Universidad Zaragoza
1542

Trabajo Fin de Máster

Monitorización de pacientes utilizando un sistema multi-cámara

Patient monitoring with multi-camera systems

Autor

Rosa Castillón Lacasa

Directores

Ana Cristina Murillo Arnal
Eduardo Montijano Muñoz

Escuela de Ingeniería y Arquitectura
2017

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

D./Dª. ROSA CASTILLÓN LACASA,

con nº de DNI 73205230 W en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) MÁSTER EN INGENIERÍA BIOMÉDICA, (Título del Trabajo) MONITORIZACIÓN DE PACIENTES UTILIZANDO UN SISTEMA MULTI-CÁMARA,

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 1 de Febrero de 2017

Fdo: ROSA CASTILLÓN LACASA

RESUMEN

MONITORIZACIÓN DE PACIENTES UTILIZANDO UN SISTEMA MULTI-CÁMARA

Cada vez son más las personas mayores que necesitan cuidados por parte de familiares o cuidadores. Por eso, es necesario el desarrollo de sistemas que ayuden a su cuidado, proporcionen seguridad y detecten situaciones de emergencia.

En este TFM se ha desarrollado un prototipo para un sistema de monitorización de pacientes y detección de eventos críticos mediante un sistema de visión. El objetivo es conseguir un sistema de bajo coste, externo, que libere al paciente de tener que llevar sensores puestos todo el tiempo, sin perder fiabilidad. El prototipo está pensado para entornos de interior y detecta posibles caídas del paciente monitorizado y el acceso a zonas peligrosas de la estancia.

El sistema de visión está compuesto por dos cámaras fijas que capturan pares de imágenes RGB y de profundidad. Los datos utilizados provienen de un dataset público, que contiene secuencias simples de imágenes en un entorno de interior, y de un dataset propio, grabado en un entorno más realista. En ambos se ha evaluado el sistema completo.

El sistema detecta automáticamente al paciente que se desea monitorizar y realiza el seguimiento del mismo a lo largo de la secuencia de imágenes. La detección de caídas se ha conseguido mediante el diseño de un sistema de clasificación de imágenes. Para implementar el control de acceso a zonas peligrosas, gracias al uso de la información 3D de los sensores, se ha calculado la trayectoria del paciente, lo que permite conocer su posición y determinar si ha sobrepasado los límites establecidos. Se ha diseñado un sistema de alarma que genera alertas visuales, en los casos en los que se produce alguno de estos eventos.

La evaluación del prototipo muestra que éste permite reconocer correctamente las caídas del paciente, así como determinar con precisión, cuándo el paciente sobrepasa los límites de la estancia. Además de validar el funcionamineto de la aplicación diseñada, la experimentación realizada confirma las ventajas de poder utilizar varias cámaras y el uso de vídeos para obtener resultados más robustos.

Índice general

1. Introducción

La población, tanto en España como en el mundo en general, está envejeciendo. Pronto serán mucho más numerosas las personas mayores que los niños (Figura 1.1). En 2010, 524 millones de personas eran mayores de 65 años, pero se prevé que este número aumente hasta el triple, y en el 2050 sean 1.5 billones [9].

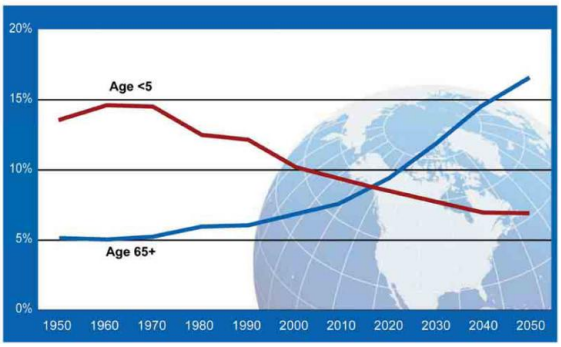


Figura 1.1: Evolución de los porcentajes globales de niños y personas mayores desde 1950 hasta la previsión del año 2050.

Una de las preguntas que se plantean actualmente, debido al aumento en la esperanza de vida, es si este aumento en la longevidad de las personas irá asociado a un incremento de las enfermedades, la dependencia y la discapacidad. Si esto ocurre, cada vez más personas van a necesitar atenciones y cuidados por parte de familiares, cuidadores o personal sanitario. Por eso, el avance de la tecnología debería servir para mejorar la calidad de vida de todos ellos, por ejemplo, desarrollando sistemas que ayuden al cuidado de estas personas, proporcionen seguridad y detecten situaciones de emergencia.

En este TFM se plantea el desarrollo de un prototipo para un sistema de monitorización de pacientes que permita detectar eventos críticos, utilizando un sistema visual. Debido al enfoque de la aplicación, se detectarán como eventos críticos las caídas de los pacientes y el acceso a zonas restringidas de una estancia. Las caídas se consideran un evento crítico, ya que entre el 28-35 % de las personas mayores de 65 años sufren entre 2 y 4 cada año [10], que pueden provocar lesiones, dolor crónico, discapacidad, o incluso desembocar en una

falta de independencia. Por otro lado, saber que el paciente accede a una zona restringida o peligrosa puede ser útil en casos de desorientación de las personas mayores, etc.

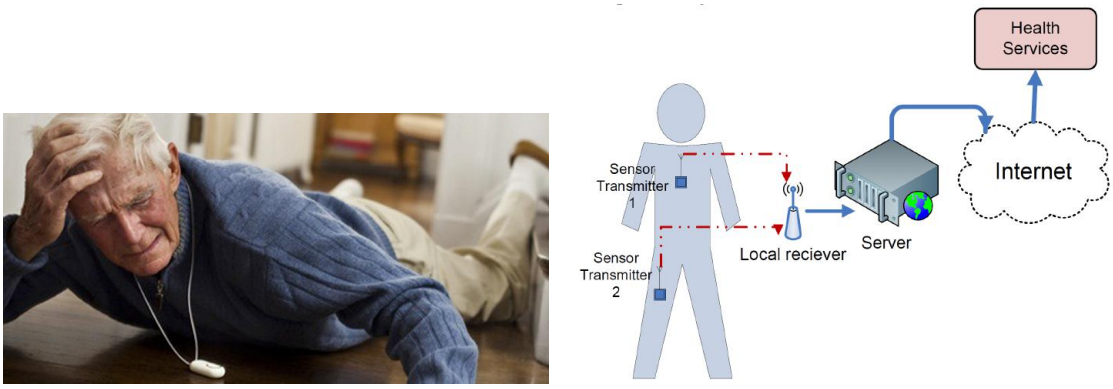
1.1. Trabajos relacionados

Además de los sistemas visuales, existen otras técnicas que se utilizan en los sistemas de monitorización con detección de eventos y seguimiento de pacientes. A continuación se comentan algunas de estas técnicas, haciendo referencia a trabajos relacionados con las mismas.

1.1.1. Sistemas de detección de caídas

Los sistemas de detección de caídas son de gran utilidad ya que permiten proporcionar una respuesta médica rápida y adecuada. Estos sistemas se pueden clasificar en diferentes grupos:

Sistemas de alarma activados por el usuario. En este tipo de sistemas, el usuario debe activar una alarma en caso de caída. Este sistema tiene la limitación de que, dependiendo de la intensidad de la caída, es posible que el paciente quede inconsciente o que, físicamente, no sea capaz de activar la alarma (Figura 1.2(a)).



(a) (b)

Figura 1.2: Dos ejemplos de los sistemas de detección, el primero activando una alarma por el paciente (a) y el segundo utilizando sensores (b)

Detección de caídas basado en la vibración del suelo. Estos sistemas utilizan sensores piezoeléctricos para medir la vibración del suelo [1]. El sistema requiere la insta-

lación de un aparato en la estancia compuesto por este tipo de sensores y un sistema de baterías.

Sensores portátiles(acelerómetros, giróscopos...). Estos son los sistemas más comunes para detectar caídas. Los acelerómetros son sensores que miden la aceleración y sirven para medir movimiento, vibración y velocidad. Utilizando algoritmos que miden las variaciones de aceleración en todas las direcciones, se pueden detectar caídas con una alta precisión [6], [8]. Sin embargo, en estas aplicaciones el paciente debe portar el sensor todo el tiempo y ésto puede resultar molesto. También puede ocurrir que el paciente no quiera o se olvide de llevarlo. En la Figura 1.2(b) se observa un ejemplo relativo al lugar en el que se pueden colocar estos sensores y cómo se realiza el proceso de transmisión de la información.

Sistemas de detección visual. Estos sistemas realizan el seguimiento del paciente utilizando cámaras fijas instaladas en la estancia y detectan los eventos mediante algoritmos de procesamiento de imágenes. Existen también sistemas que combinan la detección visual con los sensores portátiles [5]. Se utilizan cámaras Kinect [1], que proporcionan imágenes RGB e imágenes de profundidad, y acelerómetros. En [5] la información visual se extrae únicamente de las imágenes de profundidad y a partir de ellas se entrena un algoritmo SVM que permite clasificar las imágenes dependiendo de si el individuo está en una posición normal o si, por el contrario, se ha caído. Los datos que aporta el acelerómetro se combinan con los resultados obtenidos para detectar las caídas con una mayor fiabilidad. En la Figura 1.3 se muestran ejemplos de monitorización de pacientes utilizando sistemas de visión.

1.1.2. Seguimiento y localización de pacientes

Otra tarea importante en la monitorización de pacientes con cierto riesgo o dependencia consiste en realizar el seguimiento de la posición del paciente, y activar señales de alarma en el caso de que estas personas atraviesen ciertos límites establecidos, salgan de la zona vigilada, etc. De esta forma, se evitan situaciones peligrosas debidas a la desorientación, etc.

En este campo existen algunos sistemas que proporcionan la posición de las personas y permiten localizarlas en caso de que se desorienten o se pierdan. En [7] utilizan la

[1] Es un periférico de entrada desarrollado por Microsoft para jugar en la videoconsola Xbox360. Dispone de dos cámaras frontales, una convencional RGB y otra que devuelve la profundidad.

(a) (b)

Figura 1.3: Sistemas de detección visual instalados en diferentes estancias. Estos sistemas se pueden utilizar tato en entornos domésticos (a) como en entornos hospitalarios (b).

tecnología GPS para calcular la posición geográfica con bastante precisión. Sin embargo, este sistema no es útil en el interior de edificios o en zonas con mala cobertura.

La localización de pacientes también se utiliza en los hospitales, para los casos en que los pacientes se mueven entre los distintos departamentos mientras se les realiza el diagnóstico y se les aplica el tratamiento. En [12] utilizan un sistema de identificación por radiofrecuencia, que permite almacenar y recuperar datos de forma remota. Por ejemplo cada vez que un paciente atraviesa una entrada o salida, es trasladado de área, o se le aplica un nuevo tratamiento, queda registrado y se le puede identificar recuperando toda esta información.

1.2. Objetivos y alcance del proyecto

El objetivo principal de este proyecto consiste en implementar un prototipo para la monitorización de pacientes a través de distintas cámaras. Se pretende que el sistema pueda detectar automáticamente el paciente a monitorizar, seguir su posición por la zona monitorizada y detectar determinados eventos críticos, tales como posibles caídas. La aplicación permitirá delimitar zonas de la estancia, de forma que si el paciente las sobrepasa, se active una señal de alarma. Esta aplicación puede ser útil tanto para entornos domésticos como para entornos hospitalarios. Para evaluar el prototipo desarrollado se van a utilizar secuencias de un dataset público, disponible para investigación, así como un dataset propio adquirido con varias cámaras RGB-d[2].

[2]Las cámaras RGB-d son dispositivos compuestos por una cámara RGB y un sensor de profundidad que proporcionan para cada píxel un valor RGB y otro de profundidad. Se describen con más detalle en el Anexo C

Las tareas concretas a realizar en este proyecto son (la distribución de tareas de este TFM se puede observar en el Anexo D.):

- Estudiar las técnicas existentes de detección de personas y caras y detección de movimiento en secuencias.
- Proponer técnicas de detección y descriptores asociados a los elementos detectados en distintas vistas de un sistema multicamara con sensores RGB-d.
- Diseñar un sistema de monitorización multi-cámara que incluya: detección/elección de persona a monitorizar, seguimiento, generación de alertas configurables para la detección de acciones (caídas) y de posición del paciente (salir de zona permitida).
- Evaluación del sistema con base de datos pública con ground truth para comparar la precisión de los algoritmos utilizados y estudiar el impacto del uso de multiples cámaras/vistas.
- Configurar un sistema multi-cámara y adquirir datos propios para realizar experimentos de integración y validación en un entorno más realista.

Tal y como se describe a lo largo de la memoria, todas estas tareas se han realizado satisfactoriamente. En lo relativo a las tareas relacionadas con algoritmos de detección de personas y movimiento, este trabajo se ha centrado en el estudio de métodos existentes, y su integración y configuración óptima en el sistema diseñado.

Por otro lado, las tareas relacionadas con la clasificación y detección de eventos se han realizado utilizando las herramientas de la librería *Scikit-learn*[11] para entrenar y configurar clasificadores. Éstos se han construido utilizando los descriptores diseñados y propuestos en este trabajo.

Por último, los datos que se han utilizado pertenecen a *UR Fall Detection* dataset[5], un dataset público de varias secuencias de interior. Adicionalmente, se han grabado secuencias propias en un entorno de interior más realista utilizando varias cámaras fijas.

1.3. Estructura de la memoria

El resto de capítulos de esta memoria están distribuidos de la siguiente forma: en el capítulo 2 se hace una visión de conjunto del algoritmo general de la aplicación. En el capítulo 3 se desarrollan las técnicas utilizadas para la detección del paciente. En el capítulo 4 se describe la detección de eventos críticos que el sistema es capaz de detectar.

En el capítulo 5 se exponen los resultados obtenidos durante la realización de los experimentos, tanto en la evaluación de las técnicas de detección, como en el entrenamiento de los clasificadores y la implementación de los descriptores. Por último, las conclusiones obtenidas en este TFM están en el capítulo 6.

2. Sistema de monitorización de pacientes y detección de eventos críticos

Este proyecto se centra en implementar un sistema de monitorización de pacientes, que sea capaz de hacer el seguimiento de una persona en el interior de una estancia y a su vez detectar determinados eventos críticos, en particular posibles caídas o acceso a zonas restringidas. La vista general de la aplicación se puede observar en la Figura 2.1.

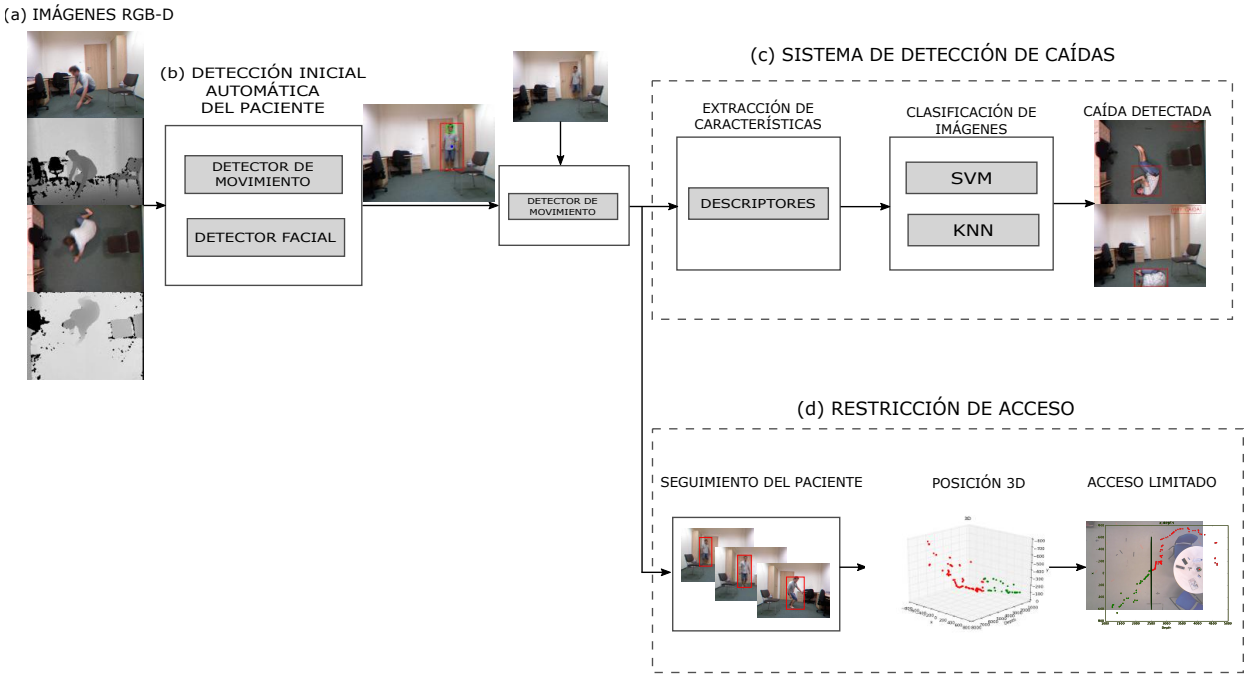


Figura 2.1: Diagrama general de la aplicación. Los datos de entrada proceden de dos cámaras RGB-d(a); estos datos se procesan para detectar y seguir la posición del paciente (b); se detectan de manera automática eventos críticos; se detectan caídas del paciente automáticamente (c); se detecta cuando el paciente sale de una determinada zona de la estancia (d).

El sistema de visión está formado por dos cámaras fijas a partir de las cuales se obtienen pares de imágenes RGB y *depth* de las que se extrae la información. En la Figura 2.2 se

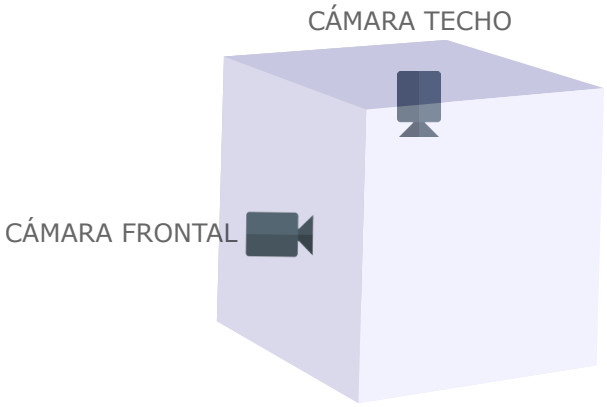


Figura 2.2: Posición de las cámaras fijas en la escena.

observa la configuración de las cámaras. Una de ellas capta las secuencias de imágenes desde una vista frontal de la estancia, mientras que la otra está colocada en el techo y capta las imágenes desde esa perspectiva.

Detección del paciente (ROI, región de interés) en la imagen. La detección del paciente se realiza mediante la combinación de distintas técnicas de detección de movimiento y de objetos (caras y personas). Como se explica más adelante, en esta memoria (Sección 3), se han evaluado diferentes técnicas de detección, aplicando finalmente, la detección de movimiento y la detección facial. La Figura 2.1(b) muestra cómo, a partir de las imágenes y aplicando los distintos detectores, se obtiene la detección del paciente.

Procesado de zona de interés de la imagen. Una vez detectado el paciente que se desea monitorizar, se realizan dos tareas. La primera de ellas es la extracción de características. Con ellas se calculan una serie de descriptores de las imágenes que servirán posteriormente en la clasificación de imágenes. La segunda tarea consiste en realizar el seguimiento del paciente por las imágenes de la secuencia, conociendo así, su posición durante todo el tiempo.

A partir de los descriptores de las imágenes y el seguimiento del paciente, se configura la detección de eventos críticos. En nuestro caso, detección de caídas y restricción de acceso a determinadas zonas.

Sistema de detección de caídas. La detección de caídas (Figura 2.1(c)) se plantea como un problema de clasificación. La extracción de características de las imágenes y los detectores da lugar a un vector de descriptores de las imágenes. Estos vectores de descriptores sirven para entrenar un algoritmo que clasifica nuevas imágenes dependiendo

del estado del paciente. El sistema en concreto clasifica las imágenes entre “Caída” y “Normal” (cuando el paciente está en alguna posición común.)

Restricción de acceso. Por otro lado, el seguimiento del paciente nos permite conocer su posición en todo momento. Las imágenes y la información extraída de los detectores, junto con los parámetros de calibración de las cámaras, nos permiten conocer la posición en 3D del paciente. Se establecen límites en la estancia y se configura un sistema de alarmas que se activa cuando el paciente sobrepasa dichas zonas. La aplicación muestra una alarma cuando se detecta un evento (Figura 2.1(d)).

Cada una de las partes del sistema se describe mas detalladamente en los siguientes capítulos de esta memoria. En el capítulo 3 se detallan las técnicas de detección utilizadas y cómo se realiza el seguimiento del paciente. En el capítulo 4 se describen los descriptores propuestos en este TFM y el sistema de detección de eventos críticos.

3. Detección inicial del paciente en las imágenes y seguimiento

El primer paso para la monitorización de un paciente es la detección del mismo en las imágenes capturadas en estancia donde se encuentra. Teniendo en cuenta que se trata de entornos de interior, el movimiento que se produce en las imágenes extraídas del vídeo corresponde, en general, a las personas de la escena. Por lo tanto, se implementa un algoritmo para detectar zonas de las imágenes/vídeo en las que hay elementos en movimiento. Para confirmar que el movimiento detectado corresponde a una persona, se han propuesto los detectores de caras y personas. A su vez, el detector de movimiento se utilizará para realizar el seguimiento del paciente monitorizado a lo largo de la grabación (*streaming*) de vídeo en tiempo real. A continuación se explican más detalladamente cada uno de estos algoritmos y componentes del sistema.

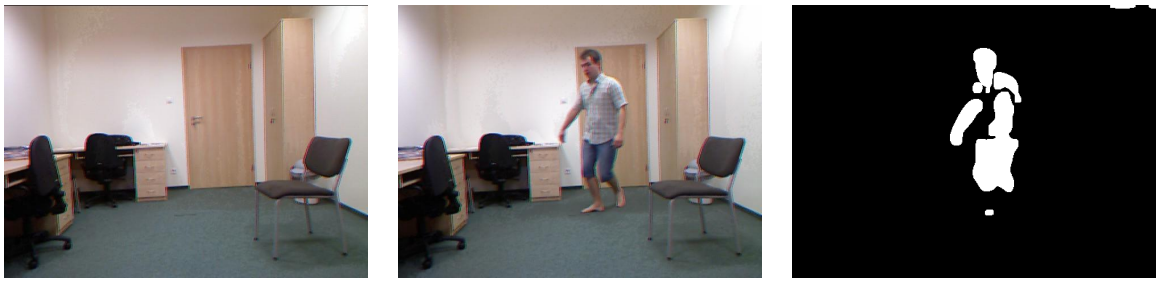
En este TFM se realiza la detección del paciente de una forma automática. Sin embargo, de una forma sencilla, se podría añadir una opción para la selección manual del paciente al que se quiere monitorizar en un interfaz visual.

3.1. Detección automática del paciente

La detección automática del paciente se realiza combinando el detector de movimiento y el detector de caras. El detector de movimiento devuelve un rectángulo que abarca los píxeles de los objetos en movimiento de la escena. Para comprobar que se trata de una persona se aplica el detector de cara, que también devuelve un rectángulo. Si se produce intersección entre ambos, se habrá detectado una persona. A continuación se explican con más detalle todos los detectores que se han evaluado en este TFM.

3.1.1. Detector de movimiento

El detector de movimiento está basado en una técnica ampliamente utilizada que se basa en la extracción del fondo de la imagen (*Background subtraction*)[4]. Esta técnica con-



(a) *Background frame* (b) Frame con el paciente. (c) Elementos en movimiento

Figura 3.1: Detección de movimiento con la técnica de extracción del fondo.

siste en obtener una imagen binaria (Figura 3.1(c)) que contiene los píxeles pertenecientes a los objetos en movimiento de la escena en un color, y los píxeles que corresponden a elementos estáticos en otro color. Se parte de una imagen del fondo (*background*), que contiene solo la parte estática de la habitación 3.1(a). Ésta se "resta" con cada imagen de la secuencia de monitorización 3.1(b) para conseguir el resultado 3.1(c). Por fin, esta imagen binaria muestra en blanco los píxeles correspondientes a elementos en movimiento.

3.1.2. Detector de caras

El detector de movimiento descrito en la sección anterior se ha combinado con distintos detectores de caras implementados en la librería *OpenCV* [1]. El método de detección de caras implementado en esta librería se corresponde con el trabajo denominado *Haar feature-based cascade classifier* [13], un método muy efectivo en cuanto a la detección de objetos. Se trata de un algoritmo de aprendizaje entrenado a partir de un gran número de imágenes positivas (imágenes de caras) y negativas (imágenes sin caras).

El clasificador se construye seleccionando pocas, pero efectivas, características, utilizando AdaBoost [3]. Dos de las posibles características seleccionadas por el algoritmo de AdaBoost aparecen en la Figura 3.2. Ambas características comparan intensidades entre distintas zonas de la imagen. La primera de ellas compara la intensidad más oscura de la zona de los ojos y la intensidad más clara de la zona de las mejillas. Lo mismo ocurre en la segunda, comparando, en este caso, la zona de los ojos con el espacio que los une.

El clasificador final es el resultado de un conjunto de clasificadores simples con características como las de la Figura 3.2. Estos clasificadores simples se organizan en cascada, permitiendo eliminar de forma rápida, y sin necesidad de un procesamiento costoso, las zonas de la imagen en las que no es probable encontrar una cara. Ésto permite realizar un

[1] www.opencv.org

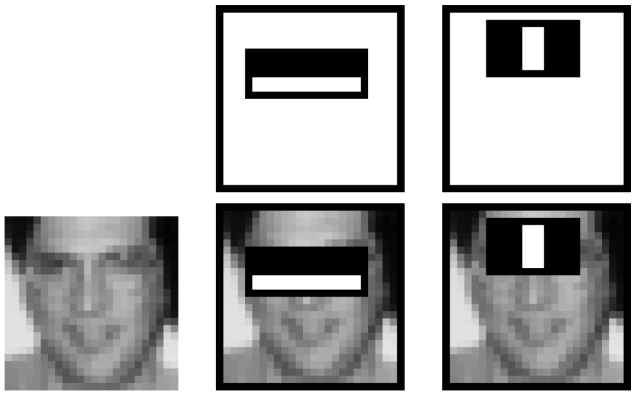


Figura 3.2: Características utilizadas por los clasificadores en los algoritmos de detección de caras. Imagen extraída de [13].

procesamiento más exhaustivo de las zonas en las que hay más probabilidad de encontrar una cara.

Los detectores ya entrenados y disponibles en *OpenCV* se han evaluado para determinar qué combinación y configuración de los mismos resulta más efectiva, teniendo en cuenta el tipo de imágenes que utilizaremos en esta aplicación. Los resultados de esta evaluación se encuentran en la Sección 5.1.

3.1.3. Detector de personas

Otro de los detectores evaluados en este proyecto es el detector de personas, también disponible en *OpenCV*, descriptores de imagen *HOG (Histogram of oriented gradients)*[2].

La apariencia y la forma de un objeto, en este caso personas, se puede caracterizar por la distribución de los gradientes de intensidad locales de los píxeles de la imagen. Para implementar estos descriptores se divide la imagen en pequeñas regiones llamadas celdas. Para cada una de estas celdas se calcula el histograma 1D de las direcciones de gradientes de todos los píxeles que componen la misma. La invarianza a la iluminación, sombras etc., se consigue normalizando estos histogramas.

Una vez calculados los descriptores, se utilizan para entrenar un algoritmo SVM. Para poder reconocer personas a diferentes escalas, la imagen se muestrea a diferentes tamaños.

En la versión final de nuestro prototipo se ha decidido no utilizar este detector ya que, después de evaluarlo junto con los detectores de caras y de movimiento, no aportaba mejoras significativas en la selección del paciente y aumentaba considerablemente el tiempo de cómputo del sistema.

Figura 3.3: Imagen del *UR Fall Detection* dataset en la que se ha detectado al paciente. El segmento o blob de la imagen en la que se ha detectado movimiento, se señala con un rectángulo envolvente en rojo. Su centroide se marca con un punto azul.

3.2. Seguimiento del paciente

Después de realizar la selección del paciente se realiza un seguimiento del mismo a lo largo de la secuencia. Ésto sirve para monitorizar los movimientos del paciente dentro de la estancia.

El paciente seleccionado se representa en la imagen mediante un rectángulo que abarca todos los píxeles que ocupa la persona en la imagen. Este rectángulo se calcula en cada imagen consecutiva de las secuencias utilizando el detector de movimiento. Una vez realizada la detección del paciente, se utiliza únicamente el detector de movimiento para realizar el seguimiento, ya que, el cálculo de este detector es sencillo, mientras que el detector facial es más costoso computacionalmente. La Figura 3.3 muestra una imagen del *UR Fall Detection* dataset en la que se ha detectado un paciente. El rectángulo rojo representa la detección del paciente realizada por el detector de movimiento.

La posición del paciente estará determinada por el centroide del rectángulo obtenido con el detector de movimiento. Para cada imagen se calcula el centroide del rectángulo como se puede observar en la Figura 3.3 y se representa con un punto azul en esta imagen. La distancia a la que se encuentra el paciente de la cámara se determina a partir de las imágenes de profundidad.

Para realizar el seguimiento del paciente durante la secuencia de imágenes, se tiene en cuenta la posición relativa entre los centroides de imágenes consecutivas. Se calcula la distancia euclídea entre el centroide del rectángulo de movimiento de la imagen actual, c_n, y el de la anterior, c_{n-1},

$$d(c_n, c_{n-1}) = \sqrt{(c_{nx} - c_{n-1x})^2 - (c_{ny} - c_{n-1y})^2}, \tag{3.1}$$

siendo c_{nx} y c_{ny} la posición del centroide en la coordenada x e y respectivamente, en la imagen n. Para determinar que el centroide pertenece al paciente seleccionado se establece una distancia máxima entre centroides. Si se supera esta distancia es posible que el seguimiento no se esté realizando correctamente, y sería necesario realizar de nuevo la detección completa, incluyendo el detector de cara, del paciente.

4. Detección de eventos críticos

La detección de eventos críticos, en nuestro caso de estudio, consiste en detectar las posibles caídas que sufra el paciente, así como, el acceso a determinadas zonas restringidas.

Para detectar caídas se utilizan algoritmos de clasificación, mediante los cuáles clasificamos automáticamente las imágenes capturadas en "Caída" y "No caída". Para entrenar este sistema de clasificación automática se han utilizado dos datasets, el *UR Fall Detection* dataset y un dataset propio. Por su parte, para entrenar un clasificador de imágenes es necesario, en primer lugar, decidir cómo representar la información de las mismas. Para ello, se han diseñado una serie de descriptores, calculados en las imágenes y en las zonas de interés obtenidas mediante los detectores de movimiento y caras, explicados en el capítulo anterior. En este capítulo, los descriptores diseñados se detallan en la sección 4.1 y los algoritmos de clasificación utilizados en la sección 4.2. Por otro lado, en la sección 4.3 se explica el algoritmo que se ha implementado para restringir determinadas zonas de una estancia a los pacientes monitorizados.

4.1. Descriptores de imágenes RGB-d

En esta sección se van a describir los descriptores propuestos para representar las imágenes RGB-d, y que servirán para entrenar los algoritmos de clasificación que detectan en las imágenes la posible caída del paciente. Estos descriptores son sencillos de implementar y no requieren mucho tiempo de cómputo, a su vez, no suponen restricciones a los cambios de configuración de las cámaras.

Los descriptores se calculan a partir de las imágenes que proporcionan las dos cámaras RGB-d fijas situadas en la estancia. La primera graba imágenes desde una vista frontal y la segunda está colocada en el techo. La figura 2.2 muestra un esquema del lugar en el que están posicionadas las dos cámaras.

4.1.1. Cámara frontal

Para representar las imágenes RGB grabadas por la cámara frontal se proponen dos descriptores. Ambos representan propiedades geométricas de las zonas de interés, en concreto, de la zona (blob) correspondiente al sujeto monitorizado en la imagen. Este blob se obtiene a partir de las técnicas de detección descritas en el capítulo anterior (Capítulo 3) y se representa como un rectángulo que abarca todos los píxeles que ocupa el sujeto en la imagen, sobre el cual se calcula lo siguiente:

Posición del centroide. Este descritptor es el centroide del rectángulo envolvente del blob de movimiento, y se obtiene

$$d_C = [x, y] = \frac{\sum_{n=1}^{m} x_n}{m}. \tag{4.1}$$

La posición del centroide indica la posición del sujeto monitorizado. Si el sujeto se encuentra de pie o sentado el centroide estará en una posción más alta que si por el contrario el sujeto se ha caído. Los puntos azules de las Figuras 4.1(a) y 4.1(b) representan los centroides del blob en las dos posiciones respectivamente.

Ratio altura/anchura del blob de movimiento. Este descriptor es el ratio entre la altura (h) y la anchura (w) del rectángulo que envuelve el blob de movimiento, y se obtiene

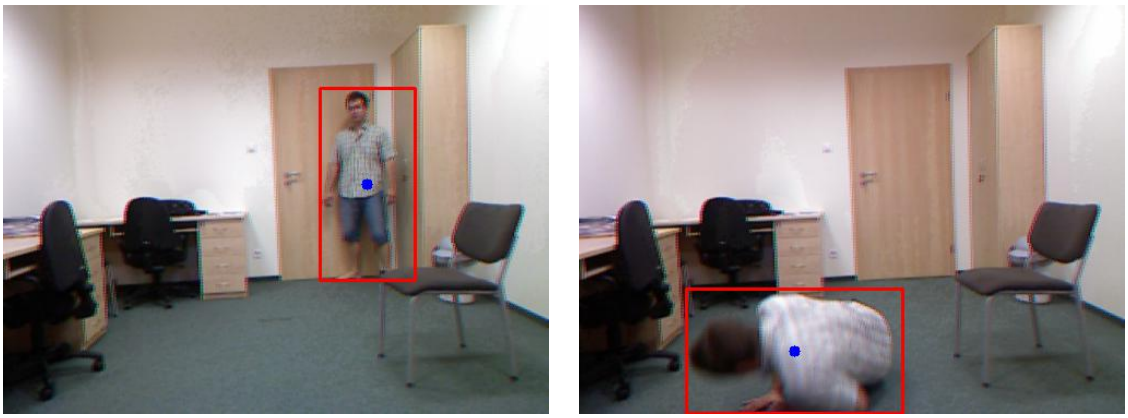
$$d_{ratio-front} = h/w. \tag{4.2}$$

La variación de este parámetro puede indicar si el sujeto se ha caído ya que la relación entre la anchura y la altura cambiará según se encuentre caminando, de pie o en una silla.

4.1.2. Cámara posicionada en el techo

Para representar las imágenes de la cámara posicionada en el techo de la estancia se proponen distintos descriptores. Por un lado se calcula un descriptor de las imágenes RGB de esta vista:

Ratio altura/anchura blob de movimiento. Al igual que en las imágenes de la vista frontal, se calcula el ratio entre la altura (h) y la anchura (w) del rectángulo de



(a) Sujeto en posición normal. (b) Sujeto caído.

Figura 4.1: Ejemplos de sujetos detectados (rectángulo rojo) y posición del centroide del segmento en movimiento correspondiente (punto azul), según distintas posiciones del sujeto. Imágenes del *UR Fall Detection* dataset.

movimiento,

$$d_{ratio-top} = h/w. \tag{4.3}$$

Por otro lado se calculan tres descriptores de las imágenes de profundidad y, en el caso que se detecte movimiento en la imagen, se calculan los descriptores únicamente con los valores de profundidad de esa sección. Si no hay movimiento, los descriptores se calculan con todos los valores de profundidad de la imagen. Los descriptores son los siguientes:

Desviación típica. La desviación típica es una medida estadística que muestra la desviación que presentan los datos en su distribución respecto de la media aritmética. Siendo m el número de píxeles, x_n la profundidad y $\bar{x}$ la media de los valores de profundidad , se calcula

$$d_{s^2} = \frac{1}{m}\sum_{n=1}^{m}(x_n - \bar{x})^2. \tag{4.4}$$

Mediana. La mediana de un conjunto representa el valor central de los datos ordenados de forma ascendente. En este caso x es el valor de profundidad, el valor central será

$$d_M = x_{(n+1)/2}. \tag{4.5}$$

Histograma. El histograma, d_{histo}, proporciona una representación de la frecuencia de aparición de los valores de profundidad. De esta manera, se ofrece una visión general que

permite observar una tendencia por parte de las muestras.

4.2. Métodos de clasificación para la detección de caídas

Para detectar caídas se han entrenado dos algoritmos de clasificación disponibles en la librería *Scikit-learn* [11]. Los clasificadores se han entrenado a partir de las imágenes del *UR Fall Detection* dataset. Para entrenar los clasificadores se utiliza una combinación de los descriptores descritos en la sección anterior (Sección 4.1).

Los dos clasificadores que se utilizan son SVM (Support Vector Machine) y KNN (K-Nearest Neighbor). En ambos casos, cada imagen i, se representa con un vector de descriptores d, y una etiqueda asociada l, con

$$d = [d_C, d_{ratio-front}, d_{ratio-top}, d_{s^2}, d_M, d_{histo}],$$

y $l \in Normal, Cayendo, Caido$. A continuación se detallan los dos tipos de clasificación utilizados. Los resultados obtenidos de cada uno de ellos estan en la sección 5.2.

4.2.1. Support Vector Machine

SVMs (Support Vector Machines) es un algoritmo de aprendizaje supervisado. Dado un conjunto de ejemplos, se entrena un SVM para predecir a que clase pertenece una muestra nueva. Al tratarse de aprendizaje supervisado es necesario que los datos estén etiquetados en las diferentes clases existentes. Cada imagen tiene una etiqueta l. Las clases que nos interesa detectar con la mayor precisión posible son la primera y la tercera, es decir, aquellas imágenes en las que el sujeto esta en una posición normal y las que está caido en el suelo.

Como se puede observar en la Figura 4.2, el SVM busca definir un hiperplano que separa las clases lo más ampliamente posible. En el caso de la figura, únicamente aparecen dos clases y dos características que permiten clasificar las muestras. Sin embargo, cuando se tiene mayor número de características y más de dos clases, se construyen un conjunto de hiperplanos en un espacio dimensional más grande, que separan las muestras de distintas clases.

De esta forma, dado el conjunto de datos $D = \{(x_i, y_i)|x_i \in \mathbb{R}^p, y_i \in \{-1, 1\}\}_{i=1}^n$, x_i es un vector p-dimensional e y_i es la etiqueta de la clase a la que pertenece.

Se busca el vector $\omega \in \mathbb{R}^p$ y el parámetro $b \in \mathbb{R}$ que formen un hiperplano que separe

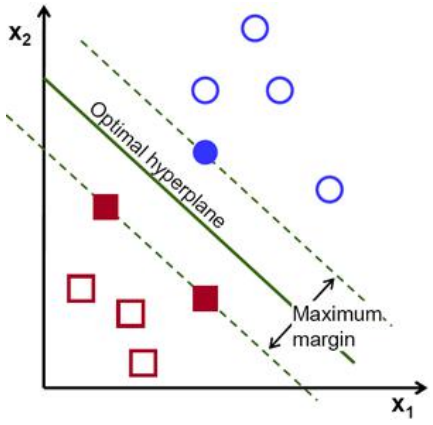


Figura 4.2: Clasificación en dos clases (cículos y cuadrados) con SVM.

las clases, con

$$H = \omega^T x + b = 0. \tag{4.6}$$

El problema de optimización que hay que resolver es:

$$\min_{\omega,b} \frac{1}{2}\omega^T\omega, \tag{4.7}$$

teniendo en cuenta que

$$y_i(\omega^T x_i + b) \geq 1. \tag{4.8}$$

4.2.2. K-Nearest Neighbors

El método K-NN (K nearest neighbors) es otro algoritmo de clasificación de aprendizaje supervisado. El algoritmo devuelve la probabilidad de que una imagen i pertenezca a una clase l. Los ejemplos de entrenamiento son vectores multidimensionales de características d, con descriptores calculados a partir de cada imagen i. En la fase de entrenamiento el algortimo almacena estos vectores de características y las etiquetas que representan la clase a la que pertenece cada una de las muestras. En la fase de clasificación, se define el parámetro K, que es una constante, y se calcula la distancia entre los vectores almacenados y el de la nueva muestra. La nueva muestra pertenece a la clase que más se repita entre los K vecinos más cercanos. Un ejemplo de cómo funciona el algoritmo se puede observar en la Figura 4.3, la nueva muestra pertenecerá a una clase distinta dependiendo del valor de K.

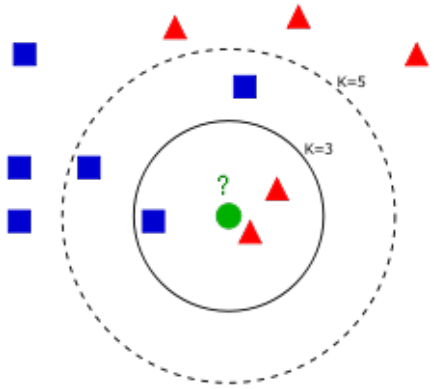


Figura 4.3: Clasificación con KNN.

4.3. Acceso restringido a determinadas zonas de la estancia.

Adicionalmente, se desea que nuestra aplicación incluya un sistema de alarma que indique situaciones en las que el sujeto abandona deteminadas zonas.

Para implementar esta función se parte del seguimiento realizado al paciente. Como se explica en la Sección 3.2, la posición del paciente en las imágenes de una secuencia se representa con el centroide del rectángulo obtenido mediante el detector de movimiento.

Los límites de la estancia a los que el paciente tendrá el acceso restringido se marcan en el 3D de la escena. Para poder relacionar la posición del paciente con estos límites, es necesario conocer la posición en 3D del centroide mencionado anteriormente.

Los puntos en una imagen se representan como coordenadas 2D y se corresponden con un punto de coordenadas 3D en la escena real. Las coordenadas 3D son la posición X e Y del punto y la profundidad Z. Las cámaras utilizadas disponen de sensores que miden la profundidad de todos los puntos de la imagen. La imagen de la Figura 4.4 muestra un esquema de la proyección de puntos 3D en la imagen.

Con la información de la profundidad de cada píxel (Z) y los parámetros internos de la cámara, las coordenadas métricas del punto 3D de la escena se pueden calcular de la siguiente manera:

$$(x, y, Z) \mapsto (X, Y, Z), \tag{4.9}$$

$$X = \frac{x - o_x}{f} * Z,$$

$$Y = \frac{y - o_y}{f} * Z,$$

$$Z = Z.$$

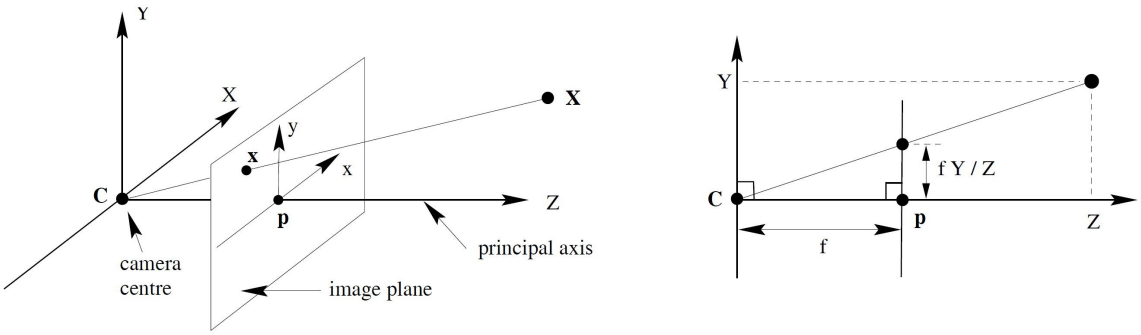


Figura 4.4: Esquema de la proyección de puntos 3D .

Donde x, y son las coordenadas del punto en la imagen, o_x, o_y son las coordenadas del origen de la cámara y f la distancia focal. La posición relativa entre la proyección 3D del centroide y los límites establecidos en la estancia sirve para determinar la salida del paciente o su acceso a una zona peligrosa. En la Sección 5.2.3 se muestra cómo ésto se ha aplicado en nuestro sistema.

5. Resultados experimentales

En este capítulo se recogen los resultados obtenidos en la evaluación de los métodos utilizados para el desarrollo de la aplicación. En primer lugar se han evaluado diferentes técnicas de detección disponibles y se ha estudiado la manera de combinarlas. Se han obtenido los resultados numéricos de la precisión de cada una de ellas.

Por otro lado se ha evaluado la precisión de dos algoritmos de clasificación entrenados a partir del *UR Fall Detection* dataset y un dataset propio. Así, en la Figura 5.1 se puede observar una muestra de las secuencias del primero y en la Figura 5.2 una muestra del segundo. El dataset propio consta de secuencias más realistas y más complejas. Los resultados obtenidos de la evaluación de los clasificadores se han comparado con los obtenidos en el trabajo que publicó el dataset público utilizado para la evaluación[5].

Figura 5.1: Imágenes de una de las secuencias del *UR Fall Detection* dataset.

Figura 5.2: Imágenes de una de las secuencias del dataset propio.

5.1. Evaluación de técnicas de detección de caras y movimiento

Descripción. Con este experimento, se pretende evaluar la combinación de los detectores disponibles que funciona mejor, teniendo en cuenta el tipo de imágenes utilizadas.

Para monitorizar a un paciente a través únicamente de imágenes, es necesario detectarlo en las mismas. Para ello, se han combinado varias técnicas de detección que permiten seleccionar al paciente automáticamente y realizar el seguimiento del mismo. Las técnicas de detección estudiadas se han desarrollado en la Sección 3 de esta memoria.

Si tenemos en cuenta que la aplicación está pensada para entornos de interior, podemos suponer que el movimiento en las imágenes se corresponderá principalmente con las personas que se encuentran en ellas. Por esta razón la detección de sujetos se basa en el detector de movimiento.

A partir de este detector se extrae un blob representando el movimiento del paciente respecto de la imagen de referencia. Para seleccionar el blob correcto se utilizan técnicas de detección facial y una vez obtenido el blob de movimiento más grande, se aplica a toda la imagen un detector de caras. En el caso de que se produzca intersección entre el blob de movimiento y la cara, éste será seleccionado para realizar el seguimiento.

Resultados: evaluación detectores de caras. La librería *OpenCV*, utilizada durante este proyecto, dispone de distintos detectores faciales pre-entrenados. Para evaluar cual de ellos funciona mejor se ha etiquetado una de las secuencias del *UR Fall Detection* dataset, y se ha calculado la precisión y el recall de cada uno.

El recall representa el número de eventos que se clasifican correctamente con respecto al número total de eventos. Por otro lado, la precisión representa la proporción de eventos que se han clasificado correctamente, y se calculan, respectivamente,

$$Recall = \frac{T_p}{T_p + F_n}, \tag{5.1a}$$

$$Precision = \frac{T_p}{T_p + F_p}. \tag{5.1b}$$

Los resultados de este experimento se muestran en la Tabla 5.1. Los detectores han sido entrenados con datasets diferentes y, por eso, para las secuencias de imágenes utilizadas en este experimento, cada uno obtiene un resultado distinto. Descartando el *frontalface_alt_tree.xml*, que no funciona para estas secuencias, el resto de detectores tienen una precisión muy similar. Sin embargo, se puede comprobar que el recall de todos ellos es mucho más bajo. Ésto quiere decir que cuando detectan algo en una imagen es my probable que sea una cara, pero solo son capaces de detectar entre el 15 % y el 30 % de las caras dependiendo de cada uno de ellos. En este caso, el que mejor funciona es el detector *frontalface_default.xml* y es el que se va a utilizar en nuestra aplicación.

Se ha notado que hay secuencias en las que el paciente no aparece de frente en ninguna

	Recall	Precision
frontalface_default.xml	29,41 %	100 %
frontalface_alt2.xml	15,68 %	94,11 %
frontalface_alt_tree.xml	0 %	0 %
frontalface_alt.xml	17,64 %	100 %

Tabla 5.1: Estadísticas de los detectores faciales de *OpenCV* utilizando las imágenes del *UR Fall Detection* dataset.

de las imágenes. Ésto puede ocurrir también cuando la aplicación esté funcionando en un entorno real, por lo tanto, se ha decidido combinar el detector de caras frontal con un detector de caras de perfil. En *OpenCV* hay disponible un detector de caras pre-entrenado con imágenes de caras de perfil, *profileface.xml*. Se ha calculado la precisión de la combinación de estos dos detectores de la misma forma que los detectores frontales. El recall es de un 34,31 % frente al 29,41 % que se obtiene con el *frontalface_default.xml*.

5.2. Evaluación del sistema de detección de acciones

Descripción. La finalidad de este experimento es evaluar el funcionamiento del sistema de detección de caídas. Para ello se evalúa la precisión del algoritmo de clasificación SVM (Support Vector Machine) utilizando tanto las secuencias del *UR Fall Detection* dataset como las secuencias del dataset propio.

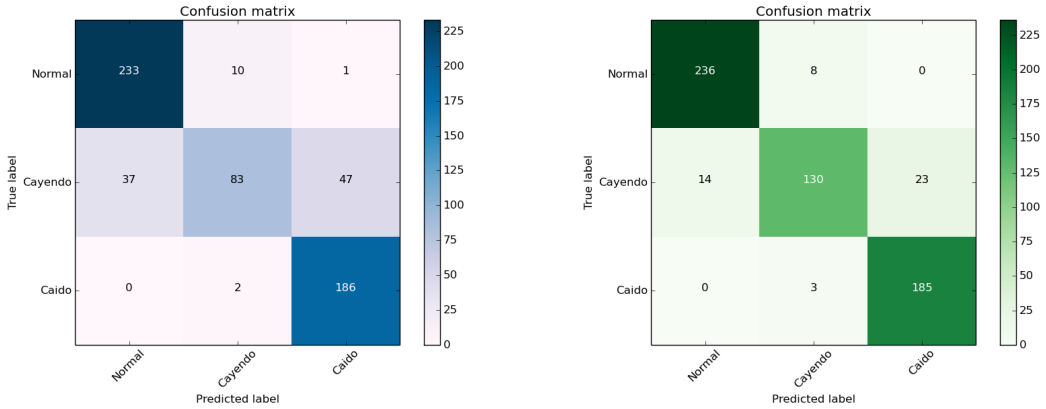
En primer lugar se utiliza el *UR Fall Detection* dataset que dispone de 30 secuencias de imágenes. El 80 % de las imágenes se han utilizado para entrenar el clasificador SVM y el 20 % para evaluar su precisión. Estas imágenes están etiquetadas en tres clases diferentes. La primera de ellas corresponde al sujeto en una posición normal, en la segunda el sujeto está cayendo al suelo y en la tercera el sujeto está caído en el suelo.

Para evaluar el clasificador SVM se calculan la precisión y el recall y ambas medidas corresponden con las ecuaciones 5.1b y 5.1a respectivamente.

Resultados y discusión. En la Tabla 5.2 se observan los resultados de precisión y recall de las tres clases. Para la clase intermedia (paciente cayendo al suelo) el clasificador tiene una precisión muy alta, similar a la de las dos clases restantes. Por contra, el recall es mucho más bajo. Ésto quiere decir que cuando una imagen es clasificada con esta etiqueta, hay un 87,36 % de probabilidad de que esté clasificada correctamente. Sin embargo de todas las imágenes que pertenecen a esta clase solamente el 49,70 % han sido clasificadas correctamente. Ésto se debe a que las imágenes de esta clase tienen similitudes con las imágenes de las otras dos, y por lo tanto el clasificador las confunde. Lo que nos interesa

	Normal	Cayendo	Caído
Recall	95,49 %	49,70 %	98,93 %
Precisión	86,29 %	87,36 %	79,48 %

Tabla 5.2: Precisión y Recall de las tres clases para el algoritmo SVM



(a) SVM (b) KNN

Figura 5.3: Matrices de confusión de los algoritmos SVM y KNN, entrenados con el 80 % de las imágenes del *UR Fall Detection* dataset.

es que el clasificador sea capaz de clasificar correctamente las clases "Normal" y "Caído".

La Figura 5.3(a) corresponde a la matriz de confusión del clasificacdor SVM. Las columnas de la matriz de confusión representan las predicciones del clasificador, mientras que las filas representan la clase a la que corresponden en realidad. Las cifras del interior de la matriz representan el número de imágenes clasificadas de cada tipo. La matriz de confusión muestra los mismos resultados que los valores de precisión y recall. El algoritmo es capaz de clasificar correctamente la mayoría de las imágenes que corresponden a las clases "Normal" y "Caído", pero solo clasifica correctamente aproximandamente la mitad de las imágenes de la clase "Cayendo".

Descripción Además del SVM se ha entrenado un algoritmo K-NN (K-Nearest Neighbor). Para entrenar se han utilizado el 80 % de las imágenes y el otro 20 % se ha utilizado para calcular la precisión, al igual que en el caso del algoritmo anterior. El parámetro K es el número de vecinos que compara el algortimo con la nueva muestra que es asignada a la clase predominante. El valor de K en este caso es $K = 5$. A la hora de elegir este parámetro hay que encontrar un compromiso ya que los valores muy altos de K reducen el ruido pero crean límites entre clases parecidas.

	Normal	Cayendo	Caído
Recall	96,72 %	77,84 %	98,40 %
Precisión	94,40 %	92,19 %	88,94 %

Tabla 5.3: Precisión y Recall de las tres clases para el algoritmo KNN.

Resultados y discusión. En la Tabla 5.3 aparecen los resultados de las dos medidas. En el caso de las clases "Normal" y "Caído" los resultados son muy similares a los del SVM. Tanto la precisión como el recall son muy altos en los dos casos.

En la Figura 5.3(b) se muestra la matriz de confusión del algoritmo KNN. Los resultados obtenidos con este dataset son muy similares a los obtenidos en el artículo [5]. En el Anexo A se muestran la comparación entre sus resultados y los obtenidos en este TFM.

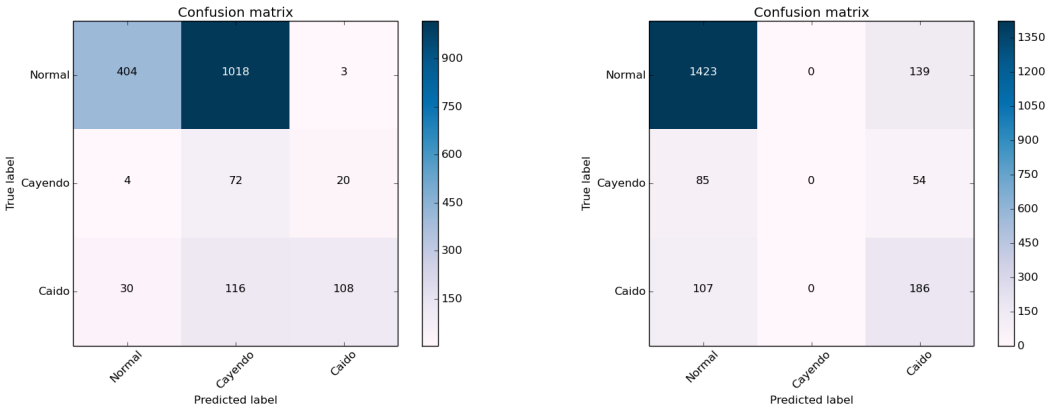
Como muestran los resultados expuestos en esta sección, la precisión de los clasificadores es muy alta, y por lo tanto, son capaces de detectar la mayor parte de las imágenes en las que el sujeto se ha caído. Ésto puede deberse a que los datos del *UR Fall Detection* dataset son muy sencillos. Tanto el entorno utilizado para la grabación de las imágenes como los movimientos que realizan, son muy simples. En las imágenes tampoco aparecen oclusiones de las personas por objetos de la habitación, ni hay cambios en la iluminación. Por estas razones se ha decidido utilizar otro conjunto de datos un poco más complejo para evaluar el sistema de detección de caídas. El Anexo E contiene una muestra más amplia de las secuencias de los datasets.

5.2.1. Evaluación del sistema de detección de eventos con un dataset propio

El nuevo dataset está compuesto por 13 secuencias grabadas en un entorno más real. En las secuencias cambia la posición de las cámaras con respecto al *UR Fall Detection* dataset, existen cambios de iluminación durante el transcurso de las mismas y hay objetos que producen oclusiones parciales de las personas. Un ejemplo de estas secuencias se muestra en la Figura 5.2.

En primer lugar, se evalúa con el nuevo dataset el algoritmo SVM entrenado con las secuencias del *UR Fall Detection* dataset. En la Figura 5.4(a) se observa que el clasificador no es capaz de clasificar correctamente las imágenes pertenecientes a las clases "Normal" y "Caído". Clasifica la mayor parte de las imágenes en la clase intermedia.

Las imágenes de la clase "Cayendo" tienen características similares a las otras dos clases, utilizar estas imágenes no aporta ninguna información relevante al clasificador. Por esta razón, se vuelve a entrenar el algoritmo SVM, esta vez, sin utilizar las imágenes con esta etiqueta. Los resultados se muestran en la Figura 5.4(b) y mejoran notablemente.



(a) (b)

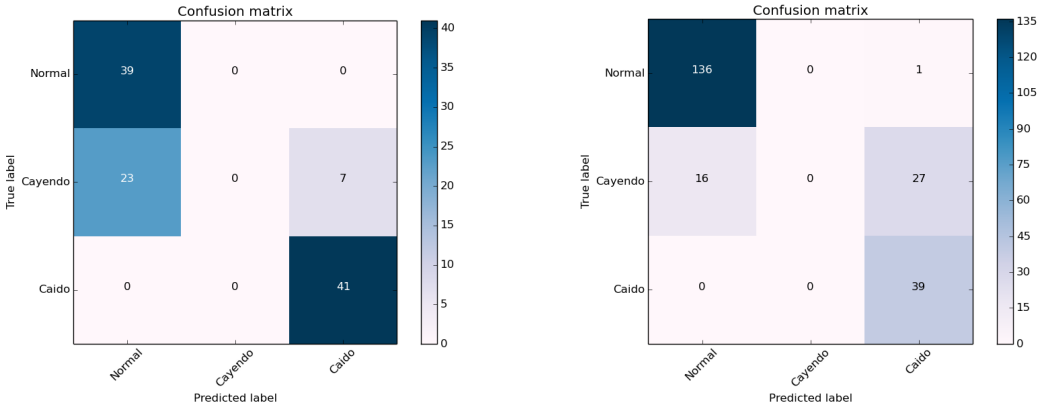
Figura 5.4: Matrices de confusión entrenando el algortimo con datos del *UR Fall Detection* dataset. En (a) se han utilizado todas las imágenes para entrenar el clasificador. En (b) se ha entrenado sin las imágenes de la clase "Cayendo". Los clasificadores se han evaluado con el dataset propio.

Aun así, sólo el 63 % de las imágenes de caídas se clasifican correctamente.

Para intentar mejorar los resultados obtenidos hasta ahora, el siguiente experimento se realiza utilizando las secuencias del dataset propio para entrenar el algoritmo de clasificación. Como ya se ha dicho, estas secuencias son más realistas y más complejas que las del *UR Fall Detection* dataset, por lo tanto, se podría pensar que se obtendrá un clasificador más genérico.

En este experimento se entrena el clasificador con 12 de las 13 secuencias disponibles del dataset propio. Previamente, para solucionar los problemas de variación en la iluminación, se han normalizado todas las imágenes de una misma secuencia respecto a su imagen de referencia. Para evaluar el clasificador se han utilizado la secuencia restante de este dataset y una de las secuencias del *UR Fall Detection* dataset. Los resultados se observan en la Figura 5.5. El recall de las clases "Normal" y "Caído" es prácticamente del 100 % en todos los casos. La matriz de confusión también muestra que es capaz de clasificar prácticamente todas las imágenes de estas dos clases correctamente.

El clasificador también se ha entrenado utilizando los descriptores de las imágenes RGB y de profundidad por separado. Se ha entrenado el algoritmo con 12 secuencias del dataset propio y se ha evaluado con la secuencia restante del mismo. Los resultados se muestran en el Anexo B.



(a) (b)

Figura 5.5: Matrices de confusión entrenando el algortimo con datos del dataset propio. En (a) se ha evaluado el clasificador con una secuencia del dataset propio. En (b) se ha evaluado con una secuencia del *UR Fall Detection* dataset.

5.2.2. Detección de caídas analizando la clasificación de las secuencias temporalmente

Hasta ahora, se han clasificado las imágenes de forma independiente unas de otras sin tener en cuenta su posición dentro de las secuencias, sin embargo, el problema de detección que se quiere resolver tiene una componente temporal.

Las imágenes de una caída en las que aparece el individuo en el suelo tienen posiciones consecutivas dentro de una secuencia. Para detectar una caída no sería necesario que el algoritmo clasifique correctamente todas las imágenes. En lugar de eso, sería suficiente que el algoritmo clasifique cierto número de imágenes consecutivas con la etiqueta de "Caído". De esta forma, se podría afirmar que se ha producido el evento.

En este experimento se evalúa el clasificador con las secuencias del dataset propio. La aplicación muestra por pantalla un mensaje cuando se ha producido el evento. Un ejemplo de cómo la aplicación muestra este mensaje se puede observar en la Figura 5.6.

Se ha realizado un ajuste de parámetros para determinar un umbral de la cifra de imágenes con etiqueta "Caído" que se deben detectar consecutivamente para afirmar que se ha producido el evento. Si esta cifra es muy alta es posible que determinados eventos no sean detectados. Por otro lado, si la cifra es muy baja puede aumentar el número de falsos positivos. El procedimiento completo se muestra en el algoritmo 1.

La Tabla 5.4 muestra los resultados de la evaluación de este experimento variando la cifra de imágenes consecutivas que se deben detectar para afirmar que se ha producido un evento, para los distintos usuarios del dataset propio. Primero se establece un *umbral* =

(a) Vista frontal (b) Vista techo.

Figura 5.6: Aplicación de deteccion de caídas. Muestra por pantalla un mensaje de alarma en caso de que se haya producido una caída del paciente monitorizado. En estas imágenes se observa el paciente detectado en las dos cámaras y el mensaje de alarma que aparece en las imágenes.

Algoritmo 1

Entrada: I_{path} Path de la nueva secuencia de imágenes.
$cont = 0$ Contador inicializado a 0

Salida: El algoritmo devuelve una imagen en la que se muestra el paciente detectado y un mensaje de alarma si ha ocurrido el evento.

```
para I en I_path hacer
    I_label-pred ←prediccion de la clase de I
    si I_label-pred == 1 entonces
        cont ← cont + 1
        si cont >= 5 entonces
            Escribir mensaje de alarma en la imagen.
        fin si
    si no
        cont = 0
    fin si
    Mostrar imagen.
fin para
```

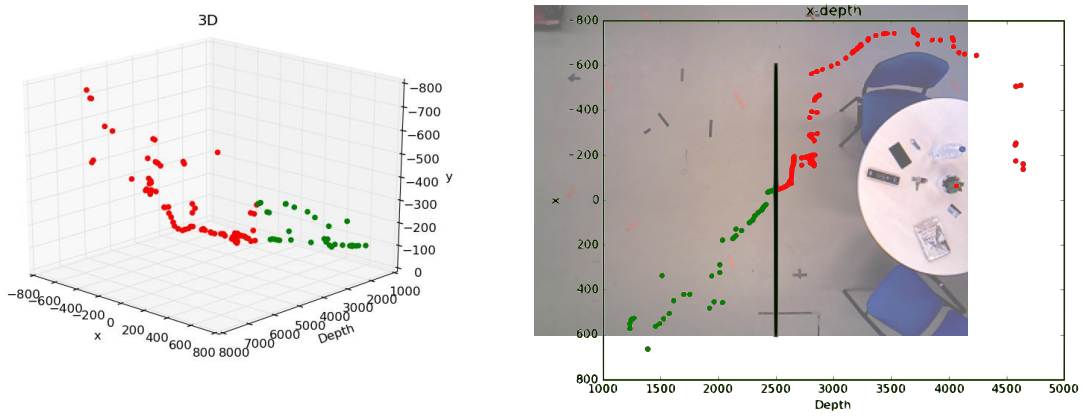
	Umbral = 10			Umbral = 5		
	TP	FP	FN	TP	FP	FN
Usuario 1	3	-	1	4	-	-
Usuario 2	2	-	-	2	-	-
Usuario 3	2	-	-	2	1	-

Tabla 5.4: Detección de caídas en las secuencias del dataset propio variando la cifra de imágenes consecutivas de caídas detectadas.

10, con esta cifra el detector de caídas detecta 3 de los 4 eventos del primer usuario. Ésto puede ocurrir, por ejemplo, cuando tras la caída el paciente queda oculto por algún objeto de la habitación, o está fuera del rango de las cámaras. Para los usuarios 2 y 3 se detectan todas las caídas. Posteriormente, se establece un *umbral* = 5, en este caso, detecta todas las caídas de los usuarios. Sin embargo, detecta un falso positivo en el usuario 3. Se prima la detección de todos los eventos por lo tanto se ha establecido el umbral en 5 imágnes.

5.2.3. Restricción de acceso a zonas de una estancia.

Esta función activa una alarma en los casos en los que el paciente sale de la zona segura. Estos límites pueden establecerse en puertas o zonas peligrosas de una estancia. Sin embargo para evaluarla y poder mostrar los resultados gráficamente, se ha establecido un límite en el centro de la estancia de las imágenes del dataset propio.



(a) Reconstrucción 3D . (b) Límites de la estancia.

Figura 5.7: (a) muestra la reconstrucción 3D de la trayectoria del paciente y (b) muestra la trayectoria del paciente vista desde la cámara del techo y los límites establecidos en la estancia.

Se conoce la posición en la imagen del paciente monitorizado, que es representada por el centroide del rectángulo que devuelve el detector de movimiento. Este centroide es un punto en la imagen en coordenadas 2D. Con los parámetros internos de las cámaras se calcula su posición 3D en la escena real de la forma que se detalla en la Sección 4.3. La representación 3D se muestra en la Figura 5.7(a). La trayectoria en 3D muestra el centroide en cada frame del blob que corresponde al paciente monitorizado. Debido al ruido y a pequeños cambios de tamaño, la posición de este punto en 3D no construye una trayectoria suave como cabría esperar. Sin embargo, el objetivo no es una reconstrucción

precisa, sino detectar cambios significativos en la posición, que impliquen que el usuario ha cruzado ciertos límites de la estancia.

En la Figura 5.7(b) se muestra el límite, establecido en la estancia, de las secuencias del dataset propio que está representado por una línea negra. En esta figura se muestra, de forma aproximada, la trayectoria que sigue el paciente vista desde la cámara que está colocada en el techo. Los ejes de la gráfica de esta imagen equivalen a las coordenadas x de la posición del paciente y a la profundidad a la que se encuentra respecto de la cámara frontal. Una vez sobrepasado el límite permitido, los puntos de la trayectoria del paciente cambian de color representando que ha salido de la zona segura.

6. Conclusiones y trabajo futuro

6.1. Conclusiones

El objetivo principal de este proyecto ha sido la implementación de un prototipo para la monitorización de pacientes utilizando un sistema multi-cámara. El objetivo era diseñar un prototipo realista y estudiar las ventajas de combinar información de distintos puntos de vista. Los resultados obtenidos han sido satisfactorios. El sistema implementado es capaz de detectar el paciente de forma automática, y realizar el seguimiento del mismo a lo largo de las secuencias de imágenes RGB y de profundidad, capturadas por dos cámaras fijas. El sistema también realiza el reconocimiento de determinados eventos críticos.

La detección automática del paciente se ha realizado combinando distintas técnicas de detección, de movimiento y de caras. Con la combinación final propuesta tras realizar los experimentos, se detectan satisfactoriamente los pacientes en todas las secuencias evaluadas. A partir de las imágenes y las zonas de interés obtenidas mediante los detectores se han diseñado una serie de descriptores de las imágenes. Los descriptores de imagen propuestos e implementados son sencillos de calcular, no suponen mucho tiempo de cómputo y son independientes de la configuración de las cámaras.

En lo relativo a la detección de eventos críticos, el sistema detecta las posibles caídas del paciente y su acceso a zonas restringidas o peligrosas de una estancia. La detección de caídas se ha realizado mediante un clasificador de imágenes, representadas por los descriptores calculados, que se ha evaluado con dos datasets de distinta complejidad. Finalmente, se ha implementado un sistema que utiliza la información temporal y la robustez que se puede obtener al tratar las imágenes de la secuencia en conjunto, para determinar si se produce el evento. La aplicación muestra un mensaje de alarma cuando ésto ocurre.

Por último, se ha implementado un sistema de alarmas para detectar cuando el paciente abandona la zona segura de la estancia. Tras establecer unos límites en la estancia, se ha

calculado la trayectoria del paciente monitorizado. La aplicación muestra visualmente un mensaje de alarma en caso de que el paciente sobrepase estos límites.

Por lo tanto, todos los objetivos del prototipo se han cubierto, tanto los funcionales, como el hecho de conseguir un sistema realista, de bajo coste y que resulte una buena alternativa en situaciones donde por ejemplo el paciente no puede o no quiere llevar sensores encima todo el tiempo.

6.2. Trabajo futuro

En la línea del trabajo realizado en este TFM sería interesante analizar si es posible utilizar únicamente la profundidad para calcular los descriptores de las imágenes. Esto supondría no depender de la iluminación de la escena y proporcionaría privacidad al ususario. Para ello, sería necesario estudiar más en detalle los descriptores que se pueden extraer de las imágenes de profundidad.

Por otro lado se podría seguir trabajando en aumentar la robustez de la detección automática del paciente, y que el sistema fuera capaz de reconocer al paciente, y no solo de detectarlo en la imagen.

Por último, otro paso futuro tras este proyecto sería analizar en más detalle los requerimientos para el funcionamiento de este sistema en tiempo real y evaluar el prototipo de manera más exhaustiva en un entorno real de aplicación.

A. Comparación de resultados

El Cuadro A.1 muestra los resultados al realizar *10-fold cross-validation* con el clasificador SVM y los datos del *UR Fall Detection* dataset, obtenidos en [5] y en este TFM.

En [5] utilizan un sistema compuesto por cámaras de imágenes RGB y *depth*, y sensores. En este caso analizaremos los resultados obtenidos únicamente con el sistema visual, para poder compararlos con los obtenidos en este TFM.

Tanto los resultados obtenidos en [5], como los obtenidos en uno de los experimentos de este TFM, muestran que los clasificadores tienen una precisión y un recall muy altos. Los resultados obtenidos en ambos casos son muy similares.

		SVM	KNN
TFM	Precisión	99,62 %	99,8 %
	Recall	99,8 %	99,15 %
UR Fall Detection	Precisión	100 %	100 %
	Recall	99,05 %	99,05 %

Tabla A.1: Comparación de los resultados obtenidos en este TFM con los obtenidos en el artículo [5].

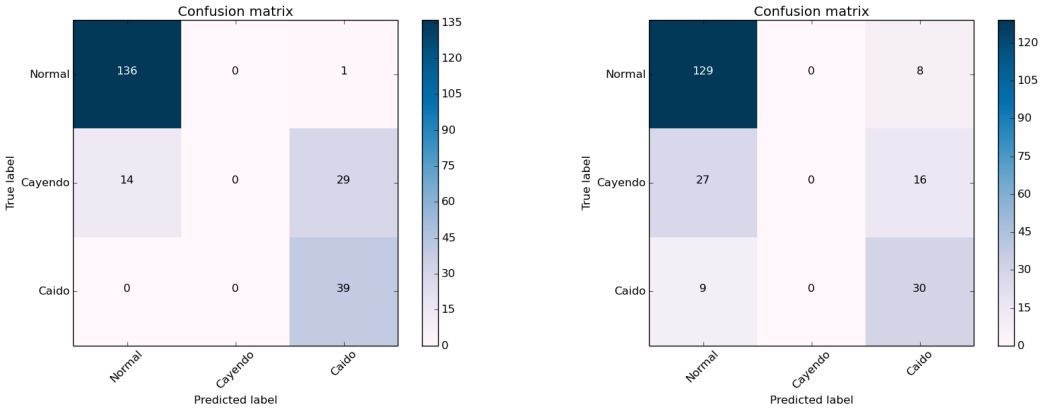
B. Experimento SVM por cámaras

En este anexo se muestran los resultados obtenidos al entrenar el algoritmo de clasificación SVM con las secuencias del dataset propio, utilizando únicamente los descriptores obtenidos de las imágenes RGB y de profundidad respectivamente.

Los resultados se muestran en el Cuadro B.1, y la Figura B.1 muestra las matrices de confusión obtenidas. Los resultados utilizando los descriptores RGB son similares, aunque inferiores, a los obtenidos en la Sección 5.2.1. Sin embargo, si se utilizan únicamente los descriptores de profundidad las cifras de la precisión y el recall son más bajas.

		Normal	Cayendo	Caído
RGB	Precisión	90,66 %	0 %	56,21 %
	Recall	99,27 %	0 %	100 %
DEPTH	Precisión	78,18 %	0 %	55,55 %
	Recall	94,16 %	0 %	76,92 %

Tabla B.1: Resultados numéricos de la evaluación de un algoritmo SVM entrenado con distintos vectores de características.



(a) SVM con características RGB

(b) SVM con carácterísticas de profundidad.

Figura B.1: En (a) el SVM se ha entrenado con los descriptores que represetan a las imágenes RGB. En (b) el SVM se ha entrenado con los descriptores que representan a las imágenes de profundidad.

C. Sensores RGB-d

En este TFM se han utilizado sensores RGB-d para capturar las secuencias de imágenes con las que se han realizado los experimentos. Los sensores RGB-d están compuestos por una cámara RGB, y un sensor de profundidad. Este sensor es un proyector de infrarrojos mediante el cual se adquieren los datos de profundidad de cada píxel de la imagen. Su rango es de 5 metros aproximadamente.

Posteriormente, los datos de profundidad se alinean con la cámara RGB. El resultado es una imagen RGB con un valor de profundidad asociado a cada píxel. Los datos se representan como una nube de puntos en un espacio tridimensional.

El sensor RGB-d utilizado en este TFM para la grabación de imágenes ha sido ASUS Xtion PRO LIVE(Figura C.1).

Figura C.1: Sensor RGB-d ASUS Xtion PRO LIVE.

D. Distribución de tareas

La distribución de tareas durante la duración de este TFM se ha realizado de la siguiente manera:

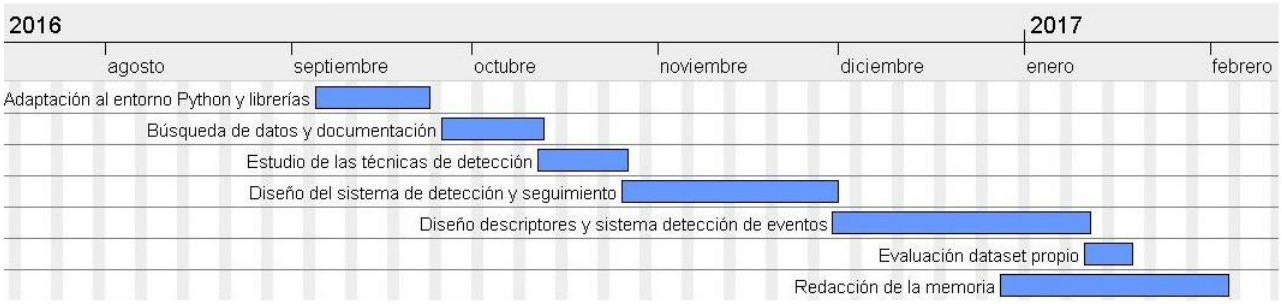


Figura D.1: Diagrama de Gantt.

E. Datasets

En este TFM se han utilizado dos datasets de secuencias de imágenes. El primero de ellos es un dataset público, *UR Fall Detection* dataset, y el segundo es un dataset propio. Ambos contienen secuencias de interior en las que los usuarios sufren caídas desde distintas posiciones.

Algunas muestras de las secuencias del *UR Fall Detection* dataset se pueden ver en las Figuras E.1 y E.2. Como se puede observar, este dataset contiene secuencias de movimientos simples, sin oclusiones de las personas por los objetos de la habitación y sin cambios en la iluminación.

Las Figuras E.3, E.4 y E.5 muestran secuencias del dataset propio. Cada una de ellas corresponde con uno de los tres usuarios que aparecen en este dataset. Se puede observar que, estas imágenes son más complejas que las del dataset público.

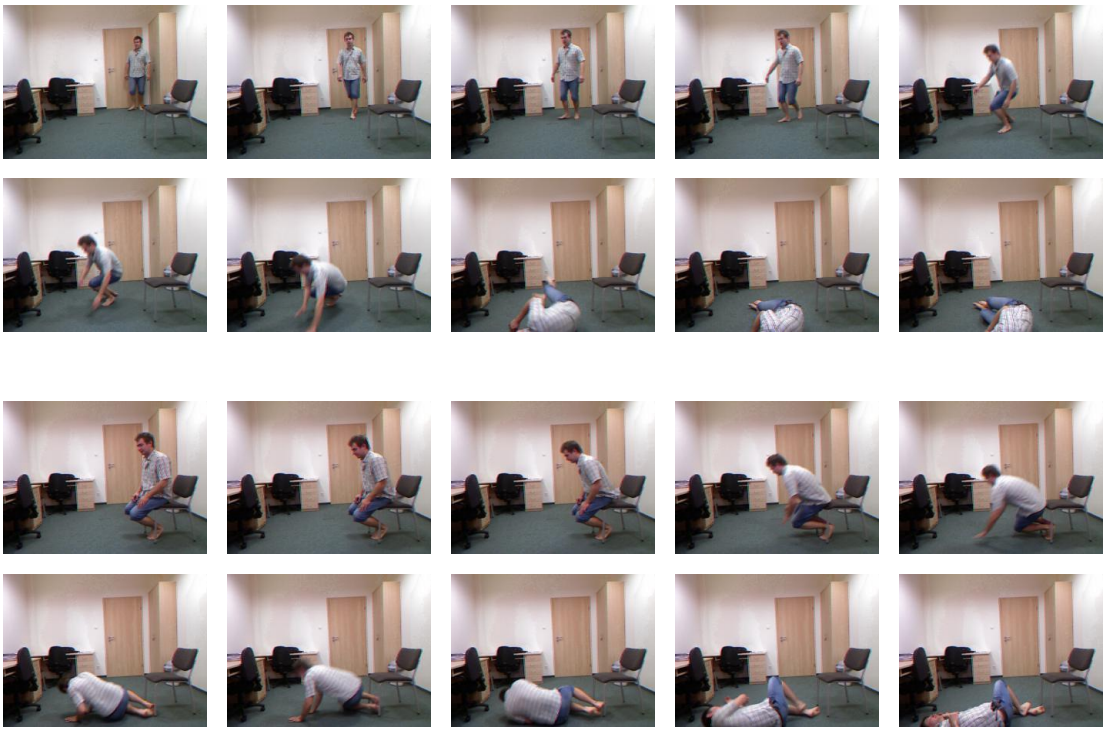


Figura E.1: Secuencias del *UR Fall Detection* dataset.

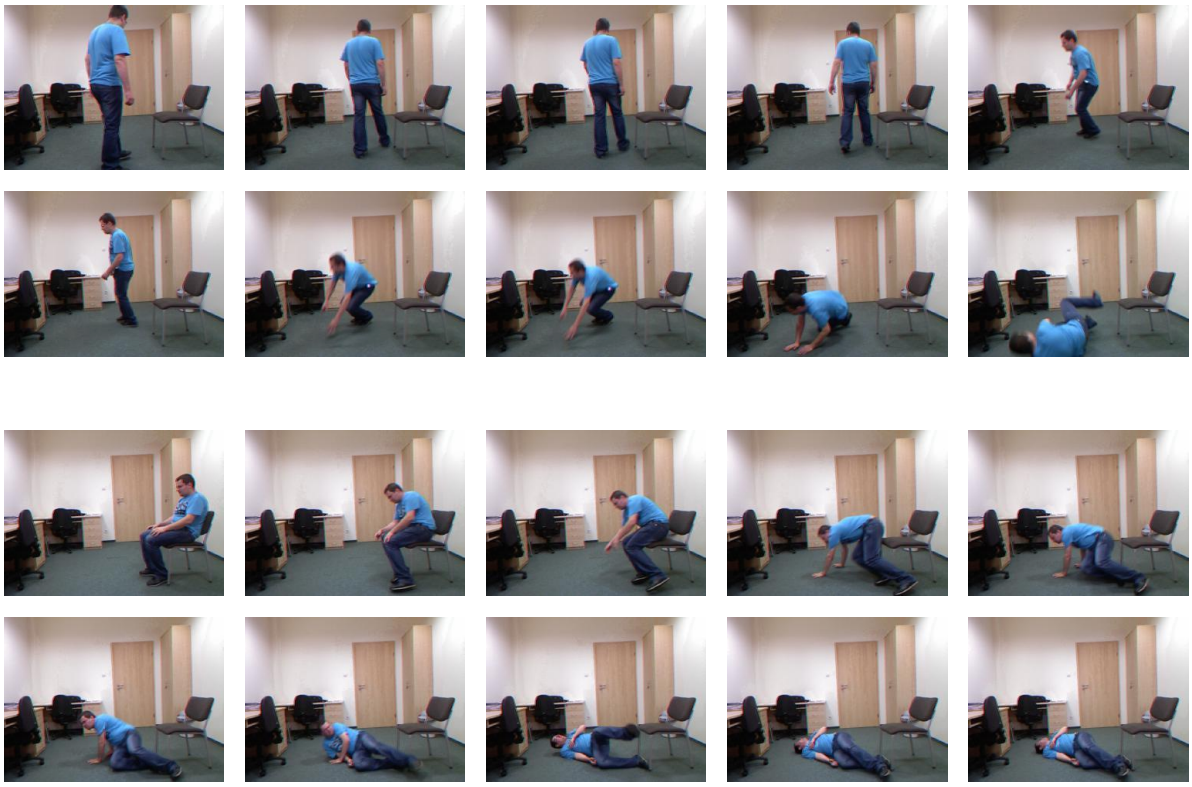


Figura E.2: Secuencias del *UR Fall Detection* dataset.

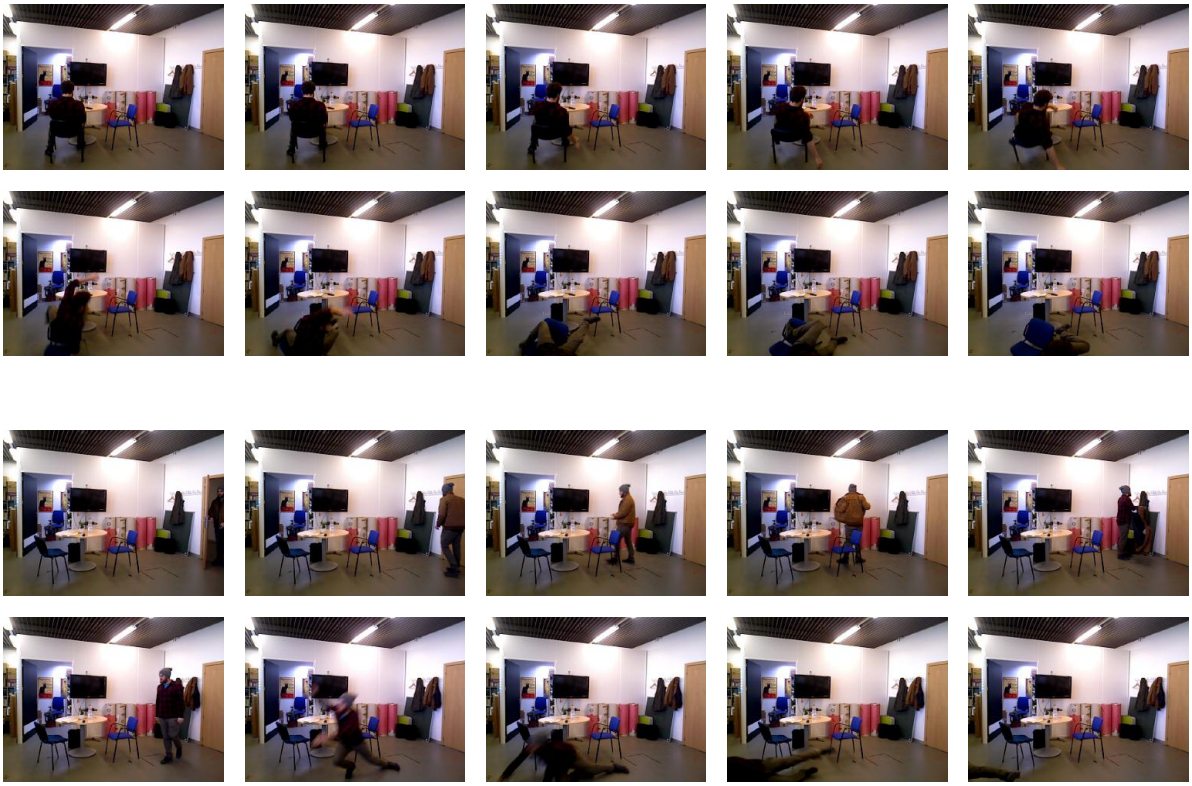


Figura E.3: Secuencias dataset propio. Usuario 1.

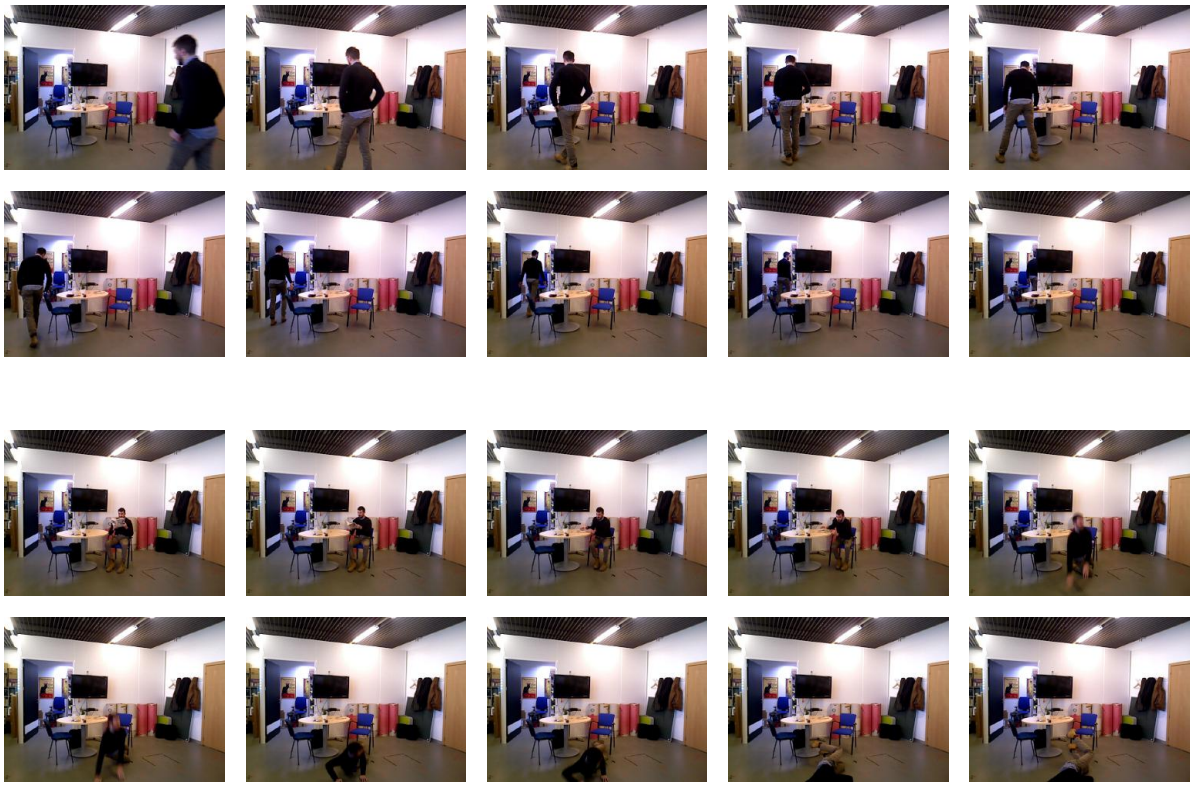


Figura E.4: Secuencias dataset propio. Usuario 2.

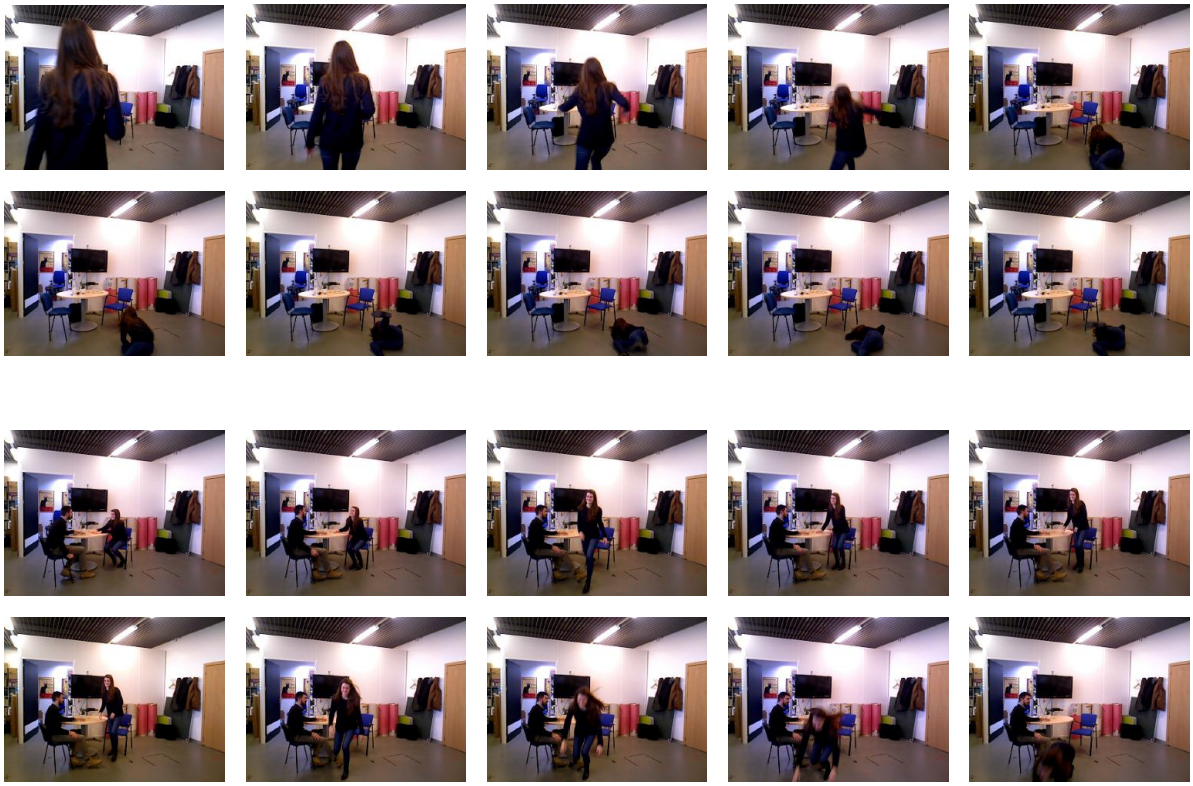


Figura E.5: Secuencias dataset propio. Usuario 3.

Bibliografía

[1] Majd Alwan, Prabhu Jude Rajendran, Steve Kell, David Mack, Siddharth Dalal, Matt Wolfe, y Robin Felder. A smart and passive floor-vibration based fall detector for elderly. En *Information and Communication Technologies, 2006. ICTTA'06. 2nd*, tomo 1, págs. 1003–1007. IEEE, 2006.

[2] Navneet Dalal y Bill Triggs. Histograms of oriented gradients for human detection. En *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, tomo 1, págs. 886–893. IEEE, 2005.

[3] Yoav Freund y Robert E Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. En *European conference on computational learning theory*, págs. 23–37. Springer, 1995.

[4] Pakorn KaewTraKulPong y Richard Bowden. An improved adaptive background mixture model for real-time tracking with shadow detection. En *Video-based surveillance systems*, págs. 135–144. Springer, 2002.

[5] Bogdan Kwolek y Michal Kepski. Human fall detection on embedded platform using depth maps and wireless accelerometer. *Computer methods and programs in biomedicine*, 117(3):489–501, 2014.

[6] Qiang Li, John A Stankovic, Mark A Hanson, Adam T Barth, John Lach, y Gang Zhou. Accurate, fast fall detection using gyroscopes and accelerometer-derived posture information. En *2009 Sixth International Workshop on Wearable and Implantable Body Sensor Networks*, págs. 138–143. IEEE, 2009.

[7] Frank Miskelly. Electronic tracking of patients with dementia and wandering using mobile phone technology. *Age and ageing*, 34(5):497–498, 2005.

[8] Petar Mostarac, Roman Malarić, Marko Jurčević, Hrvoje Hegeduš, Aimé Lay-Ekuakille, y Patrizia Vergallo. System for monitoring and fall detection of patients

using mobile 3-axis accelerometers sensors. En *Medical Measurements and Applications Proceedings (MeMeA), 2011 IEEE International Workshop on*, págs. 456–459. IEEE, 2011.

[9] U.S. Deparment of Health National Institute of Aging, National Institute of Health y Human Services. *Global Health and Aging*. 2011.

[10] World Health Organization. *Global Report on falls Prevention in Older Age*. 2007.

[11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, y E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[12] RS Sangwan, RG Qiu, y D Jessen. Using rfid tags for tracking patients, charts and medical equipment within an integrated health delivery network. En *Proceedings. 2005 IEEE Networking, Sensing and Control, 2005.*, págs. 1070–1074. IEEE, 2005.

[13] Paul Viola y Michael Jones. Rapid object detection using a boosted cascade of simple features. En *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, tomo 1, págs. I–511. IEEE, 2001.