INGENIERÍA INFORMÁTICA

Escuela Técnica Superior de Ingeniería Informática

Curso académico 2009-2010

Proyecto Fin de Carrera

Seguimiento visual de personas mediante evolución de primitivas volumétricas

Tutor: José María Cañas Plaza

Autor: Sara Marugán Alonso

"Son vanas y están plagadas de errores las ciencias que no han nacido del experimento, madre de toda certidumbre."

Leonardo Da Vinci

Agradecimientos.

En primer lugar debo dar las gracias a mi tutor José María por la confianza que ha puesto en mí en todos los proyectos que hemos compartido. Le agradezco su esfuerzo y dedicación empleados en este proyecto en concreto y finalmente quiero reconocerle el papel importante que ha supuesto en mi formación como ingeniera.

A la gente del laboratorio de Robótica, en el que hemos pasado unas "cuantas horas", le agradezco el buen ambiente de trabajo y su ayuda en los experimentos del proyecto, especialmente a Julio y a Edu.

También quiero agradecer a mi familia todo el interés y apoyo que he recibido durante estos años, permitiéndome continuar con mi formación.

A Juan Antonio quiero agradecerle el que haya estado siempre ahí, apoyándome y comprendiéndome durante todos estos años.

Por último, pero no por ello menos importante, a mis amigos, por el interés mostrado, sus opiniones "no informáticas" que también vienen bien y sobre todo por permitirme desconectar del trabajo.

Resumen.

En los últimos años, los avances en la capacidad de cómputo de los ordenadores personales y el bajo coste de las cámaras han permitido el desarrollo de nuevas aplicaciones dentro de la visión artificial. Las cámaras son sensores que proporcionan gran cantidad de información sobre el entorno donde se utilizan. Extraer e interpretar esa información permite la creación de aplicaciones que realicen tareas útiles para nuestra vida cotidiana.

Este proyecto se enmarca dentro del desarrollo del proyecto *Eldercare*, cuya finalidad es construir una aplicación robusta de asistencia y ciudado de mayores en sus propios hogares utilizando visión artificial. El objetivo principal de este proyecto es crear un sistema capaz de estimar de forma continuada la posición tridimensional, el tamaño y el color de la vestimenta de múltiples personas dentro de una habitación de gran volumen.

En el desarrollo del sistema se han probado dos alternativas.
La primera consiste en generar la localización 3D apoyándose en el seguimiento visual de las personas en la imagen. De este modo primero se analizan las imágenes por separado, en las cuales se realiza un seguimiento a través de un algoritmo evolutivo multimodal. Este algoritmo utiliza información de *movimiento*, *color*, *puntos significativos* y *continuidad espacial* en la imagen. Sobre el resultado del seguimiento en cada imagen, se generan gran cantidad de hipótesis 3D de forma abductiva, que una vez evaluadas se seleccionan las mejores como resultado del seguimiento 3D.
La segunda alternativa consiste en un algoritmo evolutivo multimodal con individuos en 3D que son evaluados directamente sobre las imágenes. Los individuos pertenecen a *razas 3D*. Cada raza se encarga de seguir a una persona, estimando su posición 3D, tamaño y color de la vestimenta.

Tras comparar ambas técnicas en diferentes aspectos, el sistema escogido ha sido el *algoritmo evolutivo multimodal con hipótesis en 3D*. El sistema es suficientemente *vivaz* para realizar un seguimiento tridimensional de varias personas en movimiento, ofrece una *precisión centimétrica* y utiliza *hardware convencional*.

Índice general

Índice de figuras

Capítulo 1
Introducción

La vista es el sentido que más utilizamos para captar información de nuestro entorno y consiste en la habilidad de detectar la luz y de interpretarla. Tanto los humanos como los animales poseemos un sistema visual que nos permite crear un esquema de nuestro entorno y conocer detalles de los objetos que nos rodean. Gracias a estos detalles somos capaces de reconocer dichos objetos por su color, por su forma, detectar movimiento en ellos o incluso estimar aproximadamente la distancia que nos separa.

En los últimos años, un área activa de investigación es la reproducción de estas habilidades mediante sistemas informáticos. Las cámaras son sensores de bajo coste cada vez más comunes en nuestra vida cotidiana. Las tenemos en los móviles, los ordenadores, los portátiles, etc. Esta proliferación junto con el aumento de la capacidad de cómputo de los ordenadores de sobremesa han avivado el interés en la visión por computador.

Este proyecto fin de carrera es una aplicación directa de la localización visual, consiste en un sistema automático que permite seguir a varias personas en el interior de una habitación y estimar sus posiciones 3D y tamaño gracias a la información extraída de las imágenes que se reciben de varias cámaras.

En este primer capítulo se describe el contexto en el que se encuadra el sistema: la visión artificial de modo genérico y como contexto particular la localización y el seguimiento visual.

1.1. Visión artificial

La visión artificial o visión computacional es el área de la inteligencia artificial que se dedica a extraer información de las imágenes con el fin de comprender y asimilar

lo que en ellas está sucediendo. La visión artificial tiene como objetivo interpretar las imágenes analizadas para obtener información relevante sobre elementos que en ellas aparecen.

A principios de los 90 comenzaron a aparecer ordenadores con velocidad de cómputo suficiente para procesar imágenes de forma ágil. Desde entonces, la visión computacional empezó a utilizarse para múltiples tareas y su desarrollo fue centrándose en problemas concretos.

Objetivos concretos y típicos de la visión artificial son:

- El procesamiento de imágenes a través de filtros, operaciones morfológicas, realzado, etc.
- La detección, segmentación, localización y reconocimiento de ciertos objetos en imágenes (por ejemplo, caras humanas).
- Registro de diferentes imágenes de una misma escena u objeto. Por ejemplo, hacer concordar un mismo objeto en diversas imágenes.
- Seguimiento de un objeto en una secuencia de imágenes.
- Generar un modelo tridimensional de la escena. Tal modelo podría ser usado por un robot para navegar por la escena, por ejemplo.
- Búsqueda de imágenes digitales por su contenido.

Estos objetivos se consiguen por medio de reconocimiento de patrones, aprendizaje estadístico, geometría proyectiva, procesado de imágenes, teoría de gráficos y técnicas inspiradas en otros campos. Por ejemplo la visión artificial cognitiva está muy relacionada con la psicología cognitiva y la computación biológica. Dependiendo del tipo de información disponible, estas técnicas posibilitan el desarrollo de sistemas para el manejo de información en 2D o en 3D.

Ejemplos de aplicaciones de la visión artificial en el terreno 2D son *reconocimientos faciales o de iris* (figura 1.1 (a)), *inspectores en la industria del empaquetado*, *identificación automática de matrículas* (figura 1.1 (b)) o *clasificadores de piezas en entornos industriales*. Una aplicación real de este tipo de sistemas se puede encontrar en la ciudad de Londres, controlando el acceso a su zona centro, ya que existe una prohibición de circular por dicha zona debido a problemas de congestión y de

contaminación que sufría la ciudad en el pasado. El elevado número de vehículos que siguen circulando por la ciudad hace imposible controlar el acceso manualmente. La solución más fácil y extensible es mediante la utilización de cámaras para obtener la matrícula de los vehículos y contrastarla con una base de datos que almacena la contabilidad de los conductores que han abonado las tasas.

Figura 1.1: Reconocimiento con visión artificial - Facial (a) y Matrículas (b).

Otra aplicación en auge es la creación de interfaces basadas en visión para que las personas interactúen con su ordenador o videoconsola. Un buen ejemplo es el periférico *Eyetoy* (figura 1.2), creado por London Studio para la PlayStation 2, que se trata de una cámara que permite que el jugador interactúe con lo que aparece en la pantalla.

Otro ejemplo es *Project Natal* de *Microsoft* para la videoconsola *Xbox 360*, un nuevo sistema de control de juegos en el que usamos nuestro cuerpo y voz para jugar.

Figura 1.2: (a) EyeToy de PlayStation 2 y (b) Proyecto Natal de Microsoft para Xbox 360

El sistema *Natal* se basa en una barra que incorpora una cámara y micrófono que se encargan de recoger todo lo que hacemos y decimos delante de ella. No solo sabe quiénes somos, lo que decimos (no sólo palabras clave) o qué hacemos, sino que conoce a qué distancia estamos del sistema. En 3D es capaz de analizar e intepretar todos los movimientos de nuestro cuerpo y la orientación de los mismos. El sistema se compone de una cámara RGB (que se encarga especialmente del reconocimiento de rostros), un sensor de profundidad, micrófono multidireccional y software propietario de *Microsoft*.

En la actualidad los avances en geometría proyectiva, pares estéreo y autocalibración han abierto la puerta a la extracción de *información tridimensional* de las imágenes. El uso de estas nuevas técnicas permite alcanzar nuevas cotas en el área de la visión artificial, pues la información tridimensional permite conocer mejor y con una mayor precisión el entorno. Por ejemplo es posible hoy día realizar *reconocimiento 3D* de rostros u objetos, que puede ser utilizado en el reconocimiento de personas mediante el estudio biométrico en áreas de seguridad, diseño infográfico para la industria del cine o de videojuegos, etc.

Figura 1.3: Reconstrucción 3D.

La *reconstrucción 3D* (figura 1.3 (a)) es un problema clásico dentro de la visión artificial y para su resolución típicamente se necesitan dos o más cámaras. La técnica clásica consiste en que en primer lugar se extraen los puntos relevantes de la escena, seguidamente se calculan sus homólogos y por último se triangula para generar la representación 3D del objeto. Otro tipo de reconstrucción 3D se realiza mediante una cámara de estéreo visión[1], que permite la adquisición de dos o tres imágenes (según

[1]http://www.ptgrey.com/index.asp

el modelo) simultáneamente y procesarlas desde la propia cámara para establecer una relación entre los puntos que conforman cada imagen. La información obtenida de esta relación son valores de profundidad (disparidad) del ambiente sensado. Después, con la información geométrica de la cámara y con el valor de disparidad se calcula la distancia a la que encuentran los puntos del entorno, obteniendo una nube de puntos 3D con su respectivo valor de color (figura 1.3 (b)).

1.2. Seguimiento 2D

Uno de los campos de la visión artificial es el campo del seguimiento visual 2D. En esta sección se realiza una breve presentación del estado actual en este campo de investigación.

Un ejemplo particular de seguimiento 2D es la técnica de *seguimiento de blobs*. En ella, el objeto a seguir es caracterizado como un agregado de píxeles, que constituyen el *blob*. Para ello es necesario una segmentación previa siguiendo algún criterio como el color, la textura o la presencia de movimiento. El *blob* se parametriza mediante su centroide, de forma que el seguimiento tendrá como objetivo ofrecer la nueva posición del centroide. El método es atractivo por su simplicidad, pero el éxito del seguimiento dependerá fuertemente de los resultados de la segmentación. Pueden encontrarse multitud de trabajos basados en esta técnica para diferentes aplicaciones [Allen *et al.*, 1991] [Stauffer y Grimson, 2000].

Figura 1.4: Seguimiento de blobs - (a) Segmentación por color, (b) Definición de blobs.

Otra técnica popular es el *seguimiento mediante patrones*. Se trata de buscar una correspondencia píxel a píxel entre una región de píxeles de un fotograma y los fotogramas sucesivos. Habitualmente se empleará alguna técnica de correlación de patrones. Las posiciones de máxima correlación encontradas para el patrón en cada fotograma proporcionan la trayectoria seguida por el objeto. En general, estos

métodos requieren mayor esfuerzo de cómputo que el anterior, ya que la correlación debe evaluarse para cada posible localización del patrón.

Seguimiento de contornos es una técnica que fue introducida originalmente por Kas y Witkin [Kass *et al.*, 1988] con un planteamiento de minimización de energía y posteriormente se han desarrollado diversas variantes [Blake y Isard, 1994]. Se trata de ajustar de forma dinámica (a cada fotograma) un modelo del contorno del objeto de imagen seguido. Este planteamiento tiene la ventaja de que puede ser menos costoso computacionalmente que los anteriores ya que sólo requiere de búsquedas de bordes a lo largo de líneas normales al contorno y estimación del nuevo contorno por mínimos cuadrados.

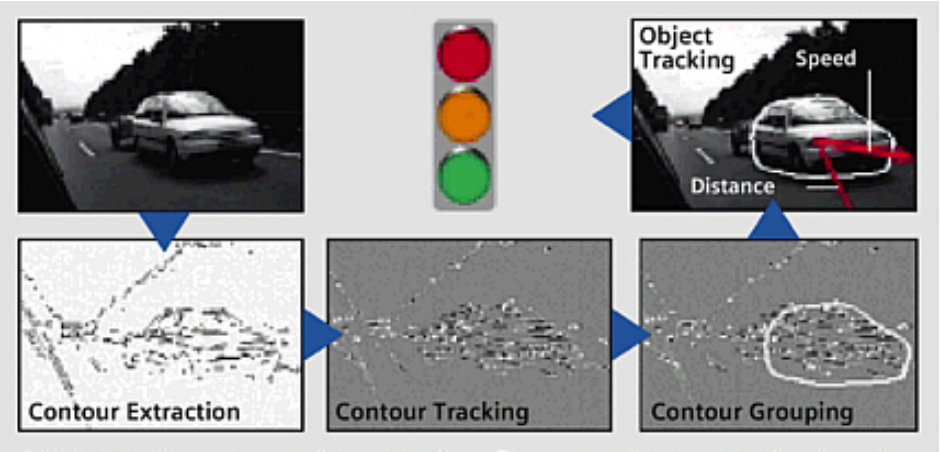

Figura 1.5: Seguimiento de contornos.

En algunos trabajos se recurre a caracterizar el elemento de imagen a seguir a partir del conocimiento a priori del que se dispone sobre su apariencia (*seguimiento basado en reconocimiento de formas*). Una posibilidad consiste en utilizar un conjunto de muestras del objeto de imagen para introducir técnicas de reconocimiento que permitan su identificación y localización en la imagen en los sucesivos fotogramas. Así, en el sistema de seguimiento de caras de Darrel et al. [Darrell *et al.*, 1998] se recurre a una red neuronal para discriminar cual de los *blobs* de color similar al de la piel humana detectados en la imagen corresponde realmente a la cara del usuario y no a otras partes de su cuerpo. El reconocedor debe ser previamente entrenado con muestras de caras y "no-caras".

Otro tipo de seguimiento es el *basado en puntos característicos*. Existen diferentes

técnicas para la detección y descripción de características locales en las imágenes. Estas características pueden ser los puntos de interés del objeto que se pretende seguir, de modo que obtiene una caracterización del objeto a través de estos puntos de interés (normalmente bordes, esquinas, etc.). Un ejemplo de este tipo de técnica es el algoritmo *Scale-Invariant feature transform (SIFT)* [Lowe, 1999]. Las características locales que extrae son invariantes a escala y rotación. Además, en gran medida son robustas a cambios de iluminación, ruido, oclusión y pequeños cambios en el punto de vista.

Figura 1.6: Emparejamiento de píxeles mediante la técnica SURF.

Otra técnica similar es *Speeded Up Robust Features (SURF)* [Bay *et al.*, 2008]. Está inspirada en parte en SIFT. La versión estándar de SURF es varias veces más rápida que SIFT y según sus autores, más robusta a ciertas transformaciones de la imagen que SIFT. SURF está basado en sumas de características *Haar* 2D y hace un uso eficiente de imágenes integrales.

Existen *métodos de búsqueda y optimización* que se aplican al campo del seguimiento visual en la imagen. Buenos ejemplos son el filtro de Kalman, los algoritmos genéticos y el filtro de partículas. Este último también es conocido como el Método secuencial de Montecarlo [Fox *et al.*, 1999], que fue propuesto por N. Gordon, D. Salmond y A. Smith y empleado para estimar el estado de un sistema que cambia a lo largo del tiempo.

1.3. Seguimiento 3D

Acercándonos al contexto particular de este proyecto fin de carrera, en esta sección se pretende dar una panorámica del estado del arte de otra de las tareas comunes en visión artificial: el seguimiento 3D de objetos. Existen dos tipos de soluciones a este problema: sistemas de una única cámara en movimiento y sistemas que emplean varias cámaras fijas calibradas en común.

Sistemas con una cámara

Los sistemas de una cámara están comenzando a ser populares en los últimos años. Hay muchos algoritmos para calcular la posición 3D de la cámara y posteriormente las posiciones de los objetos visibles en la imagen, éstos pueden ser divididos en:

- Métodos basados en un modelo 3D conocido. Calculan la transformación que aplica la cámara a partir de las proyecciones 2D del modelo 3D. Un ejemplo es el algoritmo POSIT [DeMenthon y Davis, 1995].

- Sistemas basados en marcadores. Consisten en introducir en la escena unos marcadores que el sistema pueda reconocer. Estos métodos son muy rápidos y exactos pero son también muy invasivos. Un ejemplo es la librería *ArToolkit*[2] desarrollada en la Universidad de Washington, que ha sido muy utilizada para sistemas de este tipo.

- Métodos basados en características. Estos métodos no necesitan ningún tipo de marcadores ni objetos conocidos en la escena. Sin embargo, son menos exactos que otros métodos y conllevan una carga computacional más elevada. Un ejemplo cercano de algoritmo *monoSlam* para calcular la posición 3D de la cámara es el proyecto fin de carrera *Autolocalización en tiempo real mediante seguiminto visual monocular* [López, 2010] desarrollado también en el grupo de Robótica de esta universidad y basado en un sistema similar [J.Davison *et al.*, 2007].

Sistemas con varias cámaras

En el caso de sistemas que emplean varias cámaras calibradas en común también se puede distinguir entre los métodos que utilizan marcadores y los que no los utilizan.

[2]http://www.hitl.washington.edu/artoolkit/

El ejemplo más conocido de sistema con marcadores en el sistema *Vicon*[3] (figura 1.7), empleado en distintos ámbitos tales como medicina (laboratorio de análisis de marcha), deporte o en la industria cinematográfica. Mediante el uso de los marcadores especiales situados sobre el cuerpo de una persona y varias cámaras filmando desde diferentes puntos de vista, un ordenador analiza todas las trayectorias y extrae un esqueleto 3D animado.

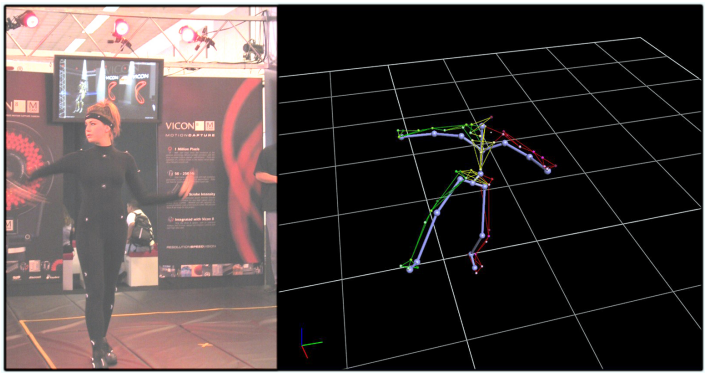

Figura 1.7: Sistema VICON.

En cuanto a los métodos sin marcadores, tenemos varias técnicas que suelen usarse sin marcadores aunque podrían utilizarlos. Son técnicas de estimación y seguimiento genéricas que se han aplicado en los últimos años al seguimiento 3D. Éstas son el filtro de partículas (Método secuencial de Montecarlo), el filtro de Kalman y los algoritmos genéticos o evolutivos.

1.4. Eldercare

Eldercare es un proyecto enfocado a la construcción de una aplicación robusta de asistencia y ciudado de mayores en sus propios hogares. La aplicación se apoya en la visión computacional para mantener localizados a una o más personas dentro de una estancia cubierta por dos o más cámaras. Esta localización se realiza en tres dimensiones, lo que permite detectar si una persona se ha caído al suelo y disparar algún tipo de alarma (sonora, visual, mensaje de texto, etc).

[3]http://www.vicon.com/

El primer antecedente de este proyecto es el proyecto fin de carrera *Aplicación de seguridad basada en visión* [Pineda, 2006], que consistió en realizar una aplicación de seguridad que permite localizar dentro de una habitación a un *único sujeto del cuál se conoce su color* y establecer un umbral en la coordenada Z para disparar la alarma. Si el sujeto cae al suelo, el sistema avisa con una alarma sonora y visual (figura 1.8).
Para el desarrollo de esa aplicación se emplearon dos técnicas distintas: un algoritmo evolutivo monomodal y una técnica probabilística denominada filtro de partículas, en ambos casos, utilizando una primitiva puntual (punto 3D).

Figura 1.8: Primera versión de Eldercare.

El siguiente proyecto en esta línea fue mi proyecto fin de carrera (figura 1.9) para el título de Ingeniero Técnico en Informática de Sistemas, *Seguimiento 3D visual de múltiples personas utilizando un algoritmo evolutivo multimodal*[Marugán, 2007], que conseguía el seguimiento de múltiples sujetos y aprender automáticamente el color de cada uno de ellos. Este nuevo sistema también se aplicó a la detección de caídas [Cañas *et al.*, 2008] [Cañas *et al.*, 2009].

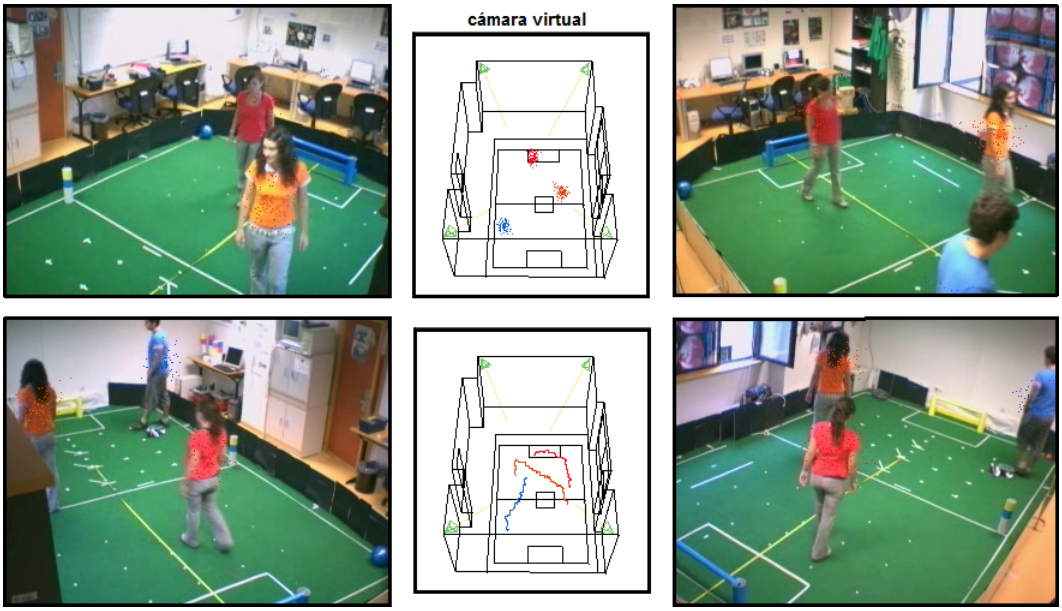

Figura 1.9: Segunda versión de Eldercare.

El proyecto actual pretende evolucionar el proyecto *Eldercare* a una tercera generación, con nuevos objetivos como: utilizar una *primitiva con volumen*, hacer más robusto el aprendizaje de color y explorar la alternativa de utilizar información bidimensional extraída del análisis individual de cada cámara para fusionarla y obtener así la localización tridimensional. Para este último objetivo se abordó previamente el problema del seguimiento visual en 2D, estudiando múltiples técnicas del estado del arte actual.

En el siguiente capítulo se exponen con detalle los objetivos y la planificación de este proyecto. Después, se presenta el entorno de desarrollo y el hardware utilizado para su realización. En el capítulo 4 se describe la aplicación de seguimiento 2D diseñada e implementada y los experimentos realizados. A continuación, en el capítulo 5, se detallan las técnicas se seguimiento 3D probadas y su comparativa. Por último, en el capítulo 6 se exponen las conclusiones y las vías futuras de continuación.

Capítulo 2
Objetivos

Una vez expuesto el contexto general y particular sobre el que se asienta este trabajo, estableceremos en este capítulo los objetivos concretos que se han planteado al igual que los requisitos que han condicionado el desarrollo del mismo.

2.1. Descripción del problema

El objetivo principal del proyecto es construir una aplicación capaz de estimar la localización y el tamaño de todas y cada una de las personas que se encuentran en una escena determinada, manteniendo un seguimiento 3D de ellas con primitivas volumétricas. Para ello se diseñarán e implementarán distintas técnicas de seguimiento visual.

Como se ha presentado en la introducción, el proyecto actual pretende evolucionar el proyecto *Eldercare* a una tercera generación. *Eldercare* es un proyecto enfocado a la construcción de una aplicación robusta de asistencia y ciudado de mayores en sus propios hogares.

El objetivo final se ha dividido en varios subobjetivos más específicos:

1. *Diseño e implementación de un seguidor 2D de personas.* En primer lugar se pretende abordar el seguimiento con una sola cámara. El seguidor 2D se apoyará en un segmentador de movimiento que le proporcionará las regiones de interés en forma de rectángulos. Para el seguimiento se probarán distintas técnicas, entre ellas un algoritmo evolutivo basado en [Marugán, 2007].

2. *Diseño e implementación de un seguidor 3D de personas mediante primitivas volumétricas.* Para este subobjetivo se implementarán dos alternativas: un seguidor 3D apoyado en varias instancias del seguidor 2D y un seguidor 3D

evolutivo. Se evaluarán las dos implementaciones y se determinará cuál es la más eficiente. La finalidad es ver si el seguimiento 2D en cada cámara aporta robustez al seguimiento 3D. Además, se pretende mejorar el aprendizaje de color respecto a la segunda versión de *Eldercare*.

3. *Experimentos*. Con la intención de caracterizar y ajustar el funcionamiento de la aplicación final de seguimiento 3D, se realizarán numerosas pruebas para asegurar y cumplir lo más fielmente posible los objetivos del proyecto.

2.2. Requisitos

El desarrollo del proyecto estará guiado por los objetivos comentados anteriormente y deberá ajustarse a los requisitos resumidos en los siguientes puntos:

1. Implementación del sistema apoyándose en la *plataforma software Jderobot*[1] sobre el sistema operativo *GNU/Linux*, concretamente la distribución Ubuntu.

2. Funcionamiento de la aplicación usando *hardware convencional*.

3. Validación del sistema en una *habitación de gran volumen*, en concreto el Laboratorio de Robótica de la URJC.

4. El software a desarrollar será *vivaz*, será capaz de seguir personas andando dentro de la habitación.

5. Seguimiento de varias personas con una *precisión centimétrica*.

2.3. Metodología

En esta sección se presenta la metodología utilizada para la realización de este proyecto. El proyecto ha sido desarrollado siguiendo el modelo de desarrollo en espiral basado en prototipos. La elección de este modelo se basa en la necesidad de separar el comportamiento final del sistema en varias subtareas que posteriormente serán integradas.

Cada subtarea se divide en un número determinado de ciclos. En cada uno de estos ciclos existen cuatro etapas: *Análisis de requisitos, Diseño e implementación, Pruebas y Planificación del próximo ciclo de desarrollo.*

[1] http://jde.gsyc.es/

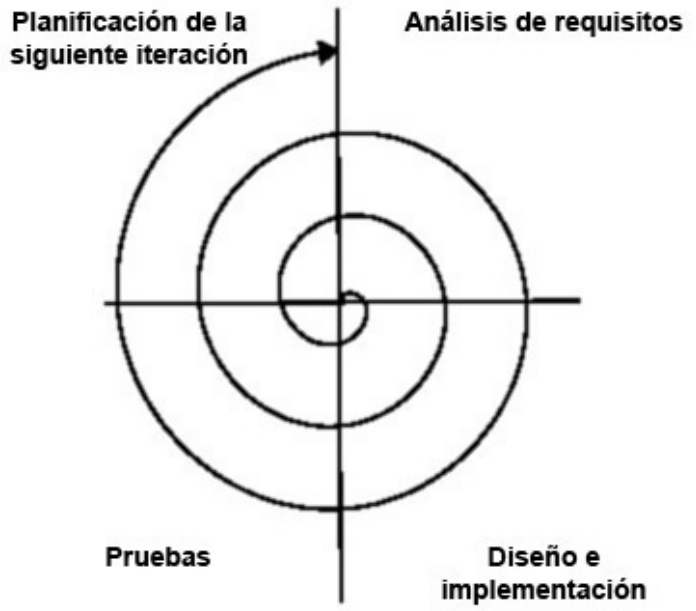

Figura 2.1: Modelo en espiral

1. *Análisis de requisitos.* Los objetivos de un ciclo de desarrollo deben ser identificados y especificados.

2. *Diseño e implementación.* Construcción del incremento de funcionalidad perteneciente al ciclo dado.

3. *Pruebas.* El sistema es validado con pruebas que verifican el correcto funcionamiento de acuerdo a los requisitos.

4. *Planificación del próximo ciclo de desarrollo.* Revisamos todo lo hecho, evaluándolo, y con ello decidimos si continuamos con las fases siguientes y planificamos el próximo ciclo.

Durante todo el desarrollo del proyecto se han mantenido reuniones semanales con el tutor para establecer y afinar los puntos que se han llevado a cabo en cada etapa, así como para comentar los resultados de etapas anteriores. A modo de bitácora, se ha ido plasmando la evolución del proyecto en un *mediawiki*[2] donde se pueden encontrar fotografías y vídeos de los experimentos realizados.

[2]http://jde.gsyc.es/index.php/Salons-ii

2.3.1. Plan de trabajo

El plan de trabajo seguido se ha dividido en diferentes fases, cuyos productos han consistido en prototipos o conocimiento adquirido.
Un prototipo es una versión preliminar de un sistema con fines de demostración o evaluación de ciertos requisitos. Se suele estimar la finalización de una iteración en el modelo de desarrollo como la obtención de un nuevo prototipo. Sin embargo, la obtención de un prototipo no implica que se utilice para el producto final, se puede desechar dicho prototipo.

1. *Estudio de la biblioteca OpenCV*[3]. En esta primera fase se exploraron las posibilidades que proporciona esta librería para el procesamiento de imágenes.

2. *Estudio del estado del arte*. En esta fase se analizó la bibliografía sobre seguimiento visual, para ello se leyeron diferentes artículos de investigación y proyectos fin de carrera.

3. *Diseño, implementación y validación de un segmentador de movimiento*. En esta fase se implementó un componente en la plataforma software *Jderobot*. Éste procesa un flujo de imágenes e idenfitica las regiones en las que se está produciendo un movimiento.

4. *Diseño, implementación y validación de un seguidor 2D de personas*. El prototipo obtenido en esta fase consiste en un seguidor de personas apoyado en el segmentador de movimiento de la fase anterior. El planteamiento consiste en desarrollar un seguimiento 2D por cada cámara, que permita acoplar posteriormente un seguimiento 3D. Además, Se probaron distintas técnicas de seguimiento 2D y se seleccionó una de ellas.

5. *Diseño, implementación y validación de un seguidor 3D de personas apoyado en el seguidor 2D*. En esta fase se construyó un prototipo que utilizaba cuatro instancias del seguidor 2D y con la información proporcionada estimaba la posición y el tamaño de las personas en la escena. Con este enfoque cada seguidor 2D obtiene su propia descripción de la misma persona en la escena, lo que es interesante debido a que, por ejemplo, el color de la vestimenta de una persona puede variar significativamente de una cámara a otra. De este modo no se parte de la premisa de que las observaciones deben ser iguales en las diferentes cámaras.

[3]http://sourceforge.net/projects/opencvlibrary/

6. *Diseño, implementación y validación de un seguidor 3D de personas con un algoritmo evolutivo.* En esta fase se implementó un algoritmo evolutivo con primitivas volumétricas para la localización tridimensional de las personas. El algoritmo utiliza los resultados del segmentador de movimiento como disparador del seguimiento.

7. *Comparativa entre las dos técnicas de seguimiento 3D desarrolladas.* Esta fase consistió en la evaluación de los dos prototipos de sistemas de seguimiento 3D y la selección de uno de ellos.

8. *Experimentos.* La última fase consistió en la realización de experimentos más exhaustivos sobre el algoritmo de seguimiento 3D seleccionado, con el objetivo de caracterizarlo y ajustarlo lo mejor posible.

Capítulo 3
Entorno de desarrollo

En este capítulo se presenta la plataforma de desarrollo y la infraestructura utilizada para el proyecto. La plataforma software utilizada ha sido *Jderobot*, desarrollada en el propio Grupo de Robótica de esta universidad.
Además, se han usado cuatro librerías: *GTK*, *OpenCV*, *SIFT* y *Progeo*. *GTK* se ha utilizado para la generación de la interfaz gráfica, *OpenCV* para el tratamiento de imágenes, *SIFT* para la detección y descripción de puntos característicos en una imagen y *Progeo* para las operaciones relacionadas con la información tridimensional.

3.1. Plataforma software Jderobot e infraestructura hardware

Jderobot[1] es una plataforma de software libre para la programación de aplicaciones que generan comportamientos autónomos en robots y/o que están relacionadas con la visión artificial y con la domótica.

Jderobot funciona sobre el sistema operativo GNU/Linux y dispone de una comunidad activa de desarrolladores, lo que garantiza incorporación continua de soporte para nuevos dispositivos hardware. *Jderobot* está desarrollada en lenguaje C, que supone un buen compromiso entre potencia y eficiencia. Además, posee una interfaz gráfica (figura 3.1) desde la que se pueden arrancar y parar los esquemas y sus interfaces gráficas. La versión de *Jderobot* utilizada para este proyecto es la 4.3.

[1] http://jde.gsyc.es/

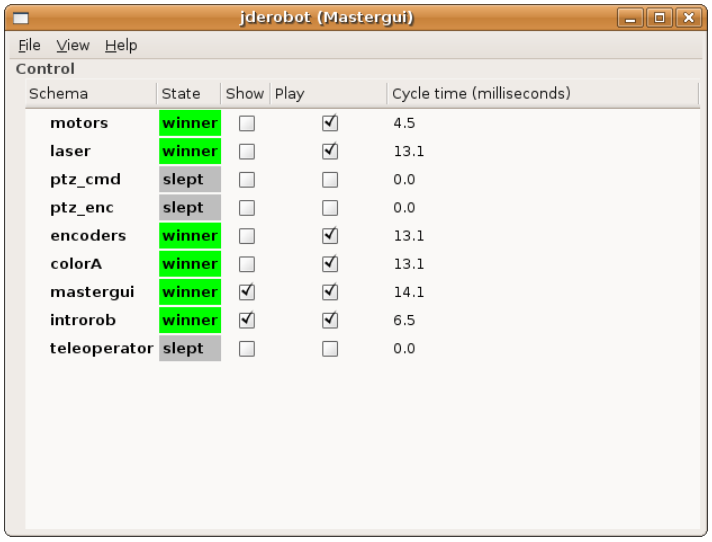

Figura 3.1: Interfaz gráfica de Jderobot.

La plataforma resuelve el problema del acceso a los sensores y actuadores, permitiendo que el programador se ocupe sólo de la inteligencia del sistema.
El acceso a los sensores es a través de variables que las aplicaciones leen y a los actuadores mediante variables en las que las aplicaciones escriben. Concretamente, estas variables son servidas por *drivers* de la plataforma y poseen un interfaz fijo.

En este proyecto se ha utilizado el interfaz *varcolor*, a través del cual se reciben imágenes con independencia de la fuente.

```
typedef struct {
  char *img;
  /* RGB order */
  unsigned long int clock;
  int width;
  int height;
} Varcolor;
```

Como vemos, el interfaz da acceso al *array* de datos de la imagen, al reloj o identificador de fotograma y al tamaño del mismo.

Los *drivers* y servicios utilizados para este proyecto han sido:

- *Firewire*: permite el acceso al flujo de imágenes de las cámaras *iSight* utilizadas.

- *Networkserver*: es un servicio que mediante una conexión *TCP/IP* ofrece todos los dispositivos que pueda tener conectados. En este caso, las imágenes de las cámaras.
- *Networkclient*: permite a *Jderobot* conectarse a través de la red a otro *Jderobot* que esté usando el servicio *networkserver* para servir algún dispositivo.
- *Mplayer*: este *driver* accede a las imágenes de un fichero de vídeo. Éste ha sido muy útil para la realización de experimentos.
- *Graphics_gtk*: es el servicio que conecta con el servidor gráfico para manejar las acciones sobre la interfaz gráfica.

El escenario de pruebas ha sido el Laboratorio de Robótica del Edificio Departamental II de esta universidad, con cuatro cámaras web *Apple iSight* (figura 3.2) a resolución 320x240 adheridas en el techo de las esquinas de la habitación.

Figura 3.2: Cámara web Apple iSight.

Cada cámara está conectada a un equipo y sus imágenes son servidas a través de *networkserver* por la red. El equipo principal donde se ejecuta la aplicación recibe esas imágenes mediante el *driver networkclient* y éste se las ofrece a la aplicación. Las características del equipo principal son: Pentium Quad-Core a 2.4 Ghz cada núcleo, 4 GB de RAM y sistema operativo GNU/Linux, concretamente la distribución Ubuntu.

En la figura 3.3 se puede ver un diagrama del montaje.

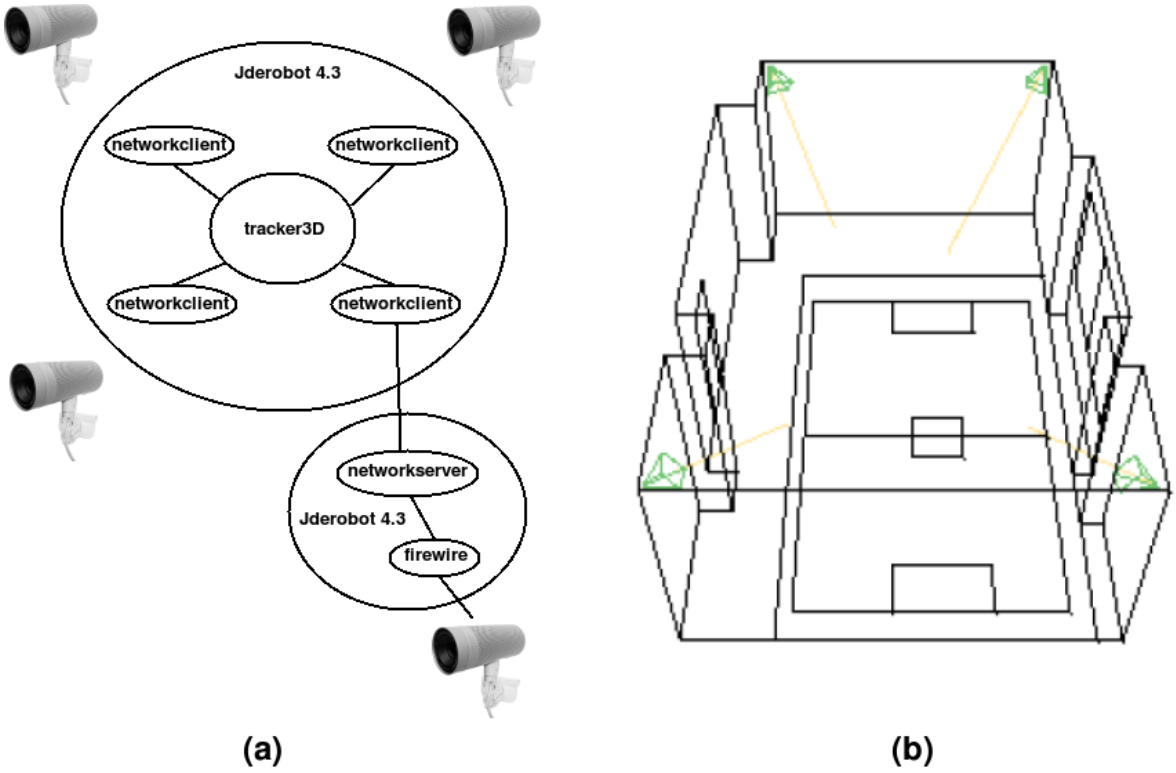

Figura 3.3: Montaje real de la infraestructura del proyecto - (a) Conexiones entre los *drivers* y servicios, (b) Imagen virtual de la habitación y las cámaras.

En cuanto a las aplicaciones en *Jderobot*, éstas se organizan en módulos denominados esquemas, que se organizan en jerarquías y que interactúan entre sí. Esta interación puede consistir en intercambiar datos o simplemente en activar o desactivar un esquema desde otro. Existen dos tipos fundamentales de esquemas: perceptivos y motores. Los perceptivos son aquellos que leen información de los sensores y la procesan, sin embargo los esquemas motores pueden también recibir información y usarla para tomar decisiones de actuación.

La aplicación desarrollada en este proyecto se ha dividido en una jerarquía de dos niveles: un esquema auxiliar que realiza un preproceso de las imágenes y un esquema principal que activa al primero y genera el comportamiento de seguimiento de personas.

3.2. GTK

La biblioteca *GTK*[2] (o Gimp Toolkit) es una de las que compone el conjunto *GTK+* y permite crear interfaces gráficas de usuario para los programas.

Proporciona una herramienta denominada *glade*[3] (figura 3.4), mediante la cuál de manera visual se pueden cear objetos gráficos en sistemas de ventanas. Estos objetos se denominan *widgets* y ejemplos son botones, etiquetas, barras de desplazamiento, etc.

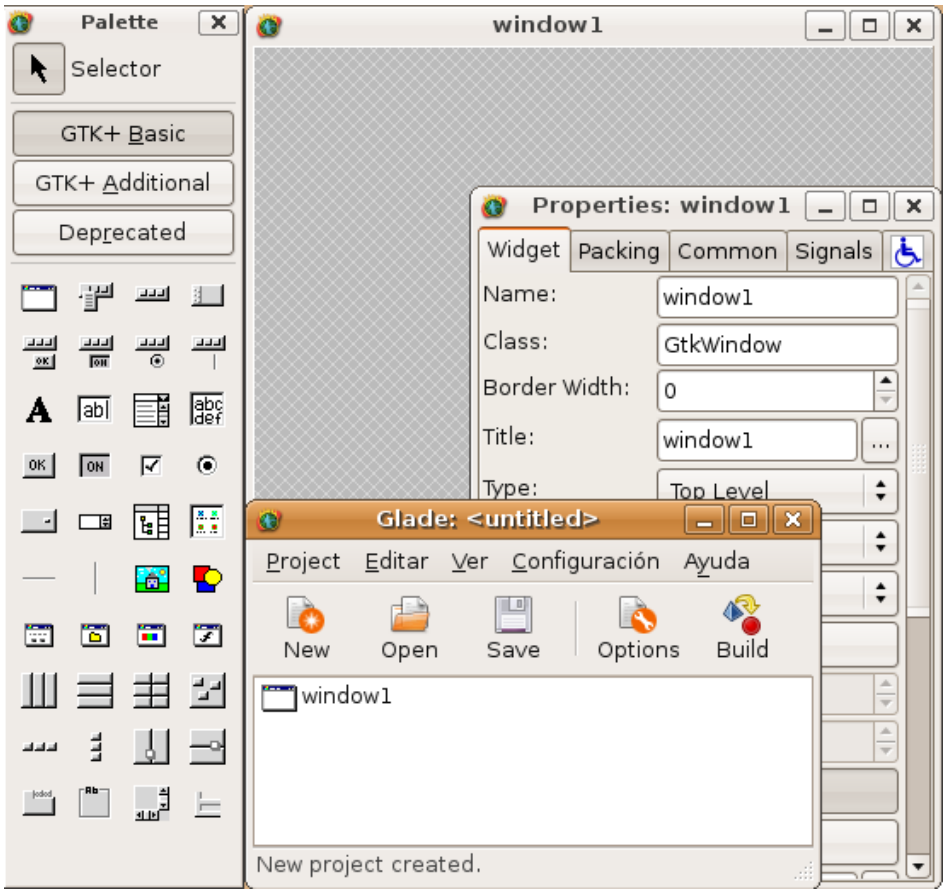

Figura 3.4: Glade.

Los eventos en *GTK* se materializan a través de señales asociadas a manejadores. *Glade* permite asociar las señales de determinado *widget* a sus funciones manejadoras o *callbacks*.

La interfaz de usuario de nuestra aplicación se ha diseñado con esta librería y el

[2]www.gtk.org/
[3]http://glade.gnome.org/

servicio de la plataforma *Jderobot* que soporta esta herramienta se llama *graphics_gtk*. Éste se encarga de la inicialización de la biblioteca y la creación de los hilos necesarios, así como de proveer la función para cargar el fichero *.glade*, que define la interfaz gráfica diseñada con *glade*.

3.3. Librería OpenCV

OpenCV[4] es una biblioteca originalmente desarrollada por Intel para visión artificial. La biblioteca es de código abierto y multiplataforma. Su versión actual y la utilizada para este proyecto es la 1.1.0.

La biblioteca ofrece algunos algoritmos habituales en la visión computacional y además una colección de tipos de datos de alto nivel como listas, matrices, árboles, imágenes, etc. Las funciones de *OpenCV* para el tratamiento de imágenes pueden clasificarse en:

- Procesado de imágenes: operaciones morfológicas, bordes, filtros, conversión, histogramas, emparejamiento, etc.
- Análisis estructural: geometría computacional, subdivisiones planares, etc.
- Detección de movimiento y seguimimento de objetos: acumulación de fondo, plantillas de movimiento, flujo óptico, estimadores, etc.
- Reconocimiento de patrones y detección de objetos.
- Calibración de cámaras y reconstrucción 3D.

Concretamente, en este proyecto se han utilizado funciones de conversión, de manejo de histogramas para el aprendizaje de color y operaciones morfológicas sobre imágenes, como la dilatación o la resta de imágenes. Además, se ha usado el estimador basado en Kalman y las funciones para el cálculo de flujo óptico.

OpenCV también ofrece funciones para la detección de características en las imágenes. *Speeded Up Robust Features* (SURF) es un robusto descriptor y detector de puntos de interés en una imagen (presentado en [Bay *et al.*, 2008]), del que *OpenCV* ha realizado una implementación a través de una colección de funciones. En este proyecto se han utilizado dichas funciones y se ha experimentado con este tipo de información

[4]https://code.ros.org/svn/opencv/trunk/opencv

para facilitar el seguimiento 2D de personas.

3.4. SIFT

Scale-Invariant Feature Transform (SIFT) es un algoritmo de visión computacional para detectar y describir características locales en imágenes.

El método fue presentado en [Lowe, 1999] y existe una implementación libre realizada por el doctorando Robert Hess[5]. Esta librería utiliza *OpenCV* y la biblioteca GSL[6]. Esta implementación ha sido la utilizada en este proyecto para el estudio de esta técnica como método de seguimiento 2D.

El algoritmo puede ser usado para el reconocimiento de objetos. Las características locales se basan en la apariencia del objeto en determinados puntos de interés y son invariantes a escala y rotación. Además, en gran medida son robustas a cambios de iluminación, ruido, oclusión y pequeños cambios en el punto de vista.

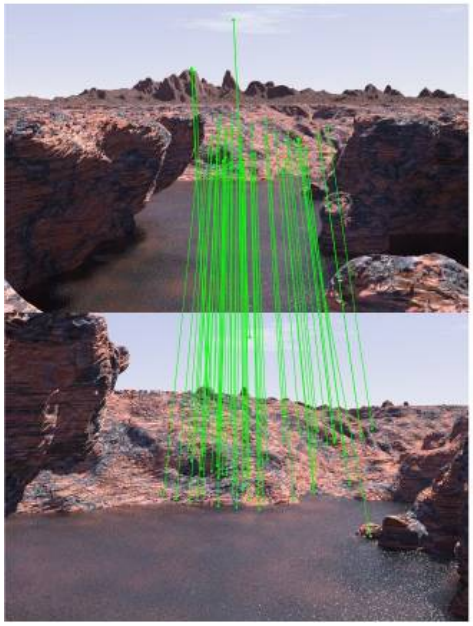

Figura 3.5: Emparejamiento mediante la técnica SIFT.

[5]http://web.engr.oregonstate.edu/ hess/
[6]http://www.gnu.org/software/gsl/

3.5. Progeo

La biblioteca de geometría proyectiva Progeo es utilizada en la aplicación para relacionar el mundo de las imágenes (2D) con el mundo real (3D). Esta relación se consigue gracias a dos funciones principales:

- *Proyectar*. Esta función permite realizar la proyección óptica de un punto 3D del espacio en el plano imagen de una de las cámaras, obteniendo así un punto 2D perteneciente a ese plano. Con ese punto 2D es posible obtener el píxel de la imagen correspondiente.
- Función *Retroproyectar*. Esta función permite obtener la recta de proyección que une el centro óptico de una cámara con el punto 3D que representa un punto 2D de su plano imagen. Ese punto 2D se corresponde con un píxel en la imagen de esa cámara y mediante esta función se puede llegar a obtener el punto 3D del que es proyección en el plano.

Progeo se basa en el *modelo de cámara Pinhole* para realizar estas transformaciones, que se puede observar en la figura 3.6.

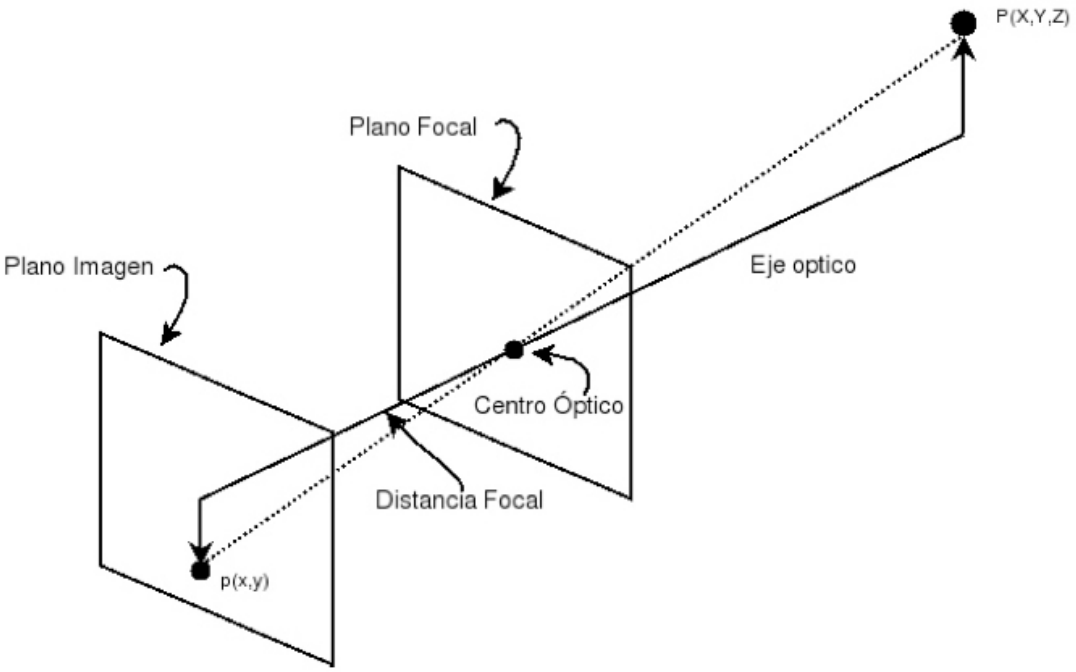

Figura 3.6: Modelo de cámara Pinhole

En este modelo se asume que cualquier punto $P(x, y, z)$ se proyecta en el plano de imagen a través de otro único punto llamado *Centro óptico*. La recta que une el punto P y el centro óptico se denomina *Línea de proyección* e intersecta al plano imagen justo en el píxel $p(x, y)$, que es la proyección de $P(X, Y, Z)$. El centro óptico está situado a la *Distancia focal* del plano imagen. Este modelo lo completan el *Eje óptico*, que es una línea perpendicular al plano imagen y que atraviesa al centro óptico, y también el *Plano focal*, que es el plano perpendicular al eje óptico cuyos puntos no se proyectan en el plano imagen, incluyendo al centro óptico.

Progeo proporciona además los tipos de datos *Punto2D* y *Punto3D* para la representación de puntos en el plano y en el espacio. También proporciona los tipos *CamaraPinhole* y *CamaraStereoPinhole*. Ambos tipos se utilizan para simular el uso de cámaras y pares estéreo en el entorno tridimensional.

Por tanto esta librería se ha utilizado para la generación de imágenes virtuales del mundo 3D y las operaciones necesarias para relacionar el mundo 3D con las imágenes 2D.

Un aspecto necesario para este proyecto ha sido la calibración de las cámaras. El proceso de calibración consiste en obtener los *parámetros intrínsecos* y *parámetros extrínsecos* de las mismas. En el caso de los parámetros intrínsecos es necesario conocer la *distancia focal* en cada eje y el tamaño del píxel central del plano imagen. Para los parámetros extrínsecos se necesita la posición 3D de la cámara y la orientación.

La obtención de los parámetros extrínsecos se ha realizado a través del esquema *extrinsics* de la plataforma *Jderobot*, que permite visualizar la imagen virtual de una cámara sobre la imagen real y modificar a través de su interfaz gráfica los parámteros extrínsecos hasta que ambas imágenes coincidan. Además, guarda la configuración seleccionada por cada cámara en un fichero de texto para cada una, que sirven como entrada a nuestra aplicación. Este esquema utiliza también la libería *Progeo*.

Capítulo 4
Seguimiento 2D

Una vez presentados los requisitos y las herramientas utilizadas en este proyecto, es interesante recordar que el objetivo del proyecto es el seguimiento tridimensional de personas dentro de una habitación. Una de las técnicas de seguimiento 3D diseñadas se apoya en el seguimiento 2D visual generado por cada una de las cámaras por separado. En este capítulo se aborda el subobjetivo concreto del seguimiento 2D de múltiples personas a través de una cámara. Se describe cómo se ha estructurado e implementado la aplicación de seguimiento 2D presentando en primer lugar el diseño global del seguidor 2D y después detallando los componentes de la misma. Además, en el desarrollo de la aplicación de seguimiento 2D se han probado diferentes alternativas, como por ejemplo el seguimiento de puntos característicos o el emparejamiento de regiones de interés. En una sección dedicada a ello se explicarán los experimentos relacionados con esto.

4.1. Diseño global

El sistema de una cámara propuesto tiene como finalidad resolver el seguimiento 2D de personas. Como aparece en el diagrama 4.1, la entrada consiste en un flujo de imágenes y la salida es una lista de personas. Las personas, a la salida, son descritas mediante regiones de interés rectangulares y un color característico.

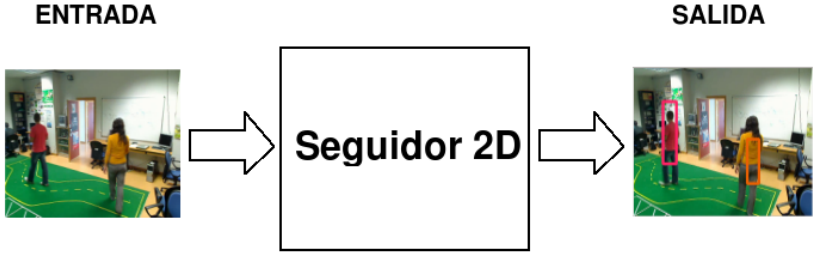

Figura 4.1: Entrada y salida de la aplicación de seguimiento 2D.

Describiéndolo a grandes rasgos, el seguimiento 2D es una adaptación del algoritmo evolutivo implementado en [Marugán, 2007] al problema del seguimiento dentro de una imagen. En este nuevo algoritmo la información extraída de las imágenes es de distintos tipos, concretamente referida a *movimiento*, *color*, *puntos significativos* y *continuidad espacial* en la imagen. Como se describe más adelante (sección 4.4), este tipo de algoritmo trabaja con hipótesis tentativas que van evolucionando hasta converger hacia soluciones reales.

La aplicación consta de varios esquemas. Los esquemas se pueden clasificar en dos niveles: en el primer nivel se extraen las regiones de interés en las imágenes y en el segundo nivel se realiza el seguimiento 2D de cada persona en la escena. Esta descomposición permite aprovechar el paralelismo *hardware* que ofrecen los ordenadores hoy en día y por consiguiente obtener mayor eficiencia.
En la figura 4.2 se presenta el diseño global del sistema y la relación entre los distintos esquemas.

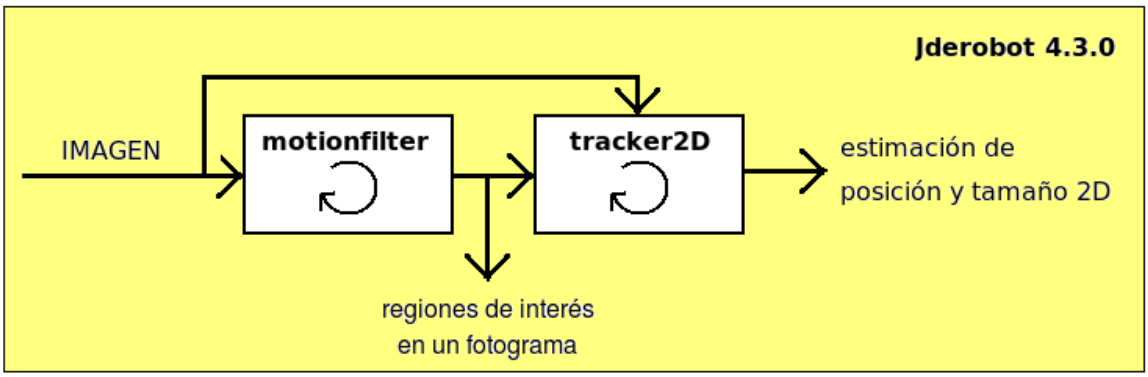

Figura 4.2: Diseño global de la aplicación de seguimiento 2D.

El primer procesado de las imágenes lo realiza el esquema **Motionfilter** y consiste en un filtro de movimiento. El movimiento se ha medido como diferencia de fotogramas consecutivos y contra el fondo aprendido. Este enfoque parte de que donde hay movimiento, se encuentra una persona moviéndose.
Tras finalizar el filtrado de movimiento, realiza una segmentación de aquellas zonas en las que se presupone que hay personas en movimiento. El objetivo es extraer de la imagen las partes interesantes para el algoritmo de seguimiento 2D, de modo que éste se pueda concentrar en ellas e ignorar el resto de la imagen. Con ello se gana eficiencia computacional.

La función del esquema **Tracker2D** es mantener un seguimiento 2D de personas sobre un flujo de imágenes provenientes de la misma cámara. El seguimiento 2D es resuelto a través de un algoritmo evolutivo similar al algoritmo propuesto en [Marugán, 2007] y descrito también en [Cañas *et al.*, 2008] y [Cañas *et al.*, 2009].

Los algoritmos genéticos o evolutivos son algoritmos de búsqueda que tratan de hallar una solución óptima al problema planteado. Estos algoritmos combinan las propiedades de las hipótesis en el espacio de soluciones para obtener nuevas generaciones más robustas.

Los algoritmos genéticos se dividen principalmente en dos fases:

1. *Generación de la próxima población.* Se aplican operadores genéticos para obtener la nueva población de individuos.

2. *Computación de la salud.* Se calcula la salud para cada uno de los individuos en función de las observaciones realizadas.

El cálculo de salud consiste en evaluar a cada uno de los individuos a partir de unos criterios muy dependientes del problema concreto. Cada uno de estos criterios se corresponde con características que determinan la calidad del individuo. Cuantos más criterios satisface un individuo mayor será su valor de salud h_i.

En este proyecto los individuos son rectángulos 2D que enmarcan a las personas. A la salida del algoritmo las personas serán representadas a través de los individuos con mayor salud, que en este caso serán aquellos más compatibles con las observaciones sensoriales obtenidas de las cámaras.

4.2. Esquema Motionfilter

Como ya se ha comentado en el diseño global del seguidor 2D, este esquema realiza un filtro de movimiento sobre las imágenes y segmenta las regiones de interés para que el seguidor se concentre en ellas. Por tanto, la entrada de este módulo son las imágenes en tiempo real de una cámara o un vídeo y, tras ser procesadas, la salida consiste en una lista de rectángulos o regiones de interés además de la máscara binaria de movimiento. El esquema ejecuta a una velocidad de 33 ips.

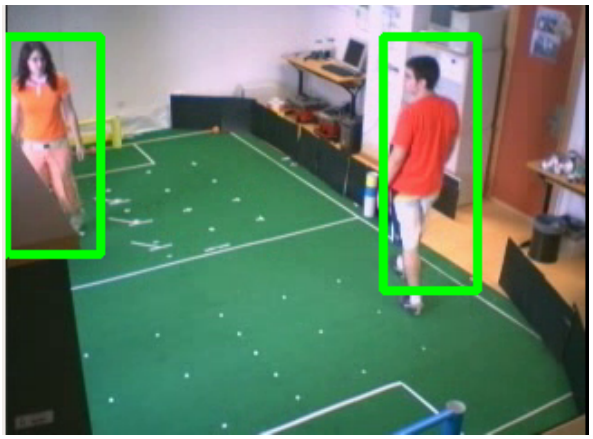

Figura 4.3: Resultado del segmentador de movimiento.

El diagrama 4.4 muestra las etapas de procesamiento que realiza este módulo. Tras obtener una imagen genera una máscara a través de aplicar un filtro por diferencias a la imagen. Posteriormente, utilizando dicha máscara calcula las regiones de mayor movimiento.

El filtro por diferencias se basa en el cálculo de la diferencia absoluta entre:

1. El fotograma actual y el fotograma del instante anterior.
2. El fotograma actual y el fondo aprendido.

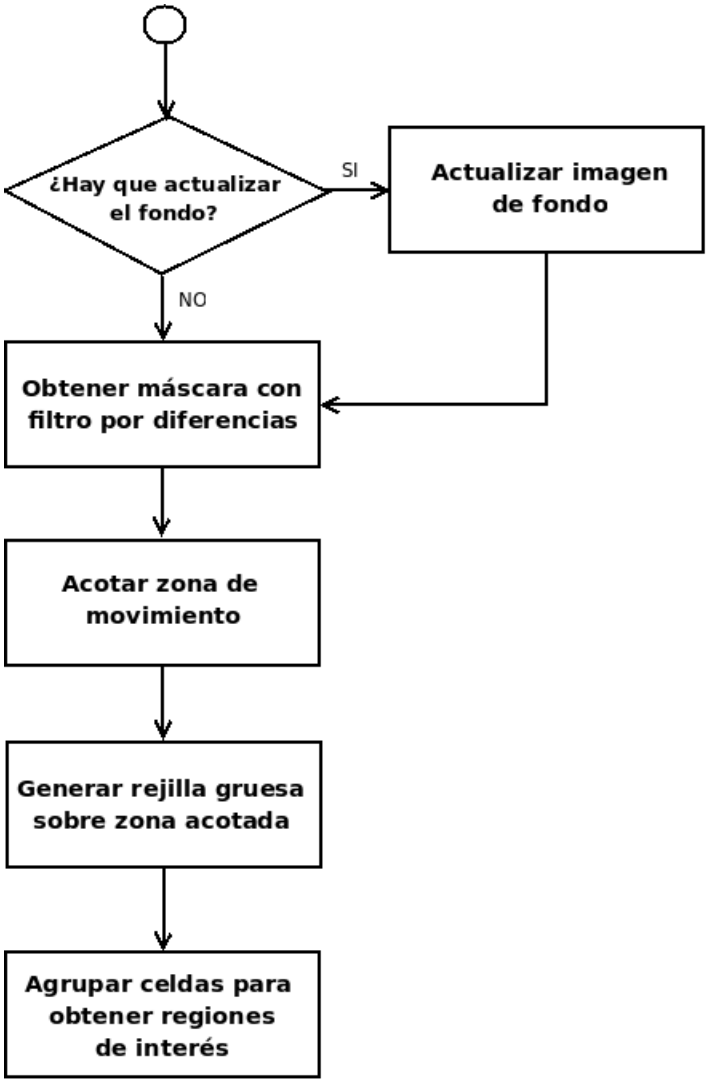

Figura 4.4: Diagrama de flujo de una iteración del procesamiento de *motionfilter*.

La diferencia se realiza entre imágenes RGB, la fórmula aplicada para el cálculo de la diferencia en un píxel p(x,y) es:

$$ImagenDiff(x,y) = (diff_R \; and \; diff_G) OR (diff_G \; and \; diff_B) OR (diff_R \; and \; diff_B) \tag{4.1}$$

donde los booleanos diffR,diffG y diffB indican si la diferencia absoluta entre los canales R, G y B, respectivamente, es superior a un umbral definido en 35 para la diferencia con el fotograma anterior y 25 para la diferencia con el fondo aprendido.
El motivo de calcular así la diferencia es que se obliga a que en al menos dos canales haya una diferencia significativa.

La máscara se calcula de la siguiente forma:

$$mascara(x,y) = ImagenDiff_{previa}(x,y) \; OR \; ImagenDiff_{fondo}(x,y) \tag{4.2}$$

donde $ImagenDiffprevia$ es la imagen bitonal diferencia con el fotograma pevio y $ImagenDiffFondo$, la diferencia con el fondo.

Motionfilter aprende el fondo de las imágenes que recibe. El aprendizaje de fondo consiste en generar una imagen de fondo dinámicamente, fruto de la acumulación ponderada de fotogramas cada ciertas iteraciones (ver ecuación 4.3). Esto permite descartar regiones semejantes al fondo en la imagen a analizar.

$$fondo(t) = \alpha \times fondo(t - \beta) + (1 - \alpha) \times fondo(t) \tag{4.3}$$

donde α concretamente es 0.7, y β indica el intervalo de tiempo que debe transcurrir entre cada actualización del fondo, en este caso unos 2 minutos aproximadamente.

Figura 4.5: Imagen actual (a), imagen de fondo aprendida (b)

Una persona que se mantenga quieta seguirá resaltando contra el fondo, sin embargo si permanece un tiempo muy prolongado parada, el sistema la incorporaría al fondo y el filtro de movimiento no la detectaría.

Tras calcular la máscara de la diferencia, se dilata para evitar que, al aplicar la máscara a la imagen actual, ésta elimine partes de una persona en caso de que la silueta de la persona esté muy ajustada.

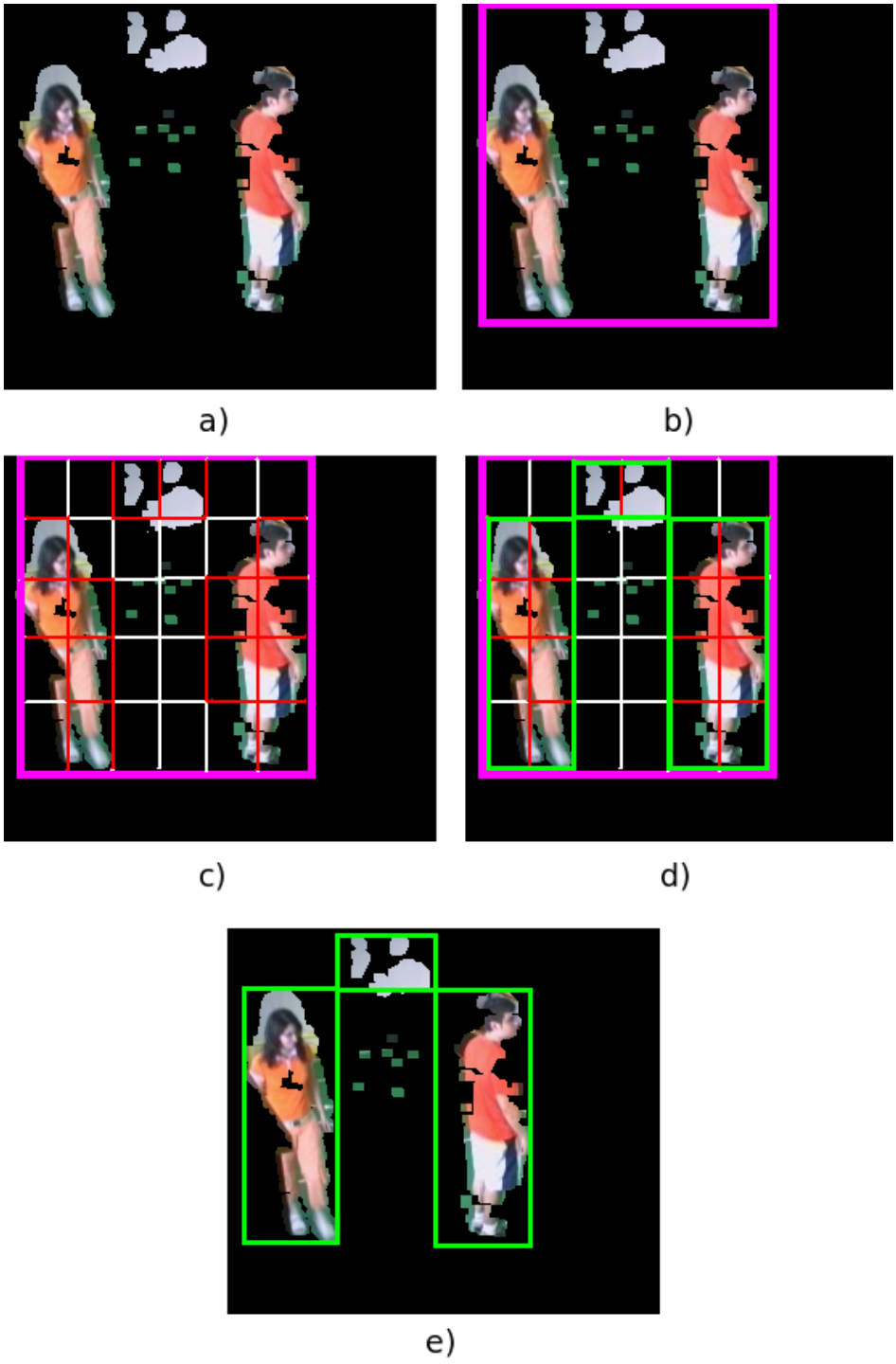

Figura 4.6: Máscara por diferencias (a),región que agrupa todo el movimiento (b),rejilla (c),agrupación de celdas (d) y regiones de interés.

En la imagen 4.6 se puede ver gráficamente todos los pasos de procesamiento que realiza el esquema *motionfilter*.

Una vez obtenida la máscara binaria de diferencias, se analizan y definen cuáles

son las regiones de interés en base a la máscara. La definición de estas zonas consiste en colocar una rejilla de grano grueso sobre la imagen filtrada. El objetivo es saber qué cuadrados son contenedores de las zonas con mayor movimiento de la imagen, para ello se cuentan los píxeles que superan el filtro de movimiento para cada cuadrado y los que obtengan un número superior a un umbral serán considerados para la generación de las zonas de interés (rejilla roja en la imagen C).
Las zonas de interés son agrupaciones de cuadrados adyacentes (regiones verdes en la imagen D y E).

4.3. Descripción de una raza

En el contexto del algoritmo evolutivo de este proyecto, una raza es una población de individuos generada para explorar localmente la región en la que se encuentra una persona dentro de las imágenes. Su finalidad es estimar continuamente la posición 2D y el tamaño de la persona seguida.

Los individuos de una raza son rectangulos 2D con cuatro grados de libertad (figura 4.7). Cada uno de estos individuos es valorado asignándole una salud en función de las observaciones sensoriales y la propia información almacenada por la raza.

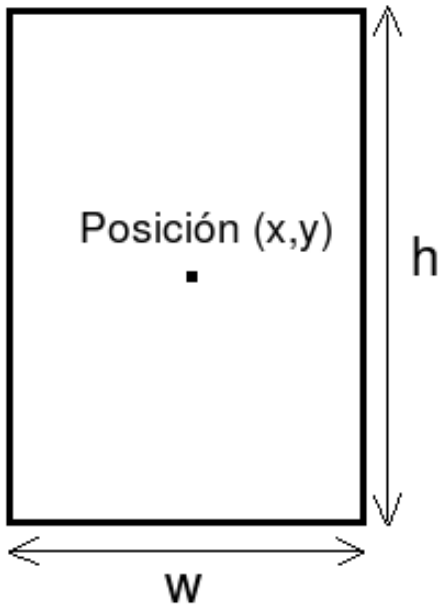

Figura 4.7: Individuo 2D.

Además, a la propia raza también se le asigna una salud, que se corresponde con la salud del mejor individuo de la raza. La salud de la raza se utiliza para aumentar o disminuir la vida de una raza. La vida es un valor que también almacena una raza y representa la fiabilidad de que la raza esté realmente siguiendo a una persona. En cada iteración es consultado para decidir si hay que eliminar la raza o no.
En una iteración dada, si una raza obtiene buena salud por su compatibilidad con la observación en la imagen actual se le aumenta la vida, si no, se le disminuye. Esto permite tener cierta paciencia a la hora de eliminar una raza, ya que puede que la persona esté siendo ocluida por un breve período de tiempo.

Un raza extrae información de las imágenes, la guarda y la actualiza cada cierto tiempo. El objetivo es caracterizar la persona seguida mediante distintos tipos de información y así facilitar seguimiento robusto de la misma. Además, esta información será consultada a la hora de calcular la salud de los individuos de la raza. El conjunto de información consiste en:

- Histograma de color en el modelo HSV.
- Puntos característicos (esquinas, bordes, etc.) seguidos mediante la técnica de Lucas Kanade.
- Continuidad espacial dada por la estimación de la posición y velocidad en la imagen a través de un filtro de Kalman.

A continuación se describe en detalle cada tipo de información que almacena una raza.

4.3.1. Histograma de color

El aprendizaje de color ayuda al seguimiento de las personas. Aunque varíe según la iluminación de la zona en la que se mueve la persona, se puede definir un filtro de color adaptativo a través de un histograma de color acumulativo.

La información de color de una raza se guarda en un histograma en el modelo de color HSV. El modelo HSV (del inglés *Hue, Saturation, Value* - Tonalidad, Saturación, Valor), también llamado HSB (*Hue, Saturation, Brightness* - Tonalidad, Saturación, Brillo), define un modelo de color en términos de sus componentes constituyentes en coordenadas cilíndricas:

- Tonalidad, el tipo de color (como rojo, azul o amarillo). Se representa como un grado de ángulo cuyos valores posibles van de 0 a 360 grados (aunque para algunas aplicaciones se normalizan del 0 al 100 %). Cada valor corresponde a un color. Ejemplos: 0 es rojo, 60 es amarillo, 120 es verde y 240 es azul.
- Saturación. Se representa como la distancia al eje de brillo negro-blanco. Los valores posibles van del 0 al 100 %. A este parámetro también se le suele llamar "pureza"por la analogía con la pureza de excitación y la pureza colorimétrica de la colorimetría. Cuanto menor sea la saturación de un color, mayor tonalidad grisácea habrá y más decolorado estará.
- Valor del color, el brillo del color. Representa la altura en el eje blanco-negro. Los valores posibles van del 0 al 100 %. 0 siempre es negro. Dependiendo de la saturación, 100 podría ser blanco o un color más o menos saturado.

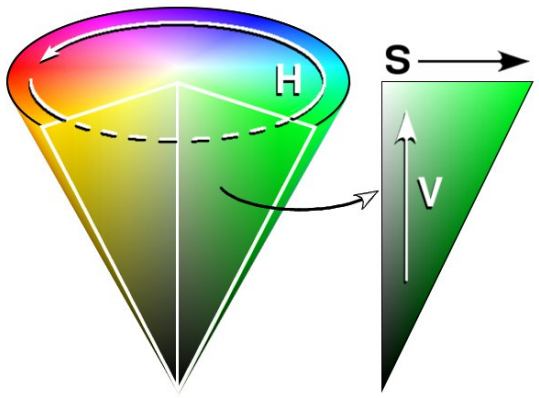

Figura 4.8: Cono de colores del espacio HSV.

La generación y consulta del histograma se realiza a través de funciones de la librería *OpenCV* (ver sección 3.3). Concretamente, éstas son:

```
CvHistogram* cvCreateHist( int dims, int* sizes,
              int type,float** ranges=NULL, int uniform=1 );

void cvCalcHist( IplImage** image, CvHistogram* hist,
                 int accumulate=0, const CvArr* mask=NULL );

float cvQueryHistValue_3D(CvHistogram* hist,int idx0,int idx1,int idx2 );
```

Para generar un histograma basta con proporcionar una imagen y una región de interés sobre ella. En el caso de una raza, esta región de interés es el individuo o rectángulo representante de la raza. El histograma se construye iterativamente acumulando muestras cada cierto intervalo de tiempo, concretamente cada 5 iteraciones.

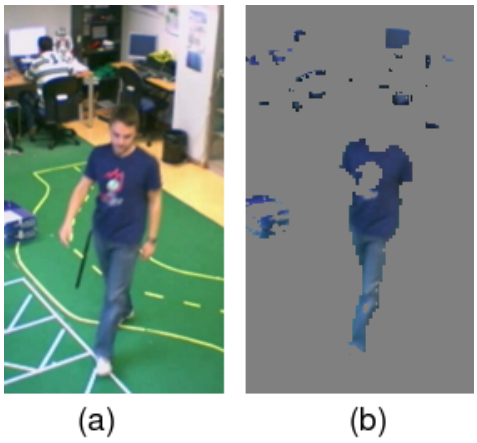

Figura 4.9: Filtro de color - (a) Imagen original, (b) Imagen filtrada.

El objetivo es generar dinámicamente un filtro de color basado en el histograma, es decir, obtener el color predominante del histograma para poder filtrar las imágenes respecto a ese color con un umbral de tolerancia. Como interesa que el color sea lo más fiel posible al de la vestimenta de la persona, en la toma de muestras para aprender el histograma *OpenCV* permite indicar una máscara que omita aquellos píxeles que no interesan. En este caso son los píxeles que pertenecen al fondo y no a la persona. Para la generación del histograma de color de una raza se utiliza la máscara de movimiento, que descarta la mayoría de los píxeles que no pertenecen a la persona. La continua actualización del histograma permite que el color se redefina y por tanto sea robusto frente a cambios de iluminación, como se muestra en la imagen 4.10.

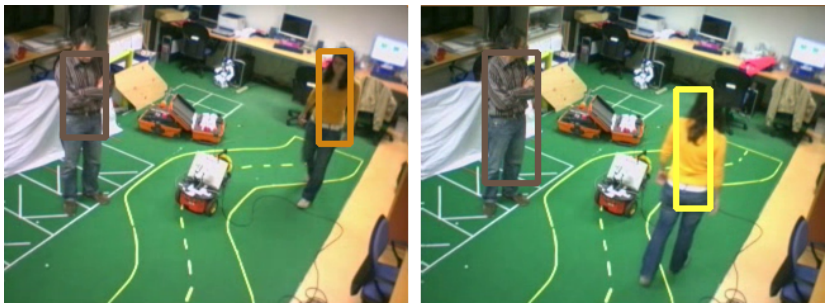

Figura 4.10: Redefinición del color de una raza.

4.3.2. Puntos característicos

El seguimiento de puntos característicos enriquece el seguimiento 2D de una persona, ya que es un tipo de información que es independiente de la iluminación de la escena y el color de la vestimenta de la persona. En ocasiones el filtro de color generado es muy amplio y dificulta la discriminación de la persona respecto del fondo. Los puntos característicos no sufren ese tipo de problemas.

El algoritmo utilizado para la extracción de puntos característicos es *Good Features To Track* presentado en [Shi y Tomasi, 1994], concretamente la implementación de *OpenCV*. Estos puntos normalmente se corresponden con esquinas, bordes o puntos de alto contraste en la imagen. Además del algoritmo de extracción de puntos, para ajustar la posición de los puntos extraídos se ha utilizado la función *FindCornerSubPix*.

```
void cvGoodFeaturesToTrack( IplImage* image, IplImage* eigImage,
                            IplImage* tempImage,
                            CvPoint2D32f* corners, int* cornerCount,
                            double qualityLevel, double minDistance );

void cvFindCornerSubPix( IplImage* I, CvPoint2D32f* corners,
                         int count, CvSize win, CvSize zeroZone,
                         CvTermCriteria criteria );
```

Para el seguimiento de los puntos se ha empleado el método de [Lucas y Kanade, 1981], ya que *OpenCV* provee funciones relacionadas con él. Para realizar el emparejamiento, asume que el entorno del punto se mantiene constante. Por tanto, se crea una ventana alrededor del píxel que define su entorno y se intenta encontrar su ventana correspondiente en la imagen siguiente. El seguimiento se ha realizado a través de otra función de *OpenCV*:

```
void cvCalcOpticalFlowPyrLK( const CvArr* imgA, const CvArr* imgB,
                             CvArr* pyrA, CvArr* pyrB,
                             CvPoint2D32f* featuresA,
                             CvPoint2D32f* featuresB, int count,
                             CvSize winSize, int level, char* status,
                             float* error, CvTermCriteria criteria,
                             int flags );
```

Cada raza posee su conjunto de puntos característicos que en cada iteración son emparejados dentro de una ventana de influencia que pertenece a la raza. Los puntos no emparejados se eliminan del conjunto. Si la cantidad de puntos desciende por debajo de cierto umbral, se calculan nuevos puntos característicos y se añaden al conjunto. Por tanto, los puntos se van redefiniendo.

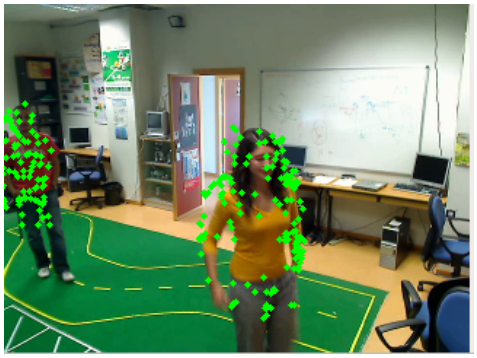

Figura 4.11: Puntos Lucas-Kanade.

4.3.3. Filtro de Kalman

El filtro de Kalman [Kalman, 1960] proporciona un buen marco para la estimación de una variable, de la que se dispone de medidas a lo largo del tiempo. Es una técnica de estimación bayesiana utilizada para seguir sistemas estocásticos dinámicos observados mediante sensores ruidosos. En visión artificial es un algoritmo recursivo que estima la posición de una característica (punto, borde, esquina, etc.) en movimiento y la incertidumbre de la medida.

En este proyecto, el estado **x** se corresponde con el vector posición de la persona en la imagen, determinado por las coordenadas x_x y x_y, y el vector velocidad de coordenadas v_x y v_y. Sin embargo, la observación **z** es un vector de dos componentes z_x y z_y, correspondientes a las coordenadas de la posición observada de la persona.

La matriz A_{NxM} relaciona el estado en tiempo **k** con el estado en tiempo **k+1**. Esta relación se manifiesta en las siguientes ecuaciones, dando como resultado la matriz **A**.

$$x_{x_{k+1}} = x_{x_k} + v_x t$$

$$x_{y_{k+1}} = x_{y_k} + v_y t$$

$$v_{x_{k+1}} = v_{x_k}$$

$$v_{y_{k+1}} = v_{y_k}$$

$$A = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.4}$$

La matriz H_{NxM} relaciona el estado con la medida z_k.

$$H_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{4.5}$$

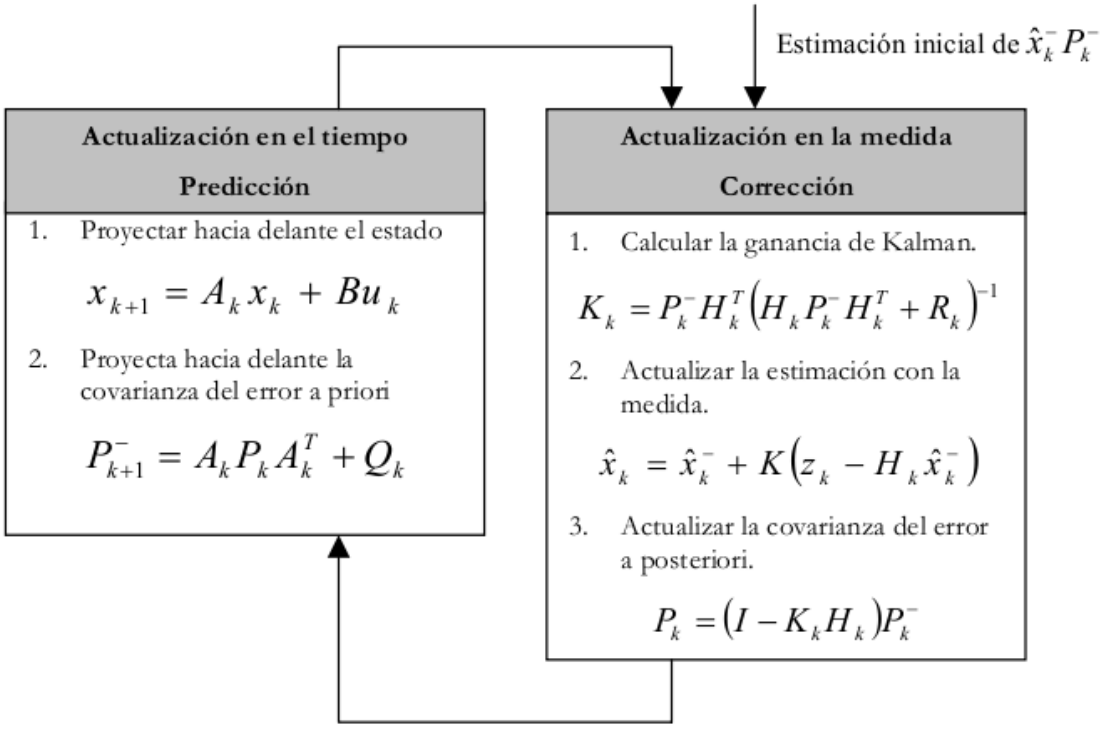

Figura 4.12: Diagrama del filtro de Kalman.

El filtro de Kalman proporciona una ecuación que computa un estimador del estado a posteriori x_k como combinación lineal del estimador a priori x_k- y la diferencia ponderada entre la observación actual z_k y una predicción de medida $H_k x_k-$. La descripción del filtro del Kalman con sus ecuaciones puede verse en el diagrama 4.12.

Para la implementación del filtro de Kalman se han utilizado las siguientes funciones de *OpenCV*:

```
CvKalman* cvCreateKalman( int dynamParams, int measureParams,
  int controParams=0 );

const CvMat* cvKalmanPredict( CvKalman* kalman, const CvMat* control=NULL );

void cvKalmanCorrect( CvKalman* kalman, const CvMat* measurement=NULL );
```

Aplicado al seguimiento 2D de este proyecto, cada raza posee su propio filtro de Kalman. La observación **z** para el filtro es el centro(x,y) de la región de interés instantánea más próxima a la posición de la raza en una iteración dada. Para ello cada región que da el esquema *motionfilter* (ver sección 4.2) se intenta asociar a una raza existente y su centro se toma como observación para la raza. El motivo es evitar que si la posición estimada por la raza comienza a desviarse de la posición de la persona, el filtro de Kalman sea realimentado con esta información errónea.

4.4. Algoritmo evolutivo multimodal

Una vez descrito con detalle lo que es una raza, en esta sección se van a explicar los pasos del algoritmo evolutivo y los aspectos significativos de la dinámica de razas, que permiten evolucionar el conjunto de razas y que éstas converjan a las posiciones 2D de las personas en la imagen.

Una nueva raza es generada si entre las regiones de interés que detecta el esquema *motionfilter* aparece una que aún no está siendo analizada, en ese caso se crea una raza para explorar esa zona. Así mismo, si una persona abandona la escena cubierta por la cámara, la raza que mantenía su seguimiento es eliminada.

La figura 4.13 representa el diagrama de flujo del algoritmo propuesto.

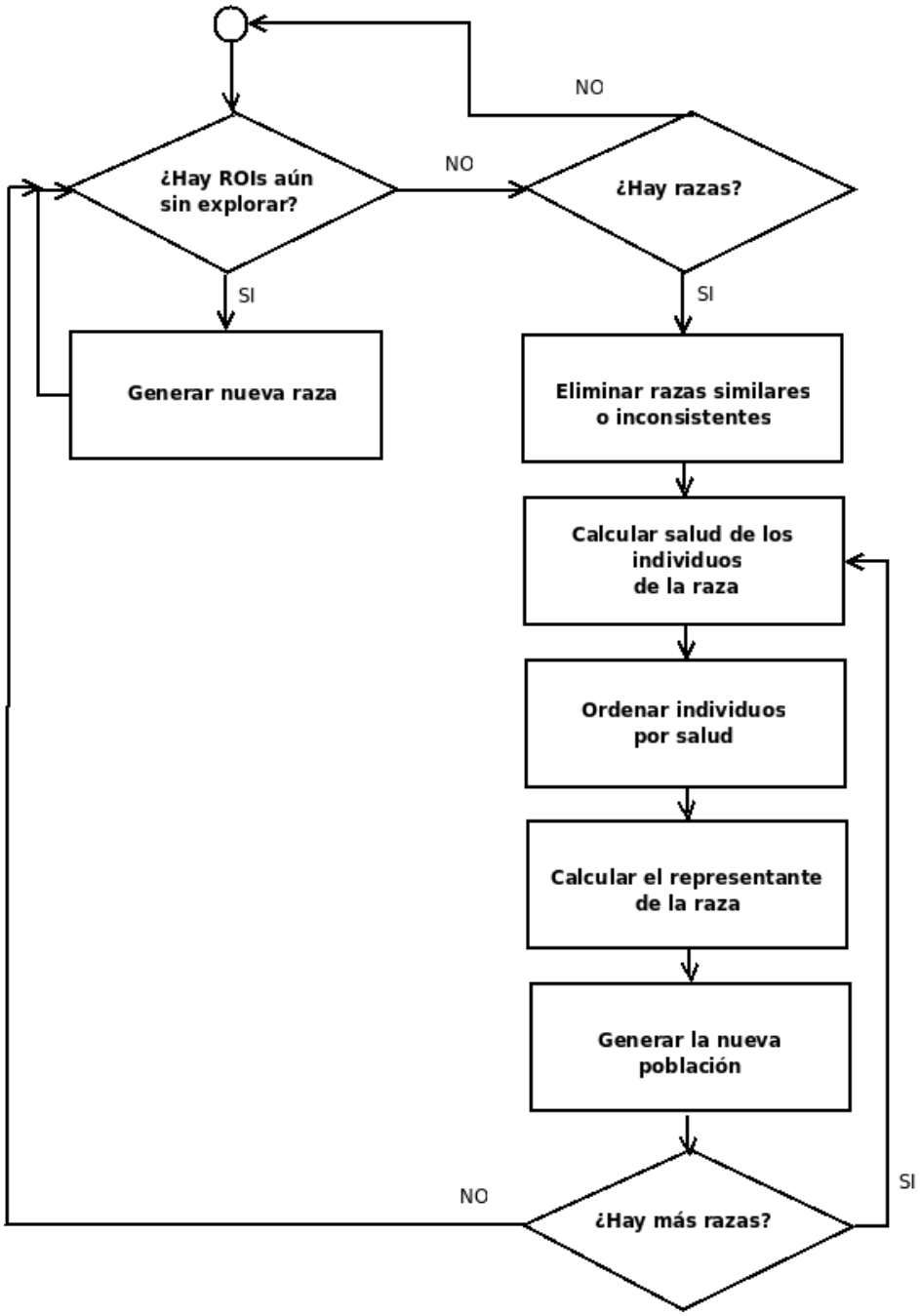

Figura 4.13: Diagrama de flujo del algoritmo evolutivo.

Una vez que se tiene un conjunto de razas analizando las regiones de interés, en primer lugar se comprueba que no haya razas similares entre sí ni inconsistentes. Una raza es inconsistente cuando su salud acumulada (vida) es inferior a un umbral.

Tras estas comprobaciones, para cada raza se calcula la salud de sus individuos en función de cuatro tipos de características distintas, que se detallan en la siguiente sección. Después, se ordenan por salud los individuos y se calcula el individuo representante de la raza como una media de los individuos con mejor salud. Por último, se genera la siguiente población utilizando dos operadores genéticos: *elitismo* y *ruido térmico*.

El elitismo consiste en seleccionar los individuos con mejor salud y mantenerlos sin ningún cambio en la siguiente generación. Esto permite mantener las mejores soluciones encontradas hasta el momento. El ruido térmico, sin embargo, aplica pequeños cambios en la posición y el tamaño de los individuos elitistas. De esta manera las razas exploran sus alrededores y son capaces de mantener el seguimiento de una persona en movimiento.

Resumiendo, una raza surge cuando en una zona de la imagen se está detectando movimiento que aún no está explicado y evoluciona adaptándose a las nuevas posiciones de la persona a la que está siguiendo.

4.5. Computación de salud

La salud de un individuo, en este caso un rectángulo 2D, indica si para cierto instante el rectángulo representa fiablemente con su posición y su tamaño a una persona en la imagen.

Para el cálculo de salud de los individuos se han tomado en cuenta los cuatro tipos de información extraída de las imágenes: movimiento, color, emparejamiento de puntos característicos y grado de parecido a la estimación de la posición dada por un filtro de Kalman.

La función salud se calcula basándose en cuatro funciones de salud parciales, cada una definida sobre un tipo de característica. Por tanto, para el individuo i tendremos una salud total (Hi) y las parciales que serían para el movimiento (Hmo), el color (Hc), el emparejamiento de puntos característicos (Hma) y el parecido con la estimación del filtro de Kalman (Hpos), que es el criterio de continuidad espacial.

En las siguiente secciones denominaremos a cada individuo como Ri, ya que es representado por un rectángulo.

4.5.1. Salud de movimiento

Esta salud mide la densidad de píxeles de movimiento de Ri.

La máscara de movimiento es proporcionada por el esquema *motionfilter* (ver 4.2).

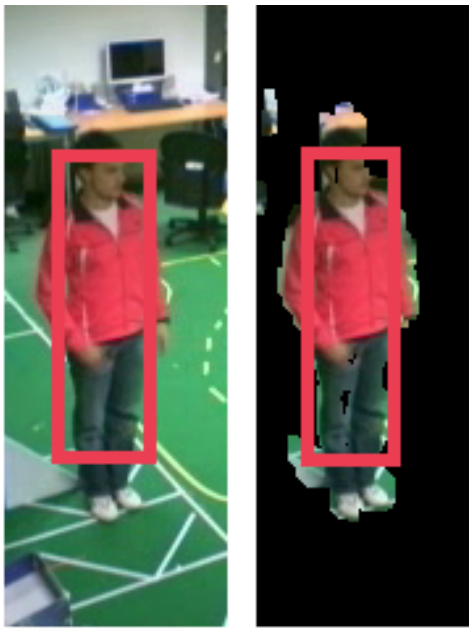

Figura 4.14: Cálculo de la salud de movimiento.

El cálculo de la salud de movimiento para el individuo i se realiza de la siguiente forma:

$$Hmo_i = K \ / \ (\ ancho(R_i) \ * \ alto(R_i)\ ) \tag{4.6}$$

Siendo K el número de píxeles que supera el filtro de movimiento dentro del rectángulo Ri.

4.5.2. Salud de color

Esta salud mide la densidad de píxeles de color de Ri.

Para esta salud se utiliza el histograma de color que la raza almacena y actualiza cada ciertas iteraciones. De éste se obtienen los máximos valores (H,S,V) registrados por el histograma y con ellos se genera un filtro de color.

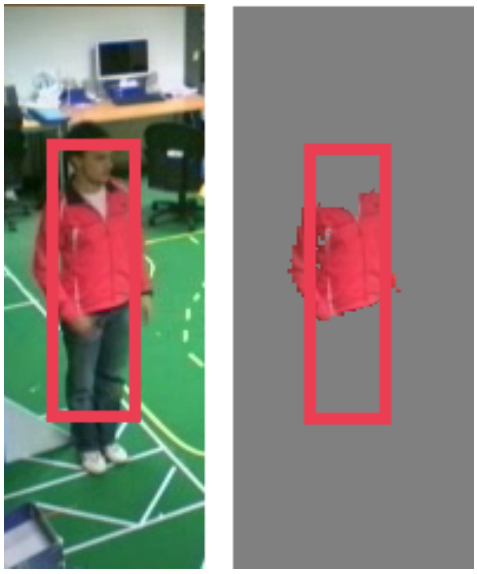

Figura 4.15: Cálculo de la salud de color.

La fórmula que calcula si un píxel de color (h,s,v) supera el filtro de color definido por los valores (H,S,V) es la siguiente:

$$Filtro_i = abs(H - h_i)^1 < Htol\ AND\ abs(S - s_i) < Stol\ AND\ abs(V - v_i) < Vtol \quad (4.7)$$

Siendo Htol, Stol y Vtol las tolerancias del filtro.

El cálculo de la salud de color para el individuo i se realiza de la siguiente forma:

$$Hc_i = K\ /\ (\ ancho(R_i)\ *\ alto(R_i)\ ) \quad (4.8)$$

Siendo K el número de píxeles que supera el filtro de color dentro del rectángulo Ri.

[1] Para hallar la resta en este canal se ha tenido en cuenta que los valores son ángulos.

4.5.3. Salud de emparejamiento de puntos característicos

Esta salud mide la densisdad de puntos característicos que cubre el rectángulo Ri respecto al número total de puntos y también la densidad respecto a la cantidad de píxeles que caen dentro de él.

En la descripción de una raza (sección 4.3) se vió que el emparejamiento de puntos se realiza dentro de una ventana de influencia que se le asigna a cada raza. Esto se hace así debido a que en muchas ocasiones, como muestra la imagen 4.16, el representante de la raza no abarca todos los puntos característicos y la ventana de influencia (en rosa) sí.

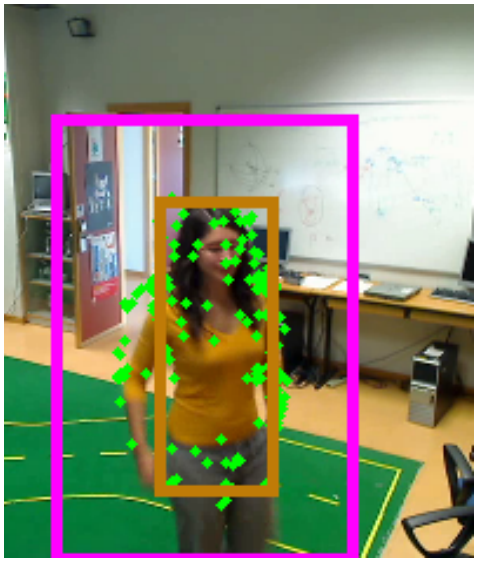

Figura 4.16: Cálculo de la salud de emparejamiento de puntos característicos.

La salud de emparejamiento de un individuo i será:

$$Hma_i = MIN(\ densidad * A\ +\ K\ /\ N,\ 1\ ) \tag{4.9}$$

$$densidad = K\ /\ (\ ancho(R_i)\ *\ alto(R_i)\ ) \tag{4.10}$$

Siendo K el número de puntos característicos que caen dentro de Ri, N el número total de puntos para la raza de Ri y 1 un umbral de saturación. El factor A es un factor de ajuste para que el rango de esta salud sea coherente con el de las demás, concretamente es 10.

4.5.4. Salud con filtro de Kalman

El cálculo de la salud de Kalman para el individuo i se realiza de la siguiente forma:

$$Hpos_i = MAX(\ 1\ -\ (\ D\ /\ alto(Ri)\ ,\ 0\ ) \tag{4.11}$$

Siendo D la distancia entre el centro de Ri y la posición estimada por el filtro de Kalman y la altura de Ri un umbral que si es superado la salud será 0.

Esta salud mide el parecido entre la estimación de la posición de la persona dada por un individuo i de una raza y la estimación dada por el filtro de Kalman.
Como se detalla en el capítulo de experimentos, el filtro de Kalman por si solo mantiene un buen seguimiento para una única persona pero no para varias, ya que únicamente tiene como observación las regiones de interés debidas al movimiento y esto provoca intercambios del seguimiento entre distintas personas.

No obstante, el filtro de Kalman aporta al algoritmo evolutivo continuidad espacial y cierta incercia en sus estimaciones, como es propio del movimiento natural de las personas.

4.5.5. Salud total

La salud total Hi de un individuo i perteneciente a una raza se compone de las parciales anteriormente explicadas, quedando la ecuación así:

$$H_i = \alpha * Hc_i\ +\ \beta * Hmo_i\ +\ \gamma * Hma_i\ +\ \omega * Hpos_i \tag{4.12}$$

Con α, β, γ y ω igual a 0.3, 0.1, 0.2 y 0.4, respectivamente. Estos valores se obtuvieron experimentalmente, como se detallará más adelante.

Esta función permite combinar con mayor o menor peso los cuatro criterios anteriores.

4.6. Interfaz gráfica

La interfaz gráfica es un elemento importante de la aplicación, ya que permite visualizar los resultados del seguimiento 2D. Otra utilidad es como herramienta de

depuración, ya que fácilmente se puede mostrar en ella por ejemplo los filtros tanto de color como de movimiento. Además, permite cambiar en tiempo de ejecución ciertos parámetros del algoritmo y ver su impacto en el funcionamiento.

Figura 4.17: Interfaz gráfica de la aplicación de seguimiento 2D.

Como se ve en la imagen 4.17, la interfaz muestra las imágenes que la aplicación recibe de la cámara y sobre éstas dibuja el resultado del seguimiento, que son los rectángulos representantes de cada raza.

A la mitad de la interfaz tenemos los deslizadores que permiten seleccionar qué pesos le asignamos a cada tipo de salud. Éstos son coeficientes en el rango [0,1].

En cuanto a los botones, los tres primeros empezando por la derecha (*Use SIFT*, *Use SURF* y *Use Lucas-Kanade*) dan a elegir qué tipo de técnica se quiere utilizar para

la extracción y seguimiento de puntos característicos. Por defecto se activa la técnica de Lucas Kanade, que ha sido la seleccionada entre las tres.
Pulsando el botón de *Reset* se puede borrar la lista de razas existente, que es como reiniciar el algoritmo. El botón *Debug Mode* permite ver los filtros de color y movimiento actuales y los puntos característicos bien emparejados de cada raza.
Los botones *Fit. Pos. Analysis* y *Fit. Size Analysis* activan las búsquedas locales en posición y en tamaño que se han realizado para los experimentos (sección 4.8.2), que se presentarán a continuación.
Por último, el botón *Exit* se utiliza para salir de la aplicación.

4.7. Experimentos

En esta sección se describen las pruebas realizadas para la validación del correcto funcionamiento de la aplicación de seguimiento 2D.

La mayor parte de los experimentos realizados con el algoritmo evolutivo están enfocados a la función salud, que es clave en la evolución de las razas y la adaptación de los individuos a las posiciones y tamaños de las personas en las imágenes.
El objetivo de estos experimentos es evaluar los distintos factores que se incluyen en la salud, es decir, los distintos tipos de información utilizados para calcularla. Además, es importante el peso de cada factor en la evaluación de la salud total.

4.7.1. Ejecución típica

En la ejecución típica se puede comprobar que el sistema es capaz de seguir y estimar el tamaño de varias personas dentro de la habitación con vivacidad y precisión, sin conocer ningún tipo de información inicial sobre éstas y realizando un aprendizaje de color para cada una.

Como ya presentamos en el capítulo 3, el escenario de pruebas ha sido el Laboratorio de Robótica del Edificio Departamental II de esta universidad. En el caso de la aplicación de seguimiento 2D, se ha utilizado sólo una cámara web *Apple iSight* a resolución 640x480 adherida en el techo de una esquina de la habitación.

La velocidad de la aplicación en la ejecución típica es de 30 ips aproximadamente,

funcionando con cámaras a 30 fps. Esta velocidad es más que suficiente para seguir a personas moviéndose naturalmente dentro de la habitación. En las imágenes siguientes (4.18 y 4.19) vemos dos ejemplos de secuencias del seguimiento 2D de dos personas que realiza el sistema sobre una secuencia de imágenes. En el *mediawiki*[2] del proyecto se pueden ver vídeos del funcionamiento de la aplicación.

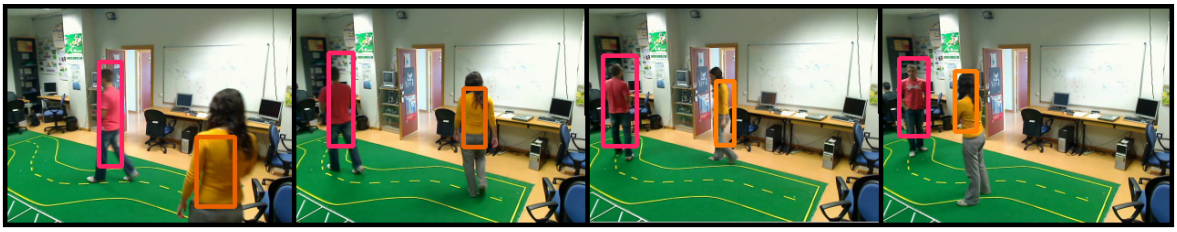

Figura 4.18: Fotogramas de una secuencia del seguimiento 2D de dos personas.

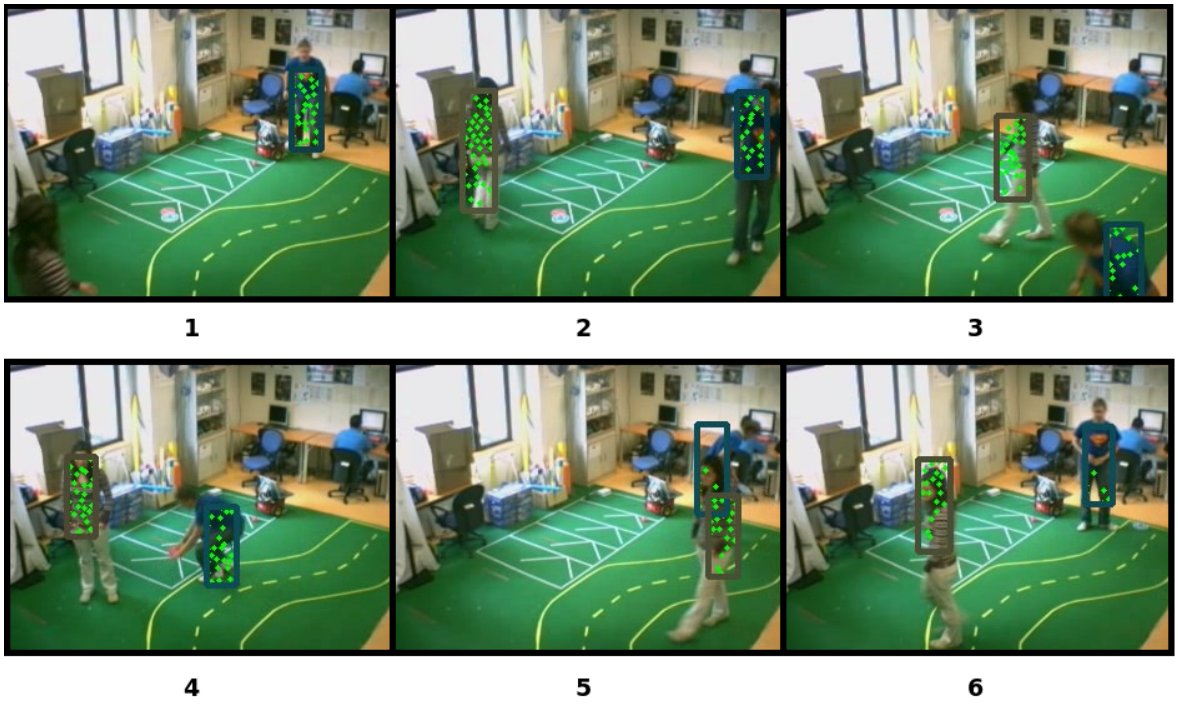

Figura 4.19: Fotogramas de una secuencia del seguimiento 2D de dos personas, dibujando los puntos Lucas Kande.

[2]http://jde.gsyc.es/index.php/Salons-ii

4.7.2. Análisis de la discriminación y corrección de la función salud

El primer experimento para el análisis de la función salud consistió en realizar una búsqueda local partiendo del estado del seguimiento de una persona en un instante dado. Tomando el representante de la raza que mantiene el seguimiento de una persona y manteniendo el tamaño fijo se le aplican desplazamientos tanto en la coordenada X como en la Y con el fin de evaluar la salud en cada posición y valorar la capacidad discriminante de ésta. El resultado expresado gráficamente se puede ver en la imagen 4.20.

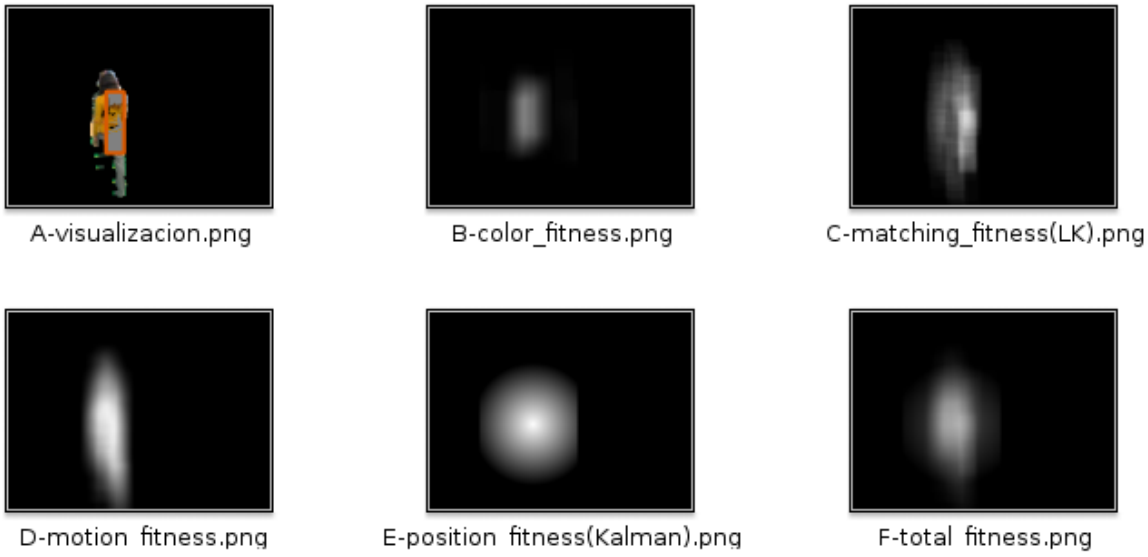

Figura 4.20: Análisis de la discriminación de la función salud en términos de posición- A) Estado del seguimiento, B) Salud parcial por color, C) Salud parcial por emparejamiento Lucas-Kande, D) Salud parcial por movimiento, C) Salud parcial por distancia a la estimación del filtro de Kalman.

La intensidad de cada píxel en las imágenes representa la salud del representante desplazado de tal modo que su centro cae en dicho píxel, por lo que las zonas más claras son aquellas con mayor salud.
Podemos ver tanto en todos los tipos de salud como en la salud total que los máximos valores se encuentran donde está la persona situada y que van descendiendo a medida que se alejan de la posición XY buena. Esto prueba que la salud discrimina bastante bien entre la persona y el fondo en la imagen.

El segundo experimento se realizó para analizar del mismo modo la salud, pero en

términos de tamaño del rectángulo. Se tomó el representante de la raza asociada a una persona en determinado instante y se exploraron distintos tamaños del rectángulo dejando fija la posición. El resultado se muestra con un mapa de temperatura (figura 4.21).

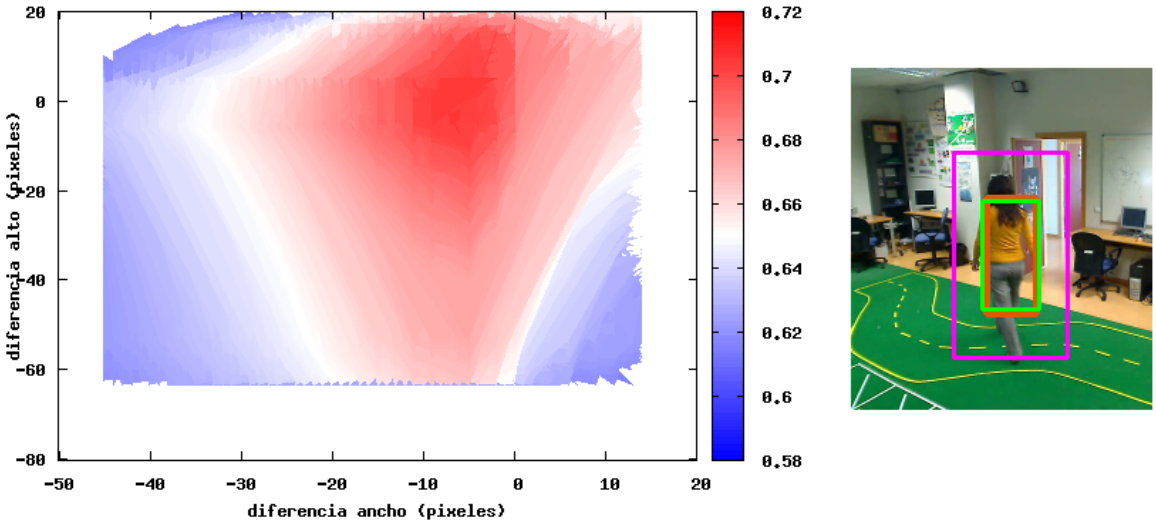

Figura 4.21: Análisis de la discriminación de la función salud en términos de tamaño.

El rectángulo morado representa el tamaño máximo empleado en la búsqueda local, el naranja es el representante de la raza y el verde es el mejor encontrado por la búsqueda local. Como se aprecia, el resultado del algoritmo evolutivo y el de la búsqueda local son muy parecidos. Esto prueba que el algoritmo evolutivo tiene salud máxima en el sitio y tamaño correctos.

Por último, se realizó un experimento para explorar los cuatro grados de libertad a la vez (coordenada X, coordenada Y, ancho y alto). Para ello hice un pequeño programa que realiza una búsqueda local por cada muestra guardada durante una ejecución típica. Una muestra consiste en las observaciones sensoriales y en el propio estado de la raza (posición, tamaño, color y puntos característicos emparejados).
Además, este programa permite indicar a través de su interfaz gráfica la solución que hemos considerado correcta en cada muestra, es decir, el rectángulo envolvente de la persona. De este modo hemos obtenido 23 muestras que hemos analizado con búsqueda local en cuatro grados de libertad y los resultados han sido comparados con las soluciones reales determinadas a mano. En la imagen 4.22 tenemos el resultado del análisis de una de las muestras y el ejemplo de ventana de búsqueda.

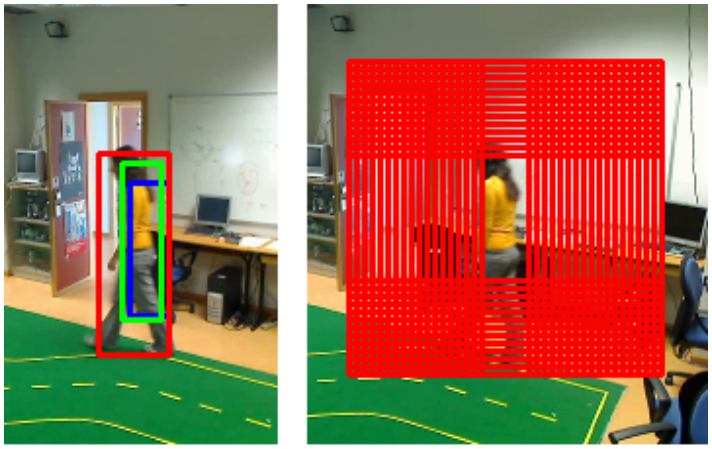

Figura 4.22: Análisis de la salud -(a) Rojo: solución real; Azul: solución del algoritmo; Verde: solución de búsqueda local. (b) Ventana de búsqueda.

A continuación podemos ver las gráficas comparativas de la estimación de la posición, el ancho y el alto del algoritmo evolutivo y de la búsqueda local, ambos respecto a la solución real.

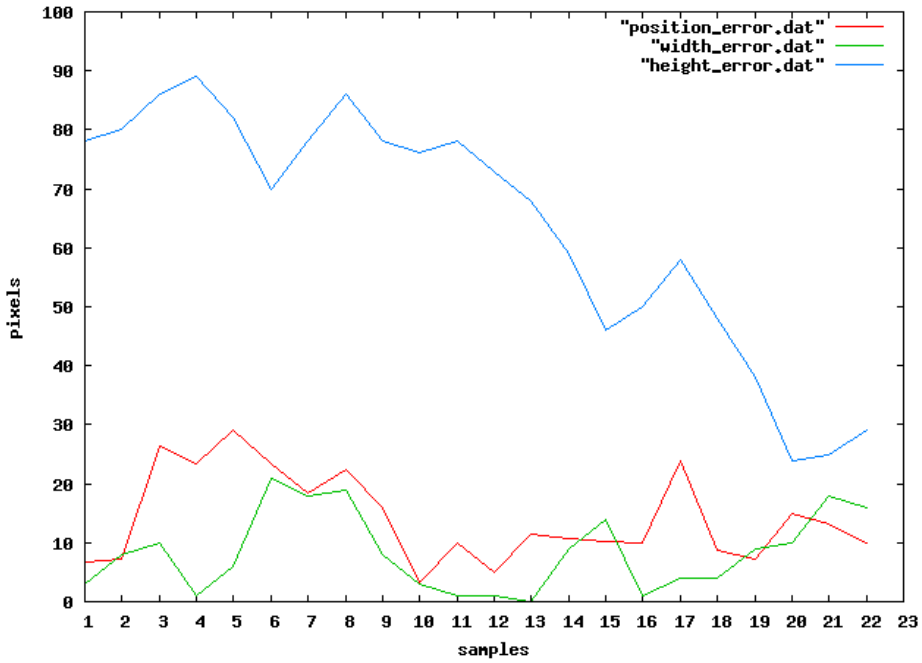

Figura 4.23: Gráfica de error del algoritmo evolutivo.

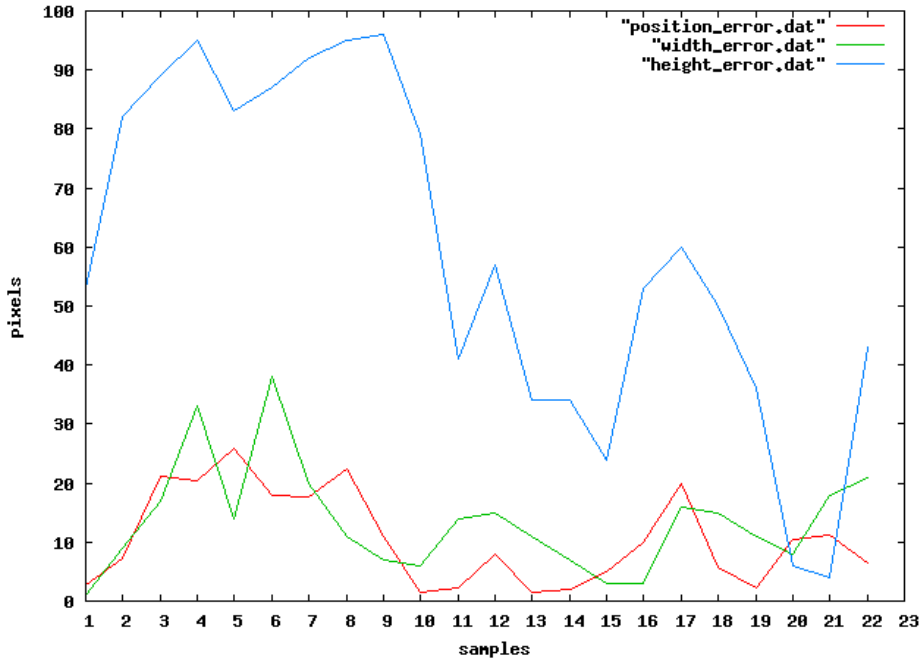

Figura 4.24: Gráfica de error de la búsqueda local.

Las gráficas muestran que el error medio en la estimación de la posición y del ancho es de 15 píxeles. En el caso de la estimación del alto es de 60 píxeles, esto es debido a que por regla general la raza aprende mejor el color de la camiseta de la persona y esto provoca que la raza en ocasiones obvie las piernas de la misma.

Otra conclusión que se puede sacar de estos experimentos es que la búsqueda local no mejora lo suficiente la estimación del algoritmo como para incluirla dentro del mismo, ya que la búsqueda local conlleva un coste computacional elevado.

4.7.3. Ajuste de los coeficientes de la función salud

En cuanto a los pesos asignados a cada tipo de salud parcial, éstos han sido determinados observando el comportamiento del algoritmo en el seguimiento de dos personas. El seguimiento de una única persona en la escena es sencillo, sin embargo con dos o más personas surgen nuevos problemas a resolver, como por ejemplo el detectar si dos razas son similares o evitar un intercambio de razas en un cruce de personas en la imagen.

La mejor combinación de pesos obtenida ha sido:

- Salud de movimiento: 15 %

- Salud de color: 30 %
- Salud de emparejamiento de puntos: 15 %
- Salud con filtro de Kalman: 40 %

La salud de color es el factor más discriminante de la función salud, por lo que es lógico que adquiera un peso importante. Sin embargo, hay ocasiones en las que el propio color de la persona no es muy discriminante, por el entorno o porque varias personas vayan del mismo color.

La salud de emparejamiento de puntos también es un factor discriminante para una raza y en general mejora el funcionamiento del algoritmo. Su peso, que no es muy elevado, se debe a que en los cruces de personas no es capaz de discriminar bien, ya que el seguimiento de los puntos se pierde y al redefinirlos pueden no pertenecer ya a la persona correcta.

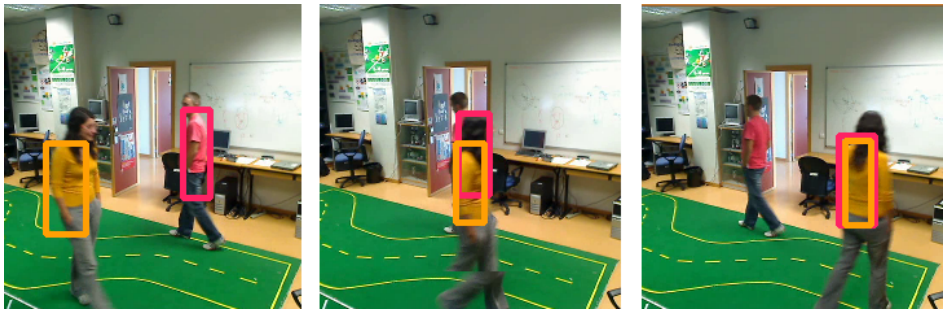

Figura 4.25: Cruce mal resuelto de dos personas. Pesos asignados: movimiento(35 %), color(30 %), emparejamiento(15 %) y filtro de Kalman(20 %).

Tanto la salud de movimiento como la calculada con el filtro de Kalman son tipos de salud que se basan en el movimiento de la persona. La primera mide el movimiento instantáneo que produce una persona en cada iteración y la segunda mide también movimiento pero además la naturalidad y la inercia del mismo, como es propio del movimiento generado por una persona. Esto ayuda enormemente a que no se den intercambios de razas en el cruce de personas (ver figura 4.26).

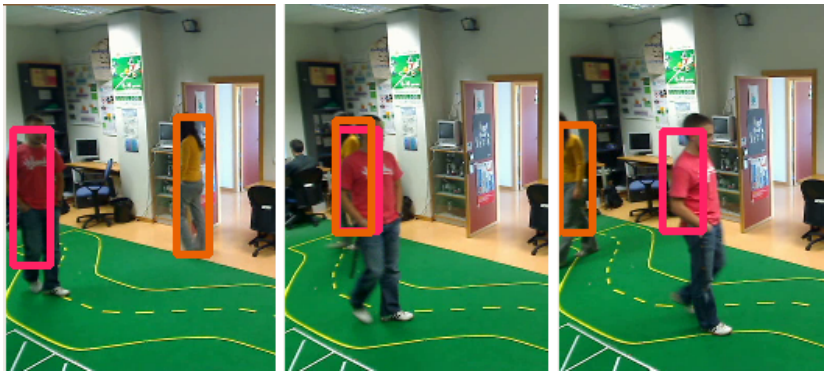

Figura 4.26: Cruce bien resuelto de dos personas. Los pesos asignados con la mejor combinación.

4.8. Alternativas probadas

En esta sección se presentan los experimentos realizados para valorar las distintas técnicas estudiadas. En primer lugar se exponen los experimentos relacionados con el seguimiento de puntos característicos y, posteriormente, los relacionados con el seguimiento de personas.

4.8.1. Seguimiento de puntos característicos

Se han estudiado distintas técnicas de extracción y seguimiento de puntos característicos y han sido comparadas entre sí. Estas técnicas, que han sido presentadas en el capítulo 1, son:

- *Scale-Invariant feature transform (SIFT).*
- *Speeded Up Robust Features (SURF)*[3].
- *Lucas Kanade.*

En primer lugar se hicieron pruebas con las técnicas SIFT, SURF y Lucas Kanade sobre un vídeo a distintos tamaños de resolución. El objetivo de ello era comprobar la influencia de la resolución en la extracción y seguimiento de estas características. Los datos han sido recogidos durante medio minuto de análisis de un vídeo. Los resultados se muestran en las siguientes gráficas.

[3]http://www.vision.ee.ethz.ch/ surf/index.html

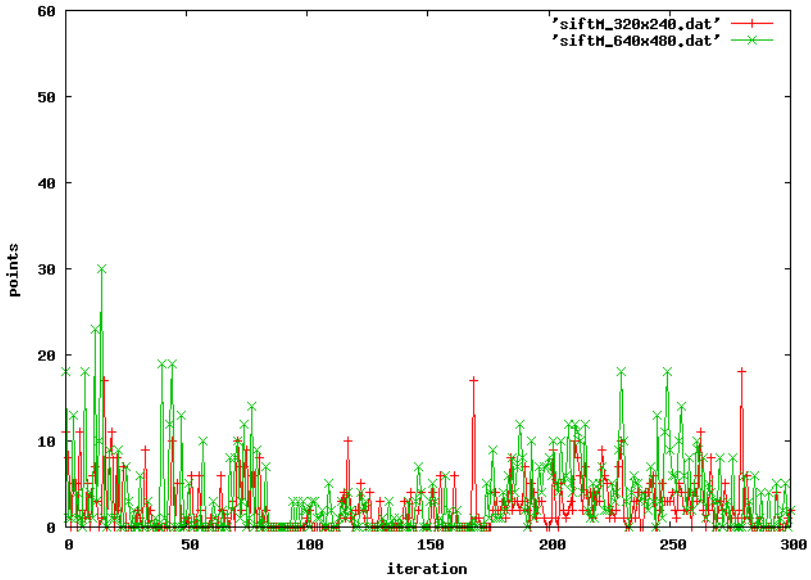

Figura 4.27: Gráfica comparativa de SIFT para 320x240 (rojo) y 640x480 (verde).

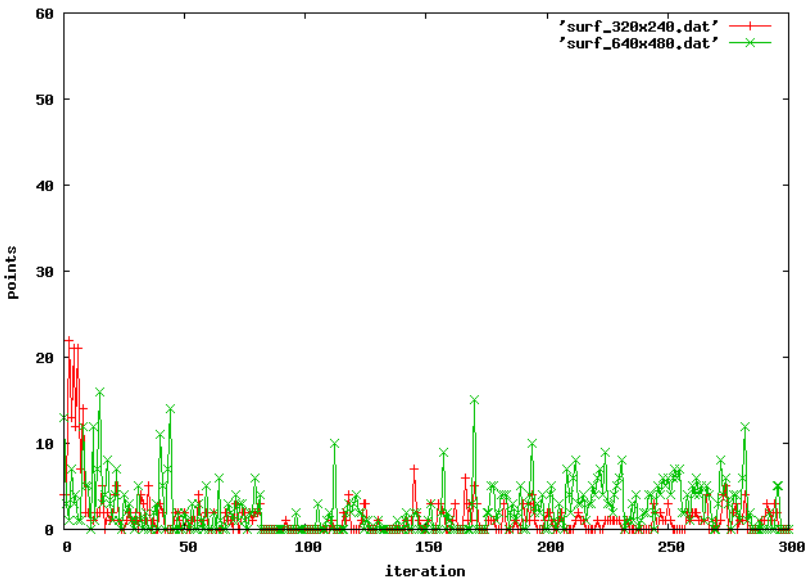

Figura 4.28: Gráfica comparativa de SURF para 320x240 (rojo) y 640x480 (verde).

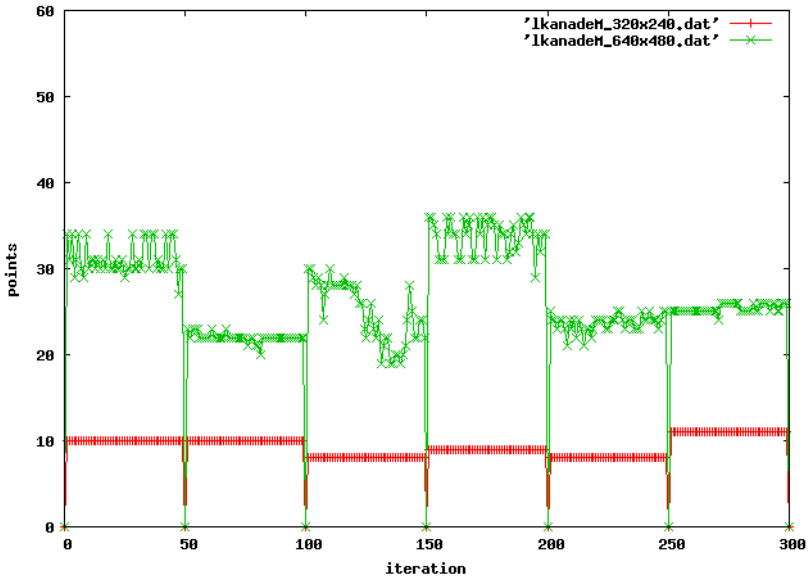

Figura 4.29: Gráfica comparativa de Lucas Kanade para 320x240 (rojo) y 640x480 (verde).

Las gráficas relacionan el número de puntos extraídos con el tiempo, que en este caso se ha medido en iteraciones. Los resultados muestran que la resolución influye significativamente en la cantidad de puntos que las técnicas extraen sobre las imágenes. Por lo que a mayor resolución, la cantidad de características extraídas aumenta.
En el caso de la técnica de Lucas Kanade la diferencia es más acusada. Para esta técnica los puntos se han renovado cada 50 iteraciones forzadamente ya que la propia técnica implementa el seguimiento de los puntos.

Las gráficas también muestran el contraste de las diferentes técnicas en el número de puntos característicos. Para verlo mejor la gráfica 4.30 compara las tres técnicas medidas en una resolución de 640x480.

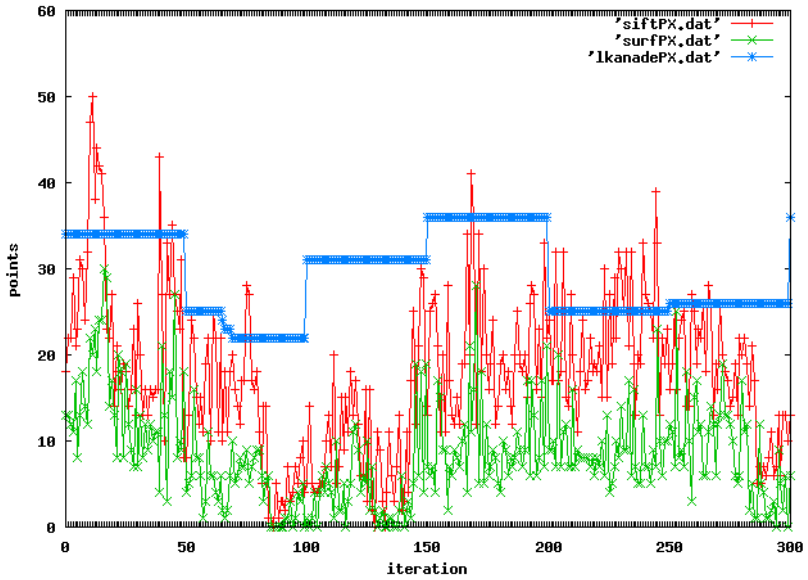

Figura 4.30: Comparativa de cantidad de puntos extraídos por SURF (verde) , SIFT (rojo) y Lucas Kanade (azul).

Se puede ver que SIFT es la técnica que esporádicamente más puntos característicos puede llegar a obtener y que Lucas Kanade es equiparable a ella. Sin embargo, hay una diferencia notable respecto a SURF, que extrae por lo general un menor número de puntos.

Por último, las técnicas han sido comparadas en seguimiento. En el caso de Lucas Kanade el seguimiento se realiza implícitamente y para SIFT y SURF se implementó un seguimiento. Éste consistía en utilizar los puntos bien emparejados del fotograma anterior para emparejarlos de nuevo con el fotograma siguiente. La gráfica 4.31 muestra los resultados.

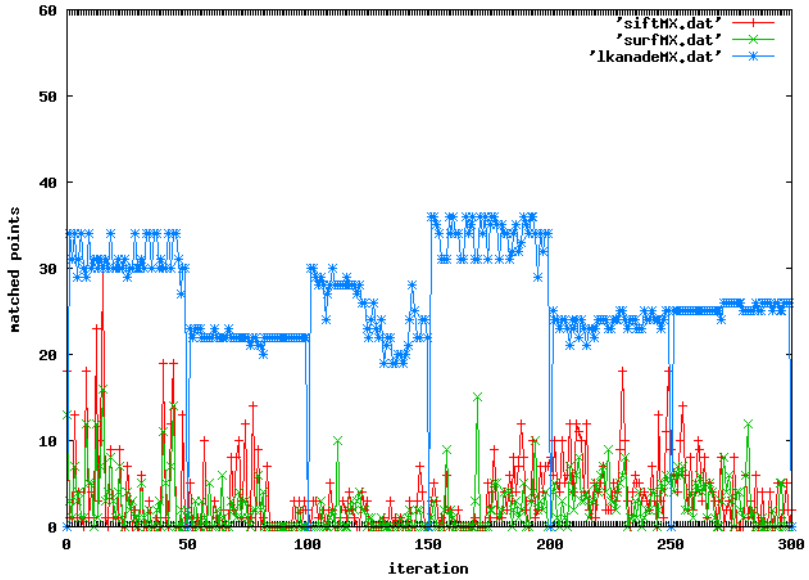

Figura 4.31: Comparativa de cantidad de puntos seguidos por SURF (verde) , SIFT (rojo) y Lucas Kanade (azul).

De igual modo, los puntos de Lucas Kanade han sido renovados forzadamente cada 50 iteraciones. Este análisis muestra que el seguimiento de puntos con la técnica Lucas Kanade es mejor que con las otras técnicas.

En cuanto al rendimiento temporal de cada técnica, en la tabla comparativa se pueden ver los tiempos medios en milisegundos que cada técnica empleaba en el análsis de un fotograma.

	640x480	320x240
SIFT	38 ms	10 ms
SURF	7.5 ms	1 ms
Lucas Kanade	10 ms	1 ms

Los resultados muestran que la técnica más costosa computacionalmente es SIFT. Por otra parte, entre SURF y Lucas Kanade no existe una diferencia significativa.

Tras comparar las técnicas en los ámbitos de cantidad de puntos extraídos, calidad del seguimiento y el rendimiento computacional, concluimos que la técnica más adecuada para nuestro sistema es Lucas Kanade.

4.8.2. Seguimiento de personas emparejando regiones de interés

Para el seguimiento de personas se plantearon dos opciones. La primera consistió en realizar emparejamientos entre las regiones de interés de un determinado fotograma con las del fotograma posterior. La segunda fue un algoritmo evolutivo, que finalmente fue la técnica escogida. En esta sección se describe la primera alternativa probada, basada en el emparejamiento de regiones de interés.

Primero se analizaron distintos tipos de información que extraída de las imágenes sirviera para el emparejamiento de regiones de interés.

Emparejamiento por histogramas de color

Tras obtener las regiones de interés de las imágenes a través de la segmentación de movimiento, se planteó si se podría emparejar las regiones extraídas de la imagen de una iteración a la siguiente. Para ello se pensó en generar un histograma de color por cada región de una imagen y compararlos con los histogramas obtenidos de la otra imagen.
Para la generación de histogramas se ha utilizado la biblioteca *OpenCV* (ver 3.3), concretamente las siguientes funciones:

```
CvHistogram* cvCreateHist( int dims, int* sizes,
int type,float** ranges=NULL, int uniform=1 );

void cvNormalizeHist( CvHistogram* hist, double factor );

double cvCompareHist( const CvHistogram* hist1,
const CvHistogram* hist2, int method );
```

La función *cvCompareHist* compara dos histogramas y te devuelve una medida de distancia entre ellos. Esta comparación puede realizarse de distintas formas:

- Correlación
- Chi-Square
- Intersección
- Bhattacharyya

Tras averiguar cuáles son los rangos de valores para los distintos tipos de distancias, se realizaron pruebas con todas. No hubo excesiva diferencia entre ellas, por lo que se eligió la distancia de Chi-Square:

$$distHSV = cvCompareHist(hist_{R1}, hist_{R2}, CV\_COMP\_CHISQR) \tag{4.13}$$

$$distanciaHIST(R1, R2) = 1 - (distHSV/MAXDIST); \tag{4.14}$$

donde MAXDIST es una ditancia máxima fijada en 2, quedando el resultado normalizado en el rango [0,1].

Para comprobar la capacidad discriminante de esta característica se implementó una herramienta que permitiera visualmente ver el resultado de la correlación entre una región seleccionada de la imagen y el resto de ella (ver figura 4.32).

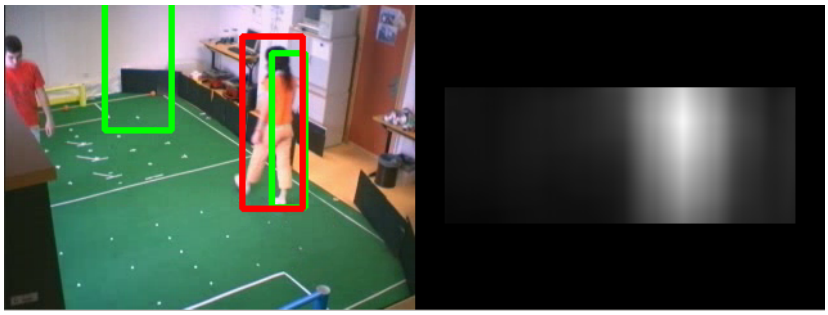

Figura 4.32: Correlación utilizando histogramas HSV.

La región seleccionada manualmente sobre la imagen izquierda se ha dibujado de color rojo. En la imagen derecha se representa el mapa de todas las distancias halladas con todas las posibles regiones del mismo tamaño dentro de la imagen. Los píxeles centrales de las regiones comparadas tomarán valores entre 0 y 255, siendo 255 el valor que indica la máxima coincidencia. Como se aprecia en la imagen de distancias, las zonas más blanqueadas coinciden con la zona seleccionada para la correlación. A medida que se alejan de esa zona, la imagen se va difuminando a negro.

Esta herramienta también permitió ajustar el umbral para decidir si dos regiones coinciden o no en su histograma de colores. Concretamente este umbral lo fijé en 0.6, siendo 1 el valor de máxima coincidencia. En la imagen 4.33 se puede ver el resultado umbralizado de la función distancia entre histogramas.

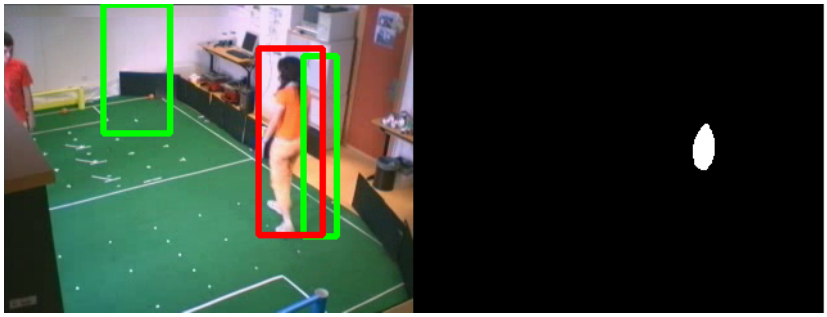

Figura 4.33: Ajuste del umbral para distancia entre histogramas HSV.

Emparejamiento SIFT

Sin entrar en los detalles de la implementación SIFT, simplemente diré que se ha utilizado una biblioteca que a través de *OpenCV* calcula los puntos relevantes de cada región así como sus descriptores y empareja dichos puntos.

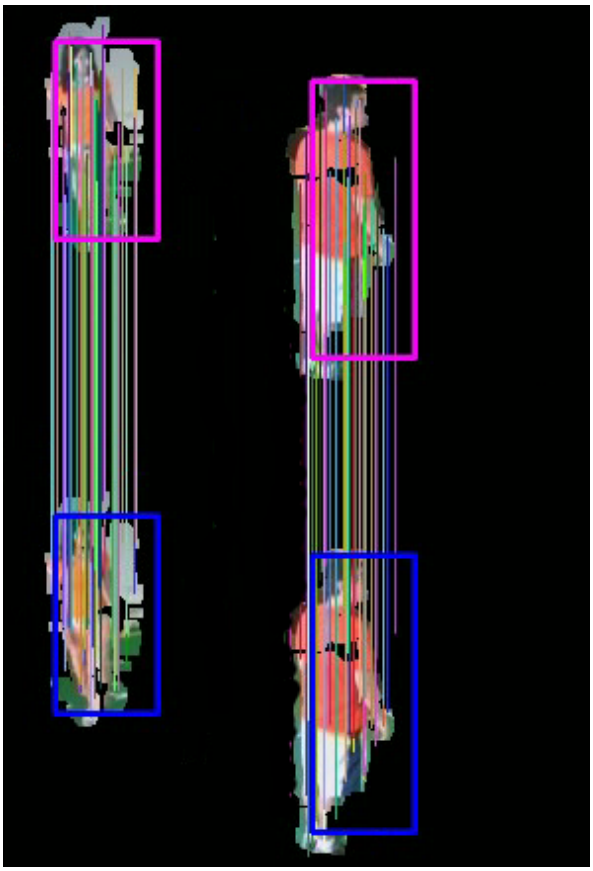

Figura 4.34: Emparejamiento sift.

En la figura 4.34 se puede apreciar la máscara en negro generada por el filtro de movimiento y las regiones calculadas en base a ésta. Cada emparejamiento ha sido

representado mediante una línea de color aleatorio que une los dos puntos. En este caso no hay un desfase entre las imágenes analizadas, lo que facilita los emparejamientos y se obtienen buenos resultados.

Sin embargo, en las pruebas con imágenes de distintas cámaras o de la misma pero con un desfase apreciable se ha podido comprobar que el número de emparejamientos es considerablemente menor.

La *distancia sift* se ha construido en función del número de emparejamientos realizados entre dos regiones. Para ello se ha fijado un número máximo (MAX) de emparejamientos que se toma como referencia para calcular la distancia:

$$distanciaSIFT(R1, R2) = 1,\ si(emparejamientos/MAX) >= 1 \tag{4.15}$$

$$distanciaSIFT(R1, R2) = emparejamientos/MAX,\ si(emparejamientos/MAX) < 1 \tag{4.16}$$

donde MAX se ha fijado en 10 emparejamientos.

Al igual que con la distancia entre histogramas HSV, se ha visualizado a través de una imagen en escala de grises el resultado de la correlación entre una región y el resto de la imagen.

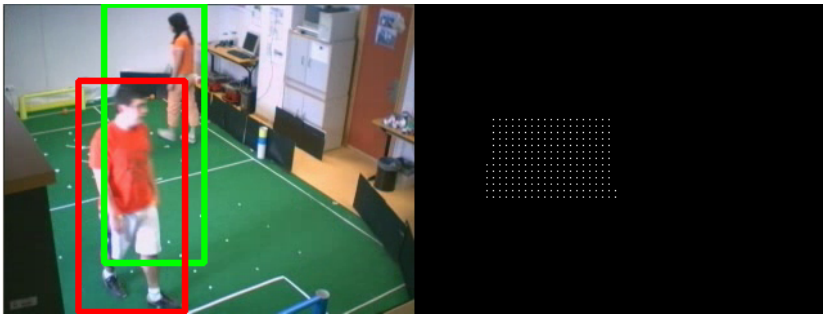

Figura 4.35: Correlación utilizando SIFT.

En este caso, entre las regiones analizadas se ha introducido una distancia de 5 píxeles, ya que SIFT es una técnica costosa computacionalmente. Además, los valores que toma la función distancia son más discretos que en el caso de la distancia entre

histogramas de color, lo que se nota en el resultado que se ve en la imagen a escala de grises. También se ha umbralizado esta distancia y en este caso el mejor umbral es 0,7.

Conclusiones

Los experimentos de correlación mostrados anteriormente hicieron pensar que el seguimiento 2D basado en el emparejamiento de regiones de interés era viable. Sin embargo, el resultado obtenido no fue de la calidad esperada. Por un lado se comprobó que la continuidad de la detección de regiones de interés en ocasiones no era suficiente para mantener el seguimiento. Además, los fallos en el emparejamiento de regiones aumentaban con la introducción de más personas en la escena, en la figura 4.36 se puede ver una secuencia de un seguimiento que se acaba perdiendo.

Figura 4.36: Fallo en el seguimiento con emparejamiento de regiones.

Este tipo de carencias en el seguimiento podían ser resueltas con la introducción de un algoritmo evolutivo como técnica de seguimiento, que es la segunda opción que se probó.

Capítulo 5
Seguimiento 3D

Una vez exploradas las posibilidades del seguimiento de personas en la imagen 2D, se planteó abordar el seguimiento de personas en el espacio tridimensional. Para ello se diseñaron e implementaron dos técnicas distintas y se compararon entre sí. Ambas técnicas son algoritmos de búsqueda en el espacio 3D que generan hipótesis y las validan hasta encontrar las soluciones óptimas.

Cada técnica se ha programado en un componente (o esquema en la tecnología de *Jderobot*[1], ver sección 3.1) diferente. Para ambos esquemas, el individuo 3D propuesto consiste en un prisma con cinco grados de libertad: centro (X,Y,Z), altura (h) y anchura (s). Este individuo 3D marca diferencia con respecto a otros sistemas previos realizados en el grupo de Robótica para la misma aplicación, que usaban el punto 3D como primitiva para representar a las personas.

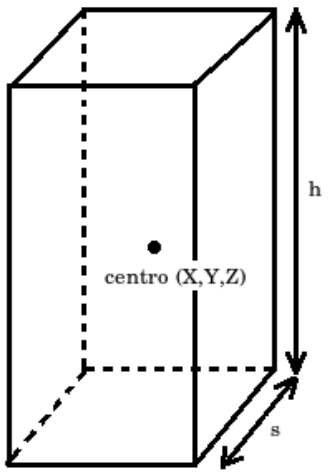

Figura 5.1: Prisma y grados de libertad.

[1] http://jde.gsyc.es/

La primera técnica desarrollada se apoya en el resultado del seguimiento 2D en cada cámara empleada, descrito en el capítulo anterior. Consiste en generar hipótesis 3D en aquellas posiciones 3D en las que probablemente se encuentre una persona por compatibilidad con el seguimiento de esa persona en las imágenes 2D. Después le asigna un valor de calidad a cada hipótesis según se parezcan sus proyecciones en las imágenes a los rectángulos 2D resultado de los seguimientos en la imagen en las diferentes cámaras. El esquema se llama *tracker3D* y se describe con detalle en la sección 5.1.

La segunda técnica consiste en un algoritmo evolutivo multimodal, similar al implementado en [Marugán, 2007], pero que dota de volumen a los individuos empleados. Con ello se pretende mejorar la estimación de las soluciones al problema del seguimiento 3D de múltiples personas. De este modo no sólo se estima la posición 3D de las personas, sino también su tamaño. El algoritmo maneja poblaciones de individuos denominadas razas y cada una se encarga del seguimiento de una persona. Estas poblaciones evolucionan continuamente hasta converger a la solución óptima en cada instante. Este componente se llama *trackerEvolutive3D* y se detalla en la sección 5.2.

5.1. Componente Tracker3D

Este esquema materializa la primera técnica probada. Utiliza la información de varias instancias (una por cada cámara) del componente *tracker2D* para realizar un seguimiento 3D de personas. En la imagen 5.2 se puede ver el diseño de la aplicación.

El algoritmo que estima las posiciones 3D y el tamaño real de las personas parte de la lista de personas que le proporciona cada seguidor 2D. Después, con la ayuda de la biblioteca *Progeo* (ver sección 3.5), genera hipótesis en 3D a lo largo de la línea de retroproyección que parte del píxel central de cada rectángulo representante de las razas en 2D.

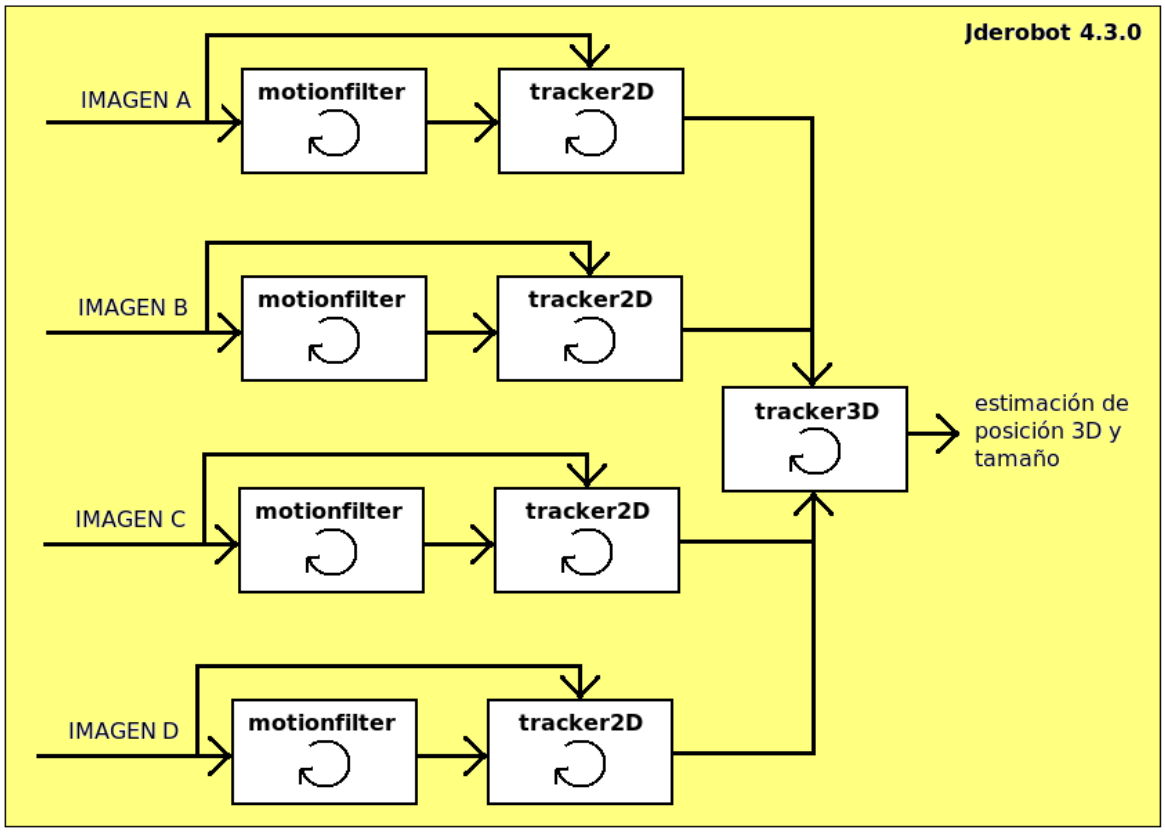

Figura 5.2: Diseño global de la aplicación de seguimiento 3D.

Una vez obtenidas las hipótesis se les asigna un valor de calidad o salud validando sus proyecciones en las imágenes, es decir, si en las otras cámaras proyectan en una persona detectada por el seguimiento 2D tendrán una salud alta. Por último, se seleccionan los prismas con mayor salud. El número de prismas seleccionados (N) corresponderá con el número estimado de personas. Este número es el número máximo de personas que haya detectado una cámara de entre todas las empleadas. En la figura 5.3 vemos un esquema del algoritmo y en la figura 5.4 vemos el diagrama de flujo.

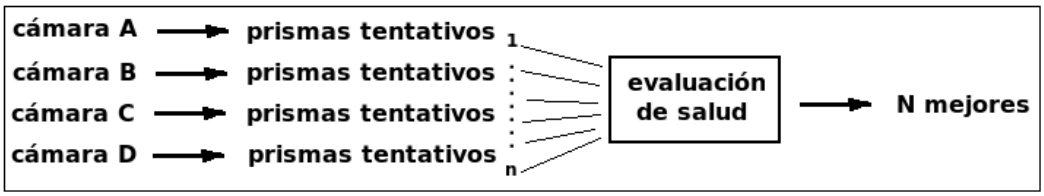

Figura 5.3: Esquema del algoritmo de seguimiento 3D.

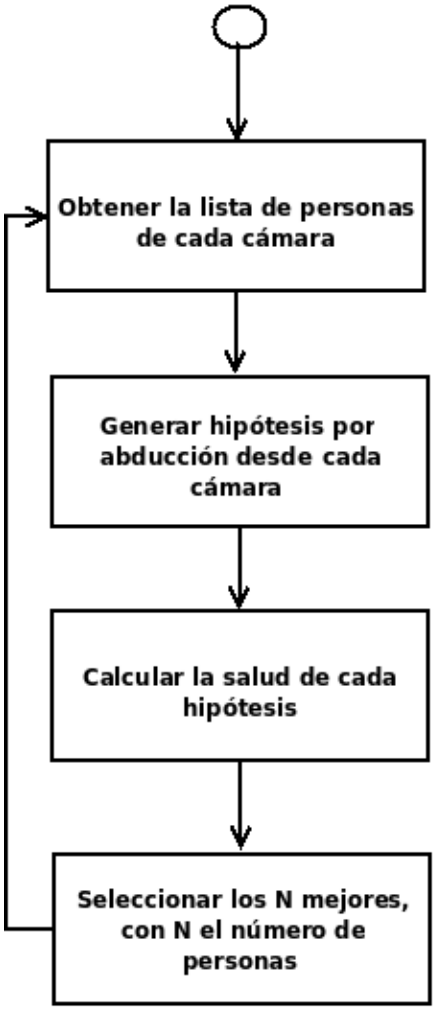

Figura 5.4: Diagrama de flujo del algoritmo de seguimiento 3D.

5.1.1. Generación de hipótesis 3D

Las hipótesis 3D son generadas utilizando la heurística de la abducción a partir de los rectangulos 2D obtenidos del seguimiento bidimensional en las cámaras. El proceso consiste en tirar líneas que unen el foco de la cámara con el píxel central de los rectángulos 2D. Después, sobre esas líneas se sitúan los centros de los prismas tentativos a diferentes profundidades, abarcando todo el largo de la habitación (figura 5.5). La anchura y altura reales de los prismas se estiman generando también líneas abductivas pero que pasen por las esquinas del rectángulo. De este modo, escogiendo la misma profundidad para las cuatro líneas de las cuatro esquinas se hallan los puntos en 3D y seguidamente una altura y una anchura tentativa calculando la distancia entre ellos.

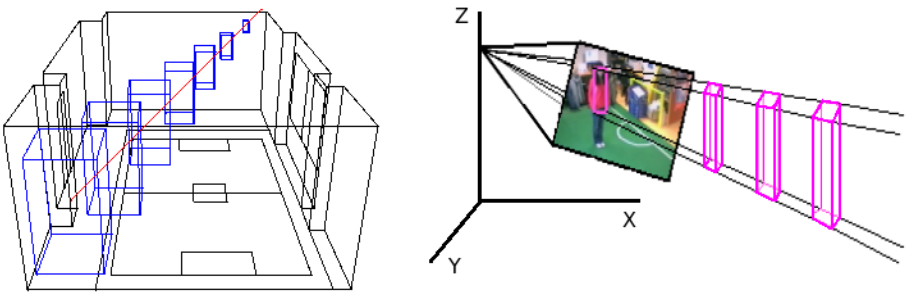

Figura 5.5: Abducción.

Desde cada cámara se generan tantas líneas abductivas como personas haya detectado el seguimiento 2D en la imagen. El muestreo sistemático de la línea abductiva es del orden de 100 muestras, aproximadamente separadas cada 10 centímetros. El objetivo es generar muchas hipótesis de modo que alguna se sitúe muy próxima a la posición real de la persona en 3D. Las hipótesis malas se descartarán más adelante.

5.1.2. Cálculo de la calidad de las hipótesis 3D

Para el cálculo de la calidad de cada prisma primero se proyecta el prisma en cada imagen y después se obtiene la mínima envolvente rectangular de la proyección del prisma en las imágenes (ver imagen 5.6). Estos rectángulos son fácilmente comparables con los rectángulos representantes de cada raza obtenidos del seguimiento 2D.

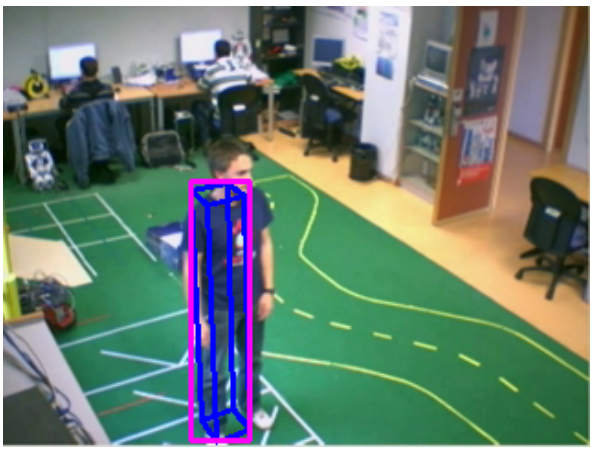

Figura 5.6: Mínima envolvente de un prisma.

Las envolventes son computacionalmente más ágiles de manejar que el polígono irregular en el que proyecta el prisma y no se pierde exactitud de manera significativa. La proyección de un prisma se realiza en todas las cámaras excepto en la cámara desde

la cual fue generado. Si el prisma proyecta lejos de cualquier rectángulo que envuelve una persona en el seguimiento 2D del resto de cámaras, entonces su calidad será baja.

La comparación entre la envolvente de la proyección del prisma (E) y los distintos rectángulos (R) del seguimiento 2D se realiza en función de dos factores:

1. Distancia entre los centros de los rectángulos

La distancia en la imagen entre los rectángulos E y R se calcula mediante la distancia euclídea entre sus centros (c_E y c_R). Para normalizarla se establece una distancia máxima, que se ha fijado que sea la diagonal de la imagen, por tanto este factor normalizado entre [0,1] se calcula así:

$$distancia(E,R) = 1 - \frac{\sqrt{(c_{Ex} - c_{Rx})^2 + (c_{Ey} - c_{Ry})^2}}{distancia_{MAX}} \tag{5.1}$$

2. Porcentaje del área de solapamiento entre los rectángulos.

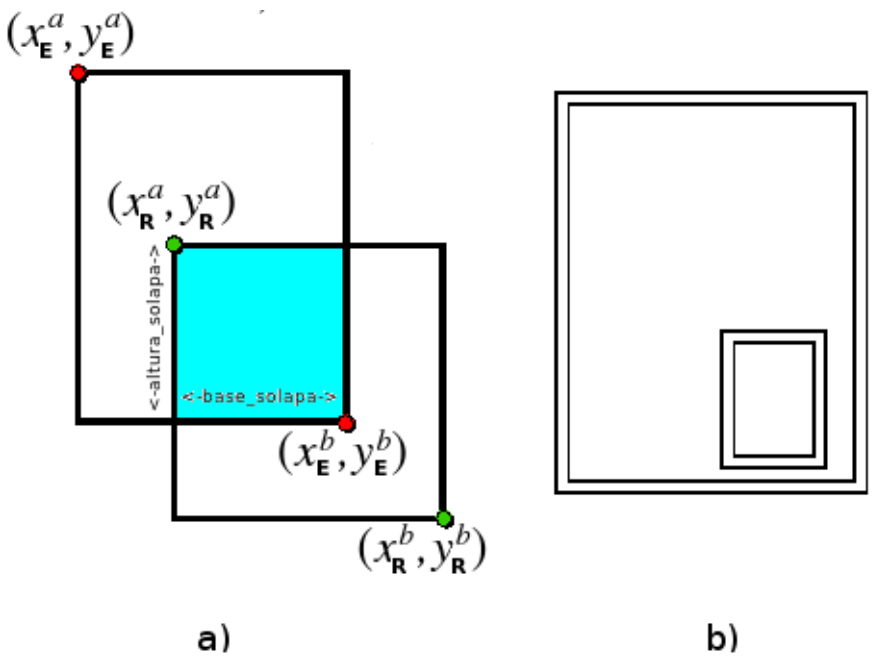

Figura 5.7: a) Solapamiento de dos regiones E y R parametrizadas. b) Región contenida en otra.

Las regiones rectangulares se definen por sus esquinas opuestas (x^a, y^a) y (x^b, y^b), tal y como ilustra la figura 5.7 (a). El porcentaje de solapamiento entre E y R se

determina según las siguientes ecuaciones:

$$solapamiento(E,R) = min\{solapa(E,R), solapa(R,E)\} \tag{5.2}$$

$$solapa(E,R) = \frac{base_{solapa}(E,R) \times altura_{solapa}(E,R)}{(x_E^b - x_E^a)(y_E^b - y_E^a)} \tag{5.3}$$

$$base_{solapa}(E,R) = max\{0, min\{x_E^b, x_R^b\} - max\{x_E^a, x_R^a\}\} \tag{5.4}$$

$$altura_{solapa}(E,R) = max\{0, min\{y_E^b, y_R^b\} - max\{y_E^a, y_R^a\}\} \tag{5.5}$$

Nótese que la función *solapa* representa un porcentaje de solape relativo a cada una de las ventanas. En la figura 5.7 (b) se observa cómo la función *solapa(E,R)* no es equivalente a *solapa(R,E)*, por lo que la función *distancia(E,R)* se define como el valor mínimo de ambas.

La calidad o salud total es la media de la suma ponderada de ambos factores en cada cámara, excepto en la que generó la hipótesis p:

$$calidad(p) = \frac{\sum_{m=1}^{M-1} \alpha \ * \ solapamiento(E,R) \ + \ \beta \ * \ distancia(E,R)}{M-1} \tag{5.6}$$

donde α y β concretamente son 0.4 y 0.6, respectivamente. M es el número de cámaras. E es la envolvente del prisma p y R es un rectángulo-persona resultado del seguimiento 2D en la imagen m.
Los valores de los coeficientes de ponderación se obtuvieron experimentalmente, analizando con qué relación se obtenían mejores resultados.

El criterio del solapamiento está muy relacionado con la distancia euclídea entre los centros de los rectángulos. Sin embargo, si dos rectángulos no solapan, sólo con este criterio no podríamos saber cómo de lejos se encuentran en la imagen y por el contrario, estando a la misma distancia el solape real puede ser menor o mayor, por lo que queda justificado el uso de ambos criterios.

Una vez calculada la calidad de cada prisma tentativo, se seleccionan los N mejores prismas en función del número máximo de personas que hayan detectado las cámaras. Para ello, en primer lugar, se dividen los prismas en grupos según el rectángulo-persona del seguimiento 2D a partir del cuál han sido generados y se comparan entre sí por su calidad, seleccionando el de mejor calidad. Después, entre estos prismas previamente seleccionados, se detecta si hay prismas similares (si se encuentran cerca espacialmente) y se descartan los peores hasta quedarnos con los N mejores prismas, que son el resultado del seguimiento 3D.

5.1.3. Interfaz gráfica

Al igual que la interfaz gráfica de la aplicación de seguimiento 2D, el GUI del sistema de seguimiento 3D (figura 5.14) ha sido un elemento clave para la depuración y visualización de resultados.

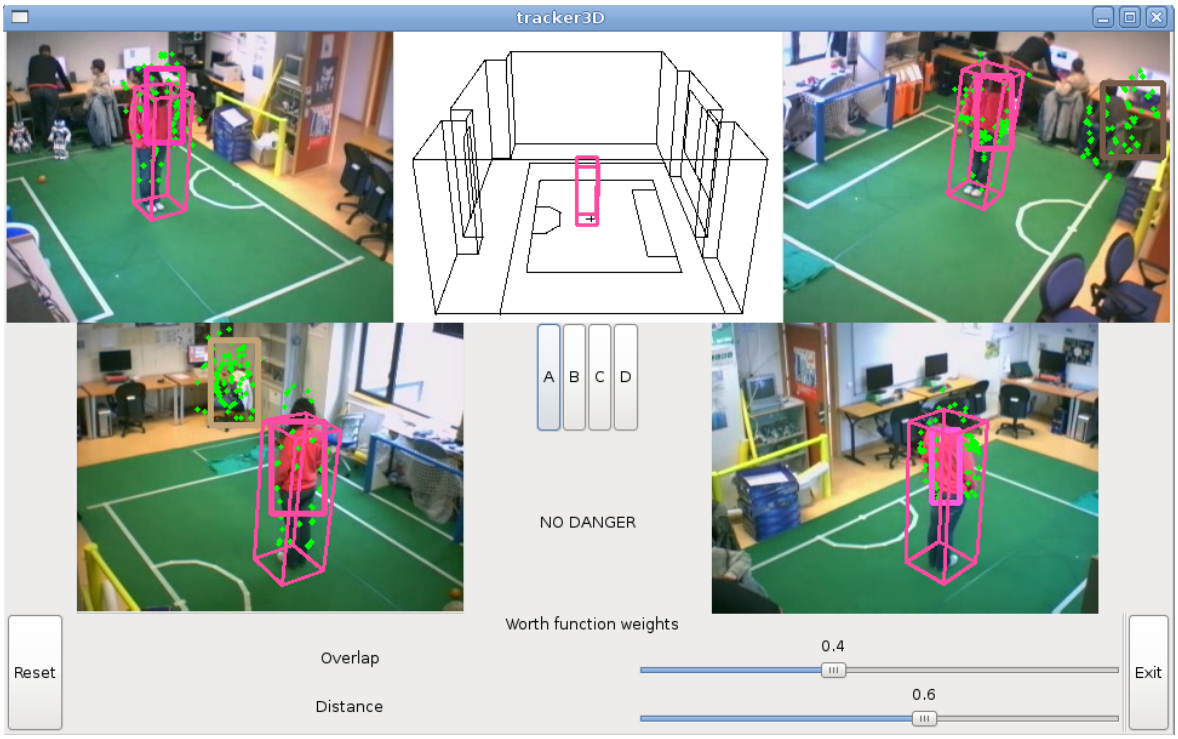

Figura 5.8: Interfaz gráfica de la aplicación.

En la interfaz podemos ver las imágenes que obtienen las cuatro cámaras reales y en el centro la imagen de la cámara virtual, en la que se visualiza un esqueleto de la habitación y el resultado del seguimiento en tres dimensiones.

Los botones *A, B, C y D* permiten seleccionar qué cámaras se tienen en cuenta para el seguimiento. Por defecto, las cuatro cámaras se toman en cuenta.

Debajo de estos botones hay una etiqueta de texto que por defecto muestra "NO DANGER", lo que significa que ninguna de las personas seguidas está caída en el suelo. Si una persona se encuentra tirada en el suelo, el sistema lo detecta y lo indica modificando esta etiqueta para poner "WARNING!!z cambiar su color a rojo.

Los deslizadores *Overlap* y *Distance* modulan los pesos que toman estos factores en la función de calidad que evalúa los prismas, α y β respectivamente.

El botón de *Reset* sirve para resetear el seguimiento actual, eliminando la información que el sistema había guardado hasta el momento.

Por último, el botón de *Exit* sirve para salir de la aplicación.

5.2. Componente TrackerEvolutive3D

Como presentamos en el capítulo 2, uno de los subobjetivos de este proyecto es diseñar e implementar dos alternativas distintas para generar el seguimiento 3D de personas. En la primera sección de este capítulo se ha descrito la primera técnica desarrollada, basada en el seguidor 2D construido también para este proyecto. En esta sección se describe la segunda opción implementada: un seguidor 3D evolutivo multimodal. El algoritmo es muy parecido al que se ha utilizado para el seguimiento bidimensional (ver sección 4.4), únicamente se ha adaptado para utilizar individuos con volumen (prismas) como los presentados al comienzo de este capítulo. Al tener una técnica de seguimiento 3D que se apoya en el seguimiento 2D en cada cámara y otra que no, esto permite estimar indirectamente si el seguimiento 2D aporta algo a la calidad del seguimiento 3D.

En la imagen 5.9 se muestra el diseño global del componente.

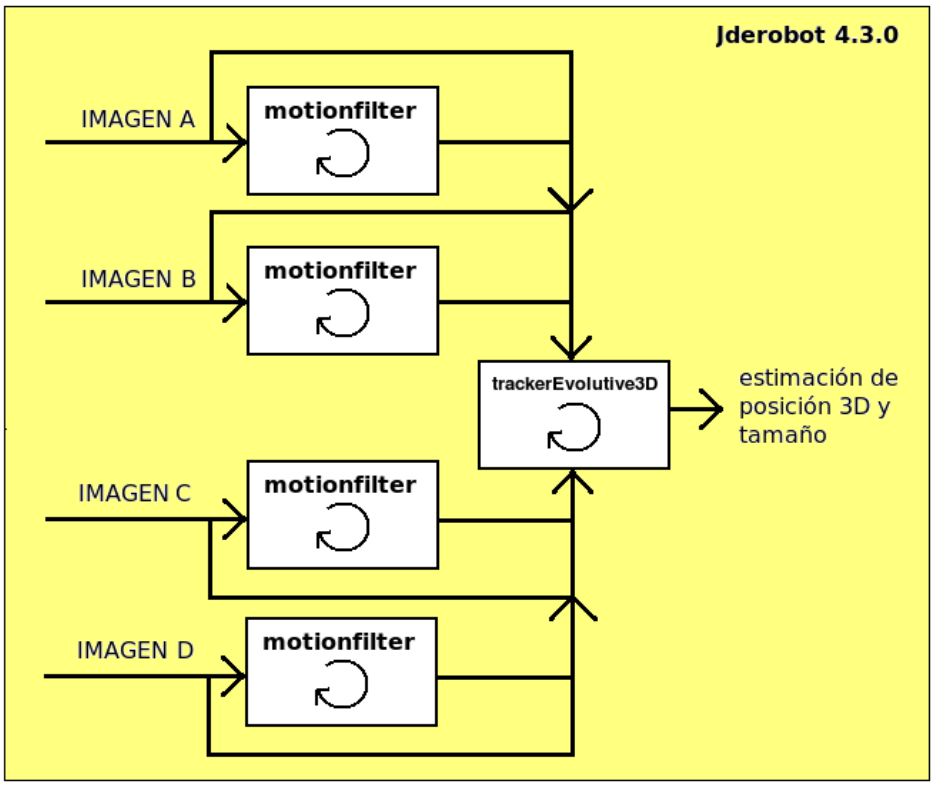

Figura 5.9: Diseño global de la aplicación de seguimiento 3D con el algoritmo evolutivo.

Como vemos, este componente recibe como entrada directamente el resultado de la segmentación de movimiento y como salida ofrece las posiciones 3D y tamaños estimados de las personas en la escena.

5.2.1. Descripción de una raza 3D

En este contexto una raza es un conjunto de individuos que mantienen el seguimiento 3D de una persona, estiman su tamaño y almacenan información sobre ella. Cada individuo representa una posible solución del problema y es calificado con un valor de salud para indicar cómo de buena es esa estimación.

La raza guarda diferentes tipos de información necesaria para su evaluación y evolución, esta información consiste en:

1. Población de individuos o hipótesis 3D.

2. Representante. Es la solución concreta que la raza da al problema de la localización y estimación del tamaño de la persona que sigue. Se detallará más adelante cómo se calcula éste.

3. Histogramas de color en el modelo HSV. *La raza almacena un color característico de la persona por cada cámara.* Esto se ha implementado del mismo modo que en el seguimiento 2D (ver sección 4.3). El motivo de tener un histograma de color por cada cámara es que dependiendo de la iluminación y la posición de la persona respecto a las fuentes de luz, el color de su vestimenta puede variar significativamente de una cámara a otra.

4. Vida. El concepto de vida también es el mismo que el explicado para una raza 2D. Es un índice que indica cómo de bueno es el seguimiento que está manteniendo la raza. Depende directamente de la salud del mejor individuo de la raza, si la salud es mayor que un umbral la vida es aumentada en cada iteración. Por el contrario, si es menor repetidamente, la vida de la raza va disminuyendo hasta que es eliminada. Esto da algo de persistencia a la raza antes de ser eliminada, por si la persona que sigue sufriera una oclusión temporal.

5.2.2. Algoritmo evolutivo multimodal

El algoritmo evolutivo propuesto para el seguimiento 3D consiste en generar una raza 3D por cada persona que aparece en la escena y evolucionar las razas adaptándolas al movimiento de las personas. Los individuos de la razas se evalúan de acuerdo con las observaciones en las imágenes. Si una persona abandona la habitación, las observaciones no alimentan a la raza que la sigue y ésta acaba siendo eliminada.

El algoritmo maneja dos tipos de individuos distintos: *exploradores* y *explotadores*. Los exploradores son prismas tentativos situados mediante abducción en zonas 3D compatibles con el movimiento detectado en las imágenes y los explotadores son prismas que pertencen a una raza que está siguiendo a una persona. Por tanto, la finalidad de los exploradores es una búsqueda de grano grueso en el espacio 3D sólo en las zonas compatibles con el movimiento en las imágenes. En cuanto a los explotadores, su objetivo es una búsqueda de grano fino en la zona en la que se encuentra la persona para converger a la mejor solución.

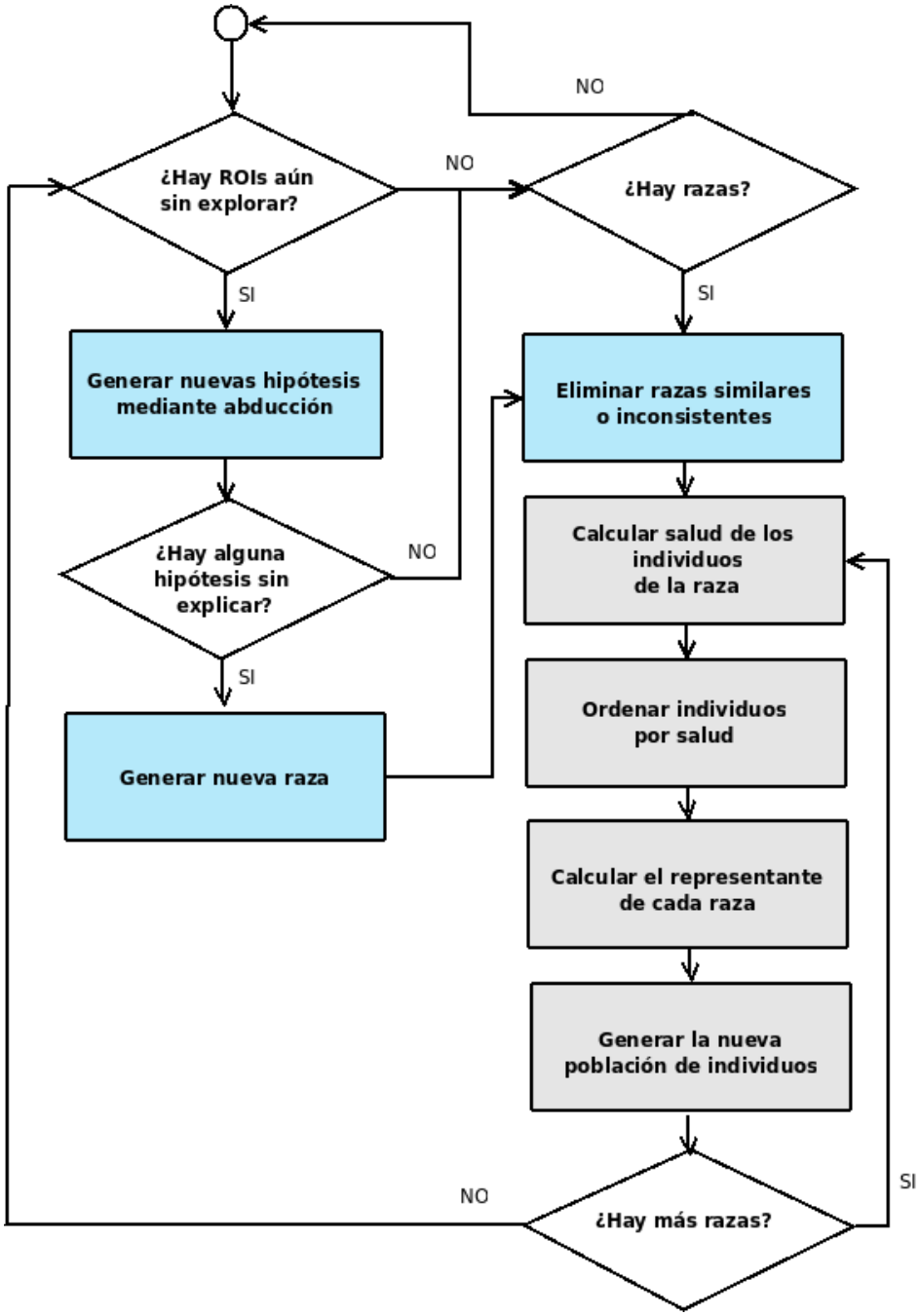

Figura 5.10: Diagrama de flujo del algoritmo evolutivo.

En la figura (5.10) tenemos el diagrama de flujo del algoritmo. Las cajas sombreadas en azul corresponden con los pasos de generación de hipótesis 3D (prismas exploradores), creación y eliminación de razas. Las sombreadas en gris corresponden a los pasos de evaluación y evolución de los individuos una raza (población de explotadores), lo que permite su adaptación al movimiento de la persona.

Creación de razas

El algoritmo utiliza las regiones de interés del segmentador de movimiento para generar las hipótesis 3D (prismas exploradores en este algoritmo) de forma abductiva, del mismo modo que se ha explicado en la sección 5.1.1. Sin embargo, estos individuos no son evaluados en cada cámara para seleccionar entre ellos las soluciones finales, sino que dan lugar a la creación de razas. Después, estas razas evolucionan y convergen hacia las soluciones reales finales.

En primer lugar, el algoritmo distingue entre regiones de interés ya explicadas y las que no lo están. Para ello mide la distancia en la imagen (figura 5.11) entre el centro del ROI de movimiento y el centro de cada uno de los representantes proyectados de las razas que haya en ese momento. Si la distancia es menor que un umbral, en este caso 50 píxeles, se dice que el ROI ya está explicado. Esto supone que dicho ROI no va a generar hipótesis 3D abductivas.

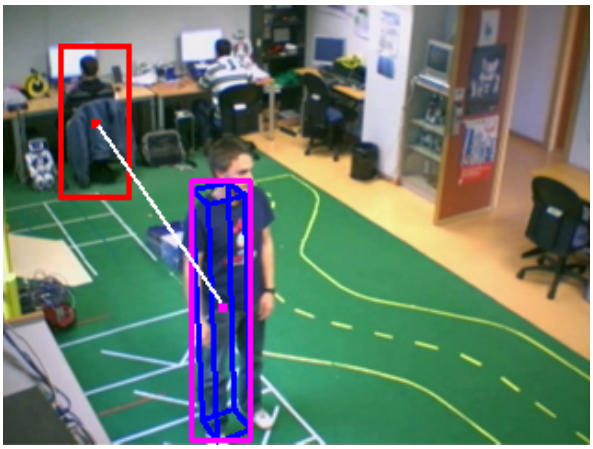

Figura 5.11: Distancia entre un ROI de movimiento y la envolvente del representante de una raza en una imagen.

En el caso de que un ROI de movimiento no esté explicado por una raza, se generan los exploradores de manera abductiva a través de ese ROI y se evalúan con la finalidad de encontrar la persona en 3D que genera ese movimiento en la imagen. Los prismas se generan del mismo modo que ya se ha explicado para el *Tracker3D* (sección 5.1.1) pero se evalúan de distinta manera. En este algoritmo no se tienen rectángulos-persona resultado del seguimiento 2D en cada cámara, por lo que la evaluación de los prismas en este caso se realiza proyectando el prisma en cada cámara y asignándoles una salud, que es la densidad de píxeles de movimiento del interior de su envolvente en la imagen.

La salud final de un prisma explorador será la media de las densidades de todas las cámaras, puesto que se busca la compatibilidad con el movimiento dectectado en todas las imágenes.

En la imagen 5.12 vemos la representación de la evaluación de un prisma explorador.

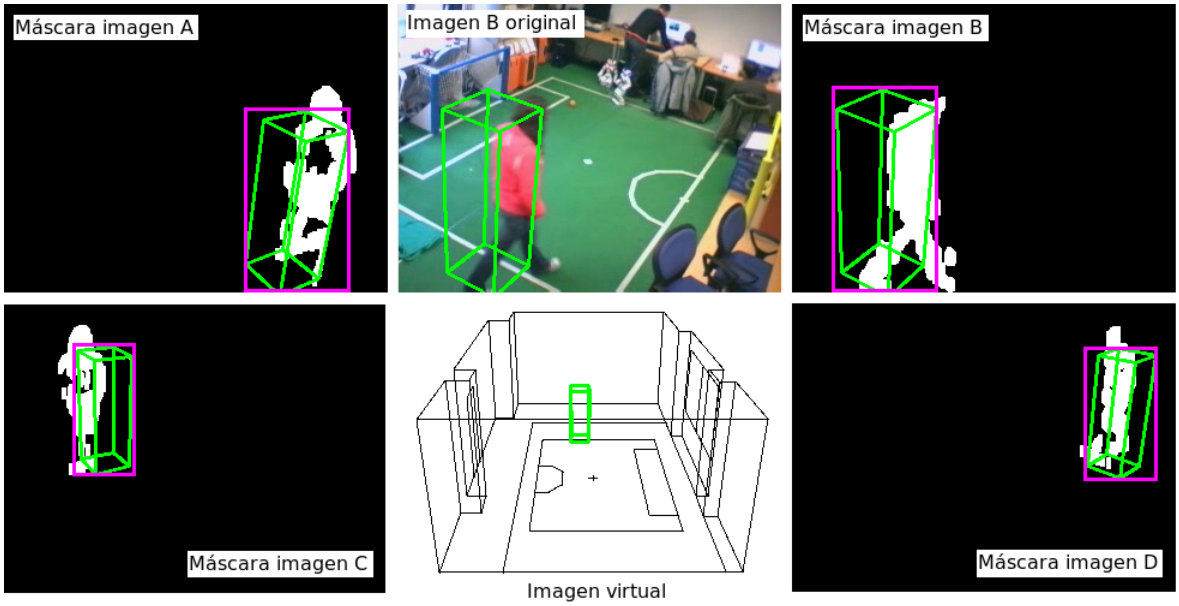

Figura 5.12: Evaluación de la salud de un prisma explorador.

La fórmula para el cálculo de la salud de un prisma explorador p es:

$$salud(p) = \frac{\sum_{m=1}^{M} densidad_m}{M} \tag{5.7}$$

$$densidad_m = K \ / \ (\ ancho(Env_m(p)) \ * \ alto(Env_m(p))\ ) \tag{5.8}$$

Con M el número de cámaras, K el número de píxeles que superan el filtro de movimiento y Env_m la mínima envolvente rectangular del prisma en la cámara m.

Los individuos exploradores cuya calidad supere un umbral pasan a ser candidatos para generar una nueva raza del algoritmo evolutivo. Un candidato se convertirá en una raza si en su vecindad en el espacio 3D no hay ya otra raza, es decir, no está explicado. Esta distancia se mide entre el centro espacial del prisma candidato y el centro del prisma representante de la raza. Cuando un candidato no está explicado significa que una nueva persona ha entrado en la escena y es necesario crear una raza para su

seguimiento.

Una vez explicado cómo se crea una nueva raza en este algoritmo, el proceso de evaluación de la raza es similar al utilizado en el algoritmo de seguimiento 2D (ver sección 4.4) y se detalla en la sección 5.2.3.

Eliminación de razas inconsistentes o similares

Una raza inconsistente significa que no está siguiendo adecuadamente a una persona, bien porque la haya perdido o porque la persona haya salido de la habitación. Para eliminar este tipo de razas el primer paso es comprobar que la vida de la raza se mantiene por encima de un umbral, si no lo está, significa que esa raza no está realizando un buen seguimiento y debe ser eliminada.

Otra comprobación necesaria es comparar la raza con las demás, para evitar que haya razas similares. Esta comprobación se basa en dos factores:

- Distancia XY en 3D entre los centros de los prismas representantes de las razas.
- Parecido en el color.

Para el cálculo de la distancia entre razas se utiliza la distancia euclídea únicamente con las coordenadas X e Y, ya que se considera que no puede haber dos personas en la misma coordenada XY por la restricción de que las personas están apoyadas en el suelo. El umbral de distancia para considerar que dos razas son similares es 700 milímetros.

Para la comparación en términos de color se obtienen los valores (H,S,V) representativos del histograma para cada raza en la cámara que esté siendo comparada y se hace una diferencia absoluta por cada canal. Después, se comprueba si la diferencia en los tres canales a la vez es inferior a un umbral dado para cada canal. Si la diferencia en al menos dos cámaras es inferior al umbral, se considera que las razas son similares en color.

De este modo, tenemos que si dos razas se encuentran próximas en el espacio 3D y tienen colores similares, el algoritmo las fusiona. Sin embargo, si tienen colores diferentes aunque estén cerca no las fusiona.

Generación de una nueva población

Para la generación de una *nueva población de prismas explotadores* pertenecientes a una raza se utilizan los siguientes operadores genéticos:

- *Elitismo*: consiste en mantener sin cambios los mejores individuos de la raza en la siguiente población. Esto permite que se tomen como referencia para la solución a la localización de la persona y para la generación de nuevas hipótesis cercanas a ellos.
- *Ruido térmico*: escoge aleatoriamente individuos elitistas y les aplica una pequeña modificación también aleatoria. La modificación se hace en cada uno de los grados de libertad del individuo, en este caso cinco (X,Y,Z,h,s). Este operador es necesario para la adaptación al movimiento 3D de la persona, ya que explora el tamaño adecuado y la posición adecuada de ésta.

Los pesos para cada operador han sido fijados en 25 % para el elitismo y por consiguiente 75 % para el ruido térmico.

Cálculo del representante de una raza

El representante de una raza es un individuo 3D con el que se realizan las comparaciones y los cálculos referentes a la raza, por lo que es importante cómo se define.

En un primer lugar se pensó que el representante debía ser el individuo con mayor salud, sin embargo al llevarlo a la práctica vimos que de una iteración a otra era demasiado brusco el cambio de representante. Por ello, se decidió que el representante se calculara como una media entre los individuos elitistas (los de mayor salud) de la raza. De este modo se apreciaba mayor suavidad en la evolución del representante a lo largo del tiempo, a la vez que suficiente dinamismo para seguir el movimiento de la persona.

5.2.3. Computación de la salud

En esta sección se muestra cómo se realiza el cálculo de salud de cada individuo explotador de una raza. La función salud permite valorar qué individuos representan

una mejor solución que otros, por lo que resulta de gran importancia que ésta sea lo más discriminante posible.

A diferencia del cálculo de salud de los individuos exploradores, para los explotadores se utilizan dos tipos de información: color y movimiento. El filtro de color se genera por cada cámara a través de los histogramas de color aprendidos de la raza y el filtro de movimiento es proporcionado por el componente *motionfilter*.

Antes de explicar los criterios para el cálculo de salud de un prisma, conviene recordar que para trabajar a nivel de imagen primero se proyecta el prisma en cada una de las cámaras y después se halla la mínima envolvente de su proyección en cada una (figura 5.6). En el cálculo de la salud de un prisma se tiene en cuenta la vecindad del prisma, para ello se utiliza en los cálculos el prisma original y un prisma engordado un 30 % a partir del original, es decir, el prisma engordado contiene el prisma original y su vecindad. En la siguiente imagen 5.13 se pueden ver la envolvente del prisma original y del prisma engordado sobre las imágenes.

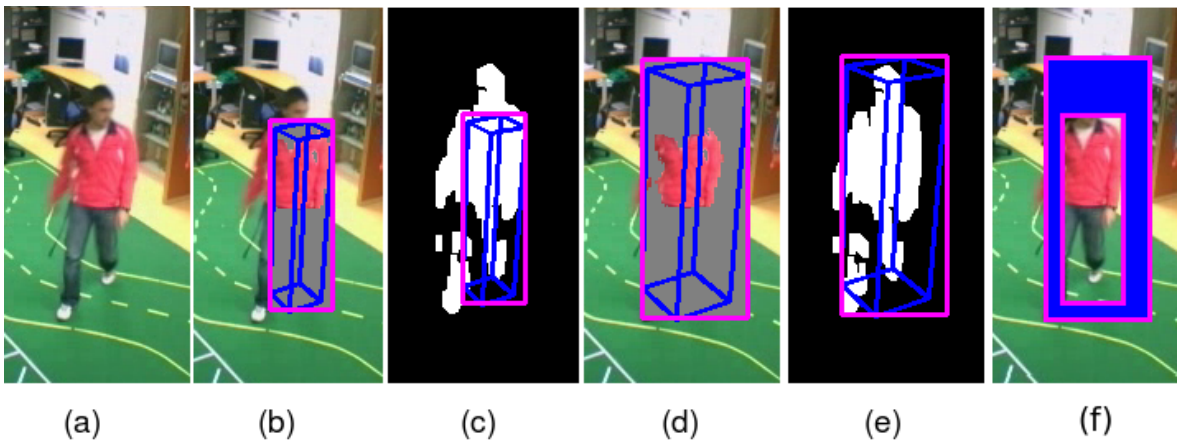

Figura 5.13: Cálculo de salud -(a) Imagen original, (b) Prisma original proyectado y filtro de color, (c) Prisma original proyectado y máscara de movimiento, (d) Prisma engordado proyectado y filtro de color,(d) Prisma engordado proyectado y máscara de movimiento, (f) En azul, área de la vecindad del prisma en la imagen.

Al igual que la función salud en el algoritmo de seguimiento 2D, la salud se define en términos de densidad de píxeles que superan el filtro de color y el de movimiento. En este caso la densidad de píxeles se calcula en la envolvente del prisma original y en su vecindad, que es la diferencia entre la envolvente del prisma engordado y la envolvente del prisma original (ver imagen 5.13 (f)). Si un prisma posee una alta

densidad de píxeles de movimiento y color en su interior y pocos en su vecindad, la salud del prisma será alta. La finalidad es que los prismas se adapten adecuadamente al tamaño de las personas, para ello la densidad de píxeles de color y de movimiento en la vecindad del prisma original deberá tender a cero.

La salud de un prisma se calcula en función de cuatro criterios:

1. Densidad de color respecto a la envolvente del prisma proyectado.
2. Densidad de movimiento respecto a la envolvente del prisma proyectado.
3. Densidad de color en la vecindad del prisma.
4. Densidad de movimiento en la vecindad del prisma.

A continuación se presentan las fórmulas para el cálculo de cada factor.

Densidad de color

Para calcular la densidad de color (DC) del individuo i en la cámara m se necesitan los valores representativos (H,S,V) del histograma de color para dicha cámara y la envolvente Ei_m de la proyección del individuo. La fórmula que calcula si un píxel de color (h,s,v) supera el filtro de color es la misma que para el cálculo de la salud en el algoritmo de seguimiento 2D (sección 4.5).

El cálculo de la salud de color para el individuo i se realiza de la siguiente forma:

$$DCi_m = K \ / \ (\ ancho(Ei_m) \ * \ alto(Ei_m)\ ) \tag{5.9}$$

Siendo K el número de píxeles que supera el filtro de color dentro de la envolvente Ei en la cámara m.

La densidad total de color es la media de la densidad medida en cada una de las cámaras:

$$DCi = \frac{\sum_{m=1}^{M} DCi_m}{M} \tag{5.10}$$

con M el número de cámaras.

Densidad de movimiento

Con la máscara de movimiento proporcionada por el componente *motionfilter* se calcula de manera semejante la densidad de movimiento:

$$DMi_m = K \ / \ (\ ancho(Ei_m) \ * \ alto(Ei_m)\ ) \tag{5.11}$$

Siendo K el número de píxeles que superan el filtro de movimiento dentro de la envolvente Ei en la cámara m.

La densidad total de movimiento es la media de la densidad medida en cada una de las cámaras:

$$DMi = \frac{\sum_{m=1}^{M} DMi_m}{M} \tag{5.12}$$

con M el número de cámaras.

A continuación se presentan las fórmulas para el cálculo de la diferencia de densidades entre el prisma original y el engordado un 30 %.

Densidad de color de la vecindad

La salud basada en la densidad de color de la vecindad (DCV) del prisma i será mayor cuanto menor sea la densidad. Esto es porque esos píxeles pertenecen al entorno de la persona y cuantos menos haya el prisma estará mejor ajustado al tamaño de la persona. Para la cámara m se calcula de la siguiente forma:

$$DCVi_m = 1 \ - \ \frac{PixC_{Gi} - PixC_i}{area_{vecindad}} \tag{5.13}$$

$$area_{vecindad} = area_{Gi} - area_i \tag{5.14}$$

siendo $PixC_{Gi}$ el número de píxeles de color dentro de la envolvente del prisma engordado y $PixC_i$ el número de píxeles de color dentro de la envolvente del prisma original. Los valores $area_{Gi}$ y $area_i$ son el área de las envolventes del prisma engordado y el original, respectivamente.

La densidad total de color de la vecindad es la media de la densidad medida en cada una de las cámaras:

$$DCVi = \frac{\sum_{m=1}^{M} DCVi_m}{M} \tag{5.15}$$

Densidad de movimiento de la vecindad

La salud basada en la densidad de movimiento de la vecindad (DMV) del prisma i para la cámara m es:

$$DMVi_m = 1 \; - \; \frac{PixM_{Gi} - PixM_i}{area_{vecindad}} \tag{5.16}$$

$$area_{vecindad} = area_{Gi} - area_i \tag{5.17}$$

siendo $PixM_{Gi}$ el número de píxeles de movimiento dentro de la envolvente del prisma engordado y $PixM_i$ el número de píxeles de movimiento dentro de la envolvente del prisma original. Los valores $area_{Gi}$ y $area_i$ son el área de las envolventes del prisma engordado y el original, respectivamente.

La densidad total de movimiento de la vecindad es la media de la densidad medida en cada una de las cámaras:

$$DMVi = \frac{\sum_{m=1}^{M} DMVi_m}{M} \tag{5.18}$$

con M el número de cámaras.

SALUD TOTAL DE UN INDIVIDUO i

La salud de un individuo i es la combinación lineal de los cuatro factores explicados:

$$H_i = \alpha * DC_i \; + \; \beta * DM_i \; + \; \gamma * DCV_i \; + \; \omega * DMV_i \tag{5.19}$$

Con α y β igual a 0,35 y γ y ω igual a 0,15. Estos valores se hallaron de manera experimetal.

5.2.4. Interfaz gráfica

La interfaz gráfica del esquema *TrackerEvolutive3D* es muy similar a la del esquema *Tracker3D* (sección 5.1.3).

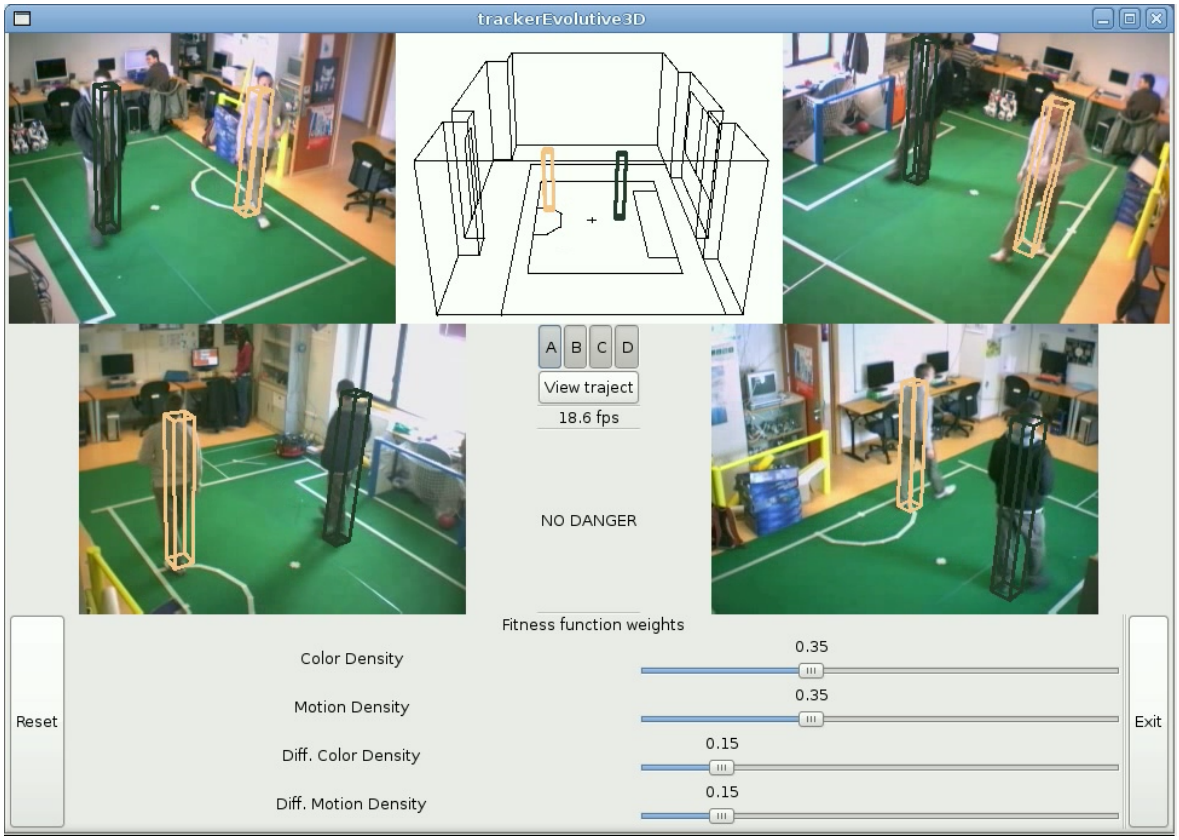

Figura 5.14: Interfaz gráfica de la aplicación.

Se le ha introducido una etiqueta que permite ver las iteraciones por segundo a las que el esquema está funcionando y se le ha añadido el botón *View traject*, que al activarlo, hace que en la imagen virtual se dibujen las trayectorias que las personas han ido realizando.
Por último, se le han añadido los deslizadores que permiten modular los coeficientes o pesos de los distintos factores que intervienen en la salud de los individuos. Por defecto están fijados en los valores que experimentalmente se han probado mejores.

5.3. Experimentos

Una vez presentadas las dos técnicas de seguimiento 3D desarrolladas, en este capítulo se expondrán los distintos experimentos realizados para validar su correcto funcionamiento, ajustar los parámetros de los sistemas y poder compararlas.

La técnica de seguimiento 3D que ha mostrado mejores prestaciones es el *algoritmo evolutivo multimodal con hipótesis en 3D*, por lo que es el que elegimos como referencia.

La decisión se argumenta en la sección 5.3.2 a través de una comparativa entre ambas técnicas.

5.3.1. Ejecución típica

El sistema se ha probado en un entorno concreto, que es el laboratorio de Robótica del edificio departamental II de esta universidad, cuyo volumen es aproximadamente 120 metros cúbicos. Las cámaras de videoconferencia se han situado a la altura del techo en las cuatro esquinas de la habitación y mirando hacia el centro de ésta.

La configuración del sistema se explicó en detalle en el capítulo 3. Cada cámara está conectada a un equipo y sus imágenes son servidas a través de la red utilizando el servicio *networkserver* de *Jderobot*[2]. El equipo principal ejecuta el sistema de seguimiento y se encarga de obtener las imágenes de las cámaras a través del *driver networkclient*. Las características de este equipo son: Pentium Quad Core a 2.4 GHz, 4MB de RAM y sistema operativo GNU/Linux, concretamente la distribución *Ubuntu*.

El sistema de seguimiento 3D desarrollado es capaz de seguir en tiempo real a varias personas dentro de una habitación (se ha probado hasta tres personas), estimando de forma continua su posición 3D, su tamaño y el color de su vestimenta. La aplicación *ejecuta a unas 25 ips*, velocidad suficiente para seguir correctamente personas andando por la estancia.

En el *mediawiki*[3] del proyecto se pueden ver vídeos del funcionamiento de la aplicación. Las siguientes imágenes muestran el seguimiento de tres personas.

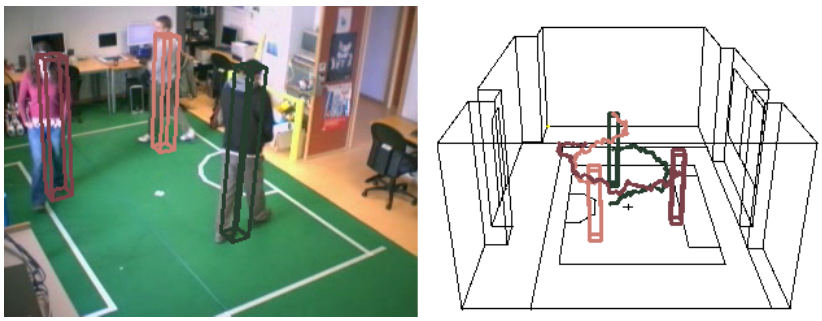

Figura 5.15: Ejecución típica - trayectorias

[2]http://jde.gsyc.es/
[3]http://jde.gsyc.es/index.php/Salons-ii

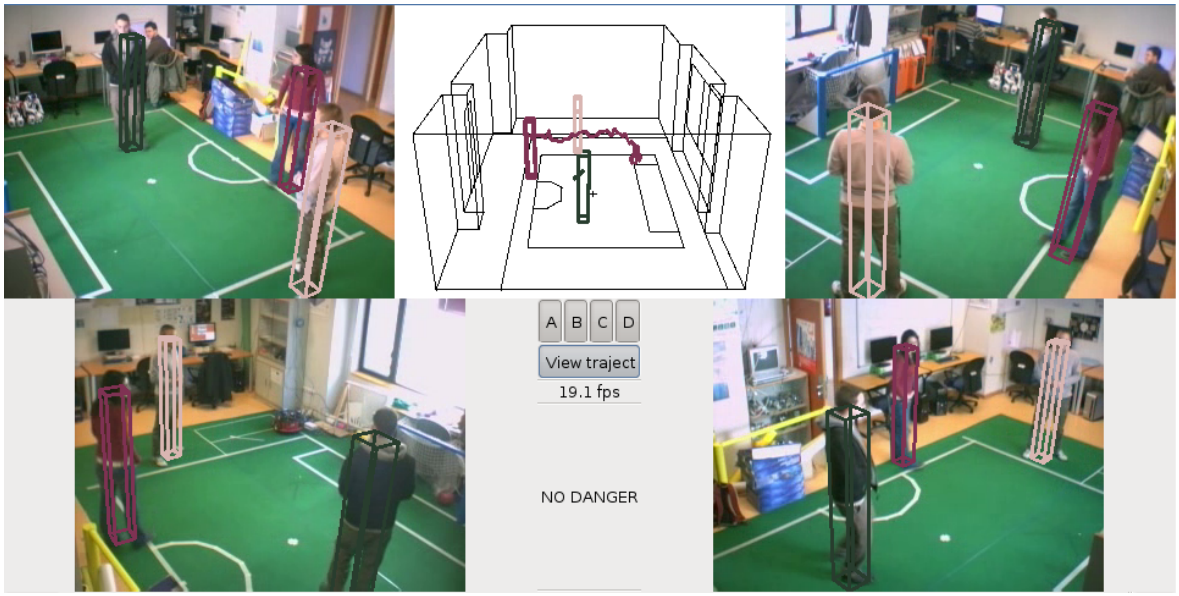

Figura 5.16: Ejecución típica.

En cuanto a la calidad de la estimación del sistema, el error en la posición 3D es de aproximadamente *5 centímetros* en distancia euclídea y respecto al tamaño y el color la estimación que se aprecia es bastante buena.

El seguimiento de las personas es coherente. Al entrar una persona en la habitación se genera una nueva raza para su seguimiento, que mantiene hasta que la persona sale de la escena y por tanto su raza asociada se elimina. Las razas se adaptan adecuadamente al movimiento de las personas, por ejemplo, cuando una persona se agacha la raza estima correctamente su altura (ver figura 5.17).

Figura 5.17: Adaptación de la raza al tamaño de la persona.

En cuanto a la estimación del color de la vestimenta de las personas, se ha comprobado que es suficientemente buena para ayudar a su seguimiento. El sistema

aprende todo tipo de colores sin restricción alguna. Esto supone un paso importante respecto al proyecto fin de carrera anterior [Marugán, 2007], que no era capaz de aprender colores claros u oscuros.

5.3.2. Comparativa

En esta sección se expone la comparación entre las dos técnicas de seguimiento 3D implementadas: seguimiento 3D apoyado en un seguimiento 2D por cada cámara (sección 5.1) y seguimiento 3D con un algoritmo evolutivo multimodal (sección 5.2), que usa las imágenes directamente.

Cada técnica ha sido valorada según distintos criterios. Por un lado se ha medido la eficiencia temporal, por otro la calidad del seguimiento y finalmente la precisión de la estimación ofrecida. Contrastando las aplicaciones podemos concluir cuál de ellas es la mejor.

Rendimiento temporal

El rendimiento temporal es un factor importante en cualquier sistema software, pero en el caso de las aplicaciones en tiempo real es un factor clave. La aplicación desarrollada en este proyecto se encuentra dentro de ese tipo de sistemas, por lo que es imprescindible tener muy en cuenta esta medida y optimizar lo máximo posible los procesamientos. Por ejemplo, para convertir este sistema en un producto real de mercado, sería interesante que pudiera ser ejecutado en placas del tipo *Pico ITX* o *Mini ITX*, ya que se conseguiría un sistema de tiempo real ejecutando en un microprocesador barato.

La medida de rendimiento con que la se ha trabajado en este proyecto es las *iteraciones por segundo* que la aplicación puede llegar a ejecutar. Los esquemas de *Jderobot* se configuran a una frecuencia nominal alta (por ejemplo a 60 ips) y se ve hasta qué frecuencia llega en su ejecución normal. Si no hay suficiente CPU para ir a esa frecuencia configurada, la plataforma degrada automáticamente las iteraciones por segundo del esquema hasta el máximo posible. Cuanto mayor sea el coste computacional de cada iteración, menor será esa frecuencia máxima.

Para medir y comparar el rendimiento de ambas técnicas es muy importante

establecer unas condiciones determinadas, ya que influyen notablemente en el rendimientos factores como el número de cámaras empleadas o la cantidad de personas en las escena. De este modo, se ha fijado el número de cámaras en cuatro y se han tomado datos para una, dos y tres personas.

Los resultados para diferente número de personas, en iteraciones por segundo, se han recogido en la siguiente tabla comparativa:

	tracker3D	trackerEvolutive3D
1 persona	33.6	32.7
2 personas	32.2	27.1
3 personas	30.1	19.1

Los datos obtenidos muestran que para el seguimiento de una persona ambas técnicas alcanzan el mismo rendimiento. Sin embargo, a medida que se introducen más personas el rendimiento de *trackerEvolutive3D* cae notablemente pero sin perder tiempo real. Evolucionar la población de los individuos del algoritmo evolutivo 3D es computacionalmente más costoso que evaluar los prismas tentativos generados por *tracker3D* en las zonas compatibles con el seguimiento 2D en las imágenes.

Calidad del seguimiento

Nos referimos con calidad del seguimiento a que el seguimiento de una persona sea coherente y continuo, es decir, que se inicie al aparecer la persona, continúe mientras ésta se encuentre en escena siguiendo su movimiento con suavidad y desaparezca al abandonar la habitación.

Las pruebas realizadas muestran que en general las dos técnicas implementadas realizan un seguimiento 3D de buena calidad en los términos que hemos explicado. En la figura 5.18 podemos ver unas imágenes extraídas de un vídeo en el que una persona entra en el laboratorio, permanece en él unos instantes y después se marcha. El sistema de seguimiento utilizado en este caso era *trackerEvolutive3D*.

Figura 5.18: Aparición y desaparición del seguimiento de una persona.

Realizando pruebas con más personas sí se aprecia una diferencia entre las dos técnicas. Con *tracker3D* se dan más errores a la hora de asignar un prisma a cada persona (ver figura 5.19) ya que ese proceso únicamente se basa en situar tentativamente prismas que se evalúan para que sean compatibles con el seguimiento 2D en las imágenes en cada instante. Además, el número de prismas que selecciona es el número de personas estimado, que es el máximo de personas que hayan detectado los seguimiento 2D en las cámaras, lo que es menos fiable ya que uno de los seguimientos 2D puede ser malo y detectar más personas de las que hay realmente. Por último, el seguimiento 2D evolutivo no es tan robusto como el seguimiento 3D en situaciones de cruces u oclusiones de personas y esto afecta directamente al seguimiento 3D apoyado en él.

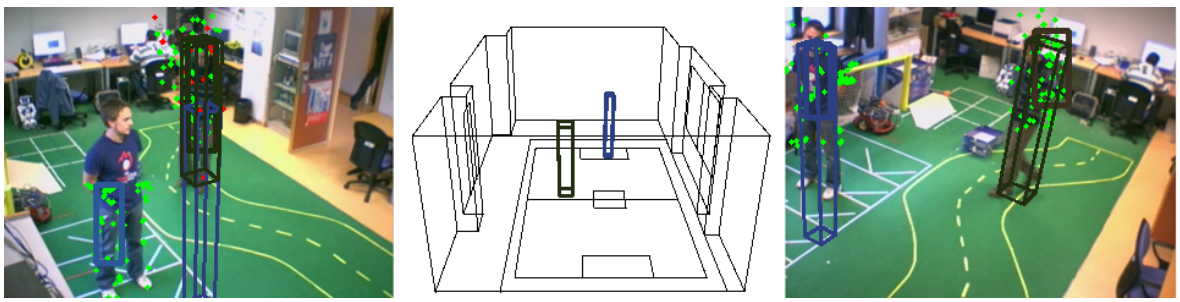

Figura 5.19: Error en el seguimiento 3D de *tracker3D*.

Con la técnica de *trackerEvolutive3D* es muy poco frecuente que este tipo de error ocurra ya que parte de la información abductiva para crear las razas del algoritmo evolutivo y éstas son evaluadas posteriormente, por lo que si realmente no están situadas donde se encuentren las personas obtienen poca salud y son eliminadas.

Error en la estimación

Medir el error cometido en la estimación dada por el sistema también es imprescindible. En nuestro sistema, los valores estimados son la posición 3D de la persona, su altura y anchura y el color de su vestimenta. Para caracterizar la estimación de la posición 3D de las personas, se ha realizado un experimento con cada técnica que consiste en fijar varios puntos 2D conocidos en el suelo del laboratorio y que una persona se situe en dichos puntos (figura 5.20). Tomando como coordenada Z el centro del torso de la persona (aproximadamente a 1200 milímetros del suelo), se calcula la distancia entre las posiciones 3D estimadas por cada técnica y las reales.

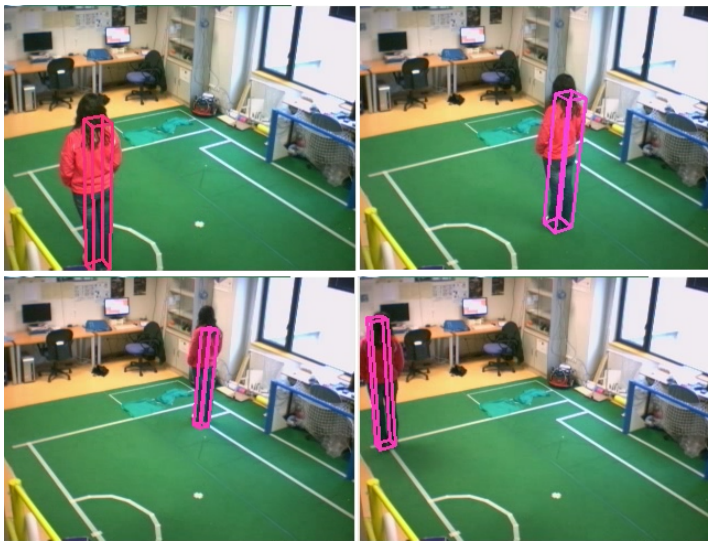

Figura 5.20: Medida del error en diferentes posiciones 3D.

La tabla de resultados para el esquema *tracker3D* es la siguiente:

Punto	Errores en distancia euclídea (mm)
P1	84.698
P2	72.148
P3	92.096
P4	112.357
p5	135.224

La tabla de resultados para el esquema *trackerEvolutive3D* es:

Punto	Errores en distancia euclídea (mm)
P1	54.311
P2	37.004
P3	45.175
P4	60.803
p5	72.455

Los resultados obtenidos muestran que *trackerEvolutive3D* estima mejor la posición 3D de la persona. Concretamente, el esquema *tracker3D* da un error aproximado de 10 centímetros frente al error de 5 centímetros en el caso de *trackerEvolutive3D*. Esto es lógico puesto que *tracker3D* genera prismas tentativos a lo largo de la línea abductiva que une un rectángulo-persona del seguimiento 2D y el foco de la cámara, por lo que la búsqueda del prisma solución en la zona en la que se encuentra la persona no es tan exhaustiva como la búsqueda que realiza una raza 3D del algoritmo evolutivo, cuyos individuos son modificados en tamaño y posición con el fin de explorar localmente las mejores soluciones.

En cuanto al tamaño estimado de la persona, el error en el ancho y el alto lo hemos evaluado de manera cualitativa viendo los resultados pintados sobre las imágenes. Ambas técnicas suelen estimar un ancho menor que el real y un alto que no tiene en cuenta la cabeza de la persona ni las piernas (ver imagen 5.21). Hay que destacar que en este proyecto se ha dado más importancia a la estimación de la posición 3D que a la estimación del tamaño.

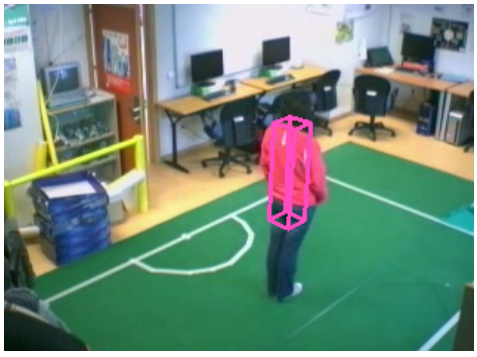

Figura 5.21: Altura estimada.

En todos los experimentos los prismas se dibujan apoyados en el suelo para facilitar la visualización de la posición de la persona en el suelo.

Respecto a la estimación del color, se realiza del mismo modo en las dos técnicas y también de manera cualitativa puede verse que es suficientemente buena para el buen funcionamiento de los algoritmos. Además, cabe recordar que el color se redefine cada cierto intervalo de tiempo.

La conclusión extraída de los experimentos de esta sección es que, en contra de lo que pudiera parecer, al incluir un análisis 2D por cada cámara el seguimiento 3D es más rápido que el algoritmo evolutivo con individuos en 3D. Evolucionar la población de estos individuos es computacionalmente costoso pero aporta más calidad al seguimiento, siendo mucho más robusto. Por estos motivos, para el sistema final se ha seleccionado la técnica del algoritmo evolutivo de seguimiento 3D de personas.

5.3.3. Pruebas con distinto número de cámaras

En todas las pruebas presentadas anteriormente se utilizaron cuatro cámaras. El error en la estimación de la posición 3D de una persona por el sistema es aproximadamente de 5 centímetros. Este valor se calculó haciendo una media entre las medidas tomadas en distintos puntos de la habitación. En esta sección se expone el experimento que mide el impacto del número de cámaras empleado en el seguimiento. Concretamente en el error cometido en la estimación de la posición 3D.

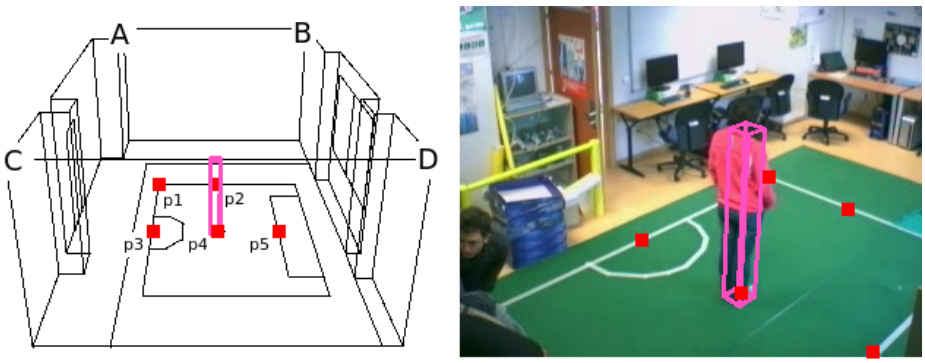

Figura 5.22: Experimento utilizando distinto número de cámaras.

En cuanto al seguimiento, las pruebas confirmaron que el algoritmo era capaz de seguir coherentemente varias personas con al menos dos cámaras. Como consecuencia, vimos que era necesario medir el aporte que daba la introducción de más cámaras en el sistema, además de cubrir una mayor zona de la estancia. Para este experimento se fijaron cinco puntos 3D de referencia (ver imagen 5.22) que son visibles desde todas las

cámaras.

Los resultados utilizando cuatro, tres y dos cámaras fueron:

- Con cuatro cámaras.

Punto	Error en X	Error en Y	Error en Z	Error total (mm)
P1	44.554	18.994	55.799	73.887
P2	31.746	39.106	22.899	55.330
P3	1.795	29.981	44.909	54.026
P4	29.125	19.281	4.508	35.218
p5	31.802	21.570	60.017	71.264

- Con tres cámaras.

Punto	Error en X	Error en Y	Error en Z	Error total (mm)
P1	32.632	35.810	87.121	99.685
P2	62.446	42.806	76.199	107.415
P3	34.602	50.830	30.817	68.779
P4	45.125	32.386	45.508	71.805
p5	38.402	20.370	91.817	101.587

- Con dos cámaras en el mismo lateral de la habitación (AB) de la figura 5.22.

Punto	Error en X	Error en Y	Error en Z	Error total (mm)
P1	39.595	74.619	44.291	95.380
P2	106.302	45.530	72.383	136.427
P3	61.102	56.630	114.817	141.856
P4	93.792	23.614	104.159	142.139
p5	174.802	56.030	106.317	212.128

- Con dos cámaras en distinto lateral de la habitación (AD) de la figura 5.22.

Punto	Error en X	Error en Y	Error en Z	Error total (mm)
P1	96.759	41.516	96.529	142.841
P2	17.695	158.581	182.709	242.577
P3	35.869	56.891	104.721	124.457
P4	167.770	178.260	150.292	287.247
p5	30.002	92.130	206.417	228.026

Los errores medios cometidos en distancia euclídea y el rendimiento temporal asociado fueron:

Número de cámaras	Errores medios (mm)	Rendimiento temporal (ips)
4 cámaras	57.945	32.7
3 cámaras	89.854	33.2
2 cámaras (mismo lateral AB)	145.586	33.7
2 cámaras (distinto lateral AD)	205.029	33.7

De los resultados se extrae la conclusión de que con mayor número de cámaras mejora la estimación de la posición 3D de la persona y el rendimiento temporal no se ve afectado significativamente. Entre utilizar cuatro y tres cámaras la diferencia no es muy significativa. Sin embargo, al usar sólo dos sí es notable el empeoramiento de la estimación. Además, también se aprecia una significativa diferencia entre utilizar las dos cámaras del mismo lateral de la habitación y utilizar una de cada lateral.

5.3.4. Pruebas sobre el aprendizaje de color

Uno de los objetivos de este proyecto era robustecer el aprendizaje automático de color de la vestimenta de las personas, eliminando la carencia en el aprendizaje de colores claros y oscuros de [Marugán, 2007].

En ese proyecto se guardaba información en un único histograma unidimensional sobre el canal H (tonalidad) mientras que los valores de S (saturación) y V (iluminación) eran fijos. La mejora consiste en que ahora se toman muestras de los tres canales y el color característico se define con los valores HSV que aperecen con mayor frecuencia en el histograma. Como ya se detalló en la sección 4.3, en este proyecto se han utilizado histogramas tridimensionales en el modelo HSV de color para almacenar el color característico de cada persona (la librería *Opencv* facilita el manejo de estos histogramas). Esto permite aprender cualquier tipo de color. Además, otro aspecto muy importante es que se almacena un histograma distinto por cada cámara, ya que el color en cada cámara puede variar significativamente. En las imágenes siguientes vemos varios ejemplos de los experimentos que validan el correcto funcionamiento del aprendizaje de color con personas cuyo seguimiento con el sistema anterior fallaba.

Figura 5.23: Ejemplos de colores aprendidos.

La actualización del color aprendido también es un aspecto clave. Una persona en movimiento sufre cambios de iluminación constantemente. El algoritmo hace frente a este problema actualizando los histogramas cada cierto tiempo. En la figura 5.24 vemos buenos ejemplos de la redefinición del color aprendido.

Figura 5.24: Redefinición del color aprendido.

5.3.5. Experimentos con la función salud

Para evaluar la salud de cada hipótesis se han programado dos alternativas distintas. Como ya se explicó en la sección 5.2.3, para calcular la salud de un prisma primero se proyecta en las imágenes y se halla la mínima envolvente rectangular de cada proyección. De este modo ya se puede operar con el prisma sobre la imagen.

Figura 5.25: Experimento con la función salud - (a) resultados con la primera alternativa probada, (b) resultados con el ajuste final de la salud.

En un principio la salud se definió exclusivamente en términos de densidad de píxeles que superan el filtro de color y el de movimiento dentro de la envolvente del prisma proyectado en cada cámara. Estos dos criterios son suficientes para mantener un buen seguimiento 3D de una persona. Sin embargo, la estimación del tamaño no es

tan buena debido a que estos criterios de densidad favorecen que los individuos tiendan a ser pequeños (ver figura 5.25 (a)), lo que facilita que su densidad de píxeles buenos sea mayor y así obtengan mayor salud.

Para solucionar este problema, pensamos en otros criterios que permitieran tener en cuenta el entorno del individuo 3D, no sólo su interior. De este modo se nos ocurrió *engordar un 30 % el prisma original* que se va a evaluar y tras proyectarlo, hallar la diferencia de áreas de las envolventes del prisma original y el engordado para obtener la densidad de píxeles sobre ella. Esta densidad debe tender a cero, puesto que son píxeles que pertenecen al entorno de la persona.

La inclusión de los factores de densidad de la diferencia hacen que mejore significativamente la estimación del tamaño de las personas, ya que la salud será máxima cuando el prisma original contenga a la persona y el engordado contenga además parte del entorno de la misma.

En otro experimento se quiso analizar la discriminación de la función salud en términos de posición y de tamaño. Partiendo del prisma representante, que es el resultado del seguimiento instantáneo de una persona, se realizó una búsqueda local en primer lugar fijando el tamaño del prisma y modificando su posición XY y en segundo lugar fijando la posición y cambiando el tamaño.

En las siguientes imágenes podemos ver el resultado de la búsqueda local en posición XY (en verde) en tres momentos distintos del seguimiento realizado por el sistema (azul).

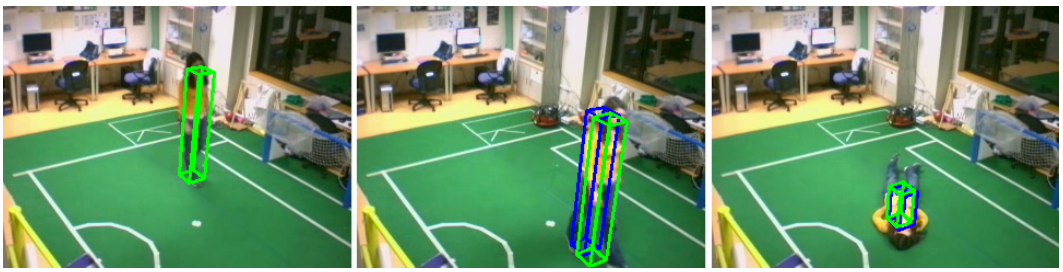

Figura 5.26: Experimento con la función salud - búsqueda local en posición (verde).

En estas imágenes tenemos el resultado de la búsqueda local en tamaño (verde).

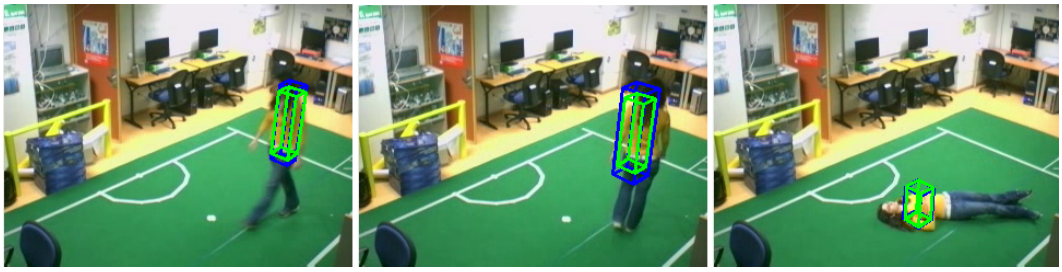

Figura 5.27: Experimento con la función salud - búsqueda local en tamaño.

El resultado de las búsquedas locales demuestran que la función salud es discriminante y que el resultado de la búsqueda del algoritmo evolutivo es muy similar al de la búsqueda local, que mucho más costosa computacionalmente si la extendiéramos a las cinco dimensiones (X,Y,Z,h,s) del espacio de búsqueda.

5.3.6. Experimento para la detección de una caída

Este experimento era necesario para validar la aplicación del sistema a la detección de caídas de personas. Como vemos en la imagen, cuando una persona se encuentra tirada en el suelo se activa una alarma visual en la interfaz de la aplicación, por lo que el sistema sirve perfectamente como detector de caídas.. El umbral definido en la coordenada Z para considerar que una persona está en el suelo es 30 centímetros.

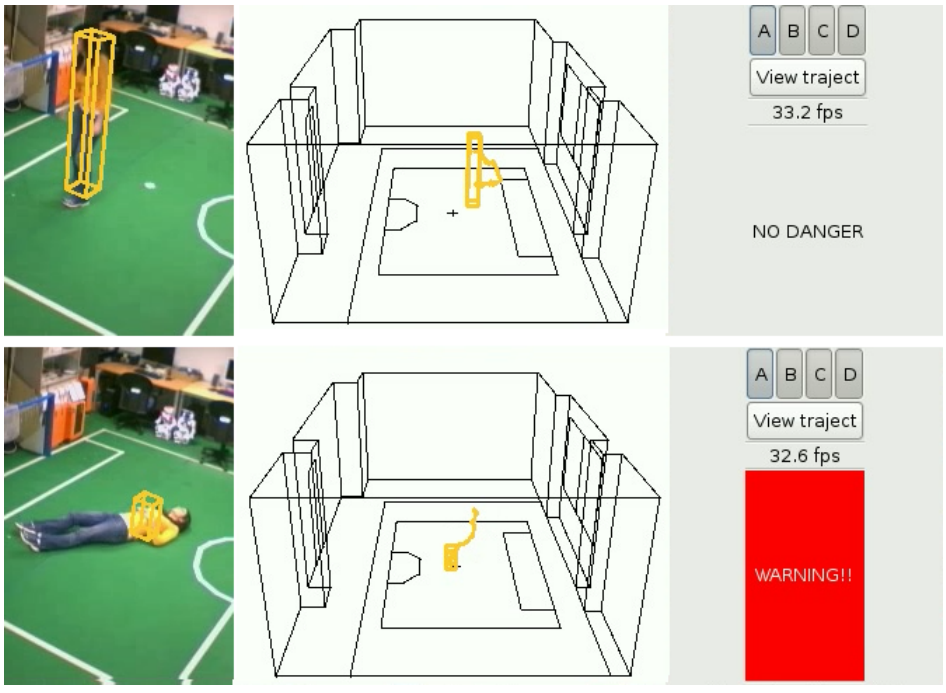

Figura 5.28: Experimento para la detección de una caída.

Capítulo 6
Conclusiones y trabajos futuros

En los capítulos anteriores se han detallado los métodos utilizados para resolver en primer lugar el seguimiento 2D de personas y posteriormente el seguimiento 3D. Además se expusieron los experimentos realizados más relevantes y la comparación entre dos técnicas diferentes de seguimiento 3D. En este capítulo se resumen las conclusiones obtenidas y las posibles vías de continuación para la investigación.

6.1. Conclusiones

El objetivo principal de este proyecto era construir una aplicación capaz de mantener un seguimiento 3D y estimar el tamaño de todas las personas dentro de una habitación utilizando cuatro cámaras.

En primer lugar es necesario destacar que *los tres subobjetivos en los que se divide este proyecto se han alcanzado con éxito*. El primero consistía en generar un seguimiento 2D de personas sobre el flujo de imágenes de una cámara. El segundo, en diseñar e implementar dos alternativas para el seguimiento 3D de varias personas y seleccionar la más eficiente. El último se trataba de realizar experimentos más exhaustivos para ajustar y caracterizar mejor el algoritmo de seguimiento 3D elegido.

En el capítulo 4 se detalló la solución final escogida para la aplicación de seguimiento 2D. Ésta consistió en un algoritmo evolutivo multimodal basado en [Marugán, 2007] pero adaptado al problema del seguimiento 2D.
En primer lugar se diseñó un esquema segmentador de movimiento, del que se obtiene como resultado una lista de regiones de interés (ROIs) rectangulares corresponditentes con las zonas en las que detecta movimiento. El algoritmo evolutivo 2D utiliza esas ROIs instantáneas para generar las razas, ya que asume que las personas se encuentran en movimiento al aparecer en la escena. Las razas son las que mantienen el seguimiento

de las personas y aprenden de manera automática su color. Estas razas son una pequeña población de individuos rectangulares. Cada uno es evaluado a través de una función salud que tiene en cuenta cuatro tipos de información: movimiento, color, puntos significativos y continuidad espacial en la imagen. Realimentados con la función salud, los individuos de la raza convergen a la solución óptima de la posición 2D, color y tamaño 2D de la persona. De este modo, cada raza sigue a una persona y mantiene su seguimiento 2D mientras permanece visible en las imágenes. Para validar el correcto funcionamiento del algoritmo se realizaron experimentos respecto a la discriminación de la función salud y se ejecutó el sistema sobre diferentes secuencias de vídeo. La velocidad de la aplicación alcanza aproximadamente 30 ips con imágenes a una resolución de 640x480.

En cuanto al segundo subobjetivo, que consiste en el seguimiento 3D, en el capítulo 5 se exponen las dos alternativas implementadas y una comparativa que justifica la elección de una de ellas para constituir el sistema final.

La primera alternativa consistió en un esquema que utiliza la información del seguimiento 2D de varias cámaras y sobre ella genera el seguimiento 3D. Para ello obtiene de cada seguidor 2D la lista de personas y genera hipótesis 3D a lo largo de las líneas abductivas de cada persona detectada por un seguidor 2D. Las hipótesis 3D son prismas que son evaluados midiendo el parecido entre sus proyecciones en cada cámara y las personas del seguidor 2D de la cámara correspondiente. De esta manera, las hipótesis con mayor parecido en todas las cámaras se toman como la representación de las personas de la escena en el seguimiento 3D.

La segunda alternativa fue un algoritmo evolutivo multimodal para el seguimiento 3D. Este algoritmo también genera hipótesis 3D de manera abductiva y, tras evaluarlas, en las posiciones 3D de las hipótesis ganadoras sitúa razas, que se encargan de explorar localmente dichas zonas de interés en las que se encuentran las personas. Los individuos de las razas son también prismas que varían su posición 3D y tamaño a lo largo del tiempo para converger a la solución óptima. La función salud de este algoritmo se apoya en cuatro factores: densidad de píxeles de color y de píxeles de movimiento respecto a la mínima envolvente rectangular del prisma proyectado y las densidades en el área de la diferencia entre las envolventes del prisma original y el engordado. Estos dos últimos criterios sirven para que la salud no sólo evalúe el contenido del prisma sino también el entorno de éste, lo que permite ajustar mejor el tamaño estimado de la persona.

Una vez implementadas ambas técnicas, se realizó una comparativa en los ámbitos de rendimiento, error en la estimación y calidad del seguimiento.
En cuanto al rendimiento temporal, se comprobó que la primera técnica es más rápida que la segunda, concretamente la primera alcanza 30 ips y la segunda 25 ips. Los errores de estimación en la posición 3D de ambas técnicas son de orden centimétrico, sin embargo, el algoritmo evolutivo consigue un menor error, aproximadamente de 5 centímetros. La calidad del seguimiento se midió en función del comportamiento coherente de aparición, seguimiento y desaparición de las razas según entren y salgan las personas en la habitación. Respecto a este criterio, se notó que con la primera técnica se producen más errores a la hora de asignar un prisma a cada persona, sobre todo al aumentar el número de personas en la escena. Esto en parte es debido a que el seguimiento con una cámara posee ciertas limitaciones en los cruces de personas y las oclusiones respecto al seguimiento utilizando varias cámaras. El seguimiento 2D diseñado en este proyecto no llega a ser tan robusto como para generar sobre él un seguimiento 3D que funcione correctamente en situaciones complejas.
Las conclusión que se extrajo de los experimentos es que *trackerEvolutive3D* a pesar de dar un menor rendimiento para varias personas, el seguimiento que proporciona es mucho más robusto y la estimación es mejor. Por estos motivos, para el sistema final se seleccionó la técnica del algoritmo evolutivo para el seguimiento 3D de personas.

Evaluar la aplicación final era la finalidad del tercer subobjetivo. A través de los experimentos y las pruebas descritas al final del capítulo 5, se ajustó el algoritmo de seguimiento 3D elegido con el fin de que el sistema cumpliera de la manera más fiel posible los objetivos propuestos. Se comprobó que el funcionamiento de la aplicación final es satisfactorio. Además, se observó que el uso de primitivas volumétricas mejora la estimación de la localización 3D de las personas respecto al uso de primitivas puntuales [Marugán, 2007].

En cuanto a los requisitos, como se vio en los experimentos del capítulo 5, *la aplicación cumple todos los planteados* en el capítulo 2. Utiliza únicamente *hardware convencional*, concretamente cámaras de videoconferencia y ordenadores personales. Funciona correctamente en una *habitación de gran volumen*, que como se ha visto en los experimentos es el Laboratorio de Robótica del Edificio Departamental II de esta universidad. Por último, es capaz de *seguir a varias personas andando* estimando su color y tamaño de forma *vivaz y precisa*.

Para terminar, conviene recordar que *este proyecto fin de carrera se enmarca dentro del proyecto Eldercare*, un sistema para la asistencia y ciudado de mayores en sus propios hogares. La primera generación era capaz de seguir a una única persona y con el color supervisado [Pineda, 2006]. En la segunda generación se pasó al seguimiento de múltiples personas mediante primitivas puntuales y de las cuales se aprendía el color automáticamente [Marugán, 2007]. Este proyecto *constituye la tercera generación*, que pasa de considerar a las personas como individuos puntuales a caracterizarlas mediante una primitiva con volumen, mejorando la localización 3D. Además, se ha mejorado el aprendizaje de color, eliminando la carencia para colores claros y oscuros y como consecuencia haciéndolo más robusto. Por último, destacar que se ha mejorado el rendimiento del sistema dividiendo el algoritmo en distintas hebras de ejecución para aprovechar la capacidad multinúcleo de los ordenadores y utilizando la biblioteca *Opencv* para el procesamiento de imágenes.

6.2. Trabajos futuros

Este proyecto fin de carrera, como todo proyecto de ingeniería, presenta algunas limitaciones. La más relevante es que la estimación del tamaño de las personas, sin imponer la restricción de que las personas se encuentran apoyadas en el suelo, en la mayoría de los casos se ajusta al torso de las personas y no al cuerpo completo. Esto es debido a que el filtro de color es monomodal, es decir, aprende un sólo color que normalmente es el de la camiseta. La dificultad de aprender el color de los pantalones es que al andar una persona, el color del fondo que deja el hueco entre las piernas se mezcla con el color de los pantalones, generando un filtro de color malo. Utilizando otro tipo de criterios que no fuera el color quizá esta carencia podría resolverse.

Una vía por donde se puede extender el sistema podría consistir en distribuir aún más el procesamiento, de modo que cada equipo que tiene conectada una cámara podría recibir la lista de personas seguidas en 3D y devolver una realimentación desde la información 2D de las imágenes de esa cámara. Así, el cálculo de salud por cada cámara se haría simultáneamente y la aplicación en el equipo principal fusionaría la información. Esto facilitaría la introducción de un mayor número de cámaras y ampliar la zona vigilada.

Otra línea interesante sería caracterizar mejor las personas. Ahora mismo se las considera como un bloque rígido, pero se podría tener en cuenta la cabeza, el torso y las extremidades por separado. Así, podría abrirse el campo del análisis de la postura de las personas.

En cuanto a la aplicación de cuidado de mayores, el sistema se podría probar en entornos reales como una casa o una residencia de ancianos. Con ello se valoraría mejor la robustez del sistema.

Bibliografía

[Allen *et al.*, 1991] P. K. Allen, B. Yoshimi, y A. Timcenko. Real-time visual servoing. *Proc. IEEE Int. Conf. on Robotics and Automation*, 1991.

[Barrera *et al.*, 2005] Pablo Barrera, José María Cañas, y Vicente Matellán. Visual object tracking in 3d with color based particle filter. *Int. Journal of Information Technology*, 2005.

[Barrera y Cañas, 2004] Pablo Barrera y José María Cañas. Seguimiento tridimensional usando dos cámaras. 2004.

[Bay *et al.*, 2008] Herbert Bay, Andreas Ess, Tinne Tuytelaars, y Luc Van Gool. Speeded up robust features. *Computer Vision and Image Understanding (CVIU)*, 2008.

[Bishop y Welch, 2005] Gary Bishop y Greg Welch. An introduction to the Kalman filter. page 16. 2005.

[Blake y Isard, 1994] A. Blake y M. Isard. 3d position, attitude and shape input using video tracking of hands and lips. *Proc. ACM Siggraph*, 1994.

[Cañas *et al.*, 2008] José María Cañas, Sara Marugán, Carlos Agüero, y Teodoro González. Detección visual de caídas para ambientes inteligentes. *Proceedings of RoboCity2030 4th Workshop, Robots personales y asistenciales*, 2008.

[Cañas *et al.*, 2009] José María Cañas, Sara Marugán, Marta Marrón, y Juan C. García. Visual fall detection for intelligent spaces. *Proceedings of the 6th IEEE International Symposium on Intelligent Signal Processing*, 2009.

[Cornelis, 2004] K. Cornelis. From uncalibrated video to augmented reaelity. *From uncalibrated video to augmented reaelity*, 2004.

[D. Margaritis, 1998] S. Thrun D. Margaritis. Learning to locate an object in 3d space from a sequence of images. 1998.

[Darrell *et al.*, 1998] T. Darrell, G. Gordon, J. Woodfill, y M. Harville. A virtual mirror interface using real-time robust face tracking. *Proceedings of the 3rd International Conference on Face and Gesture Recognition*, 1998.

[DeMenthon y Davis, 1995] D. DeMenthon y L. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 1995.

[Fox *et al.*, 1999] Dieter Fox, Wolfram Burgard, Frank Dellaert, y Sebastian Thrun. Monte Carlo localization: efficient position estimation for mobile robots. In *Proceedings of the 16th AAAI National Conference on Artificial Intelligence*, pages 343–349, Orlando (Florida, USA), July 1999.

[González, 2007] Pablo Barrera González. *Aplicación de los métodos secuenciales de Monte Carlo al seguimiento visual 3D de múltiples objetos.* PhD thesis, 2007.

[J.Davison *et al.*, 2007] Andrew J.Davison, Ian D. Reid, Nicholas D. Molton, y Olivier Stasse. Monoslam: Real-time single camera slam. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007.

[Kalman, 1960] Rudolf E. Kalman. A new approach to linear filtering and prediction problems. 1960.

[Kass *et al.*, 1988] M. Kass, A. Witkin, y D. Terzopoulos. Snakes: Active contor models. *Int. Journal on Computer Vision*, 1988.

[Lowe, 1999] David G. Lowe. Object recognition from local scale-invariant features. 1999.

[Lucas y Kanade, 1981] Lucas y Kanade. An iterative image registration technique with an application to stereo vision. 1981.

[López, 2010] Luis Miguel López. Autolocalización en tiempo real mediante seguiminto visual monocular. *Proyecto Fin de Carrera, URJC*, 2010.

[Mackay, 1999] D.J.C. Mackay. Introduction to Monte Carlo methods. In M. Jordan, editor, *Learning in Graphical Models*, pages 175–204. MIT Press, Cambridge (MA, USA), 1999.

[Martínez, 2007] Iván García Martínez. Reconstrucción 3d visual con algoritmos evolutivos. *Proyecto Fin de Carrera, URJC*, 2007.

[Marugán, 2007] Sara Marugán. Seguimiento 3d visual de múltiples personas utilizando un algoritmo evolutivo multimodal. *Proyecto Fin de Carrera, URJC*, 2007.

[Maybeck, 1979] Peter S. Maybeck. *Stochastic models, estimation, and control*, volume 141 of *Mathematics in Science and Engineering*. 1979.

[Pineda, 2006] Antonio Pineda. Aplicación de seguridad basada en visión. *Proyecto Fin de Carrera, URJC*, 2006.

[Plaza, 2003] José María Cañas Plaza. *Jerarquía Dinámica de Esquemas para la generación de comportamiento autónomo*. PhD thesis, Universidad Politécnica de Madrid, 2003.

[Plaza, 2004] José María Cañas Plaza. Manual de programación de robots con jde. *URJC*, 2004.

[Shi y Tomasi, 1994] Jianbo Shi y Carlo Tomasi. Good features to track. 1994.

[Stauffer y Grimson, 2000] C. Stauffer y W.E.L Grimson. Learning patterns of activity using real-time tracking. *Transactions on Pattern Analysis and Machine Intelligence, vol. 22*, 2000.

[Ángel Cortés Maya, 2007] Ángel Cortés Maya. Localización y construcción de mapas en un robot de interiores. *Proyecto Fin de Carrera, URJC*, 2007.