Universidad de Alcalá
Escuela Politécnica Superior

Grado en Ingeniería en Tecnologías de la Telecomunicación

Trabajo Fin de Grado

Re-identificación de personas utilizando únicamente información de profundidad

Autor: Sara Luengo Sánchez

Tutor: Cristina Losada Gutiérrez

2018

UNIVERSIDAD DE ALCALÁ

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería en Tecnologías de la Telecomunicación

Trabajo Fin de Grado

Re-identificación de personas utilizando únicamente información de profundidad

Autora: Sara Luengo Sánchez

Tutor: Cristina Losada Gutiérrez

Tribunal:

Presidente: Juan Manuel Miguel Jiménez

Vocal 1º: Alejandro Martínez Arribas

Vocal 2º: Cristina Losada Gutiérrez

Calificación: .....

Fecha: .....

A todos aquellos que creyeron en mí
y me apoyaron a lo largo del camino...

"Este mundo es a veces una tempestad.
Pero recuerda, el sol siempre vuelve a salir."

Brandon Sanderson, *El Camino de los Reyes*

Agradecimientos

En primer lugar, me gustaría agradecer a mis tutores Cristina Losada y Carlos Luna por haberme ofrecido la oportunidad de trabajar en este proyecto así como haberme guiado y ayudado durante su realización. También me gustaría darle las gracias a Manuel Mazo, que me descubrió hace dos años y me dio la oportunidad de trabajar en el grupo de investigación GEINTRA donde he conocido a gente maravillosa y aprendido un sinnúmero de cosas.

Y no podría pasar esta ocasión sin mencionar a todos los profesores que me han impartido clase a lo largo de mi vida estudiantil, en especial a los de matemáticas y electrónica, los cuales me motivaron a estudiar este grado y escoger la especialización, y gracias a los cuales hoy me encuentro aquí escribiendo este documento.

Y por supuesto, me gustaría darles las gracias a mis padres Antonio y Encarna y a mi hermano Sergio por haber estado ahí siempre que lo he necesitado y haberme ayudado durante toda mi vida a levantarme si caía y a conseguir aquello que me propusiera. También me gustaría darle las gracias a Carlos por haberme apoyado durante estos años tan difíciles y haberme escuchado y animado cuando tenía problemas. Asimismo, me gustaría agradecerle a mi tía Esperanza toda la ayuda que me ha prestado ya que sin ella estudiar el grado habría sido muy diferente y más complicado.

Por último, quiero dar las gracias a mis compañeros de laboratorio en GEINTRA David Casillas, Fran, Juanma, Leticia y en especial a David Fuentes por toda la ayuda recibida mientras realizaba este proyecto. He aprendido muchísimo de ellos y es un placer trabajar con gente así en un entorno tan inspirador.

Resumen

La finalidad de este trabajo es el diseño, implementación y evaluación de un sistema de re-identificación de personas a partir de imágenes de profundidad obtenidas por un sensor de tiempo de vuelo (ToF) ubicado en posición cenital. Este trabajo parte del detector desarrollado por el grupo GEINTRA y añade nuevas funcionalidades al sistema: ejecución en tiempo real y re-identificación de personas. La detección y la re-identificación se realizan con un clasificador basado en la técnica de Análisis de Componentes Principales (PCA). Para la validación del sistema se ha utilizado una base de datos de imágenes de profundidad cumpliendo con éxito los objetivos propuestos.

Palabras clave: Re-identificación, sensor ToF, imágenes de profundidad, Análisis de Componentes Principales (PCA), tiempo real.

Abstract

The main objective of this work is the design, implementation and evaluation of a system capable of re-identifying people using only depth images obtained by a Time of Flight (ToF) sensor placed in zenithal position. This work starts uses the detector develoved by the GEINTRA group and adds new funtionalities to the system: real-time ejecution and people re-identification. The detection and re-identification of people are done by a clasificator based on the Principal Component Analysis (PCA) technique. The validation of the systeam has been done using a data base of depth images, achieving the proposed goals.

Keywords: Re-identification, ToF sensor, depth images, Principal Components Analysis (PCA), real time.

Resumen extendido

La re-identificación de personas es la capacidad de reconocer a un individuo que ha sido previamente observado por la red de sensores. En los últimos años, este campo ha adquirido mayor importancia debido al gran número de aplicaciones que tiene en sectores como la vídeo-vigilancia, el control de aforos y el seguimiento de personas en entornos determinados. En este ámbito, se propone un sistema encargado de la re-identificación de personas mediante de imágenes de profundidad obtenidas a partir de una red de sensores de tiempo de vuelo (ToF) situados en posición cenital. A partir de los datos obtenidos de los sensores, se obtienen las características necesarias para re-identificar a la persona, preservando su privacidad al no ser posible reconocer la identidad de los usuarios únicamente mediante imágenes de profundidad.

El sistema propuesto se compone de varias etapas claramente diferenciadas entre sí: en primer lugar se realiza un preprocesamiento de la imagen para eliminar el ruido y los píxeles erróneos, en segundo lugar se detecta la posición de las personas en la escena, a continuación se extraen las características y atributos de los individuos detectados y se procede a la re-identificación de los mismos. Las características extraídas de los individuos permiten distinguir a un individuo de los demás independientemente de donde se encuentre este dentro de la escena. La re-identificación se realiza mediante un clasificador basado en la técnica de Análisis de las Componentes Principales (PCA) que permite reducir la dimensionalidad de los datos creando un espacio diferente para cada persona. Las características de estos espacios se almacenan y son utilizadas por el sistema para distinguir a una persona de otra.

Para la obtención de resultados y estadísticas del funcionamiento del sistema se han realizado diversas pruebas experimentales con la base de datos GOTPD1 proporcionada por el grupo investigador GEINTRA. Esta base de datos incluye varias secuencias de diversas personas y se ha obtenido mediante dos sensores Kinect II ubicados en posición cenital en la misma sala con campos de visión no solapados. El sistema también es capaz de realizar la detección y re-identificación de personas en tiempo real si se encuentra conectado a uno de estos sensores.

Extended Abstract

People re-identification consist on recognizing an individual who has been observed by the sensors network previously. This field has become more and more relevant over the last few years due to the huge number of applications it has on multiple sectors such as video surveillance, access control and people tracking in controlled environments. In this context, it is proposed a system able to re-identify people from depth images obtained by Time of Flight (ToF) sensor network. The data acquired by the sensors allows for people re-identification keeping their privacy untouched, since it is not possible to recognize their identities from depth information only.

The proposed system starts by preprocessing the images in order to filter the noise and fix wrong pixels. Next, the people detection is done followed by the extraction of physical characteristics and attributes of the people detected. This data contains enough information to distinguish an individual from others in a classification process, regardless of their position in the scene. The Principal Components Analysis (PCA) technique is used to the classification of the data. This technique reduces the dimensionality of the input data and creates a new data space for each new person. These spaces are stored in the system and are used during the re-identification phase in order to distinguish one person from another.

Various test has been done to obtain results and statistics of the system's behavior. The GOPTD1 database from GEINTRA group has been used with that purpose since it includes several sequences of multiple people acquired using two Kinect II sensors situated in zenithal position in the same room and whose fields of view did not overlap.

Índice general

Índice de figuras

Índice de tablas

Índice de listados de código fuente

Lista de acrónimos

CMOS	Complementary Metal-Oxide-Semiconductor.
CWM	Continuous Wave Modulation.
GEINTRA	Grupo de Ingeniería Electrónica Aplicada a Espacios Inteligentes y Transporte.
LED	Light-Emitting Diode.
PCA	Principal Component Analysis.
ROI	Region Of Interest.
SoC	System on a Chip.
SR	SubRegion.
TFG	Trabajo Fin de Grado.
ToF	Time of Flight.

Capítulo 1

Introducción

1.1. Introducción

La re-identificación de personas consiste en detectar cuando una persona que está siendo observada por una red de sensores ha sido ya previamente observada por alguno de los sensores de esa red y ser capaz de distinguir al individuo del resto de individuos que hayan pasado por el sistema. Dado el gran número de aplicaciones que tiene en el sector de la seguridad, como son el control de aforos, el análisis de flujos y comportamiento de las personas, y la vídeo-vigilancia, la re-identificación se ha constituido como un campo de gran interés para la comunidad científica ([7], [8], [9]). Además, debido a las dificultades que se presentan por los cambios en la iluminación, la posición, la distancia al sensor, etc. se trata de un tema de gran relevancia.

Por lo tanto, el objetivo de este TFG es el desarrollo e implementación de un sistema capaz de re-identificar personas de forma robusta partir de imágenes de profundidad obtenidas por una red de sensores de tiempo de vuelo ubicados en posición cenital.

Dada la gran controversia en torno a la relación existente entre la seguridad y la privacidad, el equilibrio que debe mantenerse entre ambas y donde deben situarse los límites de una en favor de la otra, en este trabajo se ha decidido abordar la re-identificación de personas empleando únicamente información de profundidad. La utilización de este tipo de información exclusivamente se debe a que no aporta información que permita reconocer la identidad del individuo, asegurando de esta forma la privacidad de las personas grabadas. La información es obtenida a través de dos sensores de tiempo de vuelo (ToF) ([10]) situados en posición cenital con el objetivo de reducir el efecto de las oclusiones que suponen uno de los mayores problemas existentes en los sistemas de seguridad actuales ([11],[12]).

Debido a la relevancia de este tema, existen numerosos documentos y trabajos que abordan la re-identificación de personas desde diferentes enfoques utilizando redes de sensores. En [13] se utiliza un descriptor local llamado quaternionic local binary pattern (QLBP) que utiliza información de color, mientras que en [14] se añade información de textura además de la de color en un espacio transformado para la medida de similitud utilizando el análisis de la correlación canónica regularizada (regularized canonical correlation analysis, RCCA). Otros autores [15] han decido incluir información sobre la pose de los individuos para mejorar la re-identificación. Los trabajos más vanguardistas [16] emplean redes neuronales convolucionales (CNNs) para la extracción de características en un espacio de baja dimensionalidad. Todos los trabajos mencionados utilizan imágenes de color, sin embargo en los últimos años han aparecido los sensores RGB-D que proporcionan información de profundidad junto con la imagen de color [17] lo cual ha supuesto que un gran número de trabajos utilicen ambos tipos de información durante la

re-identificación [18], [19], [20], [21], [22]. Para ello han utilizado diferentes estrategias, desde patrones locales [21] hasta la apariencia de la ropa [22]. Sin embargo, todos estos trabajos al utilizar información de color aportan información que permite averiguar la identidad del individuo vulnerando su privacidad. La utilización de imagen de color se debe a que los problemas de precisión que presentan los datos de profundidad como son las interferencias por multicamino [23] y [24] o los artefactos de movimiento [25] y [26].

A continuación, se describe el sistema propuesto en este trabajo de fin de grado (TFG) para la detección y re-identificación de personas a partir de únicamente información de profundidad. A lo largo del resto del documento, en el capítulo 2 se presentan las bases teóricas en las que se fundamenta el trabajo desde un punto de vista *hardware* y *software*, para después proceder a la explicación detallada del funcionamiento del sistema en el capítulo 3 y, finalmente, presentar los principales resultados (capítulo 4) y conclusiones obtenidas así como abordar los posibles trabajos futuros (capítulo 5).

1.2. Sistema propuesto

El sistema que se propone debe ser capaz detectar personas, extraer un conjunto de características que definan a la persona unívocamente, crear una clase o modelo a partir de esa persona y re-identificar a dicha persona si aparece de nuevo en la escena. Para ello se ha diseñado el sistema expuesto en la figura 1.1 que se compone de diferentes bloques cuya función se expone a continuación. Cada uno de estos bloques se explica en mayor detalle en el capítulo 3.

Para ello se ha diseñado el sistema expuesto en la figura 1.1 que está compuesto por dos partes:

- **Etapa *Offline*:** se encarga del entrenamiento de las clases. Genera las clases y las almacena en archivos para ser usadas posteriormente durante la re-identificación.
- **Etapa *Online*:** lleva a cabo la re-identificación de personas a partir de imágenes de profundidad obtenidas de una secuencia previamente grabada o de un sensor en tiempo real. Para ello, es necesario que previamente se haya realizado el entrenamiento de las clases.

Estas dos partes se dividen a su vez en distintos bloques cuya función se expone a continuación. Cada uno de estos bloques se explica en mayor detalle en el capítulo 3.

1. **Obtención de las imágenes de profundidad**: el sistema dispone de tres posibles entradas de datos. En primer lugar, las imágenes de entrenamiento son secuencias de vídeo en las que únicamente aparece una persona y son utilizadas para crear la clase que identifica a cada persona y que se usará posteriormente en la re-identificación. En segundo lugar, las imágenes de test también son secuencias grabadas previamente con las que el sistema intenta re-identificar a las personas que aparecen en ellas y generar métricas y tasas de acierto del sistema para así poder evaluar su eficacia. Por último, las imágenes del sensor son obtenidas en tiempo real de la cámara ToF para la re-identificación de las personas en la escena.
2. **Detección de personas:** una vez obtenida la imagen de profundidad de una de las fuentes, se realiza un pre-procesado sobre la misma con el objetivo de eliminar los posibles píxeles erróneos y el ruido generados por el sensor. A continuación, la imagen es introducida en el algoritmo de detección de personas que busca los máximos de altura en la imagen y determina si corresponden, o no, a una persona.

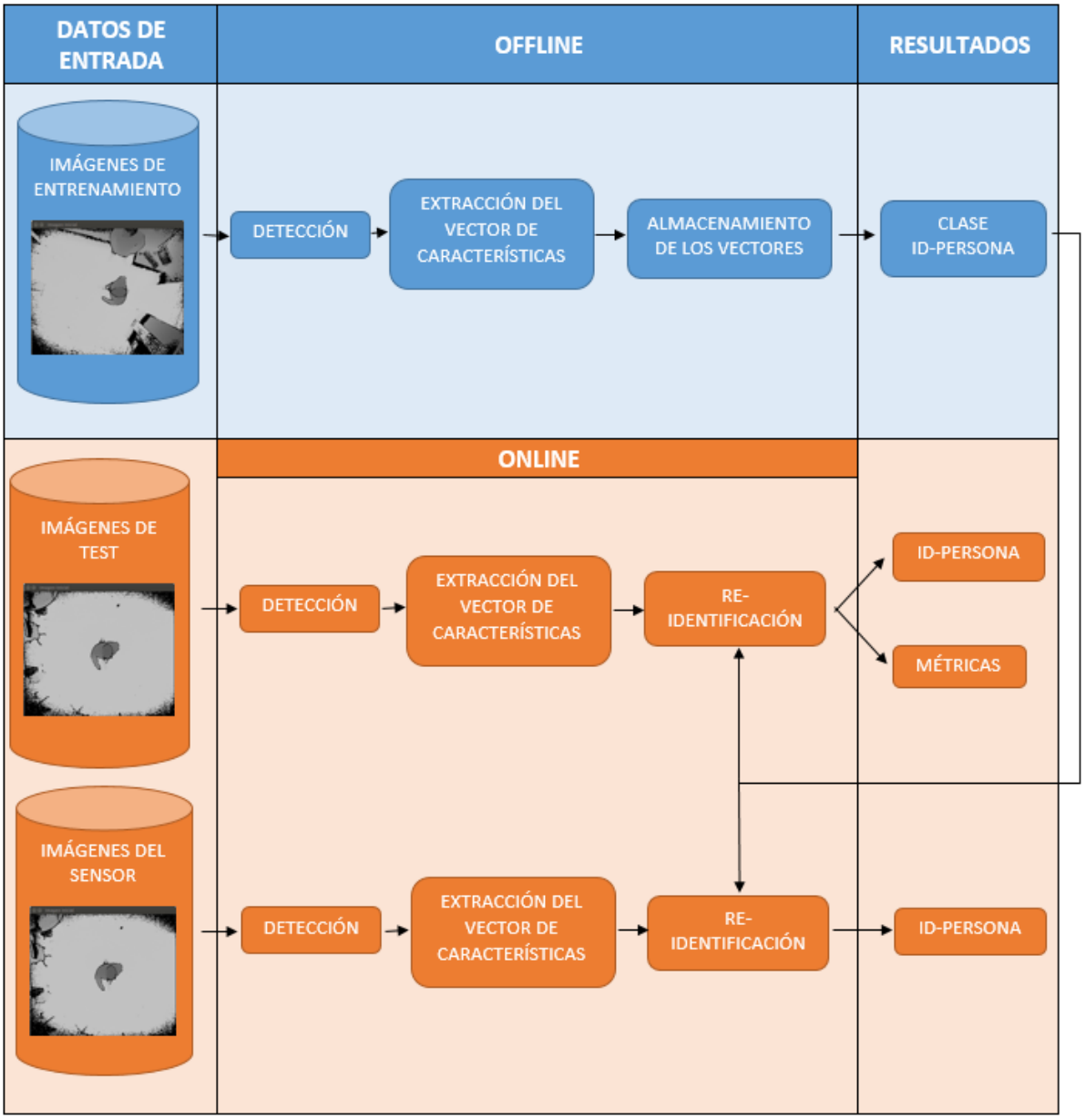


Figura 1.1: Diagrama de bloques general del sistema completo.

3. **Extracción del vector de características:** cuando el detector identifica un máximo como persona, se extraen un conjunto de características de los atributos físicos de la región de interés (ROI) definida alrededor del máximo, que son almacenados en un vector. Este vector se utiliza tanto durante la fase de re-identificación como para la creación de la clase que identifica a la persona.

4. **Creación de la Clase ID:** si el sistema está trabajando con las imágenes de entrenamiento, los vectores de características son guardados en un archivo. Al finalizar la secuencia, se recuperan todos los vectores y se crea una clase a partir de ellos. Esta clase permite comprobar posteriormente si la persona que se observa en una imagen es la misma persona a partir de la cual se creó la clase. Tanto para la creación de las clases, como para la posterior clasificación se emplea el Análisis de Componentes Principales (PCA)

5. **Re-identificación:** en el caso de que el sistema trabaje a partir imágenes de test o directamente de las imágenes obtenidas por el sensor en tiempo real, el vector de características extraído tras la detección es introducido a un clasificador que compara a la persona detectada con las personas para las que se obtuvieron las clases y, si la persona ha sido vista previamente, la re-identifica.

6. **Obtención de métricas:** dado que la base de datos GOPTD1 está etiquetada, se puede saber si el sistema se está comportando según lo esperado. Además, el sistema proporciona información sobre las re-identificaciones lo cual permite obtener diferentes métricas para evaluar el rendimiento del algoritmo desarrollado en este trabajo.

A lo largo del capítulo 3 se explican con mayor detalle los conceptos aquí expuestos y como se ha llevado a cabo del desarrollo de cada una de las secciones que componen el sistema completo. En el capítulo 4 se muestran los resultados obtenidos para, finalmente, presentar las principales conclusiones en el capítulo 5.

Capítulo 2

Estudio teórico

2.1. Introducción

En este capítulo se explican de forma detallada los fundamentos teóricos en los que se basa este trabajo y que son necesarios para comprender el resto de capítulos.

En primer lugar, se expone el principio de funcionamiento de los sensores de profundidad ToF que son los escogidos para la obtención de imágenes de profundidad de este TFG. A continuación, se comparan estos sensores con otros tipos de sensores que aportan información 2.5D y se justifica por qué se han escogido los sensores Time of Flight (ToF) sobre el resto de tecnologías en este campo. A continuación se introducen las características y el funcionamiento básico del sensor Kinect II [4] que es el que se usa en este TFG para la obtención de imágenes.

Finalmente, se procede a la explicación del método PCA ya que es el núcleo de la detección y la re-identificación. Dado que este método suele utilizarse únicamente para la reducción de la dimensionalidad de los datos, se explica de forma concisa su funcionamiento como clasificador.

2.2. Cámaras de tiempo de vuelo

Los sensores de tiempo de vuelo (ToF) destacan por generar imágenes de profundidad en las que cada píxel contiene la distancia del punto de la escena al plano en el que se encuentra el sensor. Es por ello que este tipo de sensores han revolucionado el mercado de la visión artificial ya que permiten generar modelos en 2.5D de forma mucho más sencilla y eficiente.

Los sensores de profundidad trabajan con láseres o diodos LED que emiten en el espectro infrarrojo cercano. Estas fuentes de luz se encuentran dentro del propio sensor y emiten hacia la escena haces de luz pulsada o luz modulada por onda continua (CWM) generalmente senoidal o cuadrada. Los haces de luz se reflejan en los objetos y retornan al sensor de profundidad, el cual contiene un sensor de luz basado en tecnología CMOS que convierte la energía fotónica en corriente eléctrica. La luz recibida tendrá por tanto dos componentes: la luz reflejada por el objeto y la luz ambiental, como queda ilustrado en la figura 2.1. Dado que la información de distancia está contenida dentro de la componente reflejada y el sensor no necesita de una fuente de luz externa para trabajar, la luz ambiental supone una fuente de ruido para el sistema. Hay que destacar que la principal fuente de ruido es la luz solar, ya que la mayoría de fuentes de iluminación artificial no emiten en estas longitudes de onda.

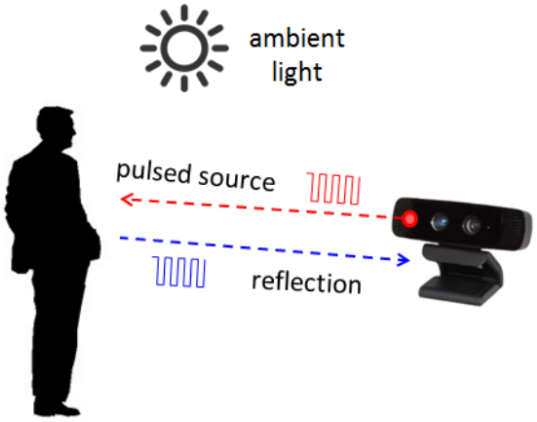


Figura 2.1: Diagrama básico del funcionamiento de un sensor de profundidad. Imagen extraída de [1]

Para obtener la medida de profundidad el sensor calcula el desfase entre el rayo emitido y el rayo reflejado. Sin embargo, la forma de calcular la distancia difiere ligeramente en función de si se trabaja con luz pulsada o modulada.

En el caso de la luz pulsada, la luz incide sobre el objeto durante un tiempo Δt y es recogida por el sensor de forma paralela en todos los píxeles en dos ventanas de tiempo (C1 y C2) no solapadas de duración Δt. Durante el tiempo que incida la luz sobre el sensor, se acumularán cargas en los transistores CMOS del sensor (Q1 y Q2) que son medidas posteriormente para calcular la distancia de acuerdo a la ecuación 2.1. Este proceso puede observarse en la figura 2.2.

$$d = \frac{1}{2} c \Delta t \left(\frac{Q_2}{Q_1 + Q_2} \right) \tag{2.1}$$

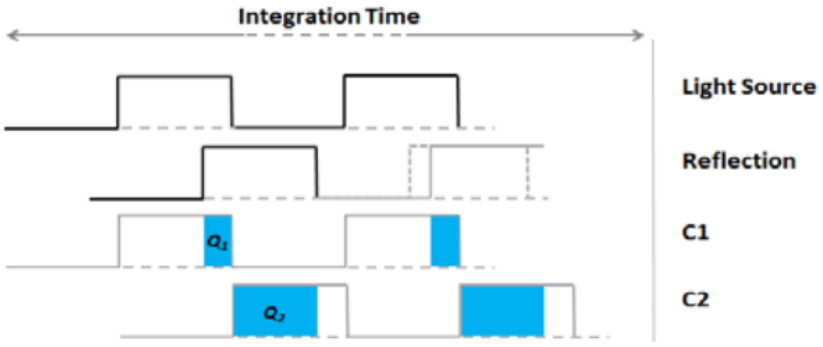


Figura 2.2: Diagrama de la medida de fase con fuente de luz pulsada. Imagen extraída de [1]

Tanto en la ecuación 2.1, como en las siguientes c se refiere a la velocidad de la luz en el vacío, es decir, $3 * 10^8$ km/h.

Por otro lado, en el método de modulación por onda continua (CWM) son necesarias cuatro ventanas de duración Δt con un desfase de 90° entre cada una de ellas. En cada ventana se mide la carga recibida (Q_1, Q_2, Q_3 y Q_4) lo cual permite calcular el desfase y la distancia mediante las ecuaciones 2.2 y 2.3. En la figura 2.3 puede observarse un diagrama en el que se explica de forma gráfica este proceso.

$$\varphi = arctan\left(\frac{Q_3 - Q_4}{Q_1 - Q_2}\right) \tag{2.2}$$

$$d = \frac{c}{4\pi f}\varphi \tag{2.3}$$

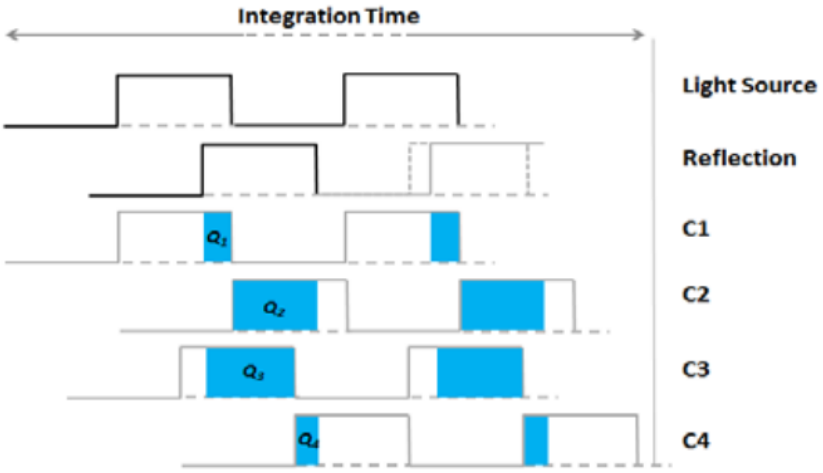


Figura 2.3: Diagrama de la medida de fase con fuente de luz modulada por onda continua (CWM). Imagen extraída de [1]

Como se puede observar, el método de modulación por onda continua (Continuous Wave Modulation (CWM)) depende de la fase y ésta de la función tangente. Por tanto, al tratarse de una función periódica de periodo 2π, no se puede conocer de forma unívoca la distancia ya que cualquier valor de fase igual a $\varphi + 2n\pi$ tiene asociado el mismo valor de la tangente para todo n positivo. Por tanto, las cámaras ToF tienen una distancia máxima (d_{max}) de forma que todas las medidas realizadas deben ser inferiores a esta distancia para que se puedan obtener sin ningún tipo de ambigüedad. Este fenómeno se puede observar en la figura 2.4 donde el objeto a medir se encuentra a una distancia mayor de d_{max} y el sensor no puede conocer la distancia exacta a la cual se encuentra el objeto. La distancia d_{max} se puede calcular sustituyendo φ por 2π en la ecuación 2.3, obteniendo la expresión mostrada en la ecuación 2.4. Observando esta ecuación se deduce que para aumentar la distancia a la cual puede medir la cámara es necesario reducir la frecuencia de modulación. Sin embargo, esta reducción trae como consecuencia una pérdida de precisión ya que supone un aumento de la varianza de las medidas de profundidad obtenidas.

$$d = \frac{c}{2f} \tag{2.4}$$

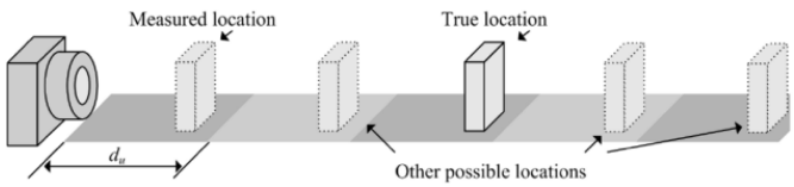


Figura 2.4: Ambigüedad en la distancia para sensores ToF monofrecuenciales. Imagen extraída de [1]

Para solucionar este problema sin tener que buscar un compromiso entre distancia y precisión, los sensores ToF más modernos incorporan técnicas multifrecuencia para extender la distancia. Estas técnicas

consisten en emitir más de una frecuencia de modulación al mismo tiempo de forma que tengan diferente d_{max}. De esta manera, la distancia real del objeto será aquella en la que las frecuencias coincidan con la medida obtenida, como se puede observar en la figura 2.5. Sin embargo, la utilización de estas técnicas no elimina por completo el problema de la ambigüedad en la medida de la distancia, únicamente reducen la frecuencia a la cual coinciden las modulaciones llamada *beat frequency*, es decir, aumentan la distancia máxima del sistema.

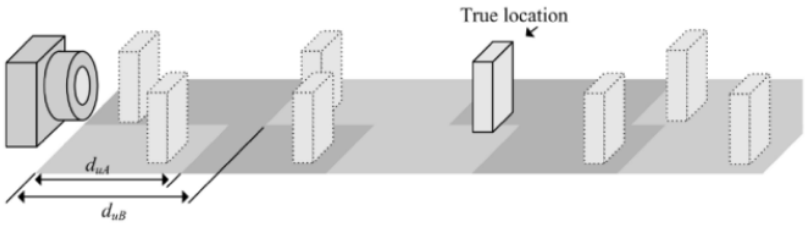


Figura 2.5: Ambigüedad en la distancia para sensores ToF multifrecuenciales. Imagen extraída de [1]

Además del ruido introducido por la luz ambiental comentada anteriormente, existen otras fuentes de errores que afectan especialmente a los sensores ToF. Las más comunes son:

- **Errores sistemáticos**: son errores intrínsecos a la arquitectura y principio de funcionamiento de las cámaras de profundidad. Este tipo de errores suelen generarse en la transformación de luz infrarroja a señal eléctrica y pueden tener gran impacto en la precisión de la medida final. Hay diversas fuentes de errores sistemáticos, pero las más destacables son:
  - Errores en la demodulación de la señal infrarroja
  - Errores en el tiempo de integración
  - Ambigüedad en la amplitud de la señal debido a iluminación no constante y reflexiones no esperadas
  - Errores generados por variaciones en la temperatura
- **Errores no sistemáticos**:
  - Errores por discontinuidades en la escena, llamados *flying pixels* [27].
  - Interferencias por multicamino [28].
  - Distorsión de objetos en movimiento (artefactos de movimiento). [29]
  - Ambigüedad al superar la distancia máxima

De todas estas fuentes de error hay tres que destacan entre las demás y que se van a analizar más en profundidad a continuación: las interferencias por multicamino, las distorsiones generadas por objetos en movimiento y los errores por iluminación no uniforme.

Las interferencias por multicamino están generadas por fenómenos de dispersión característicos de la luz. Al incidir la onda infrarroja sobre una superficie no uniforme, esquinas o bordes, parte de la luz es desviada de su trayectoria original generando ondas adicionales. Esto provoca que al sensor de profundidad lleguen varias ondas lo que genera medidas de distancia erróneas. En la figura 2.6 puede observarse cómo se conseguiría una medida correcta (subfigura a) y ejemplos de casos en los que se producen varias medidas de profundidad para un solo píxel dando lugar a un posible error en la medida(subfiguras b, c y d). Un ejemplo de este efecto sobre una imagen real puede observarse en la figura 2.7.

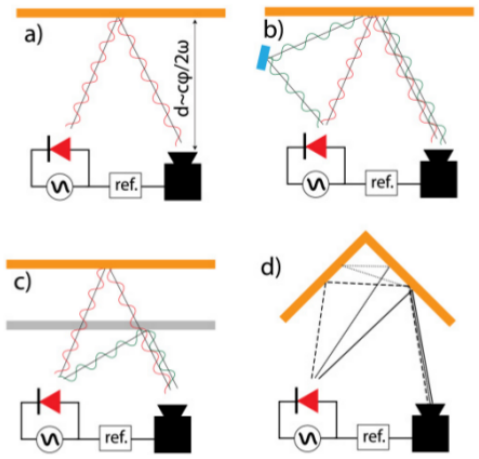


Figura 2.6: Ejemplos de errores por multicamino. Imagen extraída de [2]

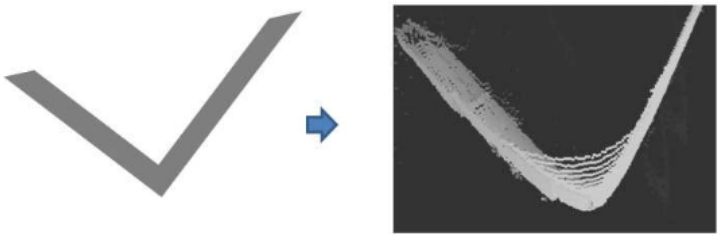


Figura 2.7: Ejemplos de errores por multicamino en esquinas. Imagen extraída de [2]

Las distorsiones por objetos en movimiento son debidas al modo de funcionamiento de los sensores basados en tecnologías CMOS. Los sensores de este tipo no capturan toda la imagen completa en un solo instante de tiempo si no que se capturan fila por fila (o columna por columna) de píxeles, por lo que requieren un cierto periodo de tiempo, llamado tiempo de integración, para capturar toda la imagen. En caso de que se produzca un movimiento durante ese periodo, la imagen no será coherente ya que las medidas realizadas en cada fila o columna pertenecerán a diferentes instantes de tiempo. En la figura 2.8 se puede observar el efecto explicado donde las dos primeras figuras representan la posición inicial y final de la silla y la tercera representa el resultado de la captura de profundidad realizada con la silla en movimiento entre ambas posiciones. Como se puede observar, se producen gran cantidad de errores en las medidas realizadas, especialmente en los bordes de los objetos.

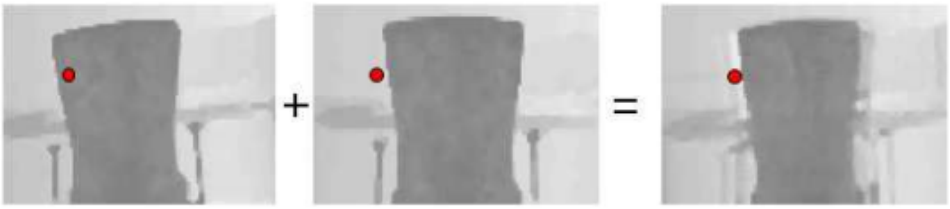


Figura 2.8: Ejemplo de errores por movimientos en la escena. Imagen extraída de [2]

Por otra parte, los errores debidos a iluminación no constante son causados porque las fuentes de luz integradas en los sensores ToF son de luz focalizada en lugar de luz difusa. Esto genera dos tipos de errores dependiendo de la zona de la imagen: en las zonas donde se encuentra focalizada la luz se producen errores por saturación mientras que en los laterales y en las esquinas de la imagen se dan errores por falta de iluminación. Para la caracterización de este error, en [3] se tomaron las medidas de profundidad

de una pared situada a 1.3 metros del sensor. Como se puede observar en la figura 2.9, en las esquinas las medidas de profundidad obtenidas difieren considerablemente de la medida real. Esto será un efecto a tener en cuenta en el desarrollo del trabajo ya que estas zonas presentarán errores que deberán ser evitados o corregidos.

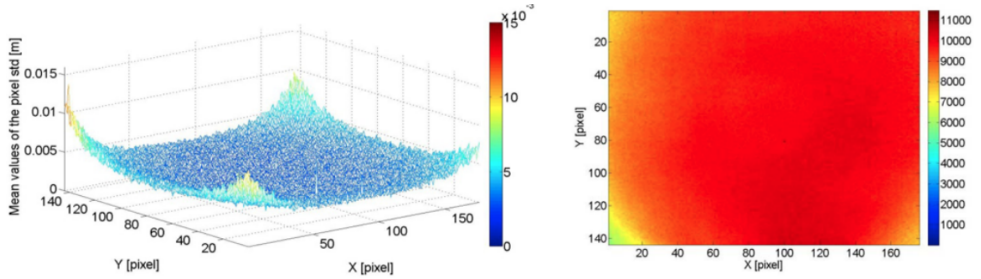


Figura 2.9: Ejemplo de errores por iluminación no constante. Imagen extraída de [3]

Finalmente, debe tenerse en cuenta la capacidad de reflexión de los objetos que aparecen en la escena. Por ejemplo, los objetos oscuros como por ejemplo el pelo de las personas, absorben la luz lo que provoca que el haz que retorna al sensor sea de baja intensidad o que incluso no retorne, dando lugar a la pérdida de información. Por otra parte, los objetos transparentes no reflejarán los haces de luz incidentes lo que imposibilitará la medida de profundidad en algunos casos. En la figura 2.11 se muestran varios ejemplos de superficies que generan problemas de este tipo (2.11a) y cuales han sido las distancias medidas en estos casos (2.11b).

2.2.1. Otras tecnologías de visión 2.5D por computador

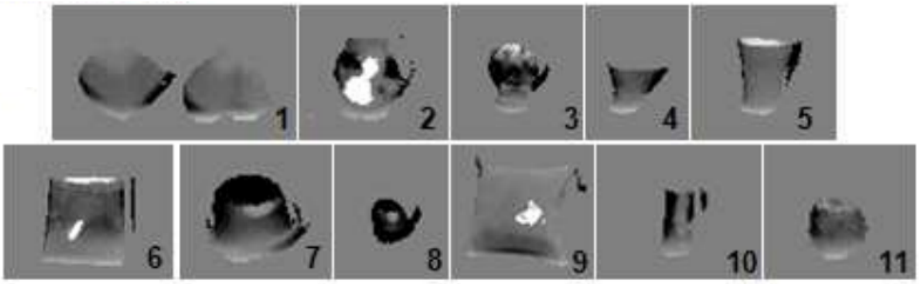
Dada la importancia de la visión en 2.5D en los sistemas de hoy en día, se han desarrollado diversas tecnologías para abordar este tema. En este apartado se va a comparar la tecnología ToF escogida para este proyecto con las dos tecnologías más utilizadas para la visión 2.5D.

En la figura 2.10 se puede observar una tabla con las características principales de cada método que resume lo que se explicará en los dos siguientes apartados.

CONSIDERATIONS	STEREO VISION	STRUCTURED-LIGHT	TIME-OF-FLIGHT (TOF)
Software Complexity	High	Medium	Low
Material Cost	Low	High	Medium
Compactness	Low	High	Low
Response Time	Medium	Slow	Fast
Depth Accuracy	Low	High	Medium
Low-Light Performance	Weak	Good	Good
Bright-Light Performance	Good	Weak	Good
Power Consumption	Low	Medium	Scalable
Range	Limited	Scalable	Scalable
APPLICATIONS			
Game		X	X
3D Movies	X		
3D Scanning		X	X
User Interface Control			X
Augmented Reality	X		X

Figura 2.10: Tabla resumen de las principales tecnologías de visión 2.5D por computador. Imagen extraída de [1]

(a) Ejemplos de superficies problemáticas.



(b) Imágenes de profundidad obtenidas de los objetos mostrados en 2.11a

Figura 2.11: Ejemplos de superficies que dan errores en las medidas de profundidad. Imágenes extraídas de [2]

2.2.1.1. Tiempo de vuelo vs Visión estéreo

La visión en estéreo emplea dos cámaras situadas de forma similar al ojo humano para obtener las medidas de profundidad. Dado un objeto en la escena y dos sensores A y B dispuestas de acuerdo a lo mostrado en la figura 2.12, la profundidad se puede obtener a partir de los ángulos α y β aplicando reglas trigonométricas básicas. La ecuación para el calculo de la profundidad puede verse también en la figura 2.12.

Sin embargo, este método lleva intrínseco un gran reto conocido como el problema de la correspondencia. Este problema consiste en que dado un punto en la escena de uno de los sensores, no se puede conocer

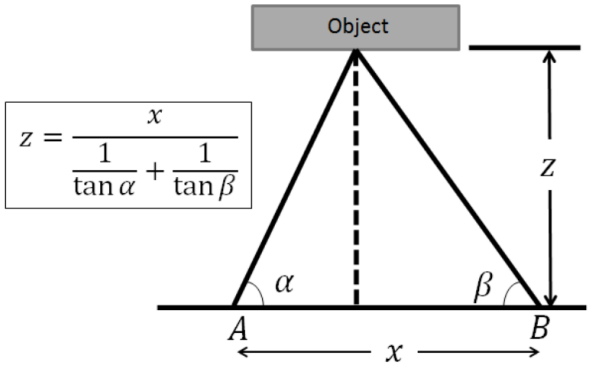


Figura 2.12: Sistema de medición de profundidad de la tecnología de visión en estéreo. Imagen extraída de [1]

a priori donde se encuentra ese mismo punto en la escena del otro sensor. Para resolver este problema es necesario aplicar algoritmos extremadamente complejos que conllevan un gran coste computacional que tienen como objetivo la extracción de características y el '*matching*'. Además, para que el algoritmo obtenga resultados robustos es necesario una intensidad mínima y una alta varianza en los colores de la imagen.

Aunque la visión estéreo tiene algunas ventajas como su bajo coste y una configuración propicia para obtener imágenes similares a las obtenidas por el ojo humano, el error cometido en las medidas de profundidad es una función cuadrática de la distancia. Aunque en los sensores ToF la distancia también influye en la precisión de la medida, este problema se puede solucionar aumentando la energía de la luz infrarroja emitida. Por otra parte, dado el coste computacional que conlleva la visión estéreo, no se trata de una tecnología óptima para trabajar en tiempo real.

2.2.1.2. Tiempo de vuelo vs Sensores de luz estructurada

El funcionamiento de los sensores de luz estructurada se basa en la proyección de patrones conocidos de luz infrarroja sobre los objetos y la inspección de la deformación de dichos patrones (figura 2.13). Para la captura de toda la escena es necesaria la proyección de patrones de forma sucesiva, lo que conlleva la obtención de un bajo número de *frames* por segundo. Además, para que la imagen no se obtenga distorsionada es necesario que la escena no se vea alterada en el proceso de obtención del *frame*, ya que movimientos en la escena pueden dar lugar a errores e incoherencias en las medidas de profundidad.

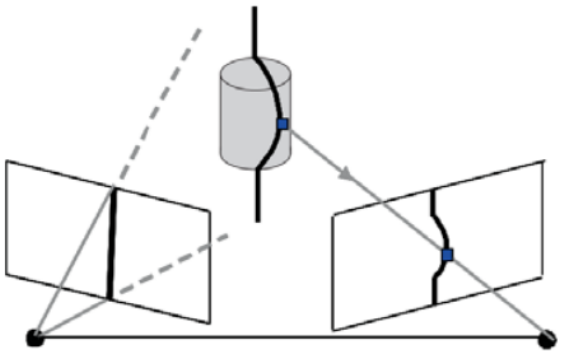


Figura 2.13: Sistema de medición de profundidad de la tecnología de visión en estéreo. Imagen extraída de [1]

Dado que las mediciones dependen de la reflexión del patrón en la escena, las medidas obtenidas son sensibles a interferencias ópticas por lo que es preferible utilizar esta tecnología en aplicaciones que se desarrollen en interiores.

La principal ventaja de esta tecnología reside en que se pueden obtener imágenes de profundidad de alta resolución con sensores de buena calidad. En comparación, los sensores ToF son más baratos y menos sensibles a cambios en la posición o la luz ambiental, además de ser sensores más pequeños.

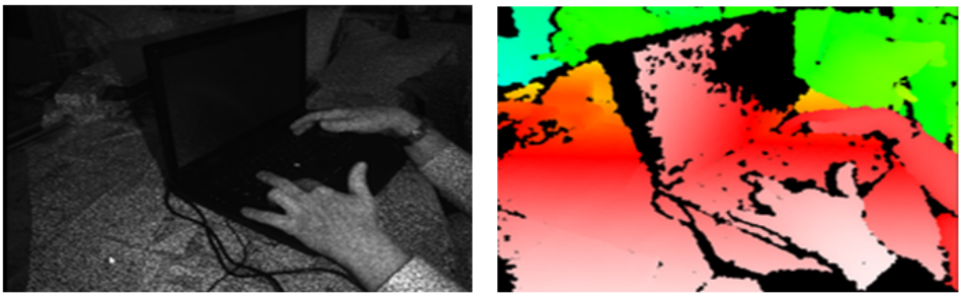


Figura 2.14: Imágenes de intensidad y profundidad obtenidas por el sensor de luz estructurada Kinect v1. Imagen extraída de [2]

2.2.2. Kinect II

Kinect II [4] (figura 2.15) es un sensor de tiempo de vuelo diseñado por Microsoft para la consola Xbox One. Sin embargo, dadas sus prestaciones y su bajo coste, ha sido utilizada ampliamente por la comunidad científica ([30], [31], [32], [33]). Este sensor obtiene tres tipos de imágenes diferentes:

- Imagen de color de 1080p a 30Hz. En situaciones con baja iluminación la frecuencia disminuye hasta 15 Hz.
- Imagen de profundidad con una resolución de 512x424 píxeles a una frecuencia de 30 Hz
- Imagen de intensidad infrarroja con una resolución de 512x424 píxeles a 30 Hz.

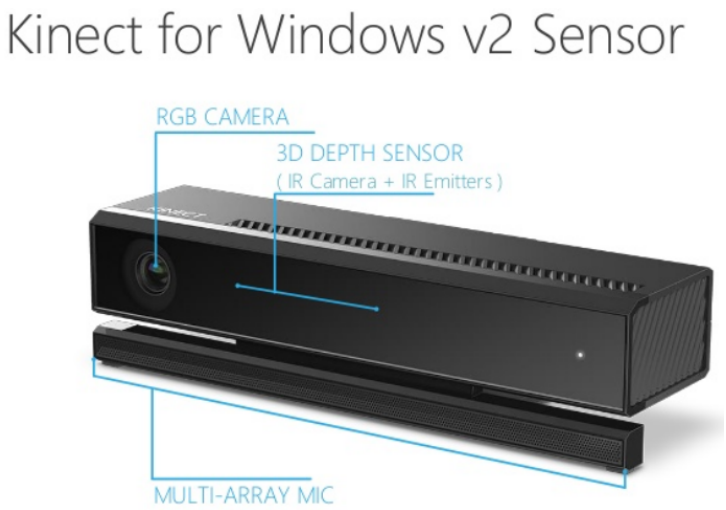


Figura 2.15: Sensor Kinect II. Imagen extraída de [4]

Dada la funcionalidad que tiene este sensor, fue diseñado para la interacción con el usuario por lo que se diseñó con las siguientes prestaciones:

- Campo de visión de 70 grados horizontales y 60 grados verticales
- Apertura focal inferior a 1.1
- Resolución de la imagen de profundidad inferior 1 % de la distancia medida
- Rango medible de 0.8 a 4.2 metros desde la cámara.
- Iluminación propia y funcionamiento independiente de la luz exterior
- Tiempo de exposición máximo de 14 milisegundos
- Tiempo de transferencia inferior a 20 milisegundos utilizando USB 3.0
- Precisión en las medidas de profundidad inferior al 2 % con independencia de la luz, el color, el número de objetos y otras condiciones dentro del rango de operación

Para lograr dichas características el dispositivo Kinect II dispone de un chip que controla todo el dispositivo, un SoC, una fuente de luz y un sensor óptico de acuerdo a lo mostrado en 2.16. El sistema SoC es el encargado de las comunicaciones a través del puerto serie y del manejo de los sensores.

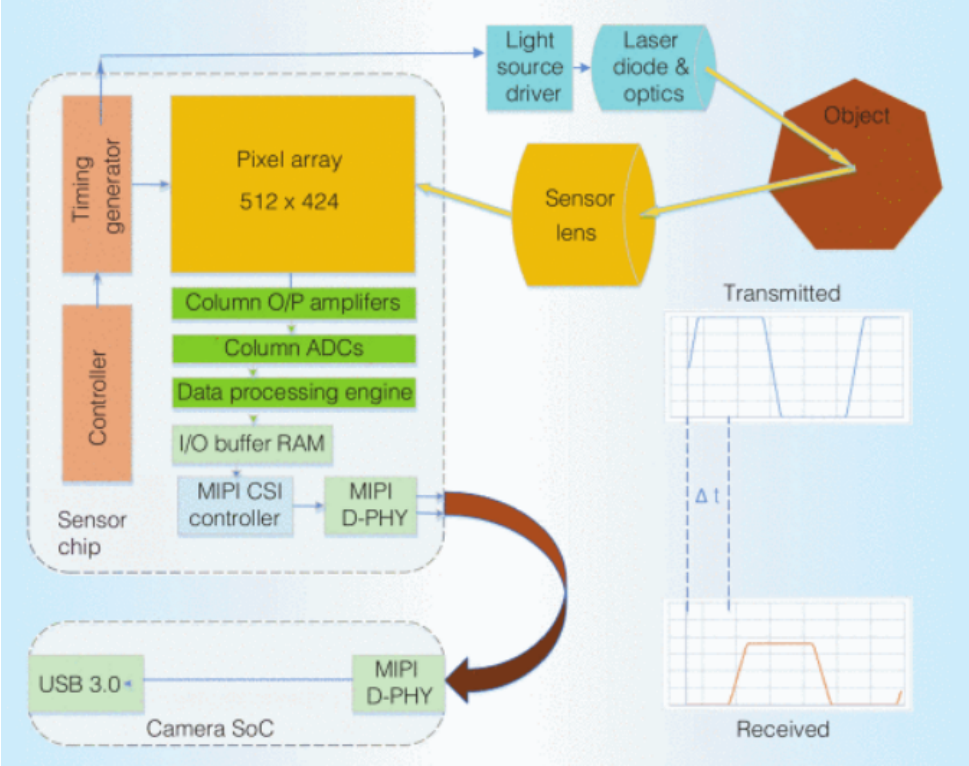


Figura 2.16: Estructura interna del sensor Kinect II. Imagen extraída de [4]

El sensor Kinect II se corresponde con un sensor ToF basado en modulación de onda continua (CWM) construido mediante tecnología CMOS. Como se comentó en el apartado 2.2, estos se sensores se caracterizan por transmitir una señal infrarroja modulada por una señal periódica, en este caso una señal cuadrada, y calcular la distancia del sensor al objeto mediante el desfase entre la onda emitida y la onda reflejada. Este tipo de sensores suelen utilizar cuatro ventanas de tiempo desfasadas 90° para el cálculo de los desfases, sin embargo, como se puede observar en la figura 2.17, el sensor Kinect II solamente posee dos ventanas (A y B) para ello.

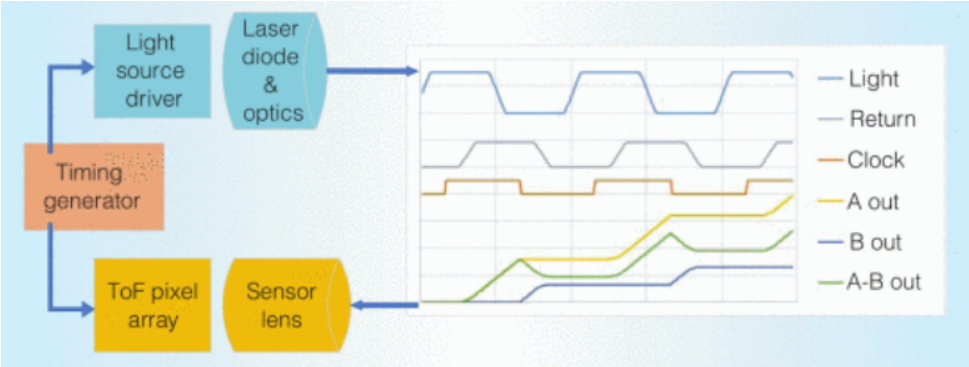


Figura 2.17: Principio de funcionamiento del sensor Kinect II. Imagen extraída de [4]

Para el cálculo de la distancia, el sensor Kinect utiliza lo que se llama el píxel diferencial ([4]), que consiste en calcular el desfase entre la onda emitida y la onda recibida a partir de la diferencia de cargas recogidas entre ambos sensores A-B. El sensor A recoge los fotones y los convierte en carga eléctrica cuando el reloj interno del sistema se encuentra a nivel alto, mientras que el sensor B hace lo propio cuando el reloj se encuentra a nivel bajo. La señal diferencial A-B (2.17) permite calcular el desfase mediante la función arcotangente. Por lo tanto, la forma que tiene el sensor Kinect II para calcular la distancia a la cual se encuentra el objeto consiste en calcular el arcotangente de la señal A-B para obtener el desfase y a continuación aplicar la ecuación 2.3 para obtener la distancia.

En cuanto a la distancia máxima de medida d_{max} característica de los sensores ToF, el sensor Kinect II utiliza múltiples frecuencias de modulación para conseguir medidas de distancia de hasta 10 metros sin ambigüedad. Las frecuencias escogidas para ello son 120 MHz, 80 Mhz y 16 Mhz.

2.3. Análisis de Componentes Principales (PCA)

El Análisis de Componentes Principales (PCA) es un procedimiento matemático que permite extraer las propiedades más importantes de un conjunto de datos. Dadas sus características, esta herramienta matemática es utilizada en el ámbito de la reducción de la dimensionalidad de datos ya que ordena las propiedades de los datos según su importancia. De esta forma permite encontrar patrones dentro de conjuntos de datos por lo que puede ser utilizado como clasificador.

En concreto, el objetivo de PCA es encontrar una proyección de los datos que permita representarlos en términos de mínimos cuadrados de la forma más óptima posible. Para la consecución de este objetivo PCA realiza transformaciones lineales de los datos y les asigna un nuevo sistema de coordenadas.

En este trabajo, PCA se ha utilizado como una herramienta para clasificar los datos extraídos de la información de profundidad obtenida por el sensor Kinect II. En el apartado 2.3.2 se explica en detalle como utilizar PCA como clasificador, pero primero se explican los principios matemáticos sobre los que se sustenta.

2.3.1. Procedimiento matemático

En primer lugar, se forma una matriz con el conjunto de datos D que se quiere comprimir, es decir, aquel del cual se quiere extraer un patrón. A este conjunto de datos se le denomina *clase* ya que los datos comparten un conjunto de características entre sí. Dado este conjunto de datos D formado por N vectores

X_n de P componentes cada uno, siendo $n = 1, 2, ..., N$ se forma la matriz $T \epsilon PxN$ como se muestra 2.5, cuyas columnas son los vectores X_n.

$$T = \begin{pmatrix} X_{1,1} & X_{2,1} & \dots & X_{N,1} \\ X_{1,2} & X_{2,2} & \dots & X_{N,2} \\ \vdots & \vdots & \ddots & \vdots \\ X_{1,P} & X_{2,P} & \dots & X_{N,P} \end{pmatrix} \tag{2.5}$$

A continuación, se calcula la media en la dimensión $j = 1, ..., N$ de acuerdo a lo mostrado en la ecuación 2.6.

$$\mu = \frac{1}{N} \sum_{n=1}^{N} X_n = \begin{pmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_P \end{pmatrix} \tag{2.6}$$

El siguiente paso es obtener una matriz cuyas columnas se corresponden con las desviaciones de la media de los vectores X_n de acuerdo a lo mostrado en 2.7.

$$B = X - hu^T, \quad h[i] = 1, i = 1, ...N \tag{2.7}$$

A partir de la matriz B se puede obtener la matriz de covarianza, la cual prepara el camino para la reducción de la dimensionalidad. En caso de que los datos sean reales, la matriz de covarianza se puede calcular como:

$$C = \frac{1}{N} B * B^T = \begin{pmatrix} cov(X_1, X_1) & cov(X_1, X_2) & \dots & cov(X_1, X_P) \\ cov(X_2, X_1) & cov(X_2, X_2) & \dots & cov(X_2, X_P) \\ \vdots & \vdots & \ddots & \vdots \\ cov(X_P, X_1) & cov(X_P, X_2) & \dots & cov(X_P, X_P) \end{pmatrix} \tag{2.8}$$

donde $cov(X_i, X_j)$ es la covarianza entre las componentes X_i y X_j, que se calcula como muestra la ecuación 2.9.

$$cov\,(X_i, X_j) = \frac{1}{N} \sum_{k=1}^{N} (X_{k,i} - \mu_i) * (X_{k,j} - \mu_j) \tag{2.9}$$

Por último se procede a obtener los autovalores de la matriz de covarianza C, para lo cual se busca la matriz V formada por los autovectores que diagonaliza la matriz de covarianza C:

$$V^{-1} C V = D \tag{2.10}$$

donde D es la matriz diagonal compuesta por los autovalores de C.

Una vez extraídos los autovalores se puede crear la matriz de transformación U, que permite el cambio de sistema de coordenadas original al creado por PCA. Para ello se ordenan los autovectores $(eig_m,\ m = 1, ..., P)$ de mayor a menor autovalor asociado y se colocan en las columnas de U. La matriz U además de realizar la transformación de un espacio a otro permite reducir el número de dimensiones de los datos. Para ello, la matriz U en lugar de formarse con todos los autovectores disponibles se forma con

el número de autovectores m asociados a los autovalores de mayor valor. Al formar un espacio de menores dimensiones ($m < P$) de produce una pérdida de información cuya magnitud dependerá de los valores de los autovalores asociados a los autovectores eliminados. Por norma general, la pérdida de información es mínima ya que la mayor parte de la información está contenida en unos pocos autovectores, que son con los que se busca crear la matriz U.

$$U = (eig_1, eig_2, ..., eig_m) \tag{2.11}$$

Por lo tanto, la matriz U permite proyectar cualquier vector v sobre el nuevo espacio transformado PCA mediante la ecuación 2.12, obteniendo el vector proyectado v_T.

$$v_T = U^T * (v - \mu) \tag{2.12}$$

2.3.2. Funcionamiento como clasificador

Una de las aplicaciones de PCA es como clasificador de aprendizaje supervisado. El aprendizaje supervisado consiste en mostrarle al algoritmo un conjunto de datos de una misma clase, es decir, con las mismas características, para que extraiga un patrón y después pueda decidir si otro conjunto de datos pertenece a dicha clase o no. Para ello, es necesario contar con tantos conjuntos de datos de entrenamiento como clases se quieran crear. Los conjuntos de datos suelen componerse de vectores de características extraídos a partir de un grupo de objetos o personas que tienen características en común.

El entrenamiento de una clase consiste en obtener los autovalores, autovectores y la matriz de transformación del conjunto de datos de entrenamiento correspondientes a la clase deseada de acuerdo a lo explicado en el apartado 2.3.1.

La clasificación se realiza proyectando el vector de características del objeto que se quiera clasificar sobre los espacios PCA de las diferentes clases creadas. Para conocer la pertenencia del vector a una clase determinada se calcula el vector proyectado v_T como muestra la ecuación 2.12.

A continuación, se retorna el vector al espacio dimensional original aplicando la ecuación 2.13.

$$v_B = U * v_T + \mu \tag{2.13}$$

Si se realizó una reducción de la dimensionalidad durante la creación del espacio PCA, el proceso realizado habrá generado un vector diferente del original debido a la pérdida de información que tiene lugar durante la proyección del vector. La diferencia entre el vector original y el vector obtenido tras la recuperación se denomina error de recuperación (ecuación 2.14) y es una medida de la similitud entre el vector del objeto (v_T) y los utilizados para la creación de la clase (X_n).

$$\epsilon = ||v_T - v_B|| \tag{2.14}$$

El error de recuperación permite clasificar los vectores entre las diferentes clases creadas. Para discernir si pertenece o no a una clase, se puede establece un umbral como se explica en [34]. Por otra parte, si se sabe a priori que el objeto pertenece a una de las clases creadas, simplemente se puede escoger la clase que menor error genere ya que es la que más probabilidades tiene de corresponderse con el objeto.

Capítulo 3

Desarrollo

3.1. Introducción

En este capítulo se explica el funcionamiento del sistema de forma extensa y detallada. En primer lugar, se da una visión general del sistema introduciendo las dos etapas que lo componen. A continuación, en los siguientes apartados se explican los diferentes bloques que componen estas dos etapas, que pueden observarse en la figura 3.1, la cual se incluyó en la introducción pero se repite aquí para facilitar la lectura del documento.

Como se ha comentado anteriormente, el sistema se compone de dos etapas, una primera etapa de entrenamiento que solo se ejecuta una vez al comienzo (*offline*), y una etapa de re-identificación que se lleva a cabo con cada nueva imagen de entrada (*online*). La selección del modo de funcionamiento (*offline* u *online*) y de la entrada de datos se explican en el anexo A, ya que no se considera relevante pare entender el funcionamiento del sistema que es el objetivo de este capítulo y podría distraer al lector. A continuación se va a proceder a explicar la función de ambas etapas. Cabe destacar que, tal como se puede ver en la figura 3.1, existen algunos módulos comunes a modos de funcionamiento, como la detección de personas o la extracción de descriptores de características.

- **Etapa de entrenamiento (*Offline*)**: permite crear las clases correspondientes a cada persona que se utilizarán después en la etapa *Online*. En primer lugar, abre las secuencias de imágenes de entrenamiento indicadas por el usuario y va extrayendo las imágenes de profundidad. Por cada fotograma se ejecuta el detector, se extrae el vector de características de las personas en la escena y se almacenan en un archivo. Dado que son imágenes de entrenamiento, sólo aparecerá una persona en la secuencia, por lo que todos los vectores almacenados pertenecerán a la misma persona. Una vez finalizada una secuencia, se crea la clase PCA correspondiente, se guarda para su posterior utilización y se continúa con la siguiente secuencia. Al crear la clase de la última secuencia indicada por el usuario, el programa finaliza con todas las clases PCA creadas almacenadas en el mismo directorio.

- **Etapa de re-identificación (*Online*)**: lleva a cabo la re-identificación de personas a partir de imágenes de profundidad que pueden provenir de secuencias de test previamente grabadas o directamente de uno de los sensores en tiempo real. Independientemente de la fuente, los datos de profundidad son enviados al detector para localizar a las personas en la escena y extraer sus vectores de características. Una vez extraído el vector, se envía al clasificador basado en PCA que devolverá la identidad de la persona más similar al vector de entrada de entre las clases guardadas en la etapa

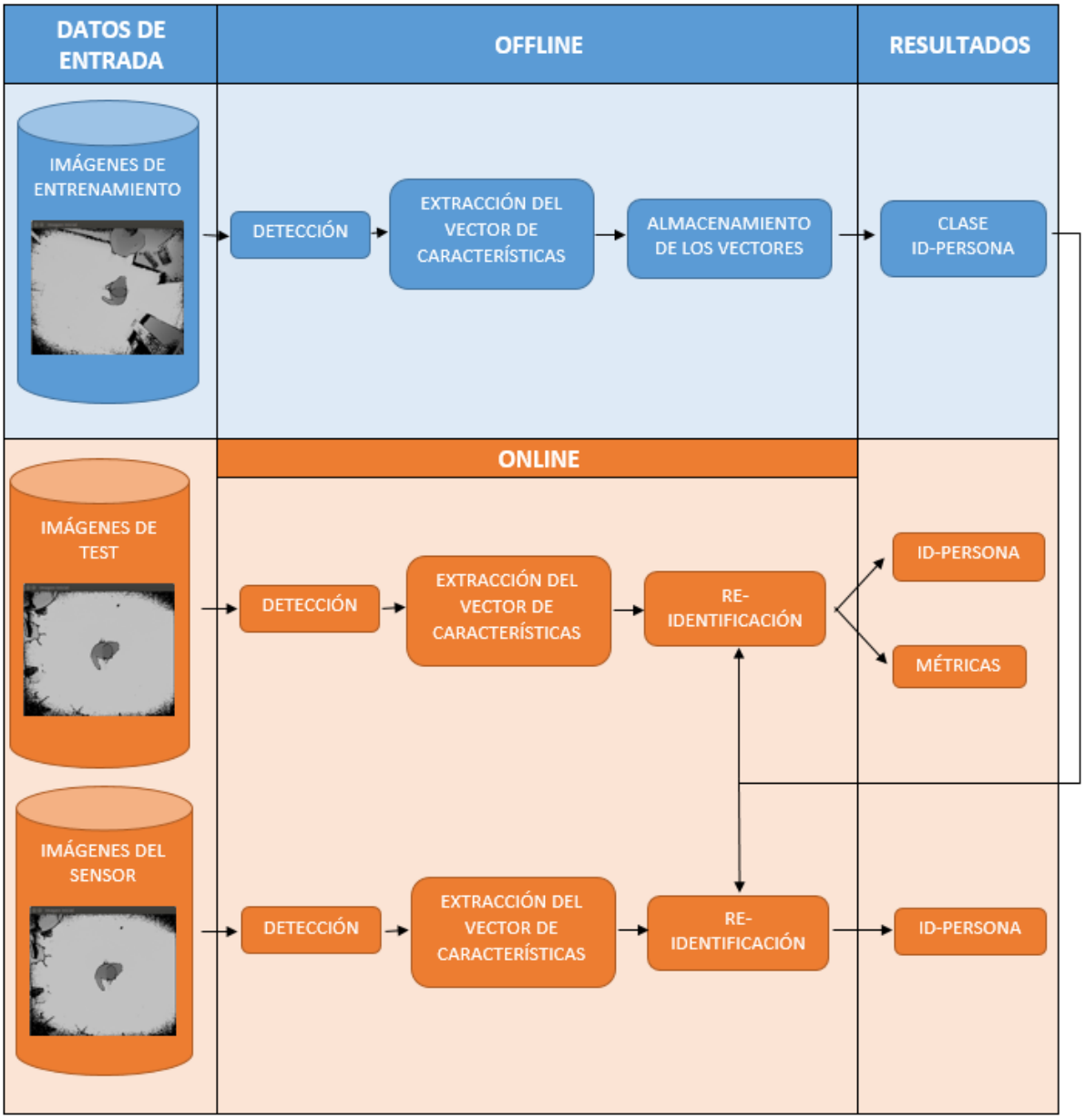


Figura 3.1: Diagrama de bloques general del sistema completo desarrollado en este TFG.

de entrenamiento *off-line*. En el caso de que los datos de entrada sean las secuencias de test, además de la identidad de la persona se calculan métricas sobre el funcionamiento del sistema fotograma por fotograma para su análisis posterior. Dado que el análisis se hace fotograma por fotograma no se mantiene información de estados anteriores, es decir, cada *frame* se analiza de forma independiente al resto. Al terminar una secuencia, se guardan las métricas correspondientes en un archivo asociado a la secuencia y se prosigue con la siguiente secuencia que se almacenará en otro archivo.

3.2. Obtención de las imágenes de profundidad

3.2.1. Obtención de las imágenes de entrenamiento y test

Tanto las imágenes de test como las imágenes de entrenamiento se extraen a partir de secuencias grabadas previamente pertenecientes a la base de datos GOTPD1 [35]. Esta base de datos se grabó con dos sensores Kinect II situado en posición cenital a una altura de 3.4 metros del suelo. Además, las secuencias se encuentran etiquetadas manualmente e indican la posición e identidad de la persona. En las figuras 3.2 y 3.3 se pueden ver un ejemplos de las imágenes obtenidas por los sensores de esta base de datos.

Las secuencias de la base de datos GOPTD1 se encuentran almacenadas en formato ".z16", llamado así porque la información de profundidad se guarda en binario, donde cada píxel representa la distancia a un punto en milímetros y se almacena como un entero de 16 bits con signo.

Figura 3.2: Ejemplo de imágenes pertenecientes a la base de datos grabadas por el sensor T0.

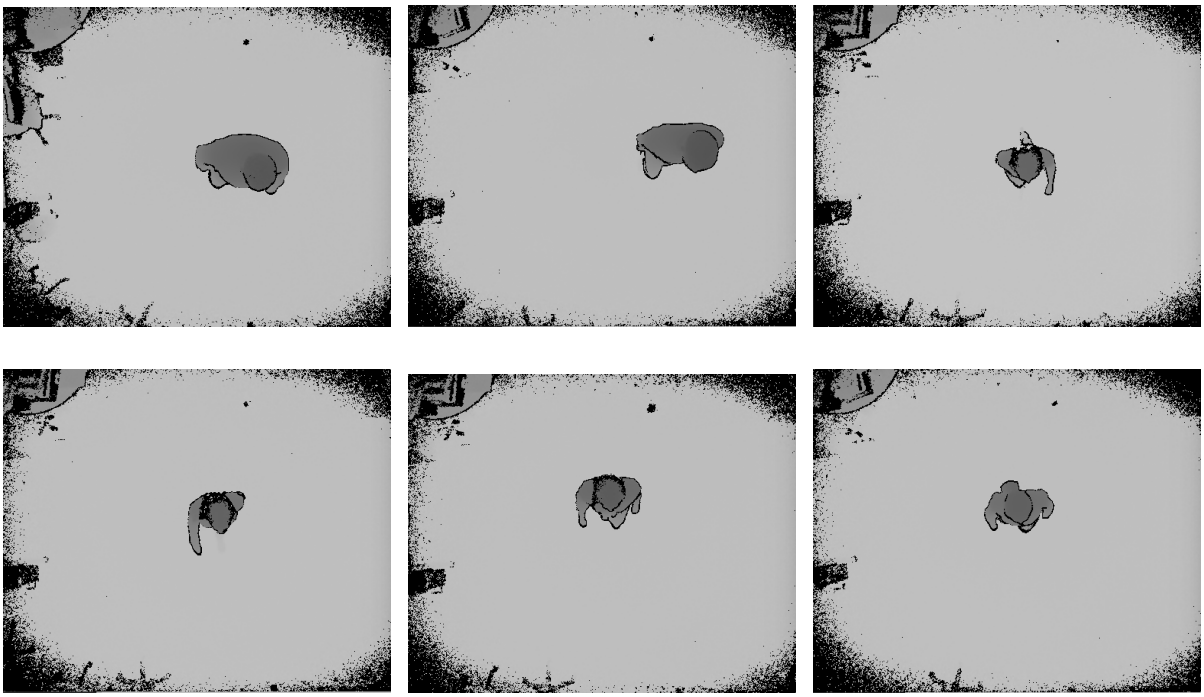


Figura 3.3: Ejemplo de imágenes pertenecientes a la base de datos grabadas por el sensor T1.

3.2.2. Obtención de las imágenes del sensor en tiempo real

Para la extracción de las imágenes del sensor se ha utilizado la librería Libfreenect2 ([36]), que proporciona las herramientas necesarias para controlar los diferentes flujos de información que proporciona el sensor Kinect II.

Para facilitar el trabajo con el sensor se han desarrollado tres funciones encargadas de gestionar toda la información relacionada con el sensor, de forma que el funcionamiento de este sea transparente tanto para el usuario como para futuros desarrolladores. Antes de su ejecución las clases listadas previamente deben estar creadas. Un ejemplo de como se pueden crear se encuentra en A.1. Las funciones creadas para la comunicación con el sensor son:

- **opencamera**: inicializa las variables necesarias para el funcionamiento del sensor y establece el canal de transmisión de información entre el sensor y el ordenador. El canal puede establecerse mediante OpenGL,Cuda, OpenCL o la CPU, siendo elegido el método por el usuario. A continuación, se establece la configuración de los filtros y distancias del sensor y se abre el canal para la transmisión de datos. Después, se escogen los tipos de información a transmitir entre color, profundidad e infrarrojos y se comienza la transmisión de aquellos tipos que se hayan escogido.

- **getframe**: almacena el siguiente conjunto de *frames* obtenido por el sensor dentro de un cierto periodo de tiempo. De cada instante se reciben los tres tipos de información: color, profundidad e infrarrojos. A continuación, se convierte en formatos representables y adaptados al sistema y se almacenan los diferentes tipos de información en una variable de tipo *map<std::string,cv::Mat>* que ha sido pasada por referencia a la función. Se retorna una variable de tipo bool siendo verdadero que se ha realizado la recepción y almacenamiento de la información correctamente y false que ha ocurrido un error durante el proceso.

- **end**: cierra los flujos de información que se le pasan por referencia y libera la memoria ocupada. Esta función debe ser llamada al final del programa para que el sensor siga funcionando de forma correcta la siguiente vez que se utilice.

Las tres funciones implementadas permiten la adquisición de información del sensor en tiempo real de forma sencilla y transparente para el programador, solo es necesario llamar a las funciones *opencamera* y *end* al principio y final del programa respectivamente, y a la función *getframe* cada vez que se quiera obtener un *frame* del sensor.

3.3. Detección de personas

Para la detección de personas se ha utilizado el detector desarrollado por David Fuentes Jiménez y Raquel García Jiménez en sus TFGs ([5] y [6]). A continuación se describe de forma resumida su funcionamiento ya que es necesario para poder entender el sistema completo, sin embargo no se entrará en detalles ya que si el lector desea profundizar más en esta parte del sistema están a su disposición los TFGs de ambos estudiantes ([5] y [6]) así como algunos artículos que han publicado sobre este tema ([37] y [38]). En estos documentos se encuentra una explicación detallada de todo lo relacionado con el detector, como sus principios teóricos, su implementación y los resultados experimentales que se obtuvieron con la base de datos utilizada en este TFG.

En la figura 3.4 se pueden ver las etapas que componen el detector y que se explican en los siguientes apartados.

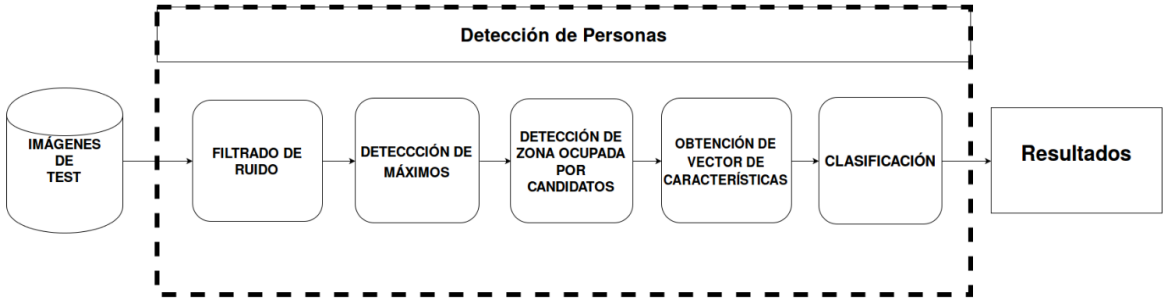


Figura 3.4: Diagrama general de bloques del detector. Imagen extraída de [5]

3.3.1. Preprocesado de la información de profundidad

Las imágenes de profundidad, independientemente de la fuente de la que provengan, representan la información de la misma manera: la información viene dada en una matriz donde cada una de las posiciones representa un píxel de la escena. El contenido de cada posición es la distancia desde el punto en la escena al plano en el que se encuentre el sensor, sin embargo dado que es más cómodo e intuitivo trabajar con alturas en lugar de con las distancias al sensor, se ha decidido obtener la altura respecto al suelo de cada punto. Para ello primero es necesario conocer la altura a la que se encuentra el sensor respecto del suelo (que en el caso de las secuencias utilizadas en este TFG es $h_{sensor} = 3{,}4$ metros). Después, para obtener la matriz de alturas en lugar de la distancia al plano del sensor es necesario aplicar la ecuación 3.1 con todos los puntos de la matriz dada por el sensor.

$$H_{m,n} = h_{sensor} - Z_{m,n} \tag{3.1}$$

Dado que la imagen obtenida por el sensor contiene gran cantidad de píxeles no válidos, es necesario realizar un filtrado sobre la imagen para eliminar el ruido y mejorar los resultados del detector. Existen dos fuentes de píxeles no válidos: los identificados por el propio sensor los cuales tienen un valor especial asignado por el mismo y los identificados por el sistema, que son aquellos que no pueden corresponderse con personas ($H_{m,n} > 2{,}3m$ o $H_{m,n} < 0{,}5m$), a los cuales se les asigna el valor especial asignado por el sensor a los otros píxeles erróneos.

Para realizar el filtrado, en primer lugar se buscan píxeles válidos en torno a cada píxel no válido con un nivel de vecindad 2 (área de 5x5 píxeles). Al finalizar la búsqueda, se le asigna el valor medio de los píxeles válidos en la vecindad especificada al píxel no válido, ya que se considera que existe un nivel de correlación suficientemente alto entre los valores de los píxeles de tal proximidad. A continuación, a partir de la matriz de alturas obtenida tras este proceso, se le aplica un filtro de mediana de nueve elementos, obteniendo la matriz de alturas final que se utilizará posteriormente para la detección y la re-identificación de personas.

En las figuras 3.5 y 3.6 se muestran dos ejemplos del resultado del proceso de filtrado. En ambas figuras se muestra la imagen de profundidad obtenida por el sensor a la izquierda y la imagen de alturas tras el filtrado a la derecha.

Figura 3.5: Ejemplo 1 de los resultados del algoritmo de filtrado.

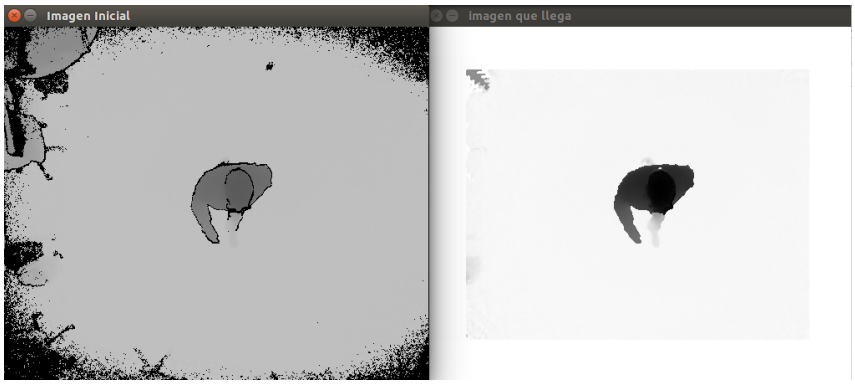


Figura 3.6: Ejemplo 2 de los resultados del algoritmo de filtrado.

3.3.2. Detección de máximos

La siguiente etapa del algoritmo consiste en la detección robusta de máximos en la matriz de alturas mediante un algoritmo de búsqueda de máximos locales. Este algoritmo comienza dividiendo la matriz de alturas en subregiones (SR) cuadradas de DxD píxeles . El número de SR y su tamaño dependerá de la altura a la cual se encuentre la cámara, de los usuarios, la precisión requerida y la velocidad de procesamiento que se precise. De esta forma, cuanta mayor precisión se requiera, más pequeñas serán las SR pero mayor será el tiempo de procesamiento.

Una vez dividida la matriz, se buscan los máximos de cada una de las SR y se agrupan de acuerdo al orden que tenían las SR en la matriz original formando la matriz H^{maxSR} (ecuación 3.2).

$$\left[H^{maxSR}\right] = \begin{pmatrix} h_{1,1}^{maxSR} & h_{1,1}^{maxSR} & . & . & . & h_{1,1}^{maxSR} \\ h_{1,1}^{maxSR} & h_{1,1}^{maxSR} & . & . & . & h_{1,1}^{maxSR} \\ . & . & . & & & . \\ . & . & & . & & . \\ . & . & & & . & . \\ h_{1,1}^{maxSR} & h_{1,1}^{maxSR} & . & . & . & h_{1,1}^{maxSR} \end{pmatrix} \tag{3.2}$$

Los máximos encontrados son evaluados para decidir si son posibles candidatos a persona, considerando como válido cuando supera la altura mínima para considerar que es una persona ($h_{pmin} = 1$ metro) y además es mayor o igual que los máximos de las SR contiguas. Los máximos que cumplen estos criterios se consideran candidatos a persona $P_{r,c}^{k}$ con $1 \leq k \leq N_P$, siendo N_P el número de máximos válidos total detectado.

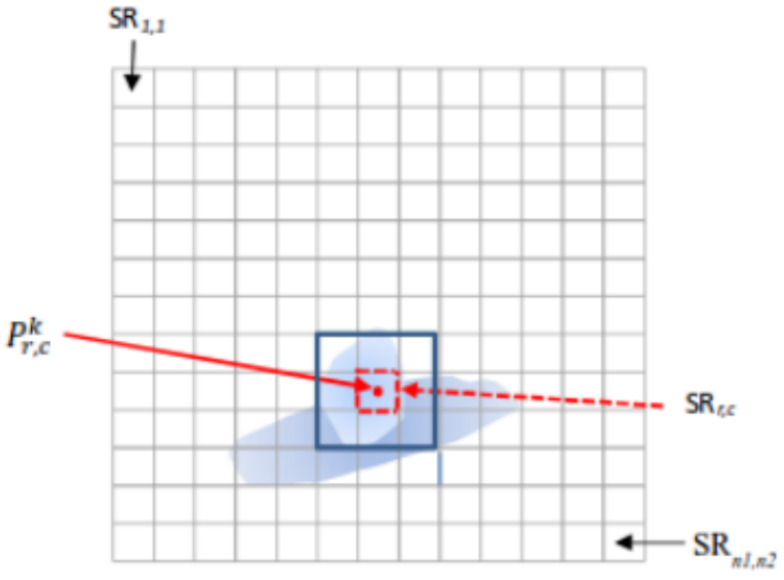


Figura 3.7: Ejemplo de las SR en un fotograma. Imagen extraída de [5]

Puede darse el caso de que las SR sean tan pequeñas que se detecten varios máximos que cumplen los criterios para ser candidatos en una misma persona, por lo que antes de continuar se compara cada máximo con los más próximos dentro de un radio prefijado y se escoge aquel que tenga mayor valor, eliminando los demás.

3.3.3. Crecimiento de regiones alrededor de cada máximo

Para poder extraer las características de la persona es necesario definir una región de interés (ROI) alrededor de cada máximo que contenga la cabeza, hombros y cuello de las personas, lo cual supone una distancia de $h_{interest} = 40cm$ elegida teniendo en cuenta consideraciones antropométricas ([39] y [40]). Esta región de interés se establece mediante un algoritmo de crecimiento de regiones, el cual partiendo del máximo válido $P^k_{r,c}$, estudia los máximos de las SR contiguas en las ocho direcciones indicadas en 3.8 hasta un nivel de vecindad 4.

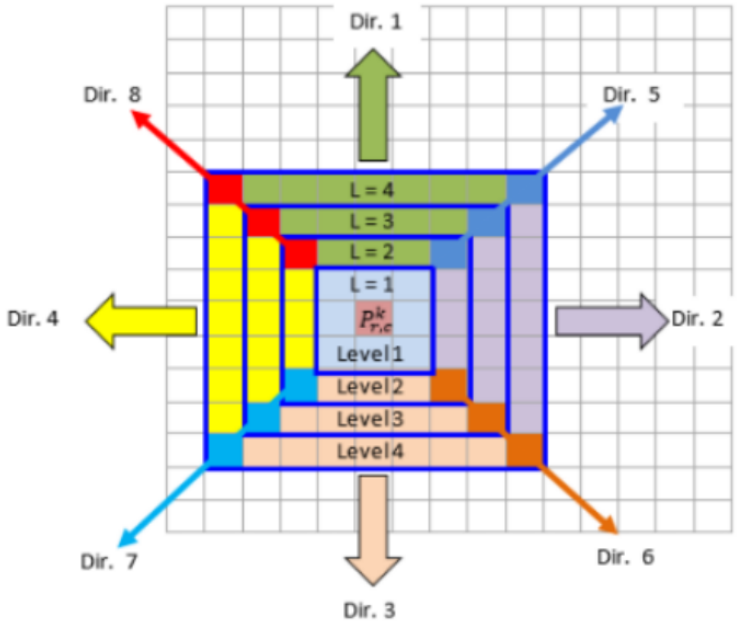


Figura 3.8: Diagrama de direcciones de expansión de la ROI. Imagen extraída de [5]

Exceptuando la SR en la que se encuentra el máximo que siempre pertenece a la ROI, el resto de SR deben cumplir entre otras las siguientes características para pertenecer a la ROI:

- El máximo asociado a la SR que se está analizando debe encontrarse dentro de $h^{interest}$.
- La SR del nivel anterior debe pertenecer a la ROI y, en el caso de las direcciones lineales, el nivel anterior tiene que tener como mínimo un número de SR de al menos su nivel de vecindad L menos uno.
- El valor del máximo asociado a la SR debe ser menor que el máximo de la SR anterior en esa misma dirección y mayor que el máximo siguiente, con el fin de poder separar a personas que se encuentren próximas en la imagen.

En la figura 3.9 se puede observar el resultado de aplicar las condiciones explicadas sobre una imagen real: en color rojo se encuentra representado el máximo candidato a persona, en azul las SR que cumplían las condiciones de pertenencia a la ROI, y en amarillo y verde las SR donde se ha realizado la búsqueda pero no cumplían las condiciones.

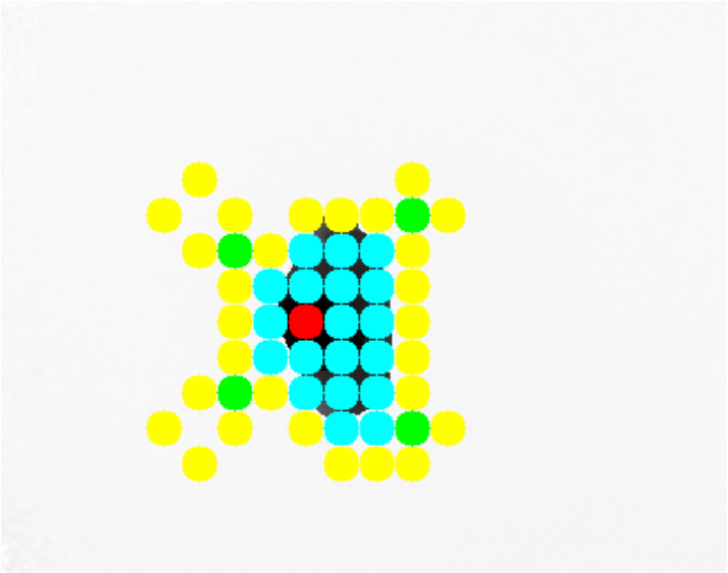


Figura 3.9: Ejemplo de funcionamiento del algoritmo de crecimiento de regiones sobre una imagen real. Imagen extraída de [5]

3.3.4. Extracción del vector de características para la detección

Al delimitar la ROI, se define el conjunto de puntos candidatos a pertenecer a una persona, a partir de los cuales es posible extraer un vector de características que permite discriminar entre personas y otros elementos que puedan aparecer en la escena.

Para la construcción del vector de características que define a la persona, se divide la región $h_{interest}$ en veinte franjas de dos centímetros cada una. A continuación, se cuenta el número de píxeles pertenecientes a la ROI contenidos en cada franja y se almacenan en un vector dando lugar a una primera versión del vector de características (ecuación 3.3). En la figura 3.10 se puede ver una representación de las franjas sobre el perfil de una persona que, aunque no se corresponde con la perspectiva de los sensores utilizados en este TFG, puede ayudar al lector a comprender el proceso que se está llevando a cabo.

$$v_s = [v_1, v_2, ..., v_{20}] \tag{3.3}$$

Las componentes del vector dependen de la altura, pose, oclusiones, tipo de pelo, etc. que en el caso de la detección de personas, suponen una fuente de ruido. Para reducir el efecto del ruido, se agrupan los elementos del vector en un total de cinco componentes de acuerdo a lo mostrado en la figura 3.10, dando lugar a un segundo vector de características (3.4). Como se puede observar, la zona del cuello no se incluye en ninguna de las componentes del nuevo vector. Esto es debido a que al estar la cámara en posición cenital, esta zona presenta mucha oclusión y ruido y no es suficientemente significativa.

$$v^R = [v_1^R, v_2^R, v_3^R, v_4^R, v_5^R] \tag{3.4}$$

Sin embargo, se dan casos en los que el máximo no se corresponde con el centroide de la persona. En estos casos, para obtener el vector de características se aplica la ecuación 3.5 para solucionar el problema, ya que elimina las componentes que presentan ruido. Para ello, se escoge de entre las tres primeras franjas aquella que más píxeles contenga. Si la mayor se corresponde con la primera franja, se agrupan las siguientes nueve franjas de tres en tres formando las primeras componentes del nuevo vector ($v_1^R, v_2^R y v_3^R$). En caso de que la que contenga mayor número de píxeles sea la segunda o tercera franja, la agrupación

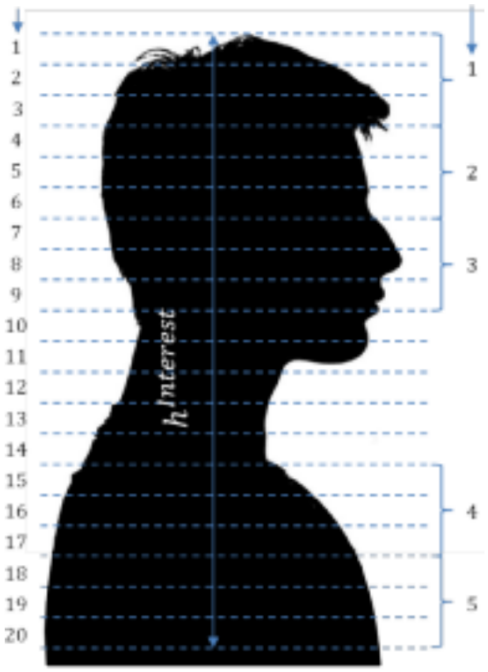


Figura 3.10: División en franjas de una persona. A la izquierda las franjas de dos centímetros y a la derecha las agrupaciones realizadas para formar el vector v^R. Imagen extraída de [5]

comienza por la franja anterior y se realiza también de tres en tres. A continuación se busca la franja con el número máximo de píxeles entre el resto de franjas que aun no han sido agrupadas (10-20). A partir de ella se obtienen la componente v_4^R del vector sumando los píxeles de la franja anterior, la franja obtenida y la franja siguiente. Finalmente, la componente v_5^R se calcula sumando las siguientes tres franjas.

$$
\begin{aligned}
&v_i = max(v_1, v_2, v_3),\ i = 1, 2, 3 \\
&if\ \ i = 1 \rightarrow v_1^R = \sum_{s=1}^{3} v_s;\ \ v_2^R = \sum_{s=4}^{6} v_s;\ \ v_3^R = \sum_{s=7}^{9} v_s; \\
&if\ \ i = 2, 3 \rightarrow v_1^R = \sum_{s=i-1}^{i+1} v_s;\ \ v_2^R = \sum_{s=i+2}^{i+4} v_s;\ \ v_3^R = \sum_{s=i+5}^{i+7} v_s;
\end{aligned}
\tag{3.5}
$$

$$
\begin{aligned}
&v_j = max(v_{10}, v_{11}, ..., v_{20}),\ j = 10, 11, ..., 20 \\
&v_4^R = \sum_{s=j-1}^{j+1} v_s; \\
&v_5^R = \sum_{s=j+2}^{j+4} v_s;
\end{aligned}
$$

Además de estas cinco componentes, se ha añadido una componente de forma v_6^R, la cual no esta relacionada con el número de píxeles sino con la distribución de estos. El objetivo de esta sexta componente es caracterizar la elipsoidalidad de la cabeza. Para ello se hayan los ejes mayor y menor que conforman la cabeza y se calcula la relación entre ellos, la cual se corresponde con la sexta componente del vector.

El vector de características resultante de este procedimiento tiene gran dependencia de la altura de la persona, presentando grandes variaciones ante distintos valores de esta. Sin embargo, existe una relación

entre la variación del número de píxeles en cada franja y la altura de la persona como puede verse en la figura 3.11. Para caracterizar la relación entre ambos factores se ha empleado el algoritmo de Levenber-Marquardt, obteniendo un polinomio de segundo orden (línea azul en la figura 3.11) que permite obtener la función de normalización 3.6.

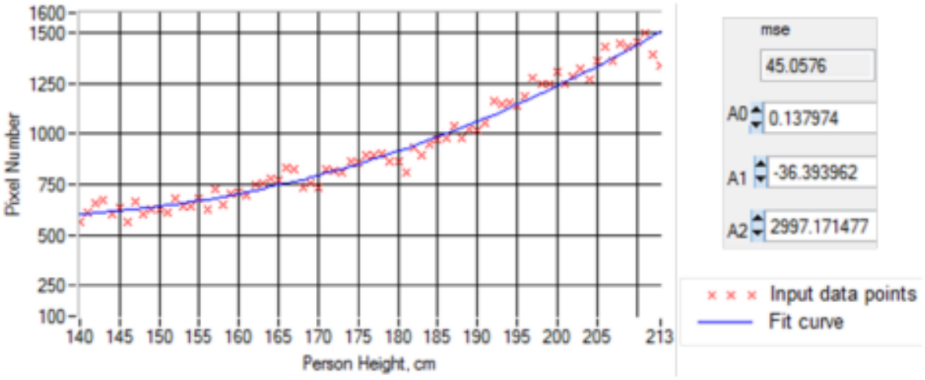


Figura 3.11: Curva de ajuste generada por el algoritmo de Levenber-Marquardt. Imagen extraída de [5]

$$p = 0,1379 * (h^{max})^2 - 36,3939 * h^{max} + 2997,1714 \tag{3.6}$$

3.3.5. Clasificación

Por último, se procede a la clasificación del vector de características con el objetivo de determinar si el máximo se corresponde con una persona o se trata de algún otro elemento. Para ello se utiliza un clasificador basado en PCA. Como ya se explicó en 2.3, dado que se trata de un clasificador de aprendizaje supervisado es necesaria una etapa de entrenamiento con datos conocidos. En este trabajo, al igual que en trabajos previos [5,6], se han utilizado dos clases diferentes creadas a partir de secuencia de la base de datos GOTPD1 [35]: una para personas con sombrero y otra para personas sin sombrero, de forma que se pueda identificar cuando un máximo es persona independientemente de los complementos que lleve. Estas clases se crearon a partir de un gran número de secuencias con el objetivo de incluir la mayor diversidad de personas con diferentes atributos físicos (altura, cabello, complexión, etc.) posible. Para su creación se utilizó un total de tres autovectores en la matriz de transformación. En las figuras 3.12 y 3.13 se pueden observar algunos ejemplos de los vectores utilizados para el entrenamiento de estas clases.

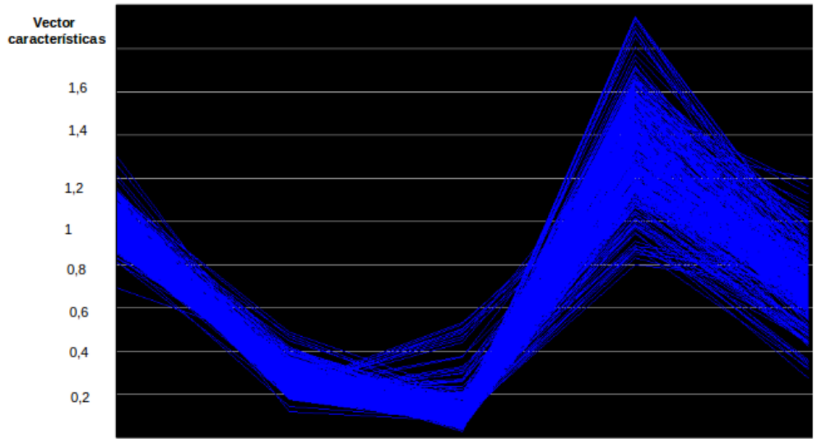


Figura 3.12: Vectores de entrenamiento de la clase pelo corto. Imagen extraída de [6]

Para decidir si el máximo es una persona o no, se proyecta su vector de características sobre los espacios creados mediante PCA de las clases persona y persona con sombrero. A continuación, se recupera el vector al sistema de coordenadas original dando lugar a un vector similar al anterior pero que ha sufrido una pérdida de información debido al cambio en el número de dimensiones. Después, se calcula la distancia euclídea entre el vector original y el vector recuperado. Si la distancia es inferior a $e_{umbral} = 0,5$ en cualquiera de las dos clases se considera persona, en caso contrario el máximo se marca como no persona y se descarta para la re-identificación. El umbral se escogió de forma experimental tras la realización de diversos experimentos.

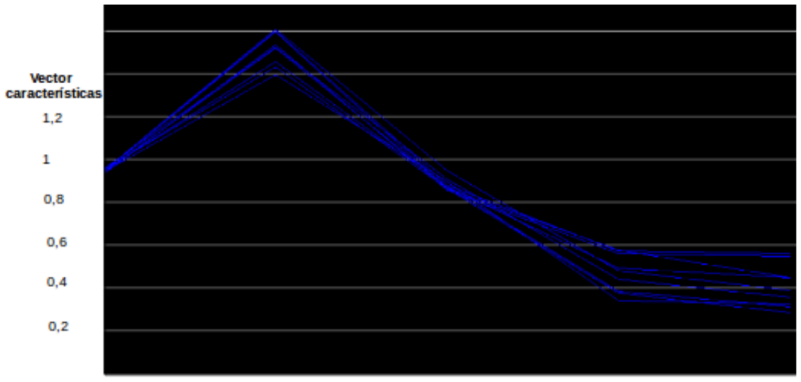


Figura 3.13: Vectores de entrenamiento de la clase sombrero. Imagen extraída de [6]

3.4. Extracción del vector de características para la re-identificación

3.4.1. Planteamiento inicial

La obtención de la información para formar el vector de características destinado a la re-identificación se realiza durante la fase de detección. En primer lugar, se forma un vector de veinte componentes que contienen el número de píxeles en cada franja de dos centímetros perteneciente a $h^{interest} = 40cm$. Como se explicó en el apartado 3.3.3, la imagen presenta ruido por lo que el máximo puede no encontrarse centrado. En el caso de la detección se procedía a aplicar un algoritmo para agrupar las secciones de acuerdo al número de píxeles en cada una. Para la re-identificación se pretende mantener el máximo de información posible por lo que no se agrupan las secciones y, por lo tanto, no se pueden aplicar las mismas ecuaciones que en el caso de la detección.

Para la eliminación del ruido en este caso se ha decidido analizar las tres primeras franjas ($v_{s,1}, v_{s,2}$ y $v_{s,3}$). En el caso de que exista ruido y por lo tanto el centroide no este correctamente ubicado, las primeras franjas presentan un número de píxeles muy inferior con respecto a las siguientes. Por lo tanto, aplicando la ecuación 3.7 se escoge la primera franja de entre $v_{s,2}$ y $v_{s,3}$ que presente un 85 % más de píxeles que la franja $v_{s,1}$ o, en caso de que ninguna cumpla dicha condición, la primera franja $v_{s,1}$. El valor del porcentaje de píxeles necesario se ha escogido de forma experimental de acuerdo a las medidas obtenidas en las pruebas realizadas. A partir de la franja escogida, se seleccionaran esa y las siguientes diecisiete componentes para formar un nuevo vector de características, descartando aquellas que queden fuera de la ventana establecida. De este proceso, se obtiene un vector de dieciocho componentes sin importar cual haya sido la franja seleccionada. El vector anterior, se completa con la altura de la persona como última componente.

$$
\begin{aligned}
&v_{top} = max(v_1, (1,85 * v_2)), \quad top = 1,2 \\
&if \quad v_{top} > (1,85 * v_3) \;\rightarrow\; v_i^I = v_{top+i} \; i = 0,..,17 \\
&else \quad v_i^I = v_{i+3} \; i = 0,..,17
\end{aligned}
\tag{3.7}
$$

3.4.2. Ampliación del vector de características

Una vez creado el vector descrito en el apartado anterior, se realizaron experimentos para evaluarlo, cuyos resultados se presentan en el apartado 4.2.1. Una vez obtenidos los resultados se plantearon algunas alternativas para su mejora. De entre las diferentes mejoras realizadas, en este apartado se explican aquellas que están relacionadas con el vector de características y sus componentes. En los apartados 3.5.2, 4.2.2 y 4.2.3 se hablará del resto de cambios relacionados con diferentes aspectos que tienen como objetivo también la mejora de los resultados.

En primer lugar, se decidió añadir la altura de los hombros al vector. Dado que la altura de los hombros no se puede hallar mediante máximos dada la proximidad de la cabeza, se ha elaborado una estrategia a partir de las franjas de dos centímetros. Como se puede observar en la imagen 3.10, a partir de la octava franja no se producen grandes incrementos en el número de píxeles por franja hasta la zona en la que cambia del cuello a los hombros. Por lo tanto, para encontrar la altura de los hombros se han calculado los incrementos del número de píxeles de cada franja con respecto a la anterior. A continuación, se haya el incremento máximo que es aquel en el que se sitúan los hombros y a partir de la altura de la persona y la posición de las franjas entre las que se produce el incremento, se calcula la altura de los hombros.

También se han añadido datos sobre la forma de la cabeza, como son la medida del eje mayor de la elipse que la conforma y la relación entre el eje mayor y el eje menor de la misma, ya utilizada en el vector usado para la detección. Dado que los hombros también tienen forma elipsoidal, se ha tratado de obtener las mismas características para ellos, sin embargo, debido a las oclusiones por la cabeza y el ruido existente en la imagen, el eje menor no podía medirse en algunas ocasiones y no podía obtenerse la relación entre ambos ejes, como ocurre por ejemplo en la figura 3.15 donde el extremo del eje no es detectado y el programa le asigna un valor por defecto. Por lo tanto, de los hombros únicamente se ha añadido la medida del eje mayor de la elipse asociada. Un ejemplo de la obtención correcta de todas estas medidas puede verse en 3.14.

El vector utilizado finalmente está formado por un total de veintitrés componentes: la altura de la persona, el número de píxeles pertenecientes a la ROI de la persona de las dieciocho franjas escogidas de entre las veinte disponibles de acuerdo al algoritmo de eliminación de ruido, la altura de los hombros, las medidas del eje mayor de la cabeza y de los hombros, y la relación entre el eje mayor y el eje menor de la cabeza.

3.5. Creación de las clases para la re-identificación de personas.

3.5.1. Planteamiento inicial

Para la creación de clases destinadas a la re-identificación el sistema debe estar configurado para operar en el modo de entrenamiento. Cómo configurar el sistema para que funcione en modo Offline y cómo indicar las secuencias de entrenamiento con las que se quiere trabajar se explicará más adelante en

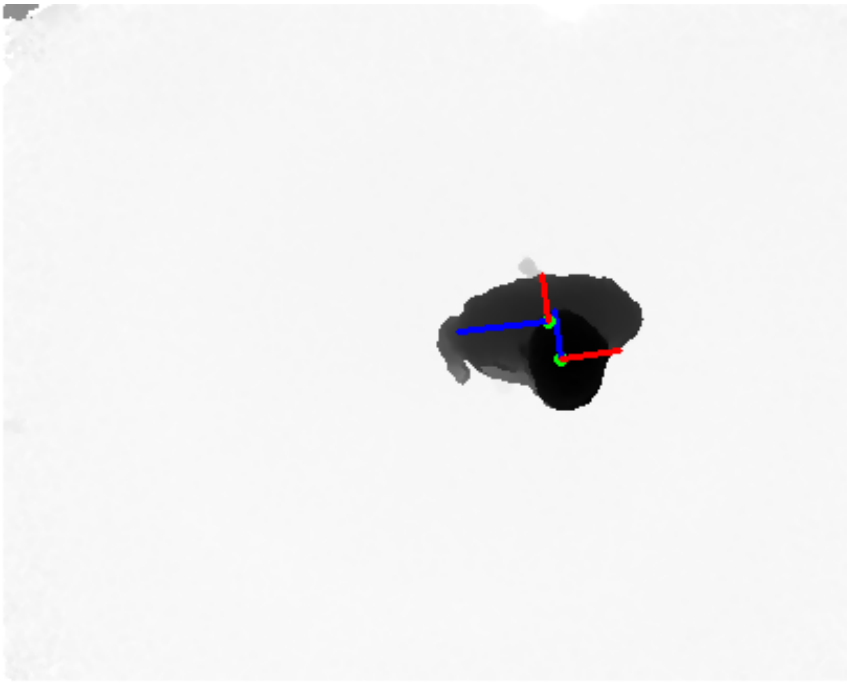


Figura 3.14: Ejemplo de la obtención de las medidas de hombros y cabeza.

Figura 3.15: Ejemplo de la obtención de las medidas de hombros y cabeza.

el anexo A. Por simplicidad, en este apartado el término 'vector de características' se refiere al vector de características extraído para la re-identificación a menos que se especifique lo contrario.

La creación de las clases comienza por abrir la primera secuencia indicada por el usuario, para a continuación extraer y analizar las imágenes de profundidad almacenadas en la secuencia una a una. Para cada imagen, ejecuta el detector de personas explicado en 3.3 y localiza a todas las personas presentes en la escena. Dado que son secuencias de entrenamiento, solo se encontrará una misma persona en la escena en todo momento, por lo que todos los vectores extraídos pertenecerán a dicha persona. De cada persona detectada se extrae el vector de características y se almacena en un archivo, que al final de la secuencia incluirá todos los vectores extraídos de la secuencia correspondiente. Este archivo se almacena en una carpeta especial junto al resto de archivos similares pertenecientes a otras secuencias para poder extraer los vectores cuando sea necesario sin tener que volver a analizar toda la secuencia. Una vez se han analizado todas las imágenes de una secuencia y por lo tanto se han almacenado todos los vectores de características encontrados en ella, se procede a la creación de la clase.

Para la creación de la clase es extraen todos los vectores almacenados en el archivo y se colocan como columnas en una nueva matriz. Esta matriz es equivalente a la matriz T del apartado 2.3. Siguiendo lo explicado en dicho apartado, a partir de la matriz T se obtiene el vector media y la matriz de covarianza y se calculan los autovalores y autovectores que permiten formar la matriz de transformación U. Esta matriz de transformación permite cambiar los vectores del sistema de coordenadas original a un nuevo espacio dimensional creado por mediante PCA. El número de dimensiones del nuevo espacio se ha decidido de forma experimental, evaluando el comportamiento del sistema con las mismas once secuencias de test y variando el número de dimensiones.

Observando la gráfica mostrada en la figura 3.16, se aprecia que los mejores resultados se producen utilizando entre 2 y 7 componentes principales, con tasas de acierto superiores al 58 %.

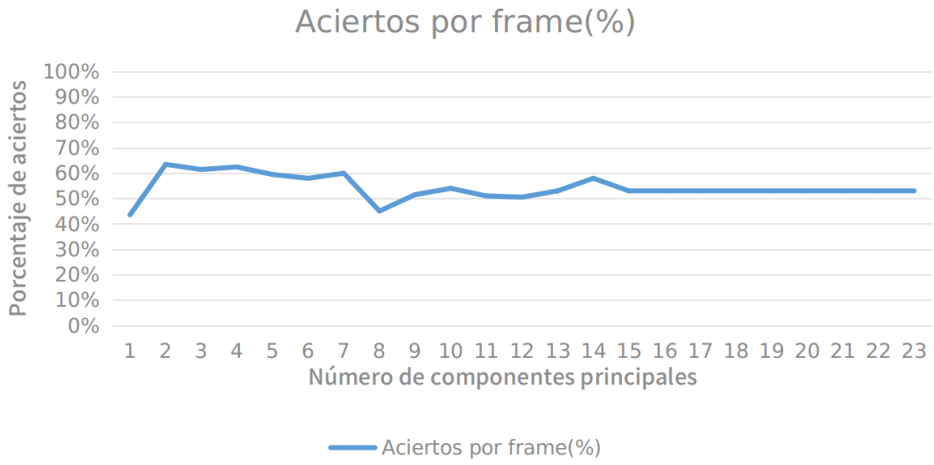


Figura 3.16: Porcentaje de aciertos por *frame* variando el número de componentes principales.

Por otra parte, en la figura 3.17 se muestra el porcentaje de secuencias en las que se produce más de un 50 % de aciertos en el conjunto de sus *frames*. Este dato es importante ya que si se evaluaran todos los *frames* de la secuencia como un conjunto, el sistema sería capaz de re-identificar a la persona correctamente. Las mejores tasas de acierto en esta gráfica se dan para las componentes 2, 3, 4, 9, 10 y 14.

En la figura 3.18 se han situado ambas gráficas en la misma imagen, de forma que la selección del número de componentes sea más sencilla considerando ambos parámetros. A partir de esta gráfica se

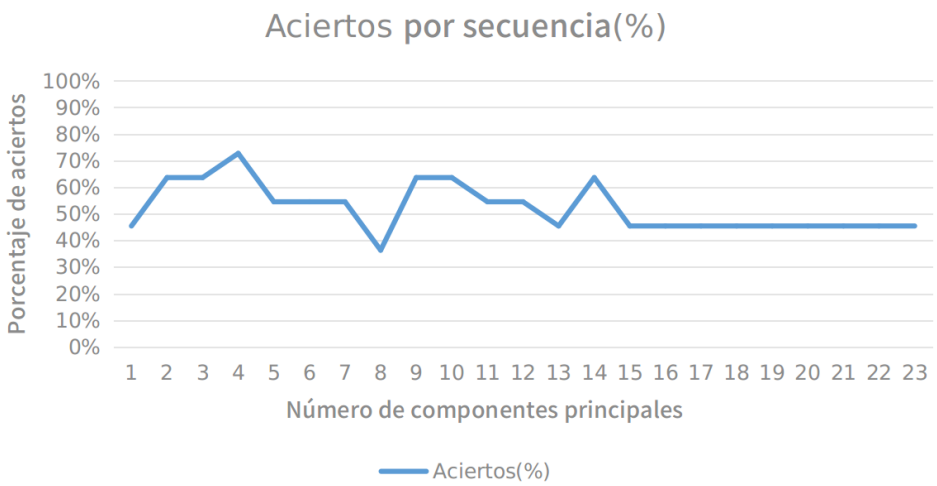


Figura 3.17: Porcentaje de aciertos por secuencia variando el número de componentes principales.

ha decidido que el espacio creado sea de cuatro dimensiones ya que es el número con el que mejores resultados se obtienen considerando todos los datos obtenidos.

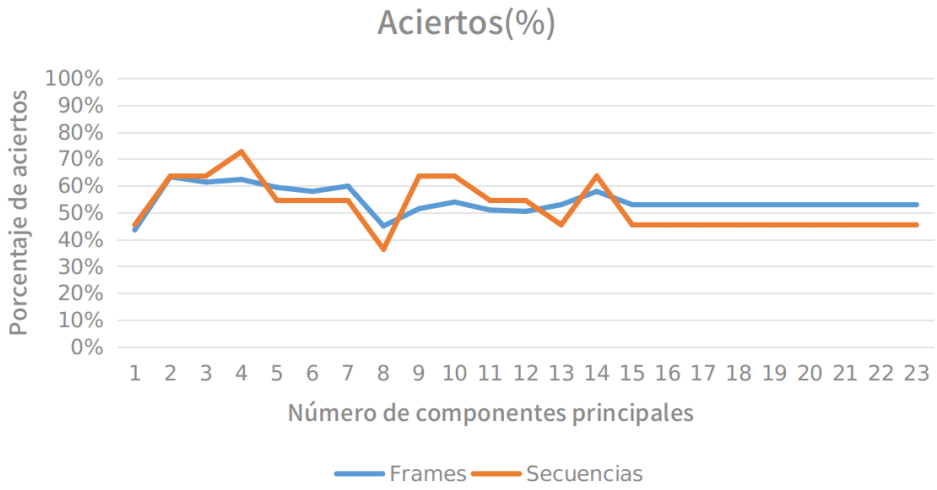


Figura 3.18: Gráfica comparativa de los porcentajes de aciertos por *frame* y por secuencia variando el número de componentes principales.

Para la reducción del espacio dimensional de la matriz en la que se almacenan los vectores de características se utiliza la librería OpenCV que dispone de una clase ya desarrollada para la realización de las operaciones asociadas a la aplicación de PCA. El espacio dimensional por lo tanto queda definido a partir de los atributos de esta clase que pueden ser almacenados en un archivo. Cada clase creada es almacenada en una carpeta para que pueda ser extraída durante la fase de re-identificación en el modo *On-line*.

3.5.2. Mejoras implementadas

Una vez establecido el número de componentes y creadas las clases, se realizaron diversos experimentos y se observó que la mayoría de los errores se producían al comienzo y final de las secuencias, es decir, cuando la persona se encuentra en los extremos de la imagen. Esto es debido a que la persona puede ser detectada aunque no aparezca completamente en la imagen y, por lo tanto, los atributos recogidos en

el vector de características no están completos e introducen ruido al sistema, como se puede ver en la imagen 3.19.

Para evitar este problema, se estableció un área rectangular en la zona central de la imagen que sería la única región donde se realizaría tanto el entrenamiento del sistema como la re-identificación evitando la evaluación de los vectores de las personas situadas fuera de dicha zona, es decir, en los bordes donde las personas no aparecían completamente. Para delimitar la zona se midió cuánto ocupa una persona alta en la imagen (aproximadamente 7 SR en la dirección del eje mayor de los hombros), ya que al estar más cerca del sensor el área de la imagen que ocupa es mayor. Una vez obtenida la medida, el área se definió de forma que entre los extremos de la imagen y el límite del área existiese cómo mínimo una distancia mayor que la mitad del eje mayor de la elipse de los hombros de una persona alta. De esta forma, todos las personas detectadas es seguro que aparecen de forma completa en la imagen

Sin embargo, al tratarse de una zona rectangular debido a la forma de la imagen, la perspectiva variaba en función de cómo la persona recorriera el rectángulo (paralelo al lado mayor o al lado menor). Este problema distorsionaba las medidas de los atributos físicos que capta el sensor introduciendo ruido al sistema. La solución final para la eliminación del ruido fue definir una zona cuadrada en el centro de la imagen dentro de la cual se realizan las acciones relacionadas con la re-identificación (imagen 3.19). Para escoger el tamaño del cuadrado se decidió reducir el lado mayor del rectángulo igualándolo al lado pequeño.

Fuera de este área, se lleva a cabo la detección de personas pero debido a los problemas comentados, no se realiza la re-identificación de las mismas.

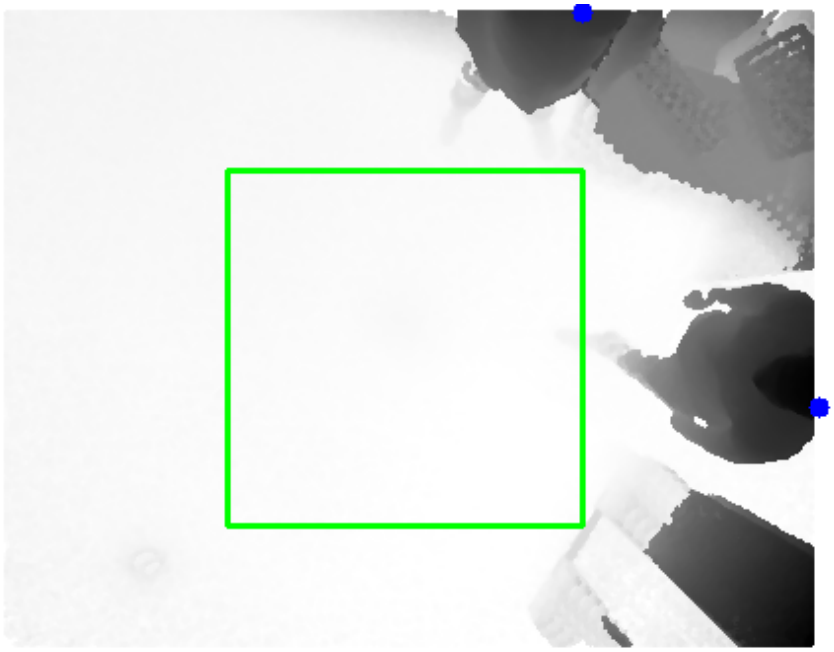


Figura 3.19: Ejemplo de persona detectada sin aparecer completamente en la imagen.

3.6. Re-identificación

Para realizar la re-identificación es necesario trabajar en el modo *On-line* y que existan clases PCA de las personas que se quieran re-identificar (las cuales han sido previamente creadas en la etapa de entrenamiento). Durante esta fase, se utilizan las secuencias de test que haya indicado el usuario según

el método indicado en el anexo A y el clasificador de aprendizaje supervisado basado en PCA explicado en el apartado 2.3.

De acuerdo a lo explicado en 3.5.2, se utilizan solo los vectores de las personas situadas dentro del cuadrado central para la re-identificación, con el fin de evitar el ruido introducido por la perspectiva y los problemas originados cuando las personas no aparecen completas.

La fase de re-identificación comienza extrayendo el vector de características de una de las personas en la escena. A continuación, (3.5) y se proyecta el vector extraído sobre el espacio de cuatro dimensiones de cada clase creada previamente. Después, se recupera dicho vector con la consiguiente pérdida de información y se calcula la distancia euclídea entre el vector de características original y el vector recuperado. Una vez calculados los errores generados por cada proyección sobre un espacio PCA, se escoge el de valor menor siempre que este sea inferior a un determinado umbral. Este umbral se establece con el fin de evitar posibles errores en la re-identificación derivados de detecciones incorrectas o intentos de re-identificación de personas que no se encuentran en el sistema. Por lo tanto, la clase asociada al menor error será aquella cuya identidad se corresponda con el usuario del cual se extrajo el vector de características. Este proceso se realiza con todas las personas que se encuentren dentro del área cuadrada previamente definida en 3.5.2.

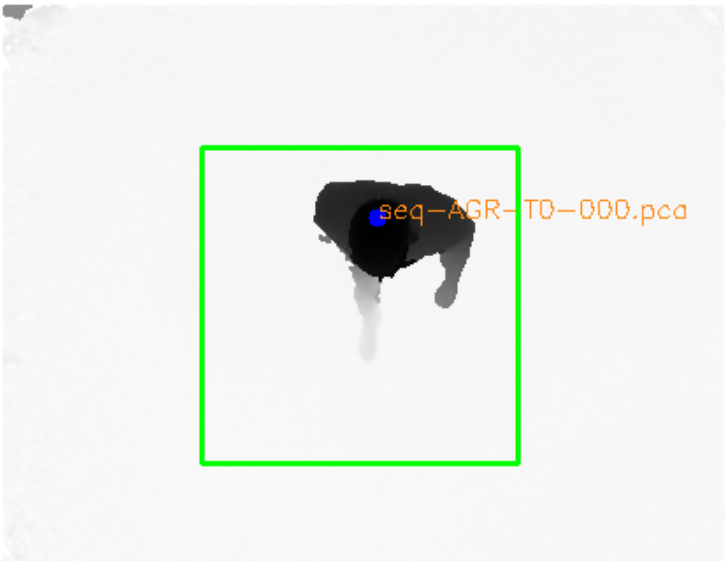


Figura 3.20: Ejemplo de re-identificación con el sensor T1.

3.7. Obtención de métricas

Para la evaluación y el análisis del comportamiento del sistema se extrae información durante la fase de re-identificación realizada sobre imágenes de test. Gracias a que la base de datos GOTPD1 con la que se trabaja fue etiquetada previamente, se puede conocer previamente la identidad de las personas que aparecen en la escena y tras la re-identificación comprobar si los resultados obtenidos han sido correctos y se corresponden con los datos de las etiquetas.

La información recogida por el sistema durante su ejecución es el valor de los errores generados en las proyecciones del vector y las tasas de acierto por secuencia:

- **Errores**: durante la ejecución del bucle que recorre las diferentes clases correspondientes a las personas, cada vez que se calcula una distancia euclídea entre los vectores se almacena su valor en

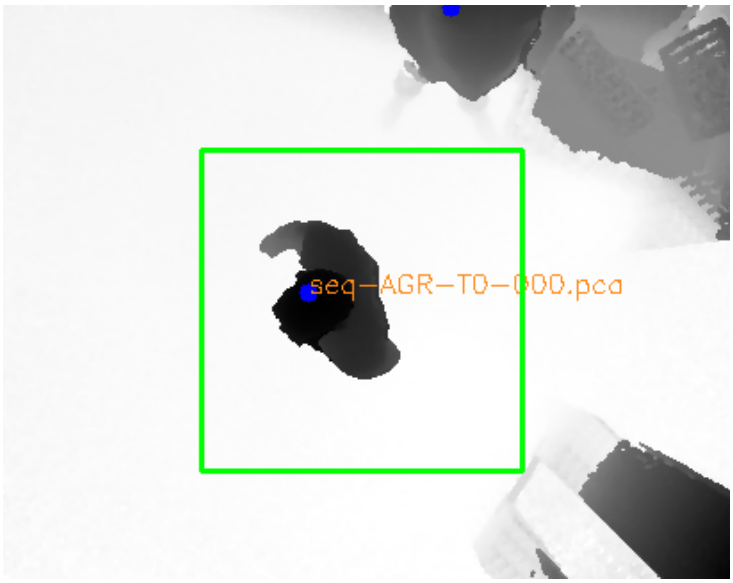


Figura 3.21: Ejemplo de re-identificación con el sensor T0.

un archivo junto con el nombre de la clase a partir de la cual se ha generado (3.22). Este proceso se realiza para cada *frame*, obteniendo el valor de los errores generados al proyectar cada uno de los vectores que aparezcan en la secuencia en cada uno de los espacios correspondientes a una clase PCA. Además, al final del análisis de cada *frame* se incluye el resultado de la re-identificación, es decir, la identidad de la persona asociada a la clase que menor error haya generado.

- **Tasas de acierto**: dado que las imágenes de test se encuentran etiquetadas, se puede obtener el número de veces que el sistema identifica a la persona de forma correcta. Para ello, tras la re-identificación que se realiza en cada *frame* se comprueba si la identidad obtenida se corresponde con la etiquetada en la base de datos. El número de aciertos y errores se guarda para que al finalizar la secuencia se pueda saber el porcentaje de aciertos. Este es almacenado en un archivo junto con los resultados del resto de secuencias que se hayan mandado analizar en esa ejecución del programa.

El cálculo del error de cada proyección sobre el espacio de una clase también se realiza cuando el sistema se encuentra funcionando en tiempo real. Sin embargo, al tratarse de imágenes que están siendo obtenidas por el sensor en ese preciso instante, no se encuentran etiquetadas, por lo que no se pueden extraer tasas de acierto.

Estos datos junto con la variedad de secuencias de la base de datos permiten realizar un análisis exhaustivo del comportamiento del sistema, obtener importantes conclusiones del trabajo realizado y un mejor entendimiento del funcionamiento del sistema, con el fin de poder realizar futuras mejoras al comprender qué está ocurriendo en cada momento.

```
Analizando secuencia seq-ACF-T1-000 :
Clase pca                       Error
Frame 149:
seq-AAR-T0-003.pca              568.029
seq-SPC-T0-002.pca              1205.14
seq-MBR-T0-000.pca              521.899
seq-AGR-T0-000.pca              387.522
seq-ALG-T0-008.pca              1412.31
seq-CAL-T0-007.pca              1551.56
seq-MGG-T0-000.pca              352.993
seq-ACF-T0-000.pca              228.24
seq-DCP-T0-000.pca              412.93
seq-SLS-T0-005.pca              1107.48
seq-JPG-T0-000.pca              402.104
seq-PPC-T0-009.pca              1004.74
seq-JMG-T0-000.pca              367.666
seq-SML-T0-004.pca              1280.73
El maximo analizado se corresponde con: seq-ACF-T0-000.pca
Frame 150:
seq-AAR-T0-003.pca              544.925
seq-SPC-T0-002.pca              1237.95
seq-MBR-T0-000.pca              491.701
seq-AGR-T0-000.pca              385.216
seq-ALG-T0-008.pca              1416.52
seq-CAL-T0-007.pca              1546.05
seq-MGG-T0-000.pca              354.652
seq-ACF-T0-000.pca              253.397
seq-DCP-T0-000.pca              388.581
seq-SLS-T0-005.pca              1102.8
seq-JPG-T0-000.pca              394.719
seq-PPC-T0-009.pca              986.908
seq-JMG-T0-000.pca              361.181
seq-SML-T0-004.pca              1306.4
El maximo analizado se corresponde con: seq-ACF-T0-000.pca
```

Figura 3.22: Ejemplo de las métricas de errores.

```
El porcentaje de acierto en la secuencia seq-ACF-T1-000 ha sido:19/20=95
El porcentaje de acierto en la secuencia seq-AGR-T1-000 ha sido:12/19=63.1579
El porcentaje de acierto en la secuencia seq-DCP-T1-000 ha sido:12/17=70.5882
El porcentaje de acierto en la secuencia seq-JMG-T1-000 ha sido:10/15=66.6667
El porcentaje de acierto en la secuencia seq-JPG-T1-000 ha sido:17/18=94.4444
El porcentaje de acierto en la secuencia seq-MBR-T1-000 ha sido:15/18=83.3333
El porcentaje de acierto en la secuencia seq-MGG-T1-000 ha sido:0/19=0
El porcentaje de acierto en la secuencia seq-SLS-T1-005 ha sido:12/20=60
El porcentaje de acierto en la secuencia seq-SPC-T1-002 ha sido:21/21=100
El porcentaje de acierto en la secuencia seq-AAR-T1-003 ha sido:22/22=100
El porcentaje de acierto en la secuencia seq-SML-T1-004 ha sido:17/19=89.4737
El porcentaje de acierto en la secuencia seq-PPC-T1-009 ha sido:17/17=100
El porcentaje de acierto en la secuencia seq-ALG-T1-008 ha sido:16/16=100
El porcentaje de acierto en la secuencia seq-CAL-T1-007 ha sido:17/19=89.4737
El porcentaje de acierto en la secuencia seq-ACF-T1-000 ha sido:19/20=95
El porcentaje de acierto en la secuencia seq-ACF-T1-000 ha sido:19/20=95
```

Figura 3.23: Ejemplo de las métricas de las tasas de acierto por secuencia.

Capítulo 4

Resultados

4.1. Introducción

Para la validación del sistema implementado en este TFG se ha realizado un evaluación experimental exhaustiva utilizando secuencias de imágenes reales. En este capítulo se describe el entorno en el cual se han realizado los experimentos y la base de datos utilizada para ello. A continuación, se explica la metodología seguida y se presentan los resultados obtenidos en los experimentos.

4.1.1. Entorno experimental

Los experimentos han sido realizados en un entorno inteligente situado en la Universidad de Alcalá, en el cual se encuentran instalados en posición cenital dos sensores Kinect II de acuerdo a lo mostrado en la figura 4.1. Estas cámaras forman la red de sensores a partir de la cual se extraen las secuencias de entrenamiento y las secuencias de test. En este mismo entorno es donde se graba cuando el sistema se encuentra en ejecución a partir de imágenes en tiempo real.

Como se ha comentado, la red de sensores está compuesta por dos sensores Kinect II lo cual permite utilizar uno de ellos para la grabación de secuencias de entrenamiento y el otro para la grabación de secuencias de test, de forma que las imágenes obtenidas pertenecen a dos espacios diferentes. Además, la trayectoria de la persona dentro del campo óptico del sensor varía de uno a otro, lo cual permite evaluar cómo se comporta el sistema en diferentes perspectivas. La trayectoria de las personas se encuentra representada con una línea roja en la figura 4.1.

El campo de visión de ambas cámaras no coincide en ningún punto, lo que permite que las imágenes de entrenamiento y de test sean independientes entre sí. En cuanto a los objetos en la escena, en la cámara T0 aparecen muebles en los extremos de la imagen (figura 4.2a) mientras que en la cámara T1 el campo de visión está despejado (figura 4.2b). Es por ello que las secuencias de entrenamiento se graban con el sensor T0 situado en la entrada de la sala, mientras que las secuencias de test se obtienen con el sensor T1.

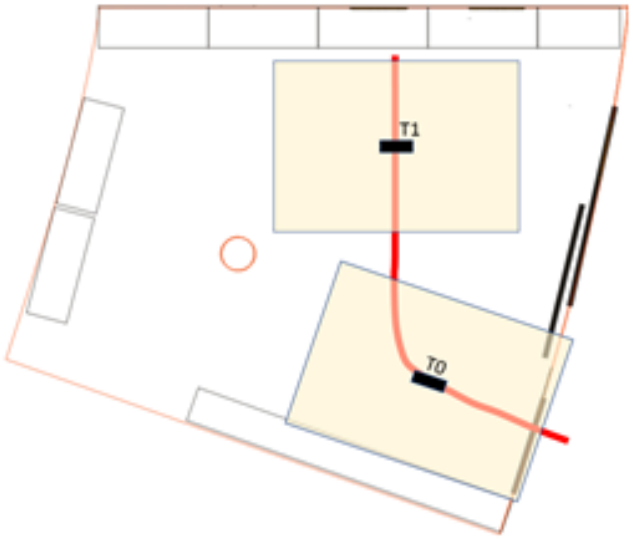
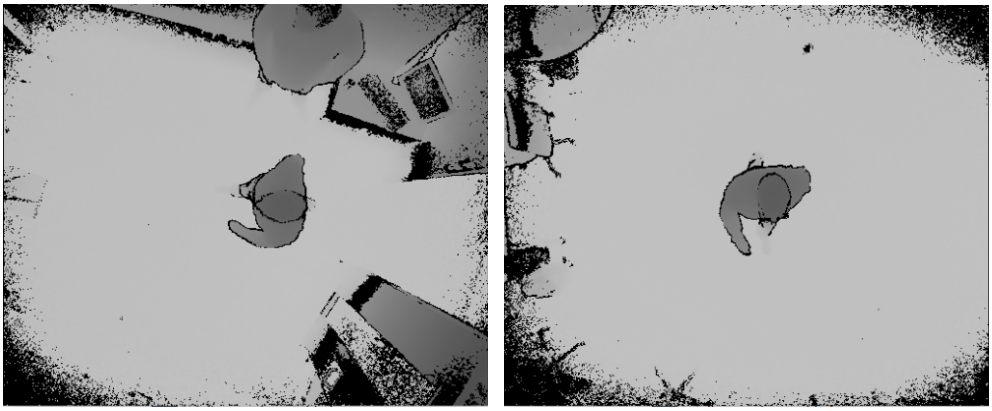


Figura 4.1: Distribución de las cámaras en el espacio inteligente



(a) Ejemplo de imagen obtenida con el sensor T0.

(b) Ejemplo de imagen obtenida con el sensor T1.

Figura 4.2: Ejemplo de imágenes grabadas por la red de sensores.

4.1.2. Base de datos utilizada

La base de datos utilizada ha sido grabada y etiquetada por el Grupo de Ingeniería Electrónica Aplicada a Espacios Inteligentes y Transporte (GEINTRA) [41] de la Universidad de Alcalá. La base de datos en cuestión es GOTPD1 ([35]), la cual ha sido grabada en el entorno descrito en 4.1.1 y etiquetada por los miembros del grupo.

La base de datos recoge varios tipos de secuencias, sin embargo para este trabajo solo se han utilizado secuencias en las que aparece una sola persona. Estas secuencias incluyen a hombres y mujeres con diferentes complementos en la cabeza.

Los archivos de estas secuencias están nombrados de acuerdo a una nomenclatura concreta que hace referencia a la persona que aparece en la escena y el complemento que lleva. Esta nomenclatura sigue la estructura: **seq-XXX-TY-ZZZ**

- **XXX**: es un código que identifica a la persona que aparece en la imagen. Gracias a este código se pueden obtener las métricas de tasas de acierto cuando se ejecutan las secuencias de test. Los códigos utilizados para los experimentos se presentan en la tabla 4.1.

- **Y**: identifica al sensor que adquirió la secuencia, por lo que puede adquirir los valores 0 o 1.

- **ZZZ**: representa el complemento que lleva la persona que aparece en la escena. La lista de complementos que se utilizaron durante la grabación de las secuencias junto con su código correspondiente se encuentra en la tabla 4.2.

Tabla 4.1: Tabla de personas.

No.	Código	Sexo	No.	Código	Sexo
1	AAR	Mujer	2	ACF	Hombre
3	AGR	Hombre	4	ALG	Mujer
5	CAL	Hombre	6	DCP	Hombre
7	DFJ	Hombre	8	FVS	Hombre
9	JMG	Hombre	10	JPG	Hombre
11	MBR	Hombre	12	MGG	Hombre
13	PPC	Mujer	14	RFR	Mujer
15	SLS	Mujer	16	SML	Mujer
17	SPC	Mujer			

Tabla 4.2: Tabla de complementos.

No.	Código	No.	Código	No.	Código
001	Sin sombrero	002	Sombrero pirata	003	Sombrero mafia
004	Gorra roja	005	Sombrero flamenco	006	Sombrero de paja
007	Sombrero del oeste 1	008	Gorra oscura	009	Sombrero del oeste 2

(a) Sombrero pirata. (b) Sombrero mafia. (c) Gorra roja. (d) Sombrero flamenco.

(e) Sombrero de paja. (f) Sombrero del oeste 1. (g) Gorra oscura. (h) Sombrero del oeste 2.

Figura 4.3: Fotos de los complementos utilizados.

En las figuras 4.4 se encuentran ejemplos de imágenes de profundidad obtenidas a partir del sensor T0 mientras que en las figuras 4.5 se encuentran las mismas personas con los mismos complementos pero grabadas con la sensor T1.

(a) AAR llevando el sombrero mafia en el campo de visión del sensor T0.

(b) ACF sin complementos en el campo de visión del sensor T0.

(c) SLS llevando un sombrero flamenco en el campo de visión del sensor T0.

Figura 4.4: Ejemplo de imágenes grabadas por el sensor T0.

(a) AAR llevando el sombrero mafia en el campo de visión del sensor T1.

(b) ACF sin complementos en el campo de visión del sensor T1.

(c) SLS llevando un sombrero flamenco en el campo de visión del sensor T1.

Figura 4.5: Ejemplo de imágenes grabadas por el sensor T1.

4.1.3. Estrategia y metodología de experimentación

La estrategia seguida para realizar los experimentos ha consistido en entrenar el clasificador con las imágenes del sensor T0 y ejecutar el programa para la re-identificación con las imágenes obtenidas del sensor T1 de acuerdo a lo especificado en el capítulo 3. A partir de ello se han obtenido diferentes métricas sobre el funcionamiento del sistema que han permitido conocer mejor su comportamiento en diferentes situaciones. El análisis de estos resultados también ha permitido realizar modificaciones sobre el re-identificador que han mejorado los resultados.

Para medir la efectividad del re-identificador, se han tomado datos acerca de la tasa de acierto y el error asociado a cada clase obtenido tras la re-identificación de cada *frame*. Estos datos se guardan en archivos dentro del ordenador durante la ejecución del programa para su posterior análisis. En el apartado 4.2 se muestran los tasas obtenidas en los experimentos realizados para este trabajo.

4.2. Resultados experimentales

En este apartado se presentan y analizan todos los experimentos realizados a lo largo del tiempo que ha durado este trabajo. Se comienza exponiendo el primer experimento que se realizó para a continuación introducir los resultados con las diversas mejoras que se añadieron al sistema: de esta forma se va poder evaluar cual fue el efecto de la introducción de estas mejoras sobre las tasas de aciertos de las secuencias. Finalmente, se van a exponer los resultados de diversas pruebas que se realizaron con el sistema final para estudiar como se comporta el sistema en diferentes situaciones.

4.2.1. Planteamiento inicial

El primer experimento se realizó con un vector de diecinueve componentes que se correspondían con la altura y las dieciocho franjas de dos centímetros obtenidas aplicando el algoritmo de eliminación de ruido explicado en el apartado 3.4.1. Para este experimento se utilizan todos los máximos identificados como persona por el detector en la imagen al completo.

Dado que se dispone de las imágenes capturadas por dos sensores, se han realizado dos experimentos para evaluar cual debía ser el sensor cuyas imágenes se utilizaran como imágenes de entrenamiento y qué sensor debía ser aquel que obtuviera las de test. En las figuras 4.6 y 4.7 se puede observar las imágenes que se van obteniendo a lo largo de todo el experimento con los resultados de cada fase para cada cámara. En la tabla 4.3 se pueden ver las tasas de acierto de ambos experimentos para cada secuencia.

Tabla 4.3: Tabla de resultados obtenidos con la primera versión del vector empleando la distancia euclídea.

Persona	Tasa de aciertos(%) Entrena:T1 Test:T0	Tasa de aciertos(%) Entrena:T0 Test:T1
AAR	54.3478	43.4783
ACF	65.1163	72.093
AGR	73.1707	65.8537
ALG	82.0513	76.9231
CAL	42.1053	41.6667
DCP	13.5135	25
FVS	73.913	86.9565
JMG	88.5714	100
JPG	39.3939	42.4242
MBR	83.3333	94.4444
MGG	42.3077	11.5385
PPC	60.4651	76.7442
RFR	45.2381	43.9024
Total	58.7329	53.1558

Los resultados obtenidos de los experimentos fueron parecidos en ambos casos sin que una configuración determinada de los sensores sobresaliera por encima de la otra. Además, aproximadamente solo la mitad de las secuencias tenía con una tasa de aciertos mayor o igual al 50 %, por lo que se procedió a introducir mejoras al sistema para incrementar la tasa de acierto y después evaluar qué configuración de las cámaras era mejor.

4.2.2. Mejoras sobre el método de entrenamiento y test

En primer lugar, se cambió la forma de calcular la distancia entre el vector original y vector recuperado tras la proyección. Se decidió calcular la distancia de Mahalanobis en lugar de la distancia euclídea, ya que esta tiene en cuenta la varianza de los datos lo que debería incrementar la tasa de aciertos. Sin embargo, como se puede observar en la tabla 4.4 los resultados no fueron los esperados, ya que se redujo la tasa de aciertos para ambos experimentos.

Tabla 4.4: Tabla de resultados obtenidos con la primera versión del vector utilizando la distancia de Mahalanobis.

Persona	Tasa de aciertos(%) Entrena:T1 Test:T0	Tasa de aciertos(%) Entrena:T0 Test:T1
AAR	12.5	47.8261
ACF	18.75	20
AGR	30	31.5789
ALG	72.4138	22.2222
CAL	46.4286	94.7368
DCP	42.8571	0
JMG	85.1852	80
JPG	0	11.1111
MBR	14.8148	11.1111
MGG	42.1053	15.7895
PPC	96.5517	70.5882
RFR	6.4516	38.8889
Total	39.0048	36.9877

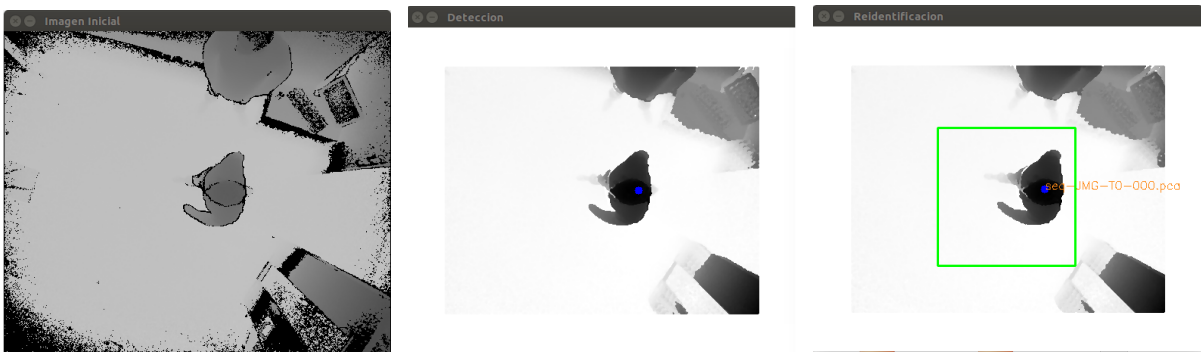
Tras los resultados obtenidos al variar el método de test, se decidió modificar en su lugar el método de entrenamiento. Para mejorar los resultados obtenidos en los dos primeros experimentos (4.2.1) se decidió actuar sobre la región de la imagen en la que se realizaban las acciones relacionadas con la re-identificación.

Para comenzar, se observó que la variación en el número de vectores utilizados en la creación del vector influía sobre el error final que generaba la proyección del vector sobre cada espacio correspondiente a una clase. Por ello se decidió escoger un número de vectores fijo para la creación de la clase, que en este caso debido al número de imágenes de las secuencias se estableció en quince. En el caso de haber más de quince imágenes en las cuales se detectara a la persona se escogerían las quince centrales ya que son en las cuales se encuentra la persona en el centro de la escena y por lo tanto las imágenes tienen menor

distorsión. A continuación, se volvieron a ejecutar los dos experimentos realizados anteriormente para analizar qué configuración de los sensores era el óptima para obtener las imágenes de entrenamiento y de test. Los resultados obtenidos se pueden ver en las tablas 4.5 y 4.6. Como se puede observar, en este caso el entrenamiento con el sensor T0 obtiene mejores resultados que el entrenamiento con el sensor T1. Esto es debido a que en la escena grabada por el sensor T1 hay menos muebles u otros elementos que en la escena del sensor T0, lo cual introduce más ruido al vector de características utilizado para la re-identificación.

Tras esta primera mejora, se observó que la mayoría de errores en la re-identificación se producían al inicio y final de las secuencias por lo que se estudió esta parte de las secuencias para averiguar a qué se debía este fenómeno. Al analizar las secuencias se observó que en los extremos de la imagen la persona era detectada aunque no apareciera completamente en la imagen. Esto suponía un problema para la re-identificación de la persona ya que la mayoría de los atributos físicos medidos diferían en gran medida de los reales. Por ejemplo, si solo aparecía la mitad de la persona, las dieciocho componentes del vector que medían el número de píxeles por franja medirían solo la mitad de los píxeles con respecto a los que medirían si apareciera la persona entera, haciendo que dieciocho de las diecinueve componentes no se correspondieran con los de la persona, haciendo prácticamente imposible la re-identificación. Por ello se decidió reducir la zona de re-identificación, es decir, la re-identificación no se llevaría acabo en los bordes de la imagen de forma que así la persona siempre apareciera completa. Esto supone eliminar las 4 SR de cada lateral de la imagen, formando un rectángulo en la zona central.

Tras realizar experimentos con esta nueva configuración, se observó que al ser una zona rectangular la perspectiva de la imagen hacía que el vector de características variara levemente en función de como recorriera la persona la imagen, paralela a la base del rectángulo o paralela a la altura. Por ello se decidió que la zona de re-identificación pasará a ser cuadrada en lugar de rectangular. Cabe destacar que no es necesario que se encuentre la persona al completo dentro de este zona cuadrada, solo es necesario que se encuentre el máximo ya que la zona se ha establecido de tal forma que con que se detecte el máximo en su interior es seguro que la persona aparecerá al completo en la imagen total. Tras reducir nuevamente la zona se procedió a realizar los experimentos pertinentes cuyos resultados se pueden ver en las tablas 4.5 y 4.6, y un ejemplo de los mismos se puede ver en las figuras 4.6 y 4.7.



(a) Imagen inicial. (b) Imagen tras pasar por el detector. (c) Imagen después de la re-identificación.

Figura 4.6: Tratamiento de la imagen de profundidad obtenida de T0 a lo largo del proceso de re-identificación.

(a) Imagen inicial.

(b) Imagen tras pasar por el detector.

(c) Imagen después de la re-identificación.

Figura 4.7: Tratamiento de la imagen de profundidad obtenida de T1 a lo largo del proceso de re-identificación.

La reducción de la zona de re-identificación ha supuesto que en la mayoría de secuencias se redujera el número de *frames* en los que se realizaba la re-identificación. Esto ha llevado a que en la secuencia de FVS no hubiera ningún *frame* en el que la persona se encontrara dentro de la zona, por lo que se ha eliminado su secuencia de los experimentos. Es por ello que en las tablas 4.5 y 4.6 dicha secuencia no aparezca a diferencia de la tabla 4.3 en la que sí que aparecía.

Tabla 4.5: Tabla comparativa de resultados tras las mejoras entrenando con T1 y re-identificando con T0.

Persona	Inicial (%)	Entrenamiento limitado(%)	ROI reducida(%)
AAR	54.3478	84.375	84.375
ACF	65.1163	31.25	59.375
AGR	73.1707	36.6667	30
ALG	82.0513	68.9655	72.4138
CAL	42.1053	35.7143	33.3333
DCP	13.5135	0	0
JMG	88.5714	100	100
JPG	39.3939	48	52
MBR	83.3333	81.4815	66.6667
MGG	42.3077	2.63158	92.5926
PPC	60.4651	86.2069	79.3103
RFR	45.2381	38.7097	43.3333
Total	57.4679	51.1668	59.45

Tabla 4.6: Tabla comparativa de resultados tras las mejoras entrenando con T0 y re-identificando con T1.

Persona	Inicial (%)	Entrenamiento limitado(%)	ROI reducida(%)
AAR	54.3478	65.2174	86.9565
ACF	65.1163	95	95
AGR	73.1707	63.1579	57.8947
ALG	82.0513	38.8889	50
CAL	42.1053	100	100
DCP	13.5135	5.88235	11.7647
JMG	88.5714	66.6667	93.3333
JPG	39.3939	38.8889	66.6667
MBR	83.3333	66.6667	44.4444
MGG	42.3077	36.8421	52.6316
PPC	60.4651	86.2069	76.4706
RFR	45.2381	77.7778	66.6667
Total	57.8390	60.9549	66.8191

A partir de los resultados obtenidos se ha elaborado una estadística del número de secuencias cuyo porcentaje de aciertos es mayor igual al 50 % para cada uno de los experimentos. Los valores obtenidos se han representado en un gráfico de barras (4.8) para poder observar claramente los resultados de las mejoras realizadas y los resultados con cada configuración de los sensores. En este gráfico se puede observar claramente como, aunque en un primer momento ambas configuraciones de los sensores tenían un comportamiento similar, a medida que se han introducido mejoras los resultados obtenidos al entrenar con el sensor T0 y realizar las pruebas con T1 son mucho mejores con un 20 % más de aciertos que al entrenar con T1 y realizar las pruebas con T0. Por tanto, a partir de ahora los experimentos que siguen se han realizado con la configuración óptima: se entrena con las secuencias de T0 y se realizan los experimentos con las secuencias de T1.

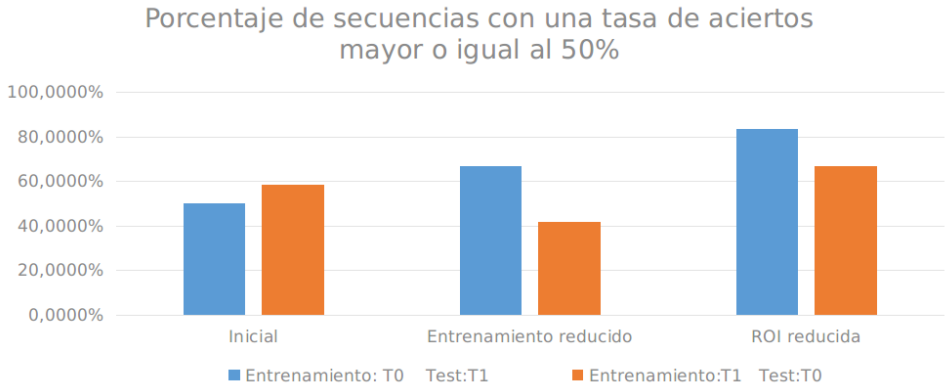


Figura 4.8: Comparación de los resultados para las dos primeras mejoras.

En la gráfica 4.8 también se puede observar cómo cada mejora ha producido un incremento de la tasa de acierto de aproximadamente un 15 %, haciendo que el número de secuencias con una tasa de aciertos mayor o igual al 50 % aumente de la mitad a más del 80 %. Esta evaluación de la secuencia en su conjunto es importante ya que en el caso de introducir un sistema de seguimiento o *tracking*, supone que la tasa de error se reduzca en gran medida siempre que la mayoría de re-identificaciones sean correctas.

4.2.3. Mejoras sobre el vector de características

Tras el incremento de la tasa de aciertos al modificar el método de entrenamiento (figura 4.8), se intentó mejorar aún más los resultados modificando el vector de características utilizado. Esto es, se añadió más información al vector de forma que el clasificador pudiera diferenciar más fácilmente una persona de otra.

Para ello, se estudiaron las componentes del vector de características que se había estado utilizando en los experimentos anteriores. Se observó que, aparte de la altura, no había ninguna otra componente que hiciera referencia a la forma en la que se distribuían los puntos, solo se tenía en cuenta el número de píxeles totales que ocupaba la persona sin importar su disposición. A partir de dicha observación se llevo a cabo una lluvia de ideas para buscar nuevas componentes que añadir al vector y que estuvieran relacionadas con la forma de la persona.

Las primeras ideas que surgieron estaban relacionadas con la forma elipsoidal de hombros y cabeza, por lo que se decidió añadir la relación entre el eje mayor y el eje menor de hombros y cabeza. También se pensó en añadir la distancia del centro de la elipse de la cabeza al centro de la elipse de los hombros, ya que esto mediría cómo de encorvada camina la persona y podría añadir una información decisiva a la re-identificación ya que aunque las dos personas se parecieran en gran medida, es poco probable que caminaran con el mismo encorvamiento. Por último, se decidió añadir la altura a la que se encuentran los hombros de la persona ya que introducía nueva información sobre las medidas de la persona. Se añadieron todos estos componentes al vector anterior formando un vector de características nuevo con un total de veintitrés componentes y se procedió a llevar a cabo los mismos experimentos cuyos resultados se pueden ver en la tabla 4.7. Como se puede observar, la tasa de aciertos se redujo con respecto a los resultados obtenidos tras las mejoras anteriores, por lo que se procedió a estudiar por qué al añadir información los resultados empeoraban.

Tras un análisis profundo de los experimentos realizados se llegó a la conclusión de que la relación de hombros y la distancia entre el centro de la cabeza y el centro de los hombros introducía ruido al sistema. Esto era debido a que la relación de los hombros no era calculada correctamente en algunas ocasiones debido a que el sistema era incapaz de encontrar el eje menor de la elipse debido al ruido de la cámara y a las oclusiones producidas por la cabeza de la persona, como por ejemplo en la figura 4.9. En cuanto a la distancia entre centros, se vio que variaba considerablemente de una cámara a otra, complicando la re-identificación. Es por ello que se decidió eliminar estas dos componentes y añadir otras dos nuevas. Se escogió la medida del eje mayor de la cabeza y de los hombros, ya que estas eran medidas constantes a lo largo de las secuencias y que no daban problemas debido a oclusiones de ningún tipo. A continuación, se realizaron los mismos experimentos para este nuevo vector, obteniendo los resultados mostrados en la tabla 4.8.

Tabla 4.7: Tabla de resultados obtenida con la segunda versión del vector de características.

Persona	Tasa de aciertos(%)
AAR	100
ACF	90
AGR	47.3684
ALG	11.1111
DCP	76.4706
JMG	66.6667
JPG	83.3333
MBR	61.1111
MGG	36.8421
PPC	0
RFR	22.2222
Total	54.1023

Tabla 4.8: Tabla de resultados con la versión final del vector de características.

Persona	Tasa de aciertos(%)
AAR	69.5652
ACF	95
AGR	52.6316
ALG	50
DCP	76.4706
JMG	60
JPG	77.7778
MBR	72.2222
MGG	26.3158
PPC	41.1765
RFR	61.1111
Total	62.0246

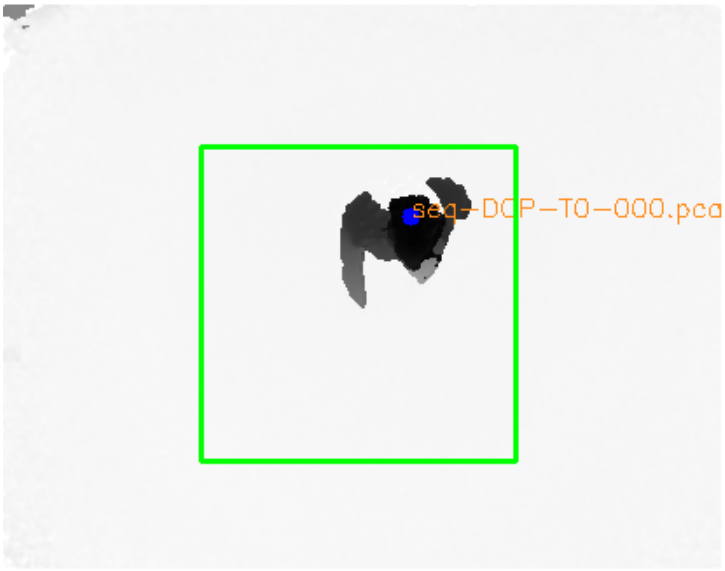


Figura 4.9: Ejemplo de error en la re-identificación debido al ruido introducido por el pelo.

En la figura 4.10 se muestran los resultados obtenidos para cada secuencia con el vector que se utilizó inicialmente y con el vector que se ha utilizado en el último experimento para poder observar cómo ha mejorado el sistema con las mejoras introducidas a lo largo de este apartado. Se puede observar como ha aumentado la robustez del sistema al no presentar resultados tan dispares dependiendo de la secuencia y ha aumentado el número de secuencias con las que se obtienen más de un 50 % de aciertos.

Observando los resultados obtenidos, se puede ver como los peores resultados se producen en las secuencias donde aparecen mujeres. Esto es debido a que el pelo oscuro y largo no es detectado correctamente por el sensor, eliminando parte del cuerpo de la persona y provocando que los atributos extraídos no se correspondan con los atributos reales. Un ejemplo claro de este fenómeno puede observarse en la imagen 4.9, donde la espalda de la persona en la imagen prácticamente ha desparecido.

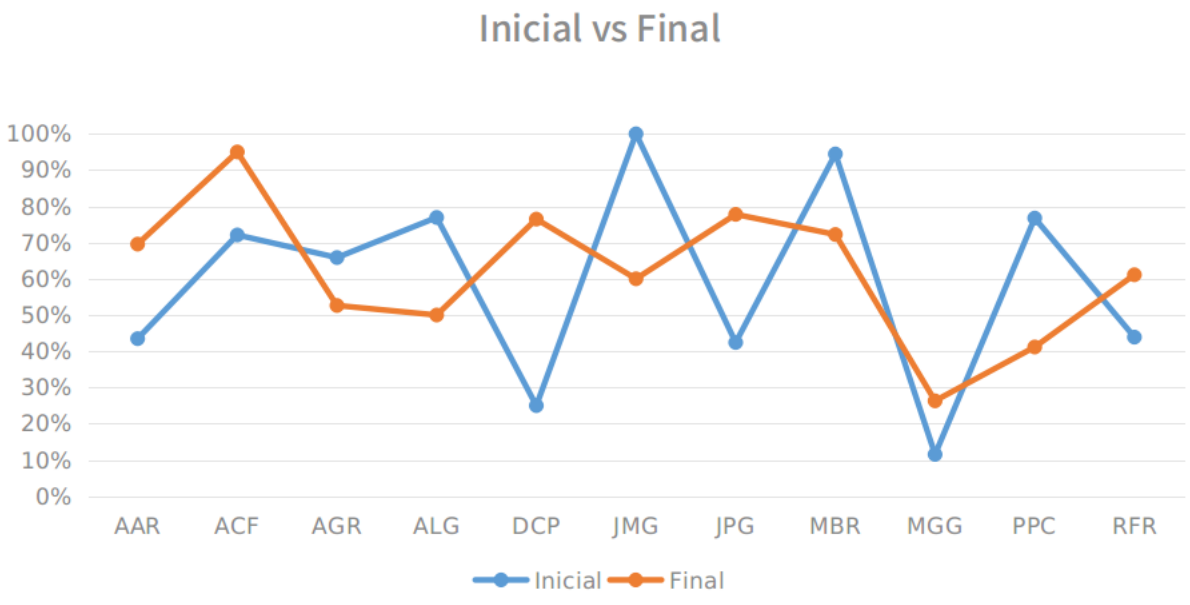


Figura 4.10: Comparación de los resultados con el vector inicial y con el vector final.

Por último, una vez se definió el vector que se iba a utilizar, se llevó a cabo un estudio del número de aciertos de los experimentos en función de las dimensiones de los espacios PCA creados para cada persona. Para ello, se realizó el mismo experimento que anteriormente pero ahora para cada una de las dimensiones posibles, y se extrajeron las tasas de acierto de cada secuencia. Los resultados obtenidos se muestran en la figura 4.11, donde se puede observar una linea azul que representa la tasa de aciertos tomando todo el conjunto de los *frames* independientemente de la secuencia a la que pertenezcan, y una línea naranja que representa el porcentaje de secuencias que tienen más de un 50 % de aciertos. Los mejores resultados en todos los experimentos se dan para valores bajos del número de dimensiones (2, 3 y 4). También se puede observar que a partir de un cierto valor, los porcentajes dejan de variar. Esto es debido a que en estas dimensiones apenas hay información por lo que no aportan nada nuevo para la re-identificación. Observando los resultados obtenidos se ha decidido utilizar espacios de cuatro dimensiones para los experimentos finales expuestos en el apartado 4.2.4.

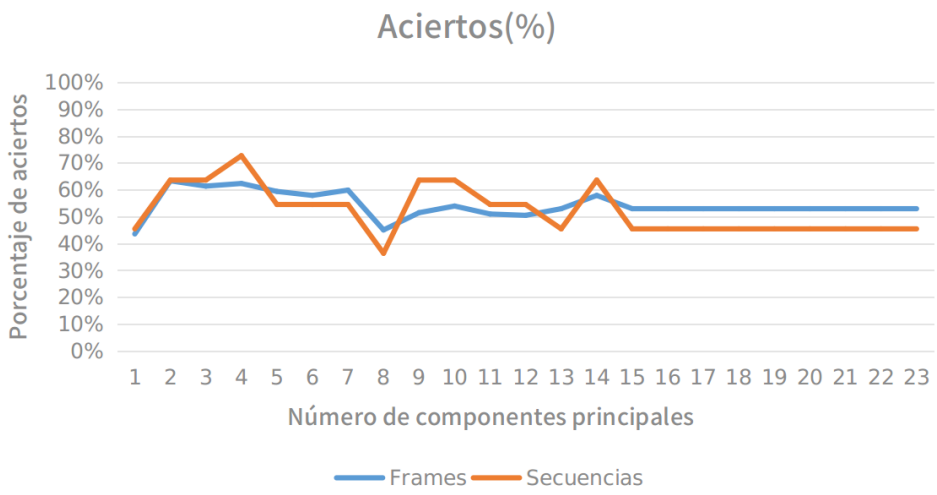


Figura 4.11: Comparación de los resultados variando el número de dimensiones de los espacios PCA.

4.2.4. Experimentos finales

Una vez definido cómo se crea el vector, cuántas componentes tiene y el número de dimensiones del espacio obtenido mediante PCA, se procede a realizar diferentes experimentos para la evaluación final del sistema. En este punto, se obtuvieron secuencias de nuevas personas, formando un total de catorce conjuntos de secuencias donde cada conjunto se corresponde con una persona.

El primer experimento realizado consistió en crear las clases correspondientes a las personas sin sombrero a partir del sensor T0 y lanzar la re-identificación con las imágenes de esas mismas personas sin complementos grabadas por el sensor T1. Los resultados obtenidos se pueden ver en la tabla 4.9, donde se pueden observar los fenómenos comentados en 4.2.3 con respecto al pelo. Un claro ejemplo es la secuencia correspondiente a SLS (4.9) donde la persona tiene pelo oscuro y es morena, lo que provoca que los vectores generados presenten mucho error y la re-identificación no se realice correctamente.

Tabla 4.9: Tabla de resultados de la re-identificación de catorce personas sin complementos.

Persona	Tasa de aciertos(%)
AAR	69.5652
ACF	95
AGR	52.6316
ALG	50
DCP	76.4706
JMG	60
JPG	77.7778
MBR	72.2222
MGG	26.3158
PPC	41.1765
RFR	61.1111
SLS	0
SML	38.0952
SPC	60.8696
Total	55.8025

Los errores originados por el pelo oscuro son debidos al sensor y no al algoritmo desarrollado para la re-identificación, por lo que en trabajos futuros (capítulo 5.2) se propone solucionar este tipo de problemas cambiando el sensor. Mientras, para obtener una idea aproximada de como se comportará el sistema una vez solucionado este problema, se ha realizado un experimento en el cual se han eliminado las secuencias de personas con el pelo largo y oscuro (tabla 4.10). Se puede observar como en todas las secuencias excepto las de MGG se tiene una tasa de acierto superior al 50 %, lo cual implica que la inclusión de un sistema de *tracking* mejorará considerablemente las tasas de acierto como se comentará en 5.2. Hay que destacar que este experimento no representa con exactitud como se comportará el sistema utilizando otro sensor con un mayor número de secuencias y con secuencias en condiciones no óptimas. Este experimento solo es para observar el comportamiento del algoritmo ante imágenes de profundidad sin errores debidos al pelo.

Tras los resultados obtenidos en el experimento anterior se decidió incluir personas con diferentes complementos para ver si el sistema era capaz de distinguir a las personas en dichos casos. Como se puede observar en la tabla 4.11, la re-identificación de personas cuando aparecen personas con y sin complementos se realiza de forma correcta y tiene resultados especialmente buenos con los complementos. También se puede observar que la inclusión de las personas con sombrero no ha supuesto ninguna modificación en los resultados de las re-identificaciones de personas sin complementos, de lo cual se deduce que el sistema es capaz de distinguir de forma muy precisa a las personas que llevan complementos de las que no.

Tabla 4.11: Tabla de resultados de la re-identificación de personas de pelo corto sin complementos y personas con complementos.

Persona	Tasa de aciertos(%)	Complemento
ACF	95	0
AGR	52.6316	0
DCP	76.4706	0
JMG	60	0
JPG	77.7778	0
MBR	72.2222	0
MGG	26.3158	0
SLS	60	5
SPC	95.2381	2
AAR	100	3
SML	89.4737	4
PPC	100	9
ALG	100	8
CAL	89.4737	7
Total	78,1860	-

Tabla 4.10: Tabla de resultados de la re-identificación de solo personas de pelo corto sin complementos.

Persona	Tasa de aciertos(%)
ACF	95
AGR	52.6316
DCP	76.4706
JMG	60
JPG	77.7778
MBR	72.2222
MGG	26.3158
Total	65.7740

Capítulo 5

Conclusiones y líneas futuras

En este apartado se exponen las principales conclusiones obtenidas tras la realización de este trabajo y se proponen ideas para futuras líneas de investigación que continúen con el trabajo realizado.

5.1. Conclusiones

En este TFG se ha llevado a cabo el desarrollo e implementación de un sistema de re-identificación de personas a partir de imágenes de profundidad obtenidas por una red de dos sensores de tiempo de vuelo colocados en posición cenital. La utilización de imágenes de profundidad se debe a que permiten extraer atributos físicos de la persona sin llegar a aportar suficiente información cómo para conocer su identidad, lo cual permite preservar la privacidad de las personas que aparecen en la escena y cumplir con la ley de protección de datos.

Las imágenes de profundidad utilizadas pertenecen a la base de datos GOTPD1 del grupo GEINTRA. Estas secuencias tienen una amplia variedad de personas y se encuentran etiquetadas, lo cual permite obtener estadísticas del funcionamiento del sistema y evaluarlo.

El sistema desarrollado se compone de tres módulos principales: el algoritmo de creacción de clases de cada persona, el detector desarrollado en [5] y [6] y el algoritmo para la re-identificación. Todos estos módulos se basan en el Análisis de Componentes Principales (PCA).

Para la evaluación del sistema se han llevado a cabo varios experimentos que han permitido, tras un profundo análisis de los resultados obtenidos, mejorar las tasas de acierto de las secuencias. A partir de los resultados se ha podido observar cómo afecta el ruido a la re-identificación: las zonas en las que se concentran los píxeles no válidos, como son los extremos de la imagen o el pelo oscuro, son las zonas en las que más errores se cometen durante la re-identificación. Para resolver estos problemas se han implementado diversas medidas como por ejemplo la reducción de la zona de re-identificación o el aumento de información en el vector de características, lo cual ha supuesto un incremento de las tasas de aciertos de un 20 %, llegando a más de un 80 % de secuencias en las que se re-identifica correctamente en la mayoría de *frames*.

Los resultados de la evaluación han permitido validar el sistema implementado, por lo que se considera que se han alcanzado con éxito los objetivos de este trabajo.

5.2. Líneas futuras

Tras la finalización de este trabajo se plantean diversas líneas de investigación que tienen por objetivo continuar con el desarrollo del sistema mejorando los resultados obtenidos:

- **Implementación de un algoritmo de seguimiento.** La inclusión de un algoritmo de seguimiento permitiría aumentar las tasas de acierto al mantener información de imágenes anteriores.
- **Utilización de otros clasificadores.** PCA ha sido escogido por sus óptimos resultados en aplicaciones de tiempo real. Sin embargo, otros tipos de clasificadores como por ejemplo los basados en redes neuronales pueden incrementar el número de aciertos pero a cambio de reducir la tasa de *frames* procesados por segundo al conllevar más carga computacional.
- **Introducción de nuevas características en el vector.** Añadir nuevos datos al vector de características permite al clasificador tener más información con la que realizar la re-identificación.
- **Creación de clases a partir de secuencias capturadas en tiempo real.** Actualmente la creación de clases se lleva a cabo a partir de secuencias previamente grabadas durante una etapa anterior a la re-identificación. Dado que las imágenes de profundidad obtenidas en tiempo real son equivalentes a las que forman las secuencias, sería posible crear las clases a partir de las imágenes obtenidas por el sensor en tiempo real. De esta forma, se incrementaría la base de datos de personas cada vez que una nueva persona apareciera en la escena de uno de los sensores mientras se está ejecutando la aplicación.
- **Utilización de otro sensor de tiempo de vuelo.** El sensor Kinect II destaca por su relación calidad-precio. Sin embargo, existen sensores en el mercado cuyo coste es mucho mayor pero las imágenes obtenidas son de mayor calidad, lo cual podría suponer una mejora del sistema considerable debido a la reducción del número de píxeles no válidos en la imagen. Además, el sensor Kinect II no posee de un mecanismo que permita aumentar la potencia de emisión de la luz infrarroja. Un incremento de la potencia supondría que las superficies oscuras fueran detectadas, lo que solucionaría los problemas surgidos con el pelo o las personas morenas.

Bibliografía

[1] L. Li, "Time-of-flight camera–an introduction," *Technical white paper*, no. SLOA190B, 2014.

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

[3] D. Piatti and F. Rinaudo, "Sr-4000 and camcube3. 0 time of flight (tof) cameras: Tests and comparison," *Remote Sensing*, vol. 4, no. 4, pp. 1069–1089, 2012.

[4] J. Sell and O. Patrick, "The xbox one system on a chip and kinect sensor," *IEEE Micro*, no. 1, pp. 1–1, 2014.

[5] D. Fuentes Jiménez *et al.*, "Diseño, implementación y evaluación de un sistema de conteo de personas basado en cámaras de tiempo de vuelo," 2016.

[6] R. García Jiménez *et al.*, "Detección y conteo de personas, a partir de mapas de profundidad cenitales capturados con cámaras tof," 2015.

[7] L. Zheng, Y. Yang, and A. G. Hauptmann, "Person re-identification: Past, present and future," *arXiv preprint arXiv:1610.02984*, 2016.

[8] A. Bedagkar-Gala and S. K. Shah, "A survey of approaches and trends in person re-identification," *Image and Vision Computing*, vol. 32, no. 4, pp. 270–286, 2014.

[9] R. Layne, T. M. Hospedales, and S. Gong, "Attributes-based re-identification," in *Person Re-Identification*. Springer, 2014, pp. 93–117.

[10] 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.

[11] C. Stahlschmidt, A. Gavriilidis, J. Velten, and A. Kummert, "Applications for a people detection and tracking algorithm using a time-of-flight camera," *Multimedia Tools and Applications*, vol. 75, no. 17, pp. 10 769–10 786, 2016.

[12] 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, 2014.

[13] R. Lan, Y. Zhou, Y. Y. Tang, and C. P. Chen, "Person reidentification using quaternionic local binary pattern," in *Multimedia and expo (ICME), 2014 IEEE international conference on*. IEEE, 2014, pp. 1–6.

[14] L. An, M. Kafai, S. Yang, and B. Bhanu, "Person reidentification with reference descriptor," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 4, pp. 776–787, 2016.

[15] Y.-G. Lee, S.-C. Chen, J.-N. Hwang, and Y.-P. Hung, "An ensemble of invariant features for person reidentification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 3, pp. 470–483, 2017.

[16] F. Wang, W. Zuo, L. Lin, D. Zhang, and L. Zhang, "Joint learning of single-image and cross-image representations for person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1288–1296.

[17] J. Han, L. Shao, D. Xu, and J. Shotton, "Enhanced computer vision with microsoft kinect sensor: A review," *IEEE transactions on cybernetics*, vol. 43, no. 5, pp. 1318–1334, 2013.

[18] M. Munaro, A. Basso, A. Fossati, L. Van Gool, and E. Menegatti, "3d reconstruction of freely moving persons for re-identification with a depth sensor," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 4512–4519.

[19] A. Albiol, J. Oliver, and J. Mossi, "Who is who at different cameras: people re-identification using depth cameras," *IET computer vision*, vol. 6, no. 5, pp. 378–387, 2012.

[20] I. B. Barbosa, M. Cristani, A. Del Bue, L. Bazzani, and V. Murino, "Re-identification with rgb-d sensors," in *European Conference on Computer Vision*. Springer, 2012, pp. 433–442.

[21] Z. Imani and H. Soltanizadeh, "Person reidentification using local pattern descriptors and anthropometric measures from videos of kinect sensor," *IEEE Sensors Journal*, vol. 16, no. 16, pp. 6227–6238, 2016.

[22] F. Pala, R. Satta, G. Fumera, and F. Roli, "Multimodal person reidentification using rgb-d cameras," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 4, pp. 788–799, 2016.

[23] S. Fuchs, "Multipath interference compensation in time-of-flight camera images," in *Pattern Recognition (ICPR), 2010 20th International Conference on*. IEEE, 2010, pp. 3583–3586.

[24] D. Jimenez, D. Pizarro, M. Mazo, and S. Palazuelos, "Modelling and correction of multipath interference in time of flight cameras," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 893–900.

[25] S. Lee, B. Kang, J. D. Kim, and C. Y. Kim, "Motion blur-free time-of-flight range sensor," in *Proceedings of the SPIE Electronic Imaging*, vol. 121, 2012.

[26] 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.

[27] A. Sabov and J. Krüger, "Identification and correction of flying pixels in range camera data," in *Proceedings of the 24th Spring Conference on Computer Graphics*. ACM, 2008, pp. 135–142.

[28] 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.

[29] 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: http://www.sciencedirect.com/science/article/pii/S0262885614001450

[30] T. Huynh, R. Min, and J.-L. Dugelay, "An efficient lbp-based descriptor for facial depth images applied to gender recognition using rgb-d face data," in *Asian Conference on Computer Vision*. Springer, 2012, pp. 133–145.

[31] T. Butkiewicz, "Low-cost coastal mapping using kinect v2 time-of-flight cameras," in *Oceans-St. John's, 2014*. IEEE, 2014, pp. 1–9.

[32] P. Fankhauser, M. Bloesch, D. Rodriguez, R. Kaestner, M. Hutter, and R. Y. Siegwart, "Kinect v2 for mobile robot navigation: Evaluation and modeling," in *2015 International Conference on Advanced Robotics (ICAR)*. IEEE, 2015, pp. 388–394.

[33] K. Otte, B. Kayser, S. Mansow-Model, J. Verrel, F. Paul, A. U. Brandt, and T. Schmitz-Hübsch, "Accuracy and reliability of the kinect version 2 for clinical measurement of motor function," *PloS one*, vol. 11, no. 11, p. e0166532, 2016.

[34] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of cognitive neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[35] J. Macias-Guarasa, C. Losada-Gutierrez, D. Fuentes-Jimenez, R. Garcia-Jimenez, C. A. Luna, A. Fernandez-Rincon, and M. Mazo, "Geintra overhead tof people detection (gotpd1) database," 2016.

[36] "Página web de la librería libfreenect2," https://github.com/OpenKinect/libfreenect2 [Último acceso 16/septiembre/2018].

[37] 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.

[38] A. Fernandez-Rincon, D. Fuentes-Jimenez, C. Losada-Gutierrez, M. M. Romera, C. A. Luna, J. M. Guarasa, and M. Mazo, "Robust people detection and tracking from an overhead time-of-flight camera." in *VISIGRAPP (4: VISAPP)*, 2017, pp. 556–564.

[39] 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.

[40] S. Matzner, A. Heredia-Langner, B. Amidan, E. J. Boettcher, D. Lochtefeld, and T. Webb, "Standoff human identification using body shape," in *Technologies for Homeland Security (HST), 2015 IEEE International Symposium on*. IEEE, 2015, pp. 1–6.

[41] "Página web del grupo de investigación GEINTRA," http://www.geintra-uah.org/ [Último acceso 20/diciembre/2017].

[42] "Información sobre gnu/linux en wikipedia," http://es.wikipedia.org/wiki/GNU/Linux [Último acceso 1/noviembre/2013].

[43] L. Lamport, *LaTeX: A Document Preparation System, 2nd edition*. Addison Wesley Professional, 1994.

[44] "Página de la aplicación cvs," http://savannah.nongnu.org/projects/cvs/ [Último acceso 1/noviembre/2013].

[45] "Página de la aplicación gcc," http://savannah.gnu.org/projects/gcc/ [Último acceso 1/noviembre/2013].

[46] "Página de la aplicación make," http://savannah.gnu.org/projects/make/ [Último acceso 1/noviembre/2013].

[47] "Página web de la librería opencv 2.4.9," https://opencv.org/releases.html [Último acceso 16/septiembre/2018].

Apéndice A

Manual de usuario

En este apartado se especifican los requisitos que del sistema que ejecute la aplicación y la estructuración en directorios de los archivos que componen el programa y se explica cómo compilar y ejecutar la aplicación.

A.1. Requisitos previos

La aplicación en la que se centra este documento se ha desarrollado en lenguaje C++ con la herramienta Eclipse Helios Service Release 2 sobre el sistema operativo Ubuntu 16.04.4 LTS. Para la ejecución del programa se recomienda al usuario utilizar las versiones del software indicadas o superiores para evitar problemas de compatibilidad. Además, se requiere tener instaladas en el sistema las librerías indicadas a continuación:

- **Librería OpenCV 2.4.9**: se utiliza para el tratamiento de imágenes y su visualización. También se encarga de la realización de las operaciones relacionadas con PCA, por lo que es una librería imprescindible para el funcionamiento de la aplicación.
- **Librería Libfreenect2**: se encarga de la configuración y manejo del sensor Kinect II. Permite obtener las distintas imágenes que proporciona dicho dispositivo por lo que esta librería es necesaria para la grabación de secuencias y la ejecución de la aplicación en tiempo real.

A.2. Estructura de directorios

En este apartado se explica cómo se han distribuido los archivos entre los diferentes directorios que componen la aplicación. El programa está formado por un total de once carpetas y nueve archivos que se encuentran en la carpeta principal (figura A.1). A continuación se describe la función de los archivos mientras que en los apartados siguientes se analizará el contenido de las carpetas.

- **pelocorto.data, person.data y sombreropca2.data**: son archivos que contienen los vectores de características utilizados para el entrenamiento de las clases homónimas utilizadas en el detector.
- **core, .cproject y .proyect**: son archivos creados por la herramienta Eclipse para la compilación y ejecución del programa.

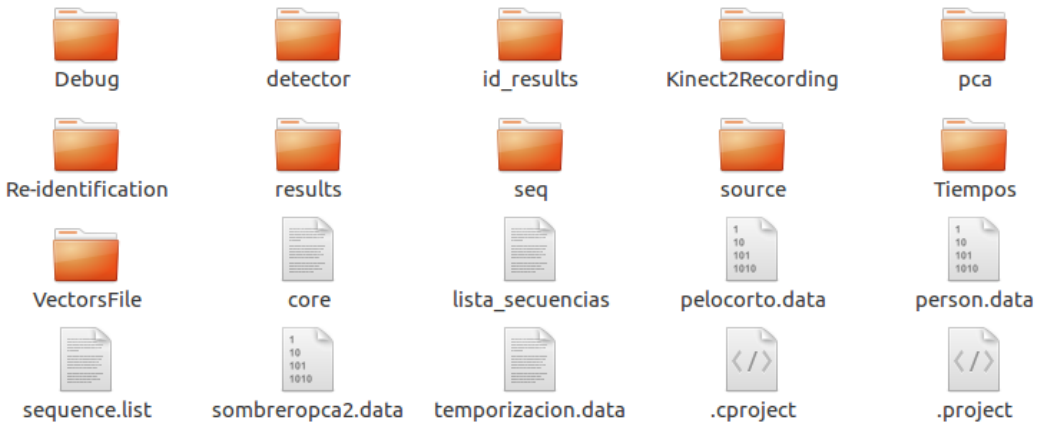


Figura A.1: Archivos y directorios que componen la aplicación desarrollada en este TFG.

- **temporizacion.data**: contiene información sobre los tiempos de ejecución del detector. Cada vez que se procesa un *frame* se añade la información correspondiente a este archivo.
- **sequence.list**: Se trata de un archivo de texto que es leído por la aplicación durante su ejecución y que indica al programa qué secuencias debe utilizar. El usuario escoge las secuencias que se utilizarán escribiendo el nombre de la secuencia dentro del archivo de texto con el formato *seq-XXX-TY-ZZZ* sin la extensión de archivo, ya que se asume que es .z16. Si se quieren utilizar varias secuencias en una sola ejecución, estas deben incluirse en el archivo separadas por un salto de línea.

A.2.1. Detector

En esta carpeta se encuentran todos los archivos fuente necesarios para la ejecución del detector. Estos archivos fueron desarrollados por David Fuentes y Raquel García. En este documento solo se comentan los archivos más importantes ya que en sus respectivos TFG ([5] y [6]) se encuentra la información detallada de todos los archivos.

El archivo más importante es *detector.cpp* ya que alberga la función que se encarga de la detección. Además, se encuentra la definición de una serie de constantes que permiten escoger la información e imágenes que se muestran al usuario. Para la ejecución de la aplicación en tiempo real se recomienda tener desactivadas todas las opciones que se encargan de mostrar información al usuario ya que consumen tiempo de ejecución lo cual puede suponer la pérdida de *frames*.

Dentro de la función de detección se llama a la función de detección de máximos que se encuentra definida en el archivo *maxdetector.cpp*. Esta función también se encarga de la extensión de la ROI y de la extracción de la información que compone el vector de características. En este archivo también existe un conjunto de constantes que controlan la muestra de información del proceso de detección de máximos al usuario.

A.2.2. Id-results

En esta carpeta se almacenan los archivos que contienen las métricas generadas por el programa. Por un lado se encuentra el archivo con el porcentaje de aciertos de todas las secuencias que se han ejecutado, y por el otro los datos sobre la re-identificación en cada secuencia. Estos datos se nombran según la secuencia a partir de la cual se generaron y su extensión es *.reid*. Se puede encontrar información más detallada sobre estos archivos en el apartado 3.7.

A.2.3. Kinect2Recording

La carpeta Kinect2Recording tiene cuatro archivos fuente con extensión *.cpp* y un archivo de cabecera con extensión *.h*. Tres de estos archivos (*OpenCamera.cpp*, *GetFrame.cpp* y *End.cpp*) contienen las funciones del mismo nombre que controlan el dispositivo Kinect II y que ya se explicaron en 3.2.2. El archivo *Init.cpp* no se utiliza para esta aplicación, por lo que no esta incluido en el proceso de compilación, sin embargo se ha decidido incluir ya que puede resultar útil a futuros desarrolladores debido a que muestra cómo utilizar las funciones desarrolladas y puede servir como ejemplo de uso de las mismas. Por último, el archivo *RTKinect.h* contiene la definición de la estructura utilizada para configurar el sensor y las definiciones de las funciones anteriores.

Por otra parte, para la utilización de las funciones desarrolladas primero es necesario crear un conjunto de clases pertenecientes a la librería Libfreenect 2 que permiten interactuar con el sensor Kinect II:

- **Freenect2**: gestiona la conexión al sistema de dispositivos compatibles con la librería.
- **Freenect2Device**: representa a uno de los dispositivos conectados al sistema y permite controlar el flujo de información que se recibe de dicho dispositivo.
- **SyncMultiFrameListener**: se encarga de la recepción de los *frames* enviados por el sensor correspondiente.
- **FrameMap**: permite el almacenamiento de los *frames* recibidos por SyncMultiFrameListener.

En este trabajo, la creación de estas clases se ha llevado a cabo según lo mostrado en A.1. Este método permite al usuario escoger el tipo de imágenes se obtienen del sensor durante la ejecución.

Listado A.1: Ejemplo de creacción de las clases Freenect2, Freenect2Device, SyncMultiFrameListener y FrameMap.

```
libfreenect2::Freenect2 freenect2;
libfreenect2::Freenect2Device *dev=0;
libfreenect2::SyncMultiFrameListener *listener;
libfreenect2::FrameMap frames;
std::map<std::string,cv::Mat> MyFrames;

if (REALTIME==1)
{
        if (options.depth==1)
                tipos|=libfreenect2::Frame::Depth;
        if (options.rgb==1)
                tipos|=libfreenect2::Frame::Color;
        if (options.ir==1)
                tipos|=libfreenect2::Frame::Ir;
        listener = new libfreenect2::SyncMultiFrameListener(tipos);
}
```

A.2.4. Pca

En esta carpeta se almacenan las clases creadas para la re-identificación de acuerdo a lo explicado en el apartado 3.5. En su interior se encuentran dos carpetas:

- **Classes**: contiene los archivos con las clases creadas. Cada clase está nombrada de acuerdo a la secuencia a partir de la cual se creó. Estos archivos tienen la extensión *.pca*.

- **Data**: contiene archivos de texto con información sobre las clases creadas. Esta información consiste en la media, los autovalores, los autovectores y la covarianza de los vectores de entrenamiento. Estos archivos tienen la extensión *.pcadata*.

A.2.5. Re-identification

Contiene los archivos con el código fuente de las funciones que realizan la re-identificación. Los archivos son:

- **CharacVector_Functions.cpp**: contiene las funciones relativas al vector de características. Estas funciones son dos: *getCharacVector*, que extrae el vector de características del máximo que se indique en los argumentos, y *writeCharacVecFromFrameToFile* que escribe el vector en el archivo que almacena los vectores de características de una secuencia.
- **defines_reid.hpp**: contiene todas las constantes relativas a los directorios y aquellas que son necesarias en el proceso de re-identificación. En este archivo no se encuentran las constantes relativas a la información e imágenes mostradas. Solo se modificará en caso de que se quiera cambiar la ruta de alguno de los directorios en los que se almacenan los archivos del programa, la región de re-identificación o el número de *frames* de entrenamiento.
- **reidentification.cpp**: en este archivo se encuentra la función encargada de la re-identificación.
- **StorePCA**: contiene todas las funciones relacionadas con las clases PCA. Dispone de una función para el almacenamiento de una clase en un archivo (*savePca*), una para la extracción de una clase a partir de un archivo (*getPca*) y, finalmente, una última función que crea una clase a partir de un archivo que contenga los vectores de características de entrenamiento (*CreatePcaClassFromFile*).

A.2.6. Results

En esta carpeta se almacena el archivo con los resultados generados por el detector. Los archivos se llaman de la misma forma que la secuencia a partir de la cual se generaron y tiene la extensión *.result*. En este archivo se indica el número de objetos presentes en la imagen, las coordenadas de los centroides, su altura y el resultado de la clasificación, es decir, si son personas o no. El archivo puede ser abierto por cualquier editor de textos y permite el análisis del comportamiento del detector lo que puede servir en un futuro para posibles mejoras del mismo.

A.2.7. Seq

En esta carpeta se deben introducir tanto las secuencias de entrenamiento como las secuencias de test. Para ello, este directorio se encuentra subdividido en dos, uno para las imágenes capturadas por el sensor T0 llamado TestT0 y otro para las imágenes capturadas por el sensor T1 llamado TestT1. Los archivos que se introduzcan en estas carpetas deben tener extensión *.z16* para que el programa pueda reproducirlas correctamente.

Como se comentó en el apartado 3.2.1, el formato *.z16* se llama así debido a que la información de cada píxeles se encuentra representada en formato binario y ocupa 16 bits. La información de profundidad es proporcionada por el sensor con el formato mostrado en la ecuación A.1, donde N_W es la anchura de la imagen y N_H la altura.

$$[Z] = \begin{bmatrix} Z_{0,0} & Z_{1,0} & \dots & Z_{N_W-1,0} \\ Z_{0,1} & Z_{1,1} & \dots & Z_{N_W-1,1} \\ \vdots & \vdots & \ddots & \vdots \\ Z_{0,N_H-1} & Z_{1,N_H-1} & \dots & Z_{N_W-1,N_H-1} \end{bmatrix} \tag{A.1}$$

Para almacenar esta matriz en el archivo, los autores de la base de datos decidieron seguir el orden de las filas de acuerdo a lo mostrado en la ecuación A.2.

$$\begin{bmatrix} Z_{0,0} & Z_{1,0} & \dots & Z_{N_W-1,0} & Z_{0,1} & \dots & Z_{0,N_H-1} & Z_{1,N_H-1} & \dots & Z_{N_W-1,N_H-1} \end{bmatrix} \tag{A.2}$$

Por lo tanto, dado que el sensor Kinect II utilizado en este trabajo tiene una resolución de 424x512 píxeles para la imagen de profundidad, cada fotograma ocupa 424x512x16=3473408 bits. Entonces, cada vez que se desea extraer un nuevo *frame* es necesario leer dicho número de bits del archivo correspondiente a la secuencia deseada. Un ejemplo para leer un fotograma o *frame* a partir de un archivo sería: *seq.read((char *)frame_z16, 512*424*sizeof(unsigned short int))*, donde 'seq' es el nombre dado a una variable tipo *ifstream* en la que se encuentra abierto el archivo deseado. El *frame* se guardaría como un vector fila (frame_z16 en el ejemplo) cuyo contenido sería el correspondiente a la ecuación A.2.

A.2.8. Source

Aquí se encuentra un único archivo con el código fuente de la función principal *main*, el cual contiene las constantes que permiten establecer el modo de funcionamiento del sistema. Estas constantes son:

- **RECORD**: indica al sistema si debe grabar los *frames* que se están analizando en una secuencia, ya sean obtenidos del sensor directamente o pertenecientes a una secuencia ya creada.
- **RecordFile**: establece la ruta en la cual se guardará el archivo que se está grabando.
- **CREAR_CLASES_PCA**: selecciona el modo de operación del sistema: 0 para el modo Online y 1 para el modo Offline.
- **REALTIME**: indica al sistema si debe obtener los *frames* de una secuencia de test (valor 0) o del sensor (valor 1) cuando el sistema se encuentra en el modo Online. En el modo Offline esta constante no tiene efecto.

Además, en este archivo se encuentran los algoritmos que gobiernan el funcionamiento del sistema en modo Online y modo Offline así como las llamadas al detector y la re-identificación.

A.2.9. Tiempos

Este directorio almacena los archivos que contienen la información del tiempo de ejecución de los algoritmos del detector. En total hay cinco archivos cuyos nombres se corresponden con la parte del algoritmo de detección cuya información contienen: *ClasificatorTime.data*, *direccionesTime.data*, *FilterTime.data*, *maxDetectorTime.data* y *vecCaractTime.data*. Esta información también se encuentra en el archivo *temporizacion.data* de la carpeta principal.

A.2.10. VectorsFile

En la carpeta VectorsFile se almacenan los archivos con los vectores de características de las secuencias que se hayan procesado. Los vectores de características se almacenan para poder crear una copia exacta de una determinada clase en caso de que existiera algún error en alguna de ellas. Los archivos están nombrados en función de la secuencia a partir de la cual se crearon, son archivos binarios por lo que no se pueden abrir con un editor de textos y tienen extensión *.data*.

A.2.11. Debug

Esta carpeta es generada automáticamente por el software Eclipse al compilar un proyecto. En ella se encuentra el archivo ejecutable, los archivos generados durante la compilación y el archivo *makefile* que permite compilar el proyecto con una herramienta externa como *cmake* en caso de ser necesario.

A.3. Compilación y ejecución de la aplicación

Dado que la aplicación ha sido desarrollada en Eclipse Helios Service Release 2, se recomienda esta herramienta para la compilación del proyecto. Sin embargo, Eclipse genera los archivos *makefile* que se encuentran dentro de la carpeta Debug, por lo que al ejecutar el comando *make* en dicha carpeta también se realizará la compilación si se tiene un compilador de C/C++ instalado en el sistema aunque no se tenga Eclipse. El ejecutable se genera dentro de la carpeta Debug con el nombre de Reidentification.

Una vez compilado el programa y generado el ejecutable es necesario que existan algunos archivos para que el programa no genere errores al ejecutarlo. En primer lugar, debe existir el archivo de texto que contenga el nombre de las secuencias a analizar. Este archivo se envía como argumento cuando se lanza el ejecutable por lo que puede variar de nombre; en este proyecto se ha nombrado como *sequence.list*. En segundo lugar, en caso de trabajar a partir de una secuencia, es necesario que exista el fichero de profundidad asociado con formato *.z16* en la carpeta correspondiente.

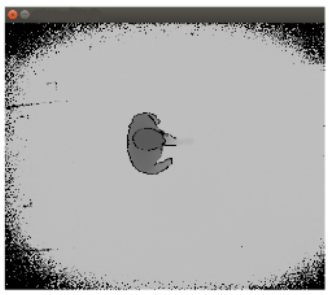
Para iniciar la aplicación tan solo es necesario escribir en consola el comando: ./Reidentification (lista de secuencias). Por ejemplo:

./Reidentification sequence.list

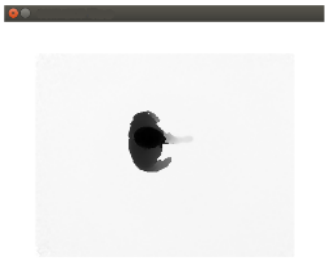
El programa se ejecutará de acuerdo a lo que se le haya indicado mediante las constantes RECORD, CREAR_CLASES_PCA y REALTIME explicadas en A.2.8.

A.4. Resultados de la aplicación

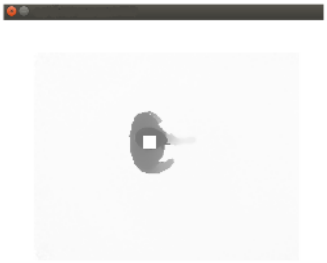
Durante la ejecución del algoritmo el algoritmo permite que se muestren varias ventanas con información el proceso que se esta llevando a cabo. En el caso del detector, las imágenes mostradas (figura A.2) no han sido modificadas con respecto a [5]. Sin embargo, para el proceso de re-identificación se ha añadido una nueva ventana que muestra los resultados del proceso (figuras A.3 y A.4). En esta nueva ventana se puede observar en verde los límites del área dentro de la cual se lleva a cabo la re-identificación, en azul el máximo identificado como persona y en naranja la identidad asignada por el sistema a la persona detectada.



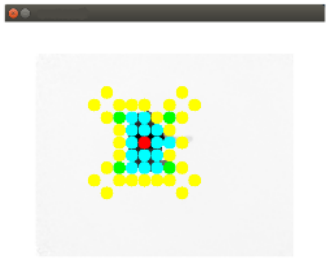
(a) Imagen Inicial



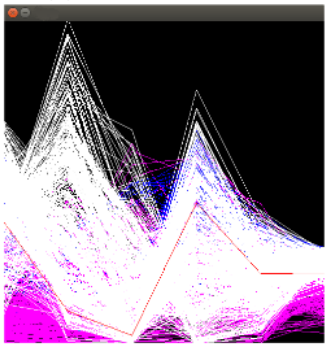
(b) Imagen tras filtrado



(c) Detección de máximos

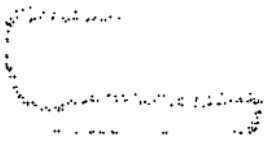


(d) Búsqueda de SR



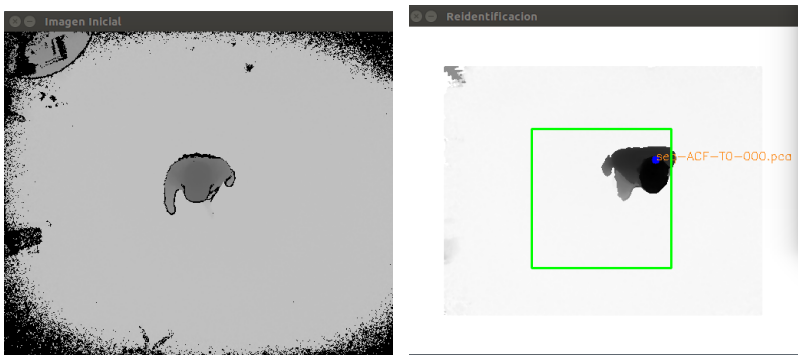
(e) Clases y vectores de características

(f) Imagen resultante tras detección



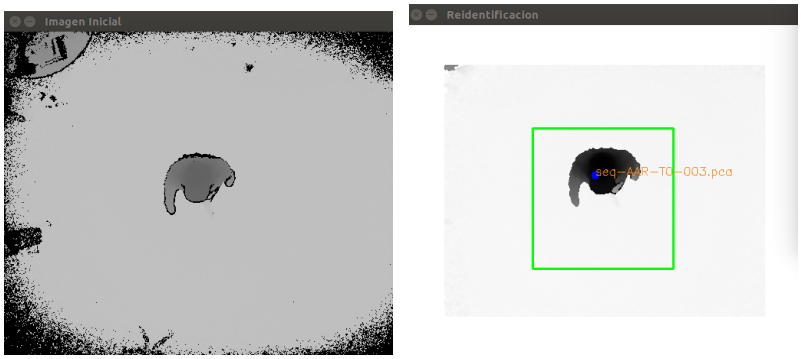
(g) Imagen de seguimiento de las personas en escena

Figura A.2: Ejemplo de imágenes proporcionadas durante el proceso de detección. Imagen extraída de [5].



(a) Imagen inicial. (b) Imagen final.

Figura A.3: Ejemplo de imagen de entrada y salida del sistema.



(a) Imagen inicial. (b) Imagen final.

Figura A.4: Ejemplo de imagen de entrada y salida del sistema.

Además de estas imágenes, si el sistema trabaja a partir de secuencias de test, se proporciona más información sobre el proceso de re-identificación en dos ficheros de texto. En primer lugar, en el fichero *Porcentaje de aciertos* se incluye la tasa de aciertos de todas las secuencias que se hayan ejecutado. Por otra parte, el sistema genera un archivo con el mismo nombre de la secuencia en el cual se incluyen los errores de recuperación obtenidos tras cada proyección del vector de características y la identidad asignada a la persona para cada *frame*. Estos archivos se encuentran en la carpeta *Id-results*. Las figuras A.5 y A.6 muestran ejemplos de los dos archivos mencionados. Estas figuras ya se mostraron en el apartado 3.7 pero se incluyen aquí para facilitar la lectura.

```
El porcentaje de acierto en la secuencia seq-ACF-T1-000 ha sido:19/20=95
El porcentaje de acierto en la secuencia seq-AGR-T1-000 ha sido:12/19=63.1579
El porcentaje de acierto en la secuencia seq-DCP-T1-000 ha sido:12/17=70.5882
El porcentaje de acierto en la secuencia seq-JMG-T1-000 ha sido:10/15=66.6667
El porcentaje de acierto en la secuencia seq-JPG-T1-000 ha sido:17/18=94.4444
El porcentaje de acierto en la secuencia seq-MBR-T1-000 ha sido:15/18=83.3333
El porcentaje de acierto en la secuencia seq-MGG-T1-000 ha sido:0/19=0
El porcentaje de acierto en la secuencia seq-SLS-T1-005 ha sido:12/20=60
El porcentaje de acierto en la secuencia seq-SPC-T1-002 ha sido:21/21=100
El porcentaje de acierto en la secuencia seq-AAR-T1-003 ha sido:22/22=100
El porcentaje de acierto en la secuencia seq-SML-T1-004 ha sido:17/19=89.4737
El porcentaje de acierto en la secuencia seq-PPC-T1-009 ha sido:17/17=100
El porcentaje de acierto en la secuencia seq-ALG-T1-008 ha sido:16/16=100
El porcentaje de acierto en la secuencia seq-CAL-T1-007 ha sido:17/19=89.4737
El porcentaje de acierto en la secuencia seq-ACF-T1-000 ha sido:19/20=95
El porcentaje de acierto en la secuencia seq-ACF-T1-000 ha sido:19/20=95
```

Figura A.5: Ejemplo de las métricas de las tasas de acierto por secuencia.

```
Analizando secuencia seq-ACF-T1-000 :
Clase pca                       Error
Frame 149:
seq-AAR-T0-003.pca              568.029
seq-SPC-T0-002.pca              1205.14
seq-MBR-T0-000.pca              521.899
seq-AGR-T0-000.pca              387.522
seq-ALG-T0-008.pca              1412.31
seq-CAL-T0-007.pca              1551.56
seq-MGG-T0-000.pca              352.993
seq-ACF-T0-000.pca              228.24
seq-DCP-T0-000.pca              412.93
seq-SLS-T0-005.pca              1107.48
seq-JPG-T0-000.pca              402.104
seq-PPC-T0-009.pca              1004.74
seq-JMG-T0-000.pca              367.666
seq-SML-T0-004.pca              1280.73
El maximo analizado se corresponde con: seq-ACF-T0-000.pca
Frame 150:
seq-AAR-T0-003.pca              544.925
seq-SPC-T0-002.pca              1237.95
seq-MBR-T0-000.pca              491.701
seq-AGR-T0-000.pca              385.216
seq-ALG-T0-008.pca              1416.52
seq-CAL-T0-007.pca              1546.05
seq-MGG-T0-000.pca              354.652
seq-ACF-T0-000.pca              253.397
seq-DCP-T0-000.pca              388.581
seq-SLS-T0-005.pca              1102.8
seq-JPG-T0-000.pca              394.719
seq-PPC-T0-009.pca              986.908
seq-JMG-T0-000.pca              361.181
seq-SML-T0-004.pca              1306.4
El maximo analizado se corresponde con: seq-ACF-T0-000.pca
```

Figura A.6: Ejemplo de las métricas de errores.

Apéndice B

Herramientas y recursos

Las herramientas necesarias para la elaboración de este trabajo han sido las siguientes:

- **Herramientas *software*:**
  - Sistema operativo GNU/Linux [42]
  - Procesador de textos LaTeX[43]
  - Control de versiones CVS [44]
  - Compilador C/C++ gcc [45]
  - Gestor de compilaciones make [46]
  - Librería Libfreenect2 [36]
  - Librería OpenCV 2.4.9 [47]
  - Eclipse Helios Service Release 2
- **Herramientas *hardware*:**
  - PC compatible
  - Sensor Kinect II

Apéndice C

Presupuesto

C.1. Costes de equipamiento

C.1.1. Recursos hardware

Concepto	Cantidad	Coste unitario (€)	Subtotal (€)
Ordenador compatible	1	1250	1250
GPU	1	800	800
Sensor ToF Kinect v2	1	177	177
Coste total			2227

C.1.2. Recursos software

Concepto	Cantidad	Coste unitario (€)	Subtotal (€)
Ubuntu 16.04.4 LTS	1	0	0
Librería Opencv 2.4.9	1	0	0
Librería Libfreenect2	1	0	0
Eclipse Helios Service Release 2	1	0	0
Texstudio	1	0	0
Coste total			0

C.2. Costes de mano de obra

Concepto	Cantidad (horas)	Coste unitario (€/h)	Subtotal (€)
Programación y desarrollo del software	240	60	14400
Elaboración de documentación	60	15	900
Coste total			15300

C.3. Costes totales

Concepto	Subtotal (€)
Equipamiento hardware	2227
Equipamiento software	0
Mano de obra	15300
Coste total	17527

El importe total del presupuesto asciende a la cifra de : DIECISIETE MIL QUINIENTOS VEINTISIETE EUROS.

En Alcalá de Henares, a _____ de septiembre de 2018.

Sara Luengo Sánchez.

Universidad de Alcalá
Escuela Politécnica Superior

ESCUELA POLITECNICA
SUPERIOR

Universidad
de Alcalá