Universidad de Alcalá
Escuela Politécnica Superior

Grado en Ingeniería en Electrónica y Automática Industrial

Trabajo Fin de Grado

Seguimiento de personas para detección de acciones a partir de información de profundidad

Autor: Alán Resa Peña

Tutor: Cristina Losada Gutiérrez

2022

UNIVERSIDAD DE ALCALÁ

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería en Electrónica y Automática Industrial

Trabajo Fin de Grado

Seguimiento de personas para detección de acciones a partir de información de profundidad

Autor: Alán Resa Peña

Tutor: Cristina Losada Gutiérrez

Tribunal:

Presidente: Manuel Mazo Quintas

Vocal 1º: Juan Carlos García García

Vocal 2º: Cristina Losada Gutiérrez

Calificación: .....

Fecha: .....

"Si puedes soñar sin que los sueños te dominen,
Si puedes pensar y no hacer de tus pensamientos tu único objetivo,
Si puedes encontrarte con el triunfo y el desastre,
y tratar a esos dos impostores de la misma manera...

... Si puedes llenar el implacable minuto,
con sesenta segundos de diligente labor,
tuya es la tierra y todo lo que hay en ella,
y lo que es más: ¡serás un hombre, hijo mío!"

Rudyard Kipling. "Si..." En *Rewards and fairies*(1910).

Agradecimientos

Gracias a mi padre, Jose, que creyó en mi desde el principio incluso más que yo mismo. Gracias a mi hermana Yaiza que siempre tuvo una palabra de aliento en los momentos más duros. A mis tios Mar y Pedro que impidieron que me rindiese y celebraron cada uno de mis triunfos. A mis compañeros de Acciona, por su inestimable ayuda y por hacerme de reir cuando más lo necesitaba. Gracias a mi tutora Cristina por haberme acompañado y guiado en este largo camino. A Jesús por sentarse cada vez conmigo y hacerme saber quién soy, aunque yo mismo dudase de ello. A mi pareja Verónica, que comenzó luchando conmigo en la universidad y ahora me acompaña en el final, fuiste mi mayor apoyo. Y por último gracias a mi madre, Almudena, porque me hizo ser quien soy y porque allá donde esté estará orgullosa de mi.

Gracias y mil veces gracias.

Resumen

El objetivo de este trabajo es el diseño, implementación y evaluación de un sistema para la detección de acciones de personas en tiempo real, empleando únicamente información de profundidad adquirida por una cámara de tiempo de vuelo (ToF) situada en posición cenital. El trabajo incluye una etapa de detección robusta de las personas, tras la que se realiza el seguimiento de las mismas mediante un banco de filtros de Kalman, junto con un proceso de asociación entre medidas y estimaciones, que permiten estimar la posición y velocidad de las personas involucradas. La información de la detección y seguimiento permite generar un vector de características, que se introduce en una SVM multiclase para la clasificación de acciones. Para la evaluación del trabajo desarrollado se ha empleado la base de datos GOTPD3, obteniendo resultados satisfactorios que han permitido la validación del sistema propuesto.

Palabras clave: Detección de actividades,sensor ToF, imágenes de profundidad, filtro de Kalman, Maquina de soporte vectorial, vecino más cercano, asignación Plantillas de trabajos fin de carrera/máster/grado y tesis doctorales, LaTeX, soporte de español e inglés, generación automática.

Abstract

The objective of this work is the design, implementation and evaluation of a system for the detection of people's actions in real time, using only depth information acquired by a time-of-flight (ToF) camera placed in an overhead position. The work includes a stage of robust detection of people, after which people are tracked using a bank of Kalman filters, together with a measurement and estimation association process, which allows estimating the position and velocity of the people involved. The detection and tracking information allows the generation of a feature vector, which is introduced in a multiclass SVM for the classification of actions. For the evaluation of the developed work, the GOTPD3 database has been used, obtaining satisfactory results that have allowed the validation of the proposed system.

Keywords:Activity detection, ToF sensor, Depth images, Kalman Filter, Support Vector Machine, Nearest neighbor, assignment Bsc., Msc. and PhD. Thesis template, LaTeX, English/Spanish support, automatic generation.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Contexto y objetivos

En los últimos años, la aparición de las cámaras *Red, Green, Blue and Depth* (RGBD) como Kinect [16], que además de una imagen en color, proporcionan información de profundidad (distancia de cada punto a la cámara) ha provocado que numerosos trabajos utilicen esta información para diferentes tareas como la detección de personas [17, 18], el reconocimiento de sus actividades [19], o la segmentación semántica [20], entre otras. La detección de personas, y el reconocimiento de acciones son tareas que han atraído la atención de la comunidad científica en la última década debido a sus múltiples aplicaciones en diferentes ámbitos [21,22], como la seguridad [23,24], el control de aforos o el apoyo a la dependencia [25].

Muchos trabajos que abordan la detección de personas y el reconocimiento de sus acciones, empleando tanto información de color [26,27], como RGBD [28–30]. Sin embargo, el uso de información de color permite conocer la identidad de las personas en la escena, invadiendo su privacidad,lo que puede suponer un problema en algunas aplicaciones. Para evitar problemas relacionados con la privacidad, algunos trabajos han comenzado a utilizar únicamente la información de profundidad, para la detección de personas o acciones [17,31].

Cabe mencionar que muchos de los trabajos de detección de personas a partir de información de profundidad, ubican la cámara en posición cenital para reducir las posibles oclusiones entre diferentes personas [17]. Esto permite mejorar los resultados en la etapa de detección, sin embargo dificulta el reconocimiento de acciones, por lo que son muy pocos los trabajos que abordan esta tarea con el sensor en posición cenital.

En este contexto, el presente Trabajo de Fin de Grado (TFG) tiene como objetivo diseñar, implementar y evaluar un sistema que permita la detección y seguimiento de un número variable de personas, y el reconocimiento de las acciones realizadas. Todo ello, empleando únicamente la información de profundidad proporcionada por un sensor de distancia basado en tiempo de vuelo (ToF) [32, 33], ubicado en posición cenital. El trabajo se desarrolla en el contexto del proyecto ARGOS+ (PIUAH21/IA-016) de la Universidad de Alcalá, dentro del grupo de investigación Grupo de Ingeniería Electrónica aplicada a Espacios inteligentes y Transporte (GEINTRA), y toma como punto de partida los trabajos previos realizados por miembros de dicho grupo, para la detección robusta de personas a partir de la información de profundidad ubicada en posición cenital [9,17,34].

1.2. Sistema propuesto

Para alcanzar el objetivo del TFG, se ha tomado como punto de partida el algoritmo de detección robusta de personas desarrollado previamente por miembros del grupo GEINTRA, que ha sido necesario analizar y poner en marcha, para posteriormente integrarlo con el resto de algoritmia desarrollada. El sistema resultante consta de diferente etapas que se muestran en el esquema general de la figura 1.1, que se presentan brevemente a continuación, y se describen con mayor detalle en el capítulo 3.

Figura 1.1: Diagrama de bloques del sistema completo

La entrada al sistema son las imágenes de profundidad, adquiridas con una cámara en posición cenital, que incluyen a una o varias personas realizando alguna de las acciones de interés. El sistema desarrollado puede funcionar realizando la adquisición de imágenes en tiempo real con una cámara Kinect v2, o a partir de secuencias previamente grabadas. Para el entrenamiento y la evaluación de los diferentes algoritmos desarrollados se ha hecho uso del *dataset* GOTPD3, grabado y etiquetado por miembros del grupo GEINTRA [35].

Como se ha comentado previamente, para la detección de personas se hace uso del algoritmo previamente desarrollado por miembros del grupo de investigación [9, 17, 34].

Tras la detección, se realiza el seguimiento de cada una de las personas en la escena mediante un banco de filtros de Kalman. Al tratarse de un sistema multipersona, es necesaria una fase de asociación de las detecciones con las estimaciones. Esto queda descrito en la sección 3.3.

Posteriormente, a partir de la información proporcionada por los algoritmos de detección y seguimiento de personas se lleva a cabo la clasificación de diferentes acciones: andar, correr, caer y estar parado, con especial atención a la detección de caídas, ya que se trata de una situación que puede suponer un riesgo para las personas.

La clasificación de acciones se lleva a cabo en dos etapas. En primer lugar es necesario realizar la extracción de características a partir de la información disponible en las etapas previas de detección y seguimiento. Posteriormente realiza la clasificación, basada en un algoritmo de clasificación multiclase Máquina de soporte vectorial, *Support Vector Machine* (SVM) que permite discriminar entre las diferentes acciones. Este algoritmo requiere de un proceso de entrenamiento que se realiza una sola vez, empleando, como ya se ha comentado, las imágenes de la base de datos GOTPD3.

Como resultado de aplicar los algoritmos previamente mencionados, se obtiene la posición de las personas detectadas con un alto nivel de precisión, así como y, por otra parte, se obtienen también las acciones que realizan de una manera robusta.

1.3. Organización de la memoria

Este documento recoge la memoria del TFG, y está dividida en cinco capítulos. Este primer capítulo presenta la introducción y los objetivos del trabajo. Posteriormente, en el capítulo 2 se incluyen los aspectos teóricos necesarios para el desarrollo del TFG. A continuación, el capítulo 3 describe en detalle el sistema desarrollado, cuyos resultados, así como las métricas empleadas, se presentan en el capítulo 4. Finalmente, las principales conclusiones y trabajos futuros se recogen en el capítulo 5.

Capítulo 2

Estudio teórico

2.1. Introducción

En este capítulo se desarrollan en profundidad los conceptos teóricos en los que se fundamenta este trabajo y que son necesarios para poder seguir el desarrollo del mismo.

En primera instancia se expone el principio de funcionamiento de las cámaras de profundidad basadas en tiempo de vuelo, *Time of Flight* (ToF) en los cuales se basa este TFG, justificando el uso de esta tecnología frente a otras también integradas en la industria para el mismo propósito: la adquisición de imágenes de profundidad. A continuación se muestran las especificaciones y características del sensor concreto usado, la cámara Kinect II [33, 36].

En segundo lugar se detallan los algoritmos empleados en las tres etapas principales del sistema desarrollado: para la detección de personas, el seguimiento de las mismas y la clasificación de acciones. Todo ello en secuencias de imágenes de profundidad, adquiridas por una cámara ToF ubicada en posición cenital.

Para la detección de personas, se hace uso del trabajo [9, 17], desarrollado por miembros del grupo de investigación GEINTRA, que se explica brevemente en el apartado 2.3

El seguimiento de las personas detectadas a lo largo de una secuencia de imágenes se lleva a cabo empleando un banco de filtros de Kalman [37], que permiten estimar las posiciones y velocidades de cada uno de los individuos que aparezcan en las imágenes. Tanto el filtro de Kalman, como los algoritmos empleados en la asociación entre hipótesis y medidas se detallan en la sección 2.4

A partir de la información obtenida a lo largo de una secuencia de imágenes de profundidad, se genera un vector de características que posteriormente es clasificado empleando una SVM, que permite determinar la acción detectada. Este clasificador se describe en el apartado 2.5

2.2. Cámaras de profundidad

Las cámaras de profundidad o sensores 2.5D, son dispositivos capaces de medir la distancia desde el sensor a cualquier punto dentro de su campo de visión y ,por tanto, capaces de generar una nube de puntos. Estos sensores están cada día más presentes en la industria a nivel global, siendo usados en campos como la robótica [38], la salud [39], etc.

Los sensores 2.5D obtienen información de profundidad haciendo uso de diferentes técnicas y tecnologías, entre las que se encuentran la tecnología de luz estructurada [40], la visión estéreo [41, 42] y

la medida de tiempo de vuelo (ToF). A continuación se describe el principio de funcionamiento de los sensores basados en tiempo de vuelo, dado que es la tecnología usada en este TFG para adquirir las imágenes de profundidad.

2.2.1. Principio de funcionamiento de las cámaras basadas en tiempo de vuelo

Una cámara ToF emite una señal modulada, en un medio de referencia, cuya velocidad de propagación es constante y conocida, y posteriormente se mide el tiempo que tarda dicha señal en retornar hasta el lugar desde el que se emitió. Conocido el tiempo que tarda la señal en retornar y la velocidad a la que va y vuelve, se puede calcular la distancia del sensor al punto.

Los sensores de tiempo de vuelo se pueden clasificar en dos grupos, en función de la manera en la que se calcula la distancia:

1. **De medición directa**:

   Estos sensores emiten ondas cortas de alta intensidad para luego medir el tiempo que tarda en retornar la señal mediante conversores *time-to-digital* [43]. Al ser conocida la velocidad de propagación del medio de referencia, en este caso la velocidad de la luz c, puede calcularse la distancia al objeto cómo:

$$d = \frac{c * \Delta t}{2} \tag{2.1}$$

   Siendo Δt el incremento de tiempo entre la emisión y la recepción de la señal.

   Este fenómeno se ilustra en la figura 2.1, a continuación:

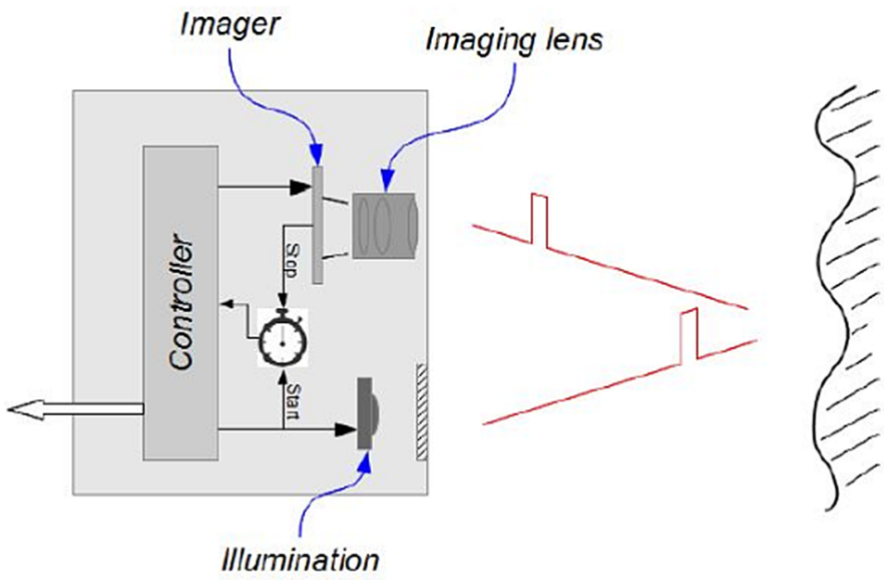


Figura 2.1: Funcionamiento de una cámara de tiempo de vuelo de medición directa (Extraída de [[1]])

2. **De medición indirecta**:

   Las cámaras ToF de medida indirecta emiten señales periódicas, moduladas y de baja intensidad. Posteriormente se mide la diferencia de fase entre las señales emitida y recibida (Ver figura 2.2:

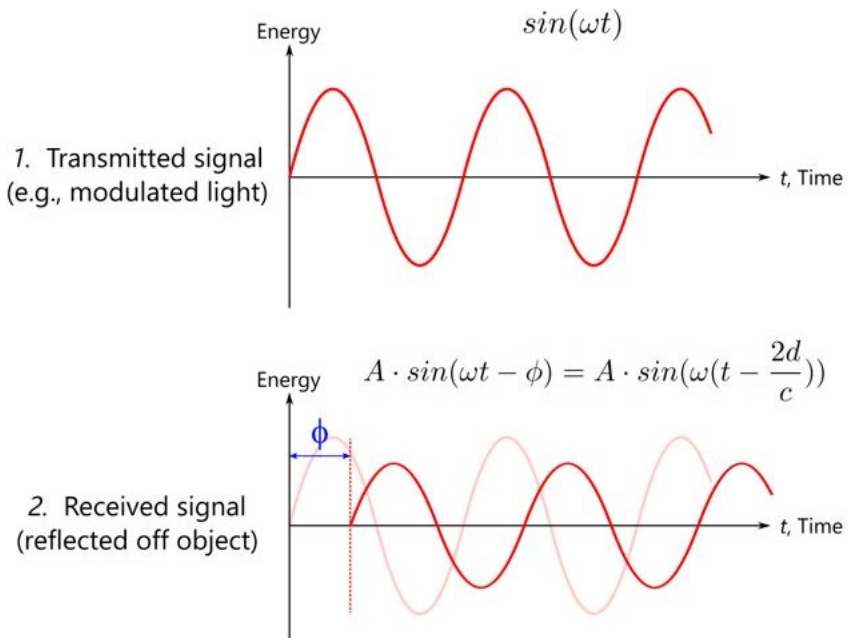


Figura 2.2: Retardo de fase entre señal emitida y señal recibida en un sensor de tiempo de vuelo con medición indirecta . (Extraida de [[2]]

Al usar señales moduladas, la distancia es directamente proporcional a la velocidad de propagación de la luz c, a la diferencia de fase Φ y además inversamente proporcional a la frecuencia de modulación fm (ecuación 2.2).

$$d = \frac{c * \Phi}{2 * 2 * \pi * fm} \tag{2.2}$$

De esta ecuación también se deduce que la máxima distancia que puede medir un sensor de tiempo de vuelo por medición indirecta sin ambigüedad es 2.3:

$$d_{max} = \frac{c}{2f} \tag{2.3}$$

El hecho de que exista una distancia máxima para las cámaras de medición indirecta es debido a que al emitir señales periódicas, una señal cuya fase sea, por ejemplo, k, será indistinguible de $2\pi + k$ o $4\pi + k$, dado que la señal se repite.

Para ambos sistemas de medición de distancia se han mostrado las ecuaciones ideales. No obstante, existe un parámetro común a ambos denominado *tiempo de integración* o τ. Este parámetro indica el tiempo que requiere el sensor de la cámara para recibir la señal de luz reflejada por la superficie de los objetos que se han establecido como objetivos en la escena. Esto significa que si el tiempo de integración es pequeño, la distancia a los objetos muy alejados no podrá ser medida correctamente, puesto que la señal de luz no habrá tenido tiempo de retornar y excitar el sensor de la cámara. τ es un parámetro configurable en la mayoría de algunas cámaras ToF, habiendo artículos como [44] que proponen sistemas de ajuste automático de τ para el mejor funcionamiento de estas cámaras.

2.2.2. Fuentes de error en cámaras ToF

Cabe destacar que el funcionamiento de estos sensores no es perfecto y por tanto se deben tener en cuenta algunos errores que pueden ocurrir durante el proceso de medición de distancias. Los más importantes, se detallan a continuación:

- **Interferencia por multicamino**:

  La reflexión de la señal en objetos con esquinas puede provocar que dicha señal no retorne directamente de nuevo al sensor que la emitió, si no que se desvíe y se refleje en otras superficies antes de regresar al sensor. Esto implica que la distancia medida por la cámara será mayor bien porque el intervalo de tiempo Δt en la ecuación 2.1 se incrementa para las cámaras basadas en medición directa o bien porque la fase Φ en la ecuación 2.2 también es mayor, provocando que haya puntos anómalos en la nube de puntos generada.

  Como se observa en la figura 2.3, la distancia medida no será proporcional a la fase ya que existe una distorsión y, por tanto, la medida será inválida. Este error puede corregirse empleando la propuesta descrita en [45]

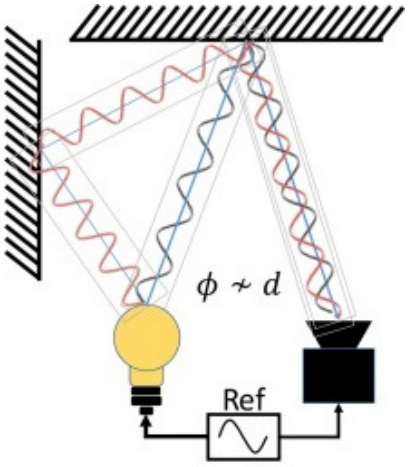


Figura 2.3: Ejemplo de distorsion por multicamino (Extraido de [[3]])

- **Artefactos de movimiento**:

  No todas las imágenes tomadas por un sensor de profundidad son imágenes estáticas, por lo que puede ocurrir que las personas u objetos en la imagen estén en movimiento. Como se ha comentado previamente, las cámaras de profundidad requieren de un tiempo de integración para calcular la diferencia de fase entre las señales emitida y recibida. Si durante ese tiempo de integración se producen cambios en la escena debidos al movimiento de dichos elementos, es habitual que se tomen medidas en puntos correspondientes al objeto, y otras en puntos del fondo, lo que provoca la obtención de medidas ruidosas de la distancia. Esto resulta en una distorsión visual de tipo "suavizado" alrededor del objeto. En la figura 2.4 se presenta un ejemplo de este efecto:

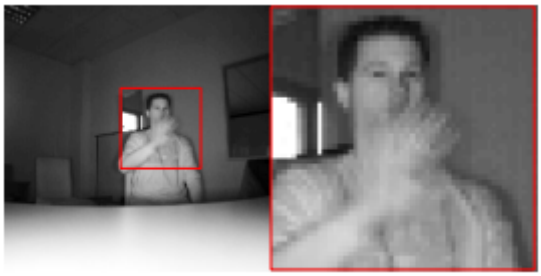


Figura 2.4: Ejemplo de artefactos de movimiento (Extraida de [[4]])

En la imagen 2.4 se señala con un recuadro rojo el lugar de la imagen en que se produce el movimiento. En la parte derecha de la imagen se amplia dicha región y se puede observar la distorsión alrededor de la mano.

Este efecto también puede corregirse, empleando diferentes métodos [4],[46]

- **Distorsión por iluminación no uniforme**

  Es un hecho que las cámaras ToF usan iluminación infrarroja focalizada para obtener las medidas de distancia a los puntos de la escena. Esto implica que la iluminación en la imagen es focalizada y no difusa, lo cual supone una posible fuente de error. Por una parte, al ser una iluminación focal e intensa, puede darse el caso que la imagen se sature en el punto en que se concentra la luz. Por otra parte, dicha iluminación al centrarse, por ejemplo, en el centro de la imagen provocará una falta de iluminación en los puntos más alejados del foco de luz, es decir, las esquinas de la imagen. Este efecto se aprecia en la figura 2.5 a continuación:

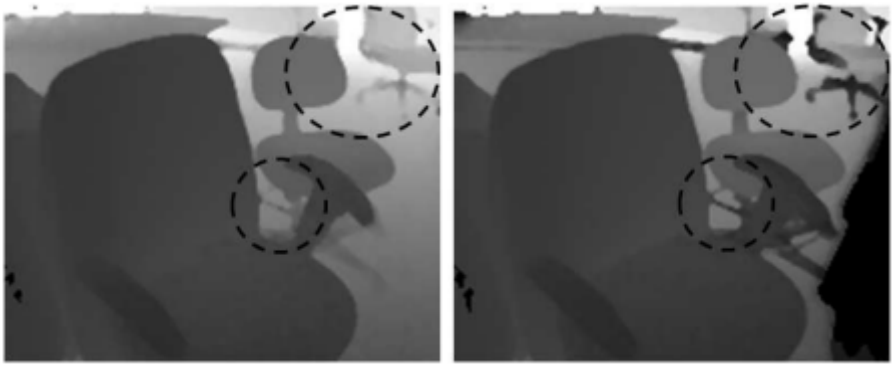


Figura 2.5: Ejemplo de distorsión producida por el efecto de iluminación no uniforme en cámaras ToF [Extraida de [5]]

- **Aliasing armónico**

  En las cámaras de profundidad, idealmente, las medidas de distancia tomadas son lineales con respecto a las reales, no obstante existen varias fuentes de distorsión que producen errores por no linealidad. Uno de estos errores lo produce el efecto conocido como *Aliasing* Armónico. Este error es producido por la imposibilidad de la cámara de reconstruir por completo la señal de correlación cruzada, es decir, el producto de la convolución de las señales cuadradas emitidas y recibidas usadas para calcular la diferencia de fase. La convolución de señales cuadradas resulta ser una señal triangular como se puede observar en la figura 2.6:

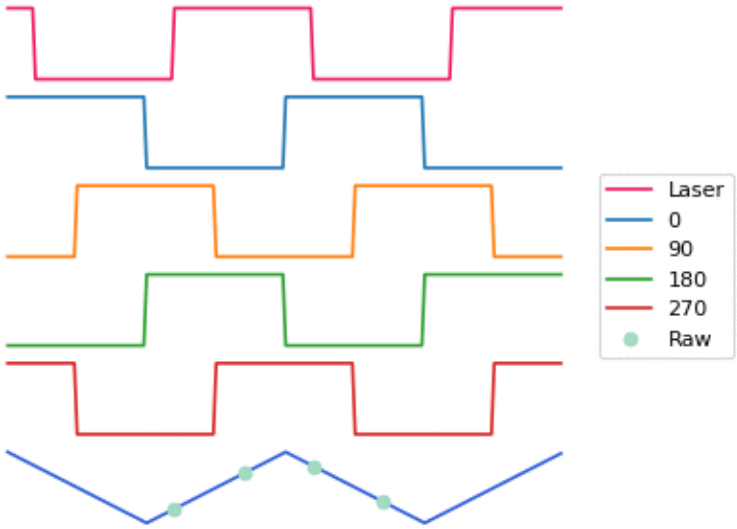


Figura 2.6: Proceso de medición de diferencia de fase en cámaras ToF [Extraida de [6]]

Para este caso concreto se comprueba que la frecuencia de muestreo es demasiado baja (4 muestras por imagen) y por tanto la reconstrucción de la señal es inexacta, lo que implica un error directo en el cálculo de la distancia. A continuación se muestra el error de distancia simulado para una cámara ToF configurada a 50 MHz capturando 4 fotogramas por segundo (figura 2.7). La distancia máxima antes de que se produzca el ajuste de fase a 50 MHz es de 2998 mm, por lo que el error de pico a pico de 58 mm es un error de precisión de aproximadamente el 2 % (58/2998).

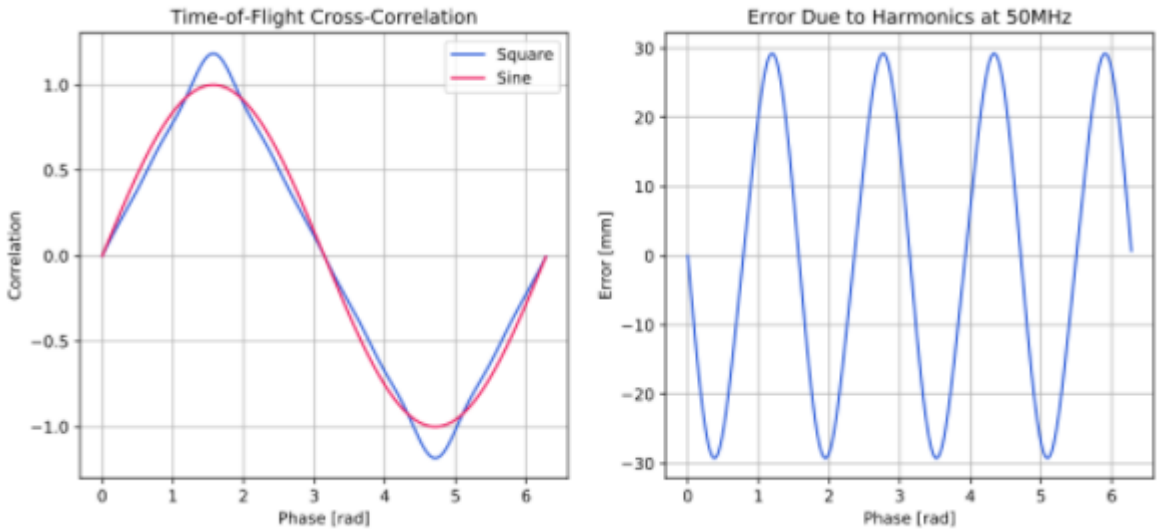


Figura 2.7: La señal de correlación cruzada medida usando señales cuadradas (azul) comparada con una sinudoide ideal (roja) a la izquierda. A la derecha aparece el error de medición obtenido [Extraido de [6]]

2.2.3. Cámara de profundidad Kinect II

Hoy en día una tecnología muy presente en el mercado e incluso en el ámbito de la investigación es la de los sensores de profundidad basados en la técnica de tiempo de vuelo. Esto es debido a la calidad de la información que son capaces de obtener, con errores máximos de un centímetro, su buen

funcionamiento en ambientes con alta luz ambiental (Para el caso de la tecnología óptica, por ejemplo) y a que la información no se ve distorsionada por condiciones como la humedad, la presión o la temperatura. Un ejemplo de un sensor de profundidad de este tipo es la cámara Kinect v2 (figura 2.8), el cual dispone de una cámara *Red, Green, Blue* (RGB) además del sensor de profundidad.

Figura 2.8: Cámara RGB y sensor de profundidad ToF en la cámara Kinect v2 [Extraida de [7]]

En la figura 2.9 se observan un conjunto de imágenes obtenidas por los diferentes sensores de la cámara Kinect V2:

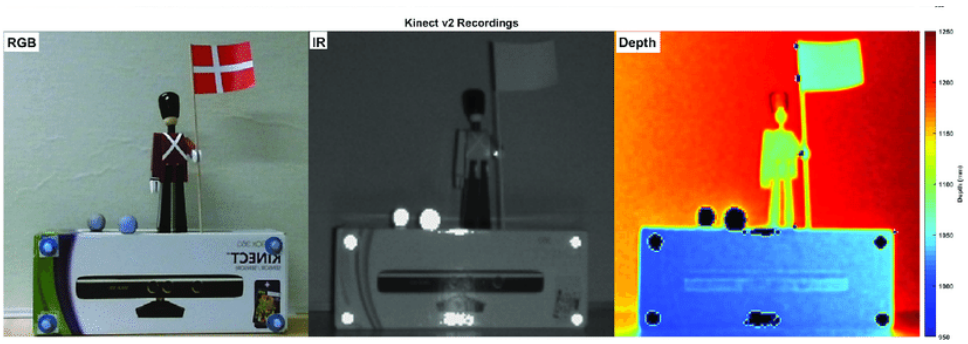


Figura 2.9: Ejemplo de imágenes adquiridas empleando una cámara Kinect V2 [Extraidas de [8]]

La cámara Kinect V2 usa el método de medición indirecta para obtener las imágenes de profundidad. Para llevarlo a cabo tiene equipado un array de emisores láser que emiten en el espectro infrarrojo, los cuales emiten haces de luz modulada por onda continua cuadradas. Estas señales emitidas son las que se reflejan en los objetos y son capturadas posteriormente por la cámara tras rebotar. Para poder capturar estos haces de luz, el sensor lleva equipado un sensor de luz basado en tecnología Semiconductor complementario de óxido metálico, *Complementary metal-oxyde-semiconductor* (CMOS) que convierte la energía lumínica en corriente eléctrica.

Está cámara se eligió para el desarrollo de este TFG dado que en el momento de su adquisición era uno de los dispositivos más competitivos del mercado debido a la relación precio/características. Además, las secuencias de imágenes de profundidad se grabaron con este dispositivo. A continuación se muestran en la tabla 2.1 las características de la cámara Kinect v2:

Característica	Valor
Resolución de cámara de color	1920 × 1080 píxeles
Ratio de captura de imágenes de cámara de color	30 imágenes por segundo
Resolución de cámara de profundidad	512 × 424 píxeles
Ratio de captura de imágenes de profundidad	30 imágenes por segundo
Campo de visión horizontal	70 grados
Campo de visión vertical	60 grados
Rango de medición operativo	De 0.5m a 4.5 metros
Tecnología de adquisición de profundidad	Tiempo de vuelo (ToF)

Tabla 2.1: Características técnicas de la cámara Kinect v2.

2.3. Detección de personas

Desde que se dispone de la tecnología para la detección y el conteo de personas a través de sistemas de visión por computador, el número de aplicaciones que requieren estas técnicas ha ido en aumento. Es por ello también que el número de soluciones a esta necesidad es amplio, incluyendo alternativas con cámaras de color [47, 48] o de profundidad [17, 18]. En el caso de este trabajo, en que se requiere la detección de personas a partir de información de una cámara de profundidad ubicada en posición cenital, para la detección de personas, se ha elegidola propuesta desarrollada en el TFG de David Fuentes Jiménez [9, 17], dentro de las líneas de investigación del grupo GEINTRA. El algoritmo se describe en detalle en los trabajos previos [9] y [17], pero para facilitar la lectura del presente trabajo, se incluye a continuación un breve resumen en que se describe la algoritmia para detección de personas empleada, cuyo esquema general se muestra en el diagrama de bloques de la figura 2.10.

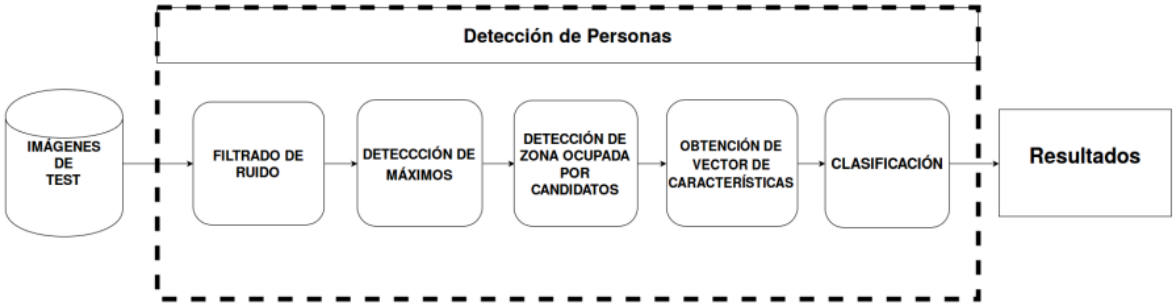


Figura 2.10: Diagrama de fases del algoritmo de detección desarrollado en el TFG [9].

Para aplicar este método sobre imágenes es necesario seguir las siguientes fases (descritas con mayor detalle en [9]):

Preprocesado de la imagen.

El punto de partida del algoritmo de detección son las imágenes de profundidad adquiridas por una cámara Kinect v2 ubicada en posición cenital, a una altura conocida, y cuyo plano imagen es paralelo al suelo, como se muestra en el esquema de la figura 2.11.

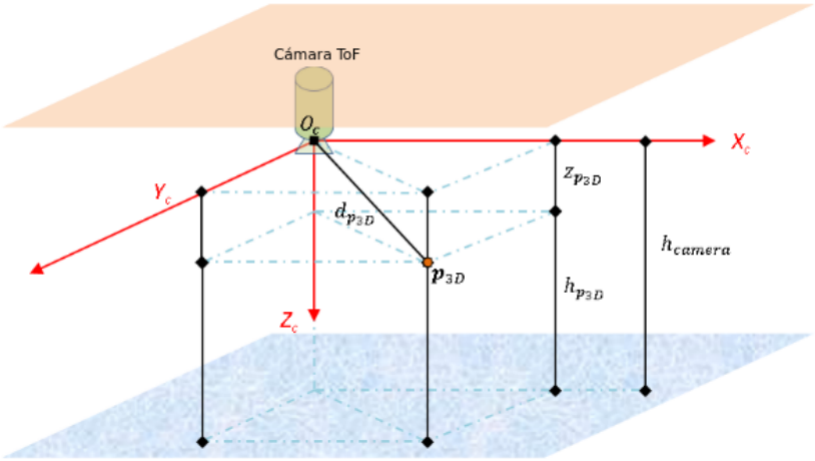


Figura 2.11: Posición del sensor de profundidad Kinect v2 [Extraído de [9]]

Para cada imagen se crea una matriz de alturas H_{mca} del mismo tamaño que la imagen de entrada (512×424) cuyos valores son la distancia del punto de interés hasta el plano del suelo. Esto se hace restando a la altura de la cámara h el valor de distancia de cada punto medido. Esta matriz se filtra para eliminar los píxeles no válidos, es decir, aquellos que o bien la cámara no es capaz de detectar correctamente y que aparecen como píxeles negros (con valor 0) en las imágenes tomadas o bien porque no pueden corresponder a personas en la escena (por ser alturas inferiores a 0.5m o superiores a 2.3m). Para finalizar el preprocesado, se asigna a cada píxel no válido el valor medio de los píxeles vecinos con nivel de vecindad 2 (siempre y cuando en ese área de búsqueda exista, al menos, un píxel válido) y se aplica un filtro de mediana de nueve elementos para suavizar la imagen. Un ejemplo en el que se muestra tanto la imagen obtenida por la cámara como el resultado de aplicar estas operaciones se presenta en la figura 2.12.

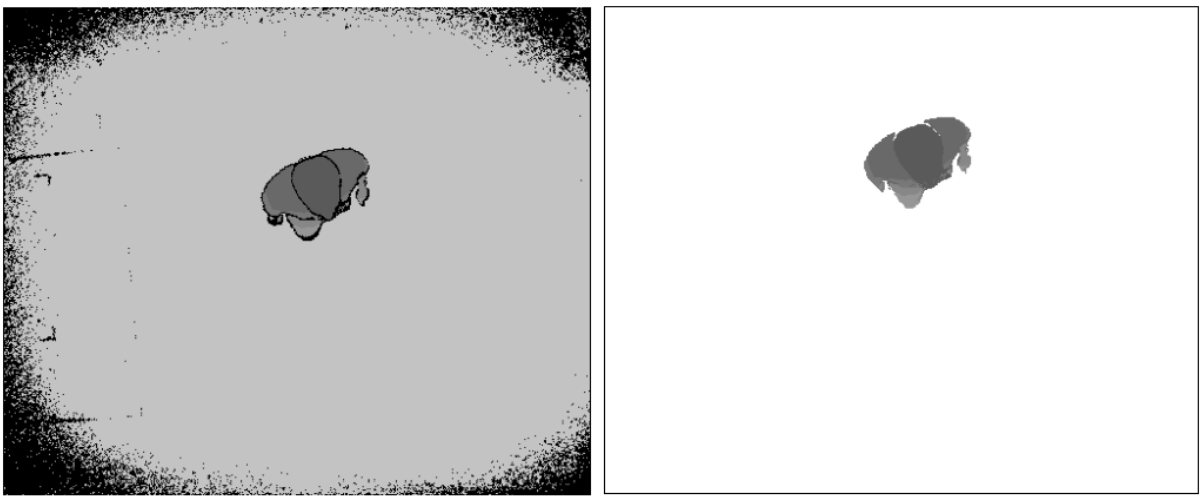


Figura 2.12: Imagen sin procesar (izquierda) e imagen procesada (derecha) [Extraídas de [9]]

Detección de máximos.

El proceso de deteccion de máximos comienza dividiendo la matriz de alturas ya filtrada en regiones más pequeñas denominadas subregiones. El tamaño de las subregiones depende directamente de los requisitos de precisión y velocidad de procesamiento requeridos, pero también de la altura de la cámara y

de las personas que vayan a ser detectadas. El objetivo es que hasta la más pequeña de las personas en la escena ocupe un número de subregiones que permita extraer suficientes datos de la misma.

De cada subregion creada se hallan los valores máximos de altura, que se almacenan en una nueva matriz H_{maxSR}. Estos máximos son posteriormente analizados para comprobar si cumplen el requisito de altura anteriormente mencionado. Debido al reducido tamaño de las subregiones, puede ocurrir que, para una misma persona, haya varios máximos. Para solucionar este problema, se compara cada máximo candidato con los de su alrededor en un radio previamente fijado y se toma como máximo final el de mayor valor.

Crecimiento de regiones alrededor de cada máximo.

Tras obtenerse los máximos de la matriz de alturas, se define una región de interés (Región de interes, *Region Of Interest* (ROI)) alrededor de cada uno de ellos, de la cual se extraen las características que permiten distinguir personas de otros objetos presentes en la escena. Para realizar esta distinción, la región debe comprender la cabeza, hombros y cuello de cada persona, esto es según estudios antropométricos [49, 50], 40 cm desde el máximo, que se ubica sobre la cabeza.

Con el fin de determinar la ROI de cada objeto o persona en escena se realiza el proceso de crecimiento de regiones. Este proceso consiste en, partiendo del máximo válido, evaluar los máximos situados en las subregiones que se encuentran en las 8 direcciones cardinales. Las subregiones se clasifican según su nivel de vecindad con la subregión en la que se encuentra el máximo válido, llamada **Núcleo**. Las subregiones vecinas al núcleo se consideran pertenecientes al máximo válido según el criterio descrito en [9].

Extracción del vector de características.

Determinadas las ROI de cada máximo candidato a ser persona, se extrae un vector de características de 6 elementos. Las características usadas se basan en: relaciones antropométricas, la densidad de puntos en las zonas seleccionadas (cabeza, hombros y cuello) y la relación de dimensiones de los ejes de la elipse que forma la cabeza.

Las primeras 5 componentes del vector de características se obtienen analizando las medidas de altura de las subregiones contenidas en la ROI. Dentro de las medidas de altura se define una zona de interés correspondiente a 40 cm por debajo del punto más alto, que posteriormente se divide en 5 franjas de 6 cm que, en caso de ser una persona, comprenden el pelo, la frente y los ojos, la nariz y la boca, la zona debajo del cuello y por último la parte superior del pecho. Esta división disminuye la sensibilidad de las características frente a factores como la altura, la pose, oclusiones, tipo de pelo, etc. Para obtener la última componente del vector de características se hallan los ejes mayor y menor de la elipse que forma la cabeza de la persona y su relación de tamaños es directamente la sexta componente.

Para finalizar se realiza una normalización del vector de características con el fin de disminuir la sensibilidad del mismo a la variabilidad de altura de las personas al acercarse y alejarse de la cámara.

Clasificación.

Para la clasificación entre persona y no persona se utiliza un clasificador SVM multiclase debido a que con un clasificador binario no se obtuvieron los resultados esperados, al haber personas con sombreros, cuya apariencia era muy diferente a las personas sin ellos en la vista cenital. Las clases del algoritmo de clasificación son 3: Clase “persona”, Clase “sombrero” y clase “No-persona”. Para el entrenamiento del clasificador se emplearon datos procedentes de usuarios con diferentes complexiones, alturas y distintos tipos de pelo, con el fin de que la separación de clases fuese lo más robusta posible, obteniendo que la mejor clasificación se realiza con un *kernel* de tipo radial, con los hiperparámetros $C = 0{,}5$ y $gamma = 0{,}00015$, que es el que se emplea también para este trabajo.

Un ejemplo del resultado de la clasificación se muestra en la figura 2.13, en la que se pueden ver tanto detecciones que se han clasificado como personas (en verde), como detecciones que no corresponden a personas (en rojo), debido a que se trata de otros elementos (por ejemplo la mano de una persona) o a que las personas no aparecen completas en la escena.

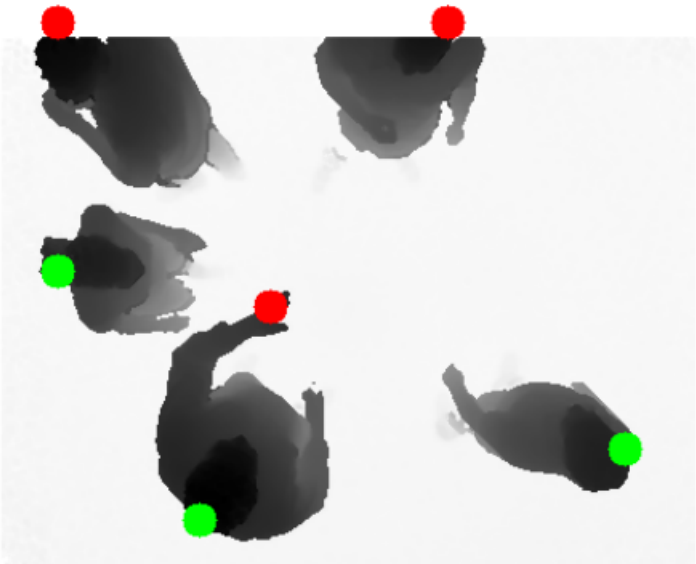


Figura 2.13: Ejemplo de detección de múltiples personas con detecciones válidas (en verde) y no válidas, en rojo, (manos, personas incompletas, etc.)

2.4. Filtro de Kalman

El filtro de Kalman es un algoritmo recursivo de estimación óptima,es decir, con menor error cuadrático medio [10]. Este algoritmo fue desarrollado para poder estimar estados de un sistema no observables a partir de otros que si lo sean, pero que son afectados por ruido blanco.

Debido a su carácter recursivo, es un sistema ampliamente usado en tiempo real en aplicaciones tales como la navegación de vehículos espaciales [51], localización GPS e incluso en econometría [52].

A continuación se exponen los fundamentos matemáticos en los que se basa este algoritmo.

Sea un sistema discreto representado en el espacio de estados cuya ecuación de estado es 2.4:

$$x_k = A_{k-1}x_{k-1} + B_{k-1}u_{k-1} + w_{k-1} \tag{2.4}$$

Y la ecuación de medida 2.5:

$$z_k = H_k x_k + v_k \tag{2.5}$$

donde $x \in \Re^n$ representa el estado que se desea estimar y k el instante de tiempo actual, $w_k \in \Re^n$ es una variable aleatoria que representa el ruido blanco del proceso de valor medio cero y cuya varianza se define como Q_k en el instante k y $v_k \in \Re^m$ es también una variable aleatoria que define el ruido blanco de la medida con media cero y varianza R_k en el instante k.

En la ecuación 2.4 la matriz $\mathbf{A}$, de dimensiones $n \times n$, también llamada matriz de transición y representada como Φ, relaciona el estado en el instante previo x_{k-1} con el estado en el instante actual x_k. Por otra parte la matriz $\mathbf{B}$ de dimensiones $n \times l$ relaciona el estado en k con la entrada $u_k \in \Re^l$. Por último, la matriz $\mathbf{H}$ en 2.5, de dimensiones $n \times m$ relaciona el estado en k con la medida $z_k \in \Re^m$.

El algoritmo se desarrolla en dos etapas: *Predicción* y *Corrección*, como se muestra en el esquema de la figura 2.14, que se explican en detalle a continuación:

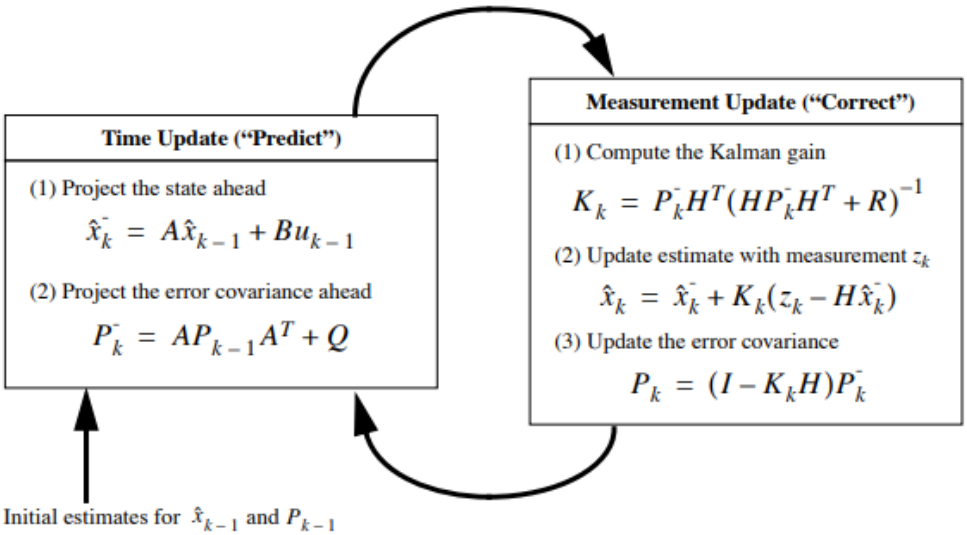


Figura 2.14: Etapas de operación del filtro de Kalman con ecuaciones (Extraídas de[[10]]

2.4.1. Etapa de predicción

En la etapa de predicción, partiendo de unas condiciones iniciales, el algoritmo trata de estimar tanto los estados como la covarianza del error de estimación P_k^- en el instante k para obtener los estados *a priori* que se representan como $\hat{x}_k^- \in \Re^n$

Se definen también el error *a priori* e^- según la ecuación 2.6 y las matrices de covarianza del error de estimación a *a priori* P_k^- (ecuación 2.7) y *a posteriori* P_k (ecuación 2.8).

$$e_k^- = x_k - x_k^- \tag{2.6}$$

$$P_k^- = E[e_k^- e_k^{-T}] = AP_{k-1}A^T + Q \tag{2.7}$$

$$P_k = E[e_k e_k^T] \tag{2.8}$$

2.4.2. Etapa de corrección

El propósito de la etapa de corrección es reducir, en cada iteración del bucle predicción-corrección, el error en la estimación de los estados *a priori*. Para ello, es necesario realizar una comparación entre los estados estimados y los medidos.

En la etapa de corrección se calcula en primer lugar la ganancia del filtro de kalman K_k que es una matriz de tamaño $n \times m$ cuya función es la de minimizar la matriz de covarianza del error de actualización *a posteriori* P_k. La ganancia de Kalman se obtiene mediante la ecuación 2.9, de la cual se puede deducir que: conforme menor es la matriz de covarianza del error de medida R, mayor es la relevancia de la matriz de medida H, es decir, el sistema confía más en las medidas que en la predicción que se esta realizando.

$$K_k = \frac{P_k^- H^T}{H P_k^- H^T + R} \tag{2.9}$$

De esta ecuación también se puede extraer que conforme menor es la matriz de covarianza del error de predicción P_k^- menor es la ganancia de Kalman y por tanto el estado *a posteriori* está mas proximo al estado *a priori* que a la medida realizada por el sistema, es decir, que el sistema confía más en la predicción que en las medidas tomadas.

Una vez obtenida la ganancia de Kalman, se calcula el estado *a posteriori* expresado como $\hat{x}_k$ a través de la ecuación 2.10:

$$\hat{x}_k = \hat{x}_k^- + K_k * (z_k - H\hat{x}_k^-) \tag{2.10}$$

Al término $(z_k - H * \hat{x}_k^-)$ se le denomina *innovación de la medida* (yk en ciertas referencias) o *residuo*. Este término representa la diferencia entre las medidas tomadas z_k y las predichas $(H * \hat{x}_k^-)$. Es justo en este punto en el que se calcula el error que ha cometido el algoritmo.

Por último, se calcula la matriz de covarianza de error *a posteriori* P_k (ecuación 2.11). Esta matriz de covarianza de error es la que en la siguiente iteración del algoritmo se utiliza para calcular la matriz de covarianza de error *a priori*.

$$P_k = (I - K_k H)\hat{(}P)_k^- \tag{2.11}$$

Cabe destacar que el comportamiento del filtro en términos de velocidad de convergencia de las estimaciones, dependerán de los valores asignados a las matrices de covarianza de error de proceso Q y la matriz de covarianza de error de medidas R, así como de lo cerca que estén los valores iniciales de P_k de lo que luego se compute con el algoritmo.

2.5. Clasificador SVM

Se llaman clasificadores a un conjunto de algoritmos que, a través del análisis de características de un conjunto de datos, son capaces de categorizar la información de entrada al algoritmo en una o más clases o conjuntos. Estos algoritmos pueden agruparse según dos grandes criterios:

- **Número de clases**: si los algoritmos pueden clasificar los datos entre dos conjuntos, se denominan *Clasificadores binarios* o bien si pueden hacerlo en más de dos conjuntos, se denominan *Clasificadores multiclase*. Un ejemplo de *clasificador binario* sería el descrito en [53] en que se usa una red neuronal convolucional para clasificar un conjunto de imágenes de perros y gatos. Por otra parte, en [54] se emplea una maquina de soporte vectorial SVM como *clasificador multiclase* para clasificar diferentes tipos de frutos secos (Nueces, almendras, avellanas, castañas y pistachos) en la imagen.

- **Tipo de aprendizaje**: los clasificadores pueden agruparse según el tipo de aprendizaje empleado: supervisado, o no supervisado. El aprendizaje supervisado es un método de análisis de conjuntos de datos etiquetados que usa algoritmos que aprenden iterativamente y a través de los cuales se desarrollan hipótesis que permiten ejecutar la clasificación. Un clasificador basado en aprendizaje supervisado muy utilizado son las redes neuronales, como por ejemplo el *perceptrón multicapa* [55].

  Por otro lado los algoritmos basados en aprendizaje no supervisado infieren patrones de un conjunto de datos no etiquetados con el fin de descubrir la estructura subyacente de dichos datos. Este es el

caso del algoritmo *K-means* [56], que usa una técnica de refinamiento iterativo, y cuyos datos de entrada son el conjunto de datos que se desea clasificar sin etiquetar y el número de clases en que se deben agrupar.

En este TFG el clasificador elegido es una maquina de vectores de soporte SVM [57], un clasificador inicialmente categorizado como binario, desarrollado por Vladimir Vapnik, basado en aprendizaje supervisado. Esto implica que el algoritmo requiere una fase de entrenamiento en la que reciba datos etiquetados en forma de vectores de características, y la clase a la que pertenece cada vector, para poder realizar la clasificación posteriormente.

El propósito del clasificador SVM es el de encontrar un hiperplano que separe las muestras de forma óptima. Se define como hiperplano óptimo aquel cuya distancia a las muestras más cercanas de cada una de las clases es máxima. Para definir este hiperplano se hacen pasar hiperplanos por las muestras más críticas. Estos vectores son paralelos al hiperplano y entre si, siendo además la distancia entre los hiperplanos que pasan por las muestras críticas el margen de la SVM, es decir, el parámetro que se debe optimizar para conseguir la máxima separación.

La siguiente figura muestra un ejemplo, para el caso de una categorización de dos dimensiones, en el que se puede observar el hiperplano óptimo, que es aquel para el que la distancia a las muestras de ambas clases es máximo

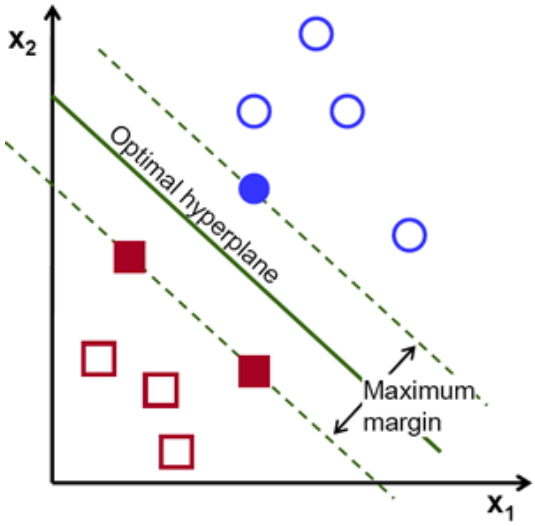


Figura 2.15: Ejemplo de SVM lineal y su plano de separación óptimo en dos dimensiones (Extraída de [[11]])

Este ejemplo muestra el uso de una SVM Lineal para la clasificación de las muestras. No obstante, dichas muestras no siempre podrán separarse con una función lineal como una recta. Es por ello que existen recursos dentro del algoritmo SVM que permiten realizar la transformación de las muestras de entrada que sean no lineales a otro espacio de características que facilite su categorización.

A continuación, se describe con detalle la base teórica y el funcionamiento de la SVM lineal y binaria. Posteriormente se explica como extender la SVM para la clasificación multiclase y se finaliza el apartado presentando las alternativas para datos que no son linealmente separables.

2.5.1. SVM lineal

Como se ha comentado previamente, en este apartado se expone el desarrollo matemático en el que se basa el algoritmo SVM de clasificación binaria y lineal.

En la figura 2.16 se aprecia que el hiperplano $H0$ se encuentra a la misma distancia de los vectores de soporte $H1$ y $H2$ y que además, dicha distancia es máxima. Es por ello que $H0$ se considera el hiperplano óptimo para este caso en particular.

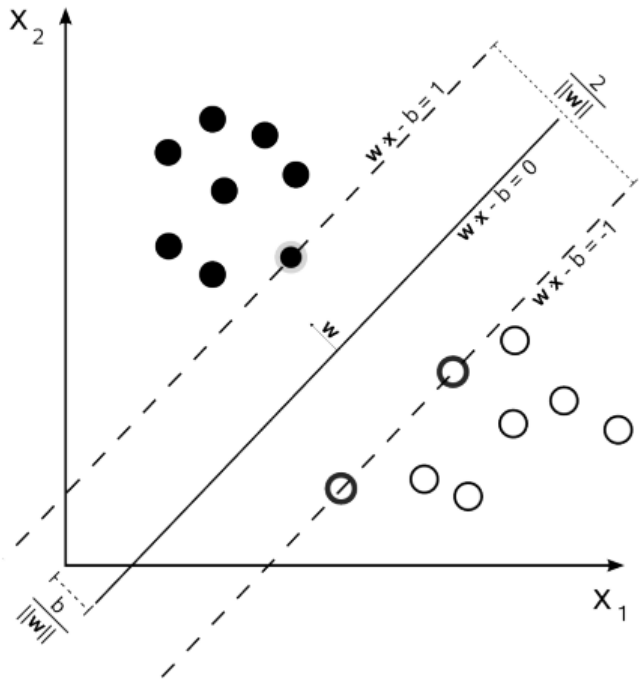


Figura 2.16: Ejemplo en 2D del plano óptimo de separación y margen de una SVM lineal (Extraído de [[9]])

Los ejes de la gráfica corresponden a cada una de las características observadas en las muestras.

Sean x_i las muestras introducidas en el algoritmo e y_i las clases entre las cuales se desea categorizar las muestras, se puede definir el hiperplano (w, b) con la ecuación 2.12:

$$wx_i + b >< y_i \;/\; \begin{cases} (y_i = -1) \quad (\forall x_i \in C_i = B) \\ (y_i = 1) \quad (\forall x_i \in C\text{I} = A) \Rightarrow \exists (w*, b*)/w * x + b* = 0 \rightarrow H0 \end{cases} \tag{2.12}$$

donde w es el vector normal al hiperplano que se busca y b es el *offset* o *bias*, es decir, la distancia del hiperplano óptimo al origen de coordenadas.

Una vez definido el hiperplano óptimo $H0$ se definen los hiperplanos $H1$ y $H2$ como los paralelos a $H0$ que pasan por las muestras críticas o *vectores de soporte* de cada clase (ecuación 2.13).

$$\begin{aligned} H1 &= wx - b = 1 \\ H2 &= wx - b = -1 \end{aligned} \tag{2.13}$$

Una vez definidos los hiperplanos que pasan por los vectores de soporte, la distancia d_{h0h1} de $H0$ a $H1$ y $H2$, se puede determinar mediante la ecuación 2.14

$$d_{h0h1} = \frac{b}{|w|} \tag{2.14}$$

Y, considerando ésta ecuación, el margen o distancia d_{h1h2} entre los hiperplanos $H1$ y $H2$, es la mostrada en la ecuación 2.15:

$$d_{h1h2} = \frac{2}{|w|} \tag{2.15}$$

De la ecuación anterior se deduce, que para hacer máxima esa distancia es necesario minimizar el término w. Es en este punto en el que se definen dos términos importantes dentro del clasificador SVM: *Soft Margin* y *Hard Margin* según si en la clasificación se admite la aparición de vectores interfranja o no.

2.5.1.1. SVM Hard Margin

Si para la realización de la clasificación se descartan los vectores interfranja (Es decir, las muestras que se encuentran entre los hiperplanos $H1$ y $H2$) se considera que se está aplicando *Hard Margin* con la limitación definida en la ecuación 2.16

$$y_i(wx_i - b), 1 \leq i \leq n. \tag{2.16}$$

En la figura 2.16 se muestra un ejemplo de clasificación usando *Hard Margin*, ya que no existe ninguna muestra situada en la interfranja.

El problema de optimización del valor de w queda resuelto al usar multiplicadores de Lagrange, un método de resolución de problemas de optimización sujetos a restricciones, como el caso de las SVM con *Hard margin* aplicado. Usando este método se obtiene la función de Lagrange de la ecuación 2.17, que al desarrollarla acaba resultando en un sistema de ecuaciones como el mostrado en 2.18. Si los datos de entrada pueden separarse linealmente, el sistema de ecuaciones tendrá una única solución.

$$L(w, b, \alpha) = \frac{1}{2}w^T w - \sum \alpha_i[y_i(w^T x_i + b) - 1] \tag{2.17}$$

$$\begin{aligned} \frac{\delta L}{\delta w} &\rightarrow w = \sum_{i=1}^{N} \alpha_i y_i x_i \\ \frac{\delta L}{\delta w} &\rightarrow w = \sum_{i=1}^{N} \alpha_i y_i = 0 \end{aligned} \tag{2.18}$$

El sistema de ecuaciones planteado permite obtener la ecuación 2.19 y sustituyendo la misma en la ecuación 2.7 se obtiene la función a maximizar $Q(\alpha)$, que se observa en 2.20. A esta ecuación se le aplica la restricción correspondiente a la segunda ecuación del sistema de la ecuación 2.18, en la cual cada α_i representa uno de los vectores de soporte, siendo $\alpha_i \geq 0$ *con* $i = 0, ..., N$.

$$w^T w = w^T \sum_{i=1}^{N} \alpha_i y_i x_i = \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i x_j \tag{2.19}$$

$$L(w, b, a) = Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i x_j \tag{2.20}$$

Con este desarrollo matemático se concluye que el hiperplano óptimo $H0$ depende, para la aplicación del *Hard Margin*, de los vectores de soporte obtenidos en la etapa de entrenamiento de la SVM.

Conocidos los vectores de soporte α_i se pueden calcular los coeficientes del hiperplano (w^* y b^*) mediante las ecuaciones 2.21 y 2.22 respectivamente.

$$\alpha_i \rightarrow w^* = \sum_{i=1}^{N} \alpha_i^* y_i x_i \tag{2.21}$$

$$b^* = 1 - w^{*T} x_s \tag{2.22}$$

2.5.1.2. SVM Soft Margin

La aplicación de la técnica de margen blando o *Soft Margin* tiene como objetivo paliar la imposibilidad de encontrar un hiperplano óptimo al aplicar *Hard Margin*, debido a la restricción de la no aparición de vectores en el espacio existente entre los hiperplanos $H1$ y $H2$, que pasan por los vectores de soporte. El *Soft Margin* pretende encontrar un hiperplano que sea capaz de separar, con el mínimo de vectores en la interfranja posibles, los datos de entrada. Estos vectores interfranja se consideran errores de clasificación.

A continuación se muestra un ejemplo de clasificación binaria aplicando *Soft Margin*:

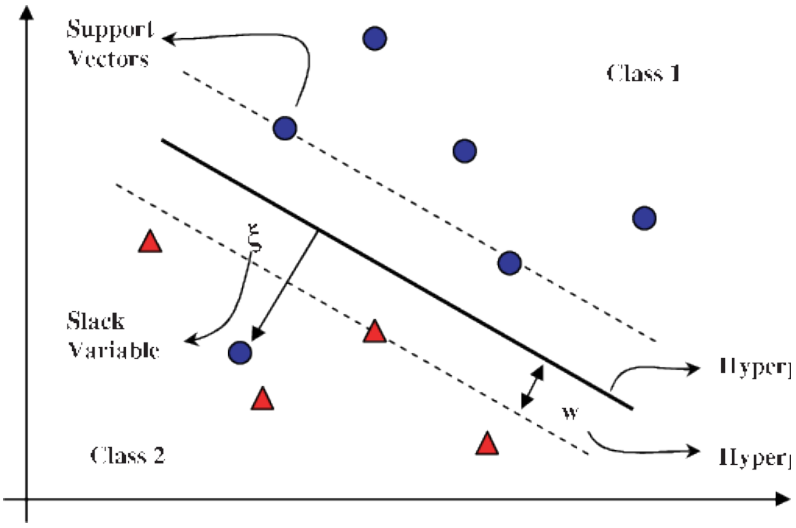


Figura 2.17: Ejemplo de clasificación binaria con SVM aplicando *Soft Margin* [Extraída de [12]].

Para llevar a cabo esta técnica es necesario definir unas variables de "holgura" $h_i = 1, ..., N$ que permiten modular el número de clasificaciones erróneas aceptables para la clasificación, o rigidez del criterio de clasificación (ecuación 2.23).

$$y_i(w^T x_i + b) \geq 1 - h_i, i = 1, ..., N \tag{2.23}$$

donde:

$$h_i \geq 0 \ \forall_i \begin{cases} 0 \leq h_i \leq 1 \rightarrow clasificación\ correcta \\ h_i > 1 \rightarrow clasificación\ incorrecta \end{cases} \tag{2.24}$$

De esta manera, en la nueva función que se debe optimizar: $\Omega(w, h)$, se introduce un término C que permite hacer balance entre el error cometido y el margen máximo obtenible. En el caso concreto del uso de la función de penalización lineal, la función $\Omega(w, h)$ antes citada es la mostrada en la ecuación 2.25, con la restricción definida en la ecuación 2.26:

$$\Omega(w,h) = \frac{1}{2}|w|^2 + C\sum_{i=1}^{N} h_i \tag{2.25}$$

$$y_i(wx_i - b) \geq 1,\ 1 \leq i \leq n. \tag{2.26}$$

Se puede apreciar que las ecuaciones son similares a las expuestas en la sección 2.5.1.1, por lo cual, siguiendo el mismo procedimiento, aplicando los multiplicadores de *Lagrange*, las variables de ajuste h_i desaparecerían y solo el parámetro C queda como restricción. Esta constante establece la relación entre la complejidad del modelo y los errores que se permite cometer. Cuanto menor sea el valor de C, más errores se cometerán a cambio de encontrar un hiperplano con mayor distancia a los vectores de soporte.

2.5.2. SVM Multiclase

Como se ha comentado previamente, es posible extender el uso de la SVM como clasificador multiclase empleando dos estrategias principales:

- "*one-against-all*": esta estrategia consiste en descomponer un conjunto de datos con M-clases en una serie de problemas de únicamente dos clases. Para ello, se generan M modelos de SVM binarias cuya función es la de encontrar el hiperplano óptimo para dividir los datos entre una de las M clases y resto de muestras, cuyo conjunto forman la otra clase. Para el entrenamiento de cada SVM se usa la totalidad de los datos de entrenamiento y se realiza un nuevo etiquetado con etiquetas positivas (+1) para la clase singular y negativas (-1) para el conjunto de muestras diferentes de la clase elegida. De cada modelo se extrae un parámetro P que representa la probabilidad de que una muestra pertenezca a la clase etiquetada como (+1). De esta forma una nueva muestra será clasificada según la clase en la cual P sea mayor. Esto se representa en la figura 2.18

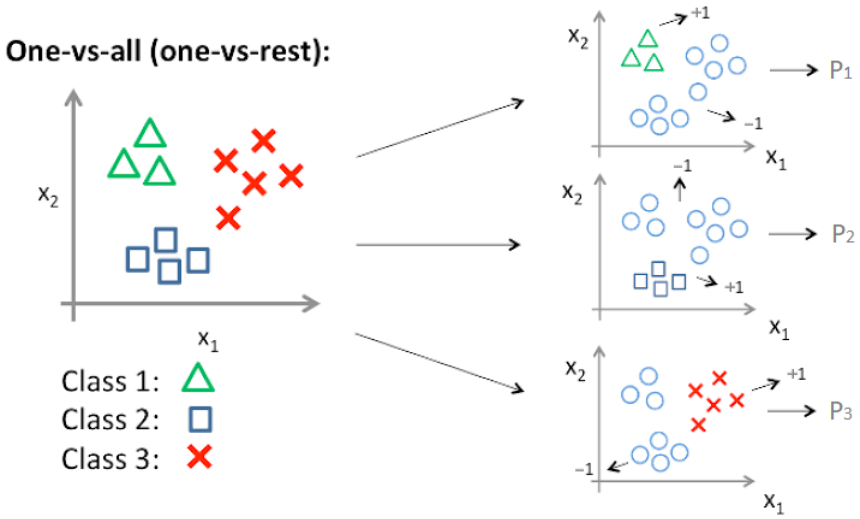


Figura 2.18: Ejemplo de la estrategia "*one-against-all*" con 3 clases.

- "*one-against-one*": esta estrategia es muy similar a "*one-against-all*". Si se dispone un conjunto de datos con M clases, se deben crear $\frac{M(M-1)}{2}$ clasificadores SVM binarios que distinguen entre cada par de clases. Ante una nueva muestra, cada clasificador binario emite un voto, referido a la clase a la que considera que pertenece la muestra. Finalmente, la clase que mayor número de votos obtenga, será la asignada a esa muestra. Se muestra un ejemplo en la figura 2.19 a continuación:

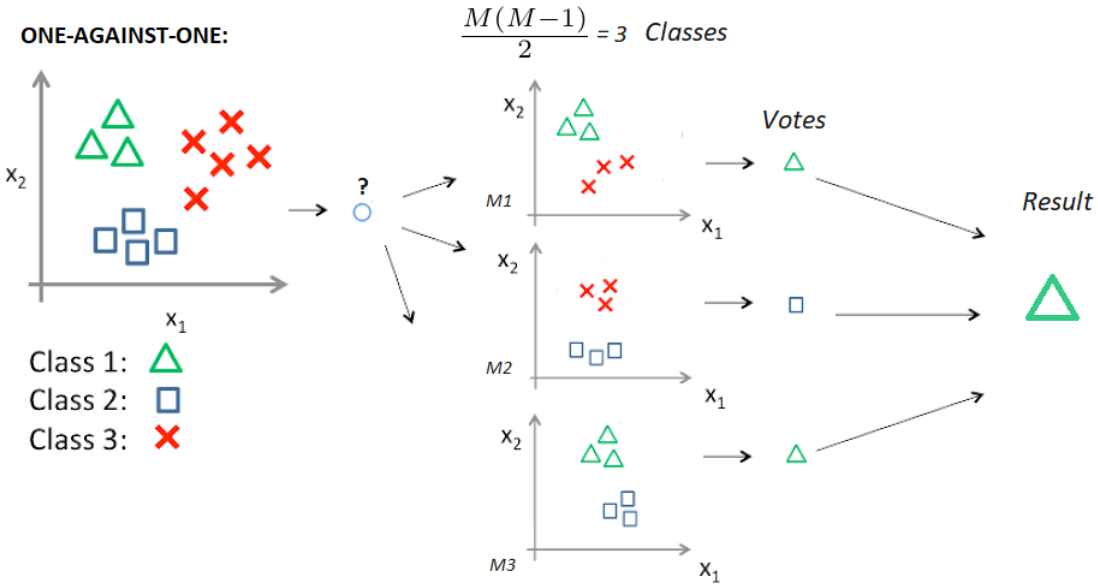


Figura 2.19: Ejemplo de la estrategia "*one-against-one*" en clasificadores SVM.

2.5.3. SVM no lineal

Como se ha comentado previamente, los datos de entrada no siempre pueden separarse linealmente debido a su distribución, que puede ser de carácter no lineal. Es por ello que se requiere otro procedimiento para poder clasificar de manera óptima dichos datos.

Para realizar esta clasificación se recurre a una transformación de los conjuntos de datos de entrada a un espacio de características transformado, mediante el uso de una función de carácter no lineal, que de ahora en adelante se denotará como $\Phi()$. Esta transformación permite ampliar el número de dimensiones del espacio en que se sitúan los datos de entrada (ecuación 2.27) y por tanto, se simplifica la separación de los mismos mediante un hiperplano óptimo de la misma forma que se haría para una SVM lineal.

$$R^n \xrightarrow{\Phi} R^m, m > n \tag{2.27}$$

En la figura 2.20 se muestra un ejemplo de lo explicado anteriormente:

De esta manera y aplicando la técnica de *Soft Margin* también para las muestras en el espacio transformado se obtiene la función que se pretende minimizar $\Omega(w, h)$, equivalente a la mostrada en la sección previa.

$$\Omega(w, h) = \frac{1}{2}|w|^T w + C \sum_{i=1}^{N} h_i \tag{2.28}$$

Teniendo en cuenta las restricciones impuestas por el uso de *Soft Margin*, la función de transformación resultante $\Phi(x)$ sustituye al vector x (ecuación 2.29):

$$\begin{gathered} y_i(w^T \Phi(x_i) + b) \geq 1 - h_i, i = 1, ..., N \\ h_i \geq 0 \ \forall i \end{gathered} \tag{2.29}$$

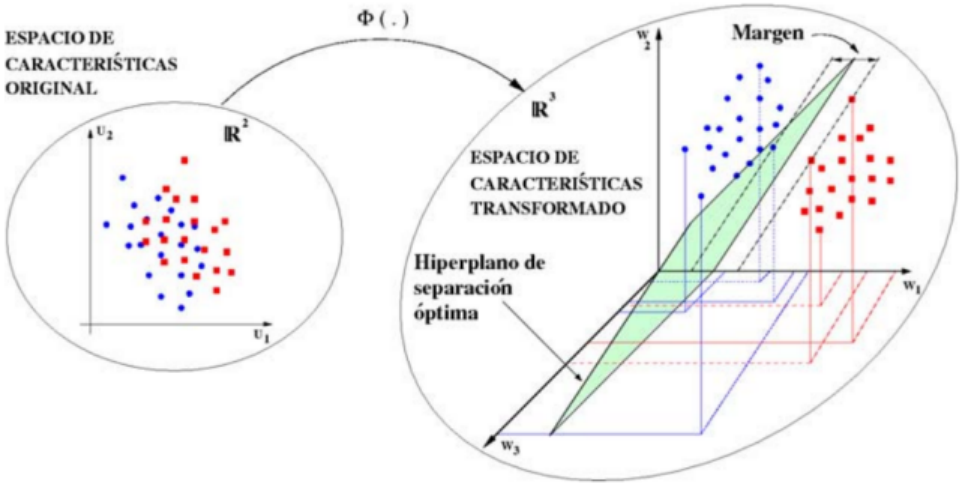


Figura 2.20: Ejemplo de SVM no lineal [Extraída de [13]]

Ésto permite obtener los parámetros (w, b) del hiperplano en el espacio transformado. Cabe destacar que la transformación explícita, $\Phi(x)$, para cada muestra es costosa en términos de computación, por lo que no es aconsejable para un número de dimensiones elevado.

Es por ello que el recurso más usado para realizar la transformación son las funciones denominadas *Kernel*. Un *Kernel* no es si no una función de similaridad que permite al algoritmo de clasificación (SVM en este caso concreto) cuantificar la similitud de las muestras de entrenamiento y las que se desean clasificar. Estas funciones permiten operar en espacios multidimensionales cuyo número de dimensiones es elevado, sin necesidad de computar cada uno de los datos. En su lugar, se computan los productos escalares de parejas de datos en el espacio de características, posteriormente a haber comprobado mediante el teorema de Mercer [58] . Esto es computacionalmente más eficiente que el procesado de cada dato individual. A continuación, en la tabla 2.2 se muestran las características de los *Kernels* más utilizados.

Tipo SVM	Función kernel	Comentario.
Polinómica	$(x^T y + 1)^p$	El valor de p lo especifica el usuario.
RBF Gaussianas	$e^{-\left(\frac{\lvert x - x_j \rvert^2}{2\sigma^2}\right)}$	El valor de σ lo especifica el usuario y es común a todas las funciones kernel.
RBF Exponencial	$e^{-\left(\frac{\lvert x - x_j \rvert}{2\sigma^2}\right)}$	El valor de σ lo especifica el usuario y es común a todas las funciones kernel.
Perceptrón de 2 capas.	$tanh(\beta_0 x^T x_i + \beta_1)$	Sólo ciertos valores de β son válidos.

Tabla 2.2: Tipos de funciones *Kernel* más usados (Extraído de [[15]]).

Capítulo 3

Desarrollo

3.1. Introducción

A lo largo de este capítulo se explican en detalle cada uno de los bloques que conforman el sistema propuesto en este TFG, cuyo diagrama de bloques general se muestra en la figura 3.1 (que ya se ha mostrado en el capítulo 1, pero se repite aquí para facilitar la lectura de la memoria).

Figura 3.1: Diagrama de bloques del sistema completo

Para este sistema se parte de la propuesta para detección robusta de personas a partir de información de profundidad adquirida por una cámara en posición cenital desarrollada en [9], que se ha descrito brevemente en la sección 2.3. A partir de estas detecciones de personas se ha desarrollado un algoritmo de seguimiento en las imágenes mediante un banco de filtros de Kalman que lleva asociado un algoritmo de gestión de los filtros para evitar la sobrecarga computacional del sistema. Por último, usando la información devuelta por el detector de personas y siendo conocida la persona en la escena a la que pertenece esa información gracias al algoritmo de seguimiento, se procede a que un algoritmo clasificador (SVM) muestre las acciones realizadas por cada persona en escena. Cabe destacar que tanto para el algoritmo de detección de personas como para el algoritmo clasificador de acciones se requiere una fase previa de entrenamiento realizada de manera "*offline*" con el fin de obtener los modelos que permitirán a dichos algoritmos realizar la clasificación de manera eficaz.

A continuación, se explican en detalle cada una de las etapas del algoritmo desarrollado en este TFG.

3.2. Algoritmo de detección robusta de personas

Como se ha comentado previamente, para la detección robusta de personas se emplea el algoritmo desarrollado en [9,17] y cuyo fundamento teórico se ha explicado en la sección 2.3.

El algoritmo de detección de personas, tiene como entrada las imágenes de profundidad, adquirida por una cámara de profundidad Kinect v2 ubicadas en posición cenital. Como se ha comentado previamente, puede funcionar tanto con imágenes adquiridas en tiempo real, como con secuencias grabadas previamente.

Este algoritmo proporciona, además del número de personas detectadas en la escena, información útil para los algoritmos de seguimiento de personas y clasificación de acciones que se detallan en los apartados posteriores, como son: el largo de los ejes que componen la elipse que forma la cabeza (vista en posición cenital), el número de puntos alrededor del máximo considerado persona que pertenecen a dicha persona, la altura con respecto al plano del suelo y la posición de la persona en la imagen. A partir de esta información se puede realizar, por ejemplo, la etapa de corrección de los diferentes filtros de Kalman 2.4.2 usando la posición y la altura, se puede calcular la velocidad a lo largo de una o más imágenes para usarla como característica en el algoritmo clasificador de acciones, etc.

La incorporación de la etapa de detección robusta de personas, permite que el algoritmo implementado en este TFG pueda funcionar correctamente en situaciones en que hay varias personas en la escena, incluso si éstas se encuentran muy próximas entre sí, o si aparecen otros elementos que no sean personas, ya que puede discriminar correctamente entre personas y otros elementos en la escena. Esto se puede apreciar en la figura 3.2, en la que se muestran dos imágenes de ejemplo y las detecciones correspondientes. A la izquierda se puede ver una imagen con una sola persona, mientras que a la derecha se presenta un ejemplo con 8 personas, muy próximas entre sí, en la que todas son detectadas correctamente.

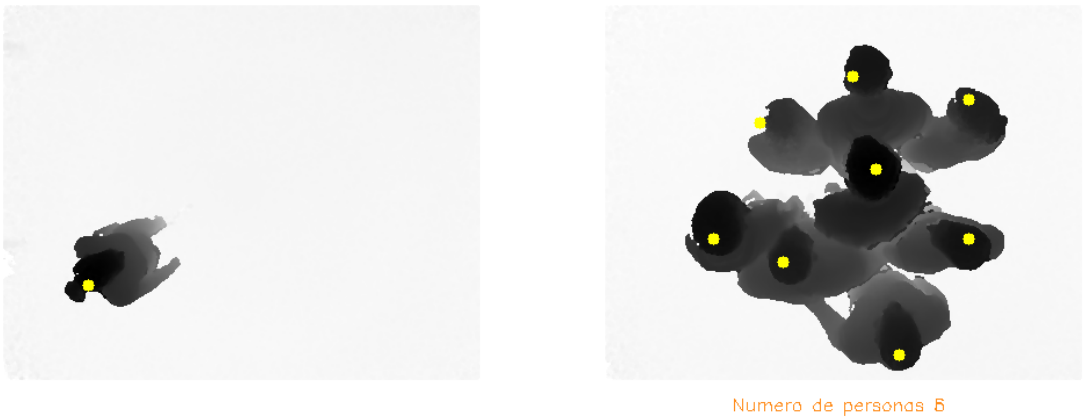


Figura 3.2: Detección de múltiples personas

3.3. Seguimiento de múltiples personas

En esta sección se detalla la algoritmia desarrollada en este TFG, para el seguimiento robusto de un número variable de personas, previamente detectadas en la escena, empleando un banco de filtros de Kalman.

Se precisa un filtro de Kalman para el seguimiento de cada persona presente en la escena, que además permite disminuir el ruido de las medidas obtenidas a través del algoritmo de detección descrito en [9].

En cada uno de los filtros se llevan a cabo las etapas de predicción y corrección del modo descrito en la sección 2.4. Además, debido a que puede haber múltiples personas en la escena, es necesario un proceso de asociación entre las estimaciones de los filtros activos y las detecciones. Este proceso de asociación se realiza siguiendo el concepto del algoritmo del "Vecino más cercano" [59].

Por último, para evitar una sobrecarga computacional innecesaria, es necesario disponer de una gestión de la creación de nuevos filtros y su posterior destrucción si la persona a la que estaban asociados desaparece de la escena. En la figura 3.3 se muestra un esquema general de los pasos a seguir para desarrollar lo explicado anteriormente. Además, cada uno de estos aspectos se explican en detalle en las siguientes secciones.

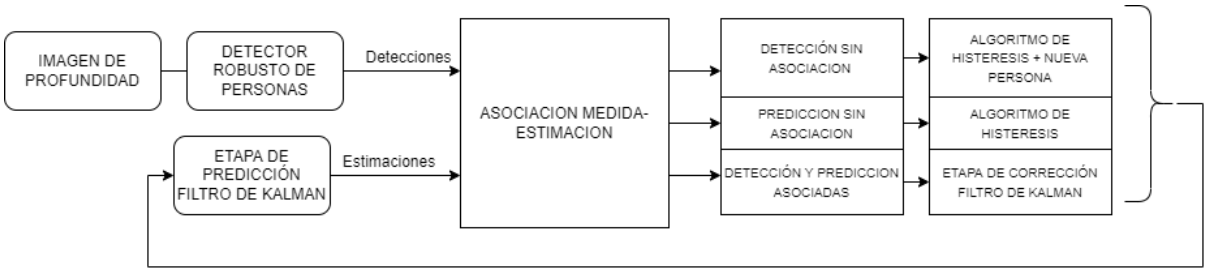


Figura 3.3: Esquema de funcionamiento del seguimiento de múltiples personas.

3.3.1. Definición del modelo de estados

Para llevar a cabo el sistema de seguimiento se requiere realizar estimaciones de la posición y de la velocidad (en x, y, z) de cada una de las personas que aparezcan en la secuencia de imágenes. Estos parámetros conforman el vector de estados de cada filtro de Kalman. No obstante, dado que el detector antes citado [9] únicamente proporciona información de la posición x, y en píxeles y la altura en mm, se considera que la velocidad en los tres ejes son estados no medibles. Además, se puede asumir que la velocidad de las personas es constante. Con estas consideraciones se obtienen las siguientes matrices de estado (ecuación 3.1) y de medida (ecuación 3.2).

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ z_{k+1} \\ v_{xk+1} \\ v_{yk+1} \\ v_{zk+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ z_k \\ v_{xk} \\ v_{yk} \\ v_{zk} \end{bmatrix} + w_k \tag{3.1}$$

$$\begin{bmatrix} z_{xk} \\ z_{yk} \\ z_{zk} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ z_k \\ v_{xk} \\ v_{yk} \\ v_{zk} \end{bmatrix} + v_k \tag{3.2}$$

3.3.2. Predicción

Una vez definido el modelo de estados es necesario llevar a cabo la fase de predicción, que proporciona una estimación de la posición y velocidad de cada filtro para la siguiente imagen de la secuencia. Ésta

se realiza gracias a las funciones referentes al filtro de Kalman dentro de la librería *OpenCV* [60], que incluye numerosas funciones para el procesamiento de imágenes. La proyección de los estados a un instante posterior se realiza cada vez que se recibe una imagen, únicamente para aquellas personas detectadas que cumplan la condición del algoritmo de creación y eliminación de filtros, descrito en el apartado 3.3.5, es decir, personas ya detectadas previamente cuyo filtro de Kalman este activo o bien aquellas detecciones que aun no tengan el filtro activo pero hayan sido detectadas en 5 imágenes consecutivas.

3.3.3. Asociación

Los datos que se obtienen del sistema detector de personas se almacenan en un vector sin estar asociados a una persona en concreto, sino por el orden en que son detectados. Es por ello que se requiere de un proceso de asociación entre las medidas (personas detectadas en la escena) y las estimaciones de los filtros de Kalman, para posteriormente ejecutar la fase de corrección de cada filtro con la medida asignada (en caso de que exista). Se trata de una etapa fundamental para el correcto funcionamiento del sistema de seguimiento de múltiples personas.

Para realizar la asociación entre estimaciones y medidas, por cada estimación existente en la imagen, se calcula la distancia euclidiana (ecuación 3.3) a cada persona detectada. Estas distancias se almacenan en una **matriz de distancias** (ver ecuación 3.4) cuyas filas son las estimaciones del *frame* anterior o personas ya detectadas (E_n) y las columnas las personas detectadas en la imagen actual (D_n). Posteriormente, se analiza la matriz de distancias y se realiza la asignación secuencial de cada estimación a una única detección, siempre y cuando se cumpla que la distancia es menor a un umbral establecido. Se establece un orden de prioridad de asignación siguiendo el criterio descrito a continuación:

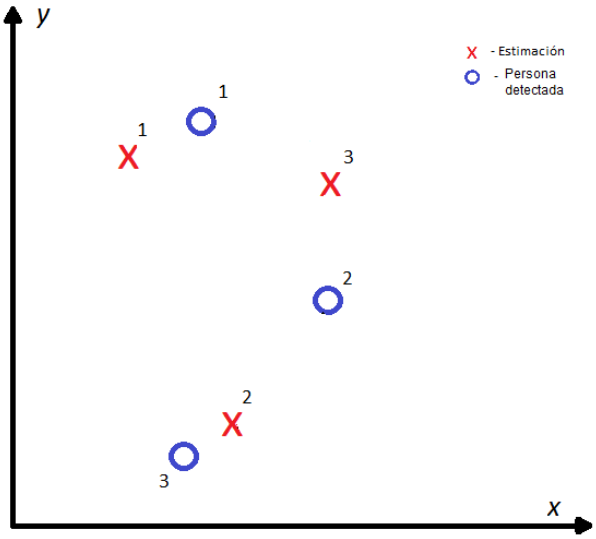
$$d_e = \sqrt{x^2 + y^2 + z^2} \tag{3.3}$$

$$\begin{array}{c|cccc} & D_1 & D_2 & \cdots & D_n \\ \hline E_1 & d_{11} & d_{12} & \cdots & d_{1n} \\ \vdots & \vdots & & \ddots & \vdots \\ E_n & d_{n1} & d_{n2} & \cdots & d_{nn} \end{array} \tag{3.4}$$

1. En primer lugar se asignan las estimaciones que tienen una única detección candidata a realizar la asociación.

2. A continuación, se comprueban las estimaciones que tienen más de una detección candidata. Dentro de las posibles detecciones candidatas se descartan las que ya han sido previamente asignadas y luego se realiza la asociación por orden de menor distancia.

La situación ideal es que en escena haya tantas estimaciones de filtros de Kalman como personas detectadas , en cuyo caso se realiza la asignación de manera normal y se procede a analizar el siguiente *frame*. Esta situación descrita se ilustra en el ejemplo de la figura 3.4, en la que se muestran las posiciones de tres estimaciones de filtros de Kalman (cruces rojas) y tres detecciones (círculos azules), y la matriz de distancias, destacando en negrita las menores distancias y con sombreado gris las asignaciones realizadas. Para este ejemplo se elige que el umbral sea de 10 píxeles. De esta forma, la estimación 1 se asigna con la primera detección ya que la menor de las distancias al resto de detecciones es de 7 píxeles, la estimación 2 se asigna con la tercera detección y así sucesivamente.



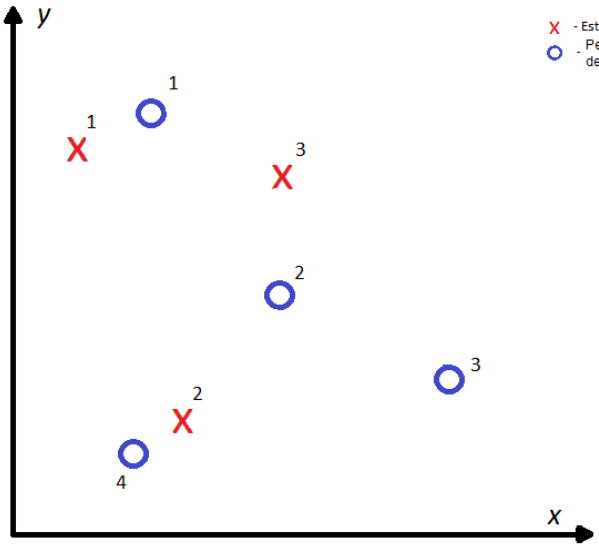
Matriz de distancias

		Persona Detectada		
		P1	P2	P3
Estimación	E1	**7**	13	17
	E2	20	11	**5**
	E3	**10**	**10**	16

Figura 3.4: Ejemplo de asociación de 3 estimaciones con 3 detecciones.

No obstante esto no siempre ocurre y da lugar a los siguientes casos:

- **Que existan más detecciones que estimaciones**. En este caso, tras realizar el resto de asignaciones, la detección que quede sin asignar se considerará una nueva persona detectada y por tanto se requerirá un filtro de Kalman adicional para realizar el seguimiento. Esto puede apreciarse en la figura de ejemplo 3.5. Considerando de nuevo un umbral de 10 píxeles, se aprecia que la única detección no asignada es la tercera. Esto puede haber sido causado por una falsa detección de una persona o bien porque una nueva persona haya entrado en escena, lo que implicaría la creación de un nuevo filtro de Kalman para realizar el seguimiento de la trayectoria de esa persona.



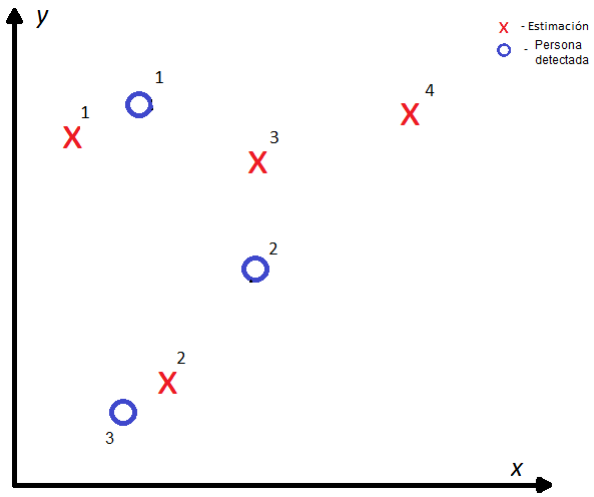
Matriz de distancias

		Persona Detectada			
		P1	P2	P3	P4
Estimación	E1	**7**	13	18	23
	E2	17	11	16	**5**
	E3	**10**	**10**	15	17

Figura 3.5: Ejemplo de asociación de 3 estimaciones con 4 detecciones.

- **Que existan más estimaciones que detecciones**. Esto puede suceder cuando una persona abandona la escena o bien si en una imagen no se detecta correctamente a la persona por diversas causas, como puede ser el caso de la figura de ejemplo 3.6. En el caso expuesto la estimación 4 está demasiado alejada como para poder ser asociada con las detecciones 1 ó 2. En caso de no realizar

la asociación, se mantiene la predicción del filtro de Kalman durante un número determinado de imágenes, a partir del cúal se detiene el seguimiento de esa persona.



Matriz de distancias

		Persona Detectada		
		P1	P2	P3
Estimación	E1	**7**	11	18
	E2	17	10	**5**
	E3	11	**10**	18
	E4	17	15	23

Figura 3.6: Ejemplo de asociación de 4 estimaciones con 3 detecciones.

3.3.4. Corrección

Como se muestra en la figura 3.3, para los filtros de Kalman a los que se les ha asociado una detección en la etapa de asociación se realiza la etapa de corrección del filtro de Kalman. Esta etapa consiste en utilizar la medida de posición devuelta por el algoritmo de detección como corrección a la estimación realizada por el filtro, disminuyendo para la siguiente imagen el error cometido en la estimación.

3.3.5. Creación y eliminación del filtros de Kalman

Dado que el número de personas a seguir es variable, es necesario que se realice una gestión de en qué momentos es necesario que el filtro esté activo recibiendo información de los estados para realizar la predicción y la corrección de los mismos y cuando debe detenerse. Para ello se ha diseñado un sistema de creación y eliminación de filtros de Kalman con "histeresis" que determina en qué instante se debe iniciar un nuevo filtro (que pasa al estado activo) y cúando se debe eliminar. Este sistema impide que un filtro se inicie ante una detección espuria, por lo que es necesario que una persona se detecte al menos en cinco imágenes consecutivas antes de iniciar un filtro, y permite al sistema eliminar filtros ya activos cuando haya cinco imágenes o más en las que no se ha detectado la persona a la que dicho filtro estaba asignada.

Para llevar a cabo este sistema de "histeresis" se plantea un algoritmo basado en contadores de detección que aumentan si la persona ha sido detectada en la escena actual, tal y como se indica en el apartado anterior 3.3.3 o disminuyen si la persona no ha sido detectada.

Para comenzar el seguimiento de una persona, se requiere que la estimación asignada a esa persona sea asociada con una detección durante cinco imágenes consecutivas, con un máximo de una imagen entre ellas en que no se detecte. Es decir, si una persona se detecta, por ejemplo, en dos imágenes seguidas, luego en la siguiente no es detectada y posteriormente se vuelve a detectar en tres imágenes consecutivas, el filtro de Kalman comenzará. Del mismo modo, para evitar desactivar los filtros de Kalman indebidamente, una persona debe no ser detectada en imágenes consecutivas siguiendo el mismo criterio que para el comienzo del seguimiento.

3.3.6. Resultados al aplicar el algoritmo

A continuación se muestran imagenes extraidas del algoritmo de seguimiento de personas. En primer lugar se muestra en la figura 3.7 una comparación entre la imagen devuelta por el algoritmo de detección de personas y el utilizado para el seguimiento de las mismas:

Figura 3.7: Imagen devuelta por el detector de personas (izquierda) y por el algoritmo de seguimiento (Derecha)

El algoritmo de seguimiento de personas muestra por una parte la predicción de la posición en que se encontrará la detección de la persona (Cuadrado verde) y por otra parte la corrección realizada con la medida de posición tomada del algoritmo de detección (Cruz azul). En la figura 3.8 a continuación se muestra cómo el seguimiento se puede realizar para múltiples personas:

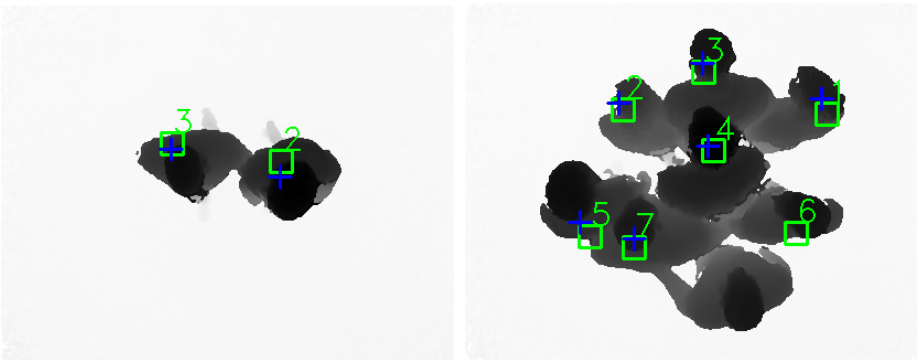


Figura 3.8: Seguimiento de múltiples personas mediante filtros de Kalman

E incluso se puede representar la trayectoria que cada persona está realizando, tal y como se muestra en la figura 3.9:

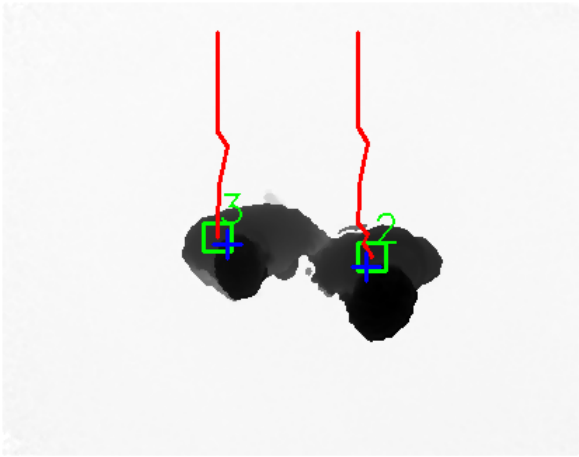


Figura 3.9: Trayectoria realizada por 2 personas seguidas por los filtros de Kalman

3.4. Clasificación de acciones

En esta sección se detallan cada una de las fases llevadas a cabo para realizar la clasificación de acciones de las personas detectadas en escena mediante un clasificador SVM multiclase como el descrito en la sección 2.5.2. En la figura 3.10 se muestra un diagrama general de las etapas que conforman el clasificador desarrollado.

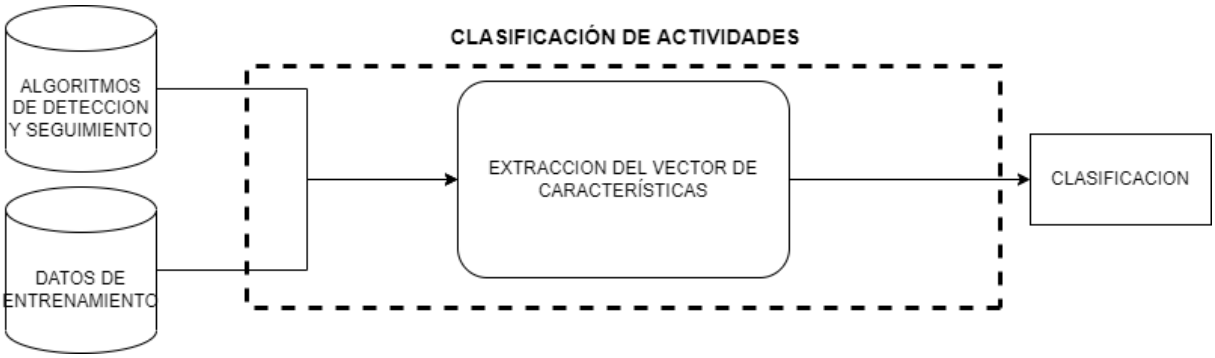


Figura 3.10: Diagrama de etapas para la clasificación de acciones.

Para realizar la clasificación de acciones, se precisa entrenar el algoritmo de clasificación con el fin de obtener un modelo fiable. En primer lugar se definen las acciones a clasificar. Tras ello, se analizan las características de las acciones de forma que se puedan relacionar con los datos obtenidos de los algoritmos de detección y de seguimiento de personas. Luego, dichos datos se agrupan en vectores de características que se introducen en el algoritmo clasificador SVM para realizar el entrenamiento, en la fase *offline*, que se realiza una única vez. Esto genera un modelo de clasificación, cuya fiabilidad debe comprobarse en última instancia viendo si cumple los objetivos esperados y, en caso de no hacerlo, empezar de nuevo este proceso eligiendo un nuevo vector de características. Tras validar el modelo obtenido, el clasificador se pone en funcionamiento en la fase *online*.

3.4.1. Acciones a detectar

Para el sistema propuesto en este TFG se consideran cuatro acciones: andar, correr, caer y parar. Estas son las clases entre las que el algoritmo clasificador SVM debe discernir. Para el entrenamiento y

validación del clasificador, se dispone de una base de datos de secuencias de imágenes etiquetadas en las que diferentes personas realizan cada una de las acciones de interés [35]. Las principales características de esas acciones se describen a continuación.

- **Andar**: se caracteriza por una variación lenta de la posición en x, y de la persona detectada en escena y variación prácticamente nula de la altura h. A continuación en la figura 3.11 se muestran tres imágenes correspondientes a la acción descrita:

Figura 3.11: fig: Secuencia de imágenes de una persona realizando la acción "Andar".

- **Correr**: la variación de la posición es más rápida que en caso de la acción "Andar". Por otra parte, al correr no se está completamente erguido, por lo que el valor de altura es menor en todas las personas. En la figura 3.12 se muestra una captura de una persona realizando la acción "Correr"

Figura 3.12: Captura de una persona realizando la acción "Correr".

- **Caer**: al caer, la altura de la persona disminuye considerablemente hasta el punto de, por como esta diseñado el algoritmo de detección de personas, perder la información de la altura dado que no se detecta nada por debajo de un umbral de 60 cm con respecto al plano del suelo. En la figura 3.13 a continuación se muestra la imagen de una persona que ha caído al suelo.

Figura 3.13: Imagen de una persona realizando la acción "Caer".

- **Parar**: Una persona parada mantiene su posición en un punto de la escena, por lo cual el módulo de la velocidad en los 3 ejes de la persona es prácticamente nula.

Como se puede observar, tres de las cuatro acciones comparten la velocidad de la persona como característica principal. La acción "Caer" por su parte puede ser detectada principalmente por la variación en los valores de posición en z o bien por información nula en determinadas imágenes cuando la persona se encuentra tendida en el suelo.

3.4.2. Elección del vector de características

Como se ha comentado anteriormente, los datos de entrada se obtienen de las etapas de detección y seguimiento de personas, que entre otra información proporcionan la posición en x, y de la persona detectada y su altura h, que permite obtener información de la cámara de profundidad Kinect v2, así como el número de puntos detectados alrededor del máximo considerado persona. Además de la información del detector, en el vector de características se planteó incorporar información de velocidad. Estas características mencionadas son las consideradas más relevantes para el desempeño de la clasificación . Se consideró extraer la velocidad estimada del filtro de Kalman asociado a cada persona, pero al ser un estado estimado no medible resultó ser demasiado ruidoso. Por ello la velocidad de cada persona se calcula como la diferencia de posición en cada eje en un tiempo determinado, según muestra la ecuación 3.5:

$$v = \frac{p_k - p_{k-1}}{\Delta t} \tag{3.5}$$

donde p es la posición en cualquiera de los tres ejes(x, y, z), k representa el instante de tiempo y Δt es el incremento de tiempo que para este sistema es el tiempo entre imágenes: 1/30 s.

Para la elección del vector de características, se han realizado diversos experimentos utilizando para ello los datos disponibles. En concreto, se dispone de un total de 28 secuencias de imágenes de profundidad etiquetadas,que incluyen a 7 personas diferentes, realizando cada una de las 4 acciones mencionadas previamente: andar, correr, caerse y permanecer parado.

Para facilitar la realización de experimentos, los datos obtenidos del detector de personas y de los filtros de Kalman se han almacenado en un archivo de tipo Valores separados por coma, *Comma-separated values* (CSV). Además, estos deben normalizarse antes de pasar por el clasificador con el objetivo de evitar distorsionar las diferencias en los intervalos de valores sin perder información. Este proceso facilita el correcto modelado del clasificador.

Es habitual en procesos de clasificación como el llevado a cabo en este sistema dividir el conjunto de datos en 2 partes con el fin de obtener un modelo fiable:

- **Entrenamiento**: que en este caso supone el 70 % de los datos disponibles, que a su vez deben dividirse en 2 partes. Una parte para el entrenamiento propiamente dicho, ajustando tanto los posos como los hiperparámetros y el *Kernel* óptimos. Mientras que, la otra parte de los datos, llamados **datos de validación**, se usan para comprobar la precisión del modelo. Para este trabajo, el 70 % de datos de entrenamiento se han dividido, a su vez, en un 67 % datos de entrenamiento y un 33 % de validación.

- **Evaluación**: los datos de test permiten evaluar la validez del modelo, en condiciones realistas, mediante diferentes métricas como las matrices de confusión, la precisión, recall, etc.

Para la selección del vector de características, se realizó un análisis, evaluando diferentes vectores y analizando el efecto de cada uno de ellos, sobre los resultados del algoritmo para el subconjunto de datos de validación.

En primer lugar se planteó usar un vector de 7 características que incluía: el número de puntos pertenecientes a la persona detectados por el algoritmo de detección, posición x, posición y, altura de la persona y velocidad calculada en los 3 ejes (ecuación 3.6).

$$V1_{car} = [x_1, y_1, h_1, dx_1, dy_1, dh_1, np_1] \tag{3.6}$$

donde los subindices indican el vector de características al que pertenecen, en este caso al pertenecer al primer vector de características utilizado tienen el subindice 1.

Tras analizar los resultados se determinó que existía una elevada confusión, especialmente entre las acciones «andar» y «correr» (además de una baja tasa de aciertos en las acciones «andar» (35 %), «correr»(0.1 %) y «caer» (54.63 %).

Con el objetivo de incrementar la información acerca de la evolución temporal de las acciones, se modificó el vector de características, generando un nuevo vector de 14 elementos, obtenidos concatenando las características anteriores para dos imágenes consecutivas (ecuación 3.7). Los resultados con este nuevo vector mostraron una notable mejora, especialmente en las acciones de “Parar”, “Andar” y “Caer”, no obstante, las tasas de acierto obtenidas para cada clase continuaban siendo inferiores a las deseadas: «andar» 58.41 %, 0 % correr y 79 % caer. Se obtuvo un 100 % de acierto en las muestras de test para la acción «parar».

$$V2_{[car]} = [x1_2, y1_2, h1_2, dx1_2, dy1_2, dh1_2, np1_2, x2_2, y2_2, h2_2, dx2_2, dy2_2, dh2_2, np2_2] \tag{3.7}$$

En esta ecuación, el número que acompaña a la característica indica a que imagen de las dos concatenadas pertenece, 1 para las características de la primera imagen y 2 para las de la segunda.

El siguiente paso consistió en realizar un análisis de las características para determinar cuáles tienen una mayor importancia en la clasificación de acciones, concluyendo que la posición (x, y) de la persona no aporta información útil para la clasificación, por lo cual se eliminaron del vector de características. Al evaluar los resultados con el nuevo vector con 5 características (Ecuación 3.8), así como con un vector de 10 características resultado de concatenar los datos para dos imágenes consecutivas (Ecuación 3.9), se obtuvo una leve mejora en la predicción de la acción «‘correr» (41.66 % de aciertos), no obstante las tasas de aciderto aun estaban fuera del margen tolerable (por debajo del 90 %), por lo que ambos vectores se descartaron.

$$V3_{[car]} = [h_3, dx_3, dy_3, dh_3, np_3] \tag{3.8}$$

$$V4_{[car]} = [h1_4, dx1_4, dy1_4, dh1_4, np1_4, h2_4, dx2_4, dy2_4, dh2_4, np2_4] \tag{3.9}$$

Posteriormente se sustituyeron las velocidades en los ejes x e y, por el módulo de dichas velocidades, al considerar que es la velocidad del movimiento, y no su dirección, lo que mejor caracteriza la acción realizada, resultando un vector de tres características (Ecuación 3.10). Sin embargo, los resultados empeoraron con respecto al vector anterior, debido a la variabilidad del módulo de la velocidad en cada instante. Por ello se procedió a realizar una media móvil con diferentes tamaños de ventana de muestras para el módulo de velocidad, incluyendo esta media en el vector de características. Para determinar el valor más adecuado para la ventana, se realizaron pruebas con ventanas de 5, 10 y 20 muestras para cada acción, obteniendo los resultados mostrados en las figuras 3.14, 3.15 y 3.16.

En las figuras se aprecia que al aplicar la media móvil los valores de velocidad para cada acción se mantienen prácticamente constantes (A excepción de los picos que aparecen producto del cambio de video y persona). Se comprueba que en la imagen 3.15 el módulo promediado de la velocidad para la acción de "Andar" es mucho menor que para la acción "Correr" y que para la acción "Parar" el valor de esta característica es cercano a cero. No obstante, para las otras imágenes esta diferencia se aprecia en menor medida. Por último, los valores dispares obtenidos para la acción "Caer" indican que otras características serán necesarias para clasificar con éxito esta acción.

$$V5_{[car]} = [h_5, np_5, |v_{xy}|_5] \tag{3.10}$$

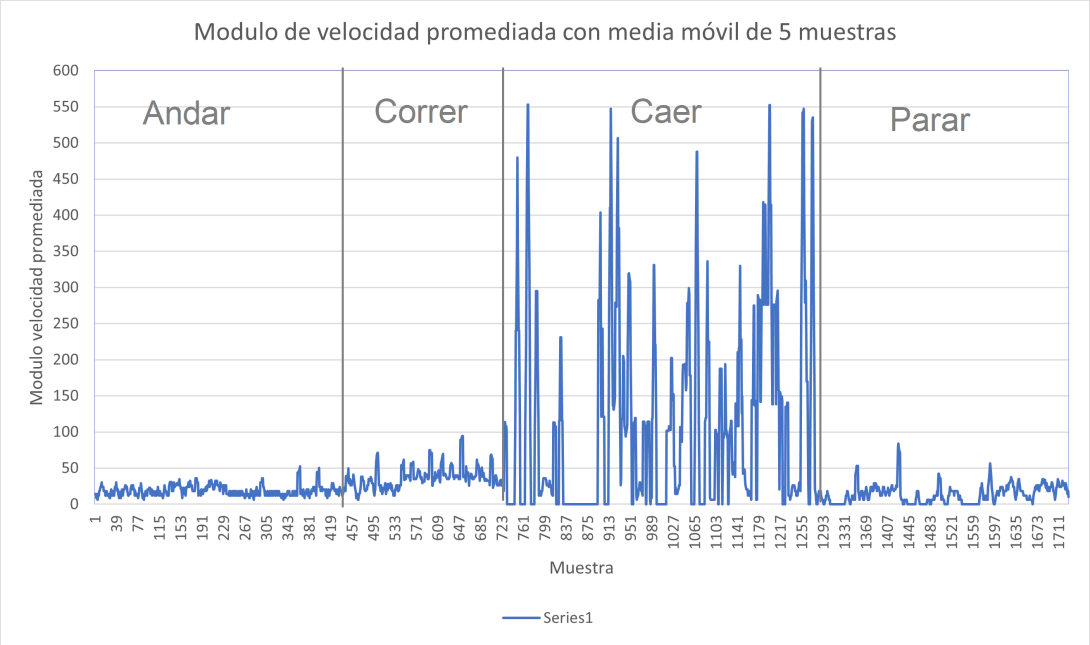


Figura 3.14: Gráfica resultado de aplicar una media móvil con ventana de 5 muestras al módulo de velocidad en x, y.

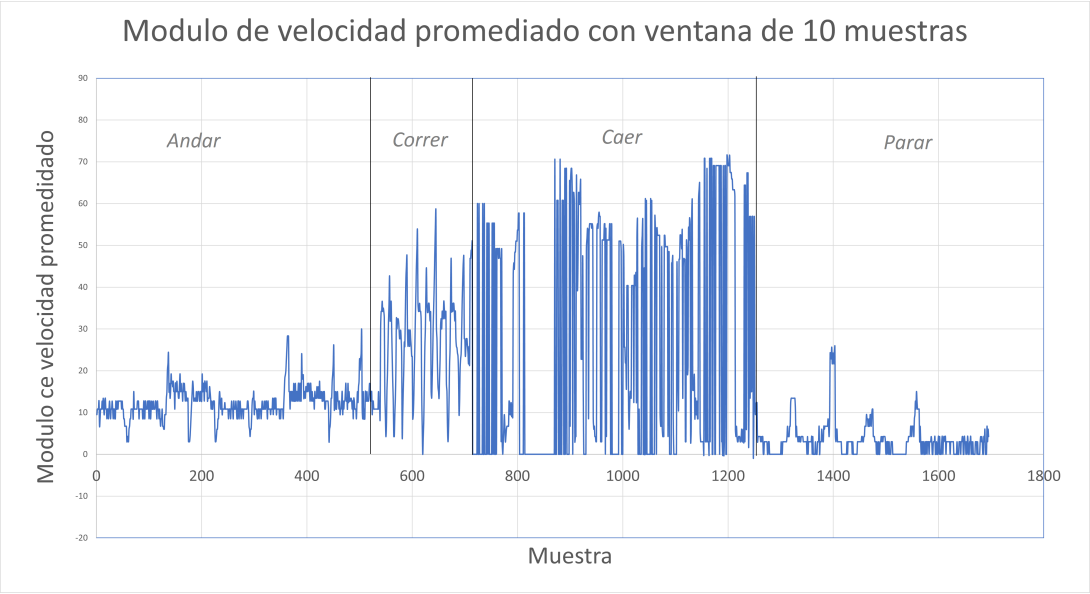


Figura 3.15: Gráfica resultado de aplicar una media móvil con ventana de 10 muestras al módulo de velocidad en x, y.

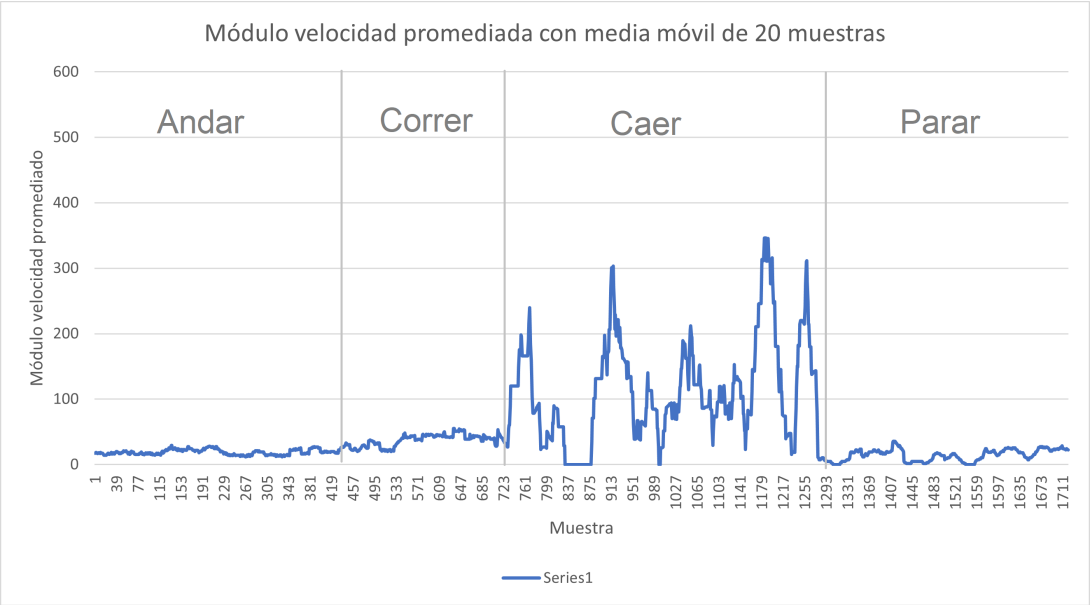


Figura 3.16: Gráfica resultado de aplicar una media móvil con ventana de 20 muestras al módulo de velocidad en x, y.

Tras el análisis realizado, finalmente se definió el vector de características mostrado en la ecuación 3.10 cuyos elementos son: Altura de la persona (coordenada z), Puntos pertenecientes a la persona devueltos por el algoritmo de detección y el módulo de la velocidad calculada, en x e y, promediado según una ventana móvil de 10 muestras. Los resultados obtenidos se presentan y analizan con detalle en la sección 4.

3.4.3. Clasificación de acciones

Para la implementación del clasificador SVM, se ha empleado la librería *Scikit-Learn* [61], una librería de software libre para aplicaciones de *Machine Learning*. Esta herramienta no solo permite normalizar

los datos, sino que también permite realizar la división automática de los mismos en los conjuntos de *entrenamiento*, *validación* y *test*. Por otra parte, esta librería también dispone de las funciones necesarias para generar un modelo de SVM para clasificación multiclase. *Scikit-Learn* usa la estrategia de clasificación multiclase "*one-versus-one*", explicada en la sección 2.5.2.

Para generar el modelo, es necesario definir el tipo de *Kernel* y los hiperparámetros que determinan el funcionamiento de cada uno de ellos, es decir, C (mencionado previamente en 2.5.1.2) y *gamma*:

- **Coeficiente C**: este parámetro establece la relación de compromiso entre la complejidad del modelo y el error cometido en la clasificación de las muestras. Valores pequeños de C permiten al modelo cometer más errores a cambio de obtener más fácilmente el hiperplano óptimo, mientras que valores altos implican un ajuste rígido del hiperplano.

- **Coeficiente Gamma**: el parámetro *Gamma* indica el peso que las muestras de entrenamiento tienen sobre el modelo, es decir, cuánto se ajusta el modelo a los datos de entrenamiento. Un mal ajuste de este parámetro podría suponer que el modelo se ajuste tanto al entrenamiento que ante datos nuevos no se comporte de la manera óptima. Este fenómeno se conoce como sobreentrenamiento u *Overffitting*.

  Gamma se considera de manera aproximada la inversa del radio de influencia de los vectores de soporte. Un valor pequeño de este parámetro implicará mayor influencia de los vectores de soporte.

Mediante el uso de funciones propias de la libreria *Scikit-Learn*, se han hallado los hiperparámetros óptimos para el conjunto de datos de entrenamiento y validación (70 % del total) siendo elegidos: $C = 1000$ y $gamma = 0{,}01$.

Capítulo 4

Resultados

4.1. Introducción

En este capítulo se detalla el proceso experimental a través del cual se ha evaluado el desempeño del sistema desarrollado en este TFG. En primer lugar se indican tanto la composición de los datos utilizados para la detección de acciones como las métricas que permiten valorar de manera objetiva la eficacia y eficiencia del sistema. Posteriormente se muestran los resultados obtenidos al aplicar el algoritmo a secuencias de imágenes de profundidad.

En esta sección se detallan las características de la base de datos [35] utilizada para el entrenamiento y evaluación de los algoritmos desarrollados en este TFG, así como las métricas utilizadas.

4.1.1. Base de datos

La base de datos GOPTD3 dispone de un total de 28 secuencias de imágenes de profundidad etiquetadas, que incluyen a 7 personas diferentes, realizando cada una de las 4 acciones a detectar: andar, correr, caerse y permanecer parado. De estas secuencias se extraen un total de 2111 muestras que se reparten entre las 4 acciones tal y como se muestra en la figura 4.1

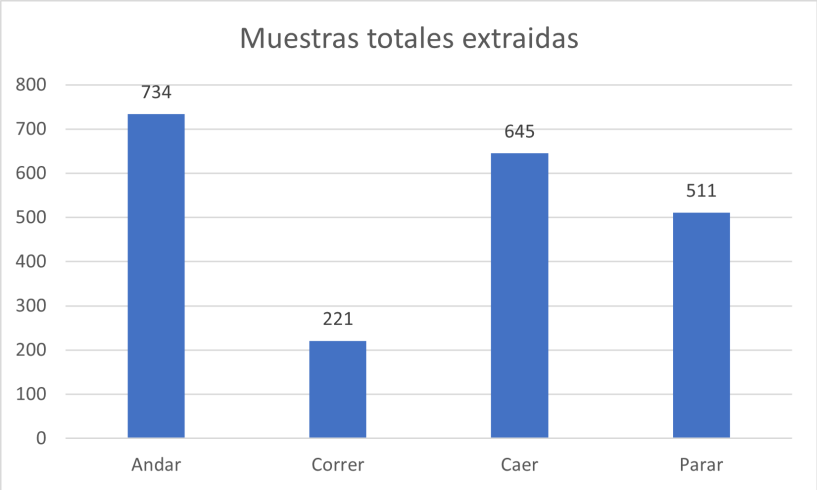


Figura 4.1: Muestras totales obtenidas de las secuencias de imágenes

Para realizar el entrenamiento y generación del modelo clasificador, se precisaba dividir los datos en datos de “Entrenamiento” y datos de “test”, tal y como se explica en la sección 3.4.2. Se decidió utilizar las secuencias de imágenes de profundidad pertenecientes a 5 de las 7 personas para la fase de entrenamiento y el resto para la validación del modelo generado mediante los datos de “test”. El número de muestras presente en cada nuevo conjunto de datos se muestra en la figura 4.2.

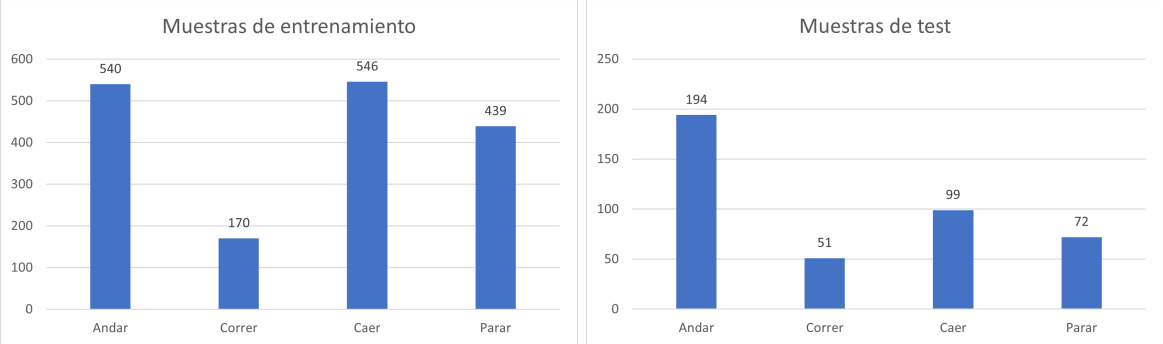


Figura 4.2: Muestras de entrenamiento (izquierda) y muestras de test (derecha)

4.1.2. Métricas para la evaluación del algoritmo clasificador

La eficacia y eficiencia del sistema desarrollado en este TFG deben evaluarse mediante parámetros objetivos que permitan identificar si el sistema es aplicable fuera de un entorno de investigación. Dado que se trata de un sistema de clasificación multiclase uno de los métodos más aplicados para evaluar el funcionamiento del algoritmo clasificador es la matriz de confusión. Dicha matriz muestra el porcentaje de veces que el modelo ha clasificado los datos de test dentro de las diferentes clases. Esto implica que el mejor de los casos sería obtener una matriz de confusión que tuviese un porcentaje del 100 % en su diagonal, ya que eso supondría la correcta clasificación del total de los datos. En la figura 4.3 se muestra un ejemplo de matriz de confusión:

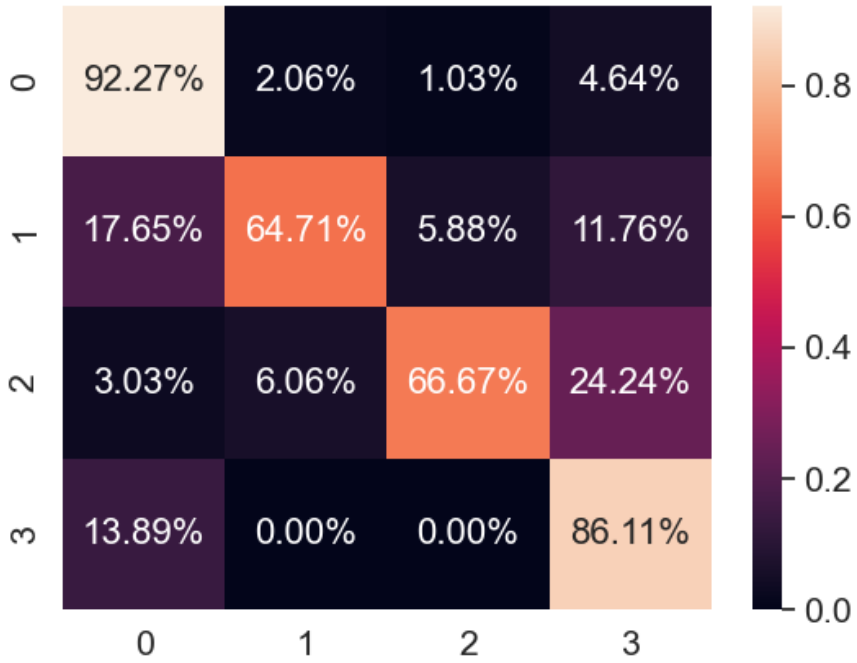


Figura 4.3: Ejemplo de matriz de confusión para un clasificador SVM con 4 clases.

en esta imagen las filas corresponden a las muestras reales, mientras que las columnas corresponden a las predicciones obtenidas del clasificador. Las acciones reflejadas en esta matriz de confusión son: andar (0), correr (1), caer (2) y parar (3).

De las matrices de confusión se pueden extraer 2 métricas comúnmente utilizadas en sistemas de clasificación binarios, "precisión" y "*recall*", para evaluar la eficacia del modelo. En estos sistemas de clasificación en que una de las clases es considerada la clase "positiva" y la otra la clase "negativa", la precisión indica el porcentaje de predicciones positivas realizadas que realmente son positivas (Verdaderos positivos), mientras que el *recall* indica el porcentaje de predicciones positivas que se han clasificado correctamente. De esta forma, un clasificador con alta precisión implica que acierta mucho al clasificar mientras que un clasificador con alto *recall* comete pocos fallos. En la figura 4.4 a continuación se muestra un ejemplo de clasificación binaria evaluada mediante las métricas comentadas:

		True/Actual	
		Positive ( )	Negative
Predicted	Positive ( )	5 (TP)	1 (FP)
	Negative	2 (FN)	2 (TN)

Figura 4.4: Ejemplo de clasificación binaria evaluada mediante las métricas precisión y *recall* [extraida de [14]]

En este ejemplo se requiere clasificar imágenes de perros (clase positiva) entre otras imágenes de animales (Clase negativa). El algoritmo clasificó 2 imágenes de perros como clase negativa y 1 foto de un animal que no era un perro como clase positiva. Una muestra de clase positiva clasificada como clase negativa se considera un "falso negativo" (FN en la imagen). Por otro lado, una muestra de clase negativa clasificada como clase positiva es un "falso positivo" (FP). Las muestras positivas correctamente clasificadas son "verdaderos positivos" (TP) y las muestras negativas correctamente clasificadas "verdaderos negativos" (TN). A partir de la matriz de confusión se puede calcular la precisión del clasificador según la ecuación 4.1 y el *recall* a partir de la ecuación 4.2.

$$precision = \frac{TP}{TP + FP} \tag{4.1}$$

$$recall = \frac{TP}{TP + FN} \tag{4.2}$$

En sistemas de clasificación multiclase como el dispuesto en este TFG, las métricas de precisión y *recall* se deben aplicar a cada clase, generando así una matriz de valores.

4.2. Resultados experimentales

En la figura 4.5 a continuación se muestra la matriz de confusión obtenida como resultado de la generación de un modelo de SVM para clasificación multiclase como se explica en la sección 3.4.3. En la matriz la correspondencia de números y acciones es: 0-Andar, 1-Correr, 2-Caer, 3-Andar.

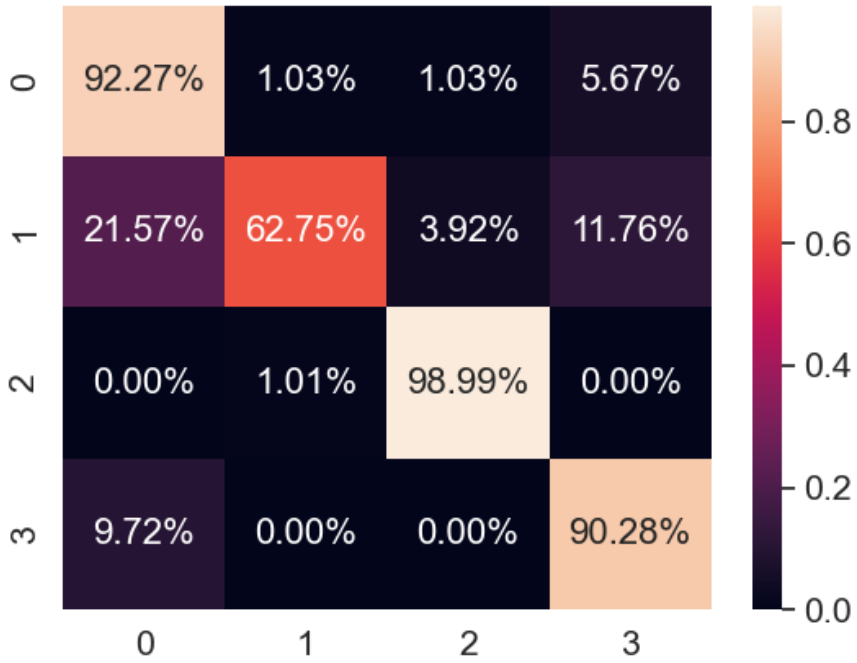


Figura 4.5: Matriz de confusión resultado de aplicar el modelo de clasificación sobre datos de test

cuyas métricas de precisión y *recall* se muestran en la tabla 4.1:

Acción	Precisión	Recall	Total de muestras
0	0,909	0,923	194
1	0,914	0,627	51
2	0,961	0,990	99
3	0,793	0,903	72

Tabla 4.1: Métricas de precisión y *recall* obtenidas.

En las figuras previas se observa como el sistema diseñado es capaz de clasificar las acciones “andar”, çaer” y “parar” de manera fiable (Porcentajes de acierto superiores al 90 %). Por otra parte, la acción “correr” tiene un porcentaje menor de aciertos que el resto de actividades y es confundida con las acciones de “andar”, en mayor medida, y “parar” en menor medida. Esto se debe principalmente a que el *dataset* no es balanceado como se observa en la figura 4.2. El número de muestras extraídas de las secuencias de imágenes de profundidad para la acción correr es aproximadamente un 50 % menor que para el resto de actividades. Esto implica que el algoritmo de clasificación tiende a clasificar las muestras dudosas como el resto de acciones en lugar de clasificarla dentro de la acción “correr”, ya que de esa forma, acertará un mayor número de veces.

4.3. Tiempos de ejecución

El tiempo de ejecución de sistemas como el desarrollado en este TFG es un parámetro determinante, ya que están pensados para funcionar en tiempo real. En este apartado se muestran los tiempos de ejecución

de los algoritmos desarrollados en este TFG. Los tiempos de ejecución han sido medidos ejecutando el algoritmo de clasificación en un equipo HP LAPTOP 15S-FQ1XXX con un procesador Intel(R) Core(TM) i7-1065G7 CPU @ 1.30GHz 1.50 GHz [62] y 8GB. En la figura 4.6 a continuación se muestran los tiempos de ejecución promediados de los algoritmos diseñados: Algoritmo de seguimiento robusto de personas (mínimo y máximo de personas según las secuencias de imágenes grabadas), fase de entrenamiento o generación del modelo clasificador y, por último, fase de clasificación de muestras.

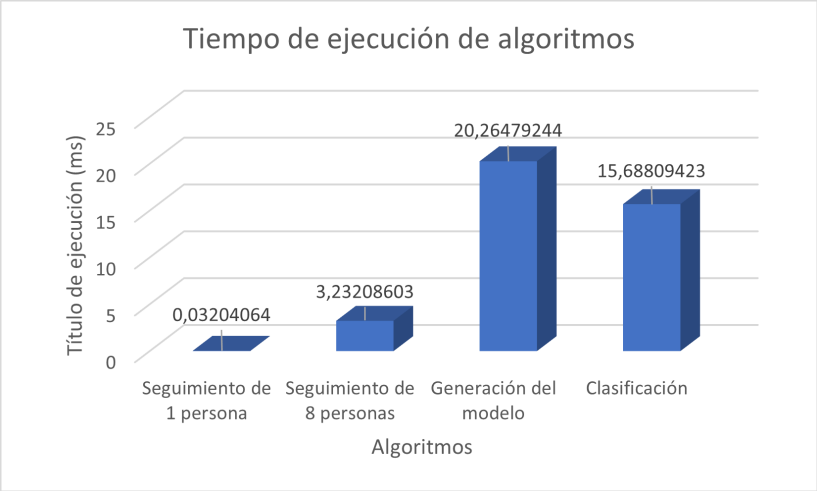


Figura 4.6: Tiempo de ejecución de algoritmos desarrollados

Según los tiempos obtenidos y teniendo en cuenta que la cámara de profundidad usada toma 30 imágenes por segundo, es decir, cada 0,333 segundos, el conjunto de los algoritmos pueden aplicarse en tiempo real dado que excluyendo el tiempo de generación del modelo de clasificación que se computa una única vez, la suma de tiempos (19 ms en total) es menor a la frecuencia de adquisición de imágenes.

Capítulo 5

Conclusiones y líneas futuras

En esta sección se exponen las conclusiones extraídas del sistema propuesto en vista de los resultados obtenidos. Posteriormente se incluyen formas de continuar con la linea de investigación propuesta en este trabajo.

5.1. Conclusiones

El sistema desarrollado en este TFG permite realizar el seguimiento de múltiples personas en escena, detectadas por el algoritmo previo desarrollado en [9], utilizando únicamente imágenes de profundidad para posteriormente detectar acciones concretas llevadas a cabo por dichas personas. Además, este sistema preserva la identidad de las personas detectadas, por lo que es posible implementarlo incluso en entornos en que la ley de protección de datos sea un impedimento.

El principal objetivo del trabajo desarrollado era discernir caídas de las personas detectadas, entre otras acciones. En las pruebas llevadas a cabo se han obtenido resultados satisfactorios respecto a este objetivo para 3 de las 4 actividades detectadas, obteniendo porcentajes de acierto en la clasificación superiores al 90 %, siendo el porcentaje de acierto de la acción "Caer" cercano al 100 %. Por otra parte se ha conseguido realizar el seguimiento de múltiples personas proximas entre sí en escena como se muestra en la figura 3.8.

El algoritmo clasificador utilizado para la detección de actividades es la SVM dado que este algoritmo funciona mejor con un número reducido de muestras y cuando se requieren tiempos de computación pequeños. Para este caso concreto, el tiempo de computación de ambos algoritmos es menor que el tiempo de adquisición de imágenes del sensor utilizado por lo que se deduce que el sistema es capaz de operar en tiempo real.

En conclusión, el sistema planteado satisface los objetivos del TFG, aunque existen aspectos a mejorar en los trabajos futuros.

5.2. Líneas futuras

A lo largo del desarrollo de este trabajo se plantean acciones que podrían mejorar el funcionamiento del sistema planteado o bien ampliar el alcance del mismo. A continuación se exponen algunas de las posibilidades:

- Emplear otros clasificadores como "*Random forest*" o redes neuronales. Para ello se precisaría un elevado número de muestras a cambio de obtener mejores resultados.
- Mejor detección de la actividad "Correr" ya que es la acción con menor porcentaje de aciertos o detección de un mayor número de acciones: Saltar, agacharse, gatear, etc.
- Seguimiento de personas mediante múltiples cámaras. Esto permitiría ampliar el área de detección y seguimiento de personas para espacios con mayor área (Pasillos, comedores, salones de actos, etc).
- Extracción de otras características de la imagen con el fin de usar otros vectores de características que permitan aumentar el porcentaje de aciertos del clasificador.

Capítulo 6

Pliego de condiciones

Un correcto funcionamiento del sistema planteado pasa por el cumplimiento de unos requisitos de hardware y software mínimos en los equipos en que se van a implementar:

6.1. Requisitos de Hardware

- 2 Gb de Ram DDR2 a 800Mhz o superior
- Procesador de 64 bits Duo Core 2,16 Ghz o superior
- Al menos 15 Gb de memoria libre en el disco duro o superior para la instalación de las librerías requeridas.

6.2. Requisitos de Software

- Sistema operativo Ubuntu 18.04.5 LTS 64-bit
- Librerias Libfreenect 2
- Librerias PCL 1.6
- Librerias OpenCv 2.4.9
- Librerias Scikit-Learn 1.0.2
- CMake

Capítulo 7

Presupuesto

7.1. Costes de equipamiento

7.1.1. Equipamiento Hardware

Concepto	Cantidad	Coste unitario (€)	Subtotal (€)
Ordenador compatible	1	800	800
Sensor TOF Kinect 2	1	176,71	176,71
Coste total			976.71

7.1.2. Recursos Software

Concepto	Cantidad	Coste unitario (€)	Subtotal (€)
Ubuntu 18.04.5 LTS 64-bit	1	0	0
Librerias Opencv 2.4.9	1	0	0
Librerias Libfreenect 2	1	0	0
Librerias PCL 1.6	1	0	0
Texstudio 2	1	0	0
Cmake	1	0	0
Librerias Scikit-Learn 1.0.2	1	0	0
Coste total			0

7.2. Costes de mano de obra

Concepto	Cantidad (horas)	Coste (€/hora)	Subtotal (€)
Montaje y fijacion cenital de la cámara	10	15	150
Desarrollo de aplicación Software	320	60	19200
Documentación y elaboración de manual de uso	70	15	1050
Coste total			20400

7.3. Costes totales

Concepto	Subtotal (€)
Hardware	976.71
Software	0
Mano de obra	20400
Coste total	21376.71

El importe total del presupuesto asciende a la cantidad de: VEINTIUN MIL TRESCIENTOS SETENTA Y SEIS EUROS CON SETENTA Y UN CENTIMOS En Alcalá de Henares, a de abril de 2022.

Alán Resa Peña Graduado en Ingeniería en Electrónica y Automática Industrial

Bibliografía

[1] "A brief introduction to time-of-flight sensing. part 2 indirect tof sensors," https://www.terabee.com/a-brief-introduction-to-time-of-flight-sensing-part-2-indirect-tof-sensors/.

[2] "Understanding indirect tof depth sensing," https://devblogs.microsoft.com/azure-depth-platform/understanding-indirect-tof-depth-sensing.

[3] M. Feigin, R. Whyte, A. Bhandari, A. Dorington, and R. Raskar, "Modeling wiggling as a multi-path interference problem in amcw tof imaging," *Opt. Express*, vol. 23, no. 15, pp. 19 213–19 225, Jul 2015. [Online]. Available: http://opg.optica.org/oe/abstract.cfm?URI=oe-23-15-19213

[4] T. Hoegg, D. Lefloch, and A. Kolb, "Real-time motion artifact compensation for pmd-tof images," in *Time-of-flight and depth imaging. Sensors, algorithms, and applications.* Springer, 2013, pp. 273–288.

[5] M. Hansard, S. Lee, O. Choi, and R. P. Horaud, *Time-of-flight cameras: principles, methods and applications.* Springer Science & Business Media, 2012.

[6] "Harmonic aliasing (wiggle error) in indirect time-of-flight depth cameras," https://medium.com/chronoptics-time-of-flight/harmonic-aliasing-wiggle-error-in-indirect-time-of-flight-depth-cameras-efa1632d4d1b.

[7] S. Giancola, D. Piron, P. Poppa, and R. Sala, "A solution for crime scene reconstruction using time-of-flight cameras," 08 2017.

[8] M. Naeemabadi, B. Dinesen, O. Andersen, and J. Hansen, "Investigating the impact of a motion capture system on microsoft kinect v2 recordings: A caution for using the technologies together," *PLOS ONE*, vol. 13, 09 2018.

[9] D. Fuentes Jiménez, "Diseño, implementación y evaluación de un sistema de conteo de personas basado en cámaras de tiempo de vuelo [trabajo fin de grado]," Bachelor's Thesis, Escuela Politécnica Superior, Universidad de Alcalá, Madrid, 2016.

[10] G. Welch, G. Bishop *et al.*, "An introduction to the kalman filter," 1995.

[11] U. FİDAN, E. UZUNHİSARCIKLI, and İ. ÇALIKUŞU, "Classification of dermatological data with self organizing maps and support vector machine," *Afyon Kocatepe Üniversitesi Fen Ve Mühendislik Bilimleri Dergisi*, vol. 19, no. 3, pp. 894–901, 2019.

[12] D. Lai, R. Begg, and M. Palaniswami, "Svm models for diagnosing balance problems using statistical features of the mtc signal." *International Journal of Computational Intelligence and Applications*, vol. 7, pp. 317–331, 09 2008.

[13] M. Baptista Ríos *et al.*, "Detección y caracterización de personas en espacios inteligentes con cámaras de color," 2015.

[14] "Multi-class metrics made simple, part i: Precision and recall," https://towardsdatascience.com/multi-class-metrics-made-simple-part-i-precision-and-recall-9250280bddc2/.

[15] M. Pérez and J. Luis, "Comunicación con computador mediante señales cerebrales. aplicación a la tecnología de la rehabilitación," Ph.D. dissertation, Ph. D. thesis, Universidad Politécnica de Madrid, 2009.

[16] J. Smisek, M. Jancosek, and T. Pajdla, "3d with kinect," in *Consumer depth cameras for computer vision*. Springer, 2013, pp. 3–25.

[17] C. A. Luna, C. Losada-Gutierrez, D. Fuentes-Jimenez, A. Fernandez-Rincon, M. Mazo, and J. Macias-Guarasa, "Robust people detection using depth information from an overhead time-of-flight camera," *Expert Systems with Applications*, vol. 71, pp. 240–256, 2017.

[18] C. A. Luna, C. Losada-Gutiérrez, D. Fuentes-Jiménez, and M. Mazo, "Fast heuristic method to detect people in frontal depth images," *Expert Systems with Applications*, vol. 168, p. 114483, 2021.

[19] G. Wang, C. Li, Y. Ma, A. Zheng, J. Tang, and B. Luo, "Rgb-t saliency detection benchmark: Dataset, baselines, analysis and a novel approach," in *Chinese Conference on Image and Graphics Technologies*. Springer, 2018, pp. 359–369.

[20] X. Qi, R. Liao, J. Jia, S. Fidler, and R. Urtasun, "3d graph neural networks for rgbd semantic segmentation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5199–5208.

[21] D. Liciotti, M. Paolanti, E. Frontoni, and P. Zingaretti, "People detection and tracking from an rgb-d camera in top-view configuration: review of challenges and applications," in *International Conference on Image Analysis and Processing*. Springer, 2017, pp. 207–218.

[22] P. Pareek and A. Thakkar, "A survey on video-based human action recognition: recent updates, datasets, challenges, and applications," *Artificial Intelligence Review*, vol. 54, no. 3, pp. 2259–2322, 2021.

[23] S. Saponara, A. Elhanashi, and A. Gagliardi, "Implementing a real-time, ai-based, people detection and social distancing measuring system for covid-19," *Journal of Real-Time Image Processing*, vol. 18, no. 6, pp. 1937–1947, 2021.

[24] A. C. Cob-Parro, C. Losada-Gutiérrez, M. Marrón-Romera, A. Gardel-Vicente, and I. Bravo-Muñoz, "Smart video surveillance system based on edge computing," *Sensors*, vol. 21, no. 9, p. 2958, 2021.

[25] A. Akula, A. K. Shah, and R. Ghosh, "Deep learning approach for human action recognition in infrared images," *Cognitive Systems Research*, vol. 50, pp. 146–154, 2018.

[26] C. Martinez, M. Baptista, C. Losada, M. Marrón, and V. Boggian, "Human action recognition in realistic scenes based on action bank," in *International Work-conference on Bioinformatics and Biomedical Engineering*, 2016, pp. 314–325.

[27] F. Dumitrescu, C.-A. Boiangiu, and M.-L. Voncilă, "Fast and robust people detection in rgb images," *Applied Sciences*, vol. 12, no. 3, p. 1225, 2022.

[28] R. Al-Akam and D. Paulus, "Rgbd human action recognition using multi-features combination and k-nearest neighbors classification," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 8, no. 10, pp. 383–389, 2017.

[29] Z. Gao, S. Li, Y. Zhu, C. Wang, and H. Zhang, "Collaborative sparse representation leaning model for rgbd action recognition," *Journal of Visual Communication and Image Representation*, vol. 48, pp. 442–452, 2017.

[30] Y. Xiao, V. R. Kamat, and C. C. Menassa, "Human tracking from single rgb-d camera using online learning," *Image and Vision Computing*, vol. 88, pp. 67–75, 2019.

[31] L. Jia and R. J. Radke, "Using time-of-flight measurements for privacy-preserving tracking in a smart room," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, pp. 689–696, 2013.

[32] R. Lange and P. Seitz, "Solid-state time-of-flight range camera," *IEEE Journal of quantum electronics*, vol. 37, no. 3, pp. 390–397, 2001.

[33] J. Sell and P. O'Connor, "The xbox one system on a chip and kinect sensor," *IEEE Micro*, vol. 34, no. 2, pp. 44–53, 2014.

[34] A. Fernández Rincón, "Contribución a la detección y conteo de personas a partir de información de profundidad," Bachelor's Thesis, University of Alcalá, 2016.

[35] "Geintra overhead tof people detection 3 (gotpd3) database: Human activity detection," 2019, https://www.geintra-uah.org/datasets/gotpd3.

[36] A. Payne, A. Daniel, A. Mehta, B. Thompson, C. S. Bamji, D. Snow, H. Oshima, L. Prather, M. Fenton, L. Kordus *et al.*, "7.6 a 512× 424 cmos 3d time-of-flight image sensor with multi-frequency photo-demodulation up to 130mhz and 2gs/s adc," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*. IEEE, 2014, pp. 134–135.

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

[38] J. Biswas and M. Veloso, "Depth camera based indoor mobile robot localization and navigation," in *2012 IEEE International Conference on Robotics and Automation*, 2012, pp. 1697–1702.

[39] J. Seo, S. Han, S. Lee, and H. Kim, "Computer vision techniques for construction safety and health monitoring," *Advanced Engineering Informatics*, vol. 29, no. 2, pp. 239–251, 2015.

[40] Z. Zhang, "Microsoft kinect sensor and its effect," *IEEE MultiMedia*, vol. 19, no. 2, pp. 4–10, 2012.

[41] S. Giancola, M. Valenti, and R. Sala, "Metrological qualification of the intel d400 active stereoscopy cameras," in *A Survey on 3D Cameras: Metrological Comparison of Time-of-Flight, Structured-Light and Active Stereoscopy Technologies*. Springer, 2018, pp. 71–85.

[42] M. S. Ahn, H. Chae, D. Noh, H. Nam, and D. Hong, "Analysis and noise modeling of the intel realsense d435 for mobile robots," in *2019 16th International Conference on Ubiquitous Robots (UR)*. IEEE, 2019, pp. 707–711.

[43] G. Roberts and M. Ali-Bakhshian, "A brief introduction to time-to-digital and digital-to-time converters," *Circuits and Systems II: Express Briefs, IEEE Transactions on*, vol. 57, pp. 153 – 157, 04 2010.

[44] P. Gil, T. Kisler, G. J. Garcia, C. A. Jara, and J. Corrales, "Calibración de cámaras de tiempo de vuelo: Ajuste adaptativo del tiempo de integración y análisis de la frecuencia de modulación," *Revista Iberoamericana de Automática e Informática industrial*, vol. 10, no. 4, pp. 453–464, 2013.

[45] D. Jiménez, D. Pizarro, M. Mazo, and S. Palazuelos, "Modeling and correction of multipath interference in time of flight cameras," *Image and Vision Computing*, vol. 32, no. 1, pp. 1–13, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0262885613001650

[46] D. Jimenez, D. Pizarro, and M. Mazo, "Single frame correction of motion artifacts in pmd-based time of flight cameras," *Image and Vision Computing*, vol. 32, no. 12, pp. 1127–1143, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0262885614001450

[47] J.-W. Kim, K.-S. Park, B.-D. Park, and S.-J. Ko, "Real-time vision-based people counting system for the security door," in *Proceedings of the IEEK Conference*. The Institute of Electronics and Information Engineers, 2002, pp. 1416–1419.

[48] B.-K. Dan, Y.-S. Kim, Suryanto, J.-Y. Jung, and S.-J. Ko, "Robust people counting system based on sensor fusion," *IEEE Transactions on Consumer Electronics*, vol. 58, no. 3, pp. 1013–1021, 2012.

[49] K. Bushby, T. Cole, J. Matthews, and J. Goodship, "Centiles for adult head circumference." *Archives of disease in childhood*, vol. 67, no. 10, pp. 1286–1287, 1992.

[50] S. Matzner, A. Heredia-Langner, B. Amidan, E. J. Boettcher, D. Lochtefeld, and T. Webb, "Standoff human identification using body shape," in *2015 IEEE international symposium on technologies for homeland security (HST)*. IEEE, 2015, pp. 1–6.

[51] M. S. Grewal and A. P. Andrews, "Applications of kalman filtering in aerospace 1960 to the present [historical perspectives]," *IEEE Control Systems Magazine*, vol. 30, no. 3, pp. 69–78, 2010.

[52] M. Athans, "The importance of kalman filtering methods for economic systems," pp. 49–64, 1974.

[53] "Ejemplo de red neuronal para la clasificación de fotos de perros y gatos," https://machinelearningmastery.com/how-to-develop-a-convolutional-neural-network-to-classify-photos-of-dogs-and-cats/, [Último acceso 15/02/2022].

[54] "Ejemplo de svm para clasificación de frutos secos," https://github.com/alvarobasi/nutClassifier, [Último acceso 15/02/2022].

[55] L. N. Kanal, "Perceptron," in *Encyclopedia of Computer Science*, 2003, pp. 1383–1385.

[56] G. Hamerly and C. Elkan, "Learning the k in k-means," *Advances in neural information processing systems*, vol. 16, pp. 281–288, 2004.

[57] C. Cortes and V. Vapnik, "Support vector machine," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[58] H. Q. Minh, P. Niyogi, and Y. Yao, "Mercers theorem, feature maps, and smoothing," pp. 154–168, 2006.

[59] A. F. Smeaton and C. J. van Rijsbergen, "The nearest neighbour problem in information retrieval: an algorithm using upperbounds," in *Proceedings of the 4th annual international ACM SIGIR conference on Information storage and retrieval: theoretical issues in information retrieval*, 1981, pp. 83–87.

[60] G. Bradski and A. Kaehler, "Opencv," *Dr. Dobbs journal of software tools*, vol. 3, p. 2, 2000.

[61] O. Kramer, "Scikit-learn," in *Machine learning for evolution strategies*. Springer, 2016, pp. 45–53.

[62] "Procesador intel core i7-1065g7," https://www.intel.es/content/www/es/es/products/sku/196597/intel-core-i71065g7-processor-8m-cache-up-to-3-90-ghz/specifications.html/.

Universidad de Alcalá
Escuela Politécnica Superior

ESCUELA POLITECNICA
SUPERIOR

Universidad
de Alcalá