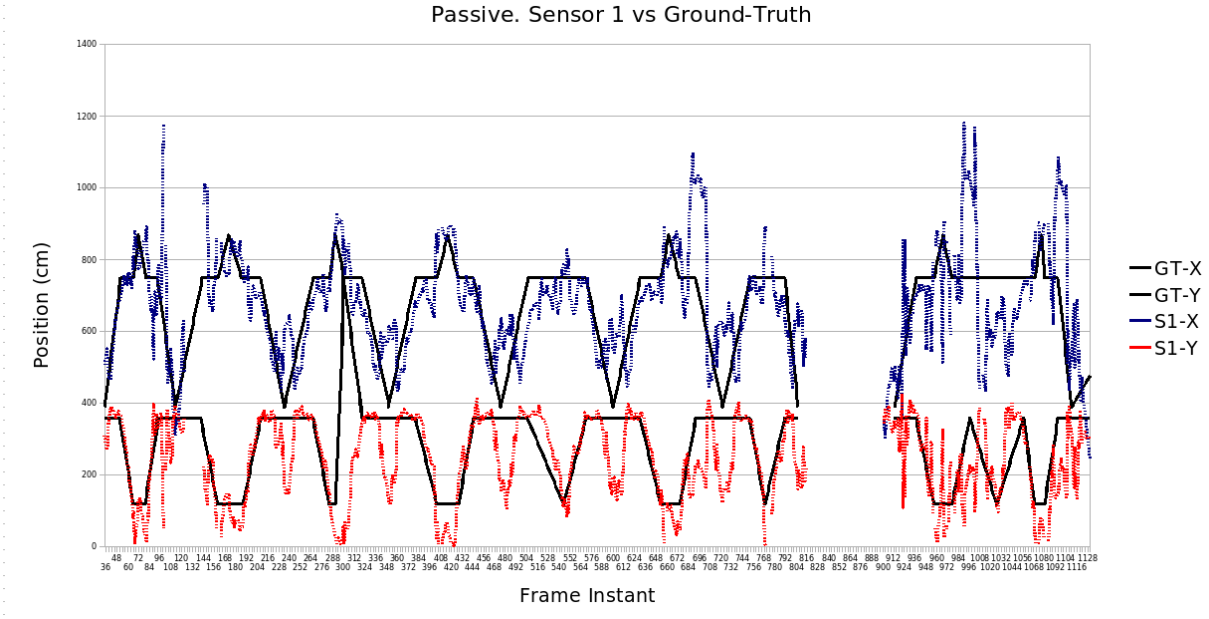

Figura 6.19: Valores de seguimiento del sensor 1 en coordenadas globales comparados con el ground-truth en el modo de fusión pasiva. Nótese el efecto de los errores sistemático en el eje Y debido a una oclusión.

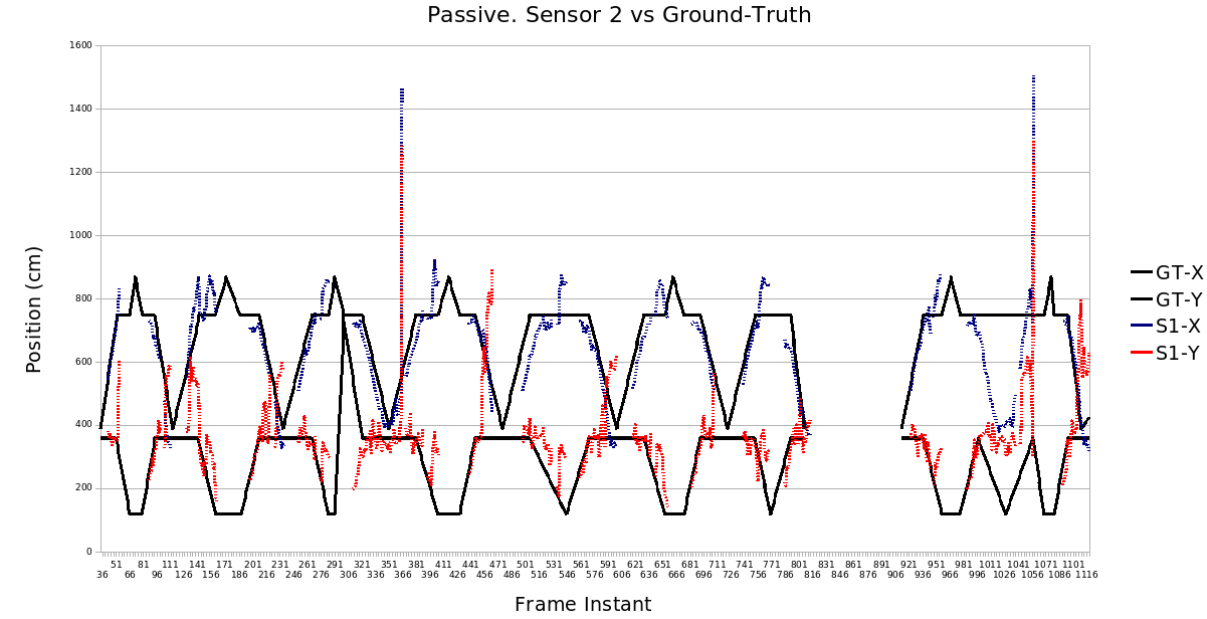

Figura 6.20: Valores de seguimiento del sensor 2 en coordenadas globales comparados con el ground-truth en el modo de fusión pasiva.

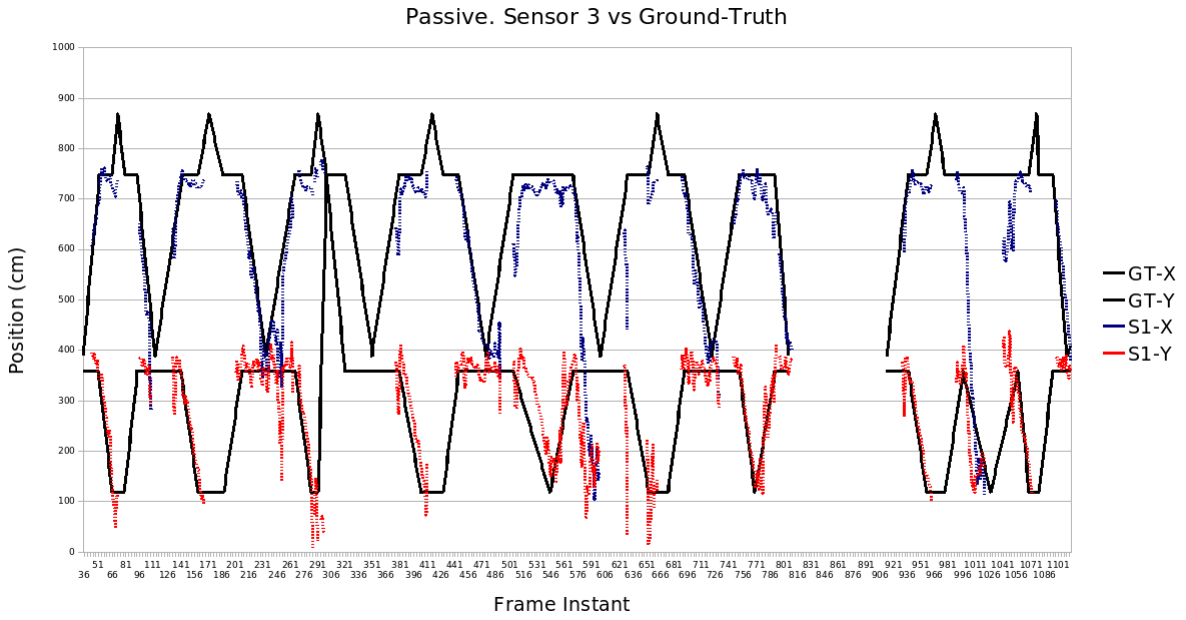

Figura 6.21: Valores de seguimiento del sensor 3 en coordenadas globales comparados con el ground-truth en el modo de fusión pasiva.

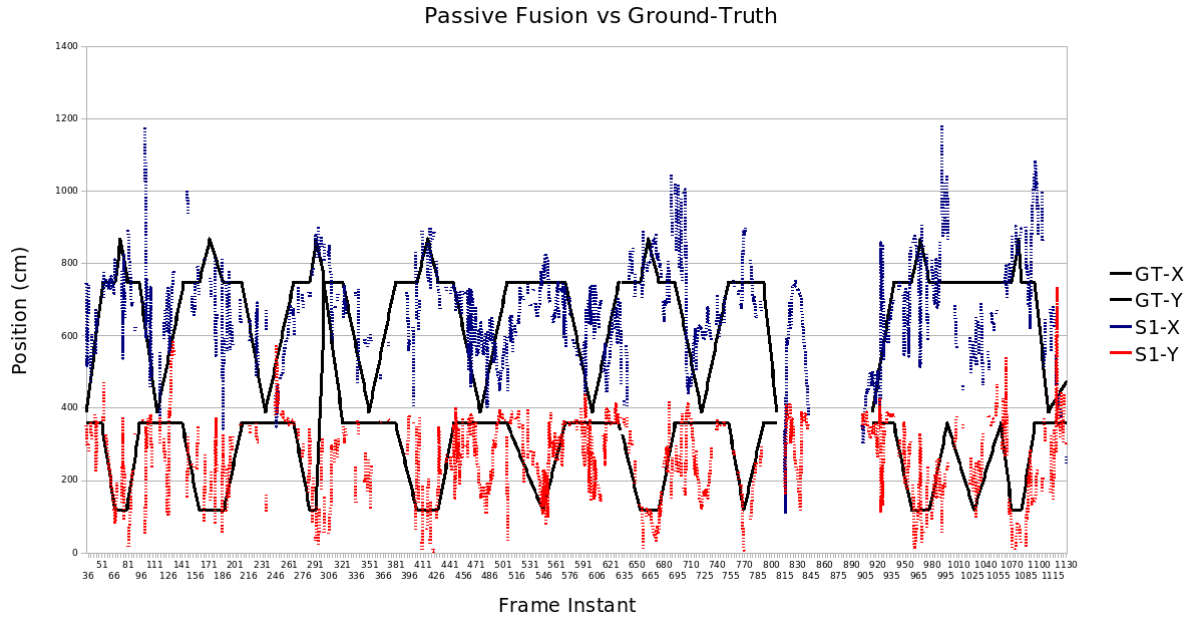

Figura 6.22: Valores de seguimiento fusionados (fusión pasiva) en coordenadas globales comparados con el ground-truth.

6.4.2. Fusión activa

El funcionamiento de la fusión activa (ver fig 5.11) consiste en involucrar a cada agente en el proceso de fusión. La diferencia de este esquema de fusión es el feedback enviado a

los agentes sensores, que les permite razonar localmente acerca de la calidad de información que están enviando con respecto a la información fusionada. Como cada agente sensor se comporta como un proceso autónomo, este puede razonar acerca de la necesidad y de la forma de corregir sus desviaciones. Por tanto, la fusión activa implica que cada agente sensor podría ser capaz de corregir valores de estado, eliminar objetos y cambiar proyecciones.

Con la información de seguimiento de cada uno de los agentes sensores por separado, se ha realizado una ejecución utilizando el esquema de la fusión activa y la información de ground-truth como datos de realimentación, es decir se realimenta a los sensores con la mejor información disponible. En un entorno real no se dispone de esta información de ground-truth, sin embargo, para estas pruebas nos proporciona un upper-bound de lo mejor que podrían comportarse los agentes sensores. Se han obtenido los resultados que se muestran en las gráficas 6.23, 6.24 y 6.25 para los agentes sensores 1,2 y 3 respectivamente. Finalmente, los resultados de la fusión de datos de los 3 sensores utilizando el esquema de la fusión activa se muestran en la gráfica 6.26. Como resumen, se muestra la tabla 6.2 con el error absoluto medio de cada una de las pruebas. En esta tabla se observa la mejora que se obtiene cuando se realiza la fusión activa frente a la fusión pasiva en cada una de las 4 configuraciones (sensores 1,2 y 3 de forma individual y los 3 conjuntos). Tomando como ejemplo únicamente el sensor 3, se observa como se reduce el error de un 7.63 % a un 2.65 % en el eje X y del 7,04 % al 4 % en el eje Y, al utilizar la información de feedback para corregir el estado del agente sensor. La mejora de la fusión activa frente a la fusión pasiva se refleja al pasar de un 10.88 % de error al 6.39 % en el eje X y del 10.88 % al 6.06 % en el eje Y.

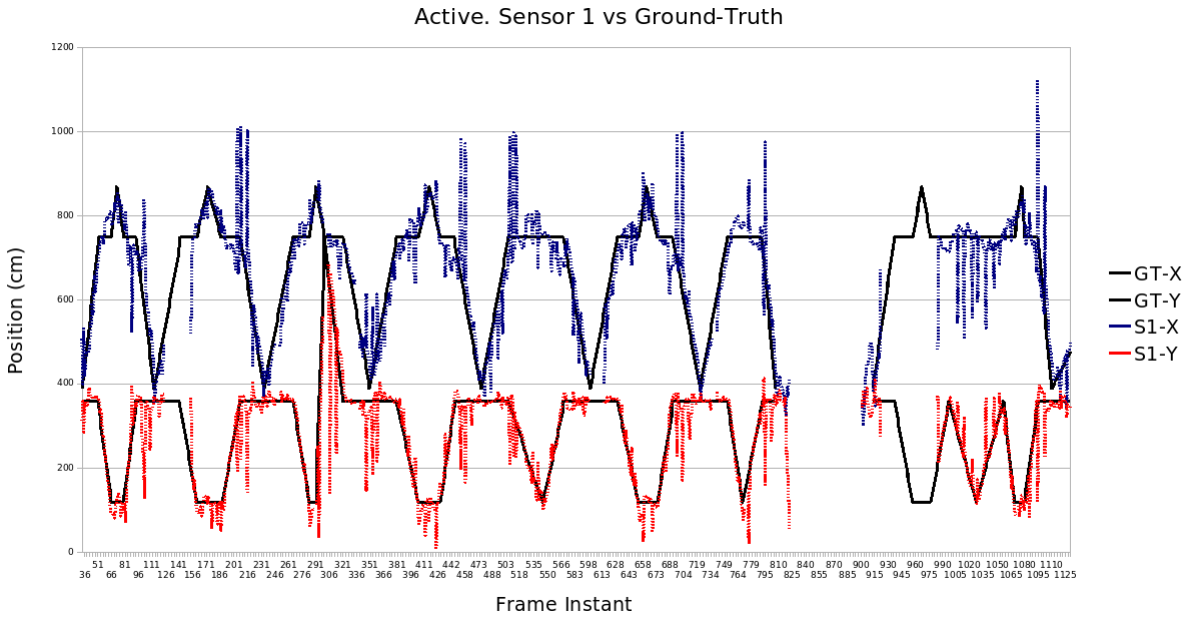

Figura 6.23: Valores de seguimiento del sensor 1 (fusión activa) en coordenadas globales comparados con el ground-truth en el modo de fusión activa. El ground-truth es utilizado como información de feedback.

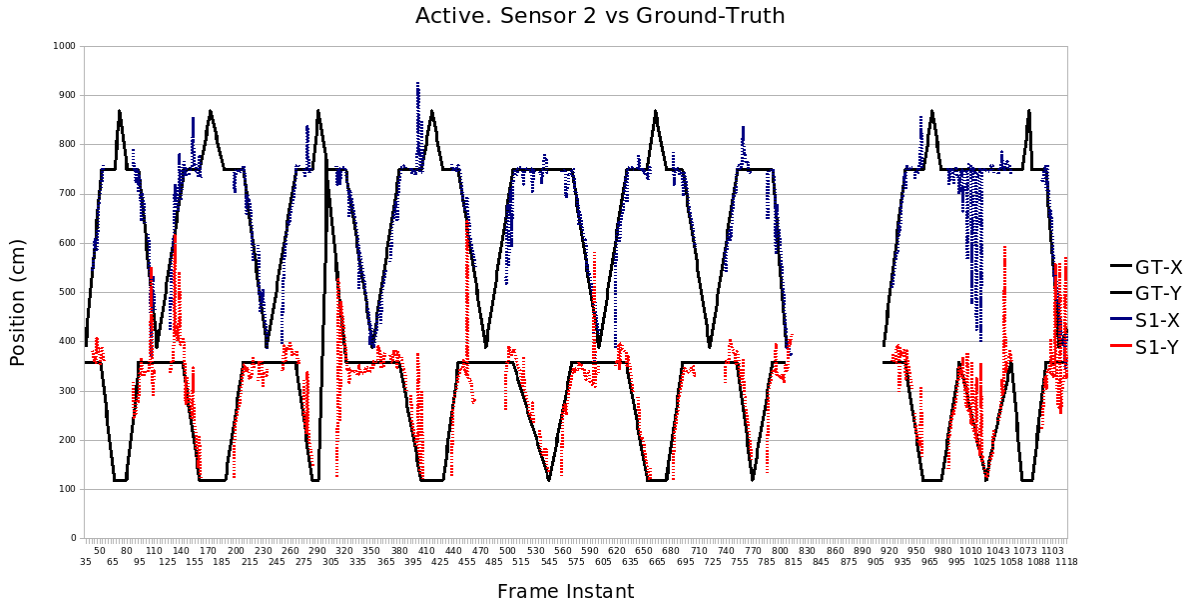

Figura 6.24: Valores de seguimiento del sensor 2 (fusión activa) en coordenadas globales comparados con el ground-truth en el modo de fusión activa. El ground-truth es utilizado como información de feedback.

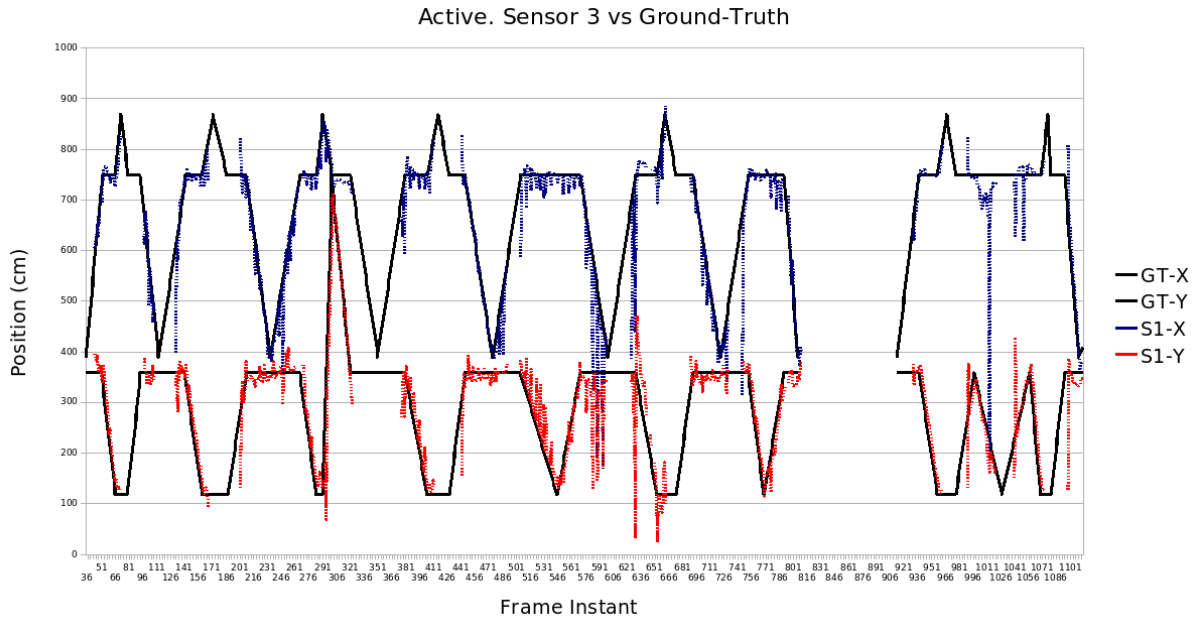

Figura 6.25: Valores de seguimiento del sensor 3 (fusión activa) en coordenadas globales comparados con el ground-truth en el modo de fusión activa. El ground-truth es utilizado como información de feedback.

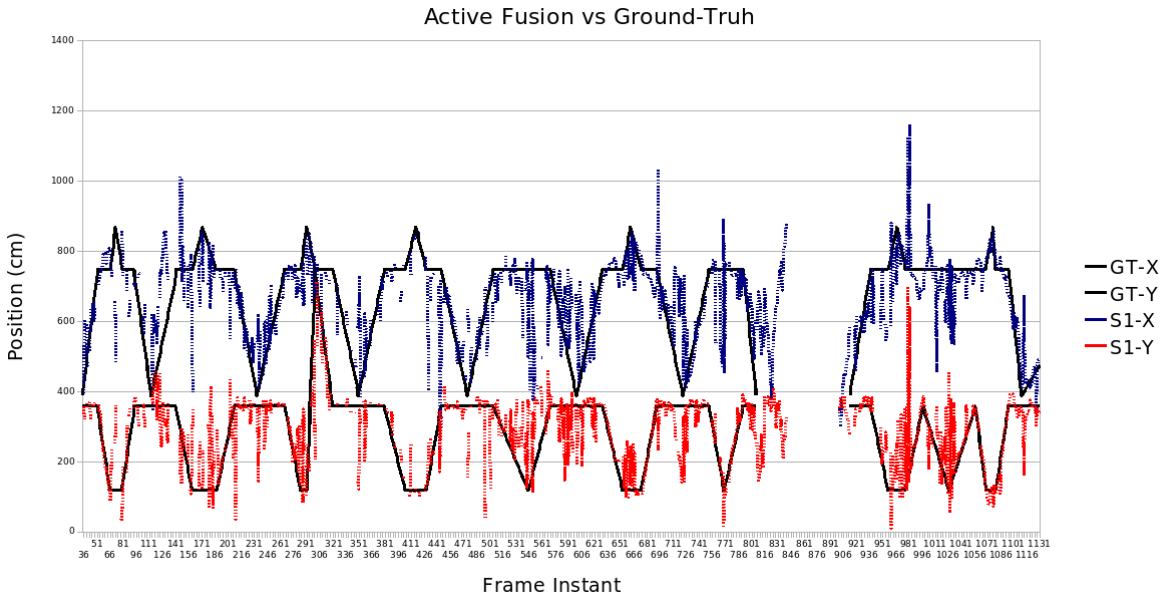

Figura 6.26: Valores de seguimiento fusionados en coordenadas globales comparados con el ground-truth en el modo de fusión activa.

Tabla 6.2: Error absoluto medio entre las posiciones de ground-truth y las posiciones de seguimiento (cm). El eje X tiene una longitud de 880cm y el eje Y de 660cm.

	S1	S2	S3	S1-S2-S3
Passive error (X)	89.28 (10.14 %)	69.39 (7.88 %)	67.17 (7.63 %)	86,04 (9.77 %)
Active error (X)	43.15 (4.9 %)	25.01 (2.84 %)	22.36 (2.65 %)	56,23 (6.39 %)
Passive error (Y)	70.37 (10.66 %)	85.2 (12.9 %)	46.48 (7.04 %)	71.85 (10.88 %)
Active error (Y)	23.28 (3.57 %)	33.5 (5.07 %)	26.42 (4 %)	40,03 (6.06 %)

6.5. Experimentación utilizando el dataset de Apidis

En este apartado, se describe la experimentación realizada con el sistema, utilizando el conjunto de datos de Apidis (APIDIS, [Online 03/2010]). El objetivo de estas pruebas es demostrar el rendimiento del sistema cuando se añaden más agentes sensores. El conjunto de datos Apidis (APIDIS, [Online 03/2010]) proporciona información de seguimiento de los jugadores en un partido de baloncesto durante 1500 fotogramas. La información del juego es adquirida por siete cámaras distintas, de 2 Megapíxels de resolución cada una, instaladas alrededor y en el techo de un campo de baloncesto. Las imágenes de todas las cámaras fueron capturadas por el mismo servidor a ∼22 fotogramas por segundo y por tanto, los fotogramas se encuentran pseudo-sincronizados en el tiempo. Apidis proporciona dos ficheros de vídeo, uno en formato nativo con un tamaño de 1600x1200 píxeles y otro con un tamaño de 800x600 píxeles. Para la realización de los experimentos se han utilizado los datos de los ficheros pseudo sincronizados con una resolución de 800x600 píxeles en MPEG-4.

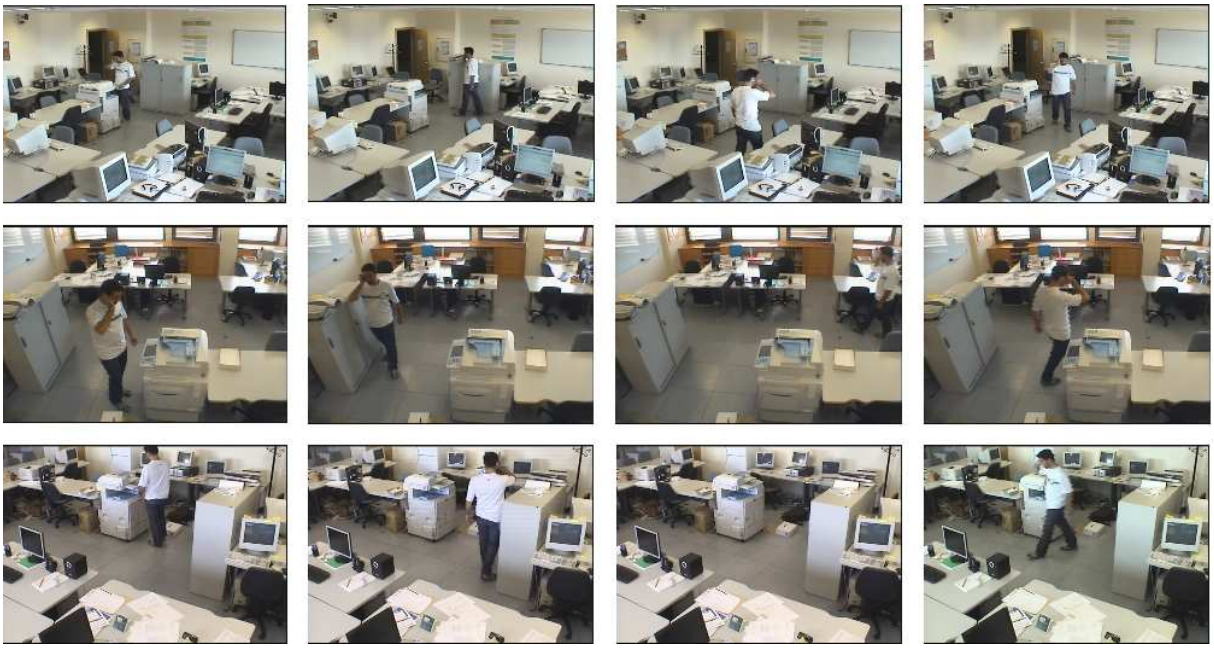

Figura 6.27: Ejemplo de imágenes de entrada (fotogramas 75, 150, 225 y 300) obtenidas por las cámaras 1, 2 y 3.

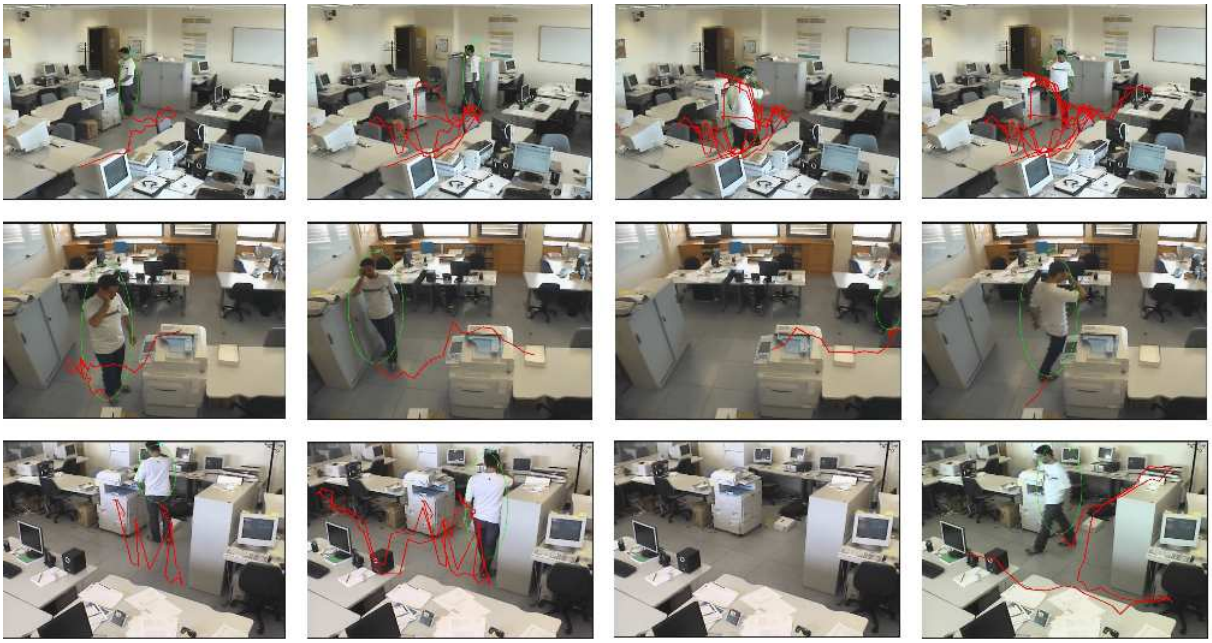

Figura 6.28: Ejemplo del seguimiento realizado para las imágenes de entrada (fotogramas 75, 150, 225 y 300) obtenidas por las cámaras 1,2 y 3.

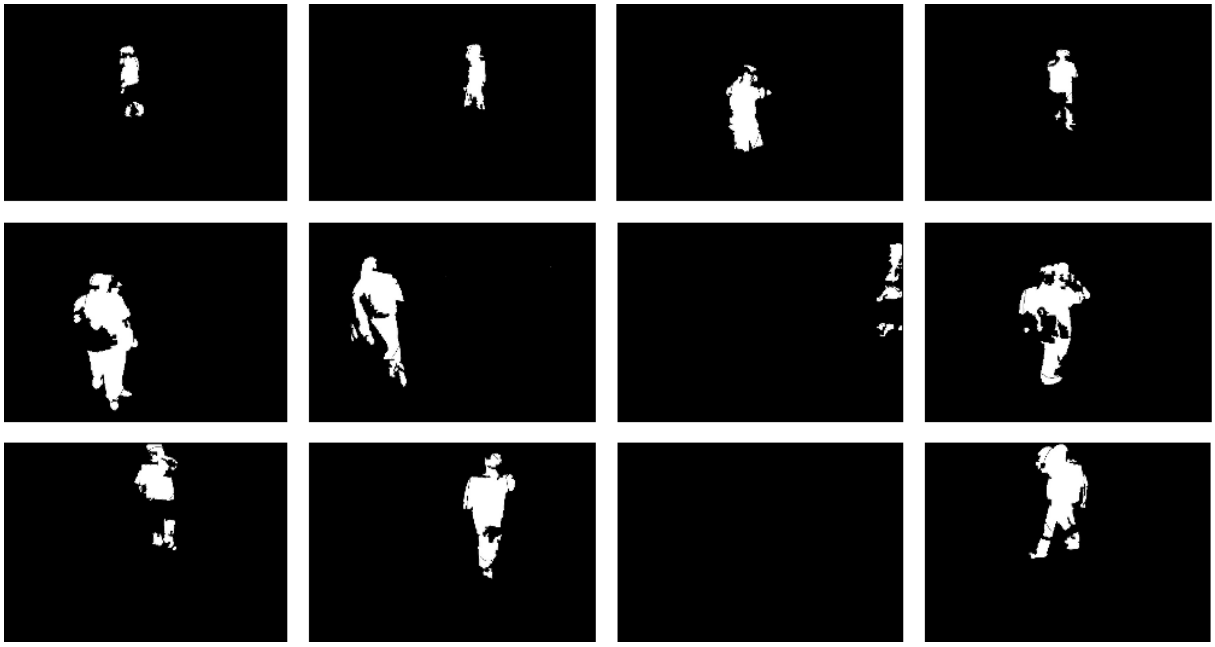

Figura 6.29: Ejemplo de los píxeles en movimiento detectados para las imágenes de entrada (fotogramas 75, 150, 225 y 300) obtenidas por las cámaras 1, 2 y 3.

6.5.1. Calibración en el conjunto de datos de Apidis

La primera fase necesaria para poder llevar a cabo una fusión de datos distribuida, en una red de sensores visuales, es la fase de calibración. Utilizando el método descrito en 3.3 se ha obtenido el conjunto de parámetros extrínsecos, los cuales se muestran en la tabla 6.3 y el valor de los parámetros intrínsecos que se muestran en la tabla 6.4.

Para obtener los valores de los parámetros de calibración, se han utilizado las 69 posiciones proporcionadas, las cuales se encuentran distribuidas en el plano del suelo (asumiendo por tanto z=0) y consisten en las posiciones conocidas de un campo de baloncesto. De los 69 puntos posibles, para la calibración de cada cámara, solo se han utilizado aquellos que se encuentran visibles. Los puntos globales conocidos de cada una de las cámaras, son asociados con los puntos en dos dimensiones del plano de la imagen obtenidos de forma manual. Para

Tabla 6.3: Valores estimados de los parámetros extrínsecos en Apidis

cámara	**Rx** (deg)	**Ry** (deg)	**Rz** (deg)	**Tx** (mm)	**Ty** (mm)	**Tz** (mm)
cámara 1	-69.769	21.125	4.219	-3347.818	-1802.199	29096.422
cámara 2	-48.16	-69.298	-42.391	-9505.713	2237.553	9324.672
cámara 4	-105.49	-46.033	13.605	-10639.491	7367.315	18124.522
cámara 6	-70.695	-20.103	-2.691	-22388.609	4.145	20168.512
cámara 7	-62.928	23.431	5.752	1646.596	-4522.187	27700.724

Tabla 6.4: Valores estimados de los parámetros intrínsecos en Apidis

cámara	**f** (mm)	**kappa1** ($1/mm^2$)	**Cx** (píxeles)	**Cy** (píxeles)	**Sx**
cámara1	19.403642	9.913269e-04	407.30926	274.804934	1.0
cámara2	19.199602	8.873476e-04	473.95128	204.872783	1.0
cámara4	26.300296	-4.107186e-05	457.79424	-179.912847	1.0
cámara6	20.569473	7.014055e-04	395.10578	264.560713	1.0
cámara7	33.042031	3.644292e-04	182.35262	393.347814	1.0

estos experimentos, las cámaras del techo que tienen lentes de ojo de pez (cámara 4 y cámara 5) han sido descartadas. En la figura 6.39, se proporciona un ejemplo de la información visual que obtienen cada una de las cámaras y del plano global que se ha utilizado para obtener los valores de calibración.

Una vez obtenidos los parámetros de calibración se procedió a obtener las métricas de evaluación de la calidad de la calibración. La tabla 6.9 muestra el error de calibración normalizado (NCE) (Weng et al., 1992) de cada una de las cámaras. Por otro lado, la tabla 6.10 muestra las estadísticas de los errores E_d, E_u y E_o para cada una de las cámaras.

6.5.2. Pruebas realizadas

El conjunto de datos del proyecto Apidis proporciona los valores de una caja envolvente con las coordenadas de los jugadores en el plano local de cada una de las cámaras. Se ha utilizado la información de esta caja envolvente como la entrada proporcionada a cada uno de los agentes sensores. Es decir, en estas pruebas los valores no se obtienen del procesado de la imagen, sino directamente de la información de ground-truth proporcionada en el dataset. El esquema de la configuración utilizada para realizar los experimentos se muestra en la figura 6.30. Se han llevado experimentos con dos margenes de error distintos, introduciendo errores aleatorios entre (1, 500) y (1, 1500) milímetros en los datos de la cámara 1. Estos errores aleatorios modifican la información de seguimiento proporcionada por el sensor 1 en coordenadas globales (después de aplicar la calibración). Se ha realizado una comprobación del comportamiento de forma incremental, añadiendo un nuevo sensor al conjunto de los datos de C1, C2 y C3 y dando lugar a 3 posibles configuraciones:

- Utilizando los datos del sensor 1, 2 y 3 e introduciendo errores en el sensor 1.
- Utilizando los datos del sensor 1, 2, 3 y 4 e introduciendo errores en el sensor 1.
- Utilizando los datos del sensor 1,2,3,4, y 5 e introduciendo errores en el sensor 1.

Para cada jugador y para cada conjunto de cámaras, el error absoluto medio (en milímetros) de las posiciones fusionadas con los errores inducidos, frente a las posiciones fusionadas sin errores (usando directamente el ground-truth) se muestran en las tablas 6.6 y 6.8 y de forma gráfica en 6.31, 6.32 6.35 y 6.36. Los valores del error cuadrático medio para cada jugador y para cada conjunto de cámaras se muestran en las tablas 6.5 y 6.7 y de forma

gráfica en 6.33, 6.34, 6.37 y 6.38. Con los datos obtenidos, se puede observar como los errores en el seguimiento se reducen cuando el número de sensores visuales utilizado aumenta. Estos resultados preliminares muestran la robustez del sistema y como se comporta frente a los errores cuando se añaden más sensores visuales.

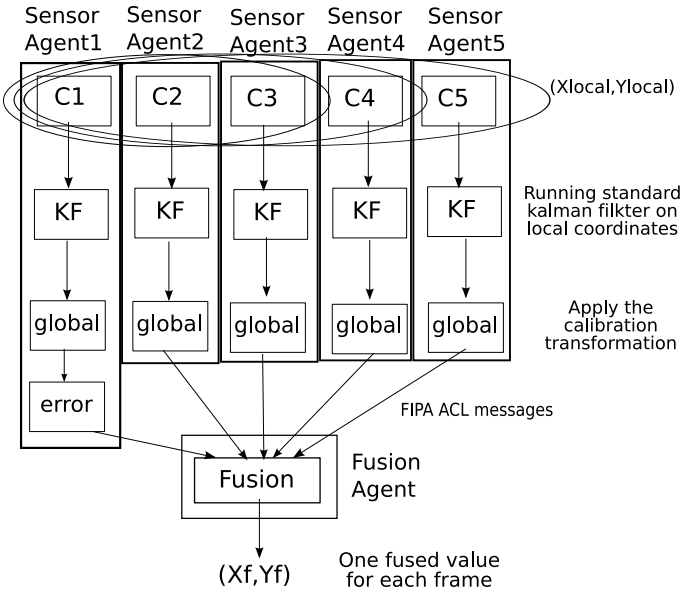

Figura 6.30: El proceso utilizado para realizar las pruebas introduciendo errores en los datos del sensor 1, con el objetivo de comprobar como mejora el sistema cuando se incrementa el número de sensores.

Tabla 6.5: Error cuadrático medio (RMSE) en mm de los valores de ground-truth frente a los valores fusionados globales con un error aleatorio de (1, 500) en la cámara 1.

player-team	3-cameras-x	3-cameras-y	4-cameras-x	4-cameras-y	5-cameras-x	5-cameras-y
p10-A	448,58	225,94	344,10	221,84	336,65	218,28
p14-A	275,05	157,54	196,49	152,30	189,75	144,81
p04-A	365,23	174,46	351,25	180,83	344,33	173,31
p05-A	290,29	243,68	249,46	234,84	235,70	216,33
p09-A	309,03	164,08	244,00	159,38	235,43	147,91
p15-B	422,37	187,25	313,60	185,29	295,92	173,72
p06-B	350,50	199,73	296,40	191,60	284,86	182,56
p04-B	344,03	144,69	329,23	165,37	319,62	162,53
p07-B	282,22	164,66	223,42	157,69	214,39	156,09
p09-B	298,62	191,51	280,64	193,72	276,27	190,03

Tabla 6.6: Error absoluto medio (MAE) en mm de los valores de ground-truth frente a los valores fusionados globales con un error aleatorio de (1, 500) en la cámara 1.

player-team	3-cameras-x	3-cameras-y	4-cameras-x	4-cameras-y	5-cameras-x	5-cameras-y
p10-A	298,8	168,66	228,86	168,54	220,21	164,45
p14-A	195,22	120,74	149,95	117,92	143,72	112,36
p04-A	209,22	135,21	202,04	134,42	191,48	125,78
p05-A	219,94	174,48	192,13	167,95	183,05	154,71
p09-A	208,93	128,18	181,60	125,77	175,06	115,27
p15-B	298,42	137,27	224,22	139,44	214,29	127,70
p06-B	257,56	154,48	227,01	150,72	217,46	143,99
p04-B	231,11	108,56	201,75	114,51	190,36	111,04
p07-B	200,29	125,92	165,89	119,55	159,97	116,94
p09-B	213,05	141,19	191,06	143,24	187,08	139,86

Tabla 6.7: Error cuadrático medio (RMSE) en mm de los valores de ground-truth frente a los valores fusionados globales con un error aleatorio de (1, 1500) en la cámara 1.

player-team	3-cameras-x	3-cameras-y	4-cameras-x	4-cameras-y	5-cameras-x	5-cameras-y
p10-A	498,89	307,81	409,53	301,11	377,49	272,81
p14-A	353,13	269,82	292,96	256,79	261,74	220,25
p04-A	425,88	279,34	416,71	284,34	381,44	238,30
p05-A	367,33	350,43	341,34	339,69	286,83	278,57
p09-A	371,71	268,86	319,48	262,99	284,15	216,04
p15-B	487,02	306,95	399,41	302,99	353,06	251,32
p06-B	444,75	315,39	399,54	314,05	337,66	240,68
p04-B	401,00	255,01	392,17	260,77	357,36	227,44
p07-B	348,42	260,88	303,32	244,69	270,89	220,18
p09-B	357,30	272,89	337,85	265,72	321,10	236,60

Tabla 6.8: Error absoluto medio (MAE) en mm de los valores de ground-truth frente a los valores fusionados globales con un error aleatorio de (1, 1500) en la cámara 1.

player-team	3-cameras-x	3-cameras-y	4-cameras-x	4-cameras-y	5-cameras-x	5-cameras-y
p10-A	366,97	231,36	300,56	227,35	267,34	207,97
p14-A	278,81	206,01	229,23	195,29	205,03	169,57
p04-A	293,33	226,75	289,55	226,24	251,06	187,06
p05-A	285,49	256,12	260,66	248,40	228,52	211,35
p09-A	276,20	211,45	246,25	203,72	219,80	171,77
p15-B	369,31	213,08	297,47	218,13	267,08	185,07
p06-B	331,71	235,66	300,83	227,18	265,96	189,58
p04-B	297,58	186,15	267,94	186,05	236,99	160,06
p07-B	264,52	195,22	230,06	180,20	208,60	164,10
p09-B	261,00	196,76	236,12	190,71	220,63	173,18

Tabla 6.9: Error de calibración normalizado (NCE) de Apidis

	cámara 1	**cámara 2**	**cámara 4**	**cámara 6**	**cámara 7**
NCE	3.031341	8.726915	7.313833	6.384913	4.983363

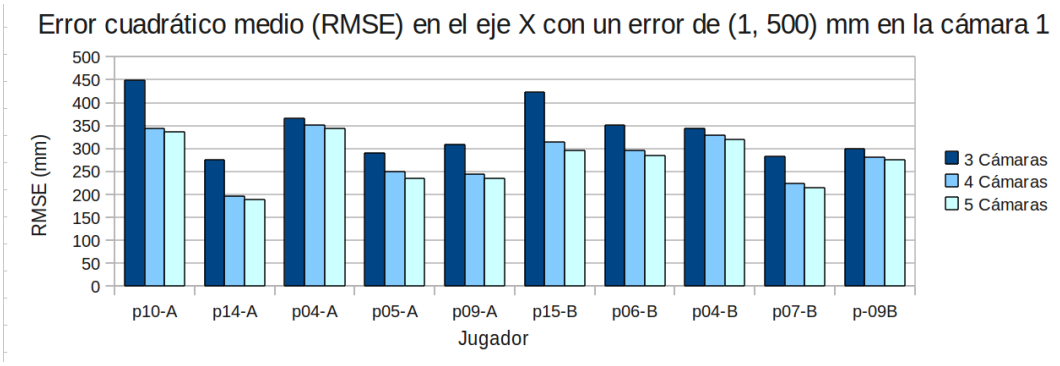

Figura 6.31: Error cuadrático medio (RMSE) en el eje X para todos los jugadores al inducir un error de (1, 500) mm en la cámara 1.

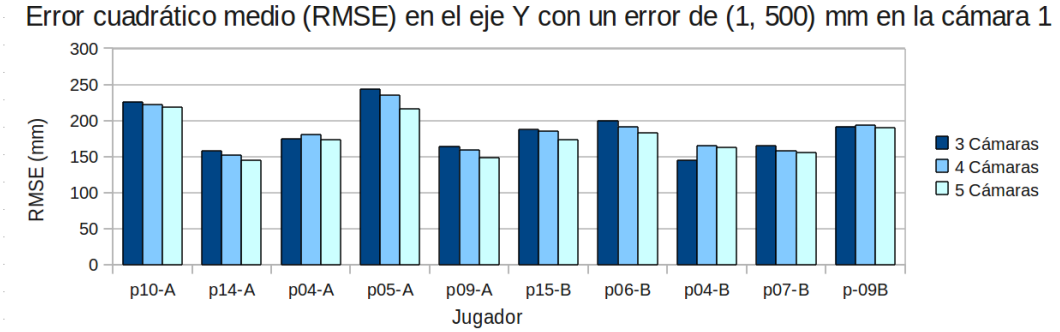

Figura 6.32: Error cuadrático medio (RMSE) en el eje Y para todos los jugadores al inducir un error de (1, 500) mm en la cámara 1.

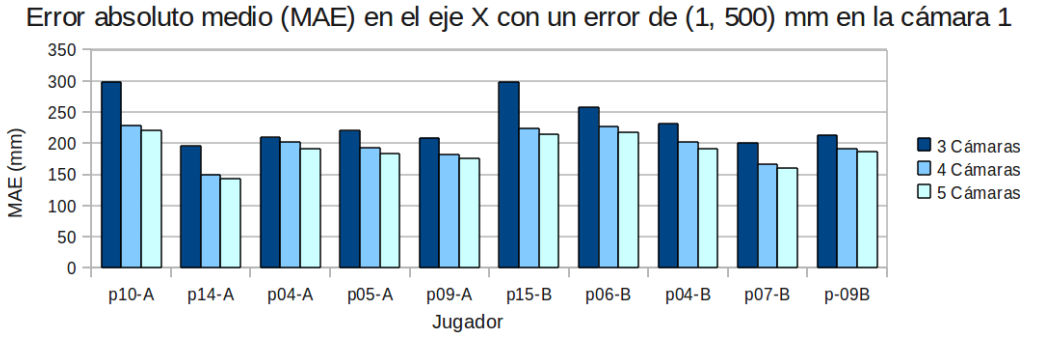

Figura 6.33: Error absoluto medio (MAE) en el eje X para todos los jugadores al inducir un error de (1, 500) mm en la cámara 1.

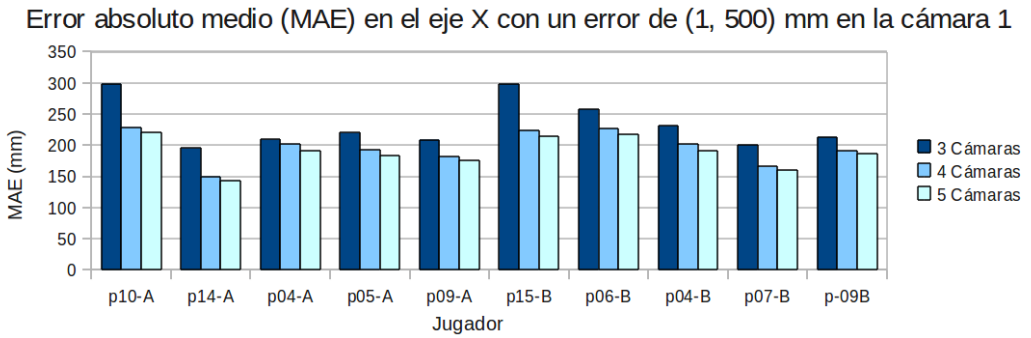

Figura 6.34: Error absoluto medio (MAE) en el eje Y para todos los jugadores al inducir un error de (1, 500) mm en la cámara 1.

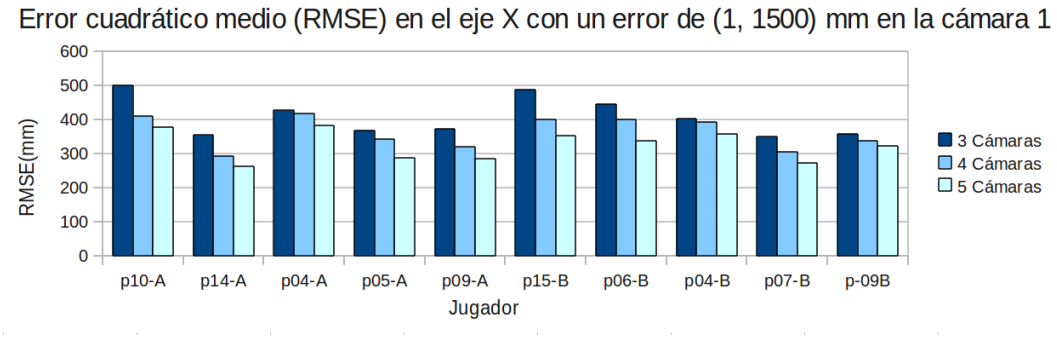

Figura 6.35: Error cuadrático medio (RMSE) en el eje X para todos los jugadores al inducir un error de (1, 1500) mm en la cámara 1.

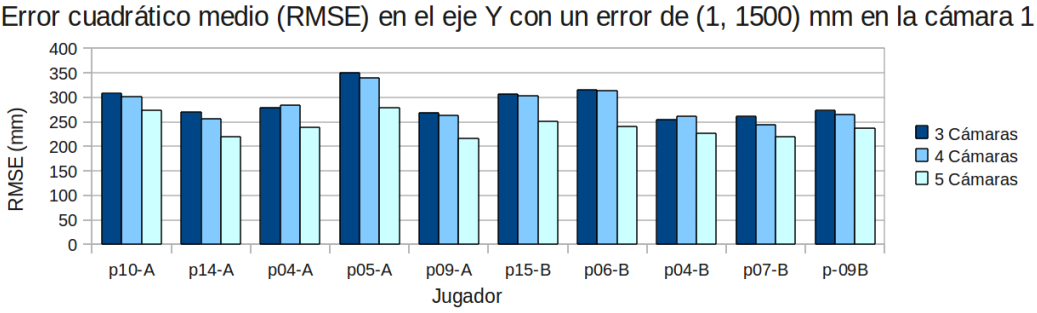

Figura 6.36: Error cuadrático medio (RMSE) en el eje Y para todos los jugadores al inducir un error de (1, 500) mm en la cámara 1.

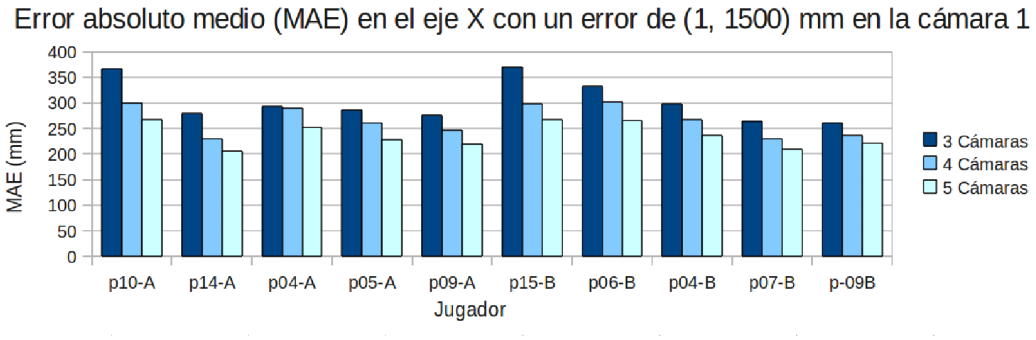

Figura 6.37: Error absoluto medio (MAE) en el eje X para todos los jugadores al inducir un error de (1, 1500) mm en la cámara 1.

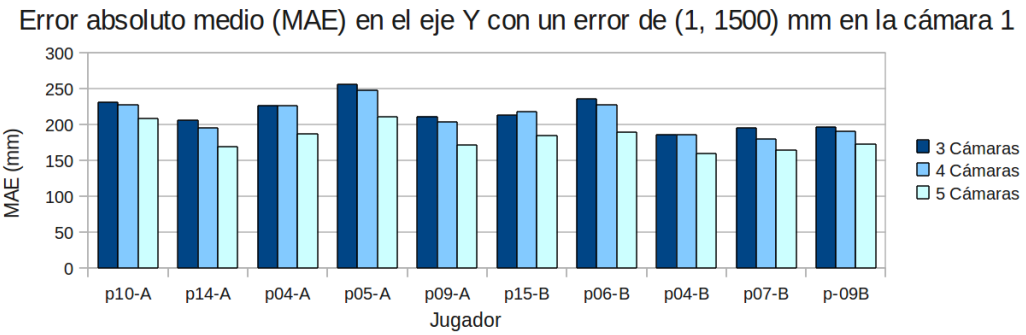

Figura 6.38: Error absoluto medio (MAE) en el eje Y para todos los jugadores al inducir un error de (1, 1500) mm en la cámara 1.

Tabla 6.10: Medidas de precisión de calibración de Apidis

medida	cámara 1	cámara 2	cámara 4	cámara 6	cámara 7
E_d media (pix)	1.188333	3.419681	3.022088	2.453974	1.980312
E_d stddev (pix)	0.587034	6.210263	3.150429	4.308640	2.133353
E_d max (pix)	2.692261	48.36222	18.59863	26.32054	11.40111
E_d sse (pix^2)	64.65491	2826.343	714.287	891.1312	215.7424
E_u media (pix)	1.237540	3.562748	2.985860	2.606630	2.034450
E_u stddev (pix)	0.606474	6.279122	3.099667	4.752018	2.131429
E_u max (pix)	2.757642	48.88498	18.285212	28.87088	11.40313
E_u sse (pix^2)	69.90688	2931.443	694.277255	1064.337	221.1883
E_o media (mm)	30.492083	88.80409	54.638430	58.28041	28.5351
E_o stddev (mm)	15.837112	225.7637	60.26738	100.9322	27.6851
E_o max (mm)	63.263249	1741.838	324.40383	616.1429	149.05
E_o sse (mm^2)	43430.692	3303790.5	247833.45	492417.7	40332.18

Figura 6.39: Una imagen con las 5 vistas distintas del conjunto de datos Apidis y el plano del suelo utilizado para la calibración y para mostrar el resultado fusionado.

7

Conclusiones y trabajo futuro

Los ideales son como las estrellas, no los alcanzas pero iluminan nuestro camino.
Demócrito.

7.1. Introducción

El procesamiento inteligente de la información multimedia ha ido ganando importancia dentro de los sistemas de vídeo-vigilancia y de las redes de sensores visuales. Entre todas las fuentes multimedia posibles (imágenes, audio, señales de sensores, datos en forma de texto, etc...), el vídeo es la fuente que proporciona una información más detallada, sin embargo su tratamiento es el más complejo.

En este trabajo, para proporcionar una buena comprensión del entorno, cada uno de los procesos involucrados en la monitorización (los agentes sensores) deben razonar acerca de las acciones a tomar en cada momento. Realizar este razonamiento al nivel de los algoritmos de procesado de imágenes, es una tarea bastante difícil debido a su complejidad; por tanto, el sistema multi-agente proporciona un marco para realizar el razonamiento distribuido de la información adquirida por los sensores visuales.

El uso de una arquitectura multi-agente para modelar las redes de sensores visuales proporciona varias ventajas: En primer lugar, la naturaleza de un sistema con bajo acoplamiento, como el de los sistemas multi-agente, proporciona una mayor flexibilidad en el proceso cooperativo. En un sistema software el acoplo está relacionado con los requisitos tecnológicos que se necesitan. Un sistema con un acoplo bajo no tiene requisitos de necesitar el mismo hardware, sistema operativo o lenguaje de programación para el funcionamiento entre los distintos componentes. En los sistemas multi-agente, solo es necesario conocer la dirección de la plataforma donde se está ejecutando el agente para comunicarse entre ellos y cada uno de los diferentes agentes se puede ejecutar sobre distintos sistemas operativos y distintas plataformas hardware. Esta característica de bajo acoplamiento es la que proporciona al sistema multi-agente una mayor flexibilidad frente a otros paradigmas de desarrollo software.

En segundo lugar, la habilidad de asignar planes de ejecución a cada uno de los agentes, es muy útil para solucionar tareas complejas en las redes de sensores visuales. De esta forma, la gestión de la ejecución de los planes en los sensores agentes se realiza de forma implícita con el procesado de la información visual, debido a que el proceso de control se encuentra embebido dentro del ciclo de razonamiento de un agente BDI.

En tercer lugar, una solución jerárquica distribuida, proporciona varias ventajas frente a una solución centralizada desde el punto de vista de la escalabilidad y de la tolerancia a fallos. En el sistema CSA, cada agente sensor solo proporciona su información local como parte de la información global y debe ser capaz de tomar decisiones basándose en su propia información local. Las redes de sensores visuales son distribuidas por definición: las fuentes de datos están espacialmente distribuidas, el procesado de los datos se lleva a cabo de una forma

distribuida y los usuarios finales interesados en los resultados obtenidos también pueden estar geográficamente dispersos. Las comunicaciones e interacciones de los agentes proporcionan una forma natural para intercambiar la información en una red de sensores visuales distribuida.

La arquitectura software propuesta en este trabajo proporciona las siguientes ventajas:

- Una arquitectura abierta y fácilmente escalable. Es muy sencillo incrementar el sistema con nuevos agentes de un tipo ya definido (con los mismos objetivos) o bien crear otros tipos de agentes (con nuevos objetivos).
- Una arquitectura basada en estándares. De esta forma es posible comunicar el sistema con otros desarrollos software, siempre y cuando soporten el estándar FIPA.
- La mejora del proceso de seguimiento distribuido por medio del razonamiento y la coordinación de los diferentes agentes implicados. Es posible mejorar el rendimiento y la precisión del sistema por medio del intercambio de la información.
- Los agentes solucionan problemas que son parte de un problema global. Cada uno de ellos lleva a cabo un procesamiento local de la información pero existe una visón global del seguimiento de todos ellos.
- El uso explícito de la información de feedback para mejorar de forma cooperativa el proceso de fusión. Los resultados mostrados en el apartado 6.4 ilustran la mejora en el proceso de fusión cuando se utiliza este canal de realimentación.

Las redes de sensores visuales, tienen características propias que las hacen diferentes a otro tipo de redes, debido a que es necesario llevar a cabo tareas de procesamiento de las imágenes para obtener la información apropiada, ya que generan una gran cantidad de datos. Además la aplicación de un proceso, utilizando las fuentes de información, como por ejemplo el seguimiento de los objetos, puede producir una salida inconsistente en el sistema; por salida inconsistente se entiende aquella que proporciona grandes diferencias entre cada una de las fuentes de entrada. En una red de sensores visuales existen varios factores que pueden producir información inconsistente entre las distintas fuentes, como pueden ser: oclusiones, cambios de iluminación, errores de hardware/software.

Uno de los objetivos de la comunicación y del razonamiento de la información en un sistema distribuido es evitar las inconsistencias y producir resultados más precisos por medio de la explotación de la redundancia proporcionada por las diversas fuentes.
La información fusionada e integrada por el agente de fusión sirve para dos propósitos:

1. Obtener una continuidad en el seguimiento a lo largo de todo el campo de visión de las cámaras que forman parte de la red distribuida.
2. Fusionar solo aquella información que proporciona un seguimiento coherente con respecto al resto de los sensores, descartando aquellos sensores que proporcionan errores en el seguimiento. Para ello, el agente de fusión, con la frecuencia determinada, selecciona la pistas que son consistentes entre sí.

7.2. Contribuciones realizadas

En este trabajo, en primer lugar se ha realizado un estudio de las técnicas de fusión de datos existentes en la actualidad, a continuación se han estudiado las redes de sensores visuales y los sistemas multi-agente de tipo BDI. Posteriormente, se ha realizado y detallado la propuesta de usar sistemas multi-agente BDI como la arquitectura software para gestionar una red de sensores visuales. Este sistema ha sido aplicado y se ha experimentado utilizando sensores visuales reales y conjuntos de imágenes, con el objetivo de fusionar posiciones en el plano del suelo.

Se ha argumentado que las redes de sensores visuales pueden ser modeladas utilizando los sistemas multi-agente y por tanto cada nodo dentro la red de sensores visuales es un agente del sistema y la fusión de datos se lleva a cabo como un proceso coordinado de intercambio de información entre los agentes del sistema. La aplicación de los sistemas multi-agente en redes de sensores visuales y en otro tipo de sistemas distribuidos ha sido objeto de estudio por parte de diversos investigadores; sin embargo, este es uno de los primeros y pocos trabajos existentes que utilizan el modelo de agentes BDI para realizar una fusión de datos distribuida en redes de sensores visuales.

Por otro lado, se ha propuesto el uso de la información de realimentación o feedback del agente de fusión en los agentes sensores por medio del esquema de funcionamiento denominado fusión activa. La fusión activa permite mantener a todos los agentes del sistema cohesionados y hacer que las inconsistencias se resuelvan de una forma cooperativa, de forma que cada agente sensor se impone como objetivo el proporcionar una información coherente con el resto de la red. Este esquema de funcionamiento es especialmente útil en las redes de sensores visuales, donde pueden aparecer multitud de causas que provocan datos inconsistentes entre los sensores y se necesita de correcciones dinámicas. Los resultados experimentales obtenidos han mostrado la ventaja y el potencial de este esquema de fusión de datos cuando se aplica en redes de sensores visuales.

Finalmente, las pruebas realizadas también demuestran la mejora en el proceso de fusión a medida que se incrementa el número de sensores visuales utilizado debido a que se explota la redundancia de los datos.

7.3. Líneas de trabajo futuro

Una de las ventajas de los sistemas multi-agente es la capacidad de negociación de los agentes utilizando protocolos de comunicación de alto nivel. En este trabajo, se han sentado las bases para la negociación de coaliciones de agentes en el dominio de las redes de sensores visuales. Sin embargo, es posible utilizar mecanismos de negociación más complejos entre los agentes con distintos objetivos como por ejemplo aclarar situaciones de alto nivel. Estos mecanismos de negociación más complejos han quedado fuera del estudio de este trabajo.

En lo referente a la fusión de datos, en este trabajo únicamente se han tenido en consideración fuentes que generan información en forma de imágenes. Sin embargo, una línea de investigación muy activa actualmente es la multimodalidad, que consiste en la fusión de datos de diversas fuentes: audio, vídeo, texto y lenguaje natural; y en este contexto un sistema multi-agente también aporta las ventajas de la facilidad de integración, comunicación y razonamiento entre los diferentes componentes del sistema.

Algunos de los sensores visuales que se han utilizado para probar el sistema, son cámaras

PTZ, que tienen capacidad de moverse en los dos ejes y hacer zoom sobre las imágenes. Existen varios trabajos en los cuales el objetivo es realizar un seguimiento activo de los objetos en el entorno, estableciendo para ello la situación óptima de las cámaras PTZ. El proceso de negociar esta situación se puede llevar a cabo utilizando agentes sensores y protocolos específicos para ello.

Finalmente, la calibración que se ha llevado a cabo en la parte experimental de este trabajo ha consistido en una calibración de los sensores de forma previa a la ejecución; no obstante, es posible realizar la calibración de los sensores visuales de forma automática utilizando por ejemplo las posiciones de los objetos comunes que se detectan en el campo de visión de cada uno de los sensores visuales.

7.4. Publicaciones generadas

El desarrollo de este trabajo ha dado lugar a las siguientes publicaciones en congresos internacionales, revistas y capítulos de libro.

1. **F. Castanedo**, J. García, M.A. Patricio and J. M. Molina. *A Multi-agent Architecture Based On The BDI Model For Data Fusion In Visual Sensor Networks*. Journal of Intelligent and Robotic Systems. (Accepted).

2. **F. Castanedo**, J. García, M.A. Patricio and J. M. Molina. *Data Fusion to improve trajectory tracking in a Cooperative Surveillance Multi-Agent Architecture*. Information Fusion. `http://dx.doi.org/10.1016/j.inffus.2009.09.002`

3. **F. Castanedo**, M. A. Patricio, J. García, J. M. Molina. *Coalition of Surveillance Agents. Cooperative Fusion Improvement in Surveillance Systems*. ISBN 978-90-78677-10-9. 2009. `http://www.worldscibooks.com/compsci/l004.html`

4. M. A. Patricio, **F. Castanedo,** A. Berlanga, O. Pérez, J. García, and José M. Molina. *Computational Intelligence in Visual Sensor Networks. Improving Video Processing Systems*. Computational Intelligence in Multimedia Processing, Studies in Computational Intelligence. Springer. `http://www.springerlink.com/content/r474026vj408u11q/`

5. **F. Castanedo**, J. García, M. A. Patricio, J.M. Molina. *Designing a Visual Sensor Network Using a Multi-Agent Architecture*, 7th International Conference on Practical Applications of Agents and Multi-Agents Systems (PAAMS'09), Salamanca, March 2009. `http://www.springerlink.com/content/u44717285p042380/`

6. **F. Castanedo**, J. García, M. A. Patricio, J.M. Molina. *A Multi-Agent Architecture to Support Active Fusion in a Visual Sensor Network*, 2nd ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC'08), Stanford, September 2008. `http://dx.doi.org/10.1109/ICDSC.2008.4635715`

7. **F. Castanedo**, J. García, M. A. Patricio, J. M. Molina. *Analysis of Distributed Fusion Alternatives in Coordinated Vision Agents*, The 11th International Conference on Information Fusion (FUSION'08). Cologne, Germany, July 2008. `http://dx.doi.org/10.1109/ICIF.2008.4632357`

8. **F. Castanedo**, M. A. Patricio, J. García. *A BDI Multi-Agent Visual Sensor Network Architecture*, Workshop User-Centric Technologies and Applications, (CAEPIA'07). ISBN 978-84-611-8852-9. Salamanca. Spain.

9. **F. Castanedo**, M. A. Patricio, J. García, J. M. Molina. *Bottom-Up/Top-Down Coordination in a MultiAgent Visual Sensor Network*. Advanced Video and Signal Based Surveillance (AVSS'07). London, September 2007.
http://dx.doi.org/10.1109/AVSS.2007.4425292

10. **F. Castanedo**, M. A. Patricio, J. García, J. M. Molina. *Robust Data Fusion in a Visual Sensor Multi-Agent Architecture*. The 10th International Conference on Information Fusion (FUSION'07). Canada, July 2007.
http://dx.doi.org/10.1109/ICIF.2007.4408121

11. **F. Castanedo**, M. A. Patricio, J. García, J. M. Molina. *Coalition of Surveillance Agents. Cooperative Fusion Improvement in Surveillance Systems*. Sixth International Joint Conference on Autonomous Agents and Multiagent Systems. First Agent Based Ubiquitous Computing Workshop (ABUC). Hawaii, USA. May 2007.

12. **F. Castanedo**, M. A. Patricio, J. García, J. M. Molina. *Extending Surveillance Systems Capabilities Using BDI Cooperative Sensor Agents*. International Multimedia Conference. ACM MM. Proceedings of the 4th ACM international workshop on Video Surveillance and Sensor Networks. Santa Barbara, California, USA. 23-27 October 2006. http://doi.acm.org/10.1145/1178782.1178802

13. **F. Castanedo**, M.A. Patricio, J. M. Molina. *Evolutionary Computation Technique Applied to HSPF Model Calibration of a Spanish Watershed*. 7th International Conference on Intelligent Data Engineering and Automated Learning. Proceedings LNCS 4224. Burgos, Spain. 20-23 September 2006.
http://dx.doi.org/10.1007/11875581_26

Referencias

Abdel-Aty-Zohdy, H. S., & Ewing, R. L. (2000). Intelligent information processing using neural networks and genetic algorithms. *Proceedings of the 43rd IEEE Midwest Symposium on Circuits and Systems, 2000.*, *2*, 840–845.

Abdelbar, A. M., Andrews, E. A. M., & Wunsch, D. C. (2003). Abductive reasoning with recurrent neural networks. *Neural Networks*, *16*(5-6), 665–673.

Abreu, B., Botelho, L., Cavallaro, A., Douxchamps, D., Ebrahimi, T., Figueiredo, P., Macq, B., Mory, B., Nunes, L., Orri, J., et al. (2000). Video-based multi-agent traffic surveillance system. *Proceedings of the IEEE Intelligent Vehicles Symposium*, (pp. 457–462).

Aguilar-Ponce, R., Kumar, A., Tecpanecatl-Xihuitl, J. L., & Bayoumi, M. (2007). A network of sensor-based framework for automated visual surveillance. *Journal of Network and Computer Applications*, *30*(3), 1244–1271.

Aji, S. M., & McEliece, R. J. (2000). The generalized distributive law. *IEEE Transactions on Information Theory*, *46*(2), 325–343.

Alexander, H. (2005). State estimation for distributed systems with sensing delay. *Proceedings of SPIE*, *1470*, 103.

Alouani, A. T., Rice, T. R., & Auger, N. (1990). On the generalized nearest neighbor filter. *Twenty-Second Southeastern Symposium on System Theory*, (pp. 261–264).

APIDIS ([Online 03/2010]).
URL `http://www.apidis.org/Dataset/`

Ash, T. (1989). Dynamic node creation in backpropagation networks. *Connection Science*, *1*(4), 365–375.

Aziz, A., Tummala, M., & Cristi, R. (1999). Fuzzy logic data correlation approach in multisensor–multitarget tracking systems. *Signal Processing*, *76*(2), 195–209.

Bar-Shalom, Y. (2002). Update with out-of-sequence measurements in tracking: exact solution. *IEEE Transactions on Aerospace and Electronic Systems*, *38*(3), 769–777.

Bar-Shalom, Y., & Tse, E. (1975). Tracking in a cluttered environment with probabilistic data association. *Automatica*, *11*, 451–460.

Bashi, A. S., Jilkov, V. P., Li, X. R., & Computrols, I. (2003). Distributed implementations of particle filters. In *Proceedings of the Sixth International Conference of Information Fusion*, vol. 2, (pp. 1164–1171).

Basir, O., & Yuan, X. (2007). Engine fault diagnosis based on multi-sensor information fusion using Dempster–Shafer evidence theory. *Information Fusion*, *8*(4), 379–386.

Bentley, J. (1975). Multidimensional binary search trees used for associative searching. *Communications of the ACM*, *18*(9), 517.

Berge-Cherfaoui, V., & Vachon, B. (1991). A multi-agent approach of the multi-sensor fusion. In *Fifth International Conference on Advanced Robotics.'Robots in Unstructured Environments', ICAR.*, (pp. 1264–1269).

Berstel, B. (2002). Extending the rete algorithm for event management. *Ninth International Symposium on Temporal Representation and Reasoning.*, (pp. 49–51).

Besada, J. A., García, J., & Miguel, G. (2004). A new approach to on-line optimal estimation of multisensor biases. *IEE Proceedings. Radar, Sonar and Navigation*, *151*(1).

Biswas, P., Qi, H., & Xu, Y. (2008). Mobile-agent-based collaborative sensor fusion. *Information Fusion*, *9*(3), 399–411.

Blackman, S. (1990). Association and fusion of multiple sensor data. *Multitarget-Multisensor Tracking: Advanced Applications*, (pp. 187–217).

Blackman, S., & Popoli, R. (1999). Design and analysis of modern tracking systems (Book). *Norwood, MA: Artech House.*.

Boissier, O., & Demazeau, Y. (1994). MAVI: a multi-agent system for visual integration. *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, (pp. 731–738).

Borotschnig, H., Sinclair, D., & Pinz, A. (1997). Fuzzy graph tracking. In *Proc. 5th International Symposium on Intelligent Robotic Systems, Centre for Autonomous Systems, KTH Stockholm*, (pp. 91–101).

Bossé, É., Valin, P., Boury-Brisset, A., & Grenier, D. (2006). Exploitation of a priori knowledge for information fusion. *Information Fusion*, *7*(2), 161–175.

Brachman, R., & Levesque, H. (1984). The tractability of subsumption in frame-based description languages. In *Proc. of the 4th Nat. Conf. on Artificial Intelligence (AAAI-84)*, (pp. 34–37).

Bratman, M. (1987). *Intentions, Plans and Practical Reasoning*. Cambridge, Massachusetts: Harvard University Press.

Bratman, M., Israel, D., & Pollack, M. (1988). Plans and resource-bounded practical reasoning. *Computational Intelligence*, *4*(4), 349–355.

Breiman, L. (1996). Stacked regressions. *Machine Learning*, *24*(1), 49–64.

Brooks, R., & Iyengar, S. (1998). *Multi-sensor fusion: fundamentals and applications with software*. Prentice-Hall, Inc. Upper Saddle River, NJ, USA.

Brown, C., Durrant-Whyte, H., Leonard, J., Rao, B., & Steer, B. (1992). Distributed data fusion using Kalman filtering: A robotics application. In M. A. Abidi and R. C. Gonzalez, editors. *Data Fusion in Robotics and Machine Intelligence*, (pp. 267–309).

Buede, D., Logistics, D., & Reston, V. (1988). Shafer-Dempster and Bayesian reasoning: a response to Shafer-Dempster reasoning with applications to multisensor target identification systems. *IEEE Transactions on Systems, Man and Cybernetics.*, *18*(6), 1009–1011.

Busetta, P., Ronnquist, R., Hodgson, A., & Lucas, A. (1999). JACK intelligent agents-components for intelligent agents in Java. *AgentLink News Letter*, *2*, 2–5.

Cai, Q., & Aggarwal, J. (1996). Tracking human motion using multiple cameras. *Proceedings of ICPR*, *3*, 68–72.

Cai, Q., & Aggarwal, J. (1999). Tracking Human Motion in Structured Environments Using a Distributed-Camera System. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (pp. 1241–1247).

Castanedo, F., García, J., Patricio, M. A., & Molina, J. M. (2008a). Analysis of distributed fusion alternatives in coordinated vision agents. *11th International Conference on Information Fusion, 2008*.

Castanedo, F., García, J., Patricio, M. A., & Molina, J. M. (2008b). A multi-agent architecture to support active fusion in a visual sensor network. *Second ACM/IEEE International Conference on Distributed Smart Cameras. ICDSC 2008*.

Castanedo, F., García, J., Patricio, M. A., & Molina, J. M. (2010). Data fusion to improve trajectory tracking in a cooperative surveillance multi-agent architecture. *Information Fusion*, *11*(3), 243 – 255.

Castanedo, F., Patricio, M. A., García, J., & Molina, J. (2007a). Bottom-up/top-down coordination in a multiagent visual sensor network. *IEEE Conference on Advanced Video and Signal Based Surveillance. AVSS 2007*, (pp. 93–98).

Castanedo, F., Patricio, M. A., García, J., & Molina, J. M. (2007b). Robust data fusion in a visual sensor multi-agent architecture. *10th International Conference on Information Fusion, 2007*.

Challa, S., Evans, R. J., & Wang, X. (2003). A Bayesian solution and its approximations to out-of-sequence measurement problems. *Information Fusion*, *4*(3), 185–199.

Challa, S., & Legg, J. A. (2002). Track-to-track fusion of out-of-sequence tracks. *Proceedings of the Fifth International Conference on Information Fusion.*.

Chang, K., Chong, C., & Bar-Shalom, Y. (1986). Joint probabilistic data association in distributed sensor networks. *IEEE Transactions on Automatic Control.*, *31*(10), 889–897.

Chen, H. T., Lin, H. H., & Liu, T. L. (2001). Multi-object tracking using dynamical graph matching. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001.*, *2*, 210—217.

Chen, L., Cetin, M., & Willsky, A. S. (2005). Distributed Data Association for Multi-Target Tracking in Sensor Networks. In *7th International Conference on Information Fusion*.

Chen, L., Wainwright, M. J., Çetin, M., & Willsky, A. S. (2006). Data association based on optimization in graphical models with application to sensor networks. *Mathematical and Computer Modelling*, *43*(9-10), 1114 – 1135. Optimization and Control for Military Applications.

Cheng, Y., & Kashyap, R. (1988). Comparison of Bayesian and Dempster's rules in evidence combination. *Maximum-Entropy and Bayesian Methods in Science and Engineering, GJ Erickson and CR Smith, Eds. Klewer, Dordrecht, Netherlands*.

Chia, Y. S., & Huang, W. (2006). Multiple objects, tracking with multiple hypotheses dynamic updating. *Proc. IEEE International Conf. on Image Processing*, (pp. 569–572).

Chong, C. Y., Mori, S., & Chang, K. (1990). Distributed multitarget multisensor tracking. *Multitarget-Multisensor Tracking: Advanced Applications*, *1*, 247–295.

Chong, C. Y., Mori, S., & Chang, K. C. (1985). Information fusion in distributed sensor networks. In *American Control Conference, 4th, Boston, MA, June 19-21*.

Chow, C. K. (1965). Statistical independence and threshold functions. *IEEE Transactions on Electronic Computers*, *1*, 66–68.

Cicirelli, F., Furfaro, A., & Nigro, L. (2007). Exploiting agents for modelling and simulation of coverage control protocols in large sensor networks. *The Journal of Systems & Software*, *80*(11), 1817–1832.

Coates, M. (2004). Distributed particle filters for sensor networks. In *IPSN '04: Proceedings of the 3rd international symposium on Information processing in sensor networks*, (pp. 99–107). New York: ACM.

Cobb, B., & Shenoy, P. (2003). A Comparison of Bayesian and Belief Function Reasoning. *Information Systems Frontiers*, *5*(4), 345–358.

Collins, R. T., Lipton, A. J., Fujiyoshi, H., & Kanade, T. (2001). Algorithms for cooperative multisensor surveillance. In *Proceedings of the IEEE*, vol. 89, (pp. 1456–1477).

Corchado, J. M., & Laza, R. (2003). Constructing deliberative agents with case-based reasoning technology. *International Journal of Intelligent Systems*, *18*(12), 1227–1241.

Cou, C., Fraichard, T., Bessiere, P., & Mazer, E. (2002). Multi-sensor data fusion using Bayesian programming: an automotive application. *IEEE/RSJ International Conference on Intelligent Robots and System*.

Cox, I. J. (1993). A review of statistical data association techniques for motion correspondence. *International Journal of Computer Vision*, *10*(1), 53–66.

Cox, I. J., & Hingorani, S. L. (1996). An efficient implementation of reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *18*(2), 138–150.

Crisan, D., & Doucet, A. (2002). A survey of convergence results on particle filtering methods for practitioners. *IEEE Transactions on Signal Processing,[see also IEEE Transactions on Acoustics, Speech, and Signal Processing]*, *50*(3), 736–746.

Dang, V., Dash, R., Rogers, A., & Jennings, N. (2006). Overlapping Coalition Formation for Efficient Data Fusion in Multi-Sensor Networks. Twenty-First National Conference on Artificial Intelligence (AAAI-06).

Dasarathy, B. (1994). *Decision fusion*. Los Alamitos, California: IEEE Computer Society Press.

Dasarathy, B. (1997). Sensor fusion potential exploitation-innovative architectures and illustrative applications. *Proceedings of the IEEE*, *85*(1), 24–38.

Dasarathy, B. (2000). More the merrier... or is it? Sensor suite augmentation benefits assessment. *Proceedings of the Third International Conference on Information Fusion*.

DeAngelis, C. M., Whitney, J. E., Center, N. U. W., & Newport, R. I. (1998). The neurally-inspired contact estimator (NICE). *OCEANS'98 Conference Proceedings*.

Demazeau, Y., Boissier, O., Koning, J., & Lifia-Imag, G. (1994). Using interaction protocols to control vision systems. In *IEEE International Conference on Systems, Man, and Cybernetics. Humans, Information and Technology.*, vol. 2.

Dempster, A. (1968). A Generalization of Bayesian Inference. *Journal of the Royal Statistical Society. Series B (Methodological)*, *30*(2), 205–247.

Dennett, D. (1987). *The Intentional Stance*. Cambridge, Massachusetts: The MIT Press.

Detmold, H., Dick, A., Falkner, K., Munro, D., van den Hengel, A., & Morrison, R. (2006). Middleware for video surveillance networks. In *Proceedings of the international workshop on Middleware for sensor networks*, (p. 36). ACM.

D'Inverno, M., Luck, M., Georgeff, M., Kinny, D., & Wooldridge, M. (2004). The dMARS architecture: a specification of the distributed Multi-Agent reasoning system. *Autonomous Agents and Multi-Agent Systems*, *9*(1), 5–53.

Dockstader, S., & Tekalp, A. (2001). Multiple camera tracking of interacting and occluded human motion. *Proceedings of the IEEE*, *89*(10), 1441–1455.

Durfee, E. (1988). *Coordination of distributed problem solvers*. Kluwer Academic Publishers.

Durrant-Whyte, H. F., & Stevens, M. (2001). Data fusion in decentralized sensing networks. *Proceedings of the 4th International Conference on Information Fusion, Montreal, Canada*, (pp. 302–307).

Escamilla-Ambrosio, P. J., & Mort, N. (2001). A hybrid Kalman filter-fuzzy logic architecture for multisensor data fusion. *IEEE International Symposium on Intelligent Control, (ISIC'01).*, (pp. 364–369).

Escamilla-Ambrosio, P. J., & Mort, N. (2002). Multi-sensor data fusion architecture based on adaptive Kalman filters and fuzzy logic performance assessment. *Proceedings of the Fifth International Conference on Information Fusion*.

FIPA (2002). Fipa communicative act library specification.
URL `http://www.fipa.org/specs/fipa00037/`

Flynn, A. (1988). Combining sonar and infrared sensors for mobile robot navigation. *The International Journal of Robotics Research*, *7*(6), 5.

Forgy, C. L. (1991). Rete: A fast algorithm for the many pattern/many object pattern match problem. *IEEE Computer Society Reprint Collection*, (pp. 324–341).

Fortmann, T. E., Bar-Shalom, Y., & Scheffe, M. (1980). Multi-target tracking using joint probabilistic data association. *19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, *19*, 807–812.

Friedlander, D. S. (2005). Semantic Information Extraction. *Distributed Sensor Networks*.

Friedlander, D. S., & Phoha, S. (2002). Semantic information fusion for coordinated signal processing in mobile sensor networks. *International Journal of High Performance Computing Applications*, *16*(3), 235.

Ganeriwal, S., Kumar, R., & Srivastava, M. (2003). Timing-sync protocol for sensor networks. *Proceedings of the first international conference on Embedded networked sensor systems*, (pp. 138–149).

García, J., Molina, J. M., Besada, J. A., & Portillo, J. I. (2005). A multitarget tracking video system based on fuzzy and neuro-fuzzy techniques. *EURASIP Journal on Applied Signal Processing*, *14*, 2341–2358.

Garvey, T. D., Lowrance, J. D., & Fischler, M. A. (1995). An inference technique for integrating knowledge from disparate sources. *Multisensor integration and fusion for intelligent machines and systems*, (pp. 309–325).

Georgeff, M., & Lansky, A. (1987). Reactive reasoning and planning. *Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87)*, (pp. 677–682).

Georgeff, M., Pell, B., Pollack, M., Tambe, M., & Wooldridge, M. (1999). The Belief-Desire-Intention Model of Agency. *Proceedings of the 5th International Workshop on Intelligent Agents V: Agent Theories, Architectures, and Languages (ATAL'99)*, *1555*, 1–10.

Ghosh, J., & Holmberg, R. L. (1990). Multisensor fusion using neural networks. *Proceedings of the Second IEEE Symposium on Parallel and Distributed Processing, 1990.*, (pp. 812–815).

Goodman, I. R., Mahler, R. P. S., & Nguyen, H. T. (1997). *Mathematics of Data Fusion*. AA Dordrecht, The Netherlands: Kluwer Academic Publishers.

Graf, T., & Knoll, A. (2000). A Multi-Agent System Architecture for Distributed Computer Vision. *International Journal on Artificial Intelligence Tools*, *9*(2), 305–319.

Granger, C. (2001). Invited review: combining forecasts-twenty years later. *Essays in Econometrics: Collected Papers of Clive WJ Granger*, *8*, 167–173.

Gu, D. (2007). Distributed Particle Filter for Target Tracking. In *IEEE International Conference on Robotics and Automation*, (pp. 3856–3861).

Hall, D. L., & Llinas, J. (1997). An introduction to multisensor data fusion. *Proceedings of the IEEE*, *85*(1), 6–23.

Hall, D. L., & Llinas, J. (2001). *Handbook of MultiSensor Data Fusion*. Boca Raton: CRC Press.

Han, M., Sethi, A., Hua, W., & Gong, Y. (2004). A detection-based multiple object tracking method. *International Conference on Image Processing. ICIP'04.*, *5*.

Haritaoglu, I., Harwood, D., & David, L. S. (2000). W4: Real-time surveillance of people and their activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(8), 809–830.

Hashem, S. (1993). *Optimal Linear Combinations Of Neural Networks*. Ph.D. thesis, Purdue University.

Hashem, S. (1997). Algorithms for optimal linear combinations of neural networks. *International Conference on Neural Networks, 1997*.

Heikkila, J. (2000). Geometric camera calibration using circular control points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(10), 1066–1077.

Hernandez, M., Marrs, A., Maskell, S., & Orton, M. (2002). Tracking and fusion for wireless sensor networks. *Proceedings of the Fifth International Conference on Information Fusion*.

Herrero, J., Portas, J., & Corredera, J. (2007). On-line multi-sensor registration for data fusion on airport surface. *IEEE Transactions on Aerospace and Electronic Systems*, *43*(1), 356–370.

Hilton, R., Martin, D., & Blair, W. (1993). Tracking with Time Delayed Multisensor Systems. *Document Number NSWCDD/TR-9e/351, Naval Surface Warfare Center, Dahlgren*.

Hongeng, S., & Nevatia, R. (2001). Multi-agent event recognition. In *Eighth ICCV, 2001*, (pp. 84–91).

Huber, M. (1999). JAM: a BDI-theoretic mobile agent architecture. *Proceedings of the third annual conference on Autonomous Agents*, (pp. 236–243).

Iltis, R. A., & Ting, P. Y. (1991). Data association in multi-target tracking: a solution using a layered Boltzmann machine. *International Joint Conference on Neural Networks, IJCNN'91*.

Iltis, R. A., & Ting, P. Y. (1993). Computing association probabilities using parallel Boltzmann machines. *IEEE Transactions on Neural Networks.*, *4*(2), 221–233.

Ingrand, F. F., Georgeff, M. P., & Rao, A. S. (1992). An architecture for real-time reasoning and system control. *IEEE Expert,[see also IEEE Intelligent Systems and Their Applications]*, *7*(6), 34–44.

Intille, S. S., & Bobick, A. F. (1995). Closed-world tracking. *Proceedings of the Fifth International Conference on Computer Vision*, (pp. 672–678).

Iwaki, H., Srivastava, G., Kosaka, A., Park, J., & Kak, A. (2008). A novel evidence accumulation framework for robust multi-camera person detection. In *Second ACM/IEEE International Conference on Distributed Smart Cameras, ICDSC 2008*, (pp. 1–10).

Iyengar, S., Chakrabarty, K., & Qi, H. (2001). Introduction to special issue on "distributed sensor networks for real-time systems with adaptive configuration". *Journal of the Franklin Institute*, *338*(6), 651–653.

JASON ([Online 03/2010]). Jason.
URL `http://jason.sourceforge.net/`

JDL (1991). Data Fusion Lexicon. *Technical Panel for C3 (F.E. White, Code 4202, San Diego, CA)*.

JNI (1997). Java native method invocation specification v1.1.

Jo, C., Chen, G., & Choi, J. (2004). A new approach to the BDI agent-based modeling. *Proceedings of the 2004 ACM symposium on Applied computing*, (pp. 1541–1545).

Joo, S., & Chellappa, R. (2007). A Multiple-Hypothesis Approach for Multiobject Visual Tracking. *IEEE Transactions on Image Processing*, *16*(11), 2849–2854.

Jordan, M. (1998). *Learning in Graphical Models*. Cambridge, MA: The MIT Press.

Juditsky, A., & Nemirovski, A. (2000). Functional aggregation for nonparametric estimation. *Ann. Statist*, *28*(3), 681–712.

Julier, S., & Uhlmann, J. (1997a). A new extension of the Kalman filter to nonlinear systems. *Int. Symp. Aerospace/Defense Sensing, Simul. and Controls*, *3*.

Julier, S. J., & Uhlmann, J. K. (1997b). A non-divergent estimation algorithm in the presence of unknown correlations. In *Proceedings of the 1997 American Control Conference, 1997*, vol. 4.

Julier, S. J., & Uhlmann, J. K. (2005). Fusion of time delayed measurements with uncertain time delays. In *Proceedings of the American Control Conference.*, (pp. 4028–4033).

Julier, S. J., Uhlmann, J. K., & Nicholson, D. (2004). A method for dealing with assignment ambiguity. In *Proceedings of the American Control Conference.*, vol. 5.

Kacalenga, R., Erickson, D., & Palmer, D. (2003). Voting fusion for landmine detection. *IEEE Aerospace and Electronic Systems Magazine*, *18*(8), 13–19.

Kahan, J., & Rapoport., A. (1984). *Theories of Coalition Formation*. Hillsdale, New Jersey London: Lawrence Erlbaum Associatates Publishers.

Kalman, R. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, *82*(1), 35–45.

Ketchpel, S. (1994). Forming coalitions in the face of uncertain rewards. In *Proceedings of the National Conferece on Artificial Intelligence*, (pp. 414–419). Seattle, WA.

Kettnaker, V., & Zabih, R. (1999). Bayesian multi-camera surveillance. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2*, 253–259.

Khan, S., & Shan., M. (2003). Consistent labeling of tracked objects in multiple cameras with overlapping fields of view. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Khan, S. M., & Shah, M. (2009). Tracking multiple occluding people by localizing on multiple scene planes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *31*, 505–519.

Kinny, D. (1991). Commitment and effectiveness of situated agents. In *In Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91)*, (pp. 82–88).

Kinny, D., & Phillip, R. (2004). Building Composite Applications with Goal-Directed (TM) Agent Technology. *AgentLink News*, *16*, 6–8.

Klein, L., Co, H., & Fullerton, C. (1993). A Boolean algebra approach to multiple sensor voting fusion. *IEEE Transactions on Aerospace and Electronic Systems.*, *29*(2), 317–327.

Kumar, V., & Desai, U. (1996). Image Interpretation Using Bayesian Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *18*(1), 74–77.

Larsen, T. D., Andersen, N. A., Ravn, O., & Poulsen, N. K. (1998). Incorporation of time delayed measurements in a discrete-time Kalman filter. *Proceedings of the 37th IEEE Conference on Decision and Control*, *4*.

Lee, L., Romano, R., & Stein, G. (2000). Monitoring activities from multiple video streams: establishing a common coordinate frame. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(8), 758–767.

Lefebvre, E., & Helleur, C. (2002). Use of Fuzzy Logic for Data Fusion in a Recognized Maritime Picture. In *Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation (Proceedings of the 3rd WSES International Conference on Fuzzy Sets and Fuzzy Systems)*, (pp. 24–29).

Lesser, V. (1999). Cooperative multiagent systems: a personal view of the state of the art. *IEEE Transactions on Knowledge and Data Engineering*, *11*(1), 133–142.

Lesser, V., Ortiz, C. L., & Tambe, M. (2003). *Distributed Sensor Networks: A Multiagent Perspective*. London: Kluwer Academic Publishers.

Leung, H. (1996). Neural network data association with application to multiple-target tracking. *Optical Engineering*, *35*(3), 693–700.

Li, L., Huang, W., Gu, I. Y. H., & Tian, Q. (2003). Foreground object detection from videos containing complex background. In *ACMMM '03: Proceedings of the eleventh ACM international conference on Multimedia*, (pp. 2–10). New York, NY, USA: ACM.

Liggins, M. E., Chong, C. Y., Kadar, I., Alford, M. G., Vannicola, V., & Thomopoulos, S. (1997). Distributed fusion architectures and algorithms for target tracking. *Proceedings of the IEEE*, *85*(1), 95–107.

Lucas, A., & Goss, S. (1999). The potential for intelligent software agents in defence simulation. *Information, Decision and Control, 1999. IDC 99. Proceedings. 1999*, (pp. 579–583).

Luo, R., & Kay, M. (1992). Data fusion and sensor integration: State-of-the-art 1990s. *Data Fusion in Robotics and Machine Intelligence*, (pp. 7–135).

Luo, R. C., Yih, C. C., & Su, K. L. (2002). Multisensor fusion and integration: approaches, applications, and future research directions. *IEEE Sensors Journal*, *2*(2), 107–119.

Maes, P. (1990). Situated agents can have goals. *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back*, (pp. 49–70).

Mahalanobis, P. (1936). On the generalized distance in statistics. In *Proc. Nat. Inst. Sci. India*, vol. 2, (pp. 49–55).

Mallick, M., Coraluppi, S., & Carthel, C. (2001). Advances in asynchronous and decentralized estimation. *Aerospace Conference, 2001, IEEE Proceedings.*, *4*.

Mallick, M., Krant, J., & Bar-Shalom, Y. (2002). Multi-sensor multi-target tracking using out-of-sequence measurements. *Proceedings of the Fifth International Conference on Information Fusion, 2002.*, *1*.

Manyika, J., & Durrant-Whyte, H. (1995). *Data Fusion and Sensor Management: A Decentralized Information-Theoretic Approach*. Upper Saddle River, NJ: Prentice Hall.

Manzo, M., Roosta, T., & Sastry, S. (2005). Time synchronization attacks in sensor networks. *Proceedings of the 3rd ACM workshop on Security of ad hoc and sensor networks*, (pp. 107–116).

Marchesotti, L., Piva, S., & Regazzoni, C. (2003). An agent-based approach for tracking people in indoor complex environments. *12th International Conference on Image Analysis and Processing, 2003*, (pp. 99–102).

Martinez-del Rincon, J., Orrite-Urunuela, C., & Herrero-Jaraba, J. (2007). An efficient particle filter for color-based tracking in complex scenes. *IEEE Conference on Advanced Video and Signal Based Surveillance, AVSS 2007*, (pp. 176–181).

Mas, A., Belmonte, M., Mestras, J., de la Cruz, J., & Garijo, F. (2005). *Agentes software y sistemas multiagente: Conceptos, arquitecturas y aplicaciones*. Prentice Hall.

Mastrogiovanni, F., Sgorbissa, A., & Zaccaria, R. (2007). A distributed architecture for symbolic data fusion. In *Proceedings of IJCAI-07–International Joint Conference on Artificial Intelligence*.

Matsuyama, T. (1999). Cooperative Distributed Vision: Dynamic Integration of Visual Perception, Action, and Communication. *LNCS*, (pp. 75–88).

McLaughlin, S., Krishnamurthy, V., & Challa, S. (2003). Managing data incest in a distributed sensor network. In *IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings.(ICASSP'03)*, vol. 5.

Medioni, G., Cohen, I., Brémond, F., Hongeng, S., & Nevatia, R. (2001). Event detection and analysis from video streams. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *23*(8), 873–889.

Mills, D. (1991). Internet time synchronization: the network time protocol. *Communications, IEEE Transactions on*, *39*(10), 1482–1493.

Miranker, D. P. (1987). TREAT: A better match algorithm for AI production systems. In *Proceedings of the National Conference on Artificial Intelligence*, (pp. 42–47).

Mittal, A., & Davis, L. (2003). M2 Tracker: A Multi-View Approach to Segmenting and Tracking People in a Cluttered Scene. *International Journal of Computer Vision*, *51*(3), 189–203.

MODEST (1998). Multimedia Object Descriptions Extaction from Surveillance Types. http://www.tele.ucl.ac.be/PROJECTS/MODEST/index.html.

Molina, J. M., García, J., Jiménez, F. J., & Casar, J. R. (2004). Fuzzy reasoning in a multiagent system of surveillance sensors to manage cooperatively the sensor-to-task assignment problem. *Applied Artificial Intelligence*, *18*(8), 673–711.

Molina, J. M., García, J., Jiménez, F. J., & Casar, J. R. (2003). Cooperative management of a net of intelligent surveillance agent sensors. *International Journal of Intelligent Systems*, *18*(3), 279–307.

Murray, G., Steuart, S., Appla, D., McIlroy, D., Heinze, C., Cross, M., Chandran, A., Raszka, R., Tidhar, G., Rao, A., et al. (1995). The challenge of whole air mission modelling. *Proceedings of the Australian Joint Conference on Artificial Intelligence*.

Murty, K. (1968). An algorithm for ranking all the assignments in order of increasing cost. *Operations Research*, *16*(3), 682–687.

Nettleton, E., & Durrant-Whyte, H. (2001). Delayed and asequent data in decentralised sensing networks. *Proc. SPIE Conf*, *4571*, 255–266.

Norling, E. (2004). Folk psychology for human modelling: extending the BDI paradigm. *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems. AAMAS 2004.*, (pp. 202–209).

Novák, V., Perfilieva, I., & Močkoř, J. (1999). *Mathematical Principles of Fuzzy Logic*. Boston, MA: Kluwer Academic Publishers.

OpenCV ([Online 04/2010]). http://opencv.willowgarage.com/wiki/.

Orton, M., & Marrs, A. (2001). A Bayesian approach to multi-target tracking and data fusion with out-of-sequence measurements. *Target Tracking: Algorithms and Applications (Ref. No. 2001/174), IEE*, *1*.

Orwell, J., Massey, S., Remagnino, P., Greenhill, D., & Jones, G. A. (1999). A multi-agent framework for visual surveillance. In *ICIAP '99: Proceedings of the 10th International Conference on Image Analysis and Processing*, (p. 1104). Washington, DC, USA: IEEE Computer Society.

Pan, H., Liang, Z. P., Anastasio, T. J., & Huang, T. S. (1998). A hybrid NN-Bayesian architecture for information fusion. *International Conference on Image Processing, ICIP 98*, *1*.

Paskin, M., Guestrin, C., & McFadden, J. (2005). A robust architecture for distributed inference in sensor networks. In *Fourth International Symposium on Information Processing in Sensor Networks. IPSN 2005*, (pp. 55–62).

Patricio, M. A., Carbo, J., Perez, O., García, J., & Molina, J. M. (2007). Multi-agent framework in visual sensor networks. *EURASIP Journal on Advances in Signal Processing*, (pp. 1–21).

Pavlidis, I., Morellas, V., Tsiamyrtzis, P., & Harp, S. (2001). Urban surveillance systems: from the laboratory to the commercial world. *Proceedings of the IEEE*, *89*(10), 1478–1497.

Pavón, J., Gómez-Sanz, J., Fernández-Caballero, A., & Valencia-Jiménez, J. (2007). Development of intelligent multisensor surveillance systems with agents. *Robotics and Autonomous Systems*, *55*(12), 892–903.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo, CA: Morgan Kaufmann.

Peirce, C. (1955). Abduction and induction. *Philosophical Writings of Peirce*, *156*.

PETS2006 ([Online 03/2010]).
URL `http://www.pets2006.net`

Pokahr, A., Braubach, L., & Lamersdorf, W. (2003). Jadex: Implementing a bdi infraestructure for jade agents. *Search of Innovation (Special Issue on JADE)*, *3*(3), 76–85.

Provan, G. (1992). The validity of Dempster-Shafer belief functions. *International Journal of Approximate Reasoning*, *6*(3), 389–399.

Quaritsch, M., Kreuzthaler, M., Rinner, B., Bischof, H., & Strobl, B. (2007). Autonomous Multi-camera Tracking on Embedded Smart Cameras. *EURASIP Journal on Embedded Systems*, *2007*(1), 35–35.

Raiffa, H. (1984). *The Art and Science of Negotiation*. Cambridge, MA: Harvard Univ. Press.

Rao, A. (1996). AgentSpeak (L): BDI agents speak out in a logical computable language. *Lecture Notes in Computer Science*, *1038*, 42–55.

Rao, A., & Georgeff, M. (1991). Asymmetry thesis and side-effect problems in linear time and branching time intention logics. *Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91)*, (pp. 498–504).

Rao, A., & Georgeff, M. (1993). A model-theoretic approach to the verification of situated reasoning systems. *Thirteenth International Joint Conference on Artificial Intelligence (IJCAI-93)*, (pp. 318–324).

Rao, A., Georgeff, M., Technology, Commerce Department of Industry, A., & Institute, A. A. I. (1991). *Modelling Rational Agents Within a BDI-architecture*. Australian Artificial Intelligence Institute.

Rao, A., & Georgeff, M. P. (1992). An abstract architecture for rational agents. *Proceedings of Knowledge Representation and Reasoning (KR&R-92)*, (pp. 439–449).

Rao, A., Georgeff, M. P., & Sonenberg, E. (1992). Social plans: A preliminary report. *Decentralized AI*, *3*, 57–76.

Rao, A. S. (1993). Decision procedures for propositional linear-time belief-desire-intention logics. *Intelligent Agents*, *2*, 33–48.

Rao, A. S., Durrant-Whyte, H. F., & Sheen, J. A. (1993). A fully decentralized multi-sensor system for tracking and surveillance. *The International Journal of Robotics Research*, *12*(1), 20.

Rao, A. S., & Georgeff, M. P. (1995). Bdi agents: from theory to practice. (pp. 312–319).

Regazzoni, C., Ramesh, V., & Foresti, G. (2001). Scanning the issue/technology: Special issue on video communications, processing and understanding for third generation surveillance systems. *Proceedings of the IEEE*, *89*(10), 1355–1366.

Reid, D. (1979). An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, *24*(6), 843–854.

Remagnino, P., Jones, G. A., & Monekosso, N. (2001). Reasoning about dynamic scenes using autonomous agents. In *AI*IA 2001: Advances in Artificial Intelligence : 7th Congress of the Italian Association for Artificial Intelligence*. Bari, Italy.

Remagnino, P., Shihab, A., & Jones, G. (2004). Distributed intelligence for multi-camera visual surveillance. *Pattern Recognition*, *37*(4), 675–689.

Remagnino, P., Tan, T., & Baker, K. (1998). Agent orientated annotation in model based visual surveillance. *Sixth International Conference on Computer Vision, 1998.*, (pp. 857–862).

Ren, W., Beard, R. W., & Atkins, E. M. (2005). A survey of consensus problems in multi-agent coordination. In *Proceedings of the 2005 American Control Conference, 2005*, (pp. 1859–1864).

Romero Agüero, J., & Vargas, A. (2005). Inference of Operative Configuration of Distribution Networks Using Fuzzy Logic Techniques—Part II: Extended Real-Time Model. *IEEE Transactions On Power Systems*, *20*(3), 1562–1569.

Rosin, P. (1998). Thresholding for change detection. In *Sixth International Conference on Computer Vision.*, (pp. 274–279).

S. Khan, O. J., & Shah, M. (2001). Tracking in uncalibrated cameras with overlapping fields of view. In *Proc. IEEE Int'l Workshop Performance Evaluation of Tracking and Surveillance*, (pp. 84–91).

Salvi, J., Armangue, X., & Batlle, J. (2002). A comparative review of camera calibrating methods with accuracy evaluation. *Pattern recognition*, *35*(7), 1617–1635.

Sasiadek, J. Z., & Hartana, P. (2000). Sensor data fusion using Kalman filter. *Proceedings of the Third International Conference on Information Fusion*, *2*.

Sengupta, D., & Iltis, R. (1989). Neural solution to the multitarget tracking data association problem. *IEEE Transactions on Aerospace and Electronic Systems.*, *25*(1), 96–108.

Senior, A., Hampapur, A., Tian, Y., Brown, L., Pankanti, S., & Bolle, R. (2006). Appearance models for occlusion handling. *Image and Vision Computing*, *24*(11), 1233–1243.

Shafer, G. (1976). *A Mathematical Theory of Evidence*. Princeton, NJ: Princeton University Press.

Shajari, M., & Ghorbani, A. (2004). Application of Belief-Desire-Intention agents in intrusion detection and response. *Proceedings of Privacy, Security, Trust (PST'04) Conference*, (pp. 181–191).

Shehory, O., & Kraus, S. (1995). Feasible formation of stable coalitions among autonomous agents in general environments. *Computational Intelligence Journal*.

Smith, D., & Singh, S. (2006). Approaches to Multisensor Data Fusion in Target Tracking: A Survey. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, *18*(12), 1696–1710.

Smith, R. (1980). Communication and Control in Problem Solver. *IEEE Transactions on computers*, *29*, 12.

Snidaro, L., Foresti, G., Niu, R., & Varshney, P. (2004). Sensor Fusion for Video Surveillance. *Proceedings of the 7th International Conference on Information Fusion, Stockholm, Sweden*, (pp. 739–746).

Snidaro, L., Niu, R., Varshney, P., & Foresti, G. (2003). Automatic camera selection and fusion for outdoor surveillance under changing weather conditions. In *IEEE Conference on Advanced Video and Signal Based Surveillance*.

Stein, G. (1999). Tracking from multiple view points: Self-calibration of space and time. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, (pp. 521–527).

Stone, P., & Veloso, M. (1999). Task decomposition and dynamic role assignment for real-time strategic teamwork. *Artificial Intelligence*, *110*, 241–273.

Stone, P., & Veloso, M. (2000). Multiagent Systems: A Survey from a Machine Learning Perspective. *Autonomous Robots*, *8*(3), 345–383.

Stover, J. A., Hall, D. L., & Gibson, R. E. (1996). A fuzzy-logic architecture for autonomous multisensor data fusion. *IEEE Transactions on Industrial Electronics.*, *43*(3), 403–410.

Streit, R. L., & Luginbuhl, T. E. (1994). Maximum likelihood method for probabilistic multihypothesis tracking. *Proceedings of SPIE*, *2235*, 394.

Sycara, K., & Pannu, A. S. (1998). The retsina multiagent system (video session): towards integrating planning, execution and information gathering. In *AGENTS '98: Proceedings of the second international conference on Autonomous agents*, (pp. 350–351). New York, NY, USA: ACM Press.

Taylor, O., & MacIntyre, J. (1998). Adaptive local fusion systems for novelty detection and diagnostics in condition monitoring. In *Proceedings of SPIE*, vol. 3376, (p. 210).

Tenney, R., & Sandell, N. (1981). Detection with Distributed Sensors. *IEEE Transactions on Aerospace and Electronic Systems.*, (pp. 501–510).

Thomopoulos, S., & Zhang, L. (1988). Distributed filtering with random sampling and delay. *Decision and Control, 1988., Proceedings of the 27th IEEE Conference on*, (pp. 2348–2353).

Tsai, R. (1986). An efficient and accurate camera calibration technique for 3D machine vision. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, *374*.

Tsai, R. (1987). A versatile camera calibration technique for high accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal of Robotics and Automation*, *3*(4), 323–344.

Tsutsui, H., Miura, J., & Shirai, Y. (2001). Optical flow-based person tracking by multiple cameras. *International Conference on Multisensor Fusion and Integration for Intelligent Systems, 2001. MFI 2001.*, (pp. 91–96).

Uhlmann, J. K. (2003). Covariance consistency methods for fault-tolerant distributed data fusion. *Information Fusion*, *4*(3), 201–215.

Ukita, N., & Matsuyama, T. (2005). Real-time cooperative multi-target tracking by communicating active vision agents. *Computer Vision and Image Understanding*, *97*(2), 137–179.

Urlings, P., Sioutis, C., Tweedale, J., Ichalkaranje, N., & Jain, L. (2006). A future framework for interfacing BDI agents in a real-time teaming environment. *Journal of Network and Computer Applications*, *29*(2), 105–123.

Utsumi, A., Mori, H., Ohya, J., & Yachida, M. (1998). Multiple-view-based tracking of multiple humans. *14th International Conference on Pattern Recognition*.

Valera, M., & Velastin, S. (2005). Intelligent distributed surveillance systems: a review. *152*, 192–204.

Varshney, P. K. (1997). *Distributed Detection and Data Fusion*. New York: Springer.

Veenman, C. J., Reinders, M. J. T., & Backer, E. (2001). Resolving Motion Correspondence for Densely Moving Points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *23*(1), 54–72.

Von Neumann, J. (1956). Probabilistic logics and the synthesis of reliable organisms from unreliable components. *In Shannon, C. E. and McCarthy, J., editors, Automata Studies. Princeton University Press.*, (pp. 43–48).

Waltz, E., & Llinas, J. (1990). *Multisensor Data Fusion*. Norwood, Massachussets, U.S: Artech House Inc.

Walzer, K., Groch, M., & Breddin, T. (2008). Time to the Rescue–Supporting Temporal Reasoning in the Rete Algorithm for Complex Event Processing. *Lecture Notes in Computer Science*, *5181*, 635–642.

Wan, E. A., & Van Der Merwe, R. (2000). The unscented kalman filter for nonlinear estimation. *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC.*, (pp. 153–158).

Weiss, Y., & Freeman, W. T. (2001). On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. *IEEE Transactions on Information Theory*, *47*(2), 736–744.

Welch, G., & Bishop, G. (2001). An Introduction to the Kalman Filter. *ACM SIGGRAPH 2001 Course Notes*.

Weng, J., Cohen, P., & Herniou, M. (1992). Camera calibration with distortion models and accuracy evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *14*(10), 965–980.

Whitehouse, K., Liu, J., & Zhao, F. (2006). Semantic Streams: a framework for composable inference over sensor data. *The Third European Workshop on Wireless Sensor Networks (EWSN), Springer-Verlag Lecture Notes in Computer Science, February*.

Whyte, D. (1988). H F. Sensor models and multisensor integration. *International Journal Robotic Research*, *7*(6), 97–113.

Winter, M., Favier, G., & CNRS, S. (1999). A neural network for data association. *IEEE International Conference on Acoustics, Speech, and Signal Processing. ICASSP'99. Proceedings.*, *2*.

Wooldridge, M. (2000). *Reasoning about Rational Agents*. Cambridge, Massachusetts: The MIT Press.

Wooldridge, M., & Jennings, N. (1995). Intelligent agents: Theory and practice. *The knowledge Engineering Review*.

Wu, H., Siegel, M., & Ablay, S. (2003). Sensor fusion using Dempster-Shafer theory II: static weighting and Kalman filter-like dynamic weighting. *IEEE International Measurement Technology Conference, IMTC'03*, *2*.

Wu, H., Siegel, M., Stiefelhagen, R., & Yang, J. (2002). Sensor Fusion Using Dempster-Shafer Theory. *IEEE International Measurement Technology Conference, IMTC'02*, (pp. 7–12).

Yang, D. B., Gonzalez-Banos, H. H., & Guibas, L. J. (2003). Counting people in crowds with a real-time network of simple image sensors. In *Proceedings. Ninth IEEE International Conference on Computer Vision, 2003.*, (pp. 122–129).

Zhang, K. S., Li, X. R., & Zhu, Y. M. (2002). Optimal Update with Out of Sequence Updates for Distributed Filtering. *Proceedings. Fifth International Conference on Information Fusion, July*.

Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence*, *22*(11), 1330–1334.

Zhou, Q., Parrott, D., Gillen, M., M. Chelberg, D., & Welch, L. (2004). Agent-based computer vision in a dynamic, real-time environment. *Pattern Recognition*, *37*(4), 691–705.

Zhu, Q., Aldridge, S., & Resha, T. (2007). Hierarchical Collective Agent Network (HCAN) for efficient fusion and management of multiple networked sensors. *Information Fusion*, *8*(3), 266–280.