UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

PROYECTO FIN DE CARRERA

Ingeniería de Telecomunicación

DETECCIÓN DE CAÍDAS MEDIANTE VÍDEO-MONITORIZACIÓN

David Dean Pulido

Marzo de 2016

DETECCIÓN DE CAÍDAS MEDIANTE VÍDEO-MONITORIZACIÓN

AUTOR: David Dean Pulido

TUTOR: José M. Martínez

Video Processing and Understanding Lab

Escuela Politécnica Superior

Universidad Autónoma de Madrid

Marzo de 2016

Proyecto parcialmente financiado por el gobierno español bajo el proyecto TEC2014-53176-R (HAVideo)

Resumen

En este Proyecto Fin de Carrera se estudia desarrollar un sistema de detección de caídas mediante video-monitorización, destinado principalmente a su implementación en entornos domésticos para favorecer la vida independiente de las personas mayores. Teniendo en cuenta que las caídas son uno de los principales problemas de la población anciana, nos encontramos dentro un campo con un gran potencial todavía en desarrollo.

En primer lugar se realiza un exhaustivo estudio del arte tanto de los métodos existentes dentro de la detección de caídas en general, así como de los algoritmos exclusivos del análisis de vídeo. Una vez detalladas las técnicas, se elige aquella que mejor se adapta a las necesidades de nuestro proyecto y ofrece unos resultados razonables en la detección de caídas.

A continuación, se ha implementado un algoritmo que caracteriza la técnica elegida. El algoritmo es evaluado mediante un conjunto vídeos de distintas características y varios enfoques que nos permiten establecer conclusiones.

Finalmente, se han enumerado las futuras líneas de trabajo para corregir los problemas existentes en la implementación y crear un algoritmo más robusto y eficiente.

Palabras clave

Detección de caídas, video-monitorización, *dataset*, segmentación frente-fondo, *bounding box*, figura humana, algoritmo de decisión

Abstract

This Master Thesis Project consists on studying the development of a fall detection video-based system, primarily intended for implementation in home environments to promote independent living for the elderly. Due that falls are one of the main problems of the elderly population, we are in a field with great potential still developing.

In the first place, we conducted a comprehensive study of the art of existing methods in the detection of falls in general, as well as exclusive video analysis algorithms. Once detailed the techniques, we chose the one that best fits the needs of our project and provides reasonable results in the detection of falls.

Then, we have implemented an algorithm that characterizes the technique chosen. The algorithm is evaluated by a different set of video with different features and various approaches that allow us to draw conclusions.

Finally, we have listed the future works to correct the problems in the implementation and create a more robust and efficient algorithm.

Key words

Fall detection, vision-based, dataset, foreground-background segmentation, bounding box, human shape, decision algorithm

Agradecimientos

En primer lugar quiero agradecer el Proyecto Fin de Carrera a mi tutor José María Martínez por el tiempo dedicado y toda la ayuda prestada. Sin olvidarme del resto de integrantes del VPU que me han ayudado cuando las cosas se torcían.

También quiero agradecer a amigos y compañeros de la carrera con los que he compartido varios años de clases, prácticas y exámenes.

Sin olvidarme de los amigos de toda la vida con los que comparto aficiones y buenos ratos.

Por último, pero los más importantes, a mis padres y mi novia que siempre han estado ahí desde el principio hasta el final y sin los cuales nada de esto habría sido posible.

David Dean Pulido

Marzo 2016

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE ECUACIONES

1 Introducción

1.1 Motivación

En los últimos años, las caídas en entornos domésticos se han convertido en un problema de salud pública entre la personas mayores. Las estadísticas muestran que las caídas son la causa principal de lesiones relacionadas con la muerte para las personas mayores de 79 años [**1**]. Las caídas son también la principal causa de fracturas de cadera en la población anciana y tiene además un gran impacto psicológico en las víctimas, incluso si no han sufrido ninguna lesión [2]. Si no se toman medidas preventivas para el futuro inmediato, el número de lesiones causadas por caídas se prevé que sea un 100% superior en el año 2030 [3]. En este contexto, es necesario desarrollar un sistema óptimo para solucionar este problema de la sociedad actual.

Un sistema de detección de caídas puede ser definido como un dispositivo de ayuda cuyo principal objetivo es enviar una señal de alarma cuando se produce una caída. Existen multitud de sistemas de detección clasificados según los dispositivos y/o técnicas utilizadas por cada uno [1][3]. Hay que tener en cuenta las desventajas propias de cada sistema a la hora de elegir el sistema adecuado, como pueden ser la necesidad de activación por la persona en cuestión, una instalación compleja o la invasión de la privacidad en el hogar [4]. Por lo tanto debemos comparar y seleccionar la solución que mejor resuelva el problema afectando lo menos posible a la persona y el entorno.

1.2 Objetivos

A continuación, se enumeran y citan brevemente, las fases de trabajo para conseguir el objetivo de este proyecto:

- **Estudio del estado del arte:** Se analizarán los métodos y dispositivos de detección de caídas que existen actualmente, teniendo en cuenta las ventajas y desventajas de cada uno de ellos. Eligiendo una de las aproximaciones para desarrollar la implementación.
- **Análisis exhaustivo del método seleccionado:** Después de la selección del sistema a implementar, se analizará dicho método y las posibles mejoras al mismo.
- **Desarrollo del algoritmo:** Se desarrollará el método seleccionado y se implementarán las mejoras seleccionadas.
- **Evaluación de los resultados:** Se evaluarán los resultados obtenidos para medir la calidad del sistema generado.

Teniendo en cuenta todos estos datos, el objetivo de este Proyecto Final de Carrera será diseñar e implementar un sistema de detección de caídas basado en el procesado de vídeo. Debemos tener en cuenta que el sistema funcionará con caídas simuladas en entornos análogos a uno doméstico, puesto que no se dispone de vídeos de caídas reales para su evaluación.

1.3 Organización de la memoria

La memoria consta de los siguientes capítulos:

- **Capítulo 1: Introducción.** Motivación, objetivos y estructura de la memoria del proyecto.
- **Capítulo 2: Estado del arte.** Estudio de los principales métodos y dispositivos de detección de caídas, enfocados a su aplicación en entornos domésticos. Análisis comparativo de las ventajas y desventajas de cada uno de ellos y recopilación de los *datasets* utilizados para cada implementación analizada.
- **Capítulo 3: Diseño y desarrollo del sistema.** Descripción tanto del sistema a reproducir como del posteriormente desarrollado, detallando las técnicas y herramientas utilizadas para conseguir detectar caídas simuladas.
- **Capítulo 4: Evaluación del sistema.** Evaluación del algoritmo implementado mediante un *dataset* de detección de caídas para verificar el correcto funcionamiento del sistema. Análisis de resultados del método propio con varios enfoques, contrastando los distintos métodos.
- **Capítulo 5: Conclusiones y trabajo futuro.** Conclusiones obtenidas tras el análisis de resultados. Problemas pendientes, posibles mejoras y líneas de trabajo futuras.
- **Referencias y anexos.**

2 Estado del arte

2.1 Introducción

Las caídas son una de las principales causas de lesiones entre la población de edad avanzada. Aproximadamente un tercio de las personas mayores de 75 años son víctimas de caídas en su hogar cada año. Las caídas son también la principal causa de fracturas de cadera en la población anciana y tiene además un gran impacto psicológico en las víctimas, incluso si no han sufrido ninguna lesión [2]. Además existe el riesgo de permanecer tendido en el suelo durante horas o incluso días, lo que puede provocar lesiones importantes como hipotermia.

Para combatir el problema se plantean distintas corrientes. Las primeras basadas en la predicción de caídas, que recogen numerosos datos, los cuales serán utilizados posteriormente para disminuir o evitar las posibles caídas. Este método es útil en entornos asistidos como residencias u hospitales.
Por otro lado, nos encontramos con la detección de caídas, cuyo principal objetivo es informar de la caída a alguien externo. Esta técnica se implementa en entornos domésticos que permite la vida independiente de las persona mayores.
Por estos motivos y por el rápido crecimiento de la población anciana, se ha incrementado notablemente la demanda de sistemas de vigilancia, especialmente para la detección de caídas.

2.2 Clasificación de métodos de detección de caídas

Un sistema de detección de caídas puede ser definido como un dispositivo de ayuda cuyo principal objetivo es enviar una señal de alarma cuando se produce una caída. La estructura de todos los sistemas de detección de caídas es similar, su principal objetivo es diferenciar entre eventos de caída y actividades de la vida diaria. A pesar de esto, existen diferentes enfoques, los cuales vamos a clasificar en tres categorías diferentes [1]:

- Métodos basados en dispositivos portátiles.
- Métodos basados en dispositivos en el entorno.
- Métodos basados en el análisis de vídeo.

A continuación se detallan estos métodos.

2.2.1 Métodos basados en dispositivos portátiles

Este método puede definirse como dispositivos basados en sensores para detectar el movimiento y la localización del cuerpo del portador. La gran mayoría de dispositivos

portátiles están basados en acelerómetros [1][3], cuya principal característica consiste en medir la aceleración del cuerpo o partes del cuerpo.
Los acelerómetros suele llevarlos encima el sujeto, como por ejemplo en su cintura para detectar la caída cuando la aceleración negativa del mismo aumenta repentinamente [1]. Algunas posibilidades de mejora consisten en añadir otro tipo de sensores que complementen al acelerómetro como los barométricos, para medir la presión del aire, o la incorporación de un airbag que se activará al detectar un comportamiento anómalo en la aceleración y velocidad angular del sujeto.

Respuestas fisiológicas tales como la variación de la frecuencia cardíaca o de la presión arterial pueden ser el resultado de la actividad física y los cambios en la posición del cuerpo, por lo que encontraremos otras implementaciones basadas en estas características [1].
La medición de otros factores como los periodos de inactividad y la distinción de posturas nos provee más información sobre características del movimiento como la frecuencia, intensidad y duración del mismo.

También hay que destacar la utilización de acelerómetros tri-axiales diseñados para detectar la aceleración en tres direcciones axiales [1]. Estos sensores se colocan en varias partes del cuerpo lo que nos permite ver si se producen lesiones cuando existe una caída.

Los *smartphones* de hoy en día poseen un amplio conjunto de sensores integrados, como acelerómetros, brújulas digitales o giroscopios. Por esta razón están apareciendo cada vez más investigaciones para desarrollar detectores de caídas en *smartphones* [3].

Figura 2-1: Smartphone con sensor de caída incorporado. Fuente: Google imágenes

2.2.1.1 Ventajas e inconvenientes

La principal ventaja de los dispositivos portátiles reside en su alta rentabilidad puesto que la instalación y funcionamiento no suponen ninguna complicación, a la vez que el coste de los mismos es bastante bajo. Otra ventaja es la alta precisión y localización del individuo que no necesita estar en un área en concreto.
Por el contrario, estos dispositivos necesitan recargar la batería de forma habitual y además

pueden resultar incómodos para la persona. También hay que añadir la dependencia del dispositivo que puede ser olvidado o desconectado en cualquier situación cotidiana por la persona.

En conclusión, encontramos que los dispositivos portátiles son un sistema sencillo y de bajo coste, pero que no resulta favorable para las personas mayores debido a los problemas de fiabilidad y dependencia.

2.2.2 Métodos basados en dispositivos en el entorno

Estos dispositivos buscan fusionar datos de audio y vídeo, así como datos de vibraciones útiles para la detección de caídas.

Una de las ideas principales consiste en crear un puente entre el usuario y la red mediante una placa de nodos inalámbrica [1]. Se detectarán las caídas a través de las funciones de detección de eventos de los nodos. También se crea un sistema de comunicación por voz entre el usuario y el monitor de control que genera una alarma cuando el sistema detecta un problema.

La detección de eventos y cambios mediante datos de vibraciones puede ser muy útil para monitorizar y localizar las caídas. Encontramos un método de detección de caídas mediante la monitorización de los patrones de vibración del suelo, lo que permite diseñar un sistema completamente pasivo y discreto [1]. Estos patrones de vibración generados por una caída son distintos de los generados por actividades cotidianas, como caminar.

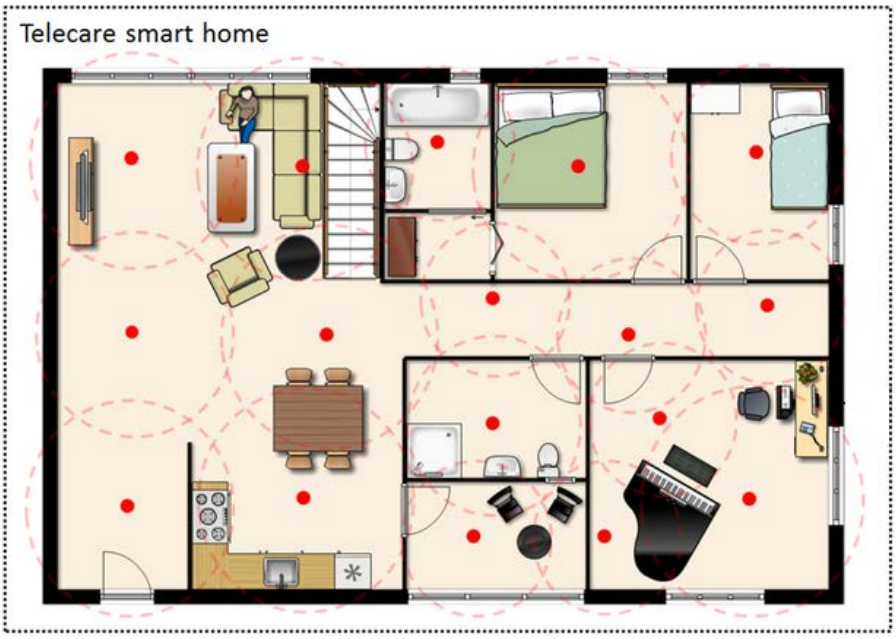

Figura 2-2: Sistema de detección de caídas mediante vibración. Fuente: Google imágenes

2.2.2.1 Ventajas e inconvenientes

La mayoría de los enfoques basados en dispositivos en el entorno utilizan sensores de presión para la detección y seguimiento de caídas, basándose en el principio de alta presión. Es muy rentable y menos intrusivo para la aplicación de sistemas de vigilancia.
Sin embargo, tiene como gran desventaja la generación de falsas alarmas debido a la constante medición de la presión alrededor del sujeto, lo que deriva en una baja precisión del sistema.
En conclusión, vemos que los dispositivos de ambiente son rentables respecto a su coste pero tienen poca fiabilidad y precisión.

2.2.3 Métodos basados en el análisis de vídeo

La incorporación de cámaras está cada vez más presente en los sistemas de asistencia y atención en el hogar que se ocupan de la detección de caídas, puesto que tiene múltiples ventajas sobre los métodos anteriores. Los sistemas con cámaras pueden ser utilizados para la detección de múltiples eventos.

2.2.3.1 Ventajas e inconvenientes

Los sistemas de análisis de vídeo poseen como principales ventajas una baja intrusión, al contrario que los métodos anteriores, y un nivel de interacción bajo entre el sujeto y el sistema puesto que no necesita llevar puesto ningún dispositivo. Además nos encontramos ante un método que centra sus esfuerzos en la utilización de ordenadores estándar y cámaras de bajo coste.

2.2.4 Comparativa de los métodos

Después de describir los distintos métodos, vamos a establecer una comparación entre ellos basándonos en las características principales:

Método	Coste	Intrusión	Precisión	Instalación	Robustez
Dispositivos portátiles	Barato	Sí	Dependiente del escenario	Fácil	No
Dispositivos en el entorno	Barato - Medio	Sí	Dependiente del escenario	Fácil - Media	No
Análisis de vídeo	Medio	Baja - Dependiente	Alta	Media	Sí

Tabla 2-1: Comparativa de los métodos de detección de caídas

Con lo visto en la comparación, parece claro centrarnos en un método de análisis de vídeo puesto que es sin duda el área con mejores características para la detección de caídas. Nos encontramos con el único sistema robusto de entre los comparados, así como una alta precisión. Por ello vamos a estudiar al detalle las técnicas de dicho método, las cuales nos darán diferentes niveles de precisión, intrusión o instalación que nos permitirá encontrar el sistema que mejor se adapte a la situación y/o el sujeto.

2.3 Algoritmos de detección de caídas basados en análisis de vídeo

2.3.1 Introducción

Las caídas son procesos complejos cuyas causas y desarrollo podrían diferir significativamente de unas a otras, pero en general la caída es un evento temporal corto que comienza con una pérdida repentina de equilibrio de la persona y resulta en un impacto con el suelo, pudiendo permanecer en el suelo inmóvil durante un tiempo a causa del impacto de la caída.

Para desarrollar un sistema debemos conocer la estructura que siguen estos algoritmos, suele ser la misma para todos los sistemas basados en análisis de vídeo [1]: en primer lugar se procede a la extracción de fondo para detectar los objetos que se mueven (personas). Cuando se detecta a la persona, procedemos a extraer las características de la misma que nos permitirán detectar cualquier tipo de caída en la etapa inmediatamente posterior. Y por último se generará una alarma que avise del evento detectado.

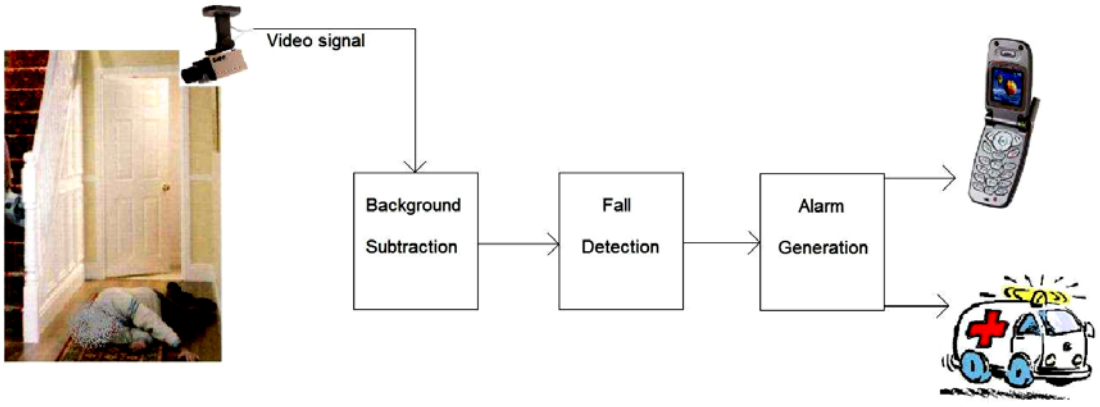

Figura 2-3: Estructura de los algoritmos de análisis de vídeo. Fuente: [1]

2.3.2 Dificultades en la implementación de un sistema de vídeo

Primero de todo, nos encontramos con el tema de la privacidad. Las personas que posean el sistema deben ser plenamente conscientes de que son grabadas y dar permiso para la instalación en su casa. Las personas deben estar bien informadas de que los datos de vídeo no son vistos por otras personas, que sólo se procesan en un ordenador. Sin embargo, las

cámaras grabarán escenas en situaciones privadas, como el baño, puesto que en esta ubicación existe un alto riesgo de caída. Actualmente se están desarrollando métodos que permitan mantener la privacidad ajustable a los deseos del usuario y, al mismo tiempo, producir la suficiente precisión en el sistema [5]. En estos métodos se utilizan diferentes técnicas, que oscurecen intencionadamente la apariencia de la persona para proteger su privacidad, como pueden ser el desenfoque de vídeo o la introducción de una elipse o *bounding box* de un color sólido.

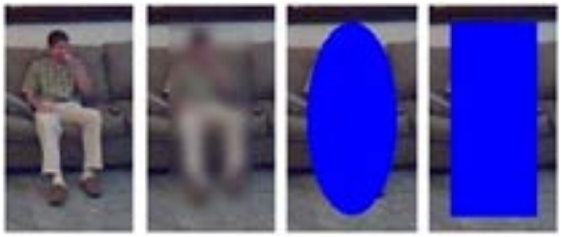

Figura 2-4: Imagen original y las obtenidas para proteger la privacidad. Fuente: [5]

Además de estas cuestiones de privacidad, hay varios desafíos técnicos a tener en cuenta al aplicar un algoritmo de detección de caídas a las imágenes de vídeo [2]:

- Problemas en la iluminación y sombras generadas por fuentes de luz artificial y por luz natural que dificulten el proceso de segmentación de fondo.
- Problemas con la movilidad de muebles y objetos situados en la casa. Debido a ello se necesitará un sistema de extracción de fondo que se actualice lo suficientemente rápido.
- Problemas de oclusión con objetos debido al movimiento de la persona.
- Problemas relacionados con el coste del sistema completo, teniendo sobre todo en cuenta el tipo de cámara o cámaras utilizadas.

2.3.3 Clasificación de las técnicas existentes

Después de la explicación del proceso que siguen los algoritmos basados en el análisis de vídeo y teniendo en cuenta las principales dificultades que nos podemos encontrar en su desarrollo, pasamos a clasificar y explicar cada una de las técnicas:

- Algoritmos 2D.
- Algoritmos 3D.
- Algoritmos 2,5D.

2.3.3.1 Algoritmos 2D

Son los algoritmos más sencillos que utilizan el procesado de vídeo y en los cuales sólo se necesita una cámara para la grabación. Son fáciles de implementar y proporcionan unos resultados bastante razonables, por el contrario aparecen problemas como la oclusión de objetos o movimientos similares a caídas que no siempre se resuelven con éxito.

A continuación detallamos los más utilizados:

2.3.3.1.1 Análisis de la silueta humana

Existen diversas formas de llevar a cabo este tipo de análisis:

- **Bounding box** [1][2][3][5].

  Se trata de una caja rectangular que se dibuja alrededor del sujeto en movimiento y se calcula el ratio entre la altura y anchura del sujeto. Cuando se produce una caída, la *bounding box* cambiará drásticamente sus valores de "x" e "y", por lo tanto su ratio también lo hará.

  Nos encontramos ante un método simple y fácil de implementar. Por el contrario, la precisión de esta técnica depende de la posición relativa de la persona y del campo de visión de la cámara, que puede dar lugar a problemas de oclusión.

- **Elipse alrededor de la silueta humana** [3][4][6][7][8][9].

  Esta técnica consiste en dibujar una elipse que integre la silueta del sujeto, de manera que se ajuste lo máximo a ella. Suele utilizarse conjuntamente con otras técnicas, como el análisis de movimiento [8], para obtener mejores resultados.

  Normalmente se obtienen mejores resultados que con la *bounding box*. Sin embargo, produce fallos en actividades donde se producen cambios bruscos, dando lugar a falsos positivos.

- **Extracción de tres puntos (cabeza, cuerpo y piernas)** [10].

  Esta técnica está basada en la variación de la silueta humana, pero usando solo tres puntos: los centroides correspondientes a cabeza, cuerpo y piernas.

  Una implementación de la técnica consiste en unir mediante dos líneas los tres centroides anteriormente extraídos. Las características extraídas de esas líneas son utilizadas para detectar la caída.

- **Ángulo de caída** [3][7][9][11].

  El ángulo de caída se define como el ángulo entre el suelo y la persona. Aunque dicho ángulo puede variar de persona a persona, una buena estimación serían 45º.

  Este método se suele utilizar como complementario a otras técnicas [9][11] debido a que produce fallos si la persona se cae en dirección a la cámara.

2.3.3.1.2 Cambios de velocidad entre actividades normales y las caídas **[1]**

Esta aproximación se basa en que, en actividades normales, la velocidad de cambio en el sujeto es menor que la velocidad que existe cuando se produce una caída, la cual aumenta rápidamente.

Aparecen problemas de eficiencia en el sistema si el sujeto se acerca a la posición de la cámara, ya que aumenta su velocidad 2D.

2.3.3.2 Algoritmos 3D

Estos algoritmos son más complejos que los anteriores y están formados por un sistema de varias cámaras calibradas que permiten extraer información 3D de la persona, por lo tanto son independientes del punto de vista. Su principal ventaja reside en resolver problemas de oclusión. Sin embargo, exige un proceso cuidadoso y lento de calibración que debe ser repetido cada vez que se mueve intencionada o accidentalmente una cámara del sistema.

Algunos de los más usados son:

2.3.3.2.1 Análisis de la posición 3D de la cabeza **[1]**

El principal objetivo de esta técnica consiste en el seguimiento de la cabeza y la búsqueda de movimientos bruscos en la secuencia de vídeo. Se crea una elipse 3D alrededor de la cabeza a partir de la proyección de elipses 2D de diferentes cámaras, extrayendo varias características.

Estos sistemas obtienen una precisión muy alta en sistemas de vídeo con oclusión. Sin embargo, pierden su efectividad si no se produce una segmentación perfecta previa a la extracción de la posición 3D de la cabeza.

2.3.3.2.2 Utilización de un voxel person **[12]**

Se trata de un método que construye un objeto tridimensional, en concreto una representación de la persona llamada *voxel person*.

El proceso para la obtención del *voxel person* comienza con la división del entorno en regiones discretas, típicamente cubos, llamados elementos de volumen (*voxel*). A continuación cada cámara crea una lista de *voxels* y registra las siluetas correspondientes en ellos. Por último, se reconstruye una única silueta a partir de las obtenidas por cada cámara dando lugar a la *voxel person.*

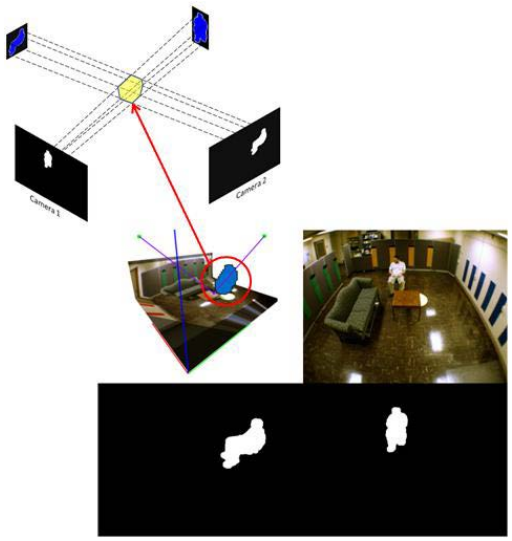

Figura 2-5: Construcción de un *voxel person*

Este tipo de sistema eliminamos los problemas de oclusión. No obstante, es necesaria una costosa calibración y sincronización de todo el sistema de cámaras.

2.3.3.3 Algoritmos 2,5D

Estos algoritmos poseen características de los mencionados anteriormente en dos dimensiones, 2D, y además incorporan la información de la profundidad de la imagen, adquiriendo propiedades de los algoritmos 3D.
Inicialmente existían pocos estudios basados en cámaras de profundidad de imagen debido al alto coste de las mismas, pero este hecho cambió con la aparición de la cámara Kinect que permite obtener unos resultados eficientes a un precio mucho menor.

2.3.3.3.1 Cámara Kinect **[13][14]**

La cámara Kinect está formada por una cámara RGB y un sensor de profundidad de infrarrojos (IR), que obtienen una resolución de 640x480 a 30fps.

Figura 2-6: Partes de una cámara Kinect. Fuente: [14]

Algunas de las técnicas utilizadas son:

- La localización de la posición de la cabeza y el estudio de sus características como la velocidad y su distancia al suelo para detectar una posible caída[13].
- La estimación de una *bounding box* 3D que incluye tres parámetros: altura, anchura y profundidad. Posteriormente se estudia el comportamiento de dicha *bounding box* [14].

Posteriormente apareció una versión mejorada, la cámara Kinect v2. Existen numerosas diferencias con respecto a su antecesora, pero vamos a centrarnos en las más útiles para la detección de caídas:

- **Mayor campo de visión:** aumenta hasta 70º en horizontal (antes 57º) y 60º en vertical (antes 43º).
- **Mayor resolución:** obtenemos una imagen en 1920x1080 Full HD.
- **Mejora en el rango de profundidad del sensor:** el rango de actuación irá desde los 0.5m (antes 0.8m) hasta los 4.5m (antes 4m).
- **Permite calcular y analizar la fuerza de nuestros músculos.**
- **Permite medir el ritmo cardíaco**.

2.4 Datasets

2.4.1 Introducción

Seguidamente vamos a enumerar y mostrar los vídeos que se han utilizado para comprobar las técnicas explicadas anteriormente. Estos *datasets* están formados por un conjunto de vídeos que incluyen situaciones propicias para evaluar el algoritmo, tales como caídas simuladas o acciones cotidianas que pueden dar lugar a falsos positivos. Intentan simular todo tipo de escenarios posibles variando las personas, mobiliario o posiciones de la/s cámara/s.

2.4.2 Datasets disponibles

En la mayoría de las publicaciones no se proporcionan los vídeos utilizados con los que se realizan las pruebas y han sido imposibles de obtener.

En [7] se hace uso de un *dataset* público [**15**] donde nos encontramos con un conjunto de 24 vídeos grabados con 8 cámaras IP de bajo coste, lo que permite la utilización de algoritmos 2D o 3D. En el conjunto de vídeos aparecen algunas de las dificultades anteriormente mencionadas como variaciones en la iluminación, sombras y posible oclusión debido al mobiliario.

Los vídeos están disponibles en:

http://www.iro.umontreal.ca/~labimage/Dataset/

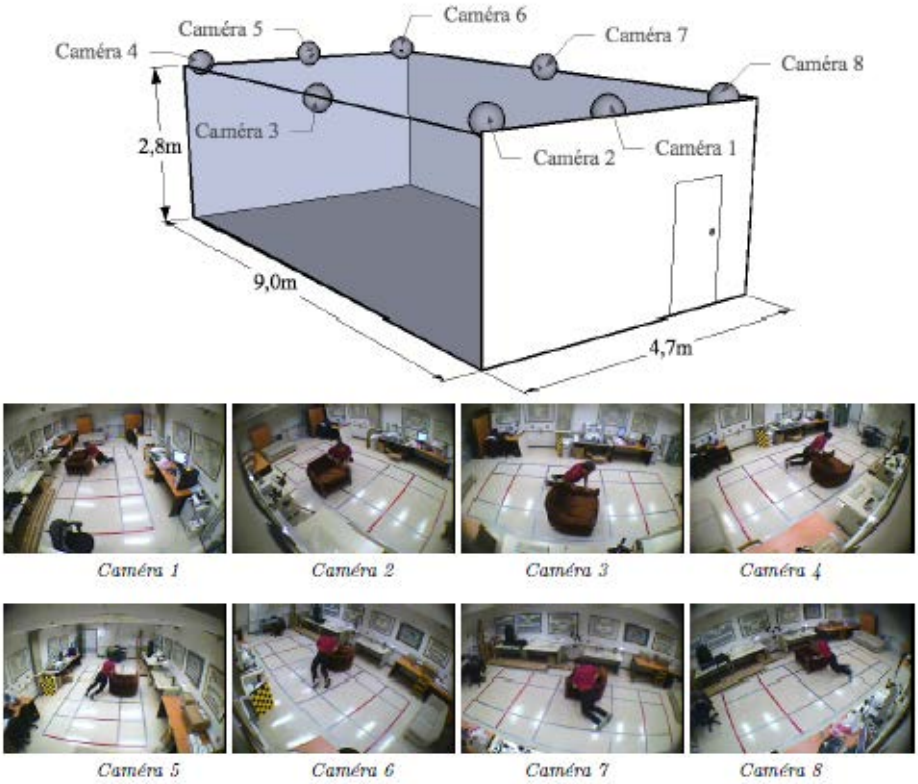

Figura 2-7: Posición de las cámaras y un ejemplo visto por todas ellas. Fuente: [15]

Por su parte, en [5] se utilizan parte de los vídeos localizados en:

http://alumni.cs.ucr.edu/~aedgcomb/3D_2D_head_an_MBR_videos.html

Nos encontramos con un conjunto de 87 vídeos de 1 minuto de duración grabados a 15 fps con una resolución de 352x240. Los vídeos están clasificados según si se produce una caída o no en ellos.

Figura 2-8: Ejemplo de vídeo. Fuente: [5]

Por último, en [10] se pueden descargar directamente los vídeos utilizados en el siguiente enlace:

http://foe.mmu.edu.my/digitalhome/FallVideo.zip

Este *dataset* está formado por un conjunto de 20 vídeos que incluye caídas simuladas, así como situaciones cotidianas. Además, se incorporan diferencias en el vestuario con distintos colores y texturas, que pueden influir en la detección de caídas.

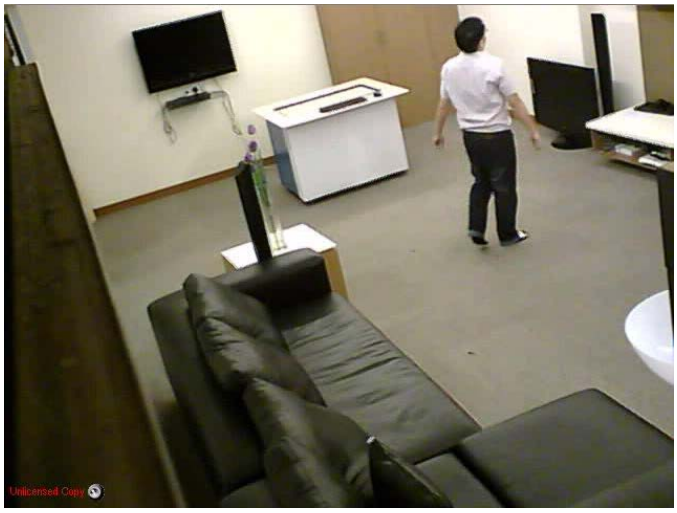

Figura 2-9: Imagen perteneciente al *dataset*. Fuente: [10]

2.5 Conclusiones del estado del arte

Después de la clasificación y estudio exhaustivo de los tres posibles métodos de detección de caídas llegamos a la conclusión de que el análisis por vídeo es el óptimo para nuestro proyecto. Esto se debe a que nos ofrece unas buenas prestaciones, como la precisión o la robustez, sin que ello repercuta en el coste o la instalación. Por contra, debemos tratar con cuidado el tema de la privacidad.

Dentro de este método, nos encontramos con diversas técnicas para llevar a cabo el proceso. Las diferentes técnicas nos ofrecen varias posibilidades dependiendo principalmente de los recursos disponibles y la eficiencia que queremos obtener. Por ejemplo, los sistemas con algoritmos 3D son muy eficaces y precisos, aunque para ello requieren un alto coste material y/o computacional. Sin embargo, los algoritmos 2D nos ofrecen unos valores de eficiencia óptimos con unos costes adecuados a los usuarios.

Finalmente, llegamos a la conclusión de que los algoritmos 2D son los ideales para realizar nuestro proyecto, debido a que queremos desarrollar un sistema formado por una única cámara. Teniendo en cuenta que queremos obtener un sistema eficiente y con una respuesta en tiempo real.

3 Desarrollo del sistema

3.1 Introducción

Como ya hemos visto existen varias técnicas dentro de los algoritmos 2D. En particular, nos centraremos en la técnica explicada en [10], intentando reproducir el algoritmo principal y añadiendo algunas mejoras.

En primer lugar obtenemos los mismos *datasets* utilizados por ellos, los cuales pueden ser descargados directamente como mencionamos en el apartado anterior. Estos vídeos fueron adquiridos mediante una cámara IP no calibrada (Dlink DCS-920) con conexión Wi-Fi en formato MJPEG con una resolución de 320*240 píxeles. El *dataset* está formado por veinte vídeos en color que representa actividades cotidianas como caminar, sentarse, agacharse y ponerse de cuclillas, así como varias caídas simuladas desde diferentes posiciones con respecto a la cámara: hacia atrás, hacia delante o de lado.

A continuación, describiremos de forma genérica el algoritmo principal y diseñaremos nuestro propio algoritmo.

3.2 Descripción general del algoritmo

La forma humana es una de las características utilizadas por muchos algoritmos para detectar una caída mediante el análisis de vídeo. Cuando una persona se cae, la forma humana cambiará rápidamente, mientras que durante las actividades normales la forma humana cambiará lentamente. Este principio es utilizado por el algoritmo de [10].

La forma humana se obtiene mediante la extracción de un número de puntos esenciales que representarán la cabeza, el cuerpo y las piernas. Esta extracción de puntos es más simple y menos complicada en comparación con otras técnicas como la elipse alrededor de la silueta humana. La representación por puntos utilizada proporciona una buena información sobre la orientación y la proporción de altura de la persona.

En primer lugar se lleva a cabo una segmentación de fondo basado en un filtro de mediana. La persona en movimiento es detectada mediante la búsqueda de las diferencias entre los frames entrantes y el modelo de fondo. Se trata de un método con una baja complejidad computacional que proporciona unos resultados eficientes comparados con otros métodos como la mezcla de gaussianas.

Después de haber detectado el primer plano (*foreground*), se representará mediante tres puntos diferentes, los cuales serán los centroides de las tres diferentes regiones del mismo.

A continuación, se forma una *bounding box* que englobe a la persona detectada en el *foreground*, la cual será dividida en tres partes según la proporción 30%, 40% y 30%. Esta división es una estimación inicial de las partes superior, medio e inferior de la persona (cabeza, cuerpo y piernas), que nombraremos como R1, R2 y R3.

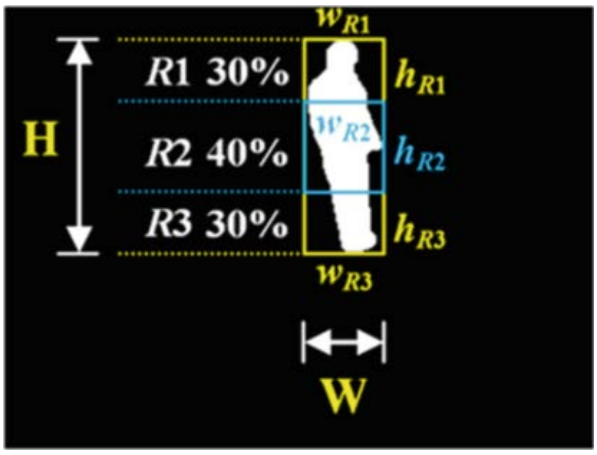

Figura 3-1: División de *la bounding box* del *foreground.* Fuente: [10]

Pasaremos a calcular las alturas, h_{R1}, h_{R2} y h_{R3}, y las anchuras, w_{R1}, w_{R2} y w_{R3}, de cada región según las siguientes ecuaciones:

$$h_{Ri} = \begin{cases} (0.4i - 0.1i^2)H, & si\ H > W \\ H, & en\ el\ resto\ de\ casos \end{cases} \qquad \textbf{Ecuación 3-1}$$

$$W_{Ri} = \begin{cases} W, & si\ H > W \\ (0.4i - 0.1i^2)W, & en\ el\ resto\ de\ casos \end{cases} \qquad \textbf{Ecuación 3-2}$$

Donde i = 1, 2, 3, corresponde a cada región, mientras que H y W son la altura y anchura, respectivamente, totales de la *bounding box*. Obtenidos ya estos datos, podemos pasar a calcular las coordenadas de los centroides como:

$$g_{Rix} = \frac{1}{N_{Ri}} \sum_{l=1}^{N_{Ri}} x_l, \ i = 1, 2, 3 \qquad \textbf{Ecuación 3-3}$$

$$g_{Riy} = \frac{1}{N_{Ri}} \sum_{l=1}^{N_{Ri}} y_l, \ i = 1, 2, 3 \qquad \textbf{Ecuación 3-4}$$

Donde N_{Ri} es el número de píxeles del *foreground* dentro de la región Ri. Los centroides serán los puntos (g_{R1x}, g_{R1y}), (g_{R2x}, g_{R2y}) y (g_{R3x}, g_{R3y}) para las regiones R1, R2 y R3, respectivamente.

A partir de estos tres centroides, se consideran dos líneas, una desde P1 a P2 y otra desde P2 hasta P3, como se muestra en la siguiente figura:

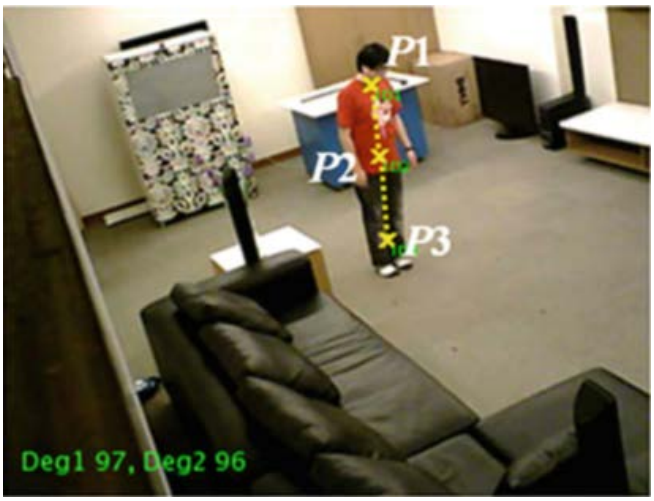

Figura 3-2: Ilustración de los tres centroides y las líneas que los unen. Fuente: [10]

Dado que cada línea representa media porción del *foreground*, cualquier cambio en la distancia y la orientación de las mismas puede indicar un cambio en la forma de la persona. Las distancias y orientaciones de cada línea son calculadas y guardadas para el posterior análisis de forma. Las distancias, D1 y D2, entre los puntos vienen dadas por:

$$\boldsymbol{D1} = \sqrt{(\boldsymbol{g_{R1x}} - \boldsymbol{g_{R2x}})^2 + (\boldsymbol{g_{R1y}} - \boldsymbol{g_{R2y}})^2} \qquad \textbf{Ecuación 3-5}$$

$$\boldsymbol{D2} = \sqrt{(\boldsymbol{g_{R2x}} - \boldsymbol{g_{R3x}})^2 + (\boldsymbol{g_{R2y}} - \boldsymbol{g_{R3y}})^2} \qquad \textbf{Ecuación 3-6}$$

Y el ángulo entre el eje horizontal X y las líneas desde P1 a P2 y desde P2 hasta P3, nos dan información sobre la orientación de las mismas. Se calculan de acuerdo a las siguientes ecuaciones:

$$\boldsymbol{\theta_1} = \mathbf{arctan}\left(\boldsymbol{g_{R1y}} - \boldsymbol{g_{R2y}} / \boldsymbol{g_{R1x}} - \boldsymbol{g_{R2x}}\right) \qquad \textbf{Ecuación 3-7}$$

$$\boldsymbol{\theta_2} = \mathbf{arctan}\left(\boldsymbol{g_{R2y}} - \boldsymbol{g_{R3y}} / \boldsymbol{g_{R2x}} - \boldsymbol{g_{R3x}}\right) \qquad \textbf{Ecuación 3-8}$$

Donde θ_1 es la orientación de la línea formada por P1 y P2, y θ_2 es la orientación de la línea formada desde P2 hasta P3.

Con todas estas características obtenidas pasamos a la parte principal de detección de caídas. Esta aproximación está basada en el hecho de que los diferentes cambios en la postura de la persona, como pasar de estar de pie a estar sentado o caerse, tendrán diferentes cambios en la parte superior y la parte inferior del cuerpo humano. Al analizar el cambio de forma en estas dos partes de una persona, podemos distinguir una caída de las actividades diarias. Para ello, calculamos el ratio entre las dos distancias, $p = D1 / D2$, y la diferencia entre las orientaciones de las líneas, θ_1 y θ_2.

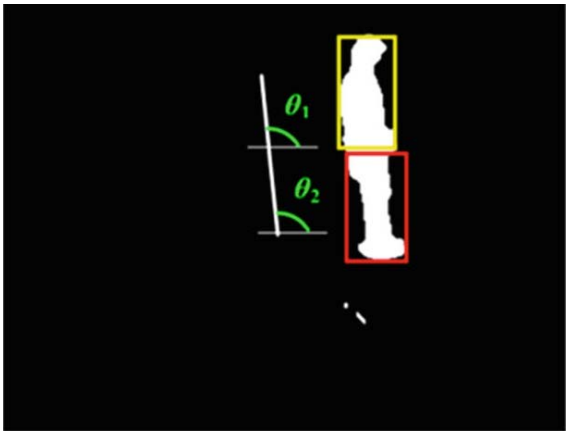

Figura 3-3: Ejemplo de orientaciones. Fuente: [10]

Como la forma de la parte superior e inferior de la persona apenas sufre cambios durante las actividades diarias, el valor del ratio *p* será 1 la mayor parte del tiempo. Por lo tanto, se considera que hay una posible caída, si *p* cambia repentinamente a un valor distinto de 1.

Por otro lado, las partes superior e inferior tendrán orientaciones similares en posiciones de estar de pie y estar tumbado. Cuando la forma de la persona cambie de estar de pie a tumbado, existirá una pequeña variación en las orientaciones de las líneas justo antes y después de la caída. Por lo tanto, se comprueba la diferencia entre θ_1 y θ_2 para cada *frame*.

Teniendo en cuenta estas consideraciones, se compara el ratio entre las distancias del *frame* anterior, p_{t-1}, con él del *frame* actual, p_t, llegando a las siguientes opciones:

1. $\Delta p = 0$, indica que no existe una posible caída y las orientaciones son similares. Se guarda θ_1 como ángulo de referencia, θ_r, y la suma de D1 y D2 se guarda como longitud de referencia, D_r.

2. $\Delta p > 0$, señala una posible caída. Se buscan las orientaciones, θ_1 y θ_2, en el décimo *frame* posterior a la posible caída y se guardan como θ_{N1} y θ_{N2}, respectivamente. El valor de diez *frames* puede variar para ajustarse a las condiciones de otros vídeos. A continuación, se calcula la diferencia entre las nuevas orientaciones, θ_{N1} y θ_{N2}, y el ángulo de referencia, θ_r:

$$\boldsymbol{\theta_{D1} = |\theta_{N1} - \theta_r|} \qquad \textbf{Ecuación 3-9}$$

$$\boldsymbol{\theta_{D2} = |\theta_{N2} - \theta_r|} \qquad \textbf{Ecuación 3-10}$$

Para solucionar el problema de mapear las dos porciones del cuerpo humano antes y después de una caída, se calcula la media entre las dos diferencias de orientación, μ_0. Detectaremos una caída si $\mu_0 > 30^o$.

Sin embargo, existen caídas donde el valor de μ_0 puede ser menor o igual a 30^o. En estas condiciones, comprobamos la diferencia entre la longitud total de la línea en el décimo *frame*, después de una posible caída, y la longitud de referencia, D_r. Basándose en las observaciones, la altura de la persona se reducirá drásticamente durante una caída en el campo de visión de la cámara. El cambio en el sumatorio de las longitudes de las líneas tras una posible caída, D_{diff}, viene dado por:

$$\boldsymbol{D_{diff} = |D_r - (D1 + D2)|} \qquad \textbf{Ecuación 3-11}$$

Una caída es detectada si D_{diff} es mayor que el 40% de la longitud de referencia, es decir 0.40*(D_r) pixeles. Por el contrario, si es menor el sistema considerará que no se ha producido caída de la persona.

Por último, se tendrá en cuenta que una caída terminará con un período de inactividad donde la persona permanece inmóvil o inconsciente. La última verificación de esta técnica consiste en comprobar si hay algún movimiento de la persona después de una posible caída. Por lo tanto, una caída se confirma si se cumple la siguiente condición:

"El cambio en la distancia recorrida por el centroide, P2, es menor o igual a 5 píxeles durante 5 segundos."

La duración del período de inactividad, en este caso 5 segundos, se puede variar para asegurar que la persona está completamente inconsciente después de la caída.

3.3 Diseño del sistema

3.3.1 Introducción

El diseño de nuestro sistema se ha fundamentado en el explicado anteriormente [10]. Además se ha tomado como referencia el TFG del departamento [16], que se basa en el mismo algoritmo.

La principal diferencia es que nuestro sistema se ha implementado en el lenguaje de programación *C++* utilizando la librería *OpenCV*, mientras que los anteriores se realizaban en *Matlab.* Además, se han incorporado mejoras con respecto al desarrollado en [16], que incluyen entre otras un análisis más exhaustivo de las secuencias de vídeo y la reducción del tiempo total utilizado por el sistema.

OpenCV (Open Source Computer Vision) fue lanzado bajo una licencia BSD y por lo tanto, es gratis para su uso académico o comercial. Cuenta con interfaces de C++, C, Python y Java, siendo compatible con distintos sistemas operativos como Windows, Linux, Mac OS, iOs y Android. *OpenCV* fue diseñado para mejorar la eficiencia computacional y con un enfoque en aplicaciones en tiempo real, lo cual la hace ideal para nuestro proyecto.

Para el desarrollo del algoritmo se ha trabajado en etapas consecutivas, asegurando el correcto funcionamiento de cada una para avanzar a la siguiente.

Las etapas siguen la siguiente estructura:

- Lectura del vídeo y obtención de características básicas como número de *frames* totales y *frame* actual.
- Realización de un modelado de fondo para obtener el *background*, y posteriormente el *foreground* sobre el que trabajar. Este proceso se realiza de forma independiente al diseño original, mediante un método básico que nos ofrece las mismas características.
- Obtención de la *bounding box* del *foreground*.
- Extracción de los parámetros necesarios para el posterior análisis de la forma humana. Esta etapa se ha implementado siguiendo fielmente los pasos del algoritmo original.
- Implementación del algoritmo de decisión, que consiste en determinar el sistema y los valores que definen si se ha producido una caída o no. Este proceso se ha realizado siguiendo el mismo esquema aunque se han introducido variaciones para mejorar el funcionamiento y obtener mejores resultados.

A continuación, detallamos las etapas anteriores en profundidad.

3.3.2 Lectura de vídeo

En esta etapa de inicialización se captura el vídeo perteneciente al *dataset* y se obtienen características básicas del mismo que serán útiles en las próximas etapas.
Para realizar el proceso se hace uso de la clase *VideoCapture* que se encarga de capturar el archivo de vídeo. Posteriormente, se obtienen los *fps* (*frames* por segundo) del vídeo, muy útiles para establecer el tiempo de espera entre las imágenes, el número de *frames* totales del vídeo y el número de *frame* actual, necesario para realizar el análisis de vídeo.

3.3.3 Modelado de fondo y segmentación frente-fondo

3.3.3.1 Obtención del background

Esta etapa podría considerarse la más importante de todo el sistema puesto que marcará el comportamiento y la precisión del mismo. Es decir, una obtención errónea del *background*, y posteriormente del *foreground*, provocará fallos en el cálculo de los parámetros y el algoritmo de decisión será inútil para la detección de caídas. Por lo tanto, el objetivo de este bloque es determinar una imagen de fondo que no incluya al sujeto o persona.

Como se ha mencionado anteriormente, en [10] y en [16] se utiliza un filtro de mediana para obtener el *background*, que en nuestro caso se simplifica utilizando un sistema de menor complejidad pero que obtiene unos resultados igualmente eficientes.
Teniendo en cuenta el arranque en ”frío” de todas las secuencias de vídeo, es decir, en la escena de vídeo no aparece un sujeto al inicio del mismo y tampoco se produce ningún cambio de posición de objetos durante el vídeo, obtenemos el *background* asignándole la primera imagen de cada vídeo. Planteamos esta simplificación puesto que el objetivo principal del proyecto era migrar a lenguaje *C++* y mejorar el algoritmo de decisión.

Figura 3-4: *Background* en escala de grises

Convertiremos el *background* a escala de grises mediante la sencilla función *cvtColor*, que nos permitirá realizar posteriormente la segmentación frente-fondo.

3.3.3.2 Segmentación frente-fondo

En la siguiente etapa se lleva a cabo la segmentación que permite extraer a la persona del vídeo de entrada, la cual podemos afirmar que es aquello que está en movimiento durante la secuencia de vídeo.

Una vez hemos obtenido el *background*, podemos calcular el *foreground* mediante la imagen diferencia que se va obteniendo al leer las siguientes (en escala de grises) e ir restándolas consecutivamente con el *background*.
Con el *foreground* calculado, pasamos a detectar si existe persona en la secuencia de vídeo. Para ello creamos la función propia *calcularPorcentajeNegro*, cuyo objetivo es evaluar el porcentaje de píxeles con valores mínimos, tomando en nuestro caso los veinte primeros valores respecto al total del *foreground* en escala de grises. Para ello, utilizamos la función *calcHist* que nos cuantifica de forma sencilla los píxeles teniendo en cuenta un rango de 0 a 255, siendo 0 el valor mínimo (negro) y 255 el máximo (blanco). Llegando a la conclusión de que existe movimiento en la imagen, es decir, hemos localizado a la persona en la secuencia de vídeo, si el conjunto de píxeles con valores mínimos es menor que un umbral muy alto, fijado experimentalmente en nuestro caso a 99%.

Figura 3-5: Imagen diferencia

Después de obtener la imagen diferencia procedemos a umbralizarla, para obtener una imagen bimodal que nos permitirá distinguir el *background* de la persona en movimiento. Para realizar la umbralización se utiliza el método Otsu, al igual que en [16], que nos ofrece unos buenos resultados, rapidez y eficacia computacional.

Debemos tener en cuenta que la umbralización debe hacerse sólo cuando el *foreground* no sea nulo, es decir, hemos localizado al sujeto o persona. De lo contrario, el método Otsu umbralizaría el ruido de la imagen, lo cual nos daría problemas en las siguientes etapas. Por ello, resulta muy importante la evaluación del porcentaje de negro anterior.

El siguiente paso consiste en aplicar operadores morfológicos a la imagen binaria para conseguir una mejor definición de la silueta humana y eliminar el ruido de la imagen.
En nuestro caso, realizamos una operación de apertura, que consiste en una primera erosión, que eliminará el ruido indeseado de la imagen diferencia, y posteriormente, una doble dilatación para recuperar el tamaño de la silueta de la persona.

Figura 3-6: *Foreground* procesado

Para estos dos últimos pasos, umbralización y operadores morfológicos, se implementa una función propia llamada *umbralizacionOtsu* donde se utiliza la función *threshold* que nos permite utilizar varias clases de umbralización directamente sobre la imagen indicando los valores máximo y mínimo, 0 y 255 respectivamente, como el método Otsu. Seguidamente aplicamos las funciones *erode* y *dilate* con un cuadrado 3x3 como elemento estructurante.

3.3.4 Bounding box

Después de la localización de la persona en la imagen, procedemos a crear una *bounding box* que la englobe, teniendo en cuenta la altura y anchura de la persona.

Este proceso, inicialmente, parece uno de los más sencillos puesto que existen funciones definidas que calculan la *bounding box*, pero existe un procesado anterior para evitar problemas en la creación de la misma. Estos problemas van desde la creación de varias *bounding boxes*, una por cada contorno, hasta cortes en las mismas por problemas de oclusión.

En nuestro proyecto se ha creado una función propia denominada *calcularBoundingBox* donde hemos seguido los siguientes pasos: primero se han encontrado todos los contornos existentes en el *foreground* con *findContours*; posteriormente se han ordenado todos los contornos próximos mediante la función *approxPolyDP* para evitar problemas con pequeños huecos existentes en la silueta de la persona debido al vestuario o ligeros problemas de oclusión; a continuación, se han unido los contornos en un solo vector y se ha encontrado el "casco" exterior de todos ellos utilizando una función muy útil llamada *convexHull*; y, por último, se ha empleado la función *boundingRect* que devuelve el rectángulo mínimo de un conjunto de puntos.

Finalmente se dibuja la *bounding box* en la imagen de salida del algoritmo, siendo de color verde cuando la persona está en estado normal y de color rojo cuando se produce una caída.

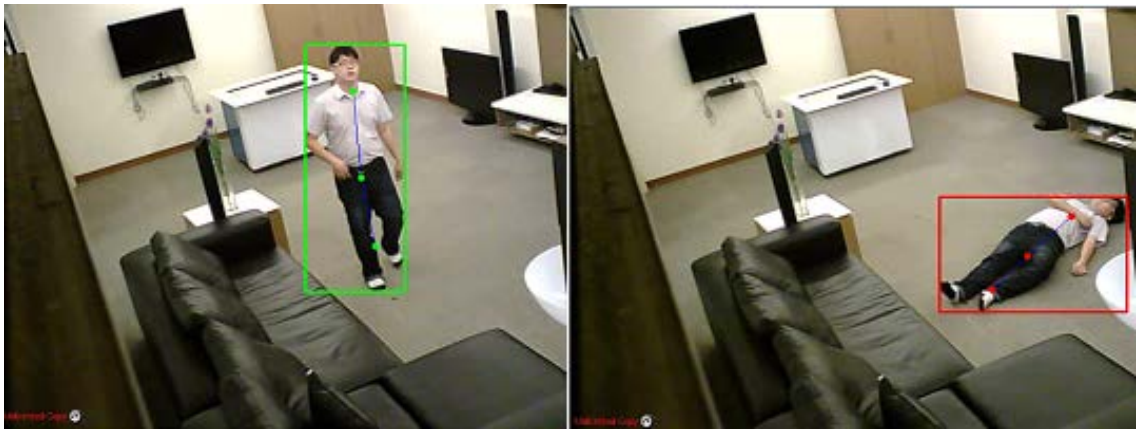

Figura 3-7: *Bounding Box* en función del estado de caída

3.3.5 Extracción de parámetros

En esta etapa procedemos a extraer los parámetros que van a caracterizar a la persona. Para ello, vamos a tener en cuenta la representación basada en tres puntos (cabeza, cuerpo y piernas) descrita en [10].

El primer paso de este bloque consiste en establecer las regiones R1, R2 y R3, a partir de las coordenadas de la *bounding box* ya obtenida. A continuación, establecemos dichas regiones en la imagen diferencia, *foreground*.
En el siguiente paso utilizamos la función *moments* en cada región, que nos devuelve todos los momentos de la figura de la imagen:

$$[m_{00}, m_{10}, m_{01}, m_{20}, m_{11}, m_{02}, m_{30}, m_{21}, m_{12}, m_{03}]$$

Obteniendo los centroides con las siguientes ecuaciones:

$$g_{Rix} = \frac{m_{10}}{m_{00}} \qquad \textbf{Ecuación 3-12}$$

$$g_{Riy} = \frac{m_{01}}{m_{00}} \qquad \textbf{Ecuación 3-13}$$

Donde i = 1, 2, 3 representa la región en la que nos encontramos.
Una vez obtenidos los valores de los centroides es necesario reajustar su posición para que quede con respecto a la imagen original, ya que inicialmente se han calculado con respecto a cada región por separado. Este reajuste nos permite visualizar los centroides correctamente sobre la imagen aunque no influye en el posterior cálculo de distancias y ángulos, él cual es independiente de la posición de los centroides con respecto a la imagen. Todo este proceso se incluye dentro de la función propia *extraerCentroides*.

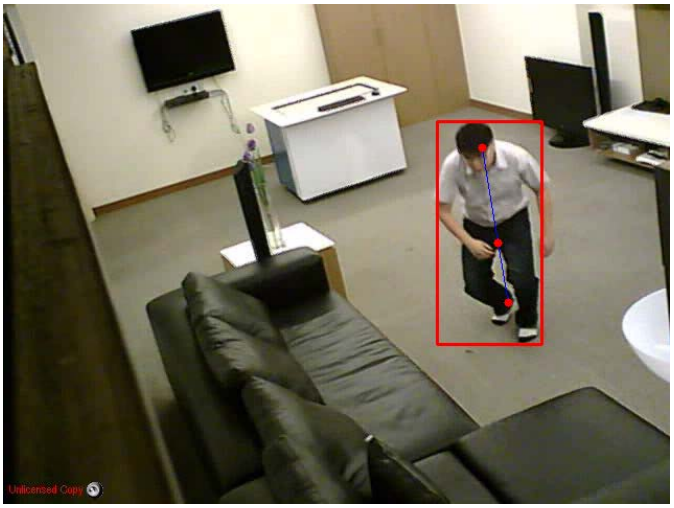

Figura 3-8: *Frame* con los parámetros

Obtenidos los tres centroides creamos las dos líneas que los unen, D1 y D2, y calculamos su longitud y el ángulo que forman con la horizontal. Finalmente, creamos dos vectores para guardar todos los parámetros obtenidos y poder acceder con facilidad en la siguiente etapa del proceso. Los vectores contienen:

- Frame_total = [número de *frame*, D1, D2, θ_1, θ_2]
- Centroides = [g_{R1}, g_{R2}, g_{R3}]

3.3.6 Algoritmo de decisión

Esta etapa analiza los valores de los parámetros obtenidos para detectar una posible caída. Para ello, vamos a explicar detalladamente cada paso realizado en el diseño del algoritmo.

En primer lugar creamos una variable *t* que nos permitirá movernos por la secuencia de imágenes en el futuro y evitar problemas de sincronización entre el algoritmo y el vídeo. Además, nombramos un nuevo vector llamado *estado_caida* en el que almacenaremos los distintos estados de caída que se detecten el vídeo, en su correspondiente *frame*. Los estados definidos son:

Estado de caída	Descripción
-1	No persona
0	Persona en estado normal; no caída
1	Falsa alarma
2	Cayéndose; caída
3	Levantándose
4	Caído e inmóvil -> Alarma

Tabla 3-1: Valores definidos para los distintos estados de caída

Iniciamos un bucle para analizar todos los *frames* de la secuencia de vídeo, donde las primeras comprobaciones son que el *frame* actual sea menor que el número total de *frames* del vídeo y que coincida nuestra variable *t* con el *frame* actual. Estas comprobaciones son necesarias para permitir el correcto funcionamiento del programa debido a los numerosos movimientos en el bucle para obtener características de anteriores y posteriores *frames*.

A continuación, obtenemos los parámetros del *frame* actual y anterior, es decir en *t* y *t-1*, del vector *frame_total*. Con estos valores calculamos los ratios descritos en el algoritmo original, p_t y p_{t-1}, que nos permiten calcular la variación que se ha producido de un *frame* a otro, Δp.
Partiendo de un estado de no caída (0), se comprueba si Δp supera un cierto valor *epsilon*, que indicará la existencia de un posible cambio o caída. El valor épsilon se ha fijado heurísticamente a 0.35, después de utilizar otros valores y comprobar que éste permite un mejor funcionamiento del sistema.
Además, calculamos la longitud de referencia y orientación de la *frame* actual, D_r y θ_r respectivamente.

En caso de cumplirse la condición, acudimos a un cierto número de *frames* posteriores al actual mediante la variable *Tsalto*, que establecemos en 10 siguiendo el algoritmo de [10]. Para ello, utilizamos la función de creación propia *extraccionParametros*, que tiene implementado todo el análisis de esta sección, para obtener los parámetros del *frame* indicado, *t + Tsalto*. Esto es necesario puesto que el *frame* que buscamos todavía no ha sido analizado y, por tanto, no disponemos de los valores en el vector *frame_total*.
Una vez obtenidos los parámetros, se restan las orientaciones para el décimo *frame* de las guardadas anteriormente como referencia y se calcula la media de ambos resultados. Si este valor es mayor que 30º, idéntico al sistema original, significará que existe una posible caída (2). En caso contrario, puesto que existen caídas que no cumplan esta condición, se calcula una diferencia, D_{diff}, entre la longitud de referencia y la longitud de la línea en el décimo *frame* tras la posible caída. Si este valor supera el 40% de la longitud de referencia indicará que la persona puede haber sufrido una caída (2) y, por el contrario, si es inferior estaremos ante una falsa alarma (1).

Para confirmar que se ha producido una caída creamos la función *checkInactivity*. Esta función comprueba el movimiento del centroide g_{R2} durante los 5 segundos posteriores a la caída, siguiendo el mismo proceso que en [10] y [16]. Teniendo en cuenta que los vídeos están grabados a 30 *fps*, esos 5 segundos posteriores son equivalentes a un salto de 150 *frames*, que denominamos *TsaltoTipo*. Por ello, calculamos la distancia euclídea existente del centroide del cuerpo, g_{R2}, entre ambos instantes. Si esta distancia es mayor que 5 píxeles significará que la persona está levantándose o levantada (3) y, si no, indicará que la persona permanece inmóvil (4).

Este último caso, activaría una alarma de caída tan sólo 5 segundos después del inicio de la misma obteniendo una detección de caída en tiempo real. Esta característica cumple uno de los objetivos principales del proyecto.

Con la parte principal del algoritmo de decisión implementado, pasamos a explicar algunos casos en particular para su correcto funcionamiento.
En primer lugar, debemos tener en cuenta los 150 *frames* posteriores a la posible caída, puesto que la persona se puede levantar en ese período y volver a caerse dando lugar a errores posteriores. Para ello, desarrollamos un bucle que compruebe que la persona está levantándose durante esos 150 *frames*. Seguimos los siguientes pasos:

- Inicializamos a cero un contador denominado *contador_levantado*.
- Calculamos la diferencia de orientaciones en el *frame* actual, posterior a la caída.
- Se imponen las siguientes condiciones: la diferencia de orientaciones debe ser inferior a 15º, obtenido experimentalmente, cuando se empieza a levantar la persona; la altura de la *bounding box* debe ser estrictamente mayor que su anchura; y las orientaciones deben encontrarse ambas entre 60º y 120º, puesto que en caso contrario indicará que la persona todavía se encuentra tendida en el suelo.
- Si se cumplen todas las condiciones se aumenta *contador_levantado* y se comprueban las dos primeras condiciones nuevamente, siendo ahora la diferencia de orientaciones menos restrictiva, 29º.
- Cuando se cumplen todas las condiciones durante cinco *frames* consecutivos se considera que la persona se ha levantado completamente y se procede a analizar el siguiente *frame* con normalidad. En caso contrario, se reinicia *contador_levantado* y se vuelve a repetir el proceso.

Otro caso a tener en cuenta aparece cuando al utilizar la función *checkInactivity,* el salto de 150 *frames* excede el número de *frames* totales de la secuencia de vídeo. En este caso no existiría llamada a dicha función puesto que no se podrían obtener los parámetros necesarios, pero si funcionaría el último bloque descrito, el cual detectaría que la persona se levanta en ese espacio de tiempo.

Para finalizar el algoritmo implementamos una variable *estabilidad* que nos permitirá eliminar espurios. A la variable le asignamos el valor 5, que indica el mínimo de veces que se tiene que repetir un estado para aplicarse un cambio, de lo contrario se mantiene el estado anterior. Como excepción incluimos que la persona se levanta después de producirse una caída, es decir, una secuencia de levantado (3) tiene que venir precedida obligatoriamente de una secuencia de cayéndose o caída (2). Estos resultados se guardan en un vector llamado *estado_estabilidad* sobre el que realizaremos las distintas evaluaciones de algoritmo.

El sistema completo trabaja en modo *on-line* con un ligero retardo. Esto implica que a cada *frame* se le asigna su resultado al instante siempre que no se produzca un cambio de

estado, en cuyo caso, se deberá esperar tantos *frames* como marque la variable estabilidad, 5 en nuestro caso.

3.4 Diferencias en la implementación

En este apartado vamos a describir las diferencias de implementación respecto a los sistemas de [10] y [16].

En primer lugar se ha introducido una obtención del *background* distinta a ambos sistemas. Se ha sustituido el anterior filtro de medianas por un sistema más simplificado que obtiene el primer *frame* de la secuencia de vídeo. Este proceso es una primera aproximación práctica puesto que el sistema estará siempre encendido y es posible gracias a la naturaleza de los vídeos de este *dataset* en concreto. Por el contrario, este método no puede utilizarse en vídeos donde la persona se encuentre inicialmente en la secuencia de vídeo, lo que se conoce como arranque en “caliente”.

Además, introducimos una diferencia muy importante en el algoritmo de decisión con respecto al implementado en [16]. En dicho sistema cuando se detecta una caída se procede a evaluar el estado de la persona avanzado un período de tiempo, 5s para este caso, dejando los *frames* intermedios sin analizar. En nuestro caso realizamos una comparación similar para comprobar si la persona sigue inmóvil pero continuamos con el análisis de los *frames* intermedios. Para ello, incluimos el estado levantándose (3) que nos indica exactamente cuando la persona vuelve a una posición normal después de la caída.
En definitiva se produce un análisis más exhaustivo, puesto que se evalúan todos los píxeles, y se amplía la detección a un mayor número de estados.

4 Evaluación

4.1 Introducción

En este apartado vamos a analizar los resultados obtenidos por nuestro algoritmo, para ello realizaremos una serie de pruebas con distintos vídeos. En nuestro caso, se realizarán las pruebas con el *dataset* disponible en [15], que es el mismo utilizado por [10] y [16] para sus respectivas evaluaciones. Destacar que el *dataset* está formado por un conjunto de veinte vídeos que incluyen caídas simuladas desde distintos ángulos y posiciones, actividades cotidianas que pueden ser confundidas por el algoritmo como una caída y entradas y salidas del campo de visión de la cámara por diferentes zonas.

Se proporcionarán los resultados obtenidos tras las pruebas, buscando conseguir un sistema de detección de caídas en tiempo real con la mayor precisión y fiabilidad posibles.

4.2 Metodología

A continuación, se explica paso a paso la metodología seguida durante evaluación para obtener unos resultados reales y fiables.

4.2.1 Ground truth

El término *ground truth* es un término utilizado para hacer referencia al conjunto de datos proporcionados por la observación directa, datos objetivos, necesarios para verificar la precisión del conjunto de entrenamiento disponible.
El *dataset* disponible no lleva asociado ningún *ground truth*, pero se ha tomado como referencia él creado en [16] para la mayoría de los vídeos y ampliándolo para los vídeos que no estaba disponible. En ambos casos, el *ground truth* se ha creado anotando cada uno de los *frames* de la secuencia de vídeo junto con su estado de caída observado experimentalmente. Para ello se ha tenido en cuenta la siguiente tabla de valores:

Estado de caída	Descripción
-1	No persona
0	Persona en estado normal; no caída
2	Cayéndose; caída
3	Caído
4	Levantándose

Tabla 4-1: Valores de los estados de caída en el *ground truth*. Fuente: [16]

4.2.2 Matching

Existen diferentes valores entre ambas tablas, la perteneciente al algoritmo y la perteneciente al *ground truth*. Por este motivo es necesario realizar un *matching* o equivalencia entre los valores que difieren entre el estado observado al crear el *ground truth* y el determinado por el algoritmo pero que corresponden a una misma acción.

En primer lugar el valor (1) del algoritmo que indicaba falsa alarma, ha desaparecido del *ground truth* puesto que a simple vista no se aprecia esa posibilidad. Otro caso consiste en el valor (2) del algoritmo, que indica si se está produciendo una caída, el cual es equivalente a los estados (2) y (3) del *ground truth* puesto que no se diferencia entre cayéndose o caído en el algoritmo. Por último, encontramos que el valor (4) de nuestro algoritmo, persona inmóvil durante 5 segundos después de una caída, no tiene ninguna correspondencia en el *ground truth*, esto se debe a que en la observación de la secuencia de vídeo no se tiene en cuenta el tiempo que la persona permanece inmóvil puesto que este valor de tiempo puede variar de una implementación a otra.

Análisis	*GT*
-1; No persona	**-1; No persona**
0; Persona en estado normal	**0; Persona en estado normal**
1; Falsa alarma	**No existe**
2; Caída	**2; Cayéndose**
	3; Caído
3; Levantándose	**4; Levantándose**

Tabla 4-2: Resumen del *matching* establecido

Después de realizar el *matching*, procedemos a comparar ambos resultados en función del estado caída.

4.2.3 Scores

En esta sección vamos a clasificar las distintas combinaciones entre los resultados del algoritmo y los asignados en el *ground truth*, que nos proporcionarán uno de los siguientes *scores* o resultados:

- ***True Positive***: Se considera TP cuando el sistema detecta caída y acierta.
- ***True Negative***: Se considera TN cuando el sistema detecta no caída y acierta.
- ***False Positive***: Se considera FP cuando el sistema detecta caída y se equivoca.
- ***False Negative***: Se considera FN cuando el sistema detecta no caída y se equivoca.

	Classified as A	Classified as not A
Is A	TP	FN
Is not A	FP	TN

TP – True Positive; FN – False Negative
FP – False Positive; TN – True Negative

Tabla 4-3: Tabla de *scores* utilizados. Fuente: Google imágenes

En nuestro caso tenemos dos estados: "caída" y "no caída", y debemos asignar los valores obtenidos del algoritmo y el *ground truth* a dichos estados. Por lo tanto consideramos que:

- En el *ground truth*: el estado "no caída" corresponde a no persona (-1) y persona en estado normal (0); mientras que el estado "caída" corresponde a cayéndose (2), caído (3) y levantándose (4).
- En el análisis del algoritmo: el estado "no caída" se corresponde con los valores de no persona (-1) y persona en estado normal (0); por otro lado, el estado "caída" incluye cayéndose (2) y levantándose (3).

Mostramos las distintas combinaciones existentes entre el análisis y el *ground truth* en el siguiente gráfico, permitiendo una mejor comprensión del sistema de *matching* y *score* utilizados.

Análisis / GT	No persona (-1)	Est. Normal (0)	Caído (2)	Levantándose (3)
No persona (-1)	TN	TN	FP	FP
Est. Normal (0)	TN	TN	FP	FP
Cayéndose (2)	FN	FN	TP	TP
Caído (3)	FN	FN	TP	TP
Levantándose (4)	FN	FN	TP	TP

Tabla 4-4: *Scores* posibles entre el *ground truth* y el análisis

4.2.4 Métrica

Con todas las consideraciones anteriores debemos tener en cuenta una serie de fórmulas que nos permiten medir la eficiencia y rendimiento de nuestro sistema. Estas fórmulas nos permitirán comparar el resultado de nuestro sistema equitativamente con otras implementaciones, incluso con otros algoritmos de detección de caídas.

Las fórmulas utilizadas son:

$$\boldsymbol{Precision} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FP}} \qquad \textbf{Ecuación 4-1}$$

$$\boldsymbol{Recall = \frac{TP}{TP+FN}}$$ **Ecuación 4-2**

$$\boldsymbol{True\ negative\ rate = \frac{TN}{TN+FP}}$$ **Ecuación 4-3**

$$\boldsymbol{False\ positive\ rate = \frac{FP}{FP+TN}}$$ **Ecuación 4-4**

$$\boldsymbol{Accuracy = \frac{TP+TN}{TP+FP+TN+FN}}$$ **Ecuación 4-5**

Todas las medidas ofrecen valores entre 0 y 1, representando los porcentajes correspondientes.
Tienen especial importancia los valores obtenidos en *precision* y *recall*, puesto que determinan que un valor clasificado como "caída" sea realmente una "caída". Indicando valores próximos al 1, un buen funcionamiento del sistema.
Por otro lado, las fórmulas de *true negative rate* y *false positive rate* nos aportan información adicional. La primera de ellas mide la proporción de eventos de "no caída" identificados correctamente y la segunda nos señala el porcentaje de falsas caídas cuantificadas en el análisis, siendo los valores más próximos al 0 los que indican un mejor análisis.
Por último, tenemos el valor de *accuracy* que nos señala la exactitud global del sistema puesto que tiene en cuenta todos los posibles scores, obteniendo todos los valores acertados entre todos los casos posibles. Un valor más cercano a 1 nos indicará un mejor funcionamiento del sistema.

4.3 Resultados

4.3.1 Resultados de la métrica

Se va a proceder a obtener el rendimiento del algoritmo implementado para la mayoría de vídeos del *dataset*. Únicamente se descartan los vídeos 19 y 20 porque incluyen la simulación de presencia de mascotas en la secuencia de vídeo, para lo cual no está preparado nuestro sistema.

Dentro de los resultados obtenidos incluimos la opción NA (No aplica) para algunos casos especiales. Por ejemplo, en vídeos donde no existe una caída y el sistema lo detecta correctamente, el número de TP y FP es igual a 0, lo que equivaldría también a una precisión igual a 0. Esto da lugar a una situación errónea porque la fórmula no tiene valores para aplicar pero en cambio el sistema funciona correctamente.

Teniendo en cuenta estas observaciones, pasamos a detallar los resultados para dos casos diferentes:

- El primero de ellos consiste en comparar directamente los valores obtenidos en nuestro sistema con el *ground truth*, es decir, comparamos los valores *frame* a *frame* individualmente. En este caso obtenemos los siguientes resultados:

	TP	***TN***	***FP***	***FN***	***Precision***	***Recall***	***True negative rate***	***False positive rate***	***Accuracy***
Vídeo 1	309	251	5	12	0.9841	0.9626	0.9805	0.0195	0.9705
Vídeo 2	281	265	16	5	0.9461	0.9825	0.9431	0.0569	0.9629
Vídeo 3	374	313	0	15	1	0.9614	1	0	0.9786
Vídeo 4	56	219	0	188	1	0.2295	1	0	0.5940
Vídeo 5	0	263	0	271	NA	NA	1	0	0.4925
Vídeo 6	126	352	2	293	0.9844	0.3007	0.9944	0.0056	0.6184
Vídeo 7	8	392	8	362	0.5	0.0216	0.9800	0.0200	0.5195
Vídeo 8	150	337	3	294	0.9804	0.3378	0.9912	0.0088	0.6212
Vídeo 9	172	300	2	6	0.9885	0.9663	0.9934	0.0066	0.9833
Vídeo 10	0	507	16	0	NA	NA	0.9694	0.0306	0.9694
Vídeo 11	0	230	0	162	NA	NA	1	0	0.5867
Vídeo 12	0	732	0	0	NA	NA	1	0	1
Vídeo 13	0	651	0	0	NA	NA	1	0	1
Vídeo 14	0	265	7	0	NA	NA	0.9743	0.02574	0.9743
Vídeo 15	0	201	0	265	NA	NA	1	0	0.4313
Vídeo 16	0	465	0	0	NA	NA	1	0	1
Vídeo 17	0	691	0	0	NA	NA	1	0	1
Vídeo 18	9	258	2	239	0.8182	0.0363	0.9923	0.0077	0.5256
Valores medios	**82.5**	**371.78**	**3.39**	**117.3**	**0.91**	**0.5332**	**0.9899**	**0.0101**	**0.7905**

Tabla 4-5: Resultados obtenidos mediante la comparación *frame-to-frame*

- La segunda evaluación se basa en introducir una variable, que llamaremos *subjetividad*, para tener en cuenta la subjetividad e imprecisión que introduce en el *ground truth* el anotador humano.
  Cada vez que se produce un cambio en el estado de caída, aplicamos la *subjetividad* a los *frames* superiores y a los inferiores, dejando sin evaluar dicho grupo de *frames*. Por ello, obtenemos una muestra menor de *frames* que el método anterior, donde se evaluaban el total de los mismos de cada secuencia de vídeo.
  Este método nos elimina pequeños fallos. Por ejemplo, en la creación del *ground truth* determinamos el inicio y final de una caída visualmente pudiendo existir discrepancias en los *frames* seleccionados entre una persona u otra.
  El valor de *subjetividad* lo situamos experimentalmente a ±4 *frames*, lo que suponen 8 *frames* sin evaluar cuando se produce un cambio en el estado. Teniendo en cuenta que los vídeos están grabados a 30 *fps*, obtenemos un período ligeramente superior a 0.25s. Con este tiempo solucionamos cualquier problema provocado por las diferentes apreciaciones del anotador humano.

	TP	***TN***	***FP***	***FN***	***Precision***	***Recall***	***True negative rate***	***False positive rate***	***Accuracy***
Vídeo 1	224	216	0	1	1	0,9956	1	0	0,9977
Vídeo 2	214	236	5	0	0,9772	1	0,9793	0,0207	0,9890
Vídeo 3	314	273	0	3	1	0,9905	1	0	0,9949
Vídeo 4	41	187	0	131	1	0,2384	1	0	0,6351
Vídeo 5	0	223	0	199	NA	NA	1	0	0,5284
Vídeo 6	107	314	0	256	1	0,2948	1	0	0,6219
Vídeo 7	4	356	4	318	0,5	0,0124	0,9889	0,0111	0,5279
Vídeo 8	123	296	0	233	1	0,3455	1	0	0,6426
Vídeo 9	138	270	0	0	1	1	1	0	1
Vídeo 10	0	491	16	0	NA	NA	0,9684	0,0316	0,9684
Vídeo 11	0	198	0	122	NA	NA	1	0	0,6188
Vídeo 12	0	716	0	0	NA	NA	1	0	1
Vídeo 13	0	635	0	0	NA	NA	1	0	1
Vídeo 14	0	253	3	0	NA	NA	0,9883	0,0117	0,9883
Vídeo 15	0	169	0	217	NA	NA	1	0	0,4378
Vídeo 16	0	449	0	0	NA	NA	1	0	1
Vídeo 17	0	679	0	0	NA	NA	1	0	1
Vídeo 18	5	228	0	203	1	0.0240	1	0	0.5344
Valores medios	**65**	**343.83**	**1.56**	**93.5**	**0.94**	**0.5446**	**0.9958**	**0.0042**	**0.8047**

Tabla 4-6: Resultados obtenidos utilizando la variable *subjetividad*

A continuación comparamos los resultados obtenidos para las dos evaluaciones realizadas:

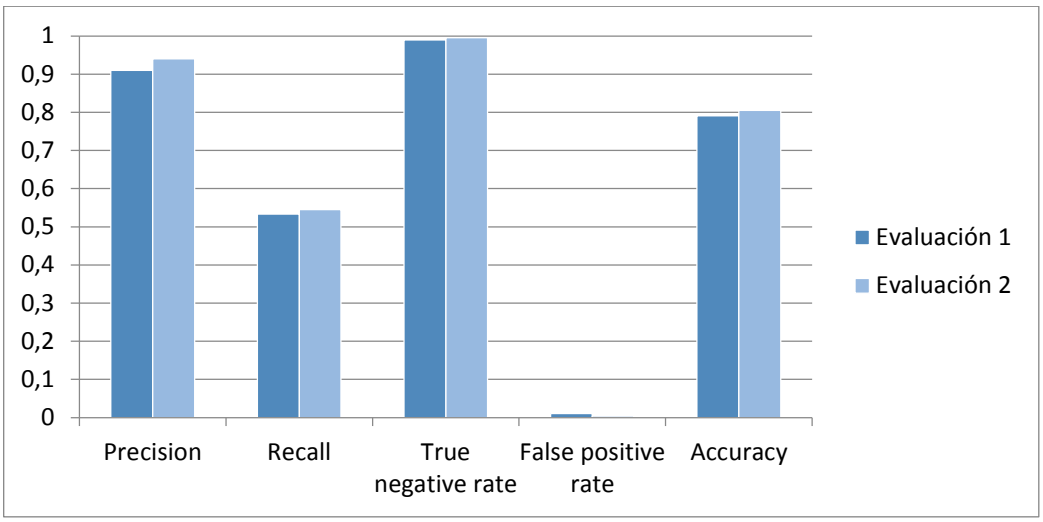

Figura 4-1: Comparación entre los dos métodos utilizados para evaluar el sistema

A la vista de los resultados, observamos que ambos métodos nos ofrecen un comportamiento aceptable, ya que detecta caídas con una precisión muy alta. Además, obtenemos unos valores, que se acercan a los ideales en las medidas de *true negative rate* y de *false positive rate*, que nos indica que la clasificación de los *frames* se hace de manera correcta. Aunque debemos tener en cuenta que las otras medidas varían en función de la secuencia de vídeo.
Comparando ambas evaluaciones apreciamos una ligera mejora en la segunda, donde aplicábamos la variable subjetividad, pero en general no existen diferencias entre ambas.

Seguidamente vamos a detallar cada uno de los vídeos, describiendo sus características, para entender mejor los resultados anteriores. Vamos a diferencias los vídeos en tres grupos:

- **Caídas desde diferentes ángulos**: En los **vídeos 1, 2 y 3** nos encontramos con diferentes caídas consecutivas, desde distintas posiciones, junto con acciones normales como caminar por la sala. En todos los casos el sistema funciona de manera muy precisa, llegando a obtener valores de *accuracy* muy próximos al 100%.
  Por otro lado, en los **vídeos 4, 5, 6, 7, 8 y 18** nos encontramos con diversas caídas producidas en línea recta respecto de la posición de la cámara. Esto produce que la *bounding box* apenas varíe y se complique la detección de caídas como observamos en las tablas de evaluaciones. Los resultados muestran un *recall* muy bajo que indica el alto número de falsos negativos, FN, por lo explicado anteriormente.

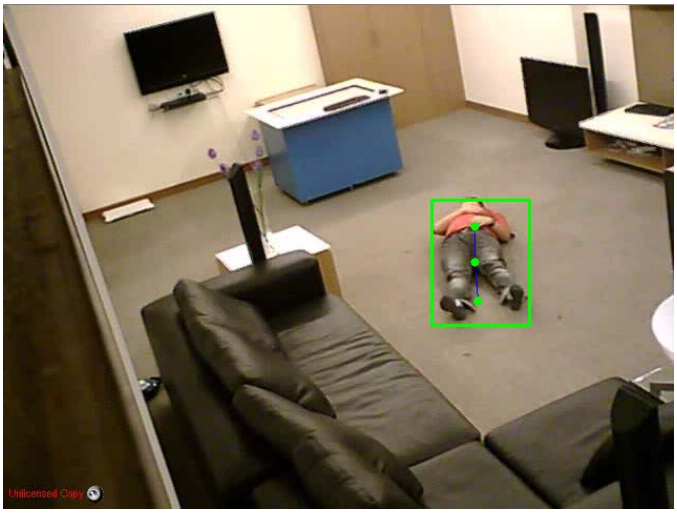

Figura 4-2: *Frame* del vídeo 5 con caída no detectada

- **Acción de sentarse**: En los **vídeos 16 y 17** observamos a la persona realizando la acción de sentarse en diferentes sitios y posiciones, sin que se produzca ninguna caída. El principal problema que podría existir sería la aparición de falsos positivos, FP, que indicasen caída cuando no se ha producido. En nuestro caso el sistema funciona perfectamente.

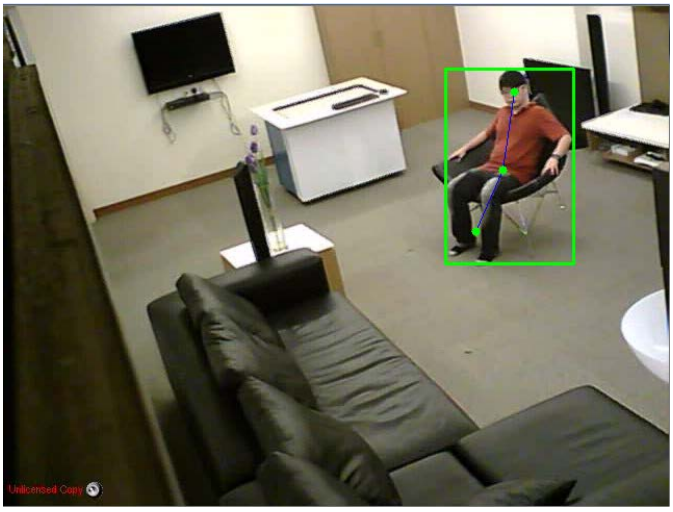

Figura 4-3: *Frame* del vídeo 17 que incluye a la persona sentada

- **Situaciones cotidianas junto con caídas:** Las secuencias de **vídeo 9, 10 y 12** nos muestran a la persona realizando acciones de agacharse así como caídas simuladas. A la vista de los resultados podemos observar que nuestro sistema discrimina bien entre ambas acciones, no dando lugar a falsos positivos, FP.

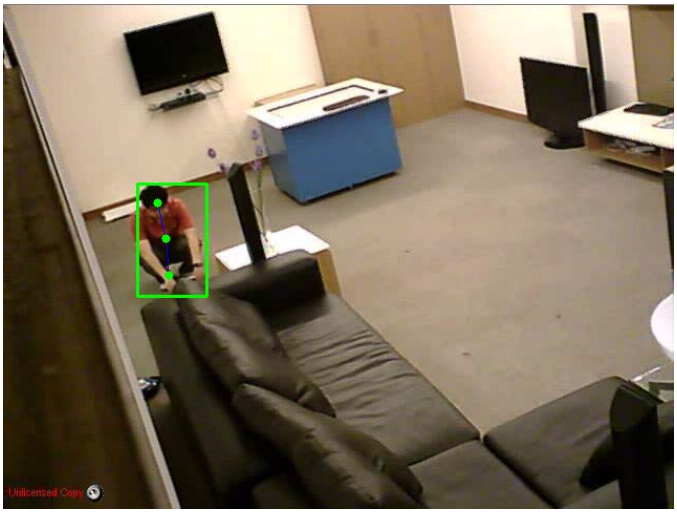

Figura 4-4: *Frame* del vídeo 10 con persona agachada sin detectar caída

En los **vídeos 11 y 15** existen caídas menos fuertes que en los otros casos, ya que la persona pierde poco a poco el equilibrio quedándose sobre el suelo de rodillas y con las manos apoyadas o tumbada, respectivamente. Según nuestro criterio hemos considerado que se produce una caída, y así lo hemos anotado en el *ground truth*, puesto que a pesar de que la persona se recupera y no es estrictamente una caída tampoco lo consideramos un suceso normal. Como podemos observar en los resultados nuestro sistema no detecta las caídas de este tipo, esto es debido a que la velocidad de cambio no es lo suficientemente alta entre los *frames* involucrados.

Por último, nos encontramos con los **vídeos 13 y 14** que nos muestran a la persona caminando y realizando carreras cortas por toda la sala sin que se produzca ninguna caída. A la vista de los resultados, el sistema funciona perfectamente cuando se encuentra con este tipo de suceso.

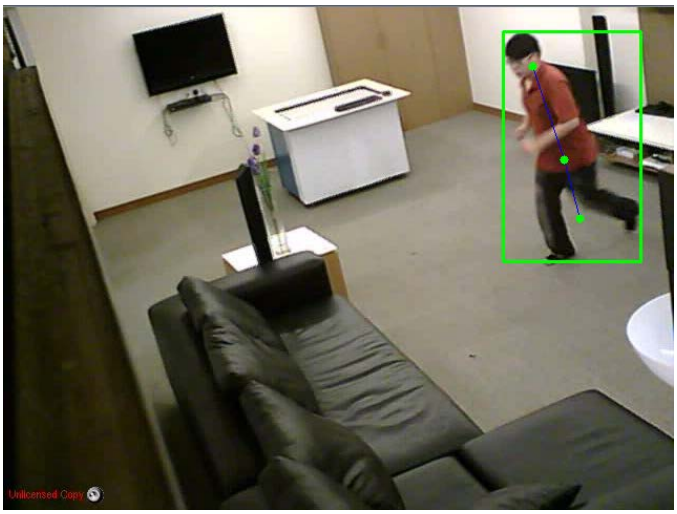

Figura 4-5: *Frame* del vídeo 14 simulando carrera

Para terminar la sección vamos a comparar los valores medios obtenidos con los pertenecientes a las implementaciones que se han tomado como referencia, [10] y [16].

4.3.2 Coste computacional

Una de las características a tener en cuenta en la detección de caídas es el funcionamiento en tiempo real, es decir, aplicar el sistema sobre una señal de vídeo en directo. En nuestro caso no ha sido posible, por lo que se ha recurrido a un grupo de vídeos previamente grabados para comprobar el funcionamiento del algoritmo. A pesar de esto, se ha diseñado un sistema que funciona en modo on-line, con un mínimo tiempo de espera en el peor de los casos, lo que permitiría su posterior implementación en sistemas reales.

Para comprobar si la detección de caídas implementada funcionaría en tiempo real se ha calculado el tiempo de ejecución de todo el sistema. El tiempo analizado incluye la lectura de vídeo, el modelado de fondo, la extracción de parámetros y el algoritmo de decisión. Obteniendo para cada vídeo los siguientes valores:

	***Frames* totales**	**Tiempo medio de ejecución por *frame* (segundos)**	**Tiempo medio de ejecución de *frames* por segundo *(fps*)**
Vídeo 1	577	0.051	19.55
Vídeo 2	567	0.051	19.01
Vídeo 3	702	0.052	19.39
Vídeo 4	463	0.052	19.38
Vídeo 5	534	0.053	18.81
Vídeo 6	773	0.051	19.68
Vídeo 7	770	0.051	19.67
Vídeo 8	808	0.052	19.38
Vídeo 9	480	0.052	19.08
Vídeo 10	523	0.049	20.36
Vídeo 11	392	0.049	20.58
Vídeo 12	732	0.049	20.60
Vídeo 13	651	0.049	20.23
Vídeo 14	272	0.050	19.83
Vídeo 15	466	0.049	20.56
Vídeo 16	465	0.053	19.01
Vídeo 17	706	0.054	18.47
Vídeo 18	508	0.051	19.71
Valor medio	**577.17**	**0.051**	**19.63**

Tabla 4-7: Tiempos de ejecución del algoritmo

Podemos observar que el tiempo medio utilizado por cada *frame* es de 0.051 segundos, lo que implica una velocidad aproximada de 19 *fps*.
De este tiempo medio, alrededor de 0.38 segundos son empleados en el algoritmo de decisión que permite detectar las caídas, dedicando el resto del tiempo a la lectura de vídeo, segmentación y extracción de parámetros. Por lo tanto, observamos como el algoritmo de decisión consume aproximadamente un **74%** del tiempo total empleado, siendo el principal cuello de botella de nuestro sistema. Esto es debido, entre otras cosas, a las diversas comprobaciones realizadas y a los saltos entre frames cuando aparece un evento de posible caída.

Además, comparamos el coste computacional de nuestro sistema con los datos del sistema implementado en [16]. En ese caso el tiempo medio de ejecución por *frame* es de 0.87 segundos y el tiempo medio de ejecución de *frames* por segundo es de 1.35 *fps*. Podemos observar que nuestro sistema reduce drásticamente el tiempo que necesita el algoritmo para llevar a cabo todas las operaciones, esto se debe principalmente a la migración del sistema desde *Matlab* a *C++*.

4.3.3 Comparativa de resultados

En esta sección vamos a comparar los resultados obtenidos por nuestro sistema con los equivalentes a los sistemas mencionados en [**10**] y [**16**].
En primer lugar realizamos la comparación con el sistema original con las métricas utilizadas por ellos. Obteniendo los siguientes resultados:

	***Recall* (%)**	***False positive rate* (%)**	**Tiempo medio de ejecución por *frame* (s)**
Sistema original [10]	**90.5**	6.7	0.19
Sistema propio	54.5	**0.42**	**0.051**

Tabla 4-8: Comparación de resultados con el sistema original

A la vista de los resultados, observamos un valor de *recall* muy inferior al sistema original, debido al problema con las caídas frontales respecto de la cámara. Por otro lado, nuestro sistema mejora la tasa de falsos positivos puesto que apenas detecta falsas alarmas y reduce considerablemente el tiempo de ejecución del sistema. Estas diferencias se deben a la variación de algunos vídeos del *dataset* original y los diferentes umbrales de cambio entre frames.

Posteriormente, realizamos una comparación con el método utilizado en [16], donde se utilizan exactamente las mismas métricas, lo que nos ofrecerá una comparación más detallada. Destacar que en el sistema [16] se realiza la evaluación sobre un *dataset*

reducido a 13 vídeos, por lo que incluimos una evaluación reducida a dichos vídeos para establecer una comparación más realista.

	Precision	***Recall***	***True negative rate***	***False positive rate***	***Accuracy***	**Tiempo medio de ejecución por *frame* (s)**
Sistema [16]	**98.66**	**65.53**	99.26	0.73	82.51	0.87
Sistema propio con 18 vídeos	94.00	54.46	**99.58**	**0.42**	80.47	**0.051**
Sistema propio con 13 vídeos	92.53	61.68	99.42	0.58	**86.47**	**0.051**

Tabla 4-9: Comparación de resultados con el TFG de [16]

En este caso hemos elegido el mejor enfoque de las dos evaluaciones posibles para ambos casos. En general podemos apreciar unos resultados muy parecidos entre los dos sistemas, destacando unos valores de *precision* óptimos en ambos casos y un valor de *recall* algo bajo, provocado por las caídas no detectadas.

4.4 Conclusiones

Después de realizar las pruebas pertinentes en las secuencias de vídeo disponibles y analizando los resultados obtenidos en la evaluación, podemos afirmar que el sistema ofrece un buen rendimiento en detección de caídas simples y, además, es capaz de diferenciar otras actividades que pueden dar lugar a confusión, como caminar, sentarse o agacharse, obteniendo un porcentaje muy alto de acierto.
Sin embargo, el sistema encuentra problemas para la detección cuando la caída se produce en línea recta respecto de la posición de la cámara y cuando la caída se produce lentamente, como por ejemplo un posible mareo. En el primer caso la *bounding box* apenas varía en esta situación y por tanto sus parámetros son similares para el sistema. En el otro caso, no se produce una variación brusca superior al valor establecido, épsilon, y a pesar de que la *bounding box* varía, el sistema no detecta la caída.

Por tanto, concluimos que el sistema desarrollado es robusto a la detección de caídas en diversas situaciones, incluyendo actividades cotidianas que pueden dar lugar a falsas caídas, debiendo mejorarse principalmente en las situaciones donde la posición de la cámara no sea la ideal.

5 Conclusiones y trabajo futuro

5.1 Conclusiones

El objetivo de este proyecto era la creación de un algoritmo que detectara caídas simuladas en un entorno similar a uno doméstico.

En primer lugar, se realizó una clasificación de los posibles métodos de detección de caídas en el estado del arte, eligiendo un sistema de análisis de vídeo puesto que era el que mejores características nos ofrecía. Posteriormente, se enumeraron los diferentes algoritmos y técnicas existentes dentro del análisis de vídeo, tomando como opción óptima un sistema basado en el análisis de la figura humana a partir de la extracción de tres parámetros [10]. La elección de este sistema se debe a su implementación sencilla pero robusta que nos ofrece unos resultados aceptables con un bajo coste material y computacional.

Una vez elegido el algoritmo de detección de caídas, se desarrolla un sistema propio que incluye un modelado frente-fondo y una segmentación, la creación de una *bounding box*, la extracción de sus parámetros característicos y la implementación de un algoritmo de decisión que determine el estado del sujeto, principalmente caída o no caída.

Finalmente, se realiza una evaluación de los resultados obtenidos, estableciendo las siguientes conclusiones principales:

- El proceso de segmentación inicial, que determina el *background* y el *foreground* de la secuencia de vídeo, debe ser realizado de una manera muy precisa y correcta, pues determinará el funcionamiento posterior del sistema.
- La implementación del algoritmo de decisión debe tener en cuenta todos los estados de caída y sus futuras posibilidades. Siguiendo esta recomendación se obtendrá un sistema eficiente y que reduce los casos más complicados.
- A la vista de los resultados, este enfoque seleccionado funciona muy bien en situaciones de caídas simples y diferencia perfectamente otras actividades cotidianas con patrones similares a una caída. Sin embargo, se reduce el porcentaje de acierto en la detección de caídas cuando éstas se producen en la línea de visión de la única cámara del sistema.
- Por último, observamos en la evaluación que el coste computacional del sistema es ínfimo, por lo tanto apenas influye en el proceso de lectura de vídeo. Esta característica acerca el sistema a una posible implementación en tiempo real y en un entorno doméstico.

5.2 Trabajo futuro

Los sistemas de detección de caídas están aumentando durante los últimos años y continuará esta tendencia debido a que las caídas en entornos domésticos son el mayor problema de salud para la población anciana.
Por lo tanto, nos encontramos en un área con mucho trabajo por realizar y numerosas aplicaciones todavía en desarrollo. Por esta razón se debe seguir progresando en este ámbito, desarrollando nuevos sistemas que solucionen los problemas actuales o implementando mejoras en los ya existentes para obtener sistemas más robustos y eficientes.

Teniendo en cuenta el proyecto realizado, se van a establecer una serie de ideas sobre las que profundizar en el trabajo futuro:

- **Eficiencia:** En primer lugar se deberían conseguir unos resultados más eficientes que los de nuestro sistema. Se debe solucionar la detección de caídas en línea recta con la cámara, para ello se puede hacer uso de un sistema de varias cámaras que extraiga los parámetros independientemente.

- **Mejoras en la segmentación:** Por otro lado se deben incluir mejoras en el sistema de segmentación que permita su uso en *datasets* más completos y con más características. Por ejemplo, la inclusión de un sistema de *background* doble que permita el análisis de la secuencia de vídeo cuando la persona se encuentra inicialmente en el mismo. Otro ejemplo, sería la inclusión de la segmentación múltiple que permita diferenciar varias personas o, incluso, mascotas que interfieran en la detección de caídas.

- **Nuevas técnicas:** Teniendo en cuenta las limitaciones de este proyecto no se ha podido profundizar más en otros enfoques. Como trabajo futuro se podrían tener en cuenta otros medios para realizar el algoritmo, destacando el uso de la Kinect que permitiría trabajar con imágenes de profundidad, lo que supondría muchas ventajas frente a aquellas basadas en el vídeo convencional.

- **Integración del algoritmo en una aplicación:** Por último, se propone la integración del algoritmo desarrollado en una aplicación que permita comprobar su funcionamiento en situaciones reales. El diseño de esta aplicación era uno de los objetivos iniciales de este proyecto, que no se ha realizado por falta de tiempo.

6 Glosario

PFC	Proyecto Fin de Carrera
IR	Infrared Radiation
GT	Ground Truth
RGB	Red Green Blue (modelo de color matemático)
FPS	Frames Per Second
MJPEG	Motion Joint Photographic Experts Group
BSD	Berkeley Software Distribution

7 Referencias

[1] M. Mubashir, L. Shao, L. Seed, *"A survey on fall detection: Principles and approaches"*, Neurocomputing; 100: 144-152, 2013.

[2] J. Willems, G. Debard, B. Bonroy, B. Vanrusmte, T. Goedemé, *"How to detect human fall in video? An overview"*, Proc. Of the International Conference on Positioning and Context-Awareness, Antwerp; 2009.

[3] R. Igual, C. Medrano, I. Plaza, *"Challenges, issues and trends in fall detection systems"*, BioMedical Engineering OnLine: 12-66, 2013.

[4] R. Hedge, Dr. B. G. Sudarshan, Dr. S. C. Prasanna Kumar, Dr. S. A. Hariprasad, Dr. B. S. Satyanarayana, *"Technical advances in fall detection system – A review"*, International Journal of Computer Science and Mobile Computing Vol.2 Issue. 7: 152-160, 2013.

[5] A. Edgcomb, F. Vahid, "*Automated fall detection on privacy-enhanced video*", in Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE , vol., no: 252-255, 2012.

[6] O. Mohamed, Ho-jin Choi, Y. Iraqi, *"Fall Detection Systems for Elderly Care: A Survey"*, in New Technologies, Mobility and Security (NTMS), 2014 6th International Conference on, vol., no: 1-4, 2014.

[7] S.F. Ali, M. Muaz, A. Fatima, F. Idrees, N. Nazar, *"Human fall detection"*, in Multi Topic Conference (INMIC), 2013 16th International , vol., no: 101-105, 2013.

[8] M. Krekovic, P. Ceric, T. Dominko, M. Ilijas, K. Ivancic, V. Skolan, J. Sarlija, *"A method for real-time detection of human fall from video"*, in MIPRO, 2012 Proceedings of the 35th International Convention , vol., no: 1709-1712, 2012.

[9] Khue Tra; T.V. Pham, *"Human fall detection based on adaptive background mixture model and HMM"*, in *Advanced Technologies for Communications (ATC), 2013 International Conference on* , vol., no: 95-100, 2013.

[10] J. Luen Chua, Y. Choon Chang, W. Keong Lim, *"A Simple Vision Based Fall Detection Technique for Video Surveillance"*, Signal, Image and Video Processing, LNCS, 1-11, 2013.

[11] V. D. Nguyen; M. T. Le, A. D. Do, H. H. Duong, T. D. Thai, D. H. Tran, *"An efficient camera-based surveillance for fall detection of elderly people"*, in Industrial Electronics and Applications (ICIEA), 2014 IEEE 9th Conference on , vol., no: 994-997, 2014.

[12] D. Anderson, R. Luke, M. Skubic, J. M. Keller, M. Rantz, M. Aud, *"Evaluation of a video based fall recognition system for elders using voxel space"*, in Proc., Int. Conf. Int. Soc. Gerontechnol., Pisa, Italy, 77–82, 2008.

[13] A. T. Nghiem, E. Auvinet, J. Meunier, *"Head detection using Kinect camera and its application to fall detection"*, in Information Science, Signal Processing and their Applications (ISSPA), 2012 11th International Conference on , vol., no: 164-169, 2012.

[14] V. Bevilacqua, N. Nuzzolese, D. Barone, M. Pantaleo, M. Suma, D. D'Ambruoso, A. Volpe, C. Loconsole, F. Stroppa, *"Fall detection in indoor environment with kinect sensor"*, in Innovations in Intelligent Systems and Applications (INISTA) Proceedings, 2014 IEEE International Symposium on , vol., no: 319-324, 2014.

[15] E. Auvinet, C. Rougier, J. Meunier, A. St-Arnaud and J. Rousseau, *"Multiple cameras fall dataset"*, Technical Report 1350, DIRO - Université de Montréal, Montreal, Canada, 2010.

[16] S. Cerro, *"Detección de caídas para vídeo-monitorización en entornos domésticos"*, Trabajo Fin de Grado, Ingeniería de Telecomunicación, Universidad Autónoma de Madrid, Mayo, 2014.

[17] Características Kinect 2, http://www.kinectfordevelopers.com/es/2014/01/28/caracteristicas-kinect-2/

[18] OpenCV documentation, http://docs.opencv.org/2.4/index.html

Anexo A: Presupuesto

1) Ejecución Material

- Compra de ordenador personal (Software incluido)..... 2.000 €
- Material de oficina..... 200 €
- Total de ejecución material..... 2.200 €

2) Gastos generales

- 16 % sobre Ejecución Material..... 352 €

3) Beneficio Industrial

- 6 % sobre Ejecución Material..... 132 €

4) Honorarios Proyecto

- 640 horas a 15 € / hora..... 9600 €

5) Material fungible

- Gastos de impresión..... 60 €
- Encuadernación..... 200 €

6) Subtotal del presupuesto

- Subtotal Presupuesto..... 12060 €

7) I.V.A. aplicable

- 21% Subtotal Presupuesto ..... 2532.6 €

8) Total presupuesto

- Total Presupuesto..... 14593,6 €

Madrid, Marzo de 2016

El Ingeniero Jefe de Proyecto

Fdo.: David Dean Pulido

Ingeniero de Telecomunicación

Anexo B: Pliego de condiciones

Este documento contiene las condiciones legales que guiarán la realización, en este proyecto, de *un algoritmo de detección de caídas mediante vídeo-monitorización*. En lo que sigue, se supondrá que el proyecto ha sido encargado por una empresa cliente a una empresa consultora con la finalidad de realizar dicho sistema. Dicha empresa ha debido desarrollar una línea de investigación con objeto de elaborar el proyecto. Esta línea de investigación, junto con el posterior desarrollo de los programas está amparada por las condiciones particulares del siguiente pliego.

Supuesto que la utilización industrial de los métodos recogidos en el presente proyecto ha sido decidida por parte de la empresa cliente o de otras, la obra a realizar se regulará por las siguientes:

Condiciones generales

1. La modalidad de contratación será el concurso. La adjudicación se hará, por tanto, a la proposición más favorable sin atender exclusivamente al valor económico, dependiendo de las mayores garantías ofrecidas. La empresa que somete el proyecto a concurso se reserva el derecho a declararlo desierto.

2. El montaje y mecanización completa de los equipos que intervengan será realizado totalmente por la empresa licitadora.

3. En la oferta, se hará constar el precio total por el que se compromete a realizar la obra y el tanto por ciento de baja que supone este precio en relación con un importe límite si este se hubiera fijado.

4. La obra se realizará bajo la dirección técnica de un Ingeniero Superior de Telecomunicación, auxiliado por el número de Ingenieros Técnicos y Programadores que se estime preciso para el desarrollo de la misma.

5. Aparte del Ingeniero Director, el contratista tendrá derecho a contratar al resto del personal, pudiendo ceder esta prerrogativa a favor del Ingeniero Director, quien no estará obligado a aceptarla.

6. El contratista tiene derecho a sacar copias a su costa de los planos, pliego de condiciones y presupuestos. El Ingeniero autor del proyecto autorizará con su firma las copias solicitadas por el contratista después de confrontarlas.

7. Se abonará al contratista la obra que realmente ejecute con sujeción al proyecto que sirvió de base para la contratación, a las modificaciones autorizadas por la superioridad o a las órdenes que con arreglo a sus facultades le hayan comunicado por escrito al Ingeniero Director de obras siempre que dicha obra se haya ajustado a los preceptos de los pliegos de condiciones, con arreglo a los cuales, se harán las modificaciones y la valoración de las diversas unidades sin que el importe total pueda exceder de los presupuestos aprobados. Por consiguiente, el número de unidades que se consignan en el proyecto o en el presupuesto, no podrá servirle de fundamento para entablar reclamaciones de ninguna clase, salvo en los casos de rescisión.

8. Tanto en las certificaciones de obras como en la liquidación final, se abonarán los trabajos realizados por el contratista a los precios de ejecución material que figuran en el presupuesto para cada unidad de la obra.

9. Si excepcionalmente se hubiera ejecutado algún trabajo que no se ajustase a las condiciones de la contrata pero que sin embargo es admisible a juicio del Ingeniero Director de obras, se dará conocimiento a la Dirección, proponiendo a la vez la rebaja de precios que el Ingeniero estime justa y si la Dirección resolviera aceptar la obra, quedará el contratista obligado a conformarse con la rebaja acordada.

10. Cuando se juzgue necesario emplear materiales o ejecutar obras que no figuren en el presupuesto de la contrata, se evaluará su importe a los precios asignados a otras obras o materiales análogos si los hubiere y cuando no, se discutirán entre el Ingeniero Director y el contratista, sometiéndolos a la aprobación de la Dirección. Los nuevos precios convenidos por uno u otro procedimiento, se sujetarán siempre al establecido en el punto anterior.

11. Cuando el contratista, con autorización del Ingeniero Director de obras, emplee materiales de calidad más elevada o de mayores dimensiones de lo estipulado en el proyecto, o sustituya una clase de fabricación por otra que tenga asignado mayor precio o ejecute con mayores dimensiones cualquier otra parte de las obras, o en general, introduzca en ellas cualquier modificación que sea beneficiosa a juicio del Ingeniero Director de obras, no tendrá derecho sin embargo, sino a lo que le correspondería si hubiera realizado la obra con estricta sujeción a lo proyectado y contratado.

12. Las cantidades calculadas para obras accesorias, aunque figuren por partida alzada en el presupuesto final (general), no serán abonadas sino a los precios de la contrata, según las condiciones de la misma y los proyectos particulares que para ellas se formen, o en su defecto, por lo que resulte de su medición final.

13. El contratista queda obligado a abonar al Ingeniero autor del proyecto y director de obras así como a los Ingenieros Técnicos, el importe de sus respectivos honorarios facultativos por formación del proyecto, dirección técnica y administración en su caso, con arreglo a las tarifas y honorarios vigentes.

14. Concluida la ejecución de la obra, será reconocida por el Ingeniero Director que a tal efecto designe la empresa.

15. La garantía definitiva será del 4% del presupuesto y la provisional del 2%.

16. La forma de pago será por certificaciones mensuales de la obra ejecutada, de acuerdo con los precios del presupuesto, deducida la baja si la hubiera.

17. La fecha de comienzo de las obras será a partir de los 15 días naturales del replanteo oficial de las mismas y la definitiva, al año de haber ejecutado la provisional, procediéndose si no existe reclamación alguna, a la reclamación de la fianza.

18. Si el contratista al efectuar el replanteo, observase algún error en el proyecto, deberá comunicarlo en el plazo de quince días al Ingeniero Director de obras, pues transcurrido ese plazo será responsable de la exactitud del proyecto.

19. El contratista está obligado a designar una persona responsable que se entenderá con el Ingeniero Director de obras, o con el delegado que éste designe, para todo relacionado con ella. Al ser el Ingeniero Director de obras el que interpreta el proyecto, el contratista deberá consultarle cualquier duda que surja en su realización.

20. Durante la realización de la obra, se girarán visitas de inspección por personal facultativo de la empresa cliente, para hacer las comprobaciones que se crean oportunas. Es obligación del contratista, la conservación de la obra ya ejecutada hasta la recepción de la misma, por lo que el deterioro parcial o total de ella, aunque sea por agentes atmosféricos u otras causas, deberá ser reparado o reconstruido por su cuenta.

21. El contratista, deberá realizar la obra en el plazo mencionado a partir de la fecha del contrato, incurriendo en multa, por retraso de la ejecución siempre que éste no sea debido a causas de fuerza mayor. A la terminación de la obra, se hará una recepción provisional previo reconocimiento y examen por la dirección técnica, el depositario de efectos, el interventor y el jefe de servicio o un representante, estampando su conformidad el contratista.

22. Hecha la recepción provisional, se certificará al contratista el resto de la obra, reservándose la administración el importe de los gastos de conservación de la misma hasta su recepción definitiva y la fianza durante el tiempo señalado como plazo de garantía. La recepción definitiva se hará en las mismas condiciones que la provisional, extendiéndose el acta correspondiente. El Director Técnico propondrá a la Junta Económica la devolución de la fianza al contratista de acuerdo con las condiciones económicas legales establecidas.

23. Las tarifas para la determinación de honorarios, reguladas por orden de la Presidencia del Gobierno el 19 de Octubre de 1961, se aplicarán sobre el denominado en la actualidad “Presupuesto de Ejecución de Contrata” y anteriormente llamado ”Presupuesto de Ejecución Material” que hoy designa otro concepto.

Condiciones particulares

1. La propiedad intelectual de los procesos descritos y analizados en el presente trabajo, pertenece por entero a la empresa consultora representada por el Ingeniero Director del Proyecto.

2. La empresa consultora se reserva el derecho a la utilización total o parcial de los resultados de la investigación realizada para desarrollar el siguiente proyecto, bien para su publicación o bien para su uso en trabajos o proyectos posteriores, para la misma empresa cliente o para otra.

3. Cualquier tipo de reproducción aparte de las reseñadas en las condiciones generales, bien sea para uso particular de la empresa cliente, o para cualquier otra aplicación, contará con autorización expresa y por escrito del Ingeniero Director del Proyecto, que actuará en representación de la empresa consultora.

4. En la autorización se ha de hacer constar la aplicación a que se destinan sus reproducciones así como su cantidad.

5. En todas las reproducciones se indicará su procedencia, explicitando el nombre del proyecto, nombre del Ingeniero Director y de la empresa consultora.

6. Si el proyecto pasa la etapa de desarrollo, cualquier modificación que se realice sobre él, deberá ser notificada al Ingeniero Director del Proyecto y a criterio de éste, la empresa consultora decidirá aceptar o no la modificación propuesta.

7. Si la modificación se acepta, la empresa consultora se hará responsable al mismo nivel que el proyecto inicial del que resulta el añadirla.

8. Si la modificación no es aceptada, por el contrario, la empresa consultora declinará toda responsabilidad que se derive de la aplicación o influencia de la misma.

9. Si la empresa cliente decide desarrollar industrialmente uno o varios productos en los que resulte parcial o totalmente aplicable el estudio de este proyecto, deberá comunicarlo a la empresa consultora.

10. La empresa consultora no se responsabiliza de los efectos laterales que se puedan producir en el momento en que se utilice la herramienta objeto del presente proyecto para la realización de otras aplicaciones.

11. La empresa consultora tendrá prioridad respecto a otras en la elaboración de los proyectos auxiliares que fuese necesario desarrollar para dicha aplicación industrial, siempre que no haga explícita renuncia a este hecho. En este caso, deberá autorizar expresamente los proyectos presentados por otros.

12. El Ingeniero Director del presente proyecto, será el responsable de la dirección de la aplicación industrial siempre que la empresa consultora lo estime oportuno. En caso contrario, la persona designada deberá contar con la autorización del mismo, quien delegará en él las responsabilidades que ostente.