UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

RECONOCIMIENTO DE ACTIVIDADES UTILIZANDO INFORMACIÓN DE COLOR Y PROFUNDIDAD.

Borja Olmo Esteban.
Tutor: Juan Carlos San Miguel Avedillo

-PROYECTO FIN DE CARRERA-

Ingeniería de Telecomunicación

Departamento de Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Julio 2016

RECONOCIMIENTO DE ACTIVIDADES UTILIZANDO INFORMACIÓN DE COLOR Y PROFUNDIDAD.

Borja Olmo Esteban
Tutor: Juan Carlos San Miguel Avedillo

Video Processing and Understanding Lab
Departamento de Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Julio 2016

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

Resumen.

El interés de la comunidad científica por dotar a sistemas informáticos de capacidades humanas, ha provocado que durante las últimas dos décadas hayan surgido sistemas de reconocimiento automático a partir de información visual tales como imágenes RGB. La aparación de los sensores de profundidad, y en especial del sensor Kinect desarrollado por Microsoft debido a sus capacidades y su reducido coste, se ha tenido acceso a una nueva fuente de información que no presenta las limitaciones y problemas de las imágenes RGB como los cambios de iluminación.

El objetivo principal de este proyecto ha sido la creación de un sistema configurable que permitiese la integración de las distintas técnicas desarrolladas, tanto para imágenes RGB como de profundidad, y utilizadas para el reconocimiento de actividades, con el objetivo de poder realizar de manera sencilla comparativas entre dichas técnicas y encontrar aquellas configuraciones del sistema que maximicen las capacidades de reconocimiento del sistema en cuestión. Este sistema se ha aplicado a varios datasets del estado del arte, obteniendo resultados competitivos con varias de las configuraciones analizadas y permitiendo discriminar aquellas características útiles para el reconocimiento de acciones.

PALABRAS CLAVE: análisis de video, reconocimiento de acciones, extracción de características, información de profundidad, Momentos de Hu, aprendizaje.

Abstract.

Providing computer systems with human capabilites has been a major field of interest and study since the creation of the first computer systems. Due to this, a large number of studies have emerged in last two decades to provide systems with automatic recognition capabilities exclusively on visual information. With the recent advances in depth sensor hardware and imaging, a large number of depth-based image research has appeared, especially with the release of the low-cost sensor Kinect developed by Microsoft. Such sensors provide a new source of information which does not suffer the same problemas as RGB data such as lighting changes.

The main goal of this project has been to develop a human activity recognition system which allows the integration of different techniques that make use of both RGB and depth data in order to compare them and find out which set of settings provides the best recognition results. The created system has been applied to various state-of-the-art datasets, achieving competitive results with some of the proposed configurations. Moreover, the resuts allow to discriminate which features are useful for activity recognition.

KEYWORDS: video analysis, action recognition, feature extraction, depth information, Hu moments, learning.

Agradecimientos.

En primer lugar me gustaría dar las gracias a mi tutor Juan Carlos, por haberme dado la oportunidad de realizar este proyecto de fin de carrera y por haberme ayudado a mantener la atención y a seguir trabajando en él durante todo este tiempo que he estado inmerso en el mundo laboral.

A todos los componentes del grupo VPULab, daros las gracias no sólo por la ayuda que me habéis prestado en momentos de necesidad, sino también por todos los conocimientos que habéis compartido conmigo, tanto en el laboratorio como en clase, y que me han permitido llegar hasta donde me encuentro hoy.

Quiero agradecer enormemente a mis compañeros y amigos de la carrera por todos los buenos momentos que hemos compartido durante estos últimos años, tanto en épocas de relax como de exámenes, y por los que volvería a realizar la carrera a pesar de todas las dificultades encontradas. Voy a echar de menos esos ratos de relax en la AET con vosotros:)

A todos y cada uno de mis amigos que me han apoyado durante todos estos años, y en especial a mi amigo Álvaro con el que llevo compartiendo pupitre desde primaria. ¡Vamos que ya terminamos!

Especial mención a mis padres y abuelos que siempre me han apoyado y creído en mí, y que me han dado ánimos hasta este momento para terminar de una vez y por todas la carrera. Y a mi hermano Guille agradecerle su apoyo y su paciencia por tragarse mi mal humor en épocas de exámenes, y darle todos los ánimos posibles para superar la ingeniería que ha escogido como carrera universitaria.

A todos, muchísimias gracias.

Borja Olmo.

Julio 2016.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción.

El ser humano, gracias al sentido de la vista y la labor de procesado de datos por parte del cerebro, tiene la capacidad de entender e interpretar los distintos gestos y movimientos del cuerpo, lo que le permite, únicamente a partir de información visual, reconocer las acciones que otras personas dentro de su rango de visión están llevando a cabo.

Esta capacidad del ser humano ha suscitado, especialmente durante las últimas dos décadas, el interés de la comunidad por las tareas de reconocimiento y se ha convertido en un importante tema de estudio dentro del campo de la visión por ordenador. El objetivo, ser capaces de dotar a los sistemas informáticos con las herramientas y técnicas necesarias para analizar los movimientos del cuerpo humano para lograr imitar, en la medida de lo posible, las capacidades de reconocimiento del ser humano.

1.1. Motivación.

El análisis y reconocimiento de actividades en video es un área de interés e investigación desde los comienzos de la visión por ordenador, ya que permitiría la mejora y automatización de los sistemas de videovigilancia actuales, monitorización de personas mayores e incluso control de sistemas informáticos y videojuegos.

Hasta hace unos años se hacía uso exclusivamente de la información aportada por cámaras convencionales (intensidad y color), por lo que los resultados se veían comprometidos en condiciones con frecuentes cambios de iluminación y camuflajes entre la persona y el fondo de escena. El rendimiento de los sistemas existentes era muy bajo.

Recientemente, con la aparición de los sensores de profundidad y en especial del sensor Kinect de Microsoft, que gracias a su bajo precio ha permitido que este tipo

Figura 1.1: Imagen RGB y de profundidad de una pelea [1]

de sensores lleguen a toda la comunidad, se ha tenido acceso a una nueva fuente de información con especial importancia para la realización de tareas de reconocimiento. Estos nuevos datos además de aportar información acerca de la estructura de la escena, presentan una mayor robustez ante los problemas que presentaban los datos capturados mediante cámaras convencionales.

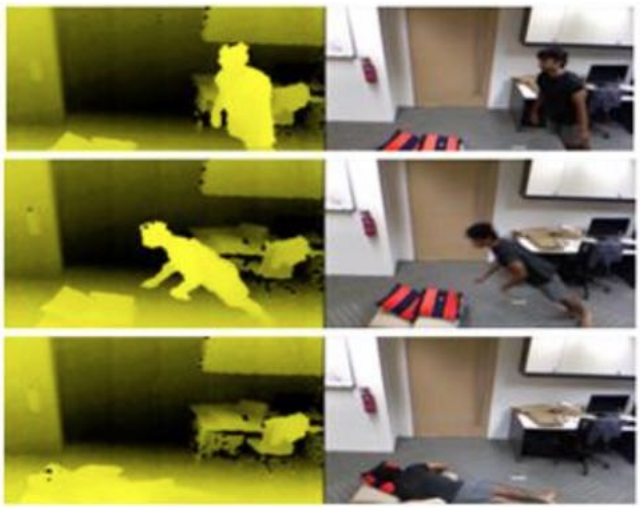

Figura 1.2: Escenificación de la caída de una persona. [2]

Todo esto ha supuesto que en los últimos años se hayan llevado a cabo numerosos estudios y se hayan desarrollado diversas aproximaciones para abordar la tarea del reconocimiento de acciones y actividades haciendo uso de este nuevo tipo de información. De igual manera, se han realizado diversos datasets (bajo diferentes condiciones y situaciones) con el fin de probar el rendimiento de cada uno de los algoritmos desarrollados

La motivación de este proyecto será el desarrollo de un sistema de reconocimiento que permita la integración de diversos algoritmos de reconocimiento que puedan hacer

uso de la información de profundidad con el fin de poder realizar una evaluación de los mismos ante diferentes datasets y diferentes parámetros de configuración.

1.2. Objetivos.

El objetivo principal del proyecto es la elaboración de un sistema de reconocimiento que pueda hacer uso de la información proporcionada por los sensores de profundidad, en especial del sensor Kinect de Microsoft, y que permita la integración de diversos algoritmos (substracción de fondo, selección y extracción de características) y clasificadores de manera sencilla. La configurabilidad de mismo es también uno de los objetivos principales, ya que se busca un sistema que permita ejecutar sobre los datasets diferentes configuraciones del mismo que permitan encontrar aquella que maximice las capacidades de reconocimiento.

1. Estudio del estado del arte y herramientas seleccionadas
2. Diseño e implementación del sistema. En este objetivo se hará especial hincapié en un diseño modular y su documentación
3. Selección de un conjunto de datos sobre los que realizar la evaluación del sistema.
4. Realización de pruebas experimentales y elaboración de conclusiones.

1.3. Estructura de la memoria.

La memoria del proyecto se divide en los siguientes capítulos:

- Capítulo 1. Introducción: introducción, motivación y objetivos del proyecto.
- Capítulo 2. Estado del arte: sistemas de reconocimiento, información de profundidad y sensores existentes.
- Capítulo 3. Diseño: Activity Recognition Chain (ARC) y estructura del sistema propuesto.
- Capítulo 4. Desarrollo.
- Capítulo 5. Experimentos realizados y resultados.
- Capítulo 6. Conclusiones y trabajo futuro.
- Referencias y anexos.

Capítulo 2

Estado del arte.

El reconocimiento de acciones y actividades humanas o HAR (Human Activity Recognition) es necesario en varios sistemas de análisis de video, incluyendo sistemas de seguridad, espacios inteligentes, videoindexado basado en acciones, videojuegos, browsing, clustering y segmentación. Algunos de los trabajos anteriores se centran en reconocer una serie de acciones predefinidas, o entornos restringidos de imágenes. Estos métodos proponen aproximaciones para capturar las características importantes de las acciones mediante modelos paramétricos especializados que por lo general dan lugar al reconocimiento de alta calidad de las acciones estudiadas, pero en su defecto de construcción, generalmente requiere una fase de aprendizaje muy amplia, donde se proporcionan muchos ejemplos de la acción estudiada.

En este capítulo se expone una introducción a los sistemas de reconocimiento, al uso de la información de profundidad (así como de las aproximaciones desarrolladas) proporcionada por los sensores de profundidad, y una explicación de los diferentes sensores existentes.

2.1. Sistemas de reconocimiento.

La estructura seguida por estos sistemas consta, de manera general, de cinco niveles o bloques (como el reflejado en la Figura 2.1). El primer bloque consiste en la entrada de datos al sistema, los cuales pueden provenir directamente del sensor utilizado (de incercia, RGB, profundidad), o de datos previamente capturados mediante el mismo. Estos datos, previo a su análisis, deben pasar por una etapa de preprocesado en la que se normalizan y adaptan para que su escala u otras propiedades no afecten a los resultados finales del sistema. La siguiente etapa consiste en un modelado de la postura y los movimientos realizados, de los cuales se extraeran a continuación

Figura 2.1: Esquema básico de un sistema de reconocimiento

una serie de características que los caracterizarán. Finalmente, estas características se utilizarán para entenar al sistema de reconocimiento, o realizar la clasificación de las mismas junto con el modelo entrenado previamente. En el Capítulo 3 se detalla la estructura y diseño seguidos durante la realización de este PFC.

Teniendo en cuenta que la tarea de reconocimiento ya es en sí misma una tarea compleja, combiene realizar una distinción entre los distintos movimientos del ser humano que se pueden reconocer en función de su complejidad, ya que la complejidad del problema dependerá en gran medida del tipo de moviemiento que se quiera analizar y reconocer. Podemos por tanto clasificar los movimientos y conjuntos de estos en tres categorías [25]: gestos, acciones y actividades.

- Los *gestos* se pueden considerar como los elementos unitarios del movimiento del ser humano. Algunos ejemplos serían girar la cabeza, levantar el brazo, levantar la pierna, etc.

- Las *acciones* son movimientos en los que están involucrados dos o más gestos, los cuales llevan asociado cierto orden temporal. Saludar, andar, correr son algunos ejemplos de acciones humanas.

- Las *actividades* son los movimientos más complejos, ya que consisten en varias acciones llevadas a cabo de manera consecutiva, y por lo general se producen

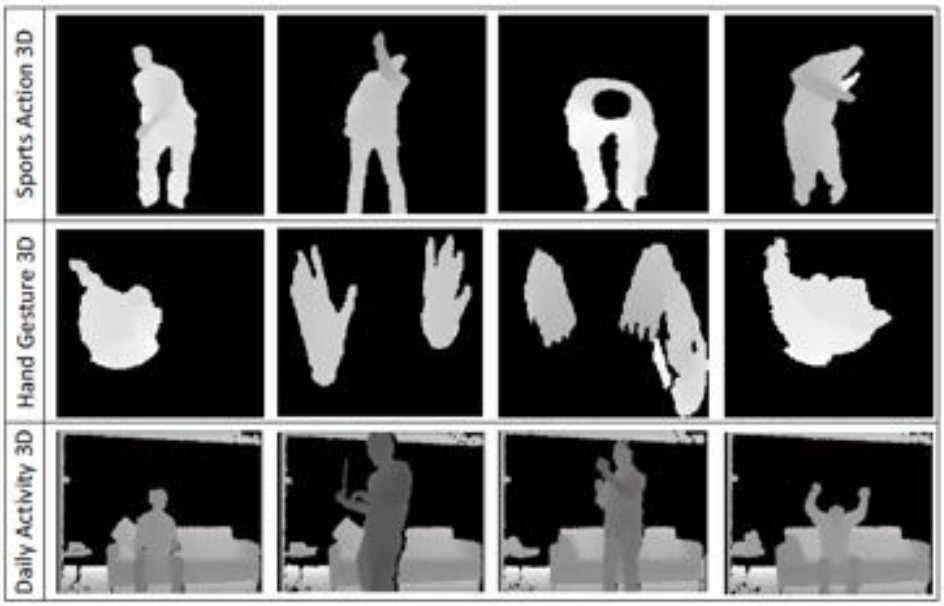

Figura 2.2: Ejemplo de diferencias entre gestos, acciones y actividades [3]

interacciones entre personas y/o personas y objetos. Algunos ejemplos de activiades serían jugar al baloncesto, usar el ordenador, abrir la puerta, etc.

A pesar de poder dividir los movimientos de manera conceptual en estas tres categorías (ver Figura 2.2), la separación entre gestos y acciones no es totalmente pura. En este campo de estudio, las expresiones "reconocimiento de gestos" y "reconocimiento de acciones" son a menudo usadas indistintamente para hacer referencia al mismo tipo de tarea.

2.1.1. Problemas y Retos de los sistemas de reconocimiento de actividades humanas.

A la hora de llevar a cabo el reconocimiento de acciones y actividades humanas, el procedimiento más común es la extracción de caracerísticas que identifiqun un cierto movimiento o secuencia de los mismos, e intentar predecir la etiqueta de la clase de acción asociada. Sin embargo, cada persona puede realizar las acciones de manera distinta e incluso una misma persona puede realizar una misma acción de manera diferente. Existen por tanto una serie de condiciones que van a dificultar la tarea del reconocimiento de acciones. De acuerdo a los trabajos de [26] y [27], algunas de las condiciones que afectan al rendimiento y capacidad del reconocimiento de una acción son:

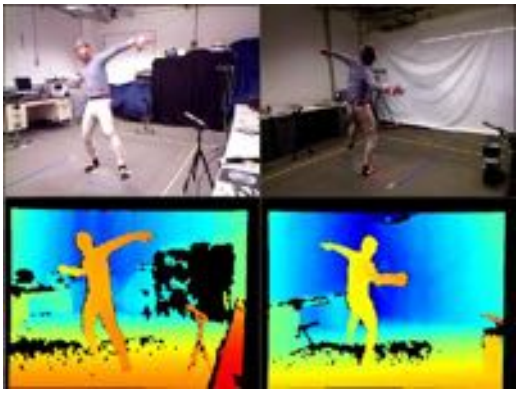

Figura 2.3: Diferencias en una misma actividad debido al punto de vista[4]

- El *punto de vista*, que expresa la relación entre la persona y la cámara. Un misma acción vista desde diferentes puntos de vista puede llevar a resultados completamente distintos, ya que los movimiento observados por la cámara varían (ver ejemplo en Figura 2.3).
- La *tasa de ejecución* se relaciona con la velocidad a la que se lleva a cabo la acción o la tasa de frames. Esta tiene un gran impacto en la información temporal de la acción, afectando especialmente a los algoritmos que hacen uso de esta información.
- *Variabilidad* o "estilo personal". Cada individuo puede realizar la acción de manera ligera o totalmente distinta al resto. Aunque el ser humano es capaz de identificar las acciones a pesar de las variaciones introducidas por cada individuo, no resulta sencillo dotar a los sistemas informáticos de esta misma capacidad.
- Vistas parciales y oclusiones. En entornos reales, en los cuales no siempre se dan las condiciones ideales, las personas pueden ser ocluidas total o parcialmente (ver ejemplo en Figura 2.4) dificultando las tareas de reconocimiento e incluso imposibilitandolo en el peor de los casos.

Además, tanto la cantidad de datos de training como el entorno en el que se realizan las acciones tienen una gran impacto en el rendimiento de estos sistemas. Para conseguir un sistema de reconocimiento robusto se requieren grandes cantidades de datos de training, ya que la escasez de los mismos implica una menor capacidad discriminativa y por tanto un menor rendimiento. En cuanto al entorno, las condiciones de luz y ubicación de la cámara son fuentes que introducen variaciones en los datos capturados,

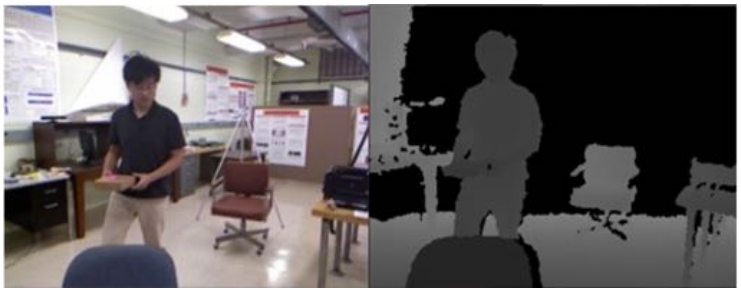

Figura 2.4: Ejemplo de oclusión parcial en el que no se tendría información acerca de los mov. de las piernas [5]

y es por esta razón por la que muchas aproximaciones se restringen a ciertos escenarios en los que las condiciones sean constantes o por lo menos conocidas para poder actuar en consonancia.

Además de la problemática mencionada, Bulling et al.[20] expone en su investigación una serie de retos a los que se enfrentan los sistemas de reconocimiento de actividades humanas.

- Definición y diversidad de las actividades: el primer reto con el que se encuentran los investigadores de este tipo de sistemas, es obtener un alto y claro entendimiento de la definición de las actividades bajo estudio y sus características específicas. Esto puede parecer una tarea trivial, sin embargo, las actividades humanas presentan un alto grado de complejidad y variabilidad, como se mencionaba previamente. Es por tanto necesario realizar un análisis en profundidad de las actividades para averiguar qué información es de especial relevancia para la tarea del reconocimiento, teniendo siempre en mente los requisitos del sistema a diseñar para focalizar todo lo posible la búsqueda.

- Desbalanceo de clases: muchos de los sistemas de reconocimiento de actividades, como por ejemplo aquellos enfocados a la monitorización de la conducta a largo plazo, se enfrentan a este problema. Por lo general el ser humano no realiza todas las actividades con la misma frecuencia, lo que se traduce en que la cantidad de datos asociados a cada clase no sea igual. Esto se puede resolver mediante la captura de datos de entrenamiento adicionales, o aternativamente generando datos de entrenamiento artificiales para extender la clase afectada e igualar la cantidad de datos. Una técnica para esto es el sobremuestreo (por ejemplo mediante duplicación de datos) de la clase con una baja cantidad de datos para igualarla a aquella con la mayor cantidad.

- Anotación Ground Truth: otro de los problemas al que se enfrentan los sistemas

HAR es la anotación de la colección de datos de entrenamiento. Esta tarea es realmente costosa y tediosa, dado que la persona encargada tiene que realizar esta anotación en tiempo real, o realizar la revisión a posteriori de los datos RAW capturados por el sensor y anotar de manera manual todos ellos.

- Colecciones de datos y diseño de experimentos. Otro de los principales retos a los que se enfrentan estos sistemas, es la obtención de una colección de datasets sobre los que realizar su evaluación. El reto surge del hecho de que, a diferencia de lo que ocurre en otros campos de investigación como el de reconocimiento de voz, la comunidad de investigación en el campo del reconocimiento de actividades no se unido bajo el objetivo común de realizar una amplia colección de datasets de proposito general, por lo que los datasets capturados se realizan persiguiendo diferentes objetivos: alta calidad de los datos, un gran número de modalidades o sensores, grabaciones de larga duración, etc. Esto dificulta la tarea de evaluación de los nuevos sistemas desarrollados, y suele dar lugar a la creación de un dataset propio que se ajuste a sus requisitos, contribuyendo a la ya extensa y dispar colección de datasets de reconocimiento de actividades.

Con la aparición de los sensores de profundidad, se han abierto nuevas líneas de investigación en este campo, ya que la información proporcionada por los mismos permite sobrepasar algunos de los límtes mencionados hasta ahora y que exisitían en las aproximaciones que hacen uso de cámaras convencionales.

2.2. Información de profundidad.

Hasta hace unos años, la mayoría de trabajos de investigación realizados en el campo del reconocimiento de acciones por ordenador hacían uso de secuencias de video capturadas mediante cámaras RGB convencionales [28, 29, 30]. Las imágenes de estas secuencias, en las que el valor de cada píxel representa la intensidad con la que la luz reflejada en la escena llega al sensor de la cámara, aportan información especialmente útil para el procesado de imágen, como el color y la textura. Sin embargo, debido a la propia naturaleza de estos dispositivos, los datos capturados son muy sensibles a cambios de iluminación, y realizar la sustracción de fondo de manera robusta es una tarea compleja ya que la información de frente y fondo aparece mezclada. Además la captura de los movimientos del cuerpo humano mediante este tipo de cámaras es especialmente compleja, debido a que estos movimientos se realizan en un espacio tridimensional y se está perdiendo la información relativa a una de las dimensiones, la correspondiente a la profundidad.

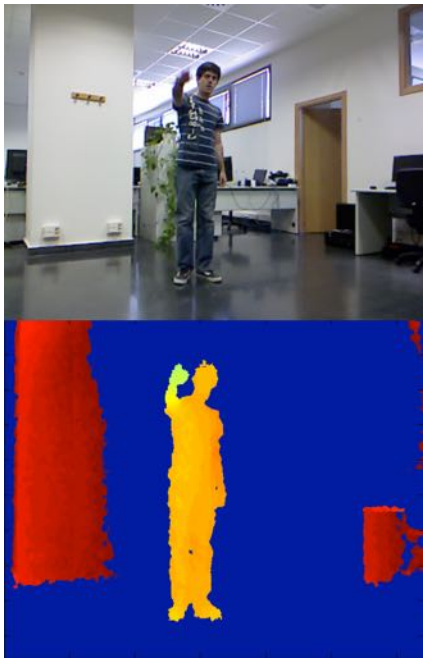

Figura 2.5: Ejemplo de imágen RGB y profundida capturadas mediante el sensor Kinect

Es por esto que con la aparición de los sensores de profundidad, de los que se hablará con más detalle más adelante, han surgido numerosos estudios (ver Tabla 2.2) en los que haciendo uso de la información tridimensional aportada por estos sensores han conseguido sistemas con un ratio de reconocimiento de más del 80 %.

Algunas de las ventajas que aportan los sensores de profundidad respecto a las cámaras convencionales, son:

- Acceso a la información de profundidad con la que capturar los movimientos de manera robusta, permitiendo así un mejor modelado del movimiento humano.
- Menor sensibilidad a cambios de iluminación.
- Reducción de la variabilidad introducida por los colores y texturas de los objetos.
- Facilita tareas como la sustracción de fondo. Al conocer la distancia a la que se encuentran los objetos, resulta más sencillo distinguir aquello que conforma el frente y el fondo.

Las diversas líneas de investigación acerca del reconocimiento de acciones a partir de los datos aportados por este tipo de sensores se pueden agrupar en dos categorías o

aproximaciones: basadas en mapas de profundidad, y basadas en las articulaciones del esqueleto. Estos a su vez se subdividen es dos categorías, en función del uso que se de a la información proporcionada por el sensor para la extracción de las características que servirán como identificador de los movimientos y acciones..

Depth Maps. a diferencia de las imágenes RGB, cuyos píxeles indican la medida de intensidad de luz y color que llega al sensor, los píxeles en un mapa de profundidad codifican la información de distancia de la escena. Esto permite obtener la información acerca de la forma de los diferentes objetos presentes en la escena y la estructura 3D de los mismos. En la Figura 2.5 puede verse la diferencia entre ambos tipos de imágen.

- Depth Map Based Space-Time Features: este tipo de aproximaciones consideran cada secuencia de acción como un volumen 4D, en el que la información espacial (x,y,z) se prolonga durante un tiempo t. La secuencia se puede procesar de manera global, o como un conjunto de características locales. La mayor parte de trabajos realizados en el campo del reoconocimiento de acciones a partir de mapas de profundidad hacen uso de estas características espacio temporales.
  Inspirados por el éxito de los métodos de template matching [31] en secuencias 2D, numerosas aproximaciones han tratado de transformar el problema 3D en 2D, llevando a cabo el reconocimiento sobre planos 2D proyectados. Un ejemplo es el trabajo llevado a cabo por Yang et al. [16], el cuál se describe con mayor detalle en la Sección 2.4.

- Depth Map Based Sequential Features: este tipo de características se extraen modelando de manera explícita la dinámica temporal de las secuencias de profundidad. A partir de los mapas de profundidad, se extraen una serie de puntos que permiten identificar una serie de posturas del cuerpo que, una vez analizadas de manera secuencial, permite el reconocimiento de la acción que se está llevando a cabo.
  A diferencia de las características espacio-tiempo anteriores, a día de hoy se han llevado un menor número de trabajos que hagan uso de este tipo de características. Este se debe a que analizar los movimientos humanos es una tarea compleja debido a la gran variabilidad que introduce cada persona. Sin embargo, cabe destacar los trabajos realizados por Li et al. [17] y Viera et al. [32], los cuales han conseguido unos resultados prometedores.

Skeletal Joints. Los skeletal joints o puntos del esqueleto codifican la posición espacial de las articulaciones del cuepo humano por cada frame en tiempo real (ver Figura 2.6). Dado que los movimientos realizados por el esqueleto aportan toda la información acerca del movimiento que se se está realizando, hacer uso de esta infor-

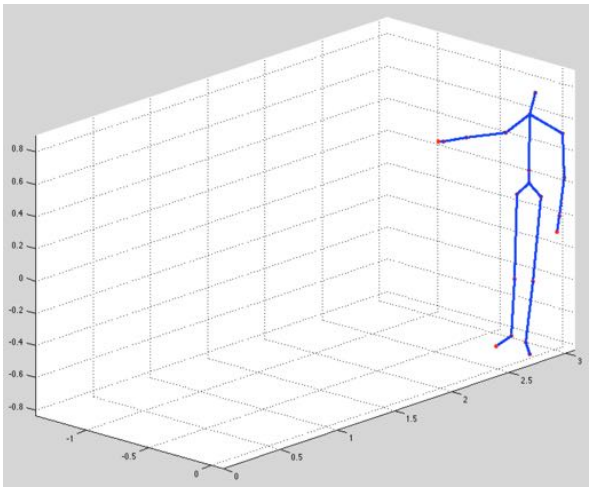

Figura 2.6: Esqueleto obtenido durante la captura de video con el sensor Kinect

mación es de especial interés a la hora de realizar reconocimiento de acciones. En el caso del sensor Kinect, es el propio sensor el que aplica su propio algoritmo sobre la información de profundidad para calcular la posición de cada una de las articulaciones (hasta un total de 20), sin embargo los datos aportados son considerablemente ruidosos en situaciones en las que se producen oclusiones parciales de la persona, por lo que aplicar directamente esta información no produce resultados prometedores. Por ello es necesario la investigación y desarrollo de algoritmos que calculen características basadas en el esqueleto y que sean robustos al ruido y oclusiones.

Una de las ventajas de las características basadas en el esqueleto, es que son computancionalmente menos costosas que las basadas en mapas de profundidad, al trabajar con una menor cantidad de información.

Destacan los trabajos realizados por Yang et al. [33] y Xia et al. [5].

2.2.1. *ToF Sensors.*

Los sensores ToF, cuyas siglas provienen de Time of Flight, se engloban dentro de lo que se conocen como sistemas de visión activa. Este tipo de sistemas basan su funcionamiento en la proyección de luz a la escena, y por lo general son menos sensibles a cambios de iluminación que los sistema de visión pasiva.

Típicamente, la iluminación generada por este tipo de sensores proviene de un laser de estado sólido o un LED operando en el rango cercano a los infrarrojos (˜850nm), invisible al ojo humano. Y es el sensor óptico, diseñado para ser sensible a la luz dentro de este espectro, el que convierte la luz que le llega en corriente eléctrica. Es importante destacar que la luz que llega al sensor está formada por dos componen-

te: una reflejada, y una ambiental. La información de distancia o profundidad está únicamente en la componente reflejada, de manera que la componente ambiental sólo reduce la relación señal a ruido (SNR).

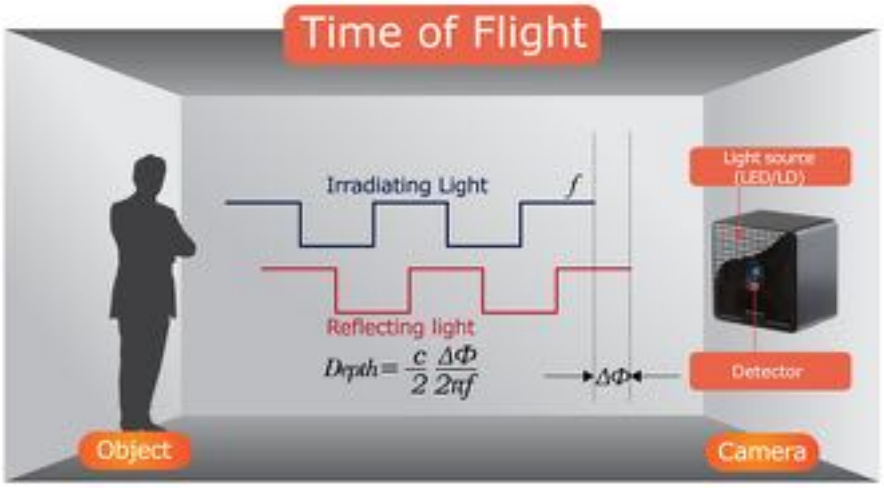

Figura 2.7: Funcionamiento de los sensores ToF [6]

Su funcionamiento consiste en iluminar la escena mediante una fuente de luz infrarroja modulada y medir el tiempo que tarda en llegar al sensor la luz reflejada por los diferente objetos presentes en la escena (ver Figura 2.7). Si consideramos t al tiempo que tarda el haz de luz en llegar al sensor óptico, c la velocidad de la luz ($3{,}10^8[m/s]$) y d la distacia al objeto, es fácil obtener el valor de profundidad mediante la Ecuación 2.1.

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

Sin embargo, la manera en la que este tipo de sensores calculan el tiempo de viaje de la onda de luz es mediante la diferencia de fase entre la onda emitida y la reflejada. Para ello, y como se menciona al anteriomente en este apartado, es necesario que la fuente de luz sea modulada mediante pulsos o mediante una onda continua. Generalmente se hace uso de una onda cuadrada debido a que es fácilmente reproducible mediante circuitos digitales.

Por tanto, se pueden distinguir dos métodos para el cálculo del valor de profundidad, en función de cómo esté modelada la fuente de luz.

- Modulación mediante luz pulsada: la fuente de luz ilumina la escena durante un breve periodo de tiempo Δt, y la energía de la luz reflejada es muestreada en cada píxel del sensor de manera paralela mediante dos ventanas temporales

fuera de fase, C1 y C2 con el mismoΔt. Las cargas eléctricas acumuladas durante los muestreos, Q1 y Q2 son cuantificadas y utilizadas para calcular la distancia de los puntos en la escena mediante la Ecuación 2.2.

$$d = \frac{1}{2} c \triangle t \left(\frac{Q_2}{Q_1 + Q_2} \right) \tag{2.2}$$

- Modulación mediante una onda continua (CW): a diferencia del método anterior, este toma múltiples muestras por medida, cada una de ellas desfasada 90º, hasta un total de 4 muestras. Con estas, se puede calcular la fase angular entre la luz emitida y la reflajada y a partir de la misma la distancia de los puntos en la escena mediante las Ecuaciones 2.3 y 2.4.

$$\varphi = arctan \left(\frac{Q_3 - Q_4}{Q_1 - Q_2} \right) \tag{2.3}$$

$$d = \frac{c}{4\pi f} \varphi \tag{2.4}$$

- El hecho de que este método base su funcionamiento en el la fase de la señal, que se repite cada 2π, implica que a una cierta distancia se empiece a producir aliasing. Esta distancia, conocida como distancia de ambigüedad y que se puede calcular como $d_{amb} = \frac{c}{2f}$, marca además la distancia máxima de funcionamiento. Por tanto, para alcanzar mayores distancias, se busca reducir la frecuencia de funcionamiento, aunque esto afecta de manera negativa a la precisión del sistema.Con el fin de sortear este incoveniente, se han desarrollado sistemas que hacen uso de múltiples frecuencias de modulación (ver Figura 2.8), cada una una distancia de ambigüedad, aunque la verdadera posición de los objetos es aquella en la que coinciden ambas frecuencias.

2.2.2. *Stereo Vision Sensors*

Este tipo de sensores, también conocidos como cámaras estereo, son unos dispositivos capaces de obtener la estructura 3D de una escena a partir de dos o más imágenes de la misma escena pero capturadas desde distintos puntos de vista, haciendo uso del principio de triangulación para conseguirlo.

Por lo general, la configuración utilizada consiste en dos cámaras instaladas una junto a la otra de manera que sus campos de visión se superpongan, imitando en cierto sentido la visión binocular humana.

Por cada punto visible en ambas imágenes, existen dos “rayos” o líneas conectan-

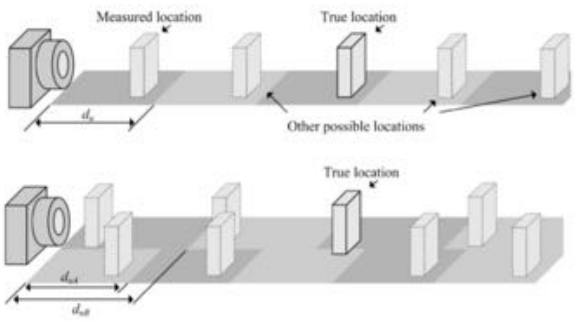

Figura 2.8: Ampliación de la distancia del sistema ToF mediante uso de multifrecuencia[7]

do esos puntos a los centros de proyección de cada cámara (ver Figura 2.9). Para poder obtener la posición 3D de la escena capturada, se necesita conocer la siguiente información.

- La correspondencia entre los puntos visibles por ambas cámaras: encontrar la ubicación de los puntos de superficie visibles en la imágen derecha en la imágen de la izquiera o viceversa.
- La geometría exacta entre ambas cámaras. Este dato es imprescindible para poder calcular el punto de intersección entre ambas líneas de visión para píxeles asociados de ambas imágenes. Dado que en estos sistemas las cámaras suelen estar montadas una junto a la otra en un soporte fijo, el cálculo de la geometría entre ambas únicamente es realiado una vez durante el proceso de calibrado.

Durante el proceso de calibrado se genera un modelo para la cámara estereo, que consiste en los datos intrínsecos de cada cámara (distancia focal y distorsión), y los llamados parámetros extrínsecos, que son el ángulo y rotación 3D entre ambas imágenes. A partir de este modelo se pueden triangular aquellos puntos indetificados en las imágenes y obtener sus coordenadas en el espacio tridimensional con respecto a la cámara.

Como se mencionaba previamente, para poder triangular los puntos de la escena es necesario encontrar la correspondencia entre puntos de ambas imágenes. Para ello se podría realizar una búsqueda exhaustiva en toda la imágen, sin embargo esto sería computacionamente costoso y requeriría de una gran cantidad de tiempo, por lo que este método no sería viable para aplicaciones en tiempo real. Haciendo uso de los datos de la geometría entre ambas cámaras calculada durante el proceso de calibrado, se puede acotar la búsqueda a una sola línea en una de las imágenes, conocida como línea epipolar (ver Figura 2.10).

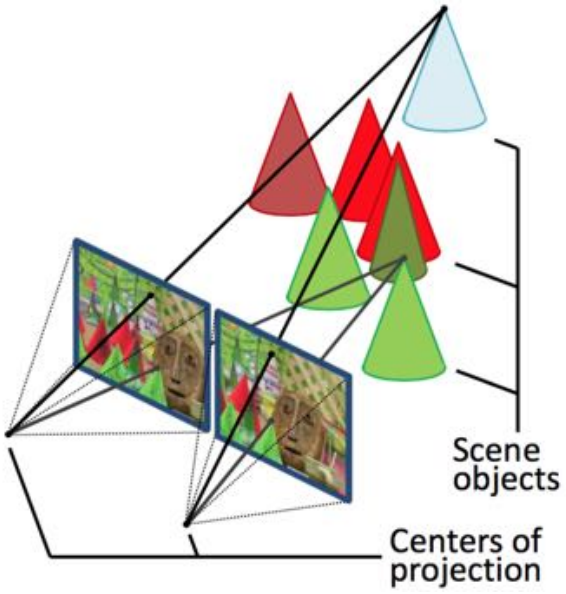

Figura 2.9: Esquema general de un sistema de visión estéreo [8]

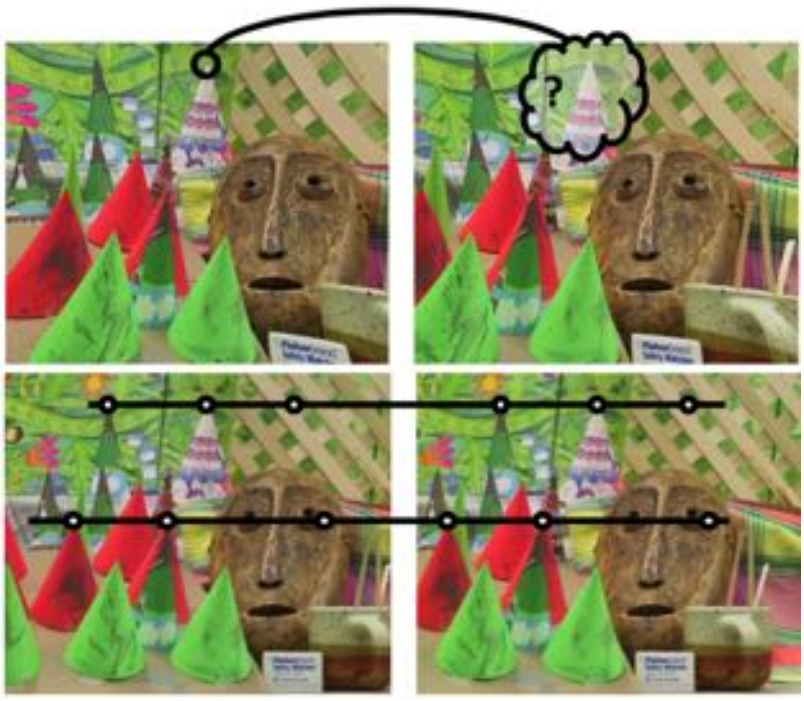

Figura 2.10: Búsqueda de equivalencia entre puntos a lo largo de la línea epipolar [8]

2.2.3. Sensores de luz estructurada

Junto a los sensores ToF dentro de la categoría de los sistema de visión activa, también se pueden encontrar lo que se conoce como sensores de luz estructurada. Este tipo de sensores proyectan sobre la escena un patrón de luz (de puntos, rayas o rejilla), el cuál sufre una distorsión al impactar sobre los diferentes objetos de la escena. El patrón reflejado es entonces capturado por una cámara o sensor infrarrojo (en función del tipo de luz utilizado), y a partir de la distorsión que éste haya sufrido (ver Figura 2.11), se consigue calcular la distancia a la cuál se encuentran los objetos en la escena. Para ello, es importante conocer la calibración o posición relativa entre el proyector y la cámara ó sensor.

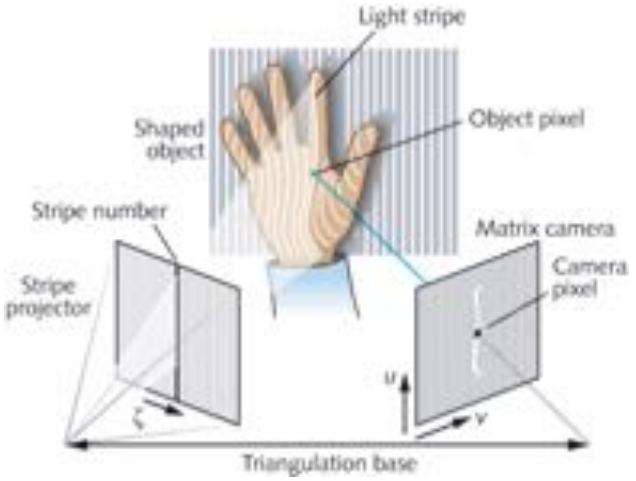

Figura 2.11: Principio de triangulación de patrones de línea [9]

La gran mayoría de este tipo de sistemas trabajan con luz infrarroja, con la consecuente ventaja de no interferir en la escena y así permitir la aplicación de otras técnicas de visión por ordenador de forma paralela. Sin embargo, dado que se tratan de sensores de visión activa, el alcance de los mismos está limitado a la distancia a la cuál la luz puede penetrar y ser reflejada de vuelta al sensor.

2.3. El sensor Kinect.

El sensor Kinect es un dispositivo lanzado en el año 2010 de la mano de Microsoft como un controlador para su videoconsola Xbox 360 lanzada en la misma época. La mayor innovación de este dispositivo es la incorporación de un sensor de luz estructurada en el mismo, con el que poder capturar la información 3D de la escena de la que extraer los movimientos de los jugadores y con ello permitir el control de la consola

y de una nueva generación de videojuegos que hacen uso de esta información.

Un año más tarde del lanzamiento del sensor Kinect, Microsoft liberó el SDK o Kit de Desarrollo, permitiendo a toda la comunidad de desarrolladores experimentar y crear sus propias aplicaciones para PC que hiciesen uso del dispositivo y todas las capacidades que este ofrece. Esto, unido al bajo precio del dispositivo, abrió un mundo de posibilidades en el campo de la visión por ordenador.

2.3.1. Características del sensor Kinect

El dispositivo Kinect, mostrado en la Figura 2.12, está compuesto por cuatro elementos diferenciables: una cámara RGB para la captura de imágenes a color, el sensor de profundidad (conformado por el emisor y sensor IR), un array de cuatro micrófonos distribuidos para la captura del sonido para su posterior procesado y reconocimiento de voz; y un motor que permite ajustar la inclinación del dispositivo.

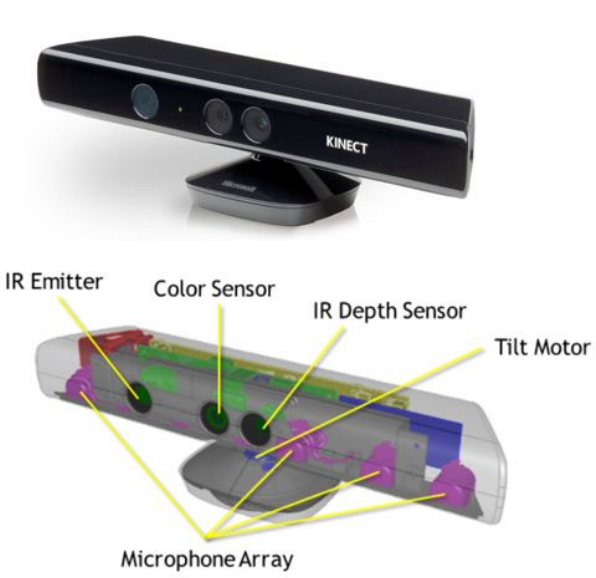

Figura 2.12: Sensor Kinect de Microsoft

Las imágenes a color capturadas por la cámara RGB están disponibles en tres formatos diferentes: RGB, YUV y Bayer. Cada uno de estos formatos permite la captura de vídeo a diferentes resoluciones y tasas de frames, como puede apreciarse en Tabla 2.1. Cada uno de estos formatos lleva asociada una configuración (tiempo de exposición, tasa de frames, gamma) por defecto, sin embargo estos parámetros son configurables mediante software.

El sensor de profundidad incorporado en la Kinect es uno de los que anteriormente

Formato	**Resolución**	**Frames/seg (fps)**
RGB (32-bit)	640x480	30
	1280x960	12
YUV (16-bit)	640x480	15
Bayer (32 bit)	640x480	30
	1280x960	12

Tabla 2.1: Formatos de imáges. Relación entre resolución y tasa de frames

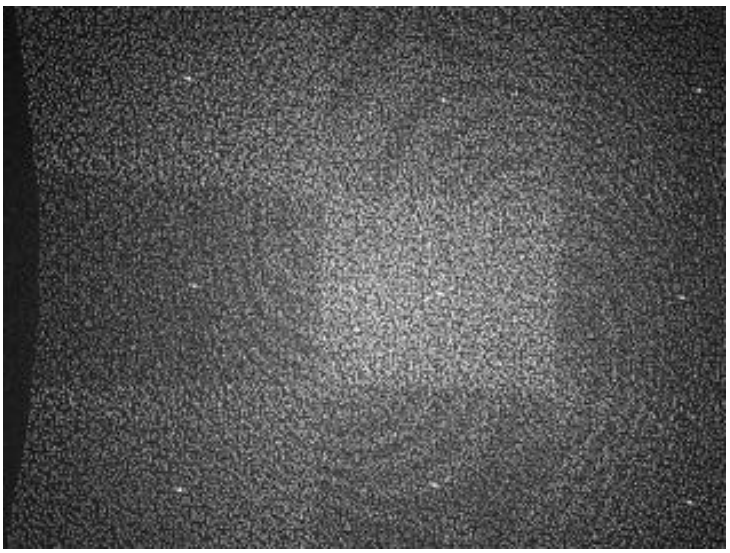

Figura 2.13: Patrón de puntos proyectado por el sensor Kinect [10]

se habían descrito como sensores de luz estructurada, dentro de los sistemas de visión activa. Como tal, el sensor está formado por un emisor infrarrojo que proyecta sobre la escena un patrón de puntos (como el que puede verse en la Figura 2.13), y un sensor monocromático CMOS que capta el patrón reflejado en la escena para su posterior procesado para el cálculo de la información de profundidad. Las resolución de las imágenes capturadas por este sensor pueden ser 640x480 (VGA), 320x240 (QVGA) y 80x60, y para todas ellas la tasa de frames se mantiene en 30 frames por segundo.

El funcionamiento de este sensor consiste en la proyección de un patrón de luz estructura infrarroja, en este caso se trata de un patrón de puntos (ver Figura 2.13), el cuál es reflejado al impactar sobre las diferentes superficies de la escena, y es capturado por el sensor CMOS. A partir de la distorsión sufrida por el patrón, el sensor es capaz de estimar la distancia a la cuál se encuentran los distintos elementos de la escena mediante triangulación referenciada a un patrón conocido por el sensor para una distancia concreta. La Figura 2.14 representa el método de triangulación utilizado.

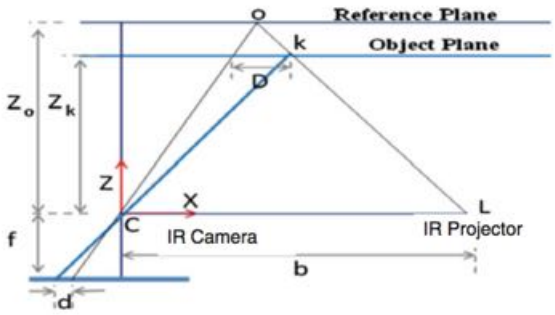

Figura 2.14: Método de triangulación para el cálculo de profundidad [11]

A partir de las diferencias entre el patrón capturado y el de referencia, se calculan un conjunto de valores de disparidad D y d (posición del objeto respecto al plano de referencia y el del objeto, y posición del objeto respecto al eje óptico de parámetros intrínsecos), y junto al valor de la distancia focal de la cámara IR, f, se consigue calcular la distancia a los diferentes elementos presentes en la escena mediante la Ecuación 2.5.

$$z_k = f * \frac{D}{d} \tag{2.5}$$

Además de la información de color y profundidad, el sensor Kinect es capaz de obtener un conjunto de datos de especial interés a la hora de llevar a cabo tareas de reconocimiento de actividades y/o acciones.

- Mapa de segmentación: el software incorporado le permite detectar hasta un máximo de seis personas, y almacenar en un mapa de bits la silueta de cada una de estas personas (siendo el valor de cada píxel de dicha imagen el índice asignado a cada silueta, 1-6). Esta información es de gran ayuda a la hora de realizar tareas de segmentación, permitiendo separar de manera sencilla el fondo de la persona cuyos moviemientos se desean analizar simplemente aplicando a las imágenes la máscara obtenida del mapa de segmentación.

- Esqueleto: de las seis personas que el sensor es capaz de reconocer, únicamente puede analizar y extraer un conjunto de características de dos de ellas. Estas características consisten en un conjunto de puntos[1] que se corresponden con diferentes articulaciones presentes en el cuerpo humano (como puede verse en la Figura 2.15), y de ahí que reciban el nombre de joints o articulaciones. Para

[1]El software de la Kinect permite configurar dos modos para la detección de las articulaciones del esqueleto: modo sentado, devolviendo únicamente información de las 10 articulaciones superiores; y modo de cuerpo completo, devolviendo la información de un total 20 articulaciones.

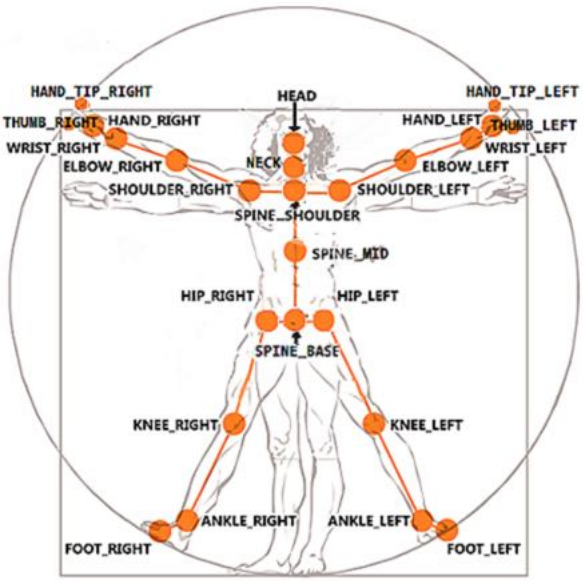

Figura 2.15: Identificación de las articulaciones identificadas por Kinect [12]

cada una de ellas, el sensor nos devuelve las coordenadas de las mismas (tanto en píxeles como en su posición espacial real), o un valor de 0 en aquellas que no sea capaz de detectar.

2.3.2. Limitaciones del sensor Kinect

Por cada frame, el sensor captura una imágen en escala de grises de todo aquello visible en el campo de visión del sensor de profundidad. Cada píxel de esta imagen contiene la distancia cartesiana, expresada en milímetros, desde el plano de cámara hasta el objeto más cercano para esas coordenadas *(x,y)* concretas. Sin embargo, debido a las limitaciones que presentan este tipo de sensores de profundidad y que ahora se expondrán, existirán píxeles negros (con valor 0) en aquellos puntos de la escena en los que el sensor ha sido incapaz de calcular la distancia.

Las limitaciones que presenta el sensor Kinect se pueden clasificar en dos categorías, atendiendo a la naturaleza de las mismas: limitaciones internas o inherentes a este tipo de sensores de profundidad; y externos o aquellos que dependen de las características de los diferentes elementos presentes en la escena.

Entre las limitaciones internas de la Kinect y este tipo de sensores, encontramos las siguientes:

- Rango de funcionamiento. Esta es una de las limitaciones más importante a tener en cuenta, ya que va a acotar la distancia de funcionamiento del sensor y

esto va a condicionar el uso que se quiera dar del mismo. Como puede verse en la Figura 2.16, el sensor Kinect tiene un rango de funcionamiento óptimo para distancias comprendidas entre 0.8m y 4m, para el modo de cálculo por defecto, y de 0.4m a 3m para el modo de campo cercano. A partir de estos rangos, la Kinect no es capaz de calcular correctamente la distancia a la cuál se encuentran los elementos de la escena, aumentando considerablemente el ruido en la escena y las zonas negras (aquellas zonas en las que no se ha podido calcular la profundidad).

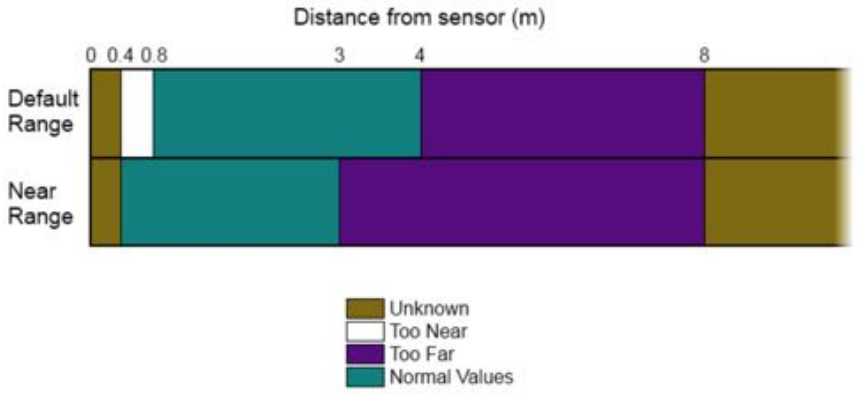

Figura 2.16: Rango de funcionamiento del sensor Kinect [13]

Para distancias muy cercanas, los puntos proyectados por el emisor IR se encuentran demasiado condensados como para poder ser correctamente reconocidos por el sensor, y de ahí que no se tengan valores fiables o directamente no se tengan para esas distancias. En el caso contrario, para distancias mayores el patrón de puntos se dispersa, de manera que hay una mayor superficie de la escena sin cubrir y por tanto el sensor no es capaz de calcular la profundidad (apareciendo huecos) o interpola los valores de profundidad obtenidos, resultando en valores poco fiables y por tanto de bajo interés para su uso. En la Figura 2.17 se reprenta una aproximación del error cometido por el sensor en función de la distancia.

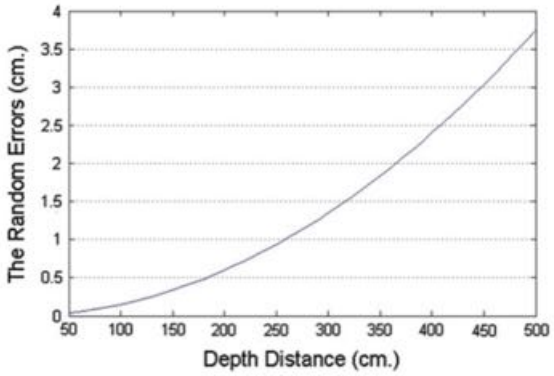

Figura 2.17: Desviación estándar en función de la distancia entre en plano de cámara y los objetos [11]

- Sombras IR. Debido a las características de este tipo de sensores de profundidad, en los que el emisor de luz y el sensor se encuentran separados el uno del otro, los objetos que se encuentran a una menor distancia del dispositivo proyectan una sombra sobre los que están más alejados, al impedir que la luz del proyector IR llegue a los mismo. Esto deriva en que no se pueda calcular la distancia a esos elementos de la escena, apareciendo en las imágenes como huecos negros (como puede verse en la Figura 2.18).

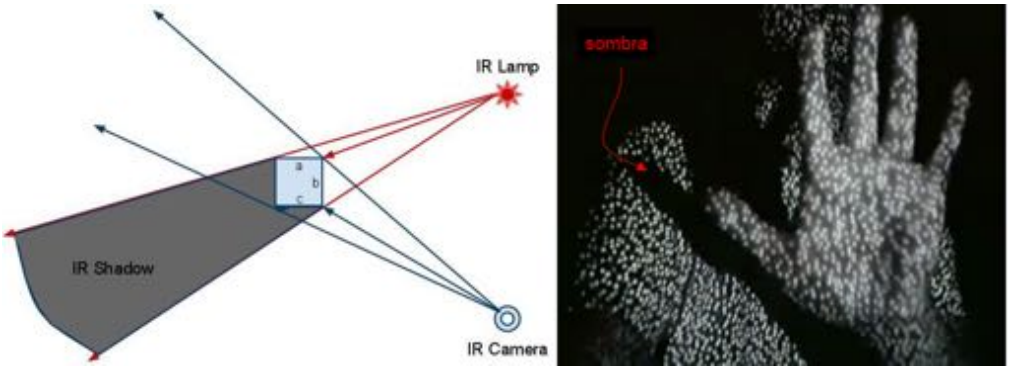

Figura 2.18: Explicación (izqda.)[14] y ejemplo (dcha.)[15] de las sombras IR del sensor Kinect

Por último, pero no por ello menos importantes, están las limitaciones externas al sensor y relativas a las propiedades de los diferentes elementos de las escenas. Los objetos transparentes. translucidos y especulares provocan que los puntos de luz que impactan sobre ellos sufran una dispersión, volviendo irreconocible la distorsión del patrón y por tanto impidiendo el correcto cálculo de la profundidad (un ejemplo de

esto serían los suelos reflectantes). La inclinación de las superficies respecto al proyecto IR también conlleva limitaciones a la hora de calcular la distancia a las mismas, dado que superficies casi o totalmente paralelas al rayo de luz provocarán que los mismos no incidan en estas, haciendo imposible la estimación de la profundad de las mismas.

2.4. Sistemas actuales basados en profundidad.

A lo largo de este capítulo se ha visto una breve introducción a los sistemas de reconocimiento de actividades humanas, con las características y problemas que los caracterizan, y se han introducido los sensores de profundidad y las ventajas aportadas por la información que son capaces de capturar. Es por tanto necesario presentar los sistemas de reconocimiento desarrollado durante estos años basados en la información de profundida. A continuación se detallan algunos de los más representativos en este campo (los artículos consultados puede verse en la Tabla 2.2).

- **HOJ3D (Histogram of Oriented Joints 3D)[5]**. Sistema de reconocimiento (ver Figura 2.19) basado en los puntos del esqueleto, utilizando histogramas con las posiciones 3D de los puntos para representar las posturas del cuerpo de manera compacta.

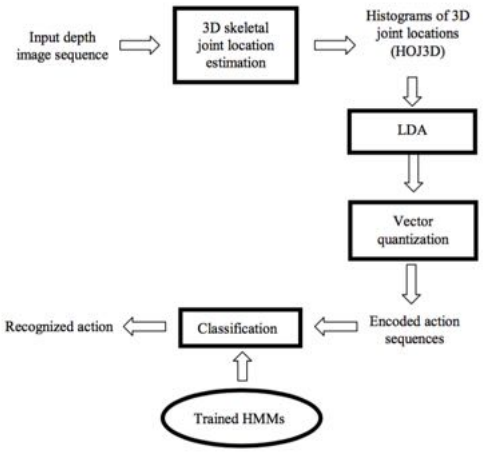

Figura 2.19: Esquema del sistema desarrollado basado en HOJ3D

De los 20 puntos del esqueleto calculados, únicamente hacen uso de 12 de ellos. Los 8 restantes son descartados ya que introducen redundancia (tobillos y muñecas, debido a su proximidad con las manos y pies) o no aportan información útil a la hora de modelar la postura del cuerpo (hombros, cabeza y cuello).

Utilizando el punto central de la cadera como eje de coordenadas y el vector entre el punto iqz. y dcho. como referencia horizontal (Figura 2.20.a), definen un sistema de coordenadas esférico (Figura 2.20.b) siempre alineado con la persona que proporciona invarianza ante cambios de vista.

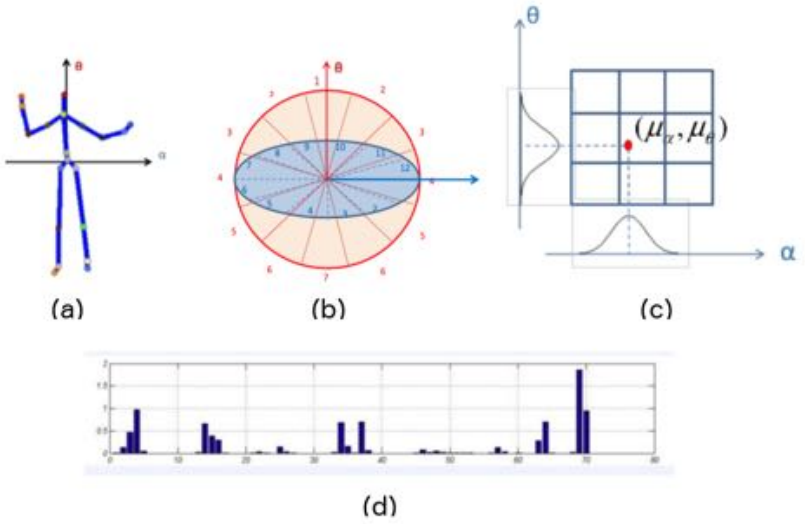

Figura 2.20: (a) Esqueleto obtenido a partir de los puntos, con el eje de coordenadas centrado sobre la cadera (b) Sistema de coordenadas propuesto (c)Distribución de la votación por pesos (d) Histograma obtenido [5]

El espacio 3D propuesto se divide en "n" porciones, siguiendo el siguiente esquema: el ángulo de inclinación se divide en 7 porciones con el ángulo θ comprendido del siguiente modo: [0, 15], [15, 45], [45, 75], [105, 135], [165, 180]. De manera similar, el azimut se divide en 12 porciones con una resolución de 30º. Esto nos da un total de 84 posibles localizaciones. La posición en la que caen los nueve puntos del esquelo restantes, junto con una votación por pesos (Figura 2.20.c) que contribuye a las porciones circundantes/vecinas, son lo que compone el descriptor HOJ3D.
Una vez obtenido el descriptor (Figura 2.20.d), realizan una selección de características aplicando LDA (Linear Discriminant Analysis) buscando las más discriminativas, y posteriormente realizan un clustering mediante k-means para reducir el número de símbolos observables, los cuáles servirán para entrenar un clasificador HMM discreto.

- **DMM-HOG (Depth Motion Maps - Histogram of Oriented Gradients)[16]**.

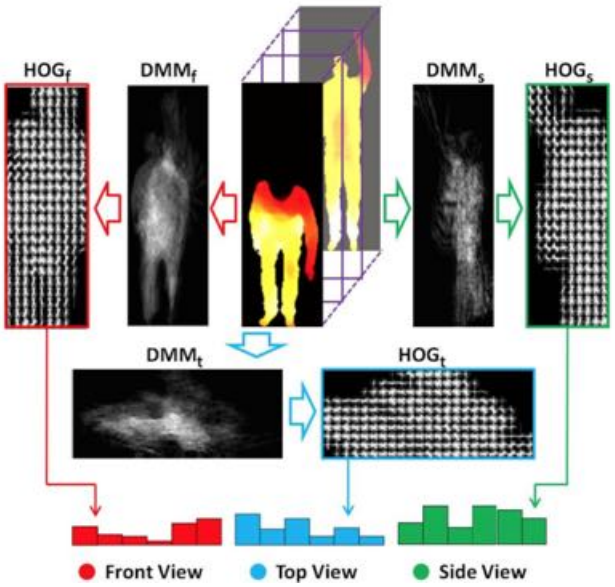

Figura 2.21: Esquema para la obtención de los descriptores DMM-HOG [16]

En el trabajo dearrollado por Yang et al., mencionado en la sección anterior, se propone un método efectivo y eficiente para el reconocimiento de acciones mediante histogramas de gradientes orientados (HOG) obtenidos a partir de Depth Motion Maps (DMM).
El método desarrollado consiste en proyectar cada frame en tres planos ortogonales, obteniendo de este modo tres mapas 2D que se corresponden con distintos puntos de vista para cada instante: vista frontal, lateral y superior (ver Figura 2.21).
Por cada plano proyectado, y a lo largo de la duración de la secuencia de vídeo, se calcula y acumula la energía de movimiento mediante la diferencia entre frames consecutivos y que superen un cierto umbral establecido de manera empírica. De este modo se obtienen lo que han definido como Depth Motion Maps.
A continuación los tres mapas de profundidad generados se particionan en una rejilla uniforme, y se calculan los histogramas de gradientes orientados. Tras aplicar por cada celda cuatro métodos de normalización basados en los histogramas adyacentes, se obtiene el decriptor DMM-HOG que consiste en la concatenación de los HOG correspondientes a cada proyección, y este será el dato de entrada para un clasificador SVM lineal que será el encargado de realizar el reconocimiento.

- **Action Recognition based on A Bag of Points[17]**. El trabajo de Li et

al. describe un método para reconocer acciones humanas a partir de secuencias de mapas de profundidad empleando grafos de acción[18], en el que cada nodo del mismo se corresponde con una posutra saliente representada por lo que denominan bags of points, y en el que cada acción se codifica como uno o varios caminos del grafo. De esta forma consiguen modelar de manera explícita la dinámica de las acciones para las que se entrena el sistema.

Para la obtención de los BOP's, dado que los mapas de profundidad están for-

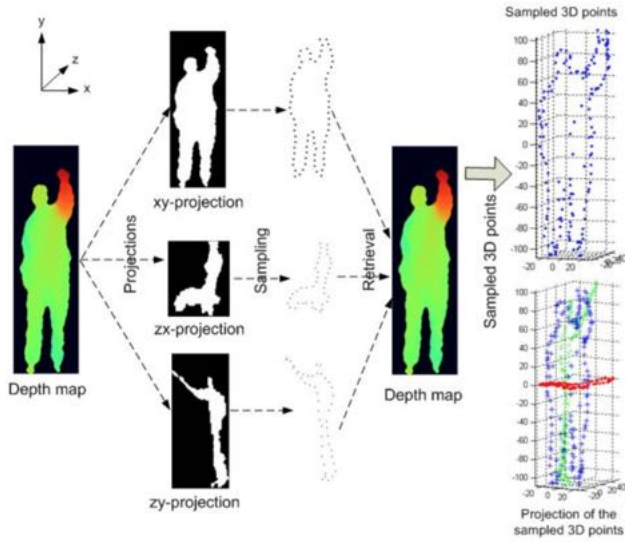

Figura 2.22: Proceso de muestreo de puntos 3D representativos[17]

mados por una gran cantidad de puntos, es necesario realizar un muestreo para reducir la cantidad de datos a procesar pero sin perder por ello la información sobre la postura del cuerpo. Para ello, de manera similar al trabajo sobre DMM, los mapas de profundidad se proyectan sobre tres planos ortogonales obteniendo de este modo tres imágenes 2D con las siluetas de la persona desde distintos puntos de vista (frontal, lateral y cenital). A continuación se seleccionan un número determinado de puntos equidistantes del contorno de las siluetas de cada proyección y se muestrean los puntos 3D correspondientes (ver Figura 2.22).

Asumiendo que la distribución de puntos se puede aproximar mediante una combinación de Q componentes gaussianas y que estos son estadísticamente independientes, cada postura corporal se puede modelar mediante la función de distribución conjunta de los puntos.

El grafo de acción (ver Figura 2.23), que codifica L acciones mediante M posturas salientes, se puede representar como $G = \{\Omega, A, A_1, A_2, A_3, ..., A_L\}$, en el que cada postura se corresponde con un nodo, A_k es la matriz de transiciones

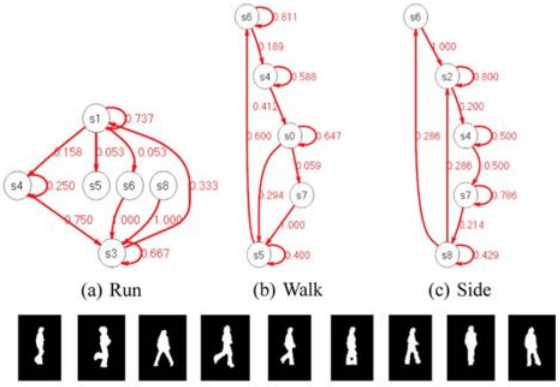

Figura 2.23: Ejemplo de grafo de acción para tres acciones diferentes [18]

para la acción k, y A es la matriz de transiciones global para todas las acciones.

- **Reconocimiento a partir de mapas de profundidad mediante MHI y momentos de HU [19]**. Megavannan et al. proponen un método para el reconocimiento de acciones a partir de mapas de profundidad basado en MHI y momentos de HU (en la Figura 2.24 se muestra es esquema del sistema propuesto).

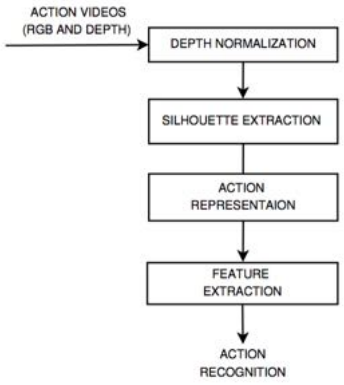

Figura 2.24: Esquema del sistema propuesto en [19]

Con el fin de modelar y representar la información espacial y temporal del movimiento mediante una única imágen, se hace uso de MHI's (Motion History Images)[31](Ec. 2.6), ampliamente utilizados en sistemas de reconocimiento 2D. En este tipo de imágenes la intensidad de cada píxel depende de la temporalidad del movimiento, siendo más intenso cuanto más reciente ha sido dicho movimiento. Y para incorporar la información de profundidad que aportan los sensores

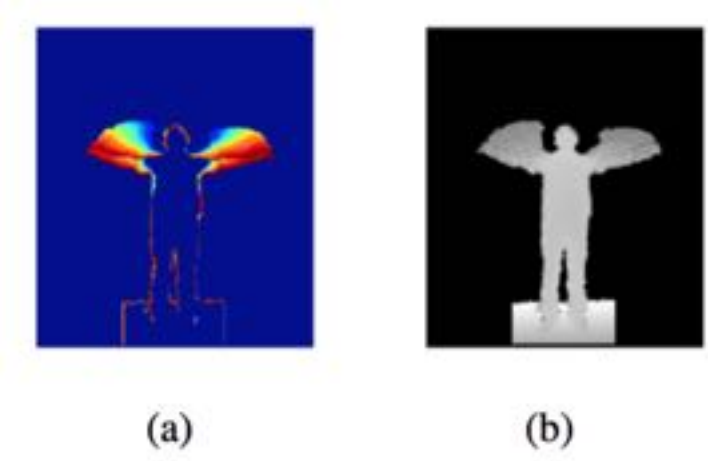

Figura 2.25: a) MHI b)Average Depth Image

como el sensor Kinect, exploran la idea de utilizar una imágen con el valor medio de profundidad (Ec. 2.7) por píxel calculado a lo largo de N frames consecutivos.

$$I_{MHI}(i,j,t) = \begin{cases} \tau & ,B_{diff}(i,j,t) = 1 \\ max(0, I_{mhi}(i,j,t-1) - \tau) & ,resto \end{cases} \tag{2.6}$$

$$I^{k}_{avg=} \frac{\sum_{t=k}^{k+N-1} D(i,j,t)}{\sum_{t=k}^{k+N-1} B(i,j,t)} \tag{2.7}$$

A partir de estas dos imágenes (Figuras 2.25.a y 2.25.b), haciendo uso de los Momentos de Hu[22], presentados por Ming-Kuei Hu, se extraen una serie de características invariantes ante rotación, escala y traslación y que constituiran los descriptores del sistema propuesto.

Artículo	Descriptor	Clasificador	Dataset	Accuracy
[17]	Bag of Points	ActionGraph	MSRAction3D	74,7 %
[21]	DMA + DMH	SVM	MSRAction3D	90,45 %
[33]	Eigen Joints	NBNN	MSRAction3D	82,33 %
[19]	Hu Moments(MHI+ADI) + HBB	SVM	Own.	89,96 %
[16]	DMM + HOG	SVM	MSRAction3D	91,63 %
[34]	Hu Moments (SEI + SHI)	SVM	FBGDB	83,62 %
			KTHDB	87,16 %
[5]	HOJ3D	HMM	MSRAction3D	78,97 %
[3]	HON4D	SVM	MSRAction3D	88,89 %

Tabla 2.2: Artículos consultados

Capítulo 3

Diseño.

3.1. Introducción.

El principal objetivo en este proyecto ha sido el de desarrollar un sistema capaz de reconocer una serie de acciones humanas en vídeo, basándose en los modelos generados a partir de un conjunto de secuencias de ejemplo extraídas de los datasets capturados por diferentes grupos de investigación (más adelante se detallan los diferentes datasets utilizados).

Con el fin de dotar al sistema desarrollado de una mayor versatilidad, durante su diseño se han tenido en cuenta los principios de la programación o desarrollo modular, que consiste en la división del programa en diversos módulos con el objetivo de facilitar la depuración, modificación y mantenimiento del sistema. Los módulos en los que se ha subdividido el sistema coinciden con los cuatro bloques principales que conforman un sistema de reconocimiento, y de los que se hablará con más detalle más adelante: preprocesado de datos, representación de la acción, extracción de características y entrenamiento o modelado.

Esto ha permitido desarrollar un sistema en el que no sólamente es fácil modificar los diferentes parámetros de cada bloque, sino que además permite una rápida integración de algoritmos (sustracción de fondo, extracción de características, ...) y clasificadores, lo que permite probar de manera sencilla diferentes configuraciones del sistema en busqueda de aquella que mejor se adapte al conjunto de situaciones y acciones que se quieren reconocer.

Para realizar este proyecto, desarrollado en Matlab, se ha tomado como punto de partida el sistema desarrollado por Bulling et al.[20], diseñado para el reconocimiento de actividades del ser humano a partir de los movimientos de la persona capturados mediante sensores de inercia. Por ello, ha sido necesaria su modificación para poder

trabajar con imágenes como fuente de datos de entrada, así como la creación de los diferentes "bloques" asociados al reconocimiento de acciones mediante procesado de video. Además, como se detallará más adelante, se han implementado e integrado algoritmos para la extracción de características para probar el funcionamiento del sistema desarrollado.

3.2. Estructura de diseño.

En esta sección se va a presentar la estructura y diseño del sistema desarrollado, así como una explicación más detallada de cada uno de los bloques en los que, como se han mencionado previamente, se subdivide el sistema.

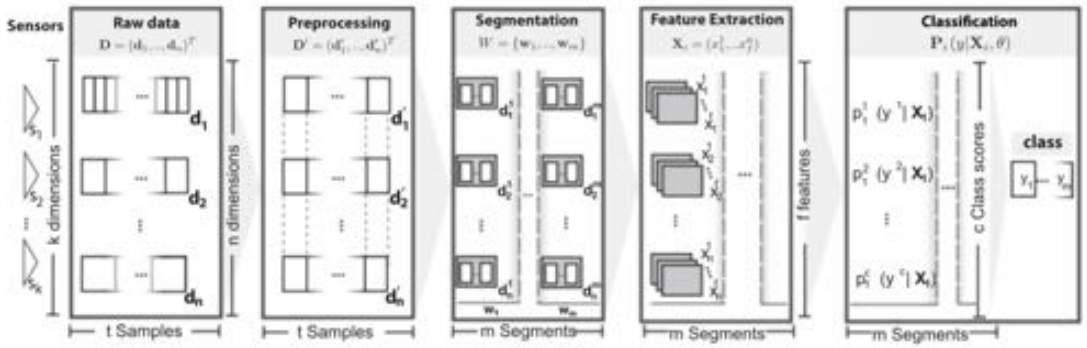

Figura 3.1: Activity Recognition Chain (ARC) [20]

3.2.1. ARC-Activity Recognition Chain

Una cadena de reconocimiento de actividades es una secuencia de técnicas de procesado de señal, reconocimiento de patrones y machine learning que conforman un sistema de reconocimiento específico y que le proporciona a este su comportamiento. Como puede verse en la Figura 3.1, la estructura de un ARC guarda un gran parecido con otros sistemas de reconocimiento de propósito general, pero como se verá en posteriores secciones, posee una serie de características y restricciones propias de estos sistemas. Es importante destacar además que la cadena se puede ejecutar en dos modos diferentes, entrenamiento y test/clasificación, en el caso de hacer uso de algoritmos de clasificación supervisada. Para algoritmos de clasificación sin supervisión, el sistema no requiere de una etapa específica de entrenamiento, sino que a partir de los datos de entrada es capaz de inferir qué actividad se está llevando a cabo.

Como entrada a la cadena de reconocimiento, se tienen los datos capturados por

el sensor de profundidad utilizado, aunque el sistema desarrollado se ha enfocado en especial a la información proporcionada por el sensor Kinect. A continuación esta información se preprocesa para ser utilizada más adelante en las siguientes etapas de la cadena, normalizando los datos de profundidad y a continuación segmentando las secuencias de videos en secciones de interés que puedan contener información referente a la actividad que se esté realizando. Estos segmentos son posteriormente procesados mediante diferentes algoritmos, como puede ser MHI [19], para modelar el movimiento en cada uno de los segmentos y es a partir de estos modelos de los que se extraen las características que van a caracterizar las acciones.

En la etapa de entrenamiento, las características extraídas y las etiquetas de las clases obtenidas meditante ground truth son usadas como entrada de los diferentes algoritmos de clasificación, de los cuáles se obtiene un modelo que contará con la información para reconocer, con menor o mayor acierto, cada una de las actividades para las que se ha entrenado al sistema. Estos modelos generados y las caracterísitcas de los datos de entrada del sistema son utilizados en la etapa de test/reconocimiento para calcular una serie de puntuaciones o scores para cada una de las clases, y después en función de estos decidir para cada uno de los datos de entrada, que actividad se estaba realizando.

3.2.2. Captura y entrada de Datos

En la primera etapa de un ARC típico, se realiza la captura de datos RAW mediante el sensor elegido para ser procesados y usados en etapas posteriores. Para este proyecto en particular, cuyo objetivo es el reconocimiento de acciones a partir de información de profundidad, el sensor utilizado es un *sensor de profundidad*. Más específicamente, el sensor Kinect desarrollado por Microsft (la versión 1.0) y que en la Sección 2.3 se explicó con más detalle.

Por tanto en esta etapa, haciendo uso del sensor Kinect, se realiza la captura de información de la escena: capturando la información de profundidad, color, así como la información extra que nos proporciona la Kinect como son los puntos del esqueleto y el mapa de segmentación. Estos datos, o los aportados por otros grupos de investigación que hayan realizado sus propios datasets, serán los datos que servirán como entrada al sistema para su posterior análisis y procesado.

3.2.3. Preprocesado y segmentación de datos

Como se mencionaba previamente, el origen de los datos que se utilizarán como entrada del sistema no tiene por qué ser necesariamente el sensor Kinect, pudiendo

haber sidos capturados por medio de otros tipos de sensores de profundidad como los mencionados en el capítulo anterior, y por tanto presentar algunas diferencias entre sí (diferente resolución de profundidad o formato de los datos). Por ello es importante que se realice una *normalización de los datos de entrada*, de manera que el sistema pueda trabajar con ellos independiente de las diferencias existentes entre datasets o datos de salida del sensor utilizado.

Además en las escenas, salvo que se capturen bajo condiciones ideales (sin que otros objetos y/o personas aparezcan u obstaculicen al sujeto principal), será necesario llevar a cabo una sustracción de fondo, de manera que se elimine la información que no resulte de interés para el reconocimiento. Esta tarea, que en imágenes RGB resulta más costoso y requiere de la aplicación de algoritmos de sustracción de fondo (como alguno de los evaluados en el trabajo de Y.Beneth et al. [35]), con los datos de profundidad se simplifica ya que estos aportan información acerca de la estructura de la escena, y en situaciones no muy complejas simplemente eliminando la información que se encuentre a determinadas distancias del sensor (entre un límite superior e inferior) permiten quedarnos con la información de interés. En el caso de hacer uso del sensor Kinect, que nos facilita una máscara binaria con la silueta de la/s persona/s encontrada en la escena, simplemente aplicando esta máscara sobre el mapa de profundidad permite obtener los datos a procesar en las siguientes etapas del sistema.

En la etapa de segmentación se identifican aquellos segmentos de datos preprocesados en los que existe una elevada probabilidad de contener información acerca de las acciones y actividades realizadas. Esta información no sólamente es útil para la tarea de reconocimiento, sino que puede servir para otros propositos como detener el funcionamiento del ARC en los momentos en los que no se detecte ninguna actividad y de ese modo ahorrar recursos y energía.

Segmentar un flujo de datos es una tarea compleja. Los seres humanos realizan las acciones y actividades de manera fluida y continua, mezclándose unas con otras en vez de estar separadas mediante pausas. Además, los límites de las actividades no están claramente definidos. Por ejemplo, la actividad de beber agua puede comenzar en el momento en el que se coge el vaso y terminar cuando se vuelve a soltar, o comenzar manteniendo el vaso y terminar en el momento después de beber. Todo esto requiere que se apliquen técnicas de segmentación que separen las actividades en segmentos o unas actividades de otra en su totalidad. Algunas de las técnicas más utilizadas son la segmentación por ventana deslizante y segmentación basada en energía.

- Sliding Window. El método de segmentación por ventana deslizante consiste en el desplazamimento de una ventana de duración t para extraer segmentos de datos del flujo que se utilizarán en las siguientes etapas del ARC. El tamaño

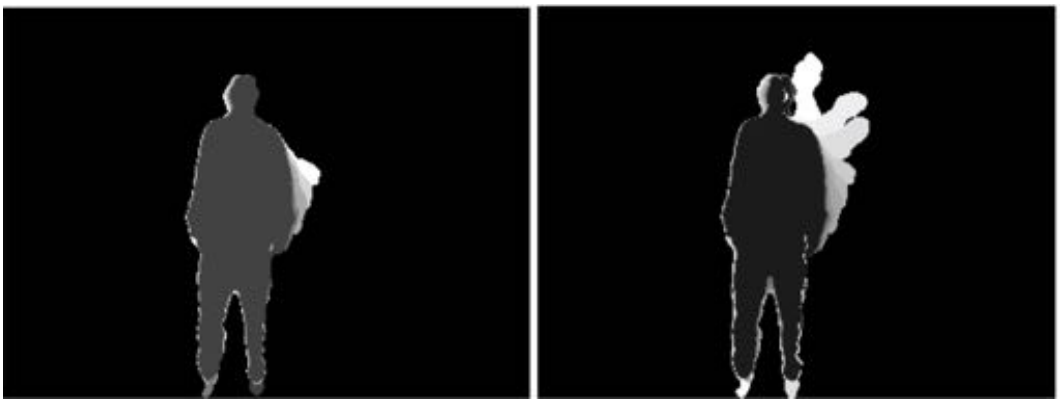

Figura 3.2: Representación de la cantidad de información en función del tamaño de ventana

de la ventana tiene un impacto directo en el tiempo de ejecución del sistema de reconocimiento: cuanto mayor es el tamaño de la ventana, más información tiene que ser procesada posteriormente (ver Figura 3.2) y por tanto más tiempo tiene que esperar el sistema a recibir un nuevo segmento de datos. Otro parámetro importante de este método es el salto entre ventanas, cuyo tamaño/duración tiene relación con la carga computacional y la precisión a la hora de definir los bordes de segmentación: cuanto mayor es el tamaño del salto, menor es el número de ventanas que se van a procesar y por tanto disminuye la carga del sistema, a costa de una disminución en la precisión.

- Segmentación basada en energía. Este método se basa en el hecho de que el movimiento captado por el sensor a lo largo de un número n de frames, se puede traducir directamente en distintos niveles de energía, los cuáles variarán en función de la actividad realizada. Por tanto, filtrando/umbralizando respecto a E, los segmentos de datos pueden ser identificados como pertenecientes a una misma actividad (con un cierto grado de probabilidad). Para la aplicación de este método, por lo general se le solicita al sujeto que entre actividades realice una posición de reposo, la cual el sistema identificará y establecerá el límite del segmento.
  Puesto que esta situación no es natural, [Zinnen et al. 2009b] propone un sistema de ventana deslizante adaptativo basado en las pausas que realizamos de manera natural.

3.2.4. Representación de la acción y extracción de características

La siguiente etapa en la cadena de reconocimiento es la extracción y selección de características, sin embargo los mapas de profundidad o puntos del esqueleto por sí mismos no son lo suficientemente representativos como para poder extraer unas características robustas ante la variabilidad de los datos de entrada, y modelar los movimientos realizados. Por tanto se require un paso previo de modelado de posturas y movimientos de los que extraer las características para formar los descriptores con los que se entrenará el sistema y/o se procesarán para su reconocimiento. Es en este punto de la cadena de reconocimiento en el que se aplican los métodos como MHI[19] o los explicados con más detalle en el estado del arte para modelar la dinámica de la escena.

La extracción y selección de características tiene como objetivo reducir las señales o datos de entrada a un conjunto de características que sean discriminativas para la actividades con las que se esté trabajando. El número total de características extraídas del conjunto de datos conforman lo que se conoce como feature space o espacio de características (ver Figura 3.3), y resulta intuitivo deducir que cuanto más separadas estén las características relativas a cada actividad dentro de este espacio, más fácil resultará separarlas y por tanto mejores resultados obtendrá el sistema a la hora de realizar el reconocimiento. El caso ideal es aquél en el que todas las características de una misma actividad se encuentran agrupadas en el espacio de características, y las correspondientes a actividades diferentes lo más alejadas posible. Estas características, además de discriminantes, es importante que sean robustas ante diferentes sujetos y la variabilidad intraclase de una actividad que se pueda producir.

Cuanto mayor sea la dimensionalidad del espacio de características, mayor será la cantidad de datos necesarios para modelar eficientemente la estimación de parámetros y más costoso computacionalmente. Esto supone que a la hora de diseñar sistemas de procesado en tiempo real, se busque reducir lo máximo posible el número de características extraídas pero sin que ello suponga un empeoramiento apreciable de la eficiencia del sistema.

3.2.5. Entrenamiento y Clasificación

Machine learning o aprendizaje automático es una rama de las ciencias de la computación cuyo objetivo es el desarrollo de técnicas y algoritmos que permitan a los sistemas informáticos aprender y realizar predicciones a partir de un conjunto de datos. Estas técnicas son ampliamente utilizadas en aplicaciones de motores de búsqueda, reconocimiento de patrones y de visión por ordenador.

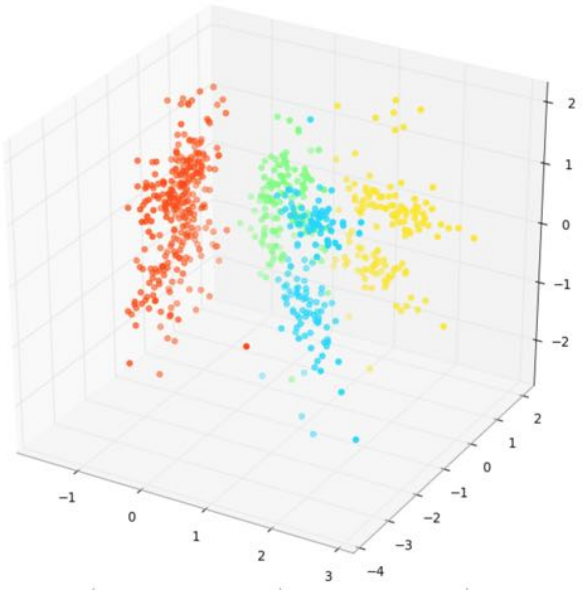

Figura 3.3: Ejemplo de un espacio de características de dimensionalidad 3, y características pertenecientes a 4 clases distintas

Dependiendo de los datos de entrada, se pueden distinguir dos tipos de aprendizaje entre los cuales se clasificarán los distintos algoritmos:

- Aprendizaje supervisado. Este tipo de algoritmos tratan de obtener una función a partir de un conjunto de datos etiquetados que les permitan mapear y clasificar los nuevos datos. En este caso, los datos de entrenamiento están formados por un vector de datos y el valor deseado de salida del algortimo (la clase a la que pertenecen los datos del vector).
  Algunos de los más conocidos y utilizados son kNN (k-Nearest Neighbor)[36], SVM (Support Vector Machine)[37], Decision Trees [38] y HMM (Hidden Markov Model)[23].

- Aprendizaje sin supervisión. En este caso, los algoritmos tratan de obtener una función que describa la estructura oculta de un conjunto de datos sin etiquetar. Dado que los datos a la entrada no están etiquetados, no se produce ninguna señal que indique el error o éxito de la clasificación que permita evaluar una posible solución. Algunas de las soluciones de esta categoría son la redes neuronales y técnicas de clustering como k-means [39].

La elección del algoritmo está sujeto a un compromiso entre la complejidad computacional y la eficiencia del reconocimiento. Con vistas al desarrollo de aplicaciones que funcionen sobre sistemas con recursos limitados, el objetivo es minimizar todo lo

posible el computacional y requisitos de memoria pero manteniendo un alto grado de rendimiento y eficiencia en el reconocimiento. La selección de características permite identificar aquellas características más representativas reduciendo de este modo el coste computacional durante la etapa de clasificación.

Se distinguen dos etapas diferenciadas:

- Training (entrenamiento): en esta etapa es en la que se genera el modelo θ a partir de los datos de entrenamiento $T = \{(X_i, y_i)\}_{i=1}^{N}$, siendo X_i los vectores de características e y_i sus correspondientes etiquetas.

- Clasificación: en un primer paso, los vectores X_i se mapean a un conjunto de etiquetas de clase $Y = \left\{y^1, ..., y^c\right\}$obteniendo un conjunto de scores o valores de confianza para cada clase $P_i = \left\{p_i^1, ..., p_i^c\right\}$. A continuación, haciendo uso de los scores obtenidos se realiza la decisión o predicción, por ejemplo mediante la elección de la clase en la que se ha obtenido el mayor score.

Estos algoritmos, partiendo de un set de datos de entrenamiento basado en observaciones del objeto de estudio, generan un modelo que les permitirá realizar decisiones o predicciones sobre unos datos de test.

Capítulo 4

Desarrollo.

En este capítulo se tratará el funcionamiento del sistema desarrollado y elementos integrados (ver Tabla 4.3), y la organización y estructura de los diferentes bloques de los que se ha hablado anteriormente (ver Sección 3.2.1). Se hablará en detalle del formato de entrada y salida de los datos a cada uno de los bloques de manera que resulte sencillo realizar modificaciones en el sistema para la integración de nuevos algoritmos o bloques de procesado.

4.1. Organización y estructura.

A lo largo del capítulo se han explicado los diferentes bloques que componen una cadena típica de reconocimiento de actividades, y en la cuál se ha basado el diseño del sistema desarrollado en este PFC. Estos bloques se han organizado en una estructura compuesta por tres módulos (ver Figura 4.1):

- Captura: en este módulo se engloban tanto el proceso de caputra de datos del sensor Kinect, como el tratamiento de los datos (pertenecientes a otros datasets y con un formato distinto) con el fin de adaptarlos para su correcto funcionamiento con el sistema.
- Procesado de datos: en este módulo se engloban tanto las tareas de preprocesado (normalización y background substraction) y segmentación de datos, así como las tareas de modelado de silueta y movimiento y extracción de características.
- Machine Learning: en el último módulo se realizan las tareas de apredizaje automático del sistema, por un lado el bloque de entrenamiento o training del sistema, y por otro el correpondiente a la parte de test y obtención de resultados (expuestos en el Capítulo 5).

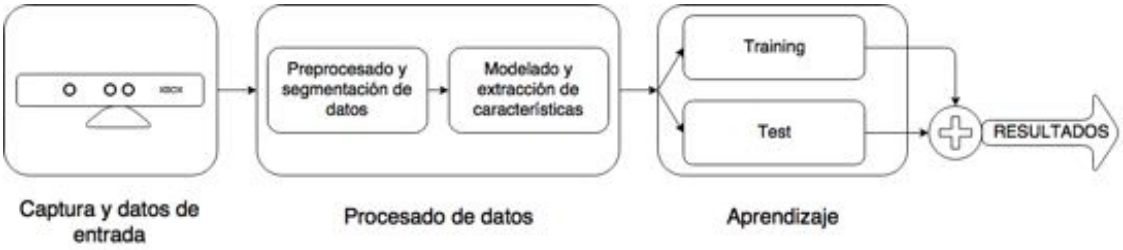

Figura 4.1: Organización y estructura del sistema desarrollado

Estos módulos, que en el sistema de referencia [20] utilizado se ejecutaban de manera consecutiva, se han rediseñado permitiendo ejecutar cada bloque de manera independiente. Para ello, a la salida de cada bloque se almacenan los datos generados en un conjunto de ficheros que a continuación pueden ser leídos por los siguientes módulos para su procesado.

La ventaja de este diseño es que permite condensar toda la carga computacional del sistema, siendo necesario procesar los datos RAW una única vez. Una vez procesados y extraídos los descriptores (vectores de características), estos quedan almacenados en disco listos para ser analizados y realizar pruebas sobre ellos (distintos algoritmos de selección de características, clasificadores, combinación de descriptores). Una ventaja inherente a lo descrito anteriormente es que una vez procesados los datos RAW, no es necesario mantenerlos en disco (salvo que se quieran aplicar nuevos algoritmos de modelado de silueta/moviemiento y extracción decaracterísticas) por lo que se pueden almacenar externamente, consiguiendo la consecuente liberación de espacio en disco. Esto es especialmente importante en sistemas informáticos con recursos y potencia limitados.

4.2. Captura y datos de entrada.

En esta sección se expondrá de manera general el proceso de captura de datos mediante el sensor Kinect y la forma en la que se tienen que organizar los datos (tanto los capturados como los pertenecientes a otros datasets) para poder hacer uso de ellos. Una explicación más detallada puede encontrarse en el Apéndice B

4.2.1. Captura de datos Kinect.

Para la captura de datos mediante la Kinect se han desarrollado una serie de scripts (detallados en los Apéndices B.1 y B.2) que haciendo uso de las herramientas disponibles para Matlab para la configuración y control del sensor Kinect, se encargan

de aplicar la configuración introducida por el usuario y de capturar los distintos datos ofrecidos por el sensor: *RGB*, *profundidad* y *metadatos* (*esqueleto*, *máscara de segmentación*).

Estos scripts además están diseñados de manera que los datos capturados se organicen de acuerdo al formato y estructura de directorios definido, de modo que sean totalmente compatibles con el sistema desarrollado y de esta manera facilitar al usuario la tarea de creación de un dataset propio sobre el que realizar pruebas.

4.2.2. Organización y Formato de datos de entrada.

El sistema desarrollado a lo largo de este PFC está pensado para usarse no sólamente con los datos capturados por el mismo, sino para poder utilizar además dataset externos. Los grupos de investigación que han creado estos datasets organizan los datos y nombran los ficheros de forma distinta y/o los almacenan en ficheros con distinto formato, y a raíz de esto surgen problemas a la hora de hacer uso de sus datos. Por ello, se ha definido una estructura de directorios (como la mostrada en la Figura 4.2) y una serie de especificaciones que deberán presentar los disintos ficheros de datos con el objetivo de unificar sus formatos y de este modo poder hacer uso de los distintos datasets existentes mediante el sistema desarrollado. En el Apéndice B.3 se muestra con más detalle tanto la estructura de directorios como el formato que deben presentar los distintos ficheros de datos.

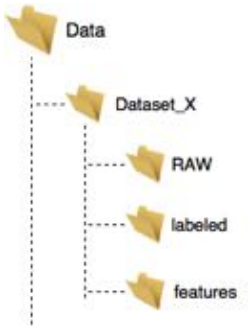

Figura 4.2: Estructura de directorios para los datos de entrada

4.3. Procesado de datos.

Las tareas de procesado de los datos capturados por el sensor Kinect o los aportados por datasets externos se realizan en el segundo módulo en que se organiza el sistema, y que como se mostraba en la Figura 4.1 se subdivide en dos bloques

diferenciados: preprocesado de datos y segmentación, y modelado y extracción de características. A lo largo de esta sección se detallarán cada uno de ellos y las funciones y scripts que se encargan de realizar estas tareas.

El modulo de procesado de datos está controlado por el script *generateTrainingData.mat*. Este es el encargado de cargar al comienzo de su ejecución la estructura con las variables de configuración del sistema (ver Apéndice A) y los datos almacenados en los ficheros del directorio /features del dataset seleccionado, los cuales pasa como parámetros a la función run_processdata.mat que es la encargada de realizar las llamadas a cada una de las funciones cuyo propósito es realizar las diferentes tareas específicas de este módulo.

Todos los datos generados y que recibe al final de su ejecución se almacenan en una estructura (ver Tabla 4.1) junto con aquella información relativa a la config. del sistema que ha permitido extraer dichas características. Como se menciona en el Apéndice B.3, por cada secuencia procesada únicamente existará un fichero en el que se guardarán estas estructuras, y sólamente tendrá una estructura para una misma configuración. Esto es, a la hora de guardar la estructura de datos en el fichero correspondiente, se realiza una comprobación de los datos almacenados en el mismo: en caso de no encontrarse la configuración se procede con el guardado de la nueva configuración en el fichero, y en caso contrario se sobreescriben los datos.

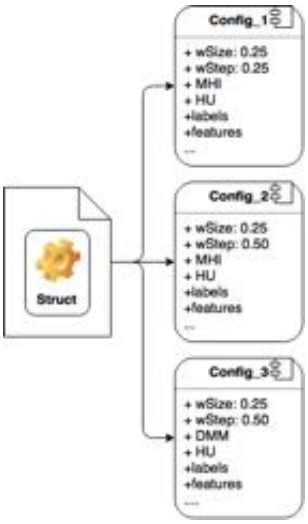

Figura 4.3: Estrutura de datos de los ficheros de features

Como se ha comentado previamente, la función en la que se realizan las llamadas a las funciones encargadas de realizar las distintas tareas de procesado sobre los datos de entrada es la función run_processdata().

Features Struct	
aType	Técnica de modelado de la silueta/movimiento (ej. MHI)
fType	Tipo de características extraídas (ej. Momentos de Hu)
features	Matriz MxN con los vectores de características extraídos
featureLabels	Vector con las etiquetas para cada segmento
segmentation	Matriz Mx2 con el número de frame de comienzo y fin de cada segmento
wSize	Tamaño de la ventana (s)
wStep	Tamaño del paso entre ventanas (s)

Tabla 4.1: Información incluída en la estructura de features

[features fDescr segments segmentation labelsSegmentation] = run_processdata(data, labels, segments, SETTINGS, varargin)

Esta función recibe los siguientes datos como argumentos:

- *data*: array de datos de dimensiones MxNxframes
- *labels*: vector Mx1 con la etiqueta de la clase correspondiente para cada frame
- *segments*: matriz de datos en la que se indica el frame de inicio y fin para cada "segmento" de etiquetas consecutivas de una misma clase, la duración del segmento y la etiqueta correspondiente.
- *SETTINGS*: estructura con los parámetros de configuración del sistema (detallado en el Apéndice A).
- *varargin*: argumento de datos de entrada de longitud variable. Dado que los datos del dataset pueden no incluir la información de metadatos y/o esqueleto, esta función los recibe como un argumento variable, de modo que se pueda realizar la llamada a la función con o sin esos datos.

Con estos datos realiza las llamadas a las funciones de extracción de silueta, segmentación y extracción de características descritas a continuación para su procesado.

4.3.1. Preprocesado de datos.

Antes de poder aplicar los distintos algoritmos de modelado de la silueta o movimiento sobre los datos, es necesario realizar un procesado previo o preprocesado de los mismos

4.3.1.1. Extracción de silueta/background substraction.

Para poder analizar correctamente el movimiento realizado por una persona y que la información del entorno capturada también en las secuencias no afecte a este

análisis, se ha desarrollado una función para la integración de diferentes técnicas para la eliminación de la información de fondo y extracción la silueta de la persona, y se han implementado algunas cuyo funcionamiento se basa en la información de profundidad.

function silhouette = silhouetteExtraction (depth, metadata,options)

La función recibe como argumentos la secuencia de mapas de profundidad *depth*, los metadatos devueltos por el sensor Kinect (o un array vacío en caso de no disponer de ellos), y la estructura SETTINGS.EXTRACT_SILHOUETTE contenida en la estructura de parámetros de configuración con el método para la extracción de la silueta y otras propiedades asociadas (umbral mínimo y máximo para la segmentación). Las opciones implementadas durante el PFC son las siguientes:

- 'None': esta opción no realiza ninguna operación sobre los datos de entrada. Elegir esta opción en aquellos casos en que los datos del dataset ya se encuentren segmentados.
- 'Manual': se aplican los umbrales de distancia especificados en el fichero de configuración, eliminando la información que se encuentre fuera del rango de distancia definido por los mismos. Método útil en aquellos casos en los que las actividades se realicen a una distancia conocida del sensor y los posibles objetos en la escena se encuentren a una distinta distinta.
- 'Auto': se aplica la máscara de segmentación devuelta por el sensor Kinect cuando se encuentra activo el tracking del esqueleto. Elegir esta opción en caso de no conocer la distancia de la persona al sensor y de disponer de los metadatos obtenidos mediante la captura de datos con el sensor Kinect.

4.3.1.2. Segmentación de datos.

La segunda tarea de preprocesado que se lleva a cabo en el sistema es la de segmentación de los datos, principalmente mediante la técnica de ventana deslizante, aunque resulta trivial la integración de nuevas técnicas.

function segments = segment(data, varargin)

La función segment() es la encargada de realizar las llamadas a las funciones que aplican las distintas técnicas de segmentado en función de lo indicado en el fichero de configuración. Por tanto a su entrada se tiene el array de datos *data* y los diferentes parámetros de configuración que puedan necesitar las funciones al cargo de la segmentación, como el tamaño de ventana y paso entre ventanas, tasa de frames, umbral de

energía, etc. La salida de la misma será un array de dimensionnes Mx2, donde M es el número de filas que se corresponde con el número segmentos calculados y para los que se indica el frame de comienzo y fin de los mismos. Es en esta función en la que se realiza la conversión de segundos a frames del tamaño de ventana y paso entre estas especificados por el usuario, donde $windowSize_{frames} = windowSize_{seconds} * frameRate_{fps}$ y de manera equivalente $stepSize_{frames} = stepSize_{seconds} * frameRate_{fps}$.

Debido a que los datos con los que trabaja el sistema utilizado como base son vectores que contienen los datos capturados mediante sensores de inercia, las funciones de segmentación incluídas, segmentSlidingWindow(), segmentEnergy() y segmentRest() se han modificado de manera que realicen correctamente su tarea cuando los datos de entrada son secuencias de imágenes.

4.3.2. Modelado de características y extracción de características.

Una vez realizadas las tareas de preprocesado descritas en la sección anterior, se da paso al procesado de los datos para la extracción de características. Dado que el objeto del sistema es el reconocimiento de acciones y actividades humanas, y estas se podrían resumir como un conjunto de movimientos del cuerpo, resulta intuitivo deducir que es necesario aplicar alguna técnica que modele o represente los movimientos realizados, de manera que posteriormente se puedan extraer una serie de características que aporten información relevante para el reconocimiento.

La función feature_extraction() es la encargada de controlar estas dos tareas, realizando las llamadas a las funciones específicas según la configuración que se haya introducido en el script de configuración.

function [features fDescr] = feature_extraction(data, segmentation, varargin)

A la entrada de la función se tienen los datos de la secuencia (previamente preprocesados) *data*, el array de segmentación *segmentation* en el que se indica el comienzo y fin de cada segmento, y una variable de parámetros variable, *varargin*, donde se incluye el método de modelado de la silueta o movimiento, el tipo de características que se desean extraer, y el esqueleto en caso de disponer de este dato. Como salida la función devolverá un array con un vector de características por cada segmento de datos procesado de la secuencia, y un array de tipo cell con el nombre o identificador de cada característica extraída (ej. {*first_Hu_moment*, *second_Hu_moment*, ...,}.

La estructura básica de esta función se ha mantenido respecto al sistema base, realizando las tareas de procesado para cada uno de los segmentos en que se han divido los datos. Sin embargo se han realizado modificaciones de acuerdo a las necesidades de este proyecto, ya que los tipos de datos que se procesan son totalmente distintos.

- Se ha incluido una estructura condicional *switch...case* extra para el control de las llamadas a las funciones que generarán el modelado o representación del movimiento de acuerdo a la configuración introducida por el usuario.

- Se han implementado e integrado diferentes funciones para el modelado o representación del movimiento (ver Sección 4.3.2.1).

- Se han descartado operaciones específicas al tipo de datos procesados por el sistema base.

- Se han implementado e integrado los Momentos de Hu como método para la extracción de características (ver Sección 4.3.2.2).

- Se han realizado, al igual que a lo largo del resto del sismtea, pequeñas modificaciones para permitir el uso de secuencias de imágenes y esqueleto como datos de entrada.

4.3.2.1. Modelado de la silueta y el movimiento.

Como previamente se ha mencionado, antes de poder extraer las características que definirán los distintos movimientos que identificarán las acciones y actividades, es necesario aplicar alguna técnica que permita modelar la postura del cuerpo o el movimiento realizado a lo largo de la secuencia. Por ello, y con el fin de poder realizar pruebas del funcionamiento del sistema desarrollado, se han implementado e integrado algunos de los métodos utilizados en los sistema mencionados en la Sección 2.4 que hacen uso de la información de profundidad para modelar los movimientos y postura del cuerpo.

- **MHI (Motion History Images).** Con el objetivo de centrar la mayor parte del esfuerzo en el desarrollo y adaptación del sistema para el reconocimiento de actividades, y debido a los buenos resultados obtenidos en este campo, se ha optado por utilizar MHI como método de partida para representar los movimientos realizados durante las secuencias. Como se mencionaba en la Sección 2.4, las imágenes MHI permiten representar la información espacial y temporal en una sóla imagen, dónde el valor de sus píxeles es un reflejo de la temporalidad de cada uno de los movimiento realizados. En la Figura 4.4 se muestra el pseudo-código para la obtención de los MHI.
  La función encarga del cálculo de la imágen MHI, devuelve una variable de tipo *cell* la imágen generada a partir de la secuencia de mapas de profundidad que

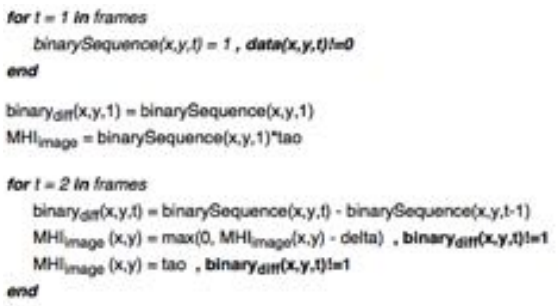

Figura 4.4: Pseudo-código del algoritmo para la obtención de MHI

recibe como argumento.

$$\{MHI\} = calculateMHI(depthData)$$

- **MHI + Average Depth Image.** Siguiendo el trabajo realizado por Megavannan et al. [19], se ha implementado un método que además del cálculo de la imagen MHI, obtiene también una imágen que representa el valor medio de profundidad de la secuencia y que de este modo captura la información del movimiento en la tercera dimensión.
  La función encargada de realizar esta representación, devuelve una variable de tipo *cell* con las imágenes motion history image y average depth image generadas a partir de los mapas de profundidad.

$$\{MHI, ADI\} = calculateMHIADI(depthData)$$

- **DMM (Depth Motion Maps).** Utilizando el trabajo desarrollado por Yang et al.[16], explicado con más detalle en la Sección 2.4, se ha implementado el sistema de proyección en los tres planos (frente, cenital y lateral) para la obtención de tres siluetas de las que extraer posteriormente las caracerísticas.
  La función encargada de realizar esta tarea, devuelve una variable de tipo *cell* con las tres imágenes correspondientes a las tres proyecciones generadas a partir de los mapas de profundidad.

$$\{DMM_{front}, DMM_{side}, DMM_{top}\} = calculateDMM(depthData)$$

- **SEI & SHI (Silhouette Energy Image y Silhouette History Image)**. Aunque similar a MHI, también se ha decidido implementar este método de

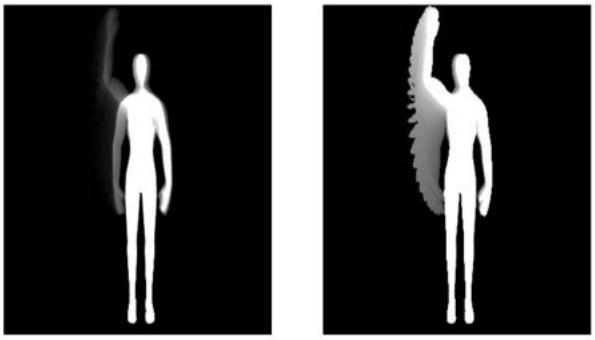

Figura 4.5: Izq. Imágen SEI; Dcha. Imágen SHI

representación del movimiento basado en la imágen SEI y SHI desarrollado por Ahmad et al. [34]. A partir de ambas imágenes calculadas en la función *calculateSEISHI()* se extraen las características como se verá en la Sección 4.3.2.2. La función encargada de realizar esta representación, devuelve una variable de tipo *cell* con las imágenes silhouette energy image y silhouette history image generadas a partir de los mapas de profundidad.

$$\{SEI, SHI\} = calculateSEISHI(depthData)$$

- ✧ SEI (Silhouette Energy Image): Ahmad et al. describen la imagen SEI como una imágen en la que la variación temporal del movimiento es representada por la figura del cuerpo. Asumiendo que $x_t = f(x, y, t)$ es la imágen con la silueta de la persona en el instante t, la imágen SEI se define mediante la Ecuación 4.1.

$$SEI = I(x, y) = \frac{1}{t_{fin} - t_{inicio}} \int_{t_{inicio}}^{t_{final}} x_t dt \tag{4.1}$$

- ✧ SHI (Silhouette History Image): esta imágen, además de capturar la variación de la figura a lo largo del tiempo que dura el movimiento, también captura la orientación global del movimiento para cualquier instante de tiempo t. Por cada frame, la información de la figura se incorporá a la imágen SHI con un valor de píxel actualizado de manera que el movimiento más reciente quede representado por los píxeles de mayor valor.

- **DMA (Depth Motion Appearance)**. El método desarrollado por DoHyung et al. es una representación volumétrica del movimiento 3-D que describe de manera general la forma y apariencia del movimiento realizado (ver Figura 4.6. Siendo $D(i,j,t)$ el valor de profundidad del píxel en la posición i,j del mapa de profundidad para el momento t, la imágen DMA queda descrita mediante la Ecuación 4.2

$$DMA(i,j,t) = \begin{cases} D(i,j,t) & , si\ DMA(i,j,t-1) \\ min(D(i,j,t), DMA(i,j,t-1)) & , resto \end{cases} \tag{4.2}$$

  Esta representación, sin embargo, únicamente codifica la forma del movimiento realizado. Para codificar la información temporal del mismo se emplean nuevamente las imágenes MHI. Por tanto la función encargada de generar esta representación, devuelve en una variable de tipo *cell* las dos imágenes de las que se extraeran las características, MHI y DMA.

$$\{DMA, MHI\} = calculateDMA(depthData)$$

Figura 4.6: Ejemplo de DMA para distintas acciones humanas [21]

4.3.2.2. Extracción de características.

Una vez obtenida la representación del movimiento para cada uno de los segmentos procesados mediante uno de los métodos descritos, se realiza la extracción de características que permitirán entrenar al sistema de manera que pueda reconocer movimientos que una vez codificados presenten unos vectores de características

$$I_1 = \eta_{20} + \eta_{02}$$
$$I_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$
$$I_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$
$$I_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$
$$I_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$
$$I_6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})$$
$$I_7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] - (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2].$$

Figura 4.7: Los siete momentos invariantes de Hu [22]

similares.

Dado que los métodos de representación seleccionados, y mencionados en la sección anterior 4.3.2.1, producen como resultado imágenes en las que se puede distinguir una figura, se ha optado por integrar en el sistema una funció para el cálculo de los Momentos de Hu descritos en [22] como las características que se van a extraer de las representaciones anteriores.

El momento de una imágen se puede definir como un cierto promedio ponderado de las intensidade de los píxeles de la imagen, o una función de los mismos que capturan una serie de características de la imagen. Los Momentos de Hu son un conjunto de momentos que se pueden extraer de una imágen, y que presentan las propiedades de ser invariantes a cambios de escala, rotación y traslación, por lo que suelen ser usados a menudo en tareas de reconocimiento de patrones y procesado de imágenes. La definición de estos momentos puede verse en la Figura 4.7.

4.4. Aprendizaje y reconocimiento.

El último módulo en que se divide el sistema desarrollado (ver Figura 4.8) es en el que se realizan las tareas de apredizaje automático, y de test del mismo a partir de un conjunto de secuencias de test o un flujo de datos que provenga directamente del sensor de profundidad, realizando en este caso el reconocimiento en tiempo real.

El proceso que siguen los datos desde la entrada a este módulo hasta el final en que se obtienen los resultados se puede resumir en los siguientes puntos:

1. Selección de las características más representativas a partir de uno de los algoritmos implementados e integrados en el sistema.

2. Generación de un modelo de predicción (o varios, dependiendo del clasificador utilizado) a partir de los vectores de características seleccionadas.

3. Selección de las características de los datos de test, en función de las seleccionadas previamente para los datos de training.

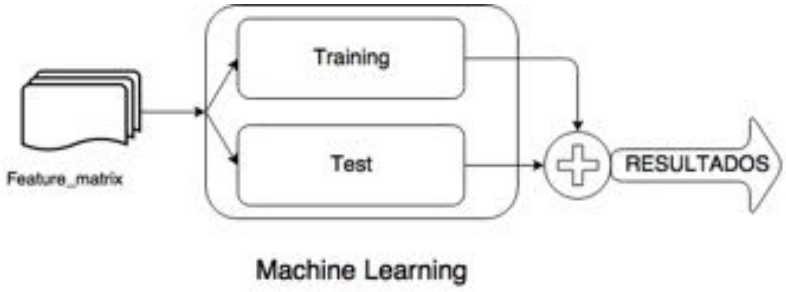

Figura 4.8: Módulo de Machine Learning

4. Clasificación de los datos de test utilizando el/los modelos generado previamente.

5. Interpretación del conjunto de scores obtenidos en el clasificador y decisión de la etiqueta de clase asociada a cada uno de los vectores de características de test.

En las próximas secciones se detallarán las funciones encargadas de realizar estas tareas y las técnicas de selección de características y clasificadores integrados en el sistema.

4.4.1. Entrenamiento.

La etapa de training es en la que los vectores de características generados en el módulo anterior de procesado de datos se utilizan para generar un modelo de predicción que posteriormente se utilizará para clasificar los nuevos datos que lleguen al sistema. El funcionamiento de este bloque queda representado por la función *run_ training()*.

function [model, selectedFeatures, tMatrix] = run_ training(trainingData, trainingLabels, SETTINGS)

Los parámetros de entrada a la misma constan de una matriz con los vectores de características obtenidos tras el procesado de los datos, el vector de etiquetas correspondientes a cada vector, y la estructura con los parámetros de configuración del sistema mediante los cuales se configuran las diferentes opciones de este módulo. Al final de su ejecución, se devuelve el modelo de predicción generado, un vector con los índices de las características seleccionadas y, para algunas técnicas de selección de características como PCA, la matriz de transformación.

En esta etapa de pueden distinguir dos tareas, las cuales se ejecutan de manera consecutiva y que se verán en los siguientes apartados de esta sección: selección de características y entrenamiento o generación del modelo de predicción.

4.4.1.1. Selección de características.

Como se ha explicado en la Sección 4.3, durante la etapa de procesado de datos se obtienen un conjunto de vectores de características que codifican una serie de propiedades de los datos procesados. Dependiendo del tipo de características extraídas y la naturaleza de los datos procesados, se pueden encontrar, tras realizar un análisis de las mismas, un conjunto de esas caraterísticas que concentran la mayor parte de la información.

Las técnicas de selección de características son por tanto aquellas que realizan este análisis sobre los conjuntos de características para encontrar aquellas que aporten con mayor poder discriminante y descartar aquellas que resulten reduntantes y por tanto no aporten ninguna información de utilidad. Esta tarea es especialmente útil en los casos en los que los vectores de características están formados por cientos o miles de elementos, ya que ayudan a reducir la cantidad de datos y de este modo facilitan la tarea de los clasificadores (generación del modelo de predicción y la posterior clasificación).

function [data, result, tMatrix] = feature_selection(data, labels, varargin)

La función del sistema encargada de realizar la selección de características es *feature_selection()*. Esta recibe la matriz de datos y vector de etiquetas, así como los parámetros para seleccionar la técnica a usar y su configuración, y devuelve la matriz de datos transformada y la matriz de transformación, para aquellas técnicas cuyo funcionamiento se basa en la tranformación a un nuevo espacio de características, (como es el caso de PCA) y un vector con los índices de las características seleccionadas para el resto de técnicas (SFS, SBS y mRMR).

En este trabajo además de las técnicas de selección de características SFS, SBS y mRMR, se ha integrado la selección de características mediante PCA y se ha preparado de manera que la integración de nuevas técnicas resulte sencilla para el usuario.

- **Sequential Features Selection**: Este método de selección de características se puede descomponer en dos componentes: una función criterio, a la que el método intenta minimizar para todos los posibles subconjuntos de datos de características; y un algoritmo de búsqueda secuencial, el cual tiene como objetivo la adición o eliminación de características de un subconjunto mientras se evalúa el criterio. Debido a que una comparación exhaustiva del criterio para los 2^n subconjuntos de un dataset con vectores de n características es típicamente inviable (dependiendo mayormente del tamaño de n), las búsquedas secuenciales únicamente se mueven en una sola dirección, aumentando o disminuyendo el

conjunto candidato/resultado. Dentro de este método, podemos distinguir dos variantes:

- ✧ SFS (Sequential Forward Selection): esta variante del método añade de manera secuencial características a un conjunto vacío hasta que la inclusión de nuevas características no hace disminuir el criterio.
- ✧ SBS (Sequential Backward Selection): al contrario que la variante anterior, este método parte del conjunto completo y va eliminando características hasta que la eliminación de un mayor número de características hace que el criterio aumente.

- **mRMR(Minimum-Redundancy Maximum-Relevancy)**: mRMR es un algoritmo frecuentemente utilizado como método de selección de características.
  La selección de características es uno de los problemas/dificultades que nos encontramos a la hora de realizar reconocimiento de patrones o "machine learning", y que consiste en la identificación dentro del conjunto de datos de trabajo aquellos subconjuntos que ofrecen información relevante/discriminativa. El problema es que a menudo estos subconjuntos incluyen información redundante.
  La ventaja de este algoritmo es que además de identificar los subconjuntos de datos más relevantes, es capaz de eliminar aquellos que aporten información redundante.

- **PCA (Principal Component Analysis)**: es un procedimiento estadístico que realiza una transformación ortogonal con el fin de convertir un conjunto de observaciones formado por variables posiblemente correladas en un conjunto de variables linealmente incorreladas, denominadas componentes principales. El número de estas componentes será menor o igual al número de variables originales.
  La transformación está definida de manera que la primera componente principal posea la mayor varianza posible, y las componentes sucesivas posean la mayor varianza posible bajo la restricción de ser ortogonales a las componentes que las preceden.
  Aunque Matlab cuenta con una función interna para el cálculo de PCA, se ha decidido integrar la función incluída en las librerías externas del sistema de referencia *[PCcoeff, PCvec] = pca(data)*. Esta función recibe a su entrada la matriz de vectores de características, y devuelve la matriz *PCcoeff* con los autovectores calculados a partir de la matriz de covarianza de los datos, y un vector *PCvec* con los valores de varianza para cada componente principal. Multiplicando la

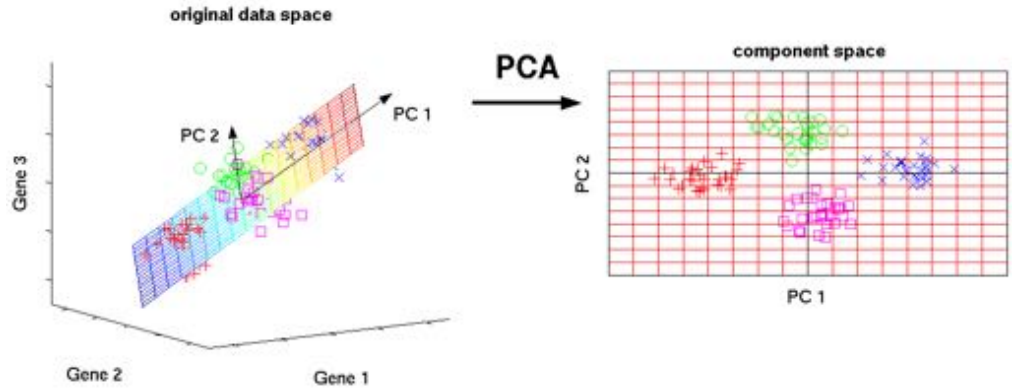

Figura 4.9: Representación del cambio de espacio realizado por PCA

matriz de datos por la matriz de autovectores obtenida se realiza la transformación de los vectores de características al espacio PCA (ver Figura 4.9), y las características transformadas seleccionadas son aquellas para las cuales la suma de sus varianzas supera el umbral definido por el usuario.

4.4.1.2. Generación del modelo de predicción.

Una vez realizada la selección de características, estas son procesadas por los distintos clasificadores integrados en el sistema (en función de la configuración indicada por el usuario) para la generación del modelo o modelos de predicción.

function [classifier trainingTime] = training(trainData, trainLabels, varargin)

La función cuya definición puede verse más arriba, es la encargada de realizar las llamadas a los distintos clasificadores, pasando como argumentos la matriz de vectores de características seleccionadas, el vector con las etiquetas de clase asociadas y las distintas opciones de configuración propias del clasificador seleccionado. El modelo de predicción generado, junto con la llamada a la función que realizará la clasificación correspondiente y otros datos que puedan ser necesarios a la hora de clasificar los datos, se agrupan en una estructura de datos (como la mostrada en la Figura 4.10)

Los principales clasificadores integrados en el proyecto son los siguientes (se presenta una comparativa en la Tabla 4.2):

- k-NN (k-Nearest Neighbors) [36]: el método k-NN es un método de clasificación supervisada, de tipo no paramétrico, en el que las muestras que llegan a su entrada son clasificadas como la clase más común entre las muestras de training

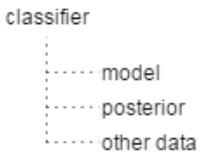

Figura 4.10: Estructura de datos del clasificador. Elementos: modelo de predicción, llamada a la función de clasificación y otros datos específicos

más cercanas. Por tanto, y como puede verse en la implementación de este método, el modelo generado de entrenamiento consiste directamente en la matriz de vectores de características y sus respectivas etiquetas.
A la hora de clasificar las nuevas muestras, la función *knnVoting(model, data, varargin)* calcula las distancias de entre los datos de entrada *data* y los de entrenamiento incluídos en el modelo *model* obtenido, y en función del valor de k elegido por el usuario durante la configuración del sistema, utilizará las k muestras más cercanas para determinar la clase de la muestra de entrada.

- Naive Bayes [40]: los clasificadores Naive Bayes con un conjunto de clasificadores probabilísticos sencillos, cuyo funcionamiento se basa en el teorema de Bayes y en la fuerte suposición de independencia entre características. Estos clasificadores asumen que el valor de una características en concreto es independiente del valor de cualquier otra característica, por lo que cada una de ellas contribuye de manera independiente a la probabilidad de que la muestra pertenezca a una clase en concreto, sin tener en cuenta ningún tipo de correlación entre características.

- SVM (Support Vector Machines) [37]: en machine learning, SVM son modelos de apredizaje supervisado con algoritmos asociados que analizan los datos para clasificación o análisis de regresión. Dado un conjunto de datos de ejemplo o entrenamiento pertenecientes a dos clases distintas, estos algoritmos buscan aquella función (recta, plano o hiperplano) que mejor separe los datos en estas dos categorías. De este modo, las nuevas muestras a su entrada se clasificarán como una u otra clase en función de la posición en la que caigan respecto de la función calculada.
  Generalmente, salvo ciertas implementaciones que permiten la clasificación multiclase, estos clasificadores son binarios, lo que implica que únicamente pueden diferencias entre dos clases distintas de datos. Debido a esta limitación, se ha implementado la función *trainSVM(trainLabels, trainData, options),* encargada de organizar los datos de manera que cada llamada a la función de Matlab *fitcsvm()*

(una llamada por cada clase del conjunto de datos) genere un modelo por cada clase respecto a las demás.
Suponiendo que se reciben N vectores de características, de los cuales M pertenecen a la clase para la cuál se va a generar el modelo, la función desarrollada selecciona M vectores de los N-M vectores restantes, los cuales se utilizan para generar el modelo SVM para esa clase en cuestión. Este proceso se realiza para todas las clases incluídas en el conjunto de datos.

- Decision trees [38]: este método de aprendizaje utiliza árboles de decisión como un modelo predictivo que mapea las observaciones acerca de un objeto a conclusiones acerca de su valor objetivo. El objetivo de este método es la creación de un modelo que permita predecir el valor de una variable objetivo basándose en múltiples variables de entrada. En este tipo de clasificadores, las hojas del árbol representan las etiquetas de clase y las ramas representan los conjuntos de características que conducen a esas etiquetas.
  Dependiendo del valor que pueda tomar la variable objetivo, se pueden distinguir dos tipos de árboles de decisión:

  - ✧ Classification Trees: en los casos en los que la variable objetivo puede tomar un conjunto finito de valores.
  - ✧ Regression Trees: en los casos en los que la variable objetivo puede tomar valores continuos, generalmente números reales.

  La función que se ha integrado en el sistema y que realiza este tipo de clasificación es la función propia de matlab *fitctree()*, que permite realizar una clasificación multiclase a partir de la matriz de datos y el vector de etiquetas.

- HMM (Hidden Markov Model) [23]: un modelo oculto de Markov es un modelo estadístico de Markov en el que se asume que el sistema que se está modelando es un proceso de Markov (ver Figura 4.11) con estados desconocidos (u ocultos). En un modelo oculto de Markov los estados de transición no son directamente observables, sin embargo las salidas sí que los son y estas son dependientes del estado. Cada estado tiene una distribución de probabilidad sobre los posibles valores de salida que puede tomar, de manera que a partir de una secuencia de salidas producidas por un HMM se puede extraer información acerca de la secuencia de estados.
  A pesar de existir en Matlab una implementación de estos clasificadores, se ha decidido utilizar la librería externa (escrita por Kevin Murphyy y distribuida bajo Licencia MIT) incluida en el sistema utilizado de referencia.

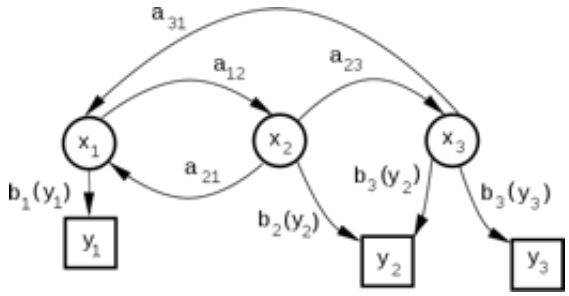

Figura 4.11: Ejemplo de proceso de Markov [23]

	Aprendizaje	Tipo	*Accuracy* general	Velocidad de clasificación	Interpretabilidad de los datos
Naive Bayes	Supervisado	Generativo	Bajo	Alta	Alta
kNN	Supervisado	Discriminativo	Bajo	Baja	Baja
SVM	Supervisado	Discriminativo	Alto	Alta	Baja
Decision Trees	Supervisado	Discriminativo	Medio	Alta	Alta
HMM	Supervisado	Generativo	Alto	Media	Baja

Tabla 4.2: Comparativa de clasificadores

4.4.2. Clasificación y Test.

Por último en el sistema de reconocimiento tenemos el bloque de test o clasificación, en el que los datos de test a reconocer se clasifican como una de las clases para las que se ha entrenado el sistema por medio del modelo generado durante la etapa de training.

[prediction] = run_prediction(model,selectedFeatures,testData,segmentation,SETTINGS)

La función del sistema que representa este bloque es *run_prediction()*, cuya definición puede verse más arriba. Esta función recibe el modelo predictivo generado en la etapa de training, el vector con los índices de las características seleccionadas, la matriz de datos de test e información de los segmentos en que se han segmentado, y los parámetros de configuración del sistema. Esta función realiza la llamada a la función *classification()*, la cuál utiliza el elemento incluído en el modelo, *model.posterior*, para llamar al clasificador correspondiente que devolverá un vector con los scores obtenidos por cada vector de características incluidos en *data*.

[scores testTime] = classification(model, data, varargin)

[prediction winnerScore] = decision(scoresTimeseries, varargin)

Finalmente estos scores se pasan como argumento a la función *decision()* que decide, en función del valor de los mismos, cuál es la clase a la que es más probable que pertenezcan los vectores de características.

	Característica	Implementación	Notas
Diseño del sistema	Estructura principal	Adaptada [20]	Adaptación y modificación del sistema base según necesidades
	Estructura modular	Propia	Ejecución independiente de cada módulo del sistema
	Gestión y formato de datos	Propia	Estándar de datos para ejecución independiente del dataset
	DEMO	Propia	Script para Demo en tiempo real.
Captura de datos	Captura Kinect	Implementada	Config. y gestión del sensor Kinect (add-on Matlab)
Preprocesado de datos	Bloque BS/ES	Propia	Bloque para la integración de técnicas de BS/ES
	Técnicas de ES	Propia	Técnicas de ES a partir de MP y MB
	Segmentación	Adaptada [20]	Segmentación por ventana deslizante y energía
Mod. del movimiento	MHI y ADI	Implementada [19]	Motion History Images y Average Depth Image
	SEI y SHI	Implementada [34]	Silhouette Energy y History Images
	DMM	Adaptada [16]	Depth Motion Maps
	DMA	Implementada [21]	Depth Motion Appearance
Feature Extraction	Momentos de Hu	Integrada [41]	Integración de la función implementada por [41]
Feature Selection	SFS, SBS y mRMR	Adaptado[20]	Adaptación de la técnica ya integrada
	PCA	Integrado [Matlab]	Integración de la func. Matlab para su funcionamiento
Clasificación	k-Nearest Neighbor	Adaptado [20]	Adaptación de la técnica ya integrada
	Naive Bayes	Integrado [Matlab]	Integración de la func. Matlab para su funcionamiento
	M-SVM	Adaptado [Matlab]	Adaptación de la func. Matlab para clasif. multiclase
	cTree	Integrado [Matlab]	Integración de la func. Matlab para su funcionamiento
	HMM	Adaptado [20]	Adaptación de la técnica ya integrada
Evaluación	Bhattachary	Implementado [42]	Implementación para análisis de descriptores
Utilidades	Adapt. de datasets	Implementado	Scripts para la adaptación de los dataset al sistema
	Present. de resultados	Implementado	Scripts autoadaptables para la repres. de resultados

BS: Background Substraction
ES: Extracción de silueta
MP: Mapas de profundidad
MB: Máscara binaria devuelta por la Kinect (previa config.)

Tabla 4.3: Resumen del proyecto desarrollado

[precission, recall, accuracy, fscore] = run_evaluation(prediction, testLabels, SETTINGS)

Con el fin de obtener cuál es la eficiencia de las diferentes configuraciones del sistema, se ha implementado la función run_evaluation(), la cuál toma como datos de entrada el vector con las etiquetas obtenidas por el clasificador y las que se conocen de los datos utilizados como test. A partir de las mismas se genera la matriz de confusión correspondiente, de la cuál se extraen los diferentes valores que pueden ser usados como métricas de evaluación: precision, recall, accuracy y fscore.

Capítulo 5

Experimentos y Resultados.

5.1. Introducción.

Este capítulo está centrado en la explicación y análisis de los experimentos llevados a cabo y la presentación de los resultados obtenidos con ellos.

El objetivo de dichos experimentos es evaluar el rendimiento y eficacia del sistema diseñado. Para ello, se han organizado un conjunto de configuraciones del sistema en las que se varían parámetros como el tamaño de ventana, tipo de representación de la acción y extracción de características y el clasificador utilizado, ...; que una vez ejecutadas sobre los datasets con los que se está trabajando nos permitan comprobar qué impacto tienen cada uno de los cambios introducidos y encontrar aquella configuración que optimice el proceso de reconocimiento para cada uno de ellos.

Puesto que cada uno de los datasets es diferente, representando diferentes actividades y bajo diferentes condiciones (frame rate, ángulo de grabación, distancia del objetivo), cada uno de los datasets presentará una configuración óptima propia. Los resultados de estas configuraciones se usarán posteriormente para realizar una comparativa con los resultados del Estado del Arte.

5.2. Diseño de Experimentos.

En esta sección se detallarán los diferentes elementos que conforman el conjunto de elementos:

- Datasets utilizados para la evaluación de las diferentes configuraciones del sistema.
- Métricas utilizadas como medio de evaluación.

- Configuraciones aplicadas al sistema bajo las que se probarán los diferentes algoritmos y datasets.

5.2.1. Datasets.

5.2.1.1. MSR Action 3D Dataset

El MSR Action 3D Dataset es uno de los datasets de referencia en el campo del reconocimiento de acciones 3D (con información de profundidad). Orientado al reconocimiento para la interacción con consolas y otros sistemas informáticos, las acciones recogidas en el dataset contienen múltiples movimientos de brazos, piernas, torso, y combinaciones de estos. Este dataset incluye 20 acciones (ver Figura 5.1), llevadas a cabo dos o tres veces, por 10 sujetos distintos orientados directamente al sensor. Estas acciones están organizadas en tres subsets, tal y como se muestra en la Tabla 5.1.

Action Set 1 (AS1)	**Action Set 2 (AS2)**	**Action Set 3 (AS3)**
Horizontal arm wave	High arm wave	High throw
Hammer	Hand catch	Forward kick
Forward punch	Draw x	Side kick
High throw	Draw tick	Jogging
Hand clap	Draw circle	Tennis swing
Bend	Two hand wave	Tennis serve
Tennis serve	Forward kick	Golf swing
Pickup & throw	Side boxing	Pickup & throw

Tabla 5.1: Subsets de acciones usados en los experimentos [17]

Los grupos de acciones AS1 y AS2 están diseñados para agrupar acciones con movimientos similares, mientras que AS3 tiene como objetivo agrupar las acciones más complejas del dataset. Como se explicará más adelante, para los experimentos de este proyecto se ha diseñado, a partir de las acciones recogidas en este dataset, un nuevo subset para probar más eficientemente la propiedades discriminativas del sistema.

Por cada repetición se proporciona tanto la secuencia de mapas de profundidad, con una resolución de 320x240 píxeles, como los puntos del esqueleto capturados por el sensor utilizado (similar al sensor Kinect de Microsoft). La información del esqueleto aportada se puede encontrar en dos formatos diferentes: uno con la información de los ejes (x, y) en coordenadas de imagen y la información del eje z en coordenadas reales; el otro con la información de los ejes *(x, y, z)* en coordenadas reales. Junto

Figura 5.1: MSRAction3D Dataset

con las coordenadas de cada punto del esqueleto, además se adjunta una medida de fiabilidad para dicho punto.

5.2.1.2. UTD-MHAD (UTD Multimodal Human Action Dataset)

Este dataset fue realizado por la Universidad de Dallas (Texas), como parte de la investigación [24] del reconocimiento de acciones a partir de la información conjunta obtenida mediante sensores de profundidad e inercia (ver Figura 5.2).

Para la elaboración del dataset se hizo uso del sensor Kinect de Microsoft, mediante el cuál se obtuvieron tres modalidades de datos: secuencia RGB (con resolución de 640x480 píxeles), secuencia de profundidad (con resolución 320x240 píxeles) y esqueleto. La cuarta modalidad de datos con la que cuenta el dataset la forman los datos obtenidos mediante los sensores de inercia. Los datos RGB se encuentran guardados en ficheros de video (.avi), mientras que los datos de profundidad, esqueleto e inercia se encuentran almacenados en ficheros .mat propios de Matlab.

UTD-MHAD Actions/Activities			
Swipe left (SL)	Draw X (DX)	Tennis swing (TSW)	Jog (JG)
Swipe right (SR)	Draw circle (clk.wise) (DCW)	Arm curl (ACU)	Walk (WK)
Wave (WV)	Draw circle (counter clk.wise) (DCCW)	Tennis serve (TS)	Sit->Stand (SST)
Clap (CL)	Draw triangle (DT)	Push (PS)	Stand->Sit (STS)
Throw (TW)	Bowling (BW)	Knock (KN)	Lunge (LU)
Arm cross (ACR)	Boxing (BX)	Catch (CT)	Squat (SQ)
Basketball shoot (BKS)	Baseball swing (BBS)	Pick&Throw (PT)	-

Tabla 5.2: 27 Acciones recogidas en el UTD-MHAD dataset [24]. Entre paréntesis se muestras las abreviaciones que se utilizarán para nombrar dichas clases a lo largo del capítulo.

Este dataset contiene secuencias de datos relativas a 27 acciones (ver Tabla 5.2), realizadas cuatro veces por 8 personas distintas (cuatro hombres y cuatro mujeres), dando un total de 864 secuencias de datos (para cada uno de los canales capturados). La versión final cuenta con 861 secuencias, debido a que se tuvieron que descartar tres secuencias debido a una corrupción de sus datos.

Los resultados obtenidos en [24] para el dataset completo, basándose únicamente en los datos obtenidos mediante el sensor Kinect, son de un 66,1 % de accuracy global.

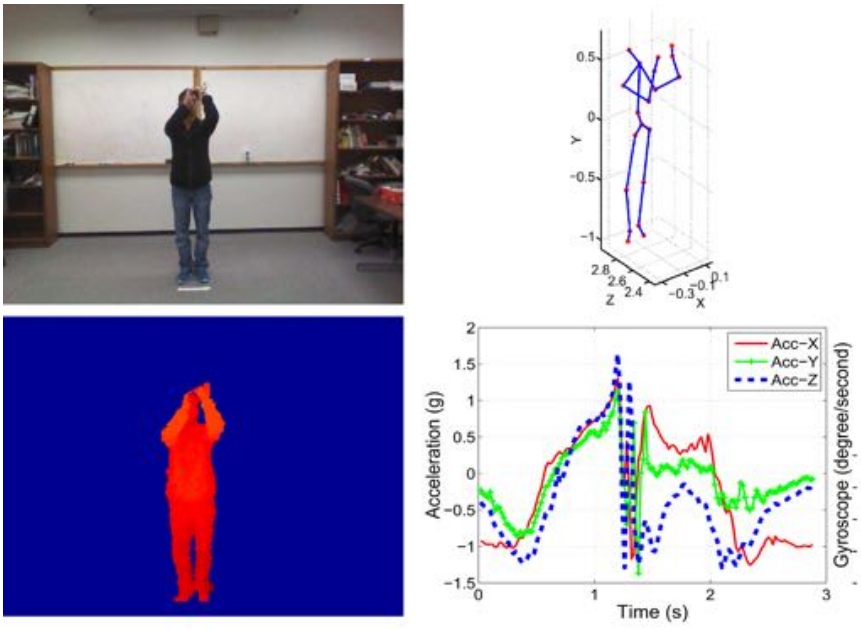

Figura 5.2: Modalidades de datos del dataset UTD-MHAD [24]: RGB, profundida, esqueleto y datos de inercia.

5.2.2. Evaluación y metricas.

La evaluación de los sistemas de reconocimiento es una tarea crucial ya que nos servirá como referencia del correcto funcionamiento y eficacia de los mismos, ya que este tipo de sistemas pueden, además de realizar una clasificación correcta dando lugar a aciertos positivos y negativos reales (*TP* y *TN*, *true positive* y *true negative*), pueden confundir las acciones que se están analizando y por ello clasificarlas de manera incorrecta, dando lugar a falsos positivos y negativos (FP y FN, *false positive* y *false negative*).

Las métricas de evaluación más utilizadas [20] en tareas de reconocimiento, e implementadas en este proyecto son:

- Matriz de confusión: matriz que resume cuántas instancias de las diferentes clases de actividades han sido clasificadas erróneamente. Generalmente las filas de la matriz indican para cada clase el número de instancias de la misma definidas por el *ground truth*, y en las columnas el número de instancias predichas para cada clase.
- *Accuracy*: definido como $\frac{TP+TN}{all}$, representa la tasa de aciertos general.
- Precision: definida como $\frac{TP}{TP+FP}$, repsenta el procentaje de aciertos positivos reales del total de positivos obtenidos.
- Recall: definido como $\frac{TP}{TP+FN}$, representa el procentaje de acierto positivos reales del total de positivos que deberían haberse detectado.
- F1 Score: definido como $\frac{2*precision*recall}{precision+recall}$, representa una medida balanceada entre la métrica de *precision* y *recall*.

5.2.3. Configuraciones.

Para la realización de las pruebas, se han seleccionado un conjunto de descriptores, tamaños de ventana de segmentación y clasificadores, y se han combinado de todas las manera posibles (ver Tabla 5.3) con el fin de obtener aquellas que mejores resultados ofrezcan.

- Descriptores (a continuación se muestras las abreviaciones que se utilizarán a lo largo del capítulo para designar cada descriptor y subdescriptor):
  - ✧ D1: HuMHI, Momentos de Hu (MHI)
  - ✧ D2: HuMHIADI

- ❏ D2_1: Momentos de Hu (MHI)
- ❏ D2_2: Momentos de Hu (ADI) (Average Depth Image)

✧ D3: HuSEISHI

- ❏ D3_1: Momentos de Hu (SEI) (Silhouette Energy Image)
- ❏ D3_2: Momentos de Hu (SHI) (Silhouette History Image)

✧ D4: HuMHIDMA

- ❏ D4_1: Momentos de Hu (MHI)
- ❏ D4_2: Momentos de Hu (DMA) (Depth Motion Appearance)

✧ D5: HuDMM

- ❏ D5_1: Momentos de Hu (DMM_{front})
- ❏ D5_2: Momentos de Hu (DMM_{top})
- ❏ D5_3: Momentos de Hu (DMM_{side})

- Tamaño de ventana (s):

✧ W1: 0.25s

✧ W2: 0.5s

✧ W3: 0.75s

✧ W4: 1.00s

✧ W5: 1.50s

✧ W6: 2.00s

- Claseficadores: kNN-Voting, SVM, Classification Trees

5.3. Resultados MSRAction3D.

5.3.1. Análisis de Características.

Como se ha mencionado previamente en la Sección 5.2.3, para la realización de las pruebas del sistema desarrollado se han utilizado un conjunto de descriptores, obtenidos cada uno mediante las diferentes técnicas de modelado del movimiento implementadas e integradas en el sistema, para probar su eficiencia a la hora de realizar el reconocimiento. Dado que cada uno de los descriptores ha sido obtenido mediante diferentes técnicas (MHI[19], DMM[16], etc), surge la necesidad de realizar

Configuración	Descriptor	Clasificador	Ventana	Configuración	Descriptor	Clasificador	Ventana
C1	D1	kNN	W1	C46	D3	SVM	W4
C2	D1	kNN	W2	C47	D3	SVM	W5
C3	D1	kNN	W3	C48	D3	SVM	W6
C4	D1	kNN	W4	C49	D3	cTree	W1
C5	D1	kNN	W5	C50	D3	cTree	W2
C6	D1	kNN	W6	C51	D3	cTree	W3
C7	D1	SVM	W1	C52	D3	cTree	W4
C8	D1	SVM	W2	C53	D3	cTree	W5
C9	D1	SVM	W3	C54	D3	cTree	W6
C10	D1	SVM	W4	C55	D4	kNN	W1
C11	D1	SVM	W5	C56	D4	kNN	W2
C12	D1	SVM	W6	C57	D4	kNN	W3
C13	D1	cTree	W1	C58	D4	kNN	W4
C14	D1	cTree	W2	C59	D4	kNN	W5
C15	D1	cTree	W3	C60	D4	kNN	W6
C16	D1	cTree	W4	C61	D4	SVM	W1
C17	D1	cTree	W5	C62	D4	SVM	W2
C18	D1	cTree	W6	C63	D4	SVM	W3
C19	D2	kNN	W1	C64	D4	SVM	W4
C20	D2	kNN	W2	C65	D4	SVM	W5
C21	D2	kNN	W3	C66	D4	SVM	W6
C22	D2	kNN	W4	C67	D4	cTree	W1
C23	D2	kNN	W5	C68	D4	cTree	W2
C24	D2	kNN	W6	C69	D4	cTree	W3
C25	D2	SVM	W1	C70	D4	cTree	W4
C26	D2	SVM	W2	C71	D4	cTree	W5
C27	D2	SVM	W3	C72	D4	cTree	W6
C28	D2	SVM	W4	C73	D5	kNN	W1
C29	D2	SVM	W5	C74	D5	kNN	W2
C30	D2	SVM	W6	C75	D5	kNN	W3
C31	D2	cTree	W1	C76	D5	kNN	W4
C32	D2	cTree	W2	C77	D5	kNN	W5
C33	D2	cTree	W3	C78	D5	kNN	W6
C34	D2	cTree	W4	C79	D5	SVM	W1
C35	D2	cTree	W5	C80	D5	SVM	W2
C36	D2	cTree	W6	C81	D5	SVM	W3
C37	D3	kNN	W1	C82	D5	SVM	W4
C38	D3	kNN	W2	C83	D5	SVM	W5
C39	D3	kNN	W3	C84	D5	SVM	W6
C40	D3	kNN	W4	C85	D5	cTree	W1
C41	D3	kNN	W5	C86	D5	cTree	W2
C42	D3	kNN	W6	C87	D5	cTree	W3
C43	D3	SVM	W1	C88	D5	cTree	W4
C44	D3	SVM	W2	C89	D5	cTree	W5
C45	D3	SVM	W3	C90	D5	cTree	W6

Tabla 5.3: Tabla de configuraciones, donde D1 es HuMHI, D2 es HuMHIADI, D3 es HuSEISHI, D4 es HuMHIDMA y D5 es HuDMM

un estudio previo en el que quede reflejado cuánta información aporta cada descriptor respecto de los demás.

Para realizar este estudio, se ha decidido hacer uso del coeficiente de Bhattacharyya[42] (definido por la Ec. 5.1) como medida de similitud entre descriptores.

$$C_{Bhatt}(p,q) = \sum_{x \in X} \sqrt{p(x) \cdot q(x)} \tag{5.1}$$

Con este objetivo en mente, se ha implementado el script *IntraClassFeatureAnalysis()* cuya función es calcular, como primera instancia, los histogramas de cada descriptor (o subdescriptor en aquellos casos en los que el descriptor está formado por la concatenación de dos o más vectores de características), y calcular para cada clase la distancia de Bhattacharyya entre los mismos. De este modo se obtienen un número de matrices igual al número de clases bajo estudio en los que se refleja una comparativa entre las distribuciones de los distintos descriptores. A continuación, y con el fin de facilitar la consulta de la información obtenida, se calcula una matriz en la que se refleja el valor medio, mínimo y máximo para cada una de las distancias obtenidas a los largo de las diferentes clases.

En la Tabla 5.4 se recogen los resultados del estudio realizado sobre los descriptores obtenidos para el dataset MSRAction3D. Como se mencionaba previamente, para cada subdescriptor se obtienen tres valores de distancia: mínimo, máximo y medio. Para facilitar su análisis, se ha aplicado un código de colores a los valores medios de cada subdescriptor en función de su similitud respecto al resto.

- Rojo: la similitud de la información aportada es igual o mayor al 70 %.
- Magenta: la similitud de la información aportada se encuentra entre el 40 % y el 70 %.
- Azul: la similitud de la información aportada es menor al 40 %.

A simple vista se puede observar como el descriptor que presenta una mayor diferencia en cuanto a información aportada respecto a los demás es el correspondiente a DMM (Depth Motion Maps)[16], por lo que la combinación de este descriptor con cualquiera de los otros daría lugar a un descriptor con una mayor información. Sin embargo, si esa diferencia de información aportada no resulta relevante para el reconocimiento, la combinación de esos descriptores podría dar lugar a un descriptor con una capacidad de reconocimiento similar y no mucho mayor a la del mejor de dichos descriptores. Es por ello que junto a esta información se requiere consultar los resultados finales obtenidos por cada descriptor para cada una de las configuraciones indicadas en la Tabla 5.3, y que se presentan en la Sección 5.3.2.

		D1	D2		D3		D4		D5		
		~D1	~D2_1	~D2_2	~D3_1	~D3_2	~D4_1	~D4_2	~D5_1	~D5_2	~D5_3
D1	~D1	1,000	1,000	0,056	0,038	0,081	1,000	0,092	0,009	0,011	0,023
		1,000	1,000	0,886	0,865	0,868	1,000	0,904	0,977	0,864	0,959
		1,000	**1,000**	**0,523**	**0,336**	**0,480**	**1,000**	**0,306**	**0,497**	**0,239**	**0,269**
D2	~D2_1	1,000	1,000	0,056	0,038	0,081	1,000	0,092	0,009	0,011	0,023
		1,000	1,000	0,886	0,865	0,868	1,000	0,904	0,977	0,864	0,959
		1,000	**1,000**	**0,523**	**0,336**	**0,480**	**1,000**	**0,306**	**0,497**	**0,239**	**0,269**
	~D2_2	0,056	0,056	1,000	0,223	0,264	0,056	0,212	0,015	0,024	0,000
		0,886	0,886	1,000	0,999	0,999	0,886	0,986	0,819	0,818	0,801
		0,523	**0,523**	**1,000**	**0,481**	**0,892**	**0,523**	**0,464**	**0,378**	**0,228**	**0,107**
D3	~D3_1	0,038	0,038	0,223	1,000	0,225	0,038	0,192	0,009	0,025	0,000
		0,865	0,865	0,999	1,000	0,997	0,865	0,990	0,807	0,806	0,108
		0,336	**0,336**	**0,481**	**1,000**	**0,489**	**0,336**	**0,601**	**0,270**	**0,382**	**0,064**
	~D3_2	0,081	0,081	0,264	0,225	1,000	0,081	0,210	0,022	0,022	0,015
		0,868	0,868	0,999	0,997	1,000	0,868	0,983	0,811	0,811	0,799
		0,480	**0,480**	**0,892**	**0,489**	**1,000**	**0,480**	**0,537**	**0,378**	**0,189**	**0,106**
D4	~D4_1	1,000	1,000	0,056	0,038	0,081	1,000	0,092	0,009	0,011	0,023
		1,000	1,000	0,886	0,865	0,868	1,000	0,904	0,977	0,864	0,959
		1,000	**1,000**	**0,523**	**0,336**	**0,480**	**1,000**	**0,306**	**0,497**	**0,239**	**0,269**
	~D4_2	0,092	0,092	0,212	0,192	0,210	0,092	1,000	0,013	0,009	0,021
		0,904	0,904	0,986	0,990	0,983	0,904	1,000	0,800	0,799	0,805
		0,306	**0,306**	**0,464**	**0,601**	**0,537**	**0,306**	**1,000**	**0,281**	**0,213**	**0,117**
D5	~D5_1	0,009	0,009	0,015	0,009	0,022	0,009	0,013	1,000	0,009	0,005
		0,977	0,977	0,819	0,807	0,811	0,977	0,800	1,000	0,998	0,998
		0,497	**0,497**	**0,378**	**0,270**	**0,378**	**0,497**	**0,281**	**1,000**	**0,308**	**0,281**
	~D5_2	0,011	0,011	0,024	0,025	0,022	0,011	0,009	0,009	1,000	0,005
		0,864	0,864	0,818	0,806	0,811	0,864	0,799	0,998	1,000	0,138
		0,239	**0,239**	**0,228**	**0,382**	**0,189**	**0,239**	**0,213**	**0,308**	**1,000**	**0,049**
	~D5_3	0,023	0,023	0,000	0,000	0,015	0,023	0,021	0,005	0,005	1,000
		0,959	0,959	0,801	0,108	0,799	0,959	0,805	0,998	0,138	1,000
		0,269	**0,269**	**0,107**	**0,064**	**0,106**	**0,269**	**0,117**	**0,281**	**0,049**	**1,000**

Tabla 5.4: Análisis de descriptores para el dataset MSRAction3D, donde se muestra para cada subdescriptor el valor mínimo, máximo y medio del coeficiente de Bhattacharyya entre clases. Los valores en rojo indican una alta similitud media entre subdescriptores, en magenta aquellos con una similitud media y en azul los que presentan una menor similitud.

5.3.2. Análisis de configuraciones.

En esta sección se presentan los resultados obtenidos para el dataset MSRAction3D tras la ejecución del sistema para cada una de las configuraciones indicadas en la Tabla 5.3.

Tal y como se mencionó en la Sección 5.2.1.1, las pruebas realizadas sobre este dataset por otros trabajos de investigación (ver Tabla 2.2), han sido sobre tres subconjuntos de las acciones que recoge y que se indican en la Tabla 5.1. Por ello, para la realización de las pruebas en este proyecto se ha divido el dataset de la misma forma, y además se han ejecutado pruebas para el conjunto completo y un subset con las acciones más dispares (en cuanto a movimientos se refiere) para mostrar como mejoran los resultados para acciones facilmente discernibles.

En la Figura 5.3 se muestran tres gráficos con los resultados de las noventa configuraciones posibles del sistema ejecutadas sobre el dataset completo. Como puede

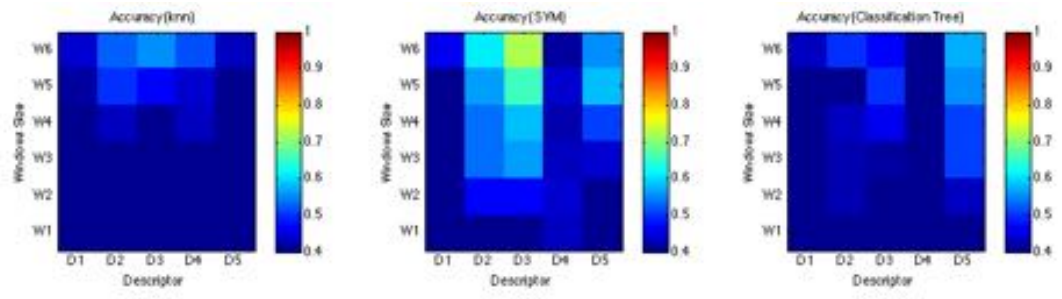

Figura 5.3: Resultados de *accuracy* para el dataset MSRAction3D completo.

verse, los resultados obtenidos para los clasificadores kNN y Decision Trees no son especialmente buenos, y es con el clasificador SVM con el que se obtienen mejores resultados (razón por la que ha sido utilizado en la mayoría de los trabajos consultados e indicados en la Tabla 2.2), en concreto para el descriptor formado por los momentos de Hu obtenidos a partir de las imágenes SEI y SHI (Silhouette Energy Image y Silhouette History Image) y para una ventana de tamaño de 2 segundos, se consigue un *accuracy* del 71 %.

En la Figura 5.4 puede verse un análisis más detallado sobre el efecto del tamaño de la ventana sobre los distintos clasificadores, y en la que se indica el descriptor con el que se obtiene el mejor resultado en cada caso. Como se puede comprobar, a medida que se aumenta el tamaño de ventana los resultados mejoran notablemente. La razón de esto se debe a que cuanto mayor es el tamaño de ventana escogido, mayor es el movimiento capturado en cada una de ellas y por tanto mayor es la información obtenida acerca del mismo.

Si se observan los resultados obtenidos para los subsets AS1, AS2 y AS3 (Figura 5.5), puede verse como los resultados mejoran de manera general respecto a los obtenidos para el dataset completo, al disminuir el número de clases y estas presentar más diferencias entre sí, y además sigue haciéndose evidente que los mejores resultados son los obtenidos para el clasificador SVM, y que de nuevo estos mejoran conforme se aumenta el tamaño de ventana.

Si se comparan estos resultados con los obtenidos para los experimentos realizados con el dataset completo, puede deducirse que este menor resultado de *accuracy* obtenido en el primer experimento con el dataset completo es debido al mayor número de clases incluidas y entre las que se tiene que realizar el reconocimiento.

Los resultados obtenidos para la mejor configuración, C48 (ver Tabla 5.3), quedan reflejados en la Tabla 5.5, donde también se muestran los mejores resultados obtenidos por uno de los trabajos vistos en el Capítulo 2. Como se puede observar, los resul-

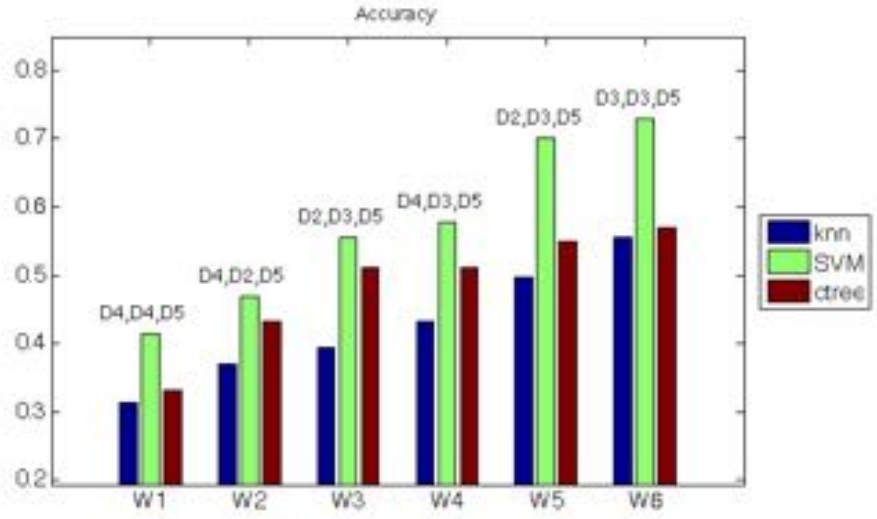

Figura 5.4: Análisis de los valores de *accuracy* para el dataset MSRAction3D

tados obtenidos mediante el sistema desarrollado y las técnicas implementadas son prometedores (superando en todos los casos el 70 % de *accuracy*), a pesar de quedarse generalmente por debajo de los resultados del Estado del Arte.

	Test	Estado del Arte			
		Bag of Points [17]	DMM-HOG[16]	HOJ3D [5]	DMA [21]
S1	74 %	72,9 %	96,2 %	88 %	92,4 %
S2	79 %	71,9 %	84,1 %	85,5 %	82,4 %
S3	81 %	79,2 %	94,6 %	64,5 %	95,6 %
Overall	**78 %**	**74,7 %**	**91,6 %**	**79 %**	**90,1 %**

Tabla 5.5: Mejores resultados obtenidos y del Estado del Arte

Para finalizar con los resultados obtenidos para este dataset, y como se comentó al inicio de esta sección, se seleccionaron un conjunto de 6 acciones (ver Tabla 5.6) del total de las recogidas en el dataset para verificar como un menor número de clases a reconocer y que presentan mayores diferencias entre sí resultan en unos mejores resultados.

Los resultados obtenidos para este nuevo subset se muestran en la Figura 5.6. Tal y como se esperaba, los resultados obtenidos han mejorado de manera general respecto a los experimentos realizados sobre los otros conjuntos de clases. Al igual que en los casos anteriores, los mejores resultados se obtienen para la configuración C48 (ver Tabla 5.3), en este caso alcanzando un *accuracy* del 89 % (resultado más cercano al obtenido por los trabajos del Estado del Arte).

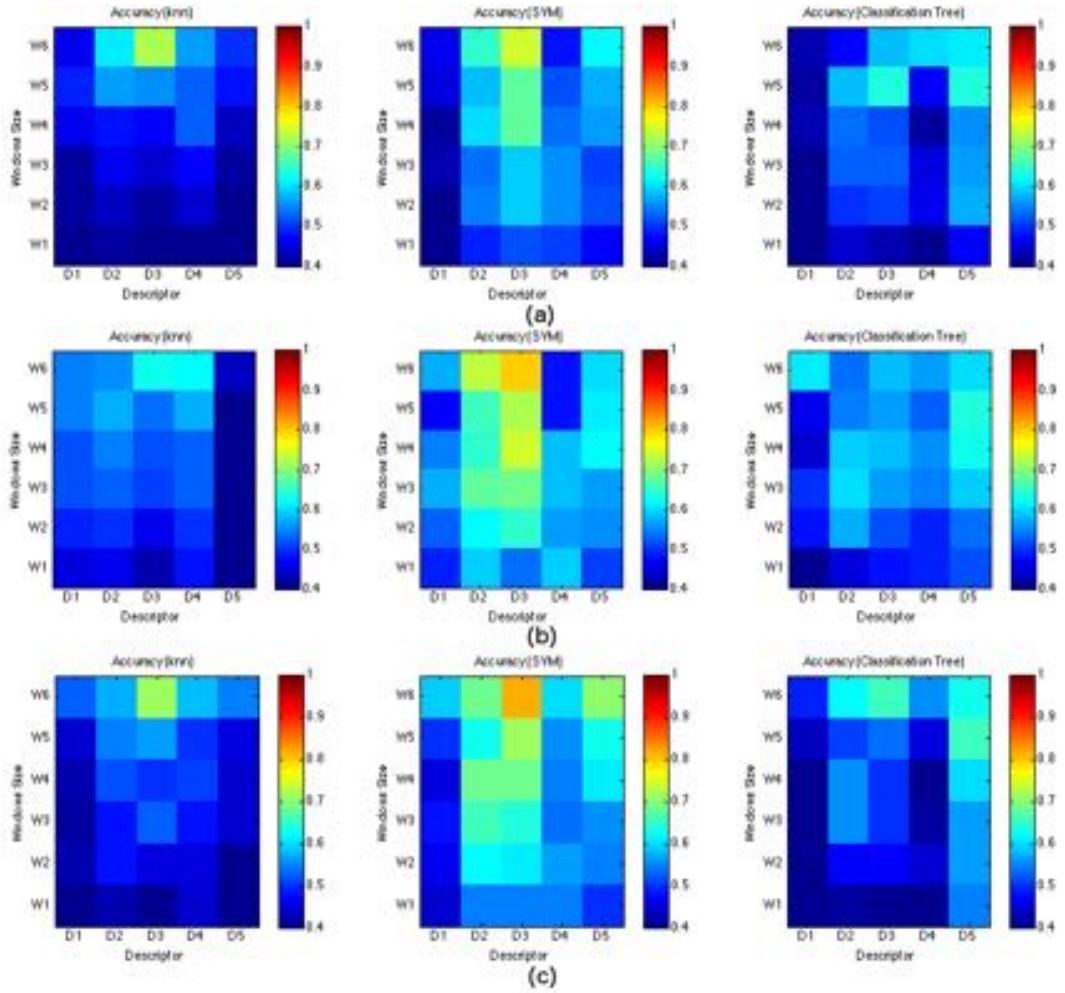

Figura 5.5: Resultados de *accuracy* para los subsets AS1 (a), AS2 (b) y AS3 (c) del dataset MSRAction3D.

5.4. Resultados UTD-MHAD.

A continuación se exponen los resultados obtenidos mediante los experimentos realizados para el dataset UTD-MHAD del que se habló en la Sección 5.2.1.2 de este capítulo.

5.4.1. Análisis de Características.

Al igual que con el dataset anterior, previamente a la realización de las pruebas mediante las diferentes configuraciones del sistema, se ha realizado un análisis de los descriptores con el fin de comprobar cuanto información de más aportan los distintos descriptores respecto a los demás. Los resultados obtenidos de dicho análisis quedan

Customs Actions Set (Custom)
High Arm Wave
Hand Catch
Two Hand Wave
Bend (Pick)
Lateral Kick
Golf Swing

Tabla 5.6: Clases incluidas en el subset seleccionado

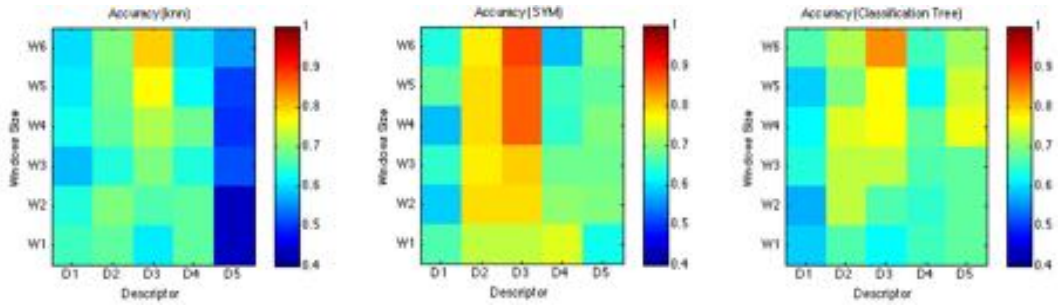

Figura 5.6: Resultados de *accuracy* para el subset Custom del dataset MSRAction3D.

recogidos en el Cuadro 5.4.1.

Como puede verse los resultados obtenidos en la Tabla 5.7, son muy parecidos a los obtenidos para el dataset MSRAction3D, siendo el descriptor formado por los Momentos de Hu[22] obtenidos a partir de DMM[16] aquél cuya información se diferencia más de la aportada por el resto de descriptores. Esta similitud entre los resultados obtenidos para ambos datasets puede deberse a que, a pesar de variar las clases recogidas de un dataset a otro, la información representada por cada descriptor cambia de manera similar de manera que la relación existente entre cada uno de los descriptores mantiene la misma distribución.

5.4.2. Análisis de configuraciones.

En esta sección del Capítulo 5 se exponen los resultados obtenidos para este dataset mediante las distintas configuraciones posibles del sistema. Dado que el trabajo [24] para el que se diseñó este dataset realiza las pruebas sobre el mismo utilizando todas las clases disponibles, la evaluación realizada en este proyecto se hará de igual manera sobre todas las clases recogidas en el dataset.

En la Figura 5.7 se muestran los resultados obtenidos para las distintas configuraciones del sistema, aplicadas sobre el dataset completo. De nuevo, y al que se observó en las pruebas realizadas sobre el otro dataset, los resultados en general no son muy

		D1	D2		D3		D4		D5		
		~D1	~D2_1	~D2_2	~D3_1	~D3_2	~D4_1	~D4_2	~D5_1	~D5_2	~D5_3
D1	~D1	1,000	1,000	0,021	0,009	0,021	1,000	0,065	0,022	0,010	0,010
		1,000	1,000	0,849	0,851	0,852	1,000	0,898	0,983	0,988	0,987
		1,000	**1,000**	**0,486**	**0,275**	**0,442**	**1,000**	**0,365**	**0,414**	**0,223**	**0,299**
D2	~D2_1	1,000	1,000	0,021	0,009	0,021	1,000	0,065	0,022	0,010	0,010
		1,000	1,000	0,849	0,851	0,852	1,000	0,898	0,983	0,988	0,987
		1,000	**1,000**	**0,486**	**0,275**	**0,442**	**1,000**	**0,365**	**0,414**	**0,223**	**0,299**
	~D2_2	0,021	0,021	1,000	0,194	0,187	0,021	0,201	0,023	0,019	0,024
		0,849	0,849	1,000	0,997	0,999	0,849	0,990	0,819	0,812	0,817
		0,486	**0,486**	**1,000**	**0,439**	**0,877**	**0,486**	**0,530**	**0,370**	**0,161**	**0,271**
D3	~D3_1	0,009	0,009	0,194	1,000	0,175	0,009	0,195	0,010	0,025	0,026
		0,851	0,851	0,997	1,000	0,998	0,851	0,993	0,819	0,795	0,799
		0,275	**0,275**	**0,439**	**1,000**	**0,475**	**0,275**	**0,649**	**0,227**	**0,427**	**0,321**
	~D3_2	0,021	0,021	0,187	0,175	1,000	0,021	0,125	0,022	0,020	0,025
		0,852	0,852	0,999	0,998	1,000	0,852	0,991	0,817	0,809	0,819
		0,442	**0,442**	**0,877**	**0,475**	**1,000**	**0,442**	**0,522**	**0,390**	**0,157**	**0,268**
D4	~D4_1	1,000	1,000	0,021	0,009	0,021	1,000	0,065	0,022	0,010	0,010
		1,000	1,000	0,849	0,851	0,852	1,000	0,898	0,983	0,988	0,987
		1,000	**1,000**	**0,486**	**0,275**	**0,442**	**1,000**	**0,365**	**0,414**	**0,223**	**0,299**
	~D4_2	0,065	0,065	0,201	0,195	0,125	0,065	1,000	0,013	0,009	0,026
		0,898	0,898	0,990	0,993	0,991	0,898	1,000	0,794	0,797	0,802
		0,365	**0,365**	**0,530**	**0,649**	**0,522**	**0,365**	**1,000**	**0,293**	**0,311**	**0,369**
D5	~D5_1	0,022	0,022	0,023	0,010	0,022	0,022	0,013	1,000	0,009	0,005
		0,983	0,983	0,819	0,819	0,817	0,983	0,794	1,000	0,998	0,997
		0,414	**0,414**	**0,370**	**0,227**	**0,390**	**0,414**	**0,293**	**1,000**	**0,139**	**0,205**
	~D5_2	0,010	0,010	0,019	0,025	0,020	0,010	0,009	0,009	1,000	0,005
		0,988	0,988	0,812	0,795	0,809	0,988	0,797	0,998	1,000	0,999
		0,223	**0,223**	**0,161**	**0,427**	**0,157**	**0,223**	**0,311**	**0,139**	**1,000**	**0,381**
	~D5_3	0,010	0,010	0,024	0,026	0,025	0,010	0,026	0,005	0,005	1,000
		0,987	0,987	0,817	0,799	0,819	0,987	0,802	0,997	0,999	1,000
		0,299	**0,299**	**0,271**	**0,321**	**0,268**	**0,299**	**0,369**	**0,205**	**0,381**	**1,000**

Tabla 5.7: Análisis de descriptores para el dataset MSRAction3D , donde se muestra para cada subdescriptor el valor mínimo, máximo y medio del coeficiente de Bhattacharyya entre clases. Los valores en rojo indican una alta similitud media entre subdescriptores, en magenta aquellos con una similitud media y en azul los que presentan una menor similitud.

buenos debido en parte a la cantidad de clases (un total de 27) entre las que es necesario reconocer. Los mejores resultados han sido los obtenido mediante el descriptor formado por los Momentos de Hu extraídos de las imágenes SEI y SHI (descritas en la Sección 4.3.2.1) y el clasificador SVM para los mayores tamaños de ventana probados, obteniendo unos valores de *accuracy* del 70 % y 76 % para los tamaños de ventana de 1 y 2 segundos respectivamente.

Como se puede ver en la Figura 5.8, y al igual que ocurría para las pruebas realizadas sobre el MSRAction3D dataset, a medida que se aumenta el tamaño de la ventana termporal de análisis, los resultados mejoran de manera general ya que la información contenida en cada una de las ventanas es mayor y por tanto mayores serán las diferencias entre las distintas clases. Además puede verse como esta mejora es más pronunciada en el caso de SVM, lo cuál puede deberse a que la mayor diferencia entre los descriptores para las distintas clases permite a este clasificador encontrar un

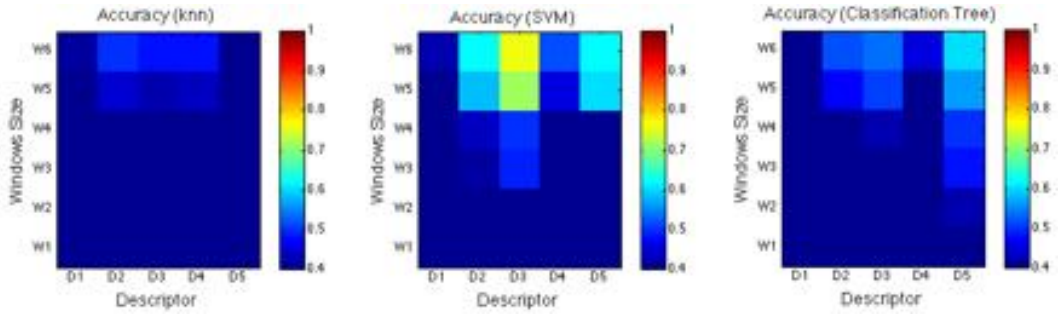

Figura 5.7: Resultados de *accuracy* para el dataset UTD-MHAD completo.

	Sistema desarrollado	**[24]**
Accuracy	**76.0 %**	66.1 %

Tabla 5.8: Resultados del sistema desarrollado y del Estado de Arte[24]

hiperplano (dado que la dimensionalidad de los datos es mayor que 3) que separe con mayor precisión los vectores de características que recibe para clasificar.

En el Cuadro 5.8 se muestra el mejor resultado obtenido por el sistema desarrollado en este proyecto para el dataset completo y el obtenido en el trabajo de Chen et al.[24]. Puede verse como los resultados obtenidos son un 10 % mejores que el trabajo previo realizado sobre este dataset para la configuración C48 (ver Tabla 5.3).

Para finalizar, y como se hizo para el anterior dataset, se ha creado un subset (ver Tabla 5.9) a partir de un conjunto de clases seleccionadas del conjunto total. Las actividades seleccionadas, relativas a actividades deportivas, son las más complejas aunque las que muestran una mayor diferencia entre sí. Los resultados obtenidos (y que pueden verse en la Figura 5.9) son incluso mejor de lo esperado, consiguiendo un valor máximo de *accuracy* del 95 % para la mejor configuración.

Analizando los mejores resultados obtenidos por los distintos descriptores utilizados en las pruebas (ver Figura 5.10), puede verse que para este subset de datos los resultados son muy positivos, consiguiendo un *accuracy* mayor al 70 % y llegando, en el mejor caso y como se comentaba anteriormente, al un 95 %.

UTD-MHAD Custom Subset	
Basket Shoot	Baseball Swing
Bowling	Tennis Swing
Front Boxing	Tennis Serve

Tabla 5.9: Clases recogidas en el subset seleccionado

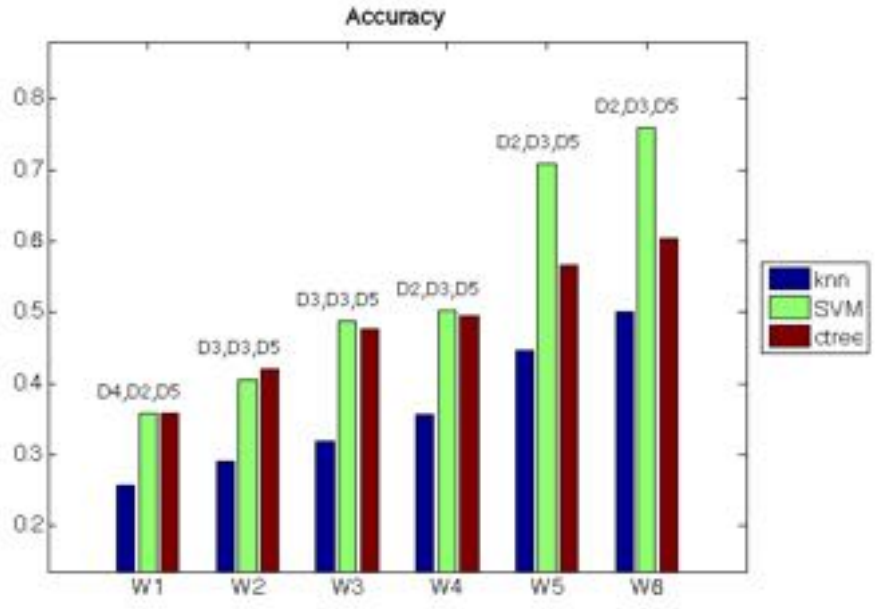

Figura 5.8: Análisis de los valores de *accuracy* en función del tamaño de ventana para el dataset UTD-MHAD

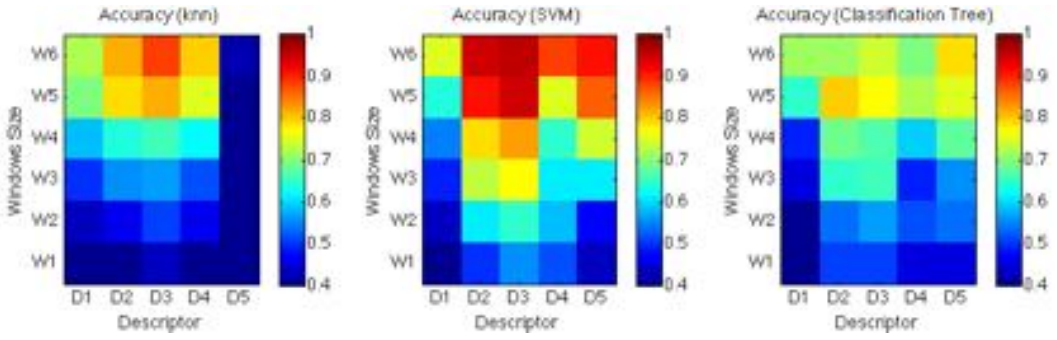

Figura 5.9: Resultados de *accuracy* para el subset Custom del dataset UTD-MHAD

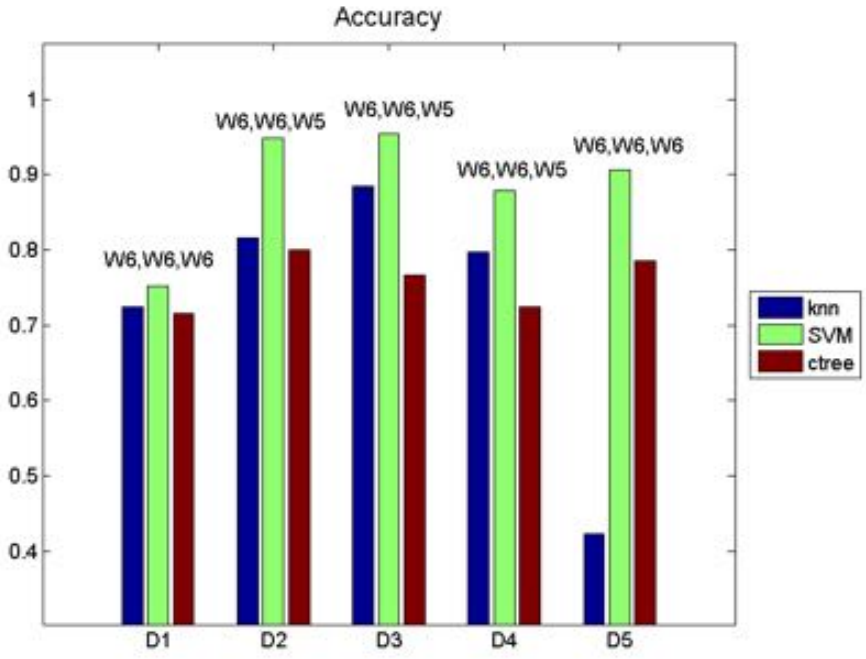

Figura 5.10: Análisis de los valores de *accuracy* en función del descriptor para el dataset UTD-MHAD

Capítulo 6

Conclusiones y trabajo futuro.

6.1. Conclusiones.

El objetivo principal de este proyecto era el desarrollo de un sistema de reconocimiento capaz de hacer uso de la información aportada por el sensor Kinect de Microsoft (u otros sensores de profundidad), siguiendo las líneas de un diseño modular que permitiese la integración de diferentes algoritmos de aprendizaje y extracción de características de manera sencilla. Este se ha logrado de forma satisfactoria como ha podido verse a lo largo de este documento, y en especial en los Capítulos 4 y 5 donde se exponen el desarrollo del sistema y los experimentos y resultados obtenidos respectivamente.

Como se ha visto a lo largo del proyecto, se ha conseguido modificar y adaptar de forma satisfactoria el sistema utilizado como base de este proyecto [20] con el fin de poder hacer uso de toda la información ofrecida por este nuevo grupo de sensores de imágen. El diseño propuesto y desarrollado, consistente en un conjunto de tres módulos funcionales, permite no sólamente la fácil integración de los diferentes elementos que los componen, sino que además permite su ejecución de manera independiente al realizarse un guardado de los datos obtenidos tras la ejecución de cada uno de ellos. La ventaja de este diseño es que permite aislar la tarea más costosa computacionalmente y que más tiempo requiere, como el modelado del movimiento y extracción de las características, en un mismo periodo temporal al poder configurar el sistema para generar estos datos para los diferentes algoritmos integrados de manera consecutiva. De esta forma, una vez obtenidos los datos de ese módulo en cuestión, se pueden realizar todas las pruebas para las distinas configuraciones del sistema de manera rápida sin tener que realizar de nuevo el procesado de los datos.

Para poder evaluar le funcionamiento del sistema desarrollado, se han imple-

mentado e integrado diversos algoritmos de modelado del movimiento: MHI (*Motion History Image*)[19], MHI+ADI (*Motion History Image* y *Average Depth Image* respectivamente)[19], SEI+SHI (*Silhouette Energy Image* y *Silhouette History Image* respectivamente)[34], DMM (*Depth Motion Maps*)[16] y DMA (*Depth Motion Appearance*)[21]; como método de extracción de características los *Momentos de Hu*[22], por sus propiedades de invarianza antes cambios de escala, rotación y traslación, y para las tareas de entrenamiento del sistema y el posterior reconocimiento, se han integrado un conjunto de clasificadores: kNN (*k-Nearest Neighbors*), SVM (*Support Vector Machine*) y cTrees (*Classification Trees*) frecuentemente utilizados para tareas de reconocimiento de patrones. Además se han seleccionado dos datasets, MSRAction3D[17] y UTD-MHAD[24], sobre los que realizar las pruebas para las distintas configuraciones posibles del sistema, y se han implementado los scripts correspondientes para convertir los datos que conforman estos datasets al definido en la Sección 4.2.2 para poder hacer uso de ellos en el sistema.

Tras las pruebas realizadas para las diferentes configuraciones del sistema y los resultados obtenidos y expuesto en el Capítulo 5, se puede concluir que de los clasificadores seleccionados, el que mejores resultados obtiene de manera general es SVM (Support Vector Machine), razón por la cuál es uno de los clasificadores más utilizados en este campo, y como ha podido verse, en los trabajos consultados y expuestos en el Cuadro 2.2. Además se ha podido comprobar como mejoran los resultados a medida que se aumenta el tamaño de ventanas, debido al mayor movimiento capturado y procesado para cada una de ellas obteniendo así un conjunto de características más discriminantes a la hora de realizar las tareas de entrenamiento y reconocimiento.

En vista de los resultados obtenidos, se puede concluir que el sistema desarrollado puede servir como un sistema base sólido sobre el que trabajar y realizar nuevas pruebas relacionadas con el reconocimiento de acciones y actividades del ser humano.

6.2. Trabajo futuro.

Tras la realización de este proyecto surgen varias líneas de trabajo a seguir en el futuro. Estas serían: la elaborción de un conjuntos de datos propio, la integración de diferentes algoritmos de background substraction, y de modelado del movimiento y extracción de características para imágenes RGB y esqueleto, y la mejora del sistema para incluir la interacción con objetos del entorno a la hora de realizar el reconocimiento de la actividad.

Para el Workshop organizado por el VPULab (Video Processing and Understanding Lab) se realizó un pequeño dataset para el reconocimiento de tres clases dife-

rentes, aunque a pesar del bajo número de clases, los resultados obtenidos resultaban irregulares debido al escaso número de secuencias de entrenamiento grabadas. Sería por ello interesante realizar como trabajo futuro la realización de un dataset que contenga información sobre múltiples clases (como los analizados en este proyecto) y que además incluya secuencias con distintos grados de complejidad (oclusiones parciales, velocidad de ejecución de los movimientos) para evaluar el sistema bajo condiciones más estrictas.

Como se ha visto a lo largo del proyecto, los algoritmos de modelado del movimiento únicamente se basan en la información de los mapas de profundidad aportados por el sensor Kinect y otros similares. El sistema deasrrollado permite el uso de imágenes RGB y esqueleto como fuente de datos, por lo que una mejora para el sistema sería la integración de distintos algoritmos que hagan uso de este tipo de información y la realización de pruebas sobre los mismos.

Por último, una mejora que se podría introducir para la mejora de las capacidades del sistema desarrollado, es el reconocimiento de interacciones persona-objeto y/o persona-persona. Hasta ahora, el funcionamiento del sistema se basa exclusivamente en el análisis de los moviemiento realizados por la persona, por lo que el rango de actividades a reconocer se ve reducido, por lo que la introducción del reconocimiento de interacciones enriquecería las posibilidades y capacidades del sistema.

Bibliografía

[1] M. M. Sardsehmukh, M. T. Kolte, P. N. Chatur, and D. S. Chaudhari, "3-d dataset for human activity recognition in video surveillance," in *Wireless Computing and Networking (GCWCN), 2014 IEEE Global Conference on*, pp. 75–78, Dec 2014.

[2] Dubey, Rachit and Ni, Bingbing and Moulin, Pierre, *A Depth Camera Based Fall Recognition System for the Elderly*, pp. 106–113. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012.

[3] O. Oreifej and Z. Liu, "Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pp. 716–723, June 2013.

[4] F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal and R. Bajcsy, "Berkeley mhad: A comprehensive multimodal human action database. in proceedings of the ieee workshop on applications on computer vision (wacv)," 2013.

[5] L. Xia, C.-C. Chen, and J. K. Aggarwal, ""view invariant human action recognition using histograms of 3d joints", the 2nd international workshop on human activity understanding from 3d data (hau3d) in conjunction with ieee cvpr 2012, providence, ri, june 2012.," 2012.

[6] Miles Hansard, Seungkyu Lee, Ouk Choi, Radu Horaud, "Time of flight cameras: Principles, methods, and applications. springer, pp.95, 2012, springerbriefs in computer science, isbn 978-1-4471-4658-2," 2012.

[7] L. Li, "Introduction to time-of-flight camera (rev. b) - technical white paper," tech. rep., Texas Instruments, 2014.

[8] I. I. D. S. GmbH, "Obtaining depth information from stereo images," 2012.

[9] "Photonic frontiers: Gesture recognition: Lasers bring gesture recognition to the home," 2011.

[10] B. Peasley and S. Birchfield, "Real-time obstacle detection and avoidance in the presence of specular surfaces using an active 3d sensor," in *Robot Vision (WORV), 2013 IEEE Workshop on*, pp. 197–202, Jan 2013.

[11] Khongma, A. and Ruchanurucks, M. and Koanantakool, T. and Phatrapornnant, T. and Koike, Y. and Rakprayoon, P., *Kinect Quality Enhancement for Triangular Mesh Reconstruction with a Medical Image Application*, pp. 15–32. Cham: Springer International Publishing, 2014.

[12] Z. Liu, "Illustrations of the joint positions."

[13] M. D. Network, "Depth space range."

[14] L. P. Serrano, "Proyecto fin de carrera: Robótica, software y telecomunicaciones.," 2010.

[15] F. A. C. Lucero, "Detección de robo/abandono de objetos en interiores utilizando cámaras de profundidad," Diciembre 2012.

[16] Yang, Xiaodong and Zhang, Chenyang and Tian, YingLi, "Recognizing actions using depth motion maps-based histograms of oriented gradients," in *Proceedings of the 20th ACM International Conference on Multimedia*, MM '12, (New York, NY, USA), pp. 1057–1060, ACM, 2012.

[17] W. Li, Z. Zhang, and Z. Liu, "Action recognition based on a bag of 3d points," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops*, pp. 9–14, June 2010.

[18] W. Li, Z. Zhang, and Z. Liu, "Expandable data-driven graphical modeling of human actions based on salient postures," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, pp. 1499–1510, Nov 2008.

[19] V. Megavannan *et al.*, "Human action recognition using depth maps," 2012.

[20] Bulling, Andreas and Blanke, Ulf and Schiele, Bernt, "A tutorial on human activity recognition using body-worn inertial sensors," *ACM Comput. Surv.*, vol. 46, pp. 33:1–33:33, Jan. 2014.

[21] DoHyung Kim, Woo-han Yun, Ho-Sub Yoon, Jaehong Kim, "Action recogntion with depth maps using hog descriptors of multi-view motion appearance and history," 2014.

[22] M.-K. Hu, "Visual pattern recognition by moment invariants," *IRE Transactions on Information Theory*, vol. 8, pp. 179–187, February 1962.

[23] L. Piyathilaka and S. Kodagoda, "Gaussian mixture based hmm for human daily activity recognition using 3d skeleton features," in *2013 IEEE 8th Conference on Industrial Electronics and Applications (ICIEA)*, pp. 567–572, June 2013.

[24] C. Chen, R. Jafari, and N. Kehtarnavaz, "Utd-mhad: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor," in *Image Processing (ICIP), 2015 IEEE International Conference on*, pp. 168–172, IEEE, 2015.

[25] Ye, Mao and Zhang, Qing and Wang, Liang and Zhu, Jiejie and Yang, Ruigang and Gall, Juergen, *Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications: Dagstuhl 2012 Seminar on Time-of-Flight Imaging and GCPR 2013 Workshop on Imaging New Modalities*, ch. A Survey on Human Motion Analysis from Depth Data, pp. 149–187. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.

[26] A. Veeraraghavan, R. Chellappa, and A. K. Roy-Chowdhury, "The function space of an activity," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 1, pp. 959–968, 2006.

[27] Zhao, Xin and Li, Xue and Pang, Chaoyi and Zhu, Xiaofeng and Sheng, Quan Z., "Online human gesture recognition from motion data streams," in *Proceedings of the 21st ACM International Conference on Multimedia*, MM '13, (New York, NY, USA), pp. 23–32, ACM, 2013.

[28] R. Poppe, "A survey on vision-based human action recognition," *Image and Vision Computing*, vol. 28, no. 6, pp. 976 – 990, 2010.

[29] T. B. Moeslund, A. Hilton, and V. Krüger, "A survey of advances in vision-based human motion capture and analysis," *Computer Vision and Image Understanding*, vol. 104, no. 2 - 3, pp. 90 – 126, 2006. Special Issue on Modeling People: Vision-based understanding of a person's shape, appearance, movement and behaviour.

[30] D. Weinland, R. Ronfard, and E. Boyer, "A survey of vision-based methods for action representation, segmentation and recognition," *Computer Vision and Image Understanding*, vol. 115, no. 2, pp. 224 – 241, 2011.

[31] A. F. Bobick and J. W. Davis, "The recognition of human movement using temporal templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, pp. 257–267, Mar 2001.

[32] Vieira, Antonio W. and Nascimento, Erickson R. and Oliveira, Gabriel L. and Liu, Zicheng and Campos, Mario F. M., *"Space-Time Occupancy Patterns for 3D Action Recognition from Depth Map Sequencesïn Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications: 17th Iberoamerican Congress, CIARP 2012, Buenos Aires, Argentina, September 3-6, 2012. Proceedings*, ch. STOP: Space-Time Occupancy Patterns for 3D Action Recognition from Depth Map Sequences, pp. 252–259. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012.

[33] X. Yang and Y. Tian, ".eigenjoints-based action recognition using naive- bayes-nearest-neighbor", in computer vision and pattern recognition workshops (cvprw), 2012 ieee computer society conference on. ieee, 2012, pp. 14 - 19.," 2012.

[34] M. Ahmad and M. Z. Hossain, "Sei and shi representations for human movement recognition," in *Computer and Information Technology, 2008. ICCIT 2008. 11th International Conference on*, pp. 521–526, Dec 2008.

[35] Y. Benezeth, P.-M. Jodoin, B. Emile, H. Laurent, and C. Rosenberger, "Comparative study of background subtraction algorithms," *Journal of Electronic Imaging*, vol. 19, July 2010.

[36] M.-L. Zhang and Z.-H. Zhou, "A k-nearest neighbor based algorithm for multi-label classification," in *2005 IEEE International Conference on Granular Computing*, vol. 2, pp. 718–721 Vol. 2, July 2005.

[37] F. F. Chamasemani and Y. P. Singh, "Multi-class support vector machine (svm) classifiers – an application in hypothyroid detection and classification," in *Bio-Inspired Computing: Theories and Applications (BIC-TA), 2011 Sixth International Conference on*, pp. 351–356, Sept 2011.

[38] A. Navada, A. N. Ansari, S. Patil, and B. A. Sonkamble, "Overview of use of decision tree algorithms in machine learning," in *Control and System Graduate Research Colloquium (ICSGRC), 2011 IEEE*, pp. 37–42, June 2011.

[39] S. Na, L. Xumin, and G. Yong, "Research on k-means clustering algorithm: An improved k-means clustering algorithm," in *Intelligent Information Technology and Security Informatics (IITSI), 2010 Third International Symposium on*, pp. 63–67, April 2010.

[40] M. Martinez-Arroyo and L. E. Sucar, "Learning an optimal naive bayes classifier," in *18th International Conference on Pattern Recognition (ICPR'06)*, vol. 3, pp. 1236–1239, 2006.

[41] I. Badami, "Hu moments of order 3 (mathworks file exchange)."

[42] T. Kailath, "The divergence and bhattacharyya distance measures in signal selection," *IEEE Transactions on Communication Technology*, vol. 15, pp. 52–60, February 1967.

Apéndice A

Script y parámetros de configuración

La configuración de los diferentes módulos se realiza desde un único script que se ejecuta al comienzo de cada módulo del sistema y en el que se recojen los diferentes parámetros configurables del sistema. Todos estos parámetros se almacenan dentro de una única estructura, SETTINGS, la cuál es cargada en el sistema una vez ejecutada.

- Captura y parámetros de datos: se recogen los parámetros de configuración del sensor Kinect para la captura de datos, y la selección del dataset (junto con sus propiedades propias que deberán ser indicadas). Los datos relativos a la Kinect se almacenan en una estructura dentro de la global, SETTINGS.KINECT. Ver Tablas A.1 y A.2.

Parámetros de configuración del sensor Kinect	
CameraElevationAngle	Controla el ángulo del sensor respecto del suelo.Por defecto se mantiene el último valor configurado. '[-27 : 27]'
FramesPerTrigger	Número de frames capturado por cada Trigger ejecutado
BodyPosture	Configura la postura para la que se van a extraer los puntos del esqueleto. 'Standing': devuelve los 20 puntos del esqueleto. 'Seated': devuelve 10 puntos del esqueleto (2-11).
DepthMode	Configura el rango de profundidad de los mapas de profundidad. 'Default': 50 - 400 cm 'Near': 40 - 300 cm
TrackingMode	Indica el modo de tracking. 'Skeleton': realiza tracking de todo el esqueleto 'Position': realiza tracking únicamente del punto de la cadera 'Off': desactiva el tracking de esqueleto (valor por defecto)
SkeletonsToTrack	Indica el Skeleton Tracking ID devuelto como parte de los metadatos.

Tabla A.1: Parámetros de config. Kinect utilizados (ver Anexo X con una descripción de todos los parámetros configurables)

Selección de dataset y propiedades	
Dataset	Selección del dataset a utilizar para el entrenamiento del sistema. Formato de nombre utilizado para los datasets: 'Dataset_X', siendo X un número.
SamplingRate	Número de frames por segundo. Este valor es configurable para cada dataset, y deberá ser introducido por el usuario. Utilizado posteriormente para la segmentación mediante ventana deslizante
ClassLabels	Cell con las etiquetas para cada clase contemplada en el dataset: {'Actividad_1', 'Actividad_2', ..., 'Actividad_N'}. Es tarea del usuario completar este dato.
SubsetName	Nombre del subset de clases a utilizar: 'All', 'S1',..., 'SM'. Definibles por el usuario.
Subset	Vector que define las clases incluidas en el dataset seleccionado. 'All'-> [1:#ClassLabels], 'S1'-> [aM, aN,..., aZ], 'S2'-> [...]. Definibles por el usuario.

Tabla A.2: Parámetros relativos al dataset utilizado

- Procesado de datos. Aquí se engloban todos los parámetros configurables relativos a la segmentación y substracción de fondo, y al modelado de la postura y movimiento y de la posterior tarea de extracción de características. Ver Tablas A.3 y A.4.

Parámetros de segmentación y substracción de fondo	
SegmentationTechnique	Selección de la técnica de segmentación a utilizar. Valores posibles: 'SlidingWindow', 'Energy', 'Rest'
SegmentationOptions.Threshold	Valor umbral para la segmentación por energía
SilhouetteExtraction.Type	Modo de extracción de silueta. 'None', 'Manual' y 'Auto'
SilhouetteExtraction.minThreshold	Valor de profundidad mínimo hasta el que se descarta la información
SilhouetteExtraction.maxThreshold	Valor de profundidad mínimo hasta el que se descarta la información
W_Size_Second	Tamaño de ventana, expresado en segundos, para la segmentación por ventana deslizante
W_Step_Second	Duración del paso entre ventanas, expresada en segundos, para la segmentación por ventana deslizante

Tabla A.3: Parámetros relativos a la segmentación y substracción de fondo

Parámetros para el modelado del movimiento y extracción de características	
ActionRepresentation	Técnica o método de modelado de la silueta o movimiento. Valores posibles: 'MHI', 'SEI&SHI', 'DMM', 'DMAMHI'
FeatureType	Tipo de características extraídas. Valores posibles: 'HU'

Tabla A.4: Parámetros relativos al modelado del movimiento y extracción de características

- Machine Learning y selección de características (ver Tabla A.5).

Parámetros de selección de características y clasificador	
FeatureSelection	Técnica de selección de características. Valores posibles: 'SFS', 'SBS', 'mRMR' y 'PCA'
FeatureSelectionOptions	# de características que se quieren seleccionar
Classifier	Tipo de clasificador. Valores posibles: 'NaiveBayes', knnVoting', 'SVM', 'cTree', cHMM'

Tabla A.5: Parámetros relativos a la selección de características y el clasificador

- Otras opciones configurables: además de los parámetros que se pueden configurar para cada uno de los módulos del sistema, en este script también se incluyen las rutas a las librerías externas que se utilizan o se quieran incluir, así como las rutas de los directorios en los que se almacenan los datos de entrada y salida del sistema.

Apéndice B

Captura y datos de entrada.

En este apéndice se detallan las funciones implementadas mediante las cuáles se realiza la captura de datos de la Kinect, y la organización de directorios definida y los formatos que deben presentar los distintos ficheros de datos para su uso con el sistema desarrollado.

B.1. `Rec.m`

Script principal para realizar la captura de secuencias de video. Con el objetivo de facilitar la tarea de organización de los datos capturados, el script cuenta con tres parámetros configurables por el usuario: *actividad*, *sujeto* y *ejecución*. Estos parámetros, que identifican la actividad que se está capturando, el sujeto que la está realizando y el número de veces que la ha realiazdo, son utilizados para dar un nombre identificativo a los ficheros de manera que resulte fácil identificarlos.
Al comienzo de su ejecución se realiza la carga de la estructura SETTINGS desde el fichero de configuración explicado en el Apéndice A, de la cuál se extrae la estructura que contiene los parámetros de configuración especificos para la Kinect, SETTINGS.KINECT, y que se pasa como parámetro a la hora de llamar a la función de captura. Esta última devuelve una variable por cada uno de los tres flujos de datos capturados por la Kinect: profundidad, color y metadatos (esqueleto, máscara de segmentación). Por último cada uno de estos datos se almacena en un fichero independiente cuyo nombre consiste en los tres parámetros antes mencionados, y el tipo de información que albergan.

B.2. `captura_kinect.m`

Esta función recibe como parámetro la estructura con la información necesaria para configurar el sensor Kinect, y se encarga de realizar la configuración del mismo y de la captura de datos. Una vez capturados estos son devuelto al script maestro desde donde fue llamada.

$$[color, depth, mdepth] = captura\_kinect(SETTINGS.KINECT)$$

El proceso que realiza la función para la captura de datos es la siguiente:

1. Creación de dos objetos de entrada video, uno por cada uno de los sensores con los que cuenta la Kinect (RGB y profundidad).

$$\begin{aligned} obj_{color} &= videoinput('kinect', 1) \\ obj_{depth} &= videoinput('kinect', 2) \end{aligned}$$

2. Acceso a las propiedades del objeto de video y configuración de las mismas.

$$\begin{aligned} src_{sensor} &= getselectedsource(obj_{sensor}) \\ src_{sensor}.propertyName &= SETTINGS.KINECT.PROPERTYNAME \end{aligned}$$

3. Configuración del trigger para la captura e inicialización de los sensores. Se ha configurado para capturar un frame por cada trigger y que este se ejecute el número de frames que se quieran capturar (conociendo la tasa de frames, la duración de la secuencia puede calcularse facilmente como $t = frames/fps$).
   Para poder conseguir secuencias de color y profundidad sincronizadas es necesario configura el trigger a modo manual, dado que el modo inmediato inicia la captura de ambos flujos de datos de manera secuencial y por tanto se produce cierto lag entre ambos.

$$\begin{aligned} obj_{sensor}.FramesPerTrigger &= 1 \\ obj_{sensor}.TriggerRepeat &= frames \end{aligned}$$

$$triggerconfig([obj_{color}\ obj_{depth}], 'manual')$$
$$start([obj_{color}\ obj_{depth}])$$

4. Captura de datos. El trigger y obtención de datos se realiza dentro de un bucle,

el cuál se ejcuta tantas veces como frames se hayan configurado para su captura.

$$trigger([obj_{color}\ obj_{depth}])$$

$$[sensorData, sensorTimeData, sensorMetadata] = getdata(obj_{sensor})$$

B.3. Estructura de directorios y formato de ficheros.

Con el fin de mantener organizados los datos de cada dataset, se ha organizado una estructura de directorios como la mostrada en la Figura 4.2. Por cada dataset que se quiera utilizar con el sistema, deberá existir una carpeta dentro del directorio *Data*, que contenga otros tres directorios: *RAW*, *labeled* y *features*.

- Directorio *'RAW'*: en esta carpeta se almacenan los datos brutos del dataset en cuestión. Cada flujo de datos, color, profundidad y esqueleto deberá tener su propio fichero .mat con los datos correspondientes y además se incluye un fichero *anot.dat* con la información de anotación para todas las secuencias de video. A continuación se muestra la estructura que deben presentar los nombres de estos ficheros y los datos que contienen.

  - ✧ Información de color:
    - ❏ Nombre: aXX_sYY_eZZ_color.mat (siendo XX el identificador de la clase, YY la persona que realiza la acción y ZZ la repetición de la misma).
    - ❏ Datos: cell de datos con un número de elementos igual al número de frames de la secuencia, y en el que cada elemento es una array de dimensiones m-n-3 que define las componentes roja, azul y verde para cada píxel de la imagen.
  - ✧ Información de profundidad:
    - ❏ Nombre del fichero: aXX_sYY_eZZ_depth.mat
    - ❏ Datos: array de dimensiones m-n-frames.
  - ✧ Esqueleto:
    - ❏ Nombre del fichero: aXX_sYY_eZZ_skeleton.mat
    - ❏ Datos: array de dimensiones j-3-frames (donde j es el número de puntos devuelto por Kinect, pudiendo ser 10 o 20 en función de la configuración del parámetro BodyPosture mencionado en el Apéndice A).

- ✧ Metadata: por lo general otros datasets no facilitan esta estructura de datos devuelta por la Kinect, únicamente los datos de esqueleto extraídos de la misma. Sin embargo, el sistema está preparado para leer los ficheros con este dato que además incluye la máscara de segmentación utilizada para la substracción de fondo.
  - ❏ Nombre del fichero: aXX_sYY_eZZ_mdepth.mat
  - ❏ Datos: estructura de dimensiones 10-1 en la que se engloban todos los datos relativos al esqueleto y e información de segmentación.
- ✧ Fichero de anotación (anot.dat): en este fichero se especifíca, en una línea para cada secuencia de video y separados por comas, el frame de inicio y fin de la actividad dentro de la secuencia, y la etiqueta que correspondiente a dicha actividad.

- Directorio *'labeled'*: una vez almacenados los datos en el directorio *RAW* correspondiente, se tiene que ejecutar el script *labelTrainingData.mat*. Este script lee la información almacenada en el fichero de anotación y genera para cada secuencia un vector con las etiquetas para los frames de la misma. Además, carga todos los datos existentes para cada secuencia y los almacena junto con el nuevo vector generado en un mismo fichero en este directorio. De este modo, cada secuencia tendrá un único archivo en el que se incluirán todos los datos.
- Directorio *'features'*: en este directorio se almacenan los ficheros con todas las características generadas durante la etapa de extracción de características. Por cada secuencia se tendrá un único fichero .mat, el cuál contendrá una estructura cuyos elementos serán estructuras en las que se almacenan los vectores de características y la configuración utilizada para generarlos: tamaño de ventana, step entre ventanas, técnica de modelado de la silueta/movimiento y de extracción de características. Por cada fichero y para cada configuración, únicamente exisitirá una structura que la contenga. La Figura 4.3 reprenta un ejemplo esquemático de la estructura interna de los archivos de features.

Apéndice C

Presupuesto

1. **Ejecución Material**
   - Compra de ordenador personal (Software incluido) ..... 2.000€
   - Alquiler de impresora láser durante 6 meses ..... 260 €
   - Material de oficina ..... 150 €
   - Total de ejecución material ..... 2.400 €
2. **Gastos generales**
   - 16 % sobre Ejecución Material ..... 352 €
3. **Beneficio Industrial**
   - 6 % sobre Ejecución Material ..... 132 €
4. **Honorarios Proyecto**
   - 1800 horas a 15 € / hora ..... 27.000 €
5. **Material fungible**
   - Gastos de impresión ..... 280 €
   - Encuadernación ..... 200 €
6. **Subtotal del presupuesto**
   - Subtotal Presupuesto ..... 32.774 €

7. **I.V.A. aplicable**

   - 21 % Subtotal Presupuesto ..... 6.882,5 €

8. **Total presupuesto**

   - Total Presupuesto ..... 39.656,5 €

Madrid, junio 2014

El Ingeniero Jefe de Proyecto

Fdo.: Borja Olmo Esteban

Ingeniero de Telecomunicación

Apéndice D

Pliego de condiciones

Este documento contiene las condiciones legales que guiarán la realización, en este proyecto, de un sistema de reconocimiento de actividades utilizando información de color y profundidad. En lo que sigue, se supondrá que el proyecto ha sido encargado por una empresa cliente a una empresa consultora con la finalidad de realizar dicho entorno. Dicha empresa ha debido desarrollar una línea de investigación con objeto de elaborar el proyecto. Esta línea de investigación, junto con el posterior desarrollo de los programas está amparada por las condiciones particulares del siguiente pliego. Supuesto que la utilización industrial de los métodos recogidos en el presente proyecto ha sido decidida por parte de la empresa cliente o de otras, la obra a realizar se regulará por las siguientes:

Condiciones generales

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

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

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

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

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

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

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

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

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

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

11. Cuando el contratista, con autorización del Ingeniero Director de obras, emplee materiales de calidad más elevada o de mayores dimensiones de lo estipulado

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

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

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

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

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

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

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

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

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

20. Durante la realización de la obra, se girarán visitas de inspección por personal facultativo de la empresa cliente, para hacer las comprobaciones que se crean

oportunas. Es obligación del contratista, la conservación de la obra ya ejecutada hasta la recepción de la misma, por lo que el deterioro parcial o total de ella, aunque sea por agentes atmosféricos u otras causas, deberá ser reparado o reconstruido por su cuenta.

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

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

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

Condiciones particulares

La empresa consultora, que ha desarrollado el presente proyecto, lo entregará a la empresa cliente bajo las condiciones generales ya formuladas, debiendo añadirse las siguientes condiciones particulares:

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

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

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

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

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

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

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

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

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

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

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

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