Reconocimiento anticipado de gestos

por

Yared Sabinas Figueroa

Tesis sometida como requisito parcial para tener el grado de

MAESTRO EN CIENCIAS EN LA ESPECIALIDAD DE CIENCIAS COMPUTACIONALES en el Instituto Nacional de Astrofísica, Óptica y Electrónica

Supervisada por:

Dr. Eduardo Morales Manzanares

Dr. Hugo Jair Escalante Balderas

Agosto 2013, Tonantzintla, Puebla

Resumen

El reconocimiento anticipado de gestos es el problema de reconocer los gestos en sus partes iniciales, por lo que el reconocimiento debe hacerse sin contar con toda la información sobre el gesto que se quiere reconocer. Entre otras aplicaciones, el reconocimiento anticipado puede ser usado para compensar el retraso de sistemas interactivos basados en gestos. En esta tesis proponemos un nuevo enfoque para el reconocimiento temprano de gestos de cuerpo completo que está basado en *dynamic time warping* (DTW) y que no necesita una fase compleja de entrenamiento. Nuestro método está basado en la comparación de secuencias de tiempo, mismas que se obtienen de los gestos conocidos y desconocidos. Nuestro método arroja una respuesta antes de que el gesto desconocido sea terminado y es capaz de funcionar bajo el esquema *one-shoot*, *i.e.*, sólo necesita un ejemplo de cada gesto para poder clasificar los gestos entrantes. Realizamos experimentos con una base de datos que nosotros construimos y con la base de datos *MSR-Action3D* propuesta en otros trabajos. Los resultados muestran que nuestro clasificador es capaz de reconocer gestos en tiempo real con sólo el ≈50 % de la información, perdiendo un máximo de 13 % de precisión con respecto a los resultados obtenidos con nuestro método de clasificación sin anticipación (en promedio).

Palabras clave: Clasificación anticipada de gestos, DTW, aprendizaje *one-shot*, reconocimiento de gestos, Kinect.

Abstract

Early gesture recognition consists in recognizing gestures at their beginning, using incomplete information. Among other applications, these methods can be used to compensate for the delay of gesture-based interactive systems. In this thesis, we propose a new approach for early recognition of full-body gestures based on dynamic time warping (DTW) that uses a single example from each category. Our method is based on the comparison between time sequences obtained from known and unknown gestures. The classifier provides a response before the unknown gesture finishes and can work under the one-shot scheme, *i.e.* only need one example of each gesture to recognize the unknown gestures. We performed experiments in the MSR-Actions3D benchmark and another data set that we built. Results show that, in average, the classifier is capable of recognizing gestures with ≈50% of the information in real time, losing only 13% of accuracy with respect to using all of the information.

Keywords: Early gesture recognition, DTW, one-shot learning, Kinect.

Agradecimientos

Quiero agradecer profundamente a mis padres por creer en mi, por estar siempre a mi lado apoyándome, por inculcarme que una persona grande es aquella que cree en sus capacidades y lucha por lo que quiere, pero sobre todo por darme todo ese cariño que me ayudó siempre a salir adelante. Quiero dedicarles esta tesis que sin duda es el resumen de mi esfuerzo durante estos años en los que estuve alejada de casa. Me complace enormemente saber que una etapa más de mi vida concluye aquí de manera muy exitosa y que de ninguna manera es el final de mi largo recorrido, al contrario, quedan muchas satisfacciones por saborear juntos. Nunca olviden que los amo.

A mis asesores por llevarme por el camino correcto, por todo el esfuerzo que invirtieron para sacar este trabajo adelante, por tenerme la paciencia que me tuvieron y por incluso preocuparse de mi bienestar. Sus consejos me ayudaron a mejorar mi manera de trabajar; en verdad he aprendido mucho de ustedes, y las experiencias adquiridas me servirán mucho en el futuro.

También quiero agradecer a mis compañeros que me ayudaron en la grabación de las bases de datos para evaluar mi método, fueron largas y tediosas horas de dar patadas y puñetazos, pero créanme cuando les digo que no fue en vano, pues aquí están los resultados de ese esfuerzo. Gracias Sr. Merlo Galeazzi, por estar siempre ahí las veces que lo necesité, es usted un gran amigo; vale la pena haber viajado de tan lejos para conocer a tan maravillosa persona. Gracias también a Miguel Ángel Valencia Serrano, Iván Garrido Márquez y Benjamín Rubalcava que también participaron en la grabación de los gestos.

Finalmente quiero agradecer con muchísimo cariño a *Ma vie*, estuvo conmigo desvelándose noches enteras, dándome ánimos para no rendirme en los momentos más difíciles, por hacerme sentir que soy una persona especial y por recordarme cada vez que lo necesité qué tan valiosa soy; por hacerme creer en mis virtudes y por ser desde siempre una persona que ocupa un lugar en mi corazón.

A todos ustedes muchas gracias, sin ustedes esto no hubiera sido posible.

Contenido

Índice de figuras

Indice de tablas

Glosario

$\vec{\mathbf{A}}, \vec{\mathbf{B}}, \vec{\mathbf{C}}, \vec{\mathbf{D}}$: Vectores.

$|\vec{\mathbf{A}}|, |\vec{\mathbf{B}}|, |\vec{\mathbf{C}}|$: Módulos de vectores.

H: Matriz.

e_c: Vector unitario.

a, b: Arreglo de valores.

$\overline{a}$: Media de los valores de x.

$\overline{b}$: Media de los valores de y.

dm: Dimensión de los datos.

dm^-: Dimensión reducida de los datos después de aplicar análisis de componentes principales.

pc_1, pc_2: Componentes principales.

eig: Eigenvectores.

Q, R, S: Secuencias de tiempo.

M: Longitud de la secuencia de tiempo Q.

N: Longitud de la secuencia de tiempo R.

C: Matriz de costo local para DTW.

$c_{m,n}$: Una celda de la matriz de costo local.

$P = p_1, \ldots, p_K$: Mejor camino de alineamiento DTW.

K: Longitud del mejor camino obtenido con DTW.

p_k: Uno de los puntos que conforman el mejor camino de alineamiento obtenido con DTW.

CA: Matriz de costo acumulado para DTW.

$ca_{i,j}$: Una de las celdas de la matriz de costo acumulado.

$\{u, r, t\}$: Cuadro del torso, donde u, r y t son componentes principales que son representados como vectores.

LE: Articulación del codo izquierdo.

LH: Articulación de la mano izquierda.

N: Articulación del cuello.

RS: Articulación del hombro derecho.

LS: Articulación del hombro izquierdo.

RHi: Articulación de la cadera derecha.

LHi: Articulación de la cadera izquierda.

T: Articulación del torso.

R_e: Radio; la distancia entre LE desde el origen de $\{u, r, t\}$.

θ_e: Zenit; el ángulo existente entre u y $\overrightarrow{(LS,\ LE)}$.

φ_e: Azimut, el ángulo existente entre r y $\overrightarrow{(LS,\ LE_p)}$.

LE_p: Es la proyección de LE en el plano cuya normal es u.

$\overrightarrow{(LS,\ LE)}$: Segmento de extremidad que va desde el hombro izquierdo hasta el codo izquierdo.

$\{b, r_p, t\}$: Sistema de coordenadas esféricas calculado a partir de las uniones de segundo grado.

S: El plano formado por r_p y t calculados con las segundas uniones.

R_h: La distancia desde el origen del nuevo sistema de coordenadas $\{b, r_p, t\}$.

θ_h: El ángulo entre b y $\left(\overrightarrow{LE,\ LH}\right)$.

φ_h: El ángulo entre r_p y la proyección de R_h en el plano S cuya normal es b, y $\left(\overrightarrow{LE,\ LH_p}\right)$.

LH_p: La proyección de LH en el plano S.

LE_p: La proyección de LE en el plano .

$\overrightarrow{r}, \overrightarrow{u}, \overrightarrow{t}$: Vectores unitarios que conforman el sistema de coordenadas esféricas obtenido a partir del esqueleto virtual.

$\mathcal{D}$: Diccionario de gestos conocidos.

R: Número de clases de gestos que contiene el diccionario de gestos $\mathcal{D}$.

G_r: *Gesto conocido.*

G_T *Gesto nuevo.*

T_r: Número de cuadros de longitud que tiene el *gesto conocido.*

f_r: Cuadro del *gesto conocido* G_r.

f_T: Cuadro del *gesto nuevo* G_T.

$A_{r,i}$: Secuencias de tiempo del *gesto conocido* G_r.

A_T: Secuencias de tiempo del *gesto nuevo* G_T.

t_{it}: Índice de la última iteración del método.

$dist_r$: Costo o distancia entre las secuencias del *gesto conocido* G_r y el *gesto nuevo* G_T.

$DTWacc$: Método DTW acumulado.

μ_m: Umbral de movimiento, si las extremidades no se mueven lo suficiente no se toman en cuenta.

P_r: Probabilidad de que el gesto G_r sea el gesto que está ejecutando el usuario.

g: Es el costo mayor obtenido de la comparación del *gesto nuevo* G_T y todos los *gestos conocidos* del diccionario.

L: El número de gestos conocidos que se descartan por ser muy diferentes al *gesto nuevo* G_T.

σ: Desviación estándar de las $L+1$ mejores probabilidades de los gestos (excluyendo la primera) .

μ: Promedio de las $L+1$ mejores probabilidades de los gestos (excluyendo la primera) .

it: El número de la iteración más reciente.

x_{it}: La mejor probabilidad de la iteración it.

n_σ: El número de desviaciones estándar que caben en la diferencia entre el gesto con la mayor probabilidad de las L mejores probabilidades de gestos a partir de la segunda mejor probabilidad.

γ: Umbral de desviaciones estándar para tomar una decisión final (cuando $n_\sigma > \gamma$).

$maxPer$: Límite superior del porcentaje que debe ejecutarse del *gesto nuevo* G_T antes de iniciar una clasificación forzada.

$minPer$: Límite inferior del porcentaje que debe ejecutarse del *gesto nuevo* G_T antes de tomar en cuenta las decisiones del clasificador.

w: Tamaño de la ventana, es decir, el número de cuadros necesarios para ejecutar una clasificación parcial.

Capítulo 1

Introducción

El reconocimiento automatizado de gestos tiene muchas aplicaciones en diversos campos, incluyendo videojuegos, reconocimiento de lenguaje de señas, sistemas médicos de monitoreo, entre otros M. Sushmita [2007], N. Ibraheem [2012]. En estos días existen muchos métodos altamente efectivos para el reconocimiento de gestos, algunos de los cuales requieren de costosos dispositivos especializados para la captura de las características de los gestos.

El sensor Kinect surgió recientemente y desde entonces el número de aplicaciones que hacen uso de la tecnología de reconocimiento de gestos se ha incrementado. Esto es debido a que este sensor es más económico que otros dispositivos similares, y provee datos de gran utilidad como video RGB-D y la posición de las articulaciones del cuerpo (esqueleto) en tiempo real Zhengyou [2012], Microsoft [2013b] . La mayoría de los métodos disponibles para el reconocimiento de gestos ofrecen una respuesta una vez que el gesto se ha terminado de ejecutar, ofreciendo en consecuencia una respuesta retardada. Sin embargo, existen ciertas aplicaciones en donde el reconocimiento de gestos inmediato es imprescindible, *e.g.* en sistemas interactivos y de seguridad o en aplicaciones médicas, por lo que es necesario poder reconocer los gestos antes de que éstos sean ejecutados por completo. Este problema llamado reconocimiento anticipado de gestos, apenas ha sido explorado, ya que existen muy pocos trabajos en los que se ha tratado de resolver A. Mori [2006], M. Kawashima [2011, 2010, 2009].

1.1. Definición del problema

Existen muchas aplicaciones y juegos que se basan en la detección de gestos para recibir órdenes y desempeñar sus tareas. Por ejemplo, las nuevas televisiones inteligentes que reciben órdenes del usuario mediante ademanes; las aplicacio-

nes pensadas para personas con capacidades diferentes, como los intérpretes de lenguaje basado en señas; y con la reciente aparición de Kinect, también los videojuegos han empezado a funcionar mediante ademanes. Sin embargo, existen casos de retraso evidente en el tiempo de respuesta de las aplicaciones mencionadas. Por ejemplo, Kinect es un potente sensor que detecta los movimientos de los jugadores y los transmite al juego para que un personaje virtual los imite. Esta imitación no se realiza en el tiempo deseado, puesto que se ha reportado un retraso de hasta 150 ms., lo que perjudica la experiencia en "tiempo real" que desea vivir el jugador. Otro ejemplo son las aplicaciones que funcionan con sonidos creadas para personas con capacidades diferentes; una persona invidente, por ejemplo, que desee utilizar un sistema mediante señas, no podrá darse cuenta mediante un monitor si el sistema la está detectando correctamente, en dicho caso se puede incluir un sistema de aviso por sonido en el que la persona realiza cierto gesto y el sistema le responde con un sonido inmediato relacionado con el gesto que la persona invidente está realizando. Otro caso es el de aquellas aplicaciones en las que se requiere un tiempo de respuesta especialmente corto, por ejemplo, en un sistema de vigilancia en donde hay que activar una alarma a tiempo para evitar que el ladrón escape; para lograrlo es necesario detectar que se está efectuando un robo antes de que éste sea consumado. Otro ejemplo es una aplicación que genere música al ritmo de un bailarín; si el clasificador tiene que esperar a que los gestos de baile sean terminados para luego reconocerlos, entonces la música se empezará a generar tardíamente. En este caso también es necesario anticipar los movimientos del bailarín para que la aplicación empiece a generar la música cuanto antes.

Detectar un gesto de una manera anticipada es un problema difícil de resolver. Como primer problema tenemos la cantidad de información que se debe procesar. Los dispositivos de captura de gestos como lo son los sensores, cámaras de video, acelerómetros, etc. (y que son comúnmente utilizados para la captura de gestos) ofrecen gran cantidad de información que es necesario analizar en un tiempo muy corto; dependiendo del sensor de captura seleccionado será la cantidad de datos generada. Kinect, que es el sensor elegido en esta tesis para la captura de gestos (y del que hablaremos más ampliamente en la Sección 2.1), construye un esqueleto virtual con 15 articulaciones en 3D, y refresca las posiciones de cada articulación a una velocidad de 15 o 30 cuadros por segundo (cps), *i.e.* en el peor de los casos se generarán 1350 datos por segundo (15 articulaciones $\times$ 3 dimensiones $\times$ 30 cuadros, en un segundo), por lo que hay que encontrar una manera de analizar toda esta información sin retrasar el tiempo de respuesta del sistema. Además, debemos considerar que ninguno de estos sensores o dispositivos de captura ofrece datos libres de ruido, por lo que también habrá que manejar de alguna manera

esta "suciedad" en los datos para evitar la propagación de errores. Otro problema importante es la similitud que puede haber entre dos o más gestos; al contar con varios gestos que queremos clasificar, es muy posible que varios de ellos sean semejantes en su parte inicial, media o final, haciendo aún más difícil la tarea de diferenciarlos. Es importante distinguir cada uno de estos gestos y hacerlo a tiempo para lograr una clasificación anticipada correcta. Otro de los problemas enfrentados fue la diferente velocidad en la que se ejecutan los gestos, ya sea por el mismo o diferentes usuarios; un sujeto puede realizar el mismo gesto varias veces pero con distintas velocidades y por lo tanto, con duraciones diferentes, esto es muy común cuando varias personas ejecutan el mismo gesto, cada una lo hace a su propio ritmo. En consecuencia, el clasificador debe ser capaz de identificar que dos gestos son iguales, a pesar de que el tiempo de ejecución sea distinto. Finalmente, las mayoría de las técnicas de detección de gestos utilizan costosos procesos de entrenamiento que requieren de una gran cantidad de repeticiones por gesto para poder funcionar.

1.2. Objetivo

Desarrollar un método para el reconocimiento anticipado de ademanes efectuados por una persona, en el menor tiempo y mayor exactitud posibles.

1.2.1. Objetivos específicos

Para movimientos realizados por una sola persona que fueron capturados con Kinect™:

- Clasificar de manera aislada movimientos predefinidos.
- Crear un método que permita la clasificación anticipada de gestos.
- Medir la precisión del detector de movimientos para diferentes tiempos de anticipación.
- Probar el método en distintos conjuntos de datos.

1.3. Alcances y limitaciones

Los alcances y limitaciones son los siguientes:

- El reconocimiento está enfocado en gestos realizados por una sola persona, por lo que los gestos que se quieren clasificar así como también los gestos realizados en tiempo real deberán ser ejecutados por la misma persona.
- Los gestos no deberán incluir giros ni movimientos en el piso; el sensor se encuentra limitado en cuanto al tipo de movimientos que puede detectar.
- La anticipación lograda dependerá directamente de la duración de cada gesto y de la similitud entre los mismos.

1.4. Solución propuesta

Para dar solución a los problemas anteriormente descritos, se propone desarrollar un método que permita detectar anticipadamente diferentes movimientos corporales, de manera que antes de que el movimiento sea terminado, el sistema ya lo tenga identificado y pueda ofrecer una pronta respuesta.

En el caso de los juegos de video que funcionan con ademanes, la anticipación serviría para compensar el retraso en el tiempo de respuesta del que hablamos anteriormente, mejorando de esta manera la experiencia del jugador. Además, con el suficiente tiempo de anticipación se podría agregar un factor de reacción inteligente a los oponentes virtuales de ciertos juegos de video. Por ejemplo, en un juego de boxeo podríamos lograr que el oponente virtual anticipe nuestro movimiento y pueda esquivar algunos de nuestros ataques, de esta manera se añadiría más complejidad e inteligencia al juego, haciéndolo más realista e interesante. En el caso de otros sistemas como los de vigilancia, esta anticipación podría permitir detectar un acto de robo mientras está sucediendo, y no una vez que haya sido consumado.

En este trabajo proponemos un nuevo método para el reconocimiento anticipado de gestos basado en *dynamic time warping* (DTW) usando el sensor Kinect. Las secuencias de entrada son comparadas con las secuencias que se encuentran almacenadas usando DTW, proponemos un criterio de predicción para determinar el momento en el que nuestro método está seguro de conocer la identidad del gesto representado en las secuencias entrantes. Adoptamos un método de representación que reduce la dimensión de los datos ofrecidos por el sensor de captura y además ayuda a evitar problemas en la clasificación provocados por la rotación, posición o lejanía del sujeto con respecto al sensor. Para identificar dos gestos iguales aunque éstos fueran ejecutados con duraciones distintas, utilizamos el algoritmo *dynamic time warping* (DTW). Desglosamos nuestros gestos en varias secuencias de tiempo y la comparamos por separado utilizando DTW, esto nos permitió detectar dos

gestos similares aunque la duración de su ejecución fuera distinta o que estuvieran desfasadas en el tiempo. Hablaremos más ampliamente de esto en la sección 2.5.

1.5. Contribuciones

Como contribución principal, propusimos un nuevo método para el reconocimiento anticipado de gestos. El método propuesto es capaz de trabajar bajo el escenario de aprendizaje *one-shot* [I. Guyon, 2012], *i.e.*, usando un solo ejemplo de cada categoría de gesto que se quiere reconocer. Esto resulta ventajoso en aplicaciones dinámicas y personalizadas así como en situaciones donde los datos etiquetados son escasos. Nuestro método es fácil de implementar, no necesita fase de entrenamiento y es muy eficiente.

Reportamos nuestros resultados utilizando la base de datos de referencia MSR-Actions3D y en otras bases de datos que nosotros construimos. Los resultados muestran que, en promedio, el método puede reconocer gestos con el ≈50 % de la información, perdiendo un máximo del 13 % de precisión en comparación con la precisión obtenida si utilizamos el 100 % de la información de los gestos. Con respecto a otros trabajos de reconocimiento anticipado, no hay una forma justa de comparación. Nuestro método está enfocado en reconocer los gestos de una sola persona mientras que los trabajos existentes A. Mori [2006], M. Kawashima [2011, 2010, 2009] reconocen los gestos de varias personas. Para establecer una comparación contrabajos existentes, tomamos en cuenta los resultados obtenidos con nuestro método de reconocimiento sin anticipación que alcanza el 98 % de precisión, contra los resultados obtenidos en W. Li [2010], W. Jiang [2012], que son del 88 % y 94 %.

Como resultado de esta tesis generamos el artículo *A One-shot DTW-Based Method for Early Gesture Recognition* que fue aceptado en el décimo octavo Congreso Iberoamericano en Reconocimiento de Patrones (*18th Iberoamerican Congress on Pattern Recognition CIARP 2013*) y está próximo a publicarse en el mes de noviembre de 2013.

1.6. Organización de la tesis

El presente documento está organizado como sigue. En el Capítulo 2 se introducen conceptos básicos relacionados con nuestro método de anticipación de gestos. Es importante conocerlos para entender el método propuesto ya que se estará utilizando a lo largo de todo el documento

En el Capítulo 3 definimos la diferencia entre el reconocimiento de gestos tradicional y el reconocimiento de gestos anticipado. Repasamos algunos métodos de detección anticipada ya existentes, incluyendo la descripción de las características de los gestos utilizados, los tiempos y precisiones logrados por cada uno; en general, describimos el modelo y funcionamiento de cada método. Además exponemos la diferencia de dichos trabajos con el nuestro, mencionando las ventajas y desventajas.

En el Capítulo 4 detallamos cada parte del método propuesto y sus características. Además, introducimos una nueva versión del algoritmo DTW en el que está basado nuestro método de reconocimiento de gestos.

En el Capítulo 5 puntualizamos las particularidades de los conjuntos de datos que utilizamos en los experimentos y la forma en que éstos fueron realizados. Para cada uno, reportamos los resultados obtenidos.

Finalmente, en el Capítulo 6 presentamos nuestras contribuciones y conclusiones después de una discusión de los resultados obtenidos, ventajas y desventajas observadas. Asimismo, planteamos las ideas que tenemos para trabajo futuro.

Capítulo 2

Marco teórico

En este capítulo se presentan los conceptos necesarios para entender las diferentes etapas involucradas en el desarrollo de esta investigación. Nuestro trabajo está enfocado en el reconocimiento de gestos de cuerpo completo, por lo que iniciamos dando una breve información acerca del sensor y las librerías que utilizamos para hacer la captura de los gestos. Posteriormente, introducimos algunos conceptos de álgebra lineal que están relacionados con los métodos de algunos trabajos a los que hacemos referencia. Finalmente, agregamos una breve explicación de los algoritmos en los que está basado nuestro método incluyendo algunos ejemplos para facilitar el entendimiento del lector. Al inicio de este trabajo se presenta un glosario de términos en donde se enlistan todas las variables utilizadas a lo largo de este trabajo incluyendo una breve explicación, el lector podrá consultarla cuando considere necesario.

2.1. Kinect

El Kinect es un dispositivo compuesto de diferentes sensores que es útil en diversas tareas. Fue introducido en el mercado en noviembre de 2010 como un dispositivo de entrada de la consola Xbox 360 producida por Microsoft. Fue un producto muy exitoso puesto que se vendieron más de 10 millones de copias para Marzo de 2011 [Zhengyou, 2012]. La comunidad de visión por computadora rápidamente descubrió que la tecnología de sensado de profundidad que usa Kinect podría ser utilizada para otros propósitos que sólo juegos de video con un precio más accesible que las conocidas cámaras 3D. La tecnología detrás del sensor Kinect fue desarrollada en primer lugar por la compañía *PrimeSense*, quienes liberaron su versión de un *software development kit* (SDK) que sería usado con Kinect como parte de la organización OpenNI [OpenNI, 2013].

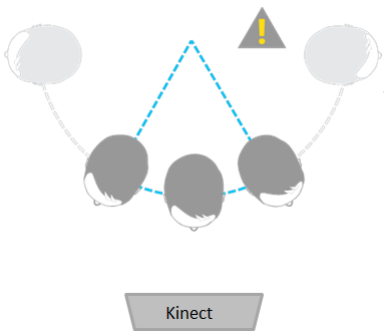


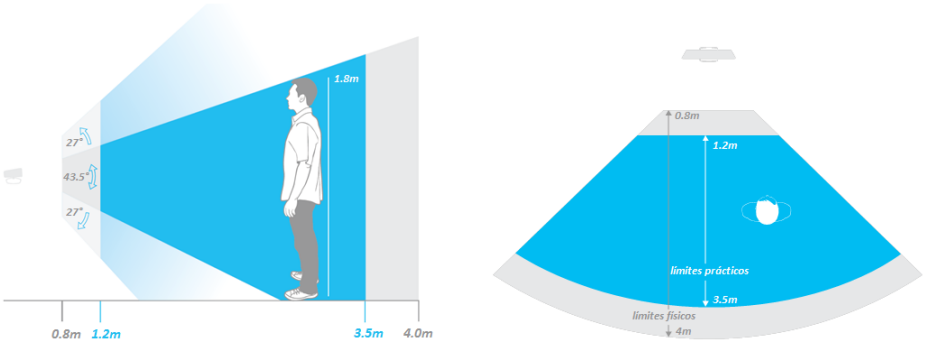
Figura 2.1: Esta imagen con vista panorámica, muestra cinco posiciones de la cabeza de un usuario y un Kinect vistos desde arriba. Ilustra las formas correctas (cabezas oscuras) e incorrectas (cabezas más claras marcadas con el signo de admiración) de posicionarse con respecto al sensor para lograr un reconocimiento facial exitoso. Figura inspirada en la figura que se muestra en Microsoft [2013b].

Las características que ofrece el dispositivo son las siguientes:

- Imagen de profundidad
- Imagen RGB
- Inclinación del sensor
- Arreglo de micrófonos
- Seguimiento de esqueleto [Microsoft, 2013b]

Lo anterior es posible gracias a que cuenta con (1) sensores de profundidad 3D que son los que hacen el seguimiento del cuerpo en un área determinada (emisor de infrarrojos), (2) cámara RGB (rojo, verde, azul) que sirve para la identificación de personas o para la captura de video o imágenes a color, (3) varios micrófonos colocados a lo largo del dispositivo que posibilitan la detección de voz (4) e inclinación motorizada para un mejor ajuste de la zona monitorizada [Microsoft, 2013b]. Este dispositivo ofrece dos resoluciones para los videos RGB-D, una es de 320×240 y la otra de 640×480 con una profundidad de color de 32 bits; ambas resoluciones son capturadas a una velocidad de 30 cuadros por segundo. Kinect soporta además el reconocimiento facial de hasta dos personas (es necesario que el rostro esté de frente al sensor como se muestra en la Figura 2.1), y el seguimiento de esqueleto de hasta seis sujetos, siempre y cuando estas personas se encuentren dentro del campo de visión del esqueleto [Microsoft, 2013b] (ver Figura 2.2). Para el seguimiento del esqueleto, Microsoft utiliza ciertos patrones de luz infrarroja

emitidos por el sensor infrarrojo de Kinect con los que calcula la profundidad de los sujetos que se encuentran dentro del área de visión del sensor, permitiendo el reconocimiento de personas o de diferentes partes del cuerpo.



(a) (b)

Figura 2.2: Campo de visión del sensor Kinect de (a) amplitud, altitud y de (b) profundidad. En la figura de la izquierda se muestra a una persona vista lateralmente y se especifican los grados y las distancias en las que deben encontrarse el sensor y el usuario para que éste último quede dentro del campo de visión. En la figura de la derecha tenemos a una persona vista panorámicamente, que se encuentra ubicada dentro de los límites (en metros) que tiene el sensor. Figuras obtenidas de Microsoft [2013b].

En este trabajo utilizamos el sensor Kinect para capturar los gestos que conformaron nuestros diccionarios. Elegimos este sensor porque es mucho más barato que otros sensores similares y existen diversas librerías gratuitas que ya realizan el análisis visual del videos y generan un esqueleto virtual, por ejemplo OpenNI [OpenNI, 2013], Kinect para Windows SDK [Microsoft, 2013a], SkelTrack [Rocha, 2013], entre otros.

2.2. OpenNI y NITE

Para controlar el sensor Kinect y extraer información significativa de los videos RGB-D que nos ofrece, utilizamos la librería construida por *Open Natural Interaction* (OpenNI). Esta librería incluye controladores que administran el funcionamiento de los diferentes dispositivos que conforman al sensor (cámaras, micrófono y motores). Para el análisis de los videos y la extracción del esqueleto virtual usamos *PrimeSense's Natural Interaction Technology* for End-User (NITE).

OpenNI es un *framework* multilenguaje y multiplataforma que define API's para escribir aplicaciones que se basan en la interacción natural (NI). Las API's de OpenNI están compuestas de un conjunto de interfaces para escribir aplicaciones NI. Su propósito principal es construir una API estándar que permita la comunicación entre:

- Sensores de audio y video (para "escuchar" y "ver" el entorno que los rodea).
- Un perceptor intermedio de audio y visión (software para analizar y posteriormente interpretar los datos de audio y video de una escena) [OpenNI, 2013].

NITE, es un software intermediario que interpreta el mundo 3D como lo conocemos, y a partir de imágenes de profundidad genera datos significativos (tratan de seguir el mismo proceso que llevaría una persona común al interpretar el mundo 3D). Mientras que el sensor captura las imágenes 3D del entorno, NITE es el motor que percibe estas imágenes e interpreta la relación que tienen el usuario y su ambiente. NITE incluye tanto algoritmos de visión computacional, como un entorno para la implementación de controles NI basados en gestos de usuarios [OpenNI, 2013].

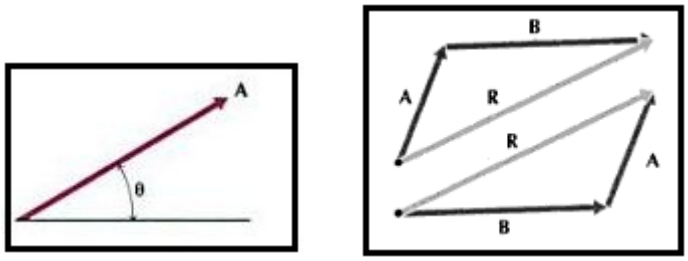
Elegimos esta librería porque es bastante rápida y no requiere de una fase de calibración para generar el esqueleto virtual como sucede con otras librerías, esto nos ayuda a ahorrar tiempo al capturar los gestos. La desventaja de no presentar una calibración al inicio de cada gesto, es que el sensor no corrobora las extremidades detectadas mediante una posición clara inicial, por lo tanto, existe una mayor posibilidad de que el posicionamiento de las extremidades al inicio de la captura sea incorrecto. Para mayor información, el lector puede referirse a los manuales que están disponibles en OpenNI [2013].

2.3. Teoría de vectores

A continuación se describen algunos conceptos de la teoría de vectores que serán útiles para comprender la Sección 4.2.1, en la que describimos un método para representar los datos crudos obtenidos de la captura de gestos. En dicho método utilizamos diversas operaciones entre vectores para crear un sistema de coordenadas esférico relativo y para calcular los ángulos zenit y azimut en los que se encuentran cada uno de los segmentos de las extremidades del esqueleto virtual, con respecto al sistema de coordenadas esférico.

Un vector está compuesto por un módulo, una dirección y un sentido y se puede representar gráficamente por una flecha como se indica en la Figura 2.3a.

El módulo del vector está representado por la longitud de la flecha, la dirección se especifica utilizando el ángulo θ que forma la flecha con una dirección de referencia conocida y el sentido lo indica la punta de la flecha situada en uno de los extremos.



(a) (b)

Figura 2.3: (a) Gráfica de un vector. (b) Representación gráfica de sumas vectoriales válidas. Figura obtenida de F. Riley [1995].

Existen diferentes operaciones posibles entre vectores, a continuación explicamos las más elementales.

Vectores unitarios Se llama vector unitario o vector normalizado a un vector con módulo igual a la unidad, esto es, $\left|\overrightarrow{A}\right| = 1$. Estos vectores pueden usarse para formar un sistema de coordenadas, en donde tres vectores unitarios tendrán las direcciones de los ejes x, y y z; el sentido de estos vectores será positivo si los vectores se encuentran dirigidos en el sentido positivo del eje x, y o z, de lo contrario será negativo.

Producto escalar El producto escalar de dos vectores $\overrightarrow{\mathbf{A}}$ y $\overrightarrow{\mathbf{B}}$ es el producto de sus módulos $\left|\overrightarrow{\mathbf{A}}\right|$ y $\left|\overrightarrow{\mathbf{B}}\right|$ y el coseno del ángulo θ que forman entre ellos, obteniendo como resultado otro escalar. Por definición, el producto escalar de $\overrightarrow{\mathbf{A}}$ y $\overrightarrow{\mathbf{B}}$ representados en la Figura 2.4 está dada en la expresión 2.1.

$$\overrightarrow{\mathbf{A}} \cdot \overrightarrow{\mathbf{B}} = \overrightarrow{\mathbf{B}} \cdot \overrightarrow{\mathbf{A}} = \left|\overrightarrow{\mathbf{A}}\right| \left|\overrightarrow{\mathbf{B}}\right| \cos\theta \tag{2.1}$$

donde $0° \leq \theta \leq 180°$. Como resultado de una multiplicación escalar obtenemos, como su nombre lo dice, un escalar y no un vector. Cuando $0° \leq \theta \leq 90°$, el escalar resultante es positivo; cuando $90° \leq \theta \leq 180°$, el escalar es negativo; y cuando el ángulo entre vectores $\overrightarrow{\mathbf{A}}$ y $\overrightarrow{\mathbf{B}}$ es $\theta = 90°$, el producto escalar será nulo.

Con el producto escalar podemos calcular las componentes escalares rectangulares de un vector. Por ejemplo, la componente escalar rectangular de $\overrightarrow{\mathbf{A}}$ en

dirección de $\overrightarrow{\mathbf{B}}$ según la Figura 2.4 es A'. Calcular $\overrightarrow{\mathbf{A}} \cdot \overrightarrow{\mathbf{B}}$es como medir el tamaño que tiene la proyección o la "sombra" que genera $\overrightarrow{\mathbf{A}}$ sobre $\overrightarrow{\mathbf{B}}$.

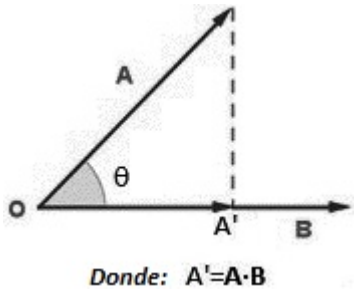


Figura 2.4: Producto punto entre dos vectores. Figura obtenida de F. Riley [1995].

Producto vectorial El producto vectorial de dos vectores $\overrightarrow{\mathbf{A}}$ y $\overrightarrow{\mathbf{B}}$ nos da como resultado un tercer vector $\overrightarrow{\mathbf{C}}$ perpendicular a ellos como muestra la Figura 2.5. Su módulo es igual al producto de los módulos de $\overrightarrow{\mathbf{A}}$ y $\overrightarrow{\mathbf{B}}$ multiplicado por el seno del ángulo θ que forman entre ellos, es decir, $\left|\overrightarrow{\mathbf{C}}\right| = \left|\overrightarrow{\mathbf{A}}\right| \times \left|\overrightarrow{\mathbf{B}}\right| \sin\theta$. Entonces, por definición, el producto vectorial de $\overrightarrow{\mathbf{A}}$ y $\overrightarrow{\mathbf{B}}$ está dado en la expresión 2.2.

$$\overrightarrow{\mathbf{C}} = \overrightarrow{\mathbf{A}} \times \overrightarrow{\mathbf{B}} = (\left|\overrightarrow{A}\right| \left|\overrightarrow{B}\right| \sin\theta) e_C \tag{2.2}$$

donde $0 \leq \theta \leq 180°$ y e_C es un vector unitario.

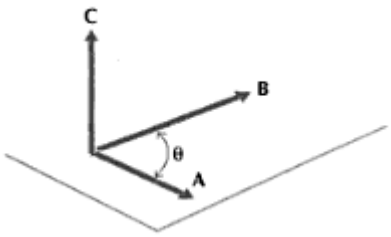


Figura 2.5: Producto cruz o vectorial entre los vectores $\overrightarrow{\mathbf{A}}$ y $\overrightarrow{\mathbf{B}}$ donde el vector $\overrightarrow{\mathbf{C}}$ es el resultado. Figura obtenida de F. Riley [1995].

Para mayor información sobre vectores, el lector puede consultar F. Riley [1995].

2.4. Análisis de Componentes Principales

La idea principal del Análisis de Componentes Principales (ACP) es reducir la dimensionalidad de un conjunto de datos que está compuesto por un número grande de variables correlacionadas, conservando la mayor parte posible de la variación presente en dicho conjunto de datos. Para lograr esto se transforma el conjunto de datos original para obtener nuevas variables que no estén correlacionadas entre sí y que se encuentren ordenadas de tal manera que al elegir sólo las primeras, se conserve la mayor parte de la variación de los datos originales. Estas variables son conocidas como componentes principales (CP). ACP nos ayuda a analizar grandes conjuntos de datos con dimensiones muy grandes que no pueden ser representadas gráficamente, por esta razón es considerado una herramienta muy poderosa para el análisis de datos. Otra ventaja que nos ofrece es la compresión de datos sin mucha pérdida de información.

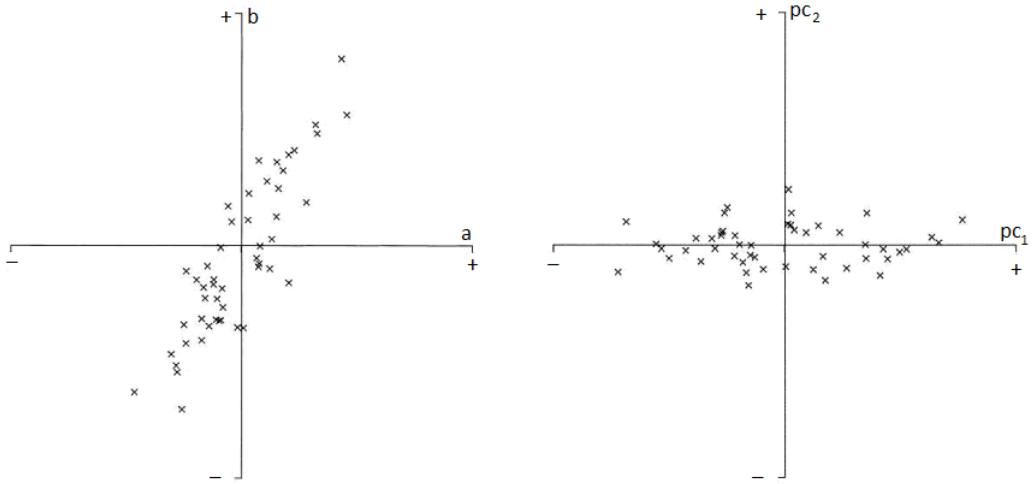


Figura 2.6: Gráfica de datos de dos dimensiones (a) antes de aplicar ACP y (b) después de aplicar ACP. Figura inspirada en Jolliffe [2002] .

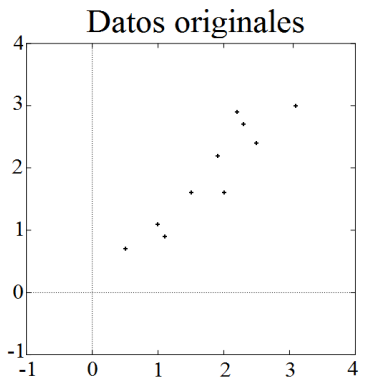
Supongamos que tenemos dos variables a y b con un conjunto de valores que puede tomar cada una, es decir, dos vectores con un número determinado de valores, en este caso estamos hablando de un problema de dimensión dm=2. Para encontrar los CP que representarán a los datos de estas variables, habrá que encontrar dm funciones lineales que no estén correlacionadas entre sí y que al mismo tiempo traten de maximizar la varianza de ambas variables mencionadas;

aunque pueden ser obtenidos dm CP's, se espera que con sólo un pequeño número dm^-, para $dm^- \ll dm$, se logre lo anteriormente dicho. En la Figura 2.6a podemos ver la gráfica de un conjunto de datos que se encuentran distribuidos a lo largo de las variables correlacionadas (a, b), y en la Figura 2.6b podemos ver los mismos datos una vez que fue aplicado ACP. Esta vez los datos se distribuyen a lo largo de las nuevas variables que en realidad son los CP's, donde la varianza de los datos es muy grande en el CP pc_1 y muy pequeña en el CP pc_2, es decir, podemos representar la mayoría de la varianza de los datos en a usando sólo a pc_1.

Datos

a	b
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3.0
2.3	2.7
2	1.6
1	1.1
1.5	1.6
1.1	0.9

(a)



(b)

Datos Ajustados

a	b
.69	.49
-1.31	-1.21
.39	.99
.09	.29
1.29	1.09
.49	.79
.19	-.31
-.81	-.81
-.31	-.31
-.71	-1.01

(c)

Figura 2.7: (a) Un ejemplo de un conjunto de datos para ilustrar el método ACP. (b) Gráfica de los datos originales. (c) Conjunto de datos después de extraer la media. Figuras y datos obtenidos de Smith [2002].

A continuación se detallan los pasos que hay que seguir para realizar un ACP, ilustrándolos con un ejemplo obtenido de Smith [2002].

1. Selección de los datos: Se seleccionan los datos que se quieren analizar. Para facilitar la explicación del método se usaron datos de dos dimensiones pero en otros dominios los datos suelen ser de dimensiones mucho más grandes. En la Figura 2.7a se muestran los datos del ejemplo, y en la Expresión 2.3 la gráfica de estos datos.

2. Extracción de la media: Se resta la media de cada dimensión por separado. Calculamos y restamos la media de los datos, cada dimensión por separado, *i.e.* a todos los valores de a se les resta $\overline{a}$, que es la media de todos los valores de a, y a todos los valores de b se les resta $\overline{b}$, que es la media de los valores de b. Esto produce un conjunto de datos cuya media es cero (ver Figura 2.7c).

3. Cálculo de la matriz de covarianza: En la Expresión 2.3 se muestra la matriz de covarianza resultante del ejemplo.

$$cov = \begin{pmatrix} 0.616555556 & 0.315444444 \\ 0.615444444 & 0.716555556 \end{pmatrix} \tag{2.3}$$

4. Cálculo de los eigenvectores y eigenvalores de la matriz de covarianza. Un eigenvector es un vector no nulo que cuando es transformado por un operador da lugar a un múltiplo escalar de sí mismo sin cambiar su dirección, i.e., dado un vector $\overrightarrow{D} \in \mathbb{R}^n$, tal que $\overrightarrow{D} \neq 0$, se dice que es un eigenvector de la matriz H si y sólo si $H\overrightarrow{D} = \lambda\overrightarrow{D}$, donde λ es un escalar y al mismo tiempo es el eigenvalor de $\overrightarrow{D}$. Los resultados de los datos de ejemplo se muestran en la Expresión 2.4 y en la Expresión 2.5.

$$eigenvalores = \begin{pmatrix} 0.0490833989 \\ 1.28402771 \end{pmatrix} \tag{2.4}$$

$$eigenvectores = \begin{pmatrix} -0.735178656 & -0.0490833989 \\ 0.677873399 & -0.735178656 \end{pmatrix} \tag{2.5}$$

5. Selección de los componentes y creación de un vector de características: Aquí se hace la compresión de los datos y la reducción de la dimensión de los mismos. Ordenamos los eigenvectores en orden descendente de acuerdo a sus eigenvalores correspondientes. Aquellos eigenvectores con los eigenvalores más altos, son las nuevas variables que obtienen la mayor varianza en los datos, mientras los eigenvectores con los eigenvalores más pequeños, se pueden descartar sin que esto implique una pérdida importante de información. Así es como ocurre la reducción de dimensión de los datos, si ignoramos algunos eigenvectores, la dimensión de los datos se reduce. Para ser precisos, si inicialmente se tienen dm dimensiones, se calculan dm eigenvectores y eigenvalores, pero si sólo conservamos los primeros dm^- eigenvectores, entonces los datos finales solamente tendrán dm^- dimensiones. En la Figura 2.8 se muestran gráficamente los eigenvectores obtenidos con los datos de ejemplo y los datos después de haber ajustado la media. Se observa que uno de los eigenvectores tiene un eigenvalor mayor, éste es más significativo porque conserva la mayor varianza en los datos, en la gráfica podemos ver cómo la línea que lo representa se alinea con la mayor parte de los puntos. Una vez elegidos los eigenvectores que queremos conservar, debemos proceder a formar un *vector de características*, que en realidad sólo es una matriz de vectores. Para construirla, tomamos los eigenvectores que

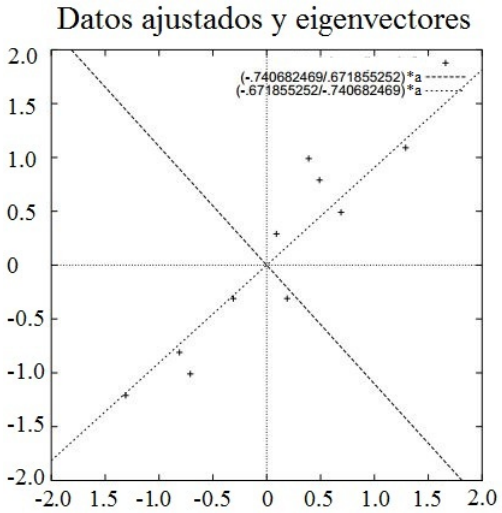


Figura 2.8: Gráfica de los datos con media ajustada y eigenvectores obtenidos.Vemos como la mayoría de los datos se encuentran alineados en el pc_1, por lo que pc_2 puede ser descartado para reducir la dimensionalidad de los datos. Figura inspirada en Smith [2002].

decidimos conservar, y formamos una matriz con estos eigenvectores en sus columnas como se indica en la expresión 2.6.

6. Derivación del nuevo conjunto de datos: Una vez que elegimos los componentes (eigenvectores) que deseamos conservar en nuestros datos y una vez que hemos formado el vector de características, simplemente tomamos la transpuesta del vector y multiplicamos por la izquierda del conjunto transpuesto de datos original después de extraer la media como se indica en la Ecuación 2.7.

$$VectorDeCaracteristicas = (eig_1, eig_2, eig_3, \ldots, eig_{dm}) \tag{2.6}$$

$$FinalData = VectorDeCaracterísticas^T \times DatosAjustados^T \tag{2.7}$$

Como se traspone la matriz que guarda los vectores de características, los eigenvectores ahora se encuentran en las filas. También la matriz de datos ajustados se traspone, por lo que ahora las filas contendrán las dimensiones y las columnas los datos.

Es importante conocer ACP para entender la Sección 4.2.1, en donde describimos el método de representación utilizado por M. Raptis [2011] *et.al.*, que es el

método de representación en el que basamos el nuestro. El método de representación de M. Raptis utiliza ACP para formar un sistema de coordenadas esférico a partir de los CP's obtenidos, mientras que nosotros usamos la dirección en la que se encuentran ciertos segmentos del torso del esqueleto para hacerlo. Este tema será tratado más profundamente en la sección antes mencionada.

Para más detalles sobre ACP,el lector puede consultar Smith [2002], Jolliffe [2002].

2.5. *Dynamic Time Warping* (DTW)

En esta sección explicaremos qué es el algoritmo DTW y para qué sirve. Esto es de suma importancia en esta tesis puesto que el método propuesto de detección anticipada está basado directamente en este algoritmo. Como describimos más adelante en el Capítulo 4, de los gestos que queremos reconocer y de los gestos que componen nuestro diccionario se obtienen varias secuencias de tiempo, mismas que son comparadas con el algoritmo DTW para determinar cuál de los gestos del diccionario se parece más al gesto que queremos reconocer.

El algoritmo DTW, es bastante popular debido a su eficiencia midiendo la similitud entre series de tiempo, minimizando el efecto negativo que pudiera ocasionar el desplazamiento o distorsión en el tiempo de dichas series. Esto se logra mediante una transformación "elástica" de las series de tiempo para detectar formas similares entre ellas que pudieran estar en diferentes fases de sus respectivas duraciones.

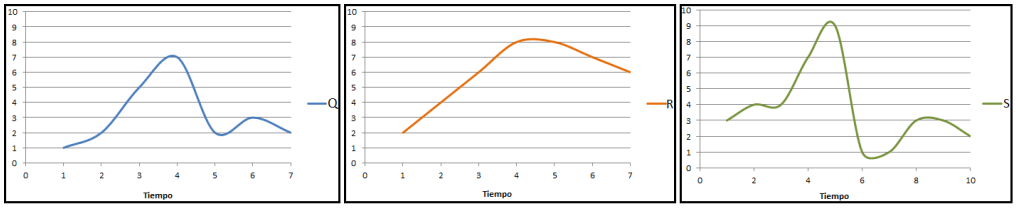


Figura 2.9: Gráficas de las series de tiempo Q,R y S. Las secuencias Q y Z son más similares entre sí.

Dadas dos series de tiempo $Q = (q_1, \ldots, q_M)$, $M \epsilon \mathbb{N}$ y $R = (r_1, \ldots, r_N)$, $N \epsilon \mathbb{N}$ representadas por una secuencia de valores (o curvas representadas por las secuencias de los vértices) DTW obtendrá una solución óptima en un tiempo con orden $O\ (MN)$. Ahora bien, existe una única restricción que ambas series de tiempo Q, R deben cumplir: El tiempo de muestreo de los datos de ambas secuencias debe ser equidistante.

Para comparar ambas secuencias Q, R necesitamos usar una medida local de distancia misma que se define como una función. Esta **función de distancia** es comúnmente conocida como "**función de costo**". Intuitivamente, si las dos secuencias que estamos comparando son parecidas, entonces la distancia d debe ser un valor pequeño, por el contrario, si dichas secuencias son diferentes, d debe ser un valor grande. Teniendo esta lógica en mente, la verdadera tarea del algoritmo DTW es buscar una correspondencia adecuada entre los valores que conforman a las secuencias Q,R, de tal manera que se minimice la función de costo (o distancia).

El algoritmo inicia construyendo una matriz de distancias $C \epsilon \mathbb{R}^{M \times N}$, que almacena las distancias que hay entre todos los posibles pares de valores entre Q, R. Esta matriz es llamada **matriz de costo local** y representada en la Ecuación 2.8.

$$C \epsilon \mathbb{R}^{M \times N} : c_{m,n} = \|q_m - r_n\|, m \epsilon [1 : M], n \epsilon [1 : N] \tag{2.8}$$

La función de costo que indicamos aquí es la más comúnmente utilizada, sin embargo, debemos seleccionar adecuadamente esta función de costo dependiendo del problema que queremos resolver. Por ejemplo, en ocasiones se utiliza la función de similitud binaria que arroja un costo igual a cero si los valores q_m y r_n son iguales, mientras que arroja un costo igual a uno si son diferentes.

Una vez que la matriz de costos es calculada, se procede a encontrar el **camino de alineamiento** con menor costo, mismo que define la correspondencia existente entre el elemento $q_m \epsilon Q$ hacia $r_n \epsilon R$. Formalmente, el camino de alineamiento P construido por DTW es una secuencia de puntos $P = p_1, \ldots, p_K$ con $p_k = (p_m, p_n) \epsilon [1 : M] \times [1 : N]$ para $k \epsilon [1 : K]$, que debe satisfacer los siguientes criterios:

1. **Condición de límites**: $p_1 = (1, 1)$ y $p_K = (M, N)$. Los puntos inicial y final del camino de alineamiento deben ser los primeros y últimos puntos de las secuencias alineadas.

2. **Condición de monotonicidad**: $m_1 \leq \ldots \leq m_k$ y $n_1 \leq \ldots \leq n_k$. Esta condición preserva el orden de los puntos en el tiempo.

3. **Condición del tamaño de paso**: Este criterio no permite saltos grandes en el camino de alineamiento. Para el algoritmo genérico de DTW, se utiliza la siguiente condición de tamaño de paso básica: $p_{l+1} - p_l \epsilon \{(1, 1), (1, 0), (0, 1)\}$.

Para encontrar el camino con menor costo, es necesario construir la **matriz de costo acumulado** $CA \epsilon \mathbb{R}^{M \times N}$, esta matriz nos dejará saber cuáles son los posibles caminos disponibles y sus respectivos costos. La matriz de costo acumulado está definida como sigue:

1. Primera fila: $CA(1,n) = \sum_{k=1}^{n} c(q_1, r_k), m\epsilon\,[1,N]$

2. Primera columna: $CA(m,1) = \sum_{k=1}^{m} c(q_k, r_1), n\epsilon\,[1,M]$

3. Todos los demás elementos: $CA\,(m,n) = min\{CA\,(m-1,n-1)\,,$

$$CA\,(m,n-1)\,, CA\,(m-1,n)\}\; + c\,(q_m, r_n)\, m\epsilon\,[1,M]\,, n\epsilon\,[1,N])$$

Al calcular completamente la matriz de costo acumulado CA, automáticamente obtenemos el costo de alineamiento total entre las secuencias de tiempo. Este costo se encuentra en la celda ubicada en la última fila y última columna de la misma $p_{end} = (M,N)$. Sin embargo, para obtener el mejor camino de alineamiento, debemos realizar una regresión desde el punto final de la matriz $P_{end} = (M,N)$ hasta el punto inicial $P_{inicial} = (1,1)$ pasando por aquellas áreas de menor costo. Dado que lo único que nos interesa es el costo total de alineación entre secuencias, el cálculo del camino óptimo de alineación queda fuera del interés de este documento. Para mayor información sobre DTW el lector deberá consultar Senin [2008].

Para poner un ejemplo, consideremos que tenemos las secuencias de tiempo $Q = (1,2,5,7,2,3,2)$, $R = (2,4,6,8,8,7,6)$ y $S = (3,4,4,7,9,1,1,3,3,2)$, cuyas gráficas se muestran en la Figura 2.9. Podemos ver que la secuencia Q es gráficamente más similar a la secuencia Z que a la secuencia R, además, debemos notar que la secuencia Z es más larga que las otras dos y que no es completamente idéntica a Q, aun así, DTW deberá detectar la similitud entre Q y Z y deberá arrojar una distancia o costo mayor para R.

		Q						
		1	**2**	**5**	**7**	**2**	**3**	**2**
S	**3**	2	1	2	4	1	0	1
	4	3	2	1	3	2	1	2
	4	3	2	1	3	2	1	2
	7	6	5	2	0	5	4	5
	9	8	7	4	2	7	6	7
	1	0	1	4	6	1	2	1
	1	0	1	4	6	1	2	1
	3	2	1	2	4	1	0	1
	3	2	1	2	4	1	0	1
	2	1	0	3	5	0	1	0

		Q						
		1	**2**	**5**	**7**	**2**	**3**	**2**
S	**3**	2	3	5	9	10	10	11
	4	5	4	4	7	9	10	12
	4	8	6	5	7	9	10	12
	7	14	11	7	5	10	13	15
	9	22	18	11	7	12	16	20
	1	22	19	15	13	8	10	11
	1	22	20	19	19	9	10	11
	3	24	21	21	23	10	9	10
	3	26	22	23	25	11	9	10
	2	27	22	25	28	11	10	**9**

Tabla 2.1: En la tabla de la derecha está la matriz de costo local y en la izquierda, la matriz de costo acumulado para Q, S. Los números rojos indican el mejor camino de alineamiento y el número azul, el costo total de alineamiento.

Primero calculamos las matrices de costo local C de Q,R y Q,Z, esto lo haremos comparando todos los valores de Q con todos los valores de R, lo mismo que para Z. Para calcular la primera fila de la matriz de C debemos comparar el valor 3 de Z contra todos los valores de Q utilizando la función de costo $c_{m,n} = |p_m - q_n|$ (ver Tabla 2.1 del lado izquierdo) por lo tanto, para $celda_{1,1}$ el costo es $c_{1,1} = |1 - 3| = 2$, para $celda_{2,1}$ el costo es $c_{2,1} = |2 - 3| = 1$, para $celda_{3,1}$ el costo es $c_{3,1} = |5 - 3| = 2$ y así sucesivamente para las demás celdas. Para calcular la matriz de costo acumulado CA (ver Tabla 2.1 del lado derecho) partimos de los valores de C. Primero calculamos los valores de la primera fila, esto lo hacemos acumulando los valores de la primera fila de C: $ca_{1,1} = c_{1,1}$ porque no hay nada que acumular, para $ca_{2,1} = c_{1,1} + c_{2,1} = 2+1 = 3$, para $ca_{3,1} = c_{1,1} + c_{2,1} + c_{3,1} = 2+1+2 = 5$ y así sucesivamente para las demás celdas de la primera fila. Después hacemos lo mismo para la primera columna, esta vez acumulando los valores de la primera columna de C. Finalmente calculamos el resto de la matriz; para calcular $ca_{2,2}$ tomaremos en cuenta los tres valores de las celdas que lo rodean que ya fueron calculadas que son $ca_{1,2}$, $ca_{2,1}$ y $ca_{1,1}$, de estos tres valores escogeremos el menor y lo acumulamos con $c_{2,2}$, esto es, $ca_{2,2} = min(ca_{1,2}, ca_{2,1}, ca_{1,1}) + c_{2,2} = min(5, 3, 2) + 2 = 4$. Esto lo hacemos para todas las celdas restantes.

		Q						
		1	**2**	**5**	**7**	**2**	**3**	**2**
R	**2**	1	1	3	5	0	1	0
	4	3	2	1	3	2	1	6
	6	5	4	1	1	4	3	4
	8	7	6	3	1	6	5	6
	8	7	6	3	1	6	5	6
	7	6	5	2	0	5	4	5
	6	5	4	1	1	4	3	4

		Q						
		1	**2**	**5**	**7**	**2**	**3**	**2**
R	**2**	1	1	4	9	9	10	10
	4	4	3	2	5	7	8	14
	6	9	7	3	3	7	10	12
	8	16	13	6	4	9	12	16
	8	23	19	9	5	10	14	18
	7	29	24	11	5	10	14	19
	6	34	28	12	6	9	12	**16**

Tabla 2.2: En la tabla de la derecha está la matriz de costo local y en la izquierda, la matriz de costo acumulado para Q, R. Los números rojos indican el mejor camino de alineamiento y el número azul, el costo total de alineamiento.

Como podemos observar, el costo total de alineamiento entre las series Q, R es de 16, mientras que el costo de alineamiento entre las series Q, Z es de 9, por lo que DTW detecta una mayor similitud entre las series Q, Z como habíamos concluido visualmente desde el inicio.

Este es un ejemplo sencillo de la manera en que trabaja DTW, el algoritmo es configurable y las restricciones mencionadas en este capítulo pueden variar dependiendo de la configuración utilizada; sin embargo, son temas que quedan fuera del alcance de este documento.

2.6. Resumen

En este capítulo se mencionaron las características de Kinect (que es el sensor de captura de gestos utilizado en este trabajo), la información que provee en cada captura y las librerías utilizadas para su manejo. De manera general, se describieron las operaciones básicas que se pueden realizar entre dos vectores, explicando de manera gráfica el resultado que ofrece cada una de ellas; vimos también el procedimiento que debe seguirse para realizar correctamente un análisis de componentes principales, explicando cuáles son los beneficios de aplicarlo sobre un conjunto de datos. Finalmente, detallamos los algoritmos DTW, incluyendo algunos ejemplos para facilitar su entendimiento.

Estos conceptos son necesarios para entender correctamente algunos métodos utilizados en esta tesis que fueron propuestos en otros trabajos. Además, también son necesarios para entender el método de reconocimiento anticipado propuesto en la presente tesis.

Capítulo 3

Estado del arte

El reconocimiento de gestos es el proceso mediante el cual los gestos realizados por un usuario son reconocidos por un receptor, donde los gestos son movimientos corporales expresivos y significativos que involucran el movimiento de las diferentes partes del cuerpo. Actualmente, el reconocimiento de gestos es usado en una amplia gama de aplicaciones M. Sushmita [2007], N. Ibraheem [2012], como por ejemplo:

- Interacción humano-computadora C. Costanzo [2003]
- Monitoreo de pacientes o adultos en la tercera edad Y. Dai [2001]
- Aplicaciones para personas con capacidades diferentes R. Anbarasi [2013], F. Soltani [2012]
- Control de robots J.L. Raheja [2010]
- Monitoreo de conductores X. Jian-Feng [2012]

Los gestos pueden ser estáticos (que es cuando un usuario permanece en una pose) y/o dinámicos (cuando el usuario inicia con una pose y se mueve a otra pose final), en algunos casos estos dos tipos de gestos son mezclados en la base de datos de gestos que se quieren clasificar. Más generalmente, existen los siguientes tipos de gestos M. Sushmita [2007]:

- Gestos de mano o brazo: gestos que involucran posiciones de manos como en los lenguajes de señas o movimientos simples de brazos en acciones simples como disparar o arrojar un objeto.

- Gestos de cabeza y cara: movimientos con la cabeza como sacudir o negar, o expresiones faciales como sorpresa, felicidad o tristeza, etc., o acciones como hablar o comer.

- Gestos corporales: involucran todo el cuerpo como en la interacción entre humanos, los movimientos de un bailarín o los movimientos de una persona en rehabilitación.

En esta tesis nos enfocamos en detectar gestos que involucran movimientos de todo el cuerpo, por lo que el tipo de gestos que son de nuestro interés son los gestos corporales. La posición, configuración y movimiento del cuerpo humano necesitan ser medidas y almacenadas para su posterior clasificación. Para lograr esto se pueden usar dispositivos de sensado conectados al cuerpo del usuario (trajes, guantes, sensores de movimiento, etc.) o cámaras y técnicas de visión por computadora. Kinect ha sido muy usado recientemente para diferentes propósitos Zhengyou [2012] incluyendo reconocimiento humano X. Lu [2011], seguimiento de manos V. Frati [2011], rehabilitación física C. Yao-Jen [2011] entre otros. En este trabajo consideramos al Kinect para la captura de los gestos a clasificar.

3.1. Reconocimiento de gestos corporales

Para solucionar el problema de clasificación de gestos de manera tradicional se han utilizado diversas herramientas como redes neuronales artificiales, algoritmos genéticos, agrupamiento difuso, entre otras N. Ibraheem [2012]. Nuestro método está basado en el reconocimiento de gestos corporales que es uno de los tipos de gestos existentes, por lo que nos limitaremos a hablar brevemente de los métodos que tratan de clasificar este tipo de gestos.

En su propuesta, A. Bobick [2002] *et.al.* hacen uso de plantillas temporales para realizar la clasificación de diferentes acciones como "sentarse" o "pararse". Utilizan imágenes de energía de movimiento (MEI), *i.e.* una imagen binaria que acumula todos aquellos pixeles de la imagen en los que se detectó movimiento a lo largo de una secuencia de cuadros; e imágenes de historial de movimiento (MHI), *i.e.* una MEI que además recuerda el tiempo en el que se movieron los pixeles. En la Figura 3.1, podemos ver en la imagen central cómo la MEI sólo acumula los pixeles en los que se detectó movimiento mientras que en la MHI, además recuerda el tiempo en el que fueron movidos; los pixeles más blancos fueron movidos más recientemente. Con estas imágenes, los autores generan un modelo estadístico para las MEI y las MHI para los gestos desconocidos y conocidos, y es mediante la comparación de estos modelos que realizan la clasificación. Para probar su método, los autores

utilizan una base de datos de 18 gestos de aerobics grabados con una cámara convencional. Como resultado de este trabajo, obtuvieron una precisión de 66 % usando sólo una de las vistas del gesto y 83 % usando varias vistas (es necesario utilizar dos cámaras para esto).

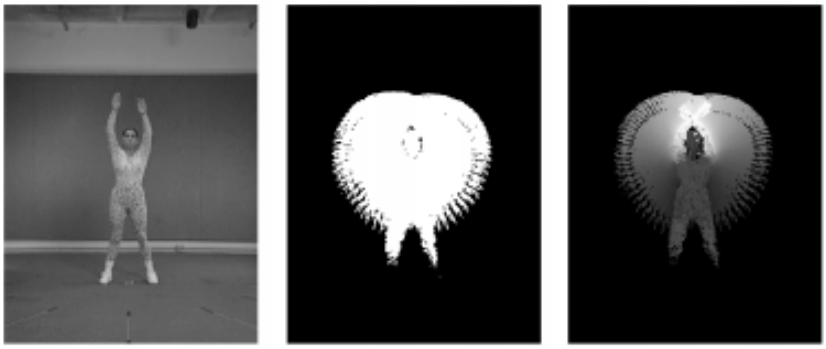
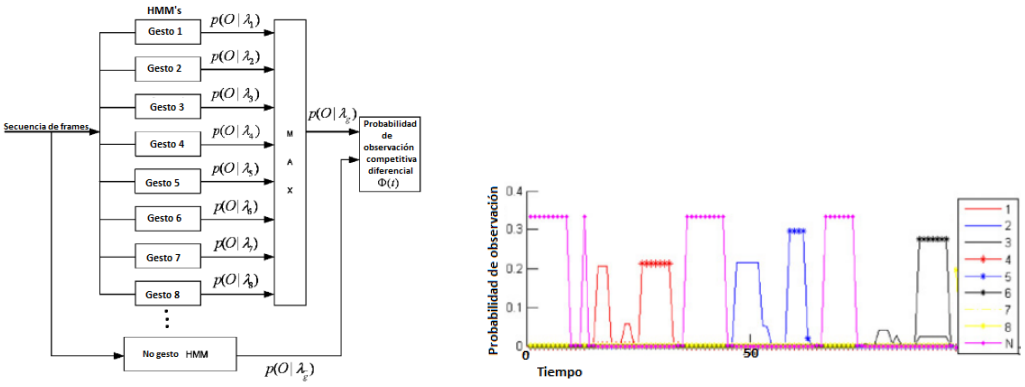


Figura 3.1: Ejemplo de una persona realizando un movimiento de aerobics. La imagen de la izquierda muestra cierto cuadro de una secuencia de movimiento, la imagen del centro muestra la MEI mientras que la imagen de la derecha muestra la MHI. Imagen obtenida de A. Bobick [2002].

Otra propuesta interesante es la descrita por K. Daehwan [2006] *et.al.*, en donde los autores utilizan HMMs para hacer la clasificación de los gestos. Propusieron un método para clasificar y segmentar al mismo tiempo los gestos realizados por una persona, dichos gestos fueron pensados para manejar una habitación inteligente y lograr acciones como "abrir cortina" o "cerrar cortina" mediante instrucciones corporales. Lo que hicieron los autores de este trabajo, fue entrenar un HMM para cada uno de los gesto en su diccionario, incluyendo el modelo de "no gesto" (ver Figura 3.2a). Para la clasificación, definieron una ventana deslizante para analizar progresivamente los cuadros del gesto entrante y con la información disponible obtuvieron una probabilidad por cada modelo en cada ventana, posteriormente graficaron estas probabilidades a través del tiempo (ver Figura 3.2b), el gesto es clasificado con la etiqueta del gesto correspondiente al HMM con la mayor probabilidad. En sus pruebas, los autores utilizaron 8 gestos diferentes, estos gestos no incluyen los movimientos de todas las extremidades, pero en las grabaciones capturaron el cuerpo completo de los sujetos; en su trabajo reportan una precisión del 95 % en reconocimiento de los gestos.

La descripción de estos métodos de clasificación tradicional nos sirve para tener una idea general del trabajo previo a la clasificación anticipada. Algunas de estas herramientas aquí vistas sirven de base para proponer nuevos métodos que ayuden a resolver el problema del reconocimiento anticipado.



(a) (b)

Figura 3.2: (a) Cálculo de la probabilidad de observación. (b) Gráfica de la probabilidad de observación de cada gesto a través del tiempo. Figura obtenida de K. Daehwan [2006].

En la siguiente sección presentamos los métodos más relevantes propuestos para dar solución al problema de reconocimiento de gestos corporales de manera anticipada.

3.2. Reconocimiento anticipado de gestos corporales

El reconocimiento anticipado de gestos es el problema de detectar diferentes gestos humanos antes de que éstos sean terminados por completo, es decir, debe realizar la clasificación de un gesto que se esté ejecutando, sin contar con toda la información sobre el mismo. A continuación se describen los principales trabajos que trataron de solucionar este problema.

3.2.1. Usando programación dinámica

La clasificación anticipada de gestos es relativamente joven; los primeros resultados fueron publicados por A. Mori [2006] *et.al.* Los autores de este trabajo deseaban utilizar el tiempo de clasificación anticipado para compensar el retraso de un robot que imitaba los movimientos de un usuario. La base de datos que utilizaron para sus experimentos tiene en total 56 gestos divididos en 18 categorías diferentes: 8 categorías son gestos donde se mueven ambos brazos, 5 categorías donde se mueve sólo el brazo derecho y 5 categorías donde se mueve sólo el brazo

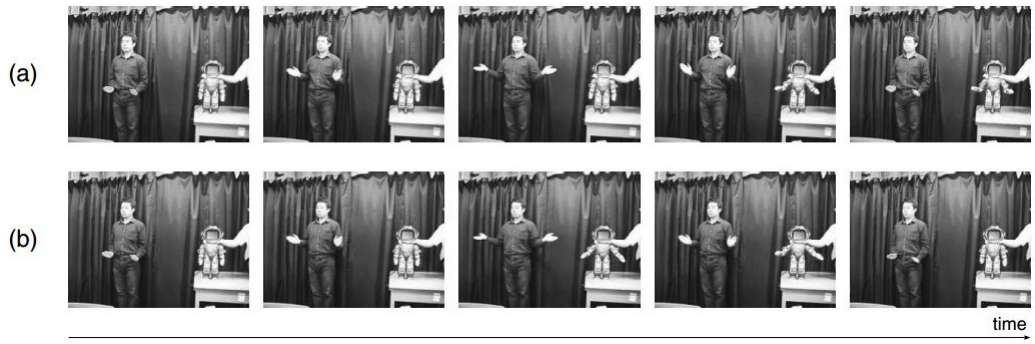


Figura 3.3: Resultados de la compensación lograda mediante el algoritmo de predicción propuesto por A. Mori [2006]; el humanoide fue conducido por (a) una postura de entrada con retraso y (b) una postura obtenida mediante reconocimiento anticipado. Figura obtenida de A. Mori [2006].

izquierdo, por lo que todos los gestos están concentrados únicamente en la parte superior del cuerpo. En promedio, los gestos tienen una duración de 87 cuadros en donde cada cuadro fue representado como un vector de características de 6 dimensiones que fue obtenido mediante un sistema de captura de movimientos estéreo.

El método que utilizaron los autores para hacer el reconocimiento anticipado está basado en un algoritmo de programación dinámica. Sus resultados fueron medidos de acuerdo al número de cuadros que fueron necesarios para realizar correctamente la mayor parte de reconocimientos: Para el algoritmo convencional, *i.e.*, sin anticipación, los autores reportan un tiempo de reconocimiento promedio y máximo de 43.4 y 83 cuadros respectivamente, y para el algoritmo de reconocimiento anticipado, un tiempo de reconocimiento promedio y máximo de 7.8 y 26 cuadros respectivamente.

Sin embargo, los autores especifican a qué velocidad (cuadros por segundo) fueron capturados ni tampoco qué duración en segundos tienen los gestos, por lo que es difícil determinar exactamente cuál es la importancia real del tiempo de anticipación reportado, que es de 1 seg. En el este trabajo tampoco se menciona qué gestos fueron los que utilizaron, solamente sabemos que su base de datos contiene gestos muy sencillos, y que en la mayor parte de ellos sólo se mueve una extremidad a la vez. En la Figura 3.3 se muestra un ejemplo de uno de sus gestos.

3.2.2. Usando mapas auto organizados

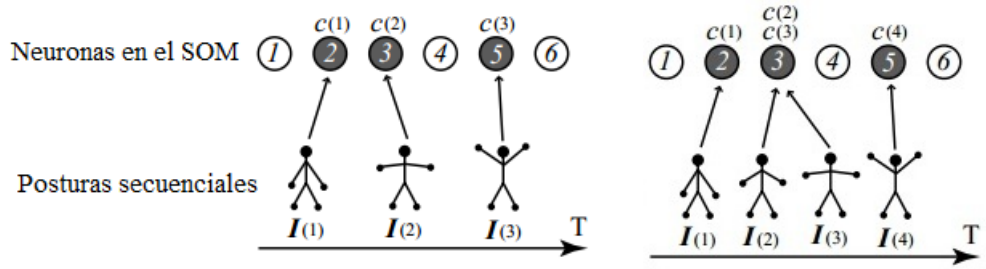
En M. Kawashima [2011, 2010, 2009] *et.al.*, los autores propusieron el reconocimiento anticipado de gestos basado en mapas auto organizados (SOM) en donde inicialmente se sugiere que todos los gestos están compuestos de una secuencia de posturas que son claves para el reconocimiento de cada gesto. En su método, los autores alimentan al SOM con todas las posturas de los gestos almacenados o conocidos y cada neurona del SOM aprende una postura diferente. Después de este proceso de entrenamiento, cada una de las posturas de cada uno de los gestos almacenados terminan representadas por una neurona. Después, del SOM entrenado se obtiene un un código disperso para cada gesto que se quiere clasificar. Cada una de las posturas que componen al gesto entrante son buscadas en el SOM, al encontrar una neurona que represente cierta postura, el indicador de dicha neurona es activado. Este indicador sólo puede adoptar dos valores, uno o cero, *i.e.*, se usa o no esa neurona para representar al gesto entrante respectivamente. Después de buscar todas las posturas que componen al gesto entrante, se juntan de manera secuencial los indicadores de todas las neuronas del SOM, obteniendo así el código disperso (ver Figura 3.4a). Las neuronas pueden ser seleccionadas como representantes de una postura más de una vez, en cuyo caso el valor del indicador de dicha neurona se establece en el valor 1 solamente una vez; esto permite que el clasificador sea capaz de hacer un reconocimiento de gestos invariante al tiempo dado que el código disperso es similar para gestos con diferente longitud como se muestra en la Figura 3.4b. Sin embargo, este código no es capaz de reconocer la diferencia entre dos gestos que tienen las mismas posturas en diferentes tiempos de la secuencia.

Una vez que se entrenó el SOM y se obtuvieron los códigos dispersos para cada *gesto conocido*, se procede a realizar el reconocimiento. Las posturas del gesto entrante que van siendo capturadas también son transformadas a código disperso; cada vez que se recibe una nueva postura se procede con los siguientes pasos:

1. Se calcula la posibilidad (*i.e.* en caso de activarse una neurona, cuánto aumentaría la probabilidad de cada gesto de ser la respuesta) y la probabilidad *a priori* (*i.e.* la probabilidad que tiene cada neurona de ser activada en caso de que ocurra cierto gesto) en el tiempo 1 para todas las neuronas y gestos.

2. Se obtiene la neurona ganadora que activa la postura entrante actual.

3. Se calcula la probabilidad *a posteriori* de todos los gestos, *i.e.*, la probabilidad que tiene cada gesto de ser la respuesta dado que la neurona del paso anterior fue elegida.



(a) (b)

Figura 3.4: (a) Obtención del código disperso. (b) Reconocimiento de tiempo invariante usando código disperso. Figura obtenida de M. Kawashima [2011]. En la Figura (a) se ilustra un gesto que está conformado por 3 posturas $(I_{(1)}, I_{(2)}, I_{(3)})$ y la neurona que reconoce a cada postura se encuentra indicada mediante una flecha. En la figura (b) tenemos un gesto compuesto por 4 posturas $(I_{(1)}, \ldots, I_{(4)})$ y de igual manera se encuentran indicadas las neuronas que representan a cada postura, sin embargo, vemos que las posturas I_2 e I_3 son reconocidas por la misma neurona, lo que ocasiona que ambos gestos sean representados por el mismo código disperso aunque tienen un numero de posturas diferente.

4. Si la probabilidad *a posteriori* de algún gesto supera cierto umbral, ese gesto es elegido como respuesta; de lo contrario se actualiza la probabilidad *a priori* y se repite el algoritmo desde el paso 1.

Para sus pruebas, los autores utilizaron 6 tipos de gestos diferentes, 120 repeticiones (20 ejemplos por tipo de gesto) para el entrenamiento y 60 repeticiones (10 por tipo de gesto) para las pruebas obteniendo una precisión total del 80 %. No obstante, omitieron los tiempos o cuadros que lograron anticipar, sólo presentan aquellos gestos que fueron clasificados correctamente con la anticipación y los gestos que presentaron más errores.

En M. Kawashima [2010] *et.al.*, los autores se enfocan en reconocer anticipadamente los gestos de dos personas que interactuan con un robot o una máquina. Ellos sostienen que los gestos que realiza una persona, a menudo tiene cierta relación con el gesto que realiza la otra persona; la idea es encontrar esta relación entre los gestos y realizar una clasificación anticipada basada en dicha relación.

El sistema por tanto siempre está observando los gestos de dos personas, y necesita construir una matriz que recuerde aquellos gestos de la persona A y de la persona B que están relacionados entre sí, esta matriz no necesariamente contiene todas las posibles combinaciones entre los gestos de A y B (ver Figura 3.5).

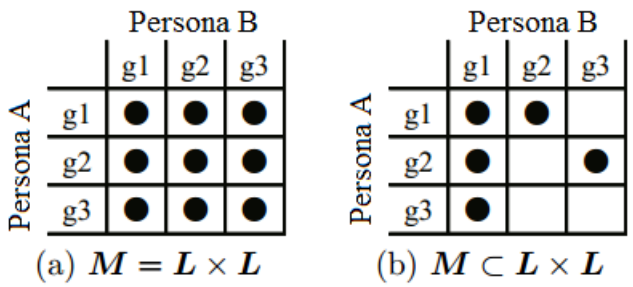


Figura 3.5: Descripción de un conjunto de gestos concurrentes M. (a) todas las posibles combinaciones de gestos concurrentes. (b) un subconjunto de todas las combinaciones. Figura obtenida de M. Kawashima [2010]. En esta figura L es el número de clases de gestos, M es el número total de combinaciones posibles de gestos concurrentes. y g_i son los diferentes gestos.

Este algoritmo de reconocimiento anticipado también está basado en SOMs y puede dividirse básicamente en dos fases: entrenamiento y pruebas. En la primera fase se realiza el entrenamiento del SOM y se extrae un código disperso por cada gesto de la misma manera que en M. Kawashima [2009]; finalmente se encuentran las relaciones entre los gestos de ambas personas, esto se hace mediante una memoria asociativa.

En la fase de pruebas el sistema observa a dos personas simultáneamente; el código disperso de cada persona es generado/actualizado inmediatamente después de que se obtiene una nueva observación (nueva postura). Finalmente, ambos códigos (el de la persona A y el de la persona B) son analizados para saber si están o no correlacionados, esto se hace accediendo a la memoria asociativa obtenida en la fase de entrenamiento; en realidad, se calcula la distancia que hay entre ambos códigos dispersos utilizando la distancia de *Hausdorff*.

Para sus pruebas, los autores utilizaron una base de datos de 10 gestos diferentes, éstos se muestran en la Figura 3.6. Los gestos fueron realizados por 7 personas diferentes, donde cada uno fue repetido 40 veces: 20 de los ejemplos de cada gesto fueron utilizados para entrenamiento y los 20 restantes para pruebas. En sus resultados reportan una precisión mayor al 95 % utilizando solamente el 25 % de la información de los gestos.

Finalmente, en el trabajo propuesto por M. Kawashima [2009] *et.al.* también utilizan SOMs para hacer el reconocimiento anticipado de gestos pero sin usar el código disperso. En lugar de ello, los autores representan cada una de las posturas de un gesto con la neurona ganadora en el SOM y el tiempo en el que fue capturada dicha postura, de esta manera todos los gestos quedan representados como una

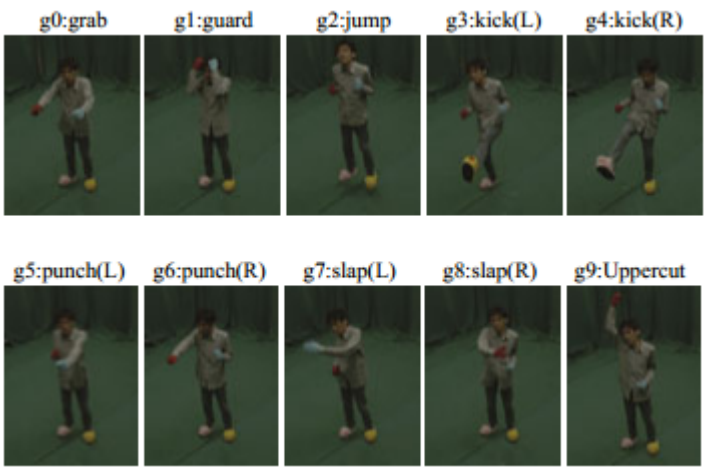


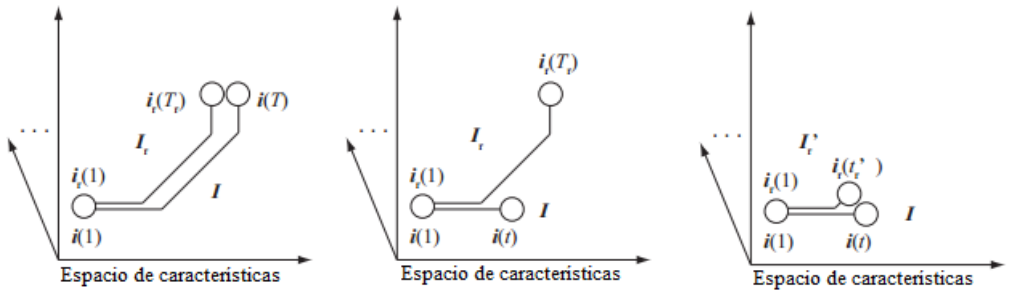
Figura 3.6: Gestos utilizados para probar el método basado en coocurrencias. Imagen obtenida de M. Kawashima [2010].

secuencia de neuronas ganadoras.

Para realizar el reconocimiento anticipado, se crea una plantilla para cada gesto almacenado de acuerdo a la información que se conoce acerca del gesto de entrada. Para generar las plantillas, se calcula la distancia euclidiana que hay entre la última postura conocida del gesto de entrada y cada una de las posturas por gesto almacenado; para cada gesto se elige la postura con menor distancia por lo que la plantilla para cada gesto estará conformada por todas las posturas anteriores a la postura con menor distancia encontrada, incluyéndose esta misma. En la Figura 3.7a podemos ver el ejemplo de un reconocimiento de gestos tradicional, donde I es el gesto de entrada y I_t es un *gesto conocido*; en la Figura 3.7b se muestra cómo en un reconocimiento anticipado, contamos con toda la información del *gesto conocido* I_t y sólo una parte del gesto de entrada I, pero en lugar de comparar todo el gesto I_t con el gesto I, se procede a tomar sólo una parte como se muestra en la Figura 3.7c, *i.e.*, se crea una plantilla para el gesto I_t.

Para hacer la comparación entre la plantilla de cada *gesto conocido* y el gesto de entrada, se utiliza la distancia de *Hausdorff* T. Cham [2007]. Una vez que la similitud de alguno de los gestos supere cierto umbral, entonces se arroja una respuesta.

Para sus experimentos, los autores utilizaron los mismos gestos que en M. Kawashima [2011], en donde reportan que en promedio es necesario solamente el 31 % del gesto para obtener resultados con precisión de un poco más del 80 %.



(a) (b) (c)

Figura 3.7: (a) Reconocimiento de gestos tradicional. (b) Reconocimiento anticipado de gestos. (c) Selección de la plantilla para un gesto. Figura obtenida de M. Kawashima [2009]. En esta figura $i_r(t)$ representa a la postura i del gesto G_r en el tiempo t, $i(t)$ es una postura del gesto desconocido G en el tiempo t, T y T_r son el número de cuadros que tiene el gesto desconocido G y el gesto G_r respectivamente. I'_r es la plantilla obtenida para el gesto G_r y t'_r es el tiempo que dura la plantilla I'_r.

3.3. Predicción de actividades humanas

Además del reconocimiento anticipado de gestos, también han surgido trabajos sobre el reconocimiento anticipado de actividades, y aunque son dos tareas diferentes están muy relacionadas entre sí. El reconocimiento de actividades es la detección automática de eventos realizados por humanos que fueron capturados en video Ryoo [2011], algunos ejemplos de actividades son: caminar, correr, hablar. El reconocimiento de actividades tradicional, al igual que el reconocimiento de gestos tradicional, se basa en la clasificación de cierta actividad una vez que ésta ha sido terminada, es decir, una vez que se cuenta con toda la información de la actividad que se quiere clasificar. Probabilísticamente, la clasificación de actividades se define como el cálculo de la probabilidad *a posteriori* de cierta actividad A_p dada una observación $\mathcal{O}$ con longitud t. Para cada una de las actividades del diccionario definido, se calcula esta probabilidad y se concluye que la actividad con mayor probabilidad, es la actividad que contiene el video $\mathcal{O}$ Ryoo [2011]. Ahora bien, la predicción de actividades es un problema un poco diferente, ya que el sistema debe tomar una decisión a la mitad de la ejecución de cierta actividad, por lo que los videos que contienen los datos necesarios para la clasificación también se encuentran incompletos. A continuación explicaremos las técnicas existentes que trataron de dar solución a este problema.

3.3.1. Usando bolsa dinámica de palabras visuales

En Ryoo [2011] el autor propone un método probabilista para la detección anticipada de actividades. En este trabajo, la extracción de características de cada actividad se realiza directamente sobre los videos que contienen las actividades a reconocer. Primeramente, se segmenta cada video en un conjunto de imágenes secuenciales, a partir de las cuales se crea una bolsa de palabras visuales (*visual bag of words VBoW*). Basado en esta idea, Ryoo plantea construir una bolsa de palabras dinámica, *i.e.*, un enfoque probabilista de predicción que construye histogramas integrales para representar las actividades humanas; Los histogramas integrales son la herramienta para lograr el reconocimiento.

Un histograma integral está compuesto de una secuencia de histogramas de características, y cada histograma de características está compuesto de b contenedores (ver Figura 3.8). Estos contenedores registran las ocurrencias de b palabras visuales que suceden en una actividad cuyo progreso está en el cuadro d y cuya duración total es d^*. Aquellas ocurrencias que sucedan después del cuadro d son descartadas, entonces, los contenedores en un histograma de características describen el número de ocurrencias de palabras visuales que se espera que sucedan en cierta actividad cuando ésta haya progresado hasta el cuadro d. En otras palabras, los histogramas de características describen la distribución de características espacio-temporales que tiene el histograma integral en cierto tiempo.

Para lograr la detección de actividades, Ryoo construye un histograma integral para cada actividad. Parte de un conjunto de videos de entrenamiento que contienen la misma actividad, y para cada uno de ellos calcula su respectivo histograma integral; el histograma que representará a dicha actividad será el histograma promedio.

Finalmente, para lograr un reconocimiento anticipado de la actividad, se divide el modelo de la actividad y la secuencia observada en múltiples segmentos para encontrar la similitud estructural entre ellos. La duración de los segmentos es dinámicamente seleccionada, encontrando la mejor correspondencia entre los pares de segmentos para calcular su similitud (*i.e.*, comparar entre sí los histogramas de los pares de segmentos). Hay que mencionar que para obtener el histograma de un segmento, se restarán de los histogramas de las actividades completas todas las ocurrencias sucedidas fuera del segmento seleccionado.

Para probar su método, Ryoo utilizó una base de datos con 6 actividades diferentes en donde algunas involucran la interacción de dos personas. Sus resultados muestran que es capaz de reconocer las actividades con un promedio de 70 % de precisión después de observar el 60 % de la duración total de los videos, mientras que al observar el 100 % de los videos obtiene un máximo de precisión del ≈75 %.

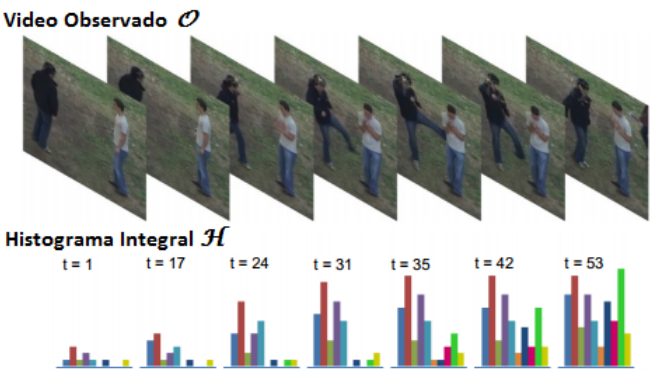


Figura 3.8: Un ejemplo de histograma integral representando un video donde el usuario lanza una patada. Un histograma integral modela cómo la distribución del histograma cambia a través del tiempo. Cada contenedor en el histograma cuenta el número de características agrupados en la palabra visual que le corresponde. Imagen obtenida de Ryoo [2011].

3.3.2. Usando modelos espacio-temporales de figuras implícitas

Posteriormente en G. Yu [2012] *et.al.* se propuso otro método para la anticipación de actividades basada en un modelo espacio temporal de figuras implícitas (*spatial-temporal implicit shape model* STISM). Los autores representan las imágenes extraídas de un video con puntos espacio-temporales de interés (*spatial temporal interest point* STIP). Su modelo está definido por $\mathcal{V} = \{f_i, s_i, c\}$ donde f_i se refiere a la descripción de características, $s_i = l_i - l_\mathcal{V}$ se refiere al desplazamiento de la ubicación espacio-temporal desde la posición del i-ésimo STIP l_i hasta la posición central del video $l_\mathcal{V}$ y c se refiere a la categoría del video. La Figura 3.9 se muestra la idea de usar el modelo STISM para encontrar la similitud entre dos actividades.

Dado un conjunto de entrenamiento $\mathcal{D} = \{(f_j, s_j, c_j)\}$ (donde los diferentes f_j compartirán la misma ubicación del centro del video y c_j, si son del mismo video de entrenamiento), el objetivo es determinar la categoría c para el video de prueba $\mathcal{V}$. Para determinar la similitud entre un video de prueba incompleto que pertenece a cierta clase, se obtiene una puntuación de similitud mediante probabilidades.

En sus resultados los autores reportan en promedio 80 % de precisión observando únicamente el 60 % de las actividades en comparación con el 91.7 % de precisión obtenido observando el 100 % de las actividades.

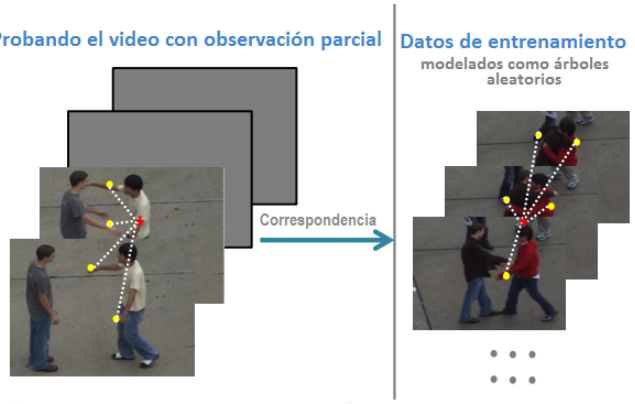


Figura 3.9: Correspondencia espacio-temporal de actividades. Los puntos amarillos indican los puntos de interés encontrados mientras que las líneas punteadas blancas indican el desplazamiento desde el punto de interés hasta el centro del video. Imagen obtenida de W. Li [2010].

3.4. Comparación

A diferencia de estas alternativas, nuestro método es capaz de reconocer gestos de cuerpo completo, en donde dos o más extremidades se mueven simultáneamente, por ejemplo, cuando una persona está bailando o cuando está practicando algún deporte. En general, los gestos que utilizan en los trabajos anteriores son más simples, *i.e.*, sólo una o dos extremidades se mueven durante todo el gesto, y ocasionalmente máximo dos se mueven simultáneamente. Es muy lógico concluir que los datos producidos por el movimiento de dos extremidades son menores que los producidos por cuatro extremidades, esto suponiendo que utilizamos el mismo sistema de captura de datos para todas las extremidades. En el presente trabajo estamos capturando los gestos con un sistema de datos en tres dimensiones; al mover solamente dos extremidades estaríamos generando una matriz de datos con dimensiones de $3 \times 3 \times 2 = 18$ datos por cuadro, que corresponden al número de articulaciones en una extremidad por el número de dimensiones por el número de extremidades por cuadro. Al capturar el movimiento de cuatro extremidades, esta cantidad de datos se duplica, por lo que se vuelve más complejo manipularlos en tiempo real y con la menor pérdida de información posible. Es por esta razón que decimos que los gestos de cuerpo completo utilizados en el presente trabajo son más complejos que aquellos en los que solamente se mueven una o dos extremidades del cuerpo.

Nuestro método está basado en DTW; a diferencia de los trabajos anteriores que trabajan con SOMs y programación dinámica, no es necesaria la fase de en-

trenamiento, lo que nos ayuda a disminuir la cantidad de repeticiones por gesto necesarias para hacer funcionar el clasificador, incluso es capaz de funcionar con un sólo ejemplo de entrenamiento. Como vimos anteriormente, los trabajos M. Kawashima [2011, 2010, 2009] requieren por lo menos de la mitad de sus ejemplos para entrenar su modelo, para posteriormente hacer la clasificación anticipada; en cambio, nuestro método basado en DTW solamente requiere un gesto de cada clase para reconocer anticipadamente; ninguno de los trabajos existentes ofrecen esta característica.

Como ya mencionamos anteriormente, existe una diferencia entre reconocer actividades y reconocer gestos, por tanto las técnicas de reconocimiento utilizados son diferentes. Sin embargo, lo que nos interesa mencionar es que en W. Li [2010], Ryoo [2011] también se aborda el problema del reconocimiento anticipado, y aunque los movimientos que se quieren clasificar son distintos, nos da una idea del auge que tiene este problema.

3.5. Resumen

Describimos cómo fue que surgió la detección de gestos y algunas de sus aplicaciones. Presentamos la definición de detección de gestos tradicional y detección de gestos anticipada para proporcionar al lector las diferencias entre ambas técnicas, que básicamente son: la diferente cantidad de información de la que se dispone para hacer el reconocimiento y el diferente nivel de certidumbre del momento adecuado para lanzar una respuesta.

Se presentó un breve resumen de las técnicas disponibles hasta el momento que abordan el problema del reconocimiento anticipado de gestos, así como también algunos trabajos que atacan el reconocimiento anticipado de actividades que, aunque son tareas diferentes, se encuentran muy relacionadas.

Al final del capítulo presentamos las diferencias principales entre los métodos existentes de detección anticipada de gestos y el propuesto en la presente tesis.

Capítulo 4

Método de reconocimiento anticipado basado en DTW

Nos propusimos clasificar anticipadamente gestos de cuerpo completo realizados por una sola persona disminuyendo el impacto que podría tener la velocidad con la que se realiza el gesto, el peso o la estatura del usuario que los realiza. Este es un problema muy complejo porque necesitamos clasificar el gesto sin disponer de la información completa del mismo y sin conocer de antemano la duración que tendrá. Además, debemos lidiar con el posible ruido que se filtra en la captura de los gestos y con la posible similitud entre los gestos en las partes iniciales; si esto último sucede, sería muy difícil de diferenciar entre varios gestos similares en sus partes iniciales provocando así una clasificación tardía. Por todo lo anterior, resulta complicado activar una respuesta correcta a tiempo.

Nuestro método consiste básicamente en tres partes:

1. **Extracción de características:** Se capturan los datos de los gestos y posteriormente se les aplica un proceso de transformación para reducir su dimensión y evitar el impacto negativo en el clasificador debido a la rotación, escala y/o traslación que presente el esqueleto virtual con el que se capturan los gestos con respecto al sensor de captura.

2. **Anticipaciones parciales:** A través de varias iteraciones, el clasificador va separando el gesto entrante y los gestos conocidos en secuencias de tiempo para luego compararlas usando DTW acumulativo; en cada iteración el clasificador toma una decisión parcial.

3. **Decisión final:** el clasificador lanza una decisión final si la decisión parcial de la iteración más reciente cumple con ciertas condiciones que son (1) un

umbral de separación de similitud, *i.e.* uno de los gestos del diccionario es notoriamente más similar al gesto que se quiere clasificar; (2) un límite máximo de tiempo de decisión, *i.e.* si cierto porcentaje del gesto que se quiere clasificar ya ha transcurrido, se toma una decisión final.

A lo largo de este capítulo explicamos detalladamente todas las partes de este método.

4.1. Captura de datos

La captura de datos es el proceso mediante el cual se extraen los datos que representan a cada gesto. Mientras el usuario ejecuta diversos gestos frente al sensor, un esqueleto virtual es generado con la ayuda de Kinect y de las librerías de OpenNI y NITE OpenNI [2013]. El esqueleto es construido con quince coordenadas 3D que corresponden con algunas de las articulaciones del cuerpo. La captura de estos datos es llevada a cabo a una velocidad de 30 cuadros por segundo (cps), por lo tanto, por cada uno de los gestos que el usuario ejecuta, nosotros creamos una matriz de $15 \times 3 \times$ *número de cuadros* para almacenar los datos crudos generados. El número de cuadros varía dependiendo del gesto realizado y del tiempo que tarda el usuario en realizarlo. Por ejemplo, suponiendo que nuestro gesto dura 20 cuadros entonces este gesto se encontraría almacenado en una matriz tridimensional de $15 \times 3 \times 20$ elementos.

La Figura 4.1 muestra las articulaciones que componen el esqueleto virtual.

4.1.1. Información básica: Suposiciones, limitantes y recomendaciones

Aún cuando se puede grabar una gran cantidad de gestos con Kinect, existen algunas limitaciones que deben tomarse en cuenta en el momento de la captura de datos; estas limitaciones dependen en gran parte también de las librerías utilizadas para crear el esqueleto. En el presente trabajo, estas limitaciones están especificadas por OpenNI y NITE, y son las siguientes:

- El sensor debe permanecer inmóvil durante toda la grabación.
- La parte superior del usuario está mayormente dentro del campo de visión del sensor.
- Debemos tratar que el cuerpo del usuario no se encuentre detrás de otros objetos.

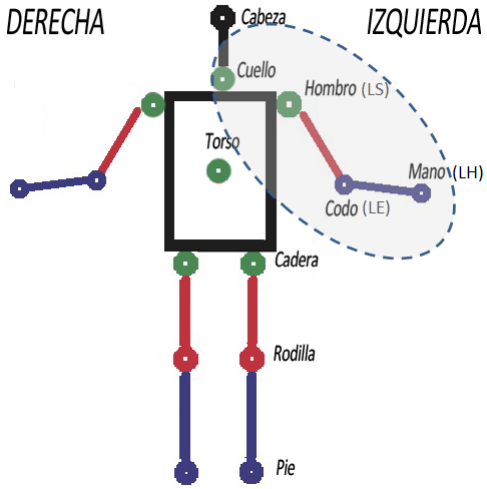


Figura 4.1: Esqueleto virtual generado por OpenNI. Primeras uniones y segmentos (marcadas con rojo); y segundas uniones y segmentos (marcadas con azul); los círculos verdes son usados para calcular el *cuadro del torso*. En la Figura 4.3, se presenta un acercamiento de la parte que se encuentra dentro del óvalo azul. .

- La distancia ideal entre el usuario y el sensor de captura es alrededor de 2.5 metros.

- Para mejores resultados, el usuario no debe usar ropa suelta o el cabello largo y suelto. Además, la captura del esqueleto con OpenNI sufre las siguientes limitantes:

    - La auto-calibración del esqueleto funciona con usuarios que se encuentran de pie. Si el usuario se encuentra sentado la calibración automática no funcionará.
    - La mayor parte del cuerpo debe estar visible para que la auto-calibración tenga lugar.
    - El usuario debe estar a no menos de 1 metro de distancia del sensor para que la calibración automática tenga lugar.

Esta información fue obtenida del manual de usuario de OpenNI que se encuentra disponible en OpenNI [2013].

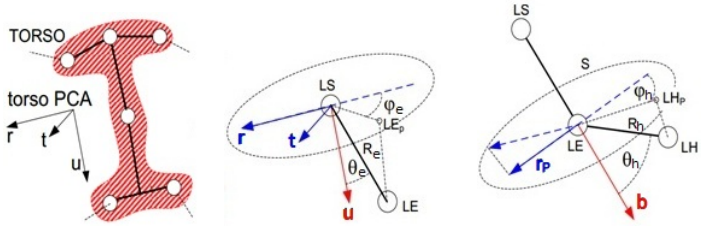
4.2. Representación de los datos

La detección anticipada de gestos exige un tiempo de respuesta muy pequeño, lo que implica que tenemos una limitante en la cantidad de datos que podemos manejar. Como mencionamos en la sección anterior, la captura de los gestos se hace a una velocidad de 30 cps, en donde cada cuadro tiene las coordenadas *x*,*y,* y *z* de cada uno de los 15 puntos que conforman el esqueleto virtual, por lo que las dimensiones de la matriz necesaria para almacenar los datos crudos de un gesto con una duración de 2 segundos sería de $(2\times30)\times15\times3$, y tendríamos que procesar toda esta información en tiempo real. Por otro lado, dado que el valor de las coordenadas correspondientes a las articulaciones del esqueleto son dependientes de la posición en la que se encuentra el usuario con respecto al sensor, cuando el usuario ejecute dos veces el mismo gesto en posiciones ligeramente diferentes, las coordenadas obtenidas serán muy disímiles. En consecuencia será mucho más difícil detectar un patrón entre estos gestos iguales. Entonces, no debe influir en la detección de gestos si el usuario está recargado hacia la derecha o hacia la izquierda del campo de visión del sensor; tampoco debe influir que el usuario sea muy alto o muy bajo, o que esté más cerca o más lejos del sensor. Incluso, tampoco debe intervenir en el reconocimiento si el usuario se encuentra ligeramente girado (*i.e.* no se encuentra totalmente de frente) con respecto al sensor. Debido a lo anterior, es necesario transformar los datos crudos y llevarlos a una representación que permita principalmente dos cosas: (1) la reducción de su dimensión y (2) lograr su invariabilidad ante las rotaciones, traslaciones y escalas del cuerpo del usuario capturado. De esta manera, una vez que los datos crudos son sometidos al proceso de representación, las coordenadas (que son obtenidas a partir de un sistema de coordenadas fijo establecido por el sensor) son transformadas en un conjunto de ángulos que describen los movimientos de amplitud y profundidad que tiene cada segmento del esqueleto. Al inicio de este trabajo se inlcuyó un glosario de términos que enlista las variables usadas a lo largo de este documento y una breve explicación, para que el lector la consulte cuando así lo requiera. A continuación se describe el método de representación adoptado.

4.2.1. Método de representación

Para la representación de los datos usamos el método propuesto por M. Raptis [2011] *et.al.* con ligeras modificaciones. Los autores hacen un análisis de componentes principales (ACP) de las coordenadas 3D que conforman el torso (ver Figura 4.1), que en nuestro trabajo es una matriz de dimensiones 6×3 por cada cuadro. De dicho análisis ellos se obtienen tres componentes principales (CP), *i.e.* una

base ortonormal 3D. El primer CP u, siempre está alineado con la dimensión más larga del torso pues es la dimensión que presenta más varianza en los datos, el segundo CP r está alineado con la línea que conecta los hombros, el último CP es $t = u \times r$, por lo que es perpendicular a los CP u y r. Los autores llaman a la base resultante *cuadro del torso* $\{u, r, t\}$ y se muestra en la Figura 4.2a. Al conjunto de articulaciones que incluye el cuello (N), hombro derecho (RS), hombro izquierdo (LS), cadera derecha (RHi), cadera izquierda (LHi) y centro del torso (T), lo llaman *torso*. Para evitar ambigüedades en este documento nos referiremos a él como *caja torácica.*



(a) (b) (c)

Figure 4.2: (a) *Cuadro del torso ACP*. Sistema de coordenadas esférico para las (b) primeras y (c) segundas uniones. Figura obtenida de M. Raptis [2011].

Las **uniones de primer grado** son los codos y las rodillas, y los segmentos de primer grado son aquellos formados por las uniones de primer grado y algunas de las articulaciones que forman parte de la *caja torácica* (ver Figura 4.1). Para representar las articulaciones de primer grado se traslada el origen del *cuadro torso* a la articulación de la *caja torácica* que está conectada a la unión de primer grado que queremos representar, posteriormente se calculan los ángulos zenit y azimut (*i.e.* ángulo lateral y de profundidad) que tiene el segmento de primer grado formado por la articulación de la *caja torácica* y la unión de primer grado que queremos representar, estos ángulos se obtienen con respecto al *cuadro del torso*. Por ejemplo, en la Figura 4.3 se muestra cómo se trasladó el origen del *cuadro del torso* a la articulación de la *caja torácica LS* y cómo se calculan los ángulos de la unión de primer grado *LE*, que finalmente queda representada por:

radio R_e – la distancia de LE desde el origen del *cuadro del torso*

inclinación θ_e – el ángulo entre u y $\overrightarrow{(LS,\, LE)}$, y

azimut φ_e – el ángulo entre r y $\overrightarrow{(LS,\, LE_p)}$

Donde LE_p es la proyección de LE en el plano cuya normal es u. Dado que la longitud del segmento $\overrightarrow{(LS,\ LE)}$ es normalizada y constante, el valor de R_e es ignorado.

Las **uniones de segundo grado** son las manos y los pies, y los segmentos de segundo grado son aquellos formados por las primeras y segundas uniones (ver Figura 4.1). La representación de las uniones de segundo grado ya no se hará con respecto a la *caja torácica* como en las uniones de primer grado, sino que estarán representados con respecto al segmento de primer grado al que están conectadas. Para lograr esto, se traslada el *torso cuadro* a la unión de primer grado con la que se conecta la unión de segundo grado que se quiere representar, posteriormente se gira todo el sistema de coordenadas hasta que el CP u se alinee con el segmento de primer grado al que está conectada la unión que queremos representar y después se obtienen los ángulos zenit y azimut de la misma forma que en las primeras uniones. Por ejemplo, en la Figura 4.2c se muestra cómo se representa la articulación mano izquierda (LH). Podemos ver cómo se trasladó el origen del *torso cuadro* a la unión de primer grado codo izquierdo (LE) porque es la unión de primer grado que está conectada a LH; también vemos cómo se giró el *torso cuadro* hasta alinear el CP u con el segmento de primer grado $\overrightarrow{(LS,\ LE)}$ (que es el segmento de primer grado al que está conectada LH). Este nuevo sistema de coordenadas esférico está compuesto por los ejes $\{b, r_p, t\}$. Finalmente, LH está descrita por:

radio R_h - la distancia desde el origen del nuevo sistema de coordenadas $\{b, r_p, t\}$,

inclinación θ_h - el ángulo entre b y $\left(\overrightarrow{LE,\ LH}\right)$, y

azimut φ_h - el ángulo entre r_p y la proyección de R_h en el plano S cuya normal es b, y $\left(\overrightarrow{LE,\ LH_p}\right)$

Donde LH_p es la proyección de LH en S. En este caso el valor de R_h también es normalizado y constante, por lo que es ignorado.

El cálculo de los sistemas de coordenadas esféricos debe repetirse en cada uno de los cuadros capturados, por lo que tendríamos que aplicar ACP en cada uno de los cuadros para obtener los CP's que describen el sistema de coordenadas del *cuadro del torso ACP*, esto obviamente requiere de varias operaciones y consumo de tiempo que necesitábamos evitar. Por esta razón aplicamos algunas modificaciones en su método de representación. Lo que propusimos fue obtener el sistema esférico de coordenadas a partir de la dirección que presentan algunos segmentos del torso del esqueleto virtual: cuello-torso nos dejará saber la dirección que tiene

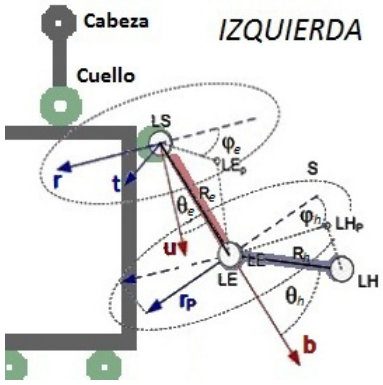


Figura 4.3: Esqueleto virtual con los sistemas de coordenadas esféricos generados para los primeros y segundos segmentos de una extremidad.

todo el cuerpo, que sería el equivalente al CP u; el segmento cuello-hombro derecho nos indicará la dirección del lado derecho del esqueleto dado que se supone que el sujeto siempre se encuentra de frente al sensor (consultar Sección 4.1.1), éste sería equivalente al CP r; finalmente se puede obtener la dirección en la que se encuentra la espalda de la persona mediante la obtención de un segmento perpendicular a los dos segmentos mencionados anteriormente, éste sería el equivalente al CP t. Para obtener el sistema de coordenadas del cuadro del torso sin usar ACP, seguimos los siguientes pasos:

1. Obtener el vector normalizado $\overrightarrow{r}$ del segmento $\left(\overrightarrow{N,\ T}\right)$.

2. Obtener un vector normalizado $\overrightarrow{u}$ del segmento $\left(\overrightarrow{N,\ RS}\right)$ ajustando $\overrightarrow{r}$ para que se cumpla que $\overrightarrow{u} \cdot \overrightarrow{r} = 1$ y que siga siendo un vector normalizado.

3. Calcular $\overrightarrow{t} = \overrightarrow{u} \times \overrightarrow{r}$,

4. Calcular los ángulos de las primeras y segundas uniones como en M. Raptis [2011].

Al hacer esto evitamos realizar todas las operaciones involucradas en ACP y finalmente obtenemos la dirección en la que se encuentra el torso respecto a Kinect, que inicialmente es lo que se busca al calcular el *cuadro del torso*. Como resultado de la representación de datos, transformamos la matriz inicial de 3×15 necesaria para almacenar un cuadro, en un vector de 16 elementos, donde estos 16 elementos son los ángulos relativos calculados de las primeras y segundas uniones (son 8 uniones y por cada una se obtienen 2 ángulos). Dado que los ángulos fueron

calculados con respecto al mismo esqueleto del usuario, éstos serán invariantes a rotaciones, traslaciones y escalas que presente el esqueleto con respecto al sensor.

Es importante entender que todos los gestos (conocidos y nuevos) pasan por el proceso de representación. Sin embargo, los gestos conocidos pasan por este proceso antes de comenzar la clasificación de los gestos nuevos, mientras que los *gesto nuevos* se van transformado poco a poco conforme van siendo ejecutados, y sus cuadros generados.

4.3. Clasificación anticipada

Sea $\mathcal{D} = \{G_1, \ldots, G_R\}$ el diccionario de gestos después del proceso de representación, y sea $G_r = \{f_r(1), \ldots, f_r(T_r)\}$ uno de los gestos en el vocabulario para $r \in \{1, \ldots, R\}$, donde R es el número de clases diferentes de gestos. Cada G_r está compuesto de una secuencia de T_r-cuadros, en donde cada cuadro está representado como se describió en la sección anterior: $f_r(t) = \{\theta_{r,1}, \ldots, \theta_{r,16}\}$ para $t < T_r$, donde $\theta_{r,i}$ es el *i-ésimo* ángulo. Debemos resaltar que tenemos un único gesto para cada clase particular *i.e.*, un escenario con aprendizaje *one-shot* I. Guyon [2012], por lo que tenemos una repetición para cada una de las R clases de gestos diferentes. El *gesto nuevo* que queremos reconocer está denotado por $G_T = \{f_T(1), \ldots, f_T(T_T)\}$, donde T_T es el número de cuadros que tiene G_T.

El clasificador recibe secuencialmente los cuadros de un nuevo gesto a una velocidad de 30 cps. Para evitar tener que hacer una predicción cada vez que un cuadro es recibido, el clasificador espera hasta que w cuadros son acumulados y después hace una predicción parcial mediante la comparación entre el *gesto nuevo* y los gestos conocidos. Si no es posible, el método espera hasta recibir w cuadros y entonces realiza otra comparación. Este proceso iterativo es repetido varias veces hasta que el gesto es reconocido o el final del nuevo gesto es alcanzado (para conocer cómo estimamos el final de un gesto, consulte la Sección 4.3.3).

Cada vez que el clasificador hace una comparación de w cuadros, en realidad lo que hace es estimar la distancia entre la información parcial disponible del *gesto nuevo* y la información parcial de los gestos conocidos en $\mathcal{D}$. Para esto, consideramos cada uno de los 16 ángulos en un gesto hasta el tiempo t_{it} como sigue: $G_r(t_{it}) = \{A_{r,1}(t_{it}), \ldots, A_{r,16}(t_{it})\}$, donde $A_{r,i}(t_{it}) = \{\theta_{r,i}(1), \ldots, \theta_{r,i}(t_{it})\}$ para $1 \leq i \leq 16$, son las 16 secuencias de tiempo del gesto G_r y $\theta_{r,i}(t_{it})$ es el ángulo θ_i del gesto G_r en el tiempo t_{it} que es el índice del último cuadro del *gesto nuevo* en la iteración más reciente it. Esta misma consideración se hace también para G_T.

Para estimar la distancia entre la información parcial de los gestos conocidos

y nuevos, usamos *dynamic time warping* (DTW) porque es uno de los métodos más usados para comparar secuencias que pueden variar en tiempo o velocidad (ver Sección 2.5). Para evitar recalcular la similitud entre la información parcial de los gestos conocidos y el *gesto nuevo* que ya ha sido calculada en iteraciones previas, hicimos una modificación al algoritmo DTW para volverlo acumulativo (DTWacc): en cada iteración DTWacc recibe una nueva parte de las dos secuencias de tiempo que se desean comparar, calcula la similitud entre estas partes y agrega esto al resultado obtenido en iteraciones pasadas (ver Figura 4.4). Al comparar dos secuencias de tiempo con DTWacc, el algoritmo nos da como resultado una distancia. Este algoritmo será mejor detallado en la siguiente sección.

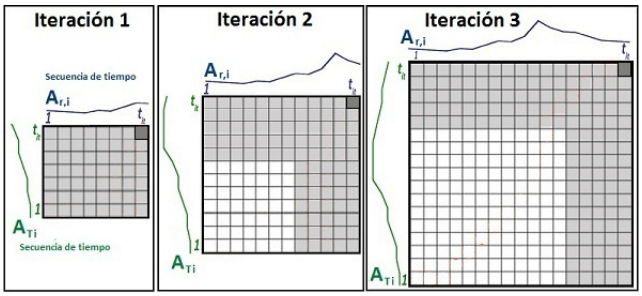


Figure 4.4: Tres iteraciones del algoritmo DTWacc. Las partes de color gris muestran las partes de cada secuencia que compara DTWacc; en color blanco se muestran los resultados obtenidos en iteraciones anteriores; La celda de las matrices marcada con gris oscuro almacena el costo total de alineamiento de las dos secuencias de tiempo..

4.3.1. DTW acumulativo

El algoritmo DTWacc es similar a DTW tradicional, con la diferencia de que las secuencias de tiempo que se van a comparar son alimentadas por partes. El algoritmo recibe una parte de ambas secuencias de tiempo, las evalúa generando y recordando las matrices de costo local y acumulado (ver Sección 2.5). Cuando llega una nueva parte de las secuencias (en una nueva iteración), el algoritmo utiliza la información de las matrices calculadas en iteraciones anteriores para determinar la similitud de los nuevos fragmentos con las partes ya conocidas de las secuencias. Como resultado tenemos que las matrices de costo local y acumulado de las diferentes iteraciones se complementan.

Como lo único que nos interesa saber es la similitud existente entre dos secuencias de tiempo, el descubrimiento del mejor camino de alineamiento no es

necesario, así que esta parte de DTW tradicional queda fuera de nuestro algoritmo de DTWacc.

La Figura 4.4 muestra el funcionamiento del algoritmo DTWacc. En la iteración 1 observamos cómo las primeras partes de las dos secuencias de tiempo A_{Ti} y $A_{r,i}$ son alimentadas en el algoritmo; la zona gris indica las partes de las matrices de costo local y acumulado que son calculadas, como es la primera parte de ambas series, todos los valores de A_{Ti} se comparan contra todos los valores de $A_{r,i}$. En la iteración 2, una nueva parte de las secuencias de tiempo son alimentadas; en esta ocasión no se comparan de nuevo todos los elementos de A_{Ti} con todos los elementos de $A_{r,i}$, sino solamente se calculan aquellos valores que no hayan sido comparados entre sí; utilizando nuestro algoritmo DTWacc después de alimentar completamente ambas secuencias de tiempo, obtenemos el mismo costo de alineamiento que al usar el algoritmo DTW tradicional, con la ventaja de que DTWacc nos ofrece costos de alinemaiento de manera iterativa sin la necesidad de conocer inicialmente la totalidad de las secuencias de tiempo. Las zonas de color gris indican las partes de las matrices de costo local y acumulado que son calculadas en la iteración correspondiente, mientras que la zona blanca indica aquellos datos obtenidos en iteraciones anteriores que son recordados y que sirven para calcular los valores de la zona gris. Las celdas resaltadas con gris oscuro son las que almacenan el costo total de alineamiento entre las secuencias de tiempo en cada iteración. Como vimos en la Sección 2.5, a pesar de que el algoritmo DTW tradicional calcula el mejor camino de alineamiento entre dos series basándose en la matriz de costo acumulado, este camino de alineamiento no influye en las decisiones parciales ni finales de nuestro clasificador, ya que lo único que tomamos en cuenta para realizar la clasificación es el costo total de alineamiento, y éste se obtiene sin necesidad de conocer el mejor camino de alineamiento. Por esta razón, el descubrimiento del mejor camino de alineamiento quedan fuera del interés del presente trabajo, y es por ello también que nuestro algoritmo DTWacc omite esta parte del algoritmo original.

4.3.2. Predicciones parciales

Antes de que el clasificador empiece a tomar decisiones parciales o finales como veremos más adelante, primero espera a que cierto porcentaje del *gesto nuevo* G_T ya haya sido ejecutado, esto es porque cuando el gesto acaba de empezar puede no haber información suficiente como para tomar una decisión, y en algunas ocasiones los gestos pueden presentar similitud entre ellos en sus fases iniciales, por lo que la decisión en esta parte del *gesto nuevo* G_T resultaría poco confiable. Entonces el clasificador espera a que cierto porcentaje *minPer* del *gesto nuevo* G_T haya sido

ejecutado para empezar a tomar decisiones parciales.

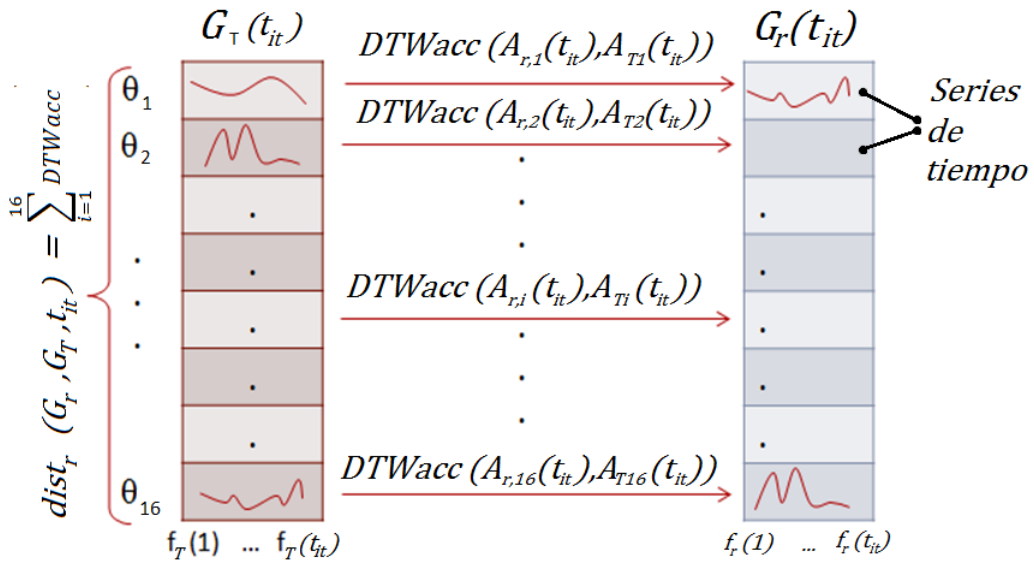


Figura 4.5: Cálculo de la distancia total resultante de la comparación entre el *gesto nuevo* G_T y el *gesto conocido* G_r hasta la iteración t_{it}.

Para calcular la distancia entre la información parcial al tiempo t_{it} de un *gesto conocido* G_r y un nuevo gesto G_T con DTW, usamos la ecuación 4.1.

$$dist_r(G_r, G_T, t_{it}) = \sum_{i=1}^{16} DTWacc(A_{r,i}(t_{it}), A_{Ti}(t_{it})) \tag{4.1}$$

donde $A_{r,i}(t_{it}) = \{\theta_{r,i}(1), \ldots, \theta_{r,i}(t_{it})\}$ para $1 \leq i \leq 16$ son las 16 secuencias de tiempo del gesto G_r y $A_{Ti}(t_{it})$ de manera semejante, son las secuencias de tiempo para el gesto G_T.
Como cada secuencia corresponde a la acumulación a través del tiempo de uno de los 16 ángulos existentes en un cuadro, todos los gestos quedan representados por 16 secuencias de tiempo. En la Figura 4.5 se muestra la forma en la que se compara el *gesto conocido* G_r con el *gesto nuevo* G_T hasta el tiempo t_{it}, este proceso se realiza para cada gesto en el diccionario $\mathcal{D}$. Como alternativa del método, incorporamos un umbral de movimiento μ_m para eliminar aquellos segmentos de las extremidades que el usuario mueve muy ligeramente o no mueve para nada, y que por lo tanto resultan inútiles para el reconocimiento; sólo aquellas secuencias de tiempo que se mueven más de μ_m, son consideradas en $dist_r(G_r, G_T, t_{it})$. En el Capítulo 5 se explica la forma en que asignamos valor a μ_m y se presentan ambos resultados, con y sin el umbral de movimiento μ_m.

Hasta este punto tenemos una distancia *dist* por cada *gesto conocido*, en donde el gesto con la menor distancia es el gesto más parecido al gesto desconocido G_T. Esta distancia es normalizada y posteriormente transformada en probabilidad inversa para que a los gestos con menor distancia corresponda una mayor probabilidad de ser la respuesta (ver Ecuación 4.2).

$$P_r(T = r \mid G_T) \approx 1 - \frac{dist_r(t_{it})}{g} \tag{4.2}$$

Donde:

$$g = \text{máx}\left(dist_1(G_1, G_T, t_{it}), \ldots, dist_R(G_R, G_T, t_{it})\right) \tag{4.3}$$

y t_{it} es el índice del último cuadro de la iteración más reciente y $P_r(T = r \mid G_T)$ es la probabilidad de que el gesto el *gesto nuevo* G_T sea reconocido con el índice de la clase r.

El gesto con mayor probabilidad de cada iteración será elegido como una predicción parcial.

4.3.3. Predicción final

Después de que el clasificador ha hecho una predicción parcial, el método debe decidir si ésta debe considerarse una predicción final. Nosotros proponemos dos formas de tomar una decisión final (*i.e.*, activar una bandera indicando que el gesto ha sido reconocido):

- Por **separación** donde uno de los gestos conocidos es notablemente más similar al nuevo gesto**.**
- Por **clasificación forzada** donde el nuevo gesto está por terminar, de acuerdo a un estimado de la duración del gesto y es necesario hacer una predicción.

Para una **decisión por separación** consideramos dos aspectos: (1) el número de desviaciones estándar n_σ que caben en la diferencia entre el gesto con la mayor probabilidad de las siguientes L mejores probabilidades de gestos, y (2) verificar que cierto porcentaje de la duración estimada del nuevo gesto haya sido ejecutada. Definimos la constante L para descartar los $R - L + 1$ gestos conocidos con las peores probabilidades evitando considerar aquellos gestos que tienen muy baja probabilidad de ser la respuesta. Con los gestos restantes calculamos la desviación estándar σ y el promedio μ para calcular n_σ. Si n_σ excede cierto umbral γ, entonces el clasificador arroja una decisión final.

$$n_\sigma(t_{it}) = \frac{|x_{it} - \mu|}{\sigma} \tag{4.4}$$

donde μ es el promedio y σ la desviación estándar de las $L + 1$ mejores probabilidades de los gestos (excluyendo la primera) para $1 \leq L < R$, y x_{it} es la mejor probabilidad de la iteración *it*.

En la **decisión forzada** el clasificador provee una respuesta porque fue estimado que más de $maxPer$ (un porcentaje definido muy cercano a 100 %) del *gesto nuevo* ha sido ejecutado y hasta entonces no hubo una decisión por separación. Como no sabemos cuánto durará el *gesto nuevo*, calculamos un estimado para prevenir que el *gesto nuevo* sea terminado sin que el clasificador arroje una respuesta final o para prevenir respuestas tardías. Consideramos que la longitud total del *gesto nuevo* es la mínima duración obtenida de los dos gestos con mayor probabilidad en la iteración más reciente, por lo que esta duración es recalculada en cada iteración.

Si cualquiera de estos dos escenarios se presenta, entonces el clasificador lanzará como respuesta aquel gesto que tenga la mejor probabilidad en la iteración más reciente.

4.4. Resumen

En este capítulo presentamos la solución propuesta basada en DTW al problema de detección anticipada de gestos. Describimos que nuestro método consta de tres partes que son: (1) Extracción de características, (2) anticipaciones parciales, (3) decisión final. La extracción de características nos permite tener una representación de los datos invariante ante la rotación y escala del usuario que efectúa los gestos; las anticipaciones parciales son decisiones que toma el clasificador en cada una de las iteraciones del método, una de estas decisiones parciales se convertirá en la decisión final cuando el gesto entrante esté a punto de terminar, o cuando uno de los gestos en el diccionario sea notoriamente más similar al gesto entrante que los demás.

En el Capítulo 5 evaluamos los aspectos del método propuesto y todas sus variaciones.

Capítulo 5

Experimentos y evaluación

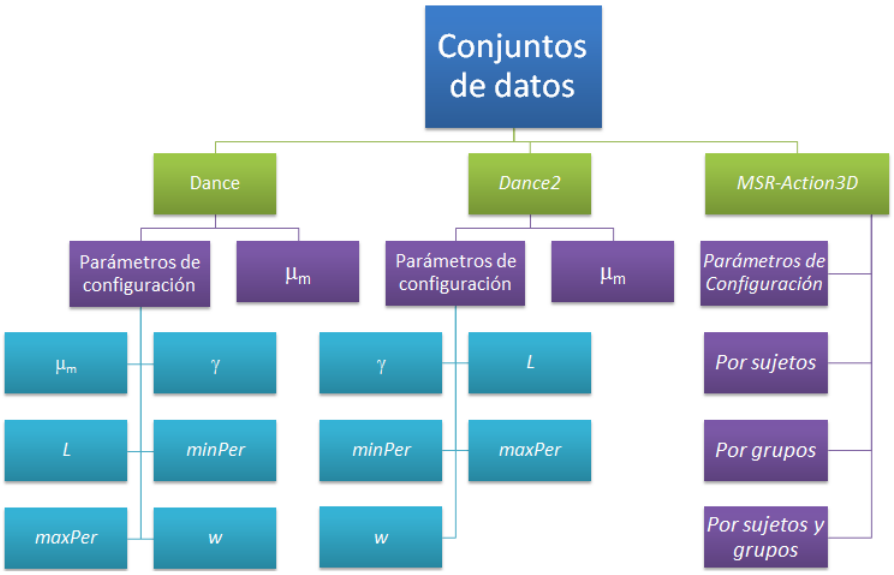
A lo largo de este capítulo mostramos los resultados obtenidos en la fase de experimentación de nuestro trabajo. Utilizamos dos diferentes conjuntos de datos creados por nosotros para probar nuestro método y una más usada en trabajos anteriores para compararnos con otros trabajos; estos se explicarán más adelante.

En la Figura 5.1 se muestra un esquema general de los experimentos que hemos realizado. Los experimentos realizados con *MSR-Action3D* fueron realizados para compararnos con otros trabajos. Dado que los experimentos con los parámetros de configuración fueron realizados para encontrar la mejor combinación de los mismos, éstos se encuentran en los anexos.

5.1. Conjuntos de datos de gestos

Para los experimentos usamos tres bases de datos diferentes. Sus características serán descritas a continuación. El primer conjunto de datos con el que trabajamos, *Dance*, fue uno que nosotros mismos construimos. Este conjunto está compuesto por ocho gestos diferentes de baile que fueron extraídos del juego *Dance Central 2* para Xbox 360 con Kinect: (1) *Make way*, (2) *step side*, (3) *head way*, (4) topple, (5) count in, (6) latino, (7) huh y (8) muscle man (ver Figura 5.1). Estos gestos fueron realizados por 3 sujetos diferentes, donde cada uno realizó una única ejecución de cada gesto, por lo que tenemos 3 repeticiones de cada uno. Este conjunto de datos fue capturado con Kinect a una tasa de velocidad de 30 cps y una resolución de 640×480. Con esta base de datos queremos evaluar la capacidad que tiene nuestro método para identificar una cantidad de clases de gestos realizados por diferentes usuarios y con una disponibilidad de repeticiones escasa.

El segundo conjunto de datos es también de gestos de baile creado por nosotros, *Dance2*: los gestos considerados son (A) subir y bajar brazo, (B) apuntar hacia el



Cuadro 5.1: Esquema que muestra los experimentos realizados para cada conjunto de datos.

cielo, (C) moviendo brazos y piernas, (D) baile vaquero. Estos gestos se ilustran en la Figura 5.2. Los pasos de baile de cuerpo completo fueron realizados 10 veces cada uno por una sola persona. Las condiciones de grabación (dispositivo utilizado, velocidad y resolución) fueron las mismas que en el conjunto de datos *Dance*. Con esta base de datos queremos evaluar la capacidad que tiene nuestro método para clasificar un número pequeño de gestos realizado por un único usuario cuando se hacen varias repeticiones por gesto

El tercer conjunto de datos es *MSR-Action3D* W. Li [2010], que está conformado por 20 acciones: *ondear alto el brazo (1), ondear horizontalmente el brazo (2), martillo (3), atrapar (4), puñetazo hacia adelante (5), tirar alto (6), dibujar x (7), dibujar una palomita (8), dibujar círculo (9), aplaudir (10), agitar ambos brazos (11), boxear de lado (12), agachar (13), pataear hacia adelante (14), patear hacia un lado (15), trotar (16), golpe de tenis (17), servicio de tenis (18), golpe de golf (19), recoger y aventar (20).* Cada uno fue realizado por 10 sujetos por un máximo de 3 veces. Este conjunto de datos fue capturado con una cámara de profundidad a una tasa de 15 fps y con una resolución de 640×480. El esqueleto está formado por 20 articulaciones, donde los lados izquierdo y derecho del esqueleto están invertidos (ver Figura 5.4), sin embargo, solamente utilizamos

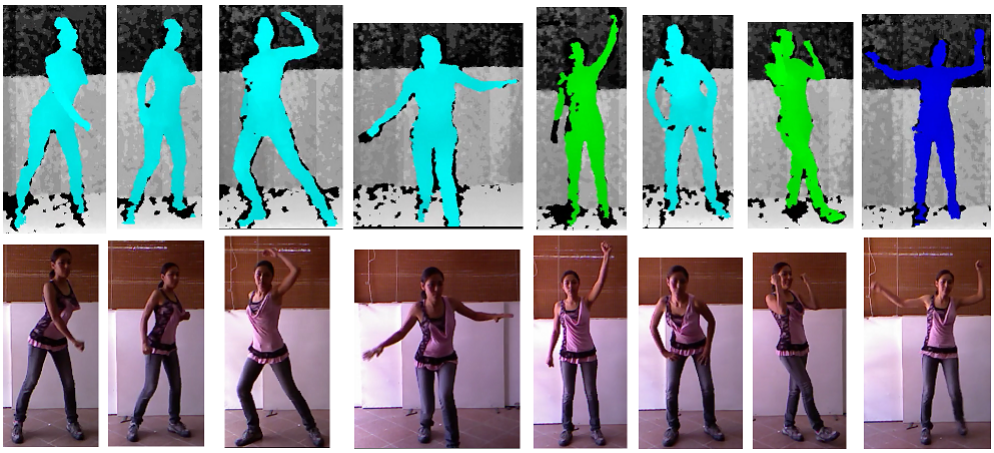


Figura 5.1: Gestos de la base de datos *Dance*. La primera fila contiene una captura del mapa de profundidad de cada gesto y en la segunda fila una captura en RGB de cada gesto.

las 15 articulaciones que son equivalentes a las que tiene el esqueleto creado por OpenNI mientras que las restantes son descartadas. Para solucionar el problema de los lados invertidos, basta con invertir las etiquetas de las articulaciones en la matriz en donde se almacenan los datos. Además *MSR-Action3D* es uno de los conjuntos de datos más usados para el reconocimiento de acciones, usando Kinect. Con esta base de datos intentamos comparar nuestro clasificador con otros existentes, además de medir la capacidad de nuestro método para clasificar una gran cantidad de clases de gestos, realizadas por varios sujetos diferentes y realizando varias repeticiones de cada uno.

5.2. Análisis de parámetros

En esta sección se describen los experimentos realizados para evaluar nuestro método y se exponen los resultados obtenidos. Nuestro método depende de un conjunto de parámetros de configuración que son los siguientes:

- Número de gestos tomados en cuenta para la toma de decisiones (L). Este parámetro nos permite dejar de tomar en cuenta aquellos gestos que son muy diferentes al gesto entrante que se quiere clasificar. De esta manera ahorramos tiempo en las comparación de gestos.

- Límite de número de desviaciones estándar γ para la toma de decisiones. Este parámetro representa la diferencia mínima que debe haber en las similitudes

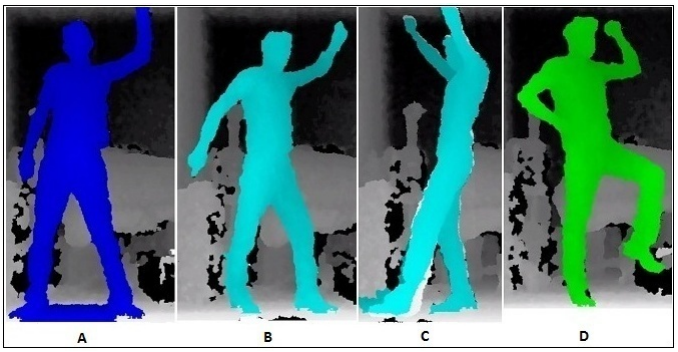


Figura 5.2: Gestos que componen el conjunto de datos *Dance2*, las imágenes muestran el mapa de profundidad construido por Kinect. La letra indica la etiqueta que se le asignó a cada gesto, mediante la cual serán reconocidos.

entre gestos para tomar una decisión final por separación. Estrictamente hablando es un número que mide cuántas desviaciones estándar caben en la diferencia de distancias entre los gestos.

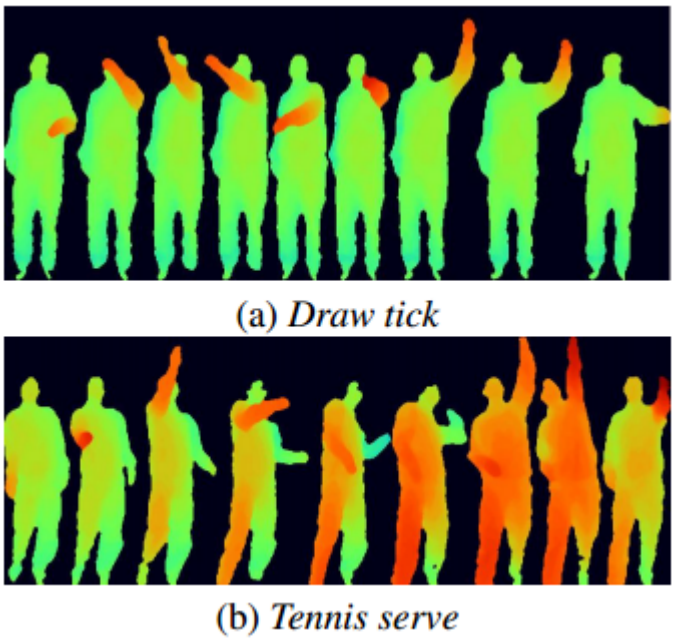
- Umbral mínimo ($minPer$) para la toma de decisiones parciales y finales. Este parámetro indica el porcentaje del gesto que debe haber sido ejecutado para que el clasificador empiece a tomar decisiones parciales, d esta manera evitamos tomar una decisión demasiado precipitada.

- Umbral de decisión forzada *maxPer*. Este parámetro indica el máximo del porcentaje del gesto que debe ser ejecutado antes de que el clasificador arroje una decisión forzada, lo que ayuda al clasificador a dar una respuesta en caso de que no haya una decisión por separación.

- Tamaño de la ventana *w*. Este parámetro establece el número de cuadros que se evaluarán en cada iteración, lo que nos permite disminuir en gran cantidad el número de evaluaciones que realizará el clasificador y aprovechar de mejor manera la información que aporta cada cuadro.

Otro parámetro para nuestro método es el umbral de movimiento μ_m que sirve para eliminar aquellas extremidades que no se mueven lo suficiente durante un gesto y no tomarlas en cuenta para la clasificación; como nuestro método no depende de este parámetro, no lo consideramos como un parámetro de configuración.

Para cada uno los parámetros realizamos un análisis para determinar el impacto que tiene cada una en nuestro método y el valor con el que obtenemos los mejores resultados en la detección anticipada. Para determinar la mejor combinación de los valores de los parámetros de configuración hicimos una evaluación



(a) *Draw tick*

(b) *Tennis serve*

Figura 5.3: Dos ejemplos de gestos que componen el conjunto de datos *MSR-Action3D*. Imagen obtenida de W. Li [2010]

sistemática, *i.e.*, primero probamos varios valores para cada parámetro y fijamos el mejor de una manera secuencial. La mejor configuración obtenida fue la siguiente: $L = 5$, $\gamma = 2.0$, $minPer = 50$,$maxPer = 80$.

El detalle de cada una de las pruebas con los diferentes parámetros de configuración, se encuentra en la sección de anexos. Sabemos que la elección de parámetros de manera secuencial puede afectar el comportamiento de los parámetros de forma individual, es por ello que especificamos en la Sección 6.4 que aplicar un método de optimización para la configuración de estos parámetros es considerado como trabajo futuro.

5.3. Experimentos con *Dance* y *Dance2*

En esta sección se presentan los resultados obtenidos con la mejor combinación de parámetros de configuración encontrada. Estos experimentos fueron realizados para conocer la precisión que nuestro método es capaz de alcanzar. En todos los experimentos realizados con *Dance* y *Dance2* obtenemos la precisión de reconocimiento con anticipación (CA) y sin anticipación (SA).

Para todos los experimentos realizados con el conjunto de datos *Dance* selec-

Figura 5.4: Esqueleto que ofrece la librería SDK de Microsoft. Las articulaciones marcadas de color naranja son consideradas como equivalentes a las 15 articulaciones del esqueleto de OpenNI; las de color negro, son descartadas. Figura obtenida de M. Sushmita [2007].

cionamos aleatoriamente 70 % de los ejemplos para entrenar y el 30 % de los gestos restantes para probar. Con el conjunto de entrenamiento formamos también de manera aleatoria 5 diccionarios diferentes , cada uno con 1 ejemplo de cada clase de gesto, de tal forma que las repeticiones de los gestos se usaran al menos una vez.

Usando la mejor configuración obtenida (ver sección 5.2) y $w = 12$, obtuvimos los resultados de la Tabla 5.2. En la Tabla 5.2b se muestran los resultados obtenidos con clasificación CA y SA para las 5 repeticiones del experimento, mientras que en la Tabla 5.2b se muestran los porcentajes promedio, mínimo y máximo de la información usada para la clasificación, y el tiempo promedio en milisegundos que le toma al clasificador reconocer el gesto.

Para este conjunto de datos logramos una precisión del 100 % CA y de 95 % SA. Podemos ver que el máximo de información usada para la clasificación de un gesto fue de 91 %, mientras que el mínimo fue 25 %; en promedio nuestro método es capaz de reconocer los gestos utilizando únicamente el 45 % de la información. El tiempo que tarda en hacer el reconocimiento fue medido en milisegundos, el tiempo que se indica en la tabla indica el tiempo que tarda desde el momento en que inicia el gesto hasta el momento en que el clasificador arroja una respuesta. En promedio los gestos de esta base de datos tienen una duración de 3.45 seg. es decir 3458 ms., lo que quiere decir que logramos una clasificación en tiempo

	Precisión CA	Precisión SA
Repetición 1	1.00±0	0.95±0
Repetición 2	1.00±0	0.95±0
Repetición 3	1.00±0	0.95±0
Repetición 4	1.00±0	0.95±0
Repetición 5	1.00±0	0.95±0
Promedio	1.00±0	0.95±0

(a)

Porcentaje promedio	0.45
Porcentaje mínimo	0.25
Porcentaje máximo	0.91
Tiempo promedio en ms.	668

(b)

Cuadro 5.2: Resultados de los experimentos realizados con el conjunto de datos *Dance*.

real, por lo que la mayor parte de los gestos es reconocida antes del 50 % de su duración total.

Para la base de datos *Dance2* también elegimos el 70 % de los gestos para entrenamiento y el 30 % restante para pruebas, obteniendo en esta ocasión 10 diccionarios diferentes (debido a la mayor cantidad de repeticiones) asegurándonos de utilizar al menos 1 vez cada repetición. Los resultados obtenidos al utilizar la mejor combinación de parámetros obtenida y $w = 5$ se muestran en la Tabla 5.3. En la Tabla 5.3a se indica la precisión obtenida para las 10 repeticiones del experimento y en la Tabla 5.3b los porcentajes promedio, mínimo y máximo de la información del gesto necesaria para la clasificación, y el tiempo en milisegundos que le tomó al clasificador hacer el reconocimiento.

	Precisión CA	Precisión SA
Repetición 1	0.75±0.07	0.98±0
Repetición 2	0.83±0.07	0.98±0
Repetición 3	0.92±0.07	0.98±0
Repetición 4	0.83±0.07	0.98±0
Repetición 5	0.83±0.07	0.98±0
Repetición 6	0.83±0.07	0.98±0
Repetición 7	0.83±0.07	0.98±0
Repetición 8	0.83±0.07	0.98±0
Repetición 9	1.00±0.07	0.98±0
Repetición 10	0.83±0.07	0.98±0
Promedio	0.85±0.07	0.98±0

(a)

Porcentaje promedio	0.53
Porcentaje mínimo	0.30
Porcentaje máximo	0.87
Tiempo promedio en ms.	19.26

(b)

Cuadro 5.3: Resultados de los experimentos realizados con *Dance2*.

En estos resultados obtuvimos una precisión promedio de 85 % CA y 98 % SA; perdimos el 13 % de precisión. Sin embargo, en promedio solamente necesitamos el

53 % de la información del gesto para hacer el reconocimiento, un máximo de 87 % y un mínimo de 30 %. Con respecto al tiempo, el clasificador solamente necesita en promedio 19.26 ms. para clasificar un gesto, donde la duración promedio de los gestos es de 2.87 seg. (2871 ms), por lo que nuestro método en este conjunto de datos también es capaz de reconocer los gestos en tiempo real. Esto quiere decir que, en promedio, los gestos son reconocidos una vez que se haya ejecutado el 53 % de la totalidad del mismo.

5.3.1. Experimentos con umbral de movimiento (μ_m)

Realizamos este experimento para comprobar si el umbral de movimiento μ_m tiene un impacto positivo en nuestro método. Lo que hicimos fue ir variando el valor del umbral cada 15 unidades (esta variable se encuentra expresada en grados) y en aquellos rangos de valores en donde notamos un comportamiento interesante en los resultados, exploramos más valores de la variable. Utilizamos los mismos diccionarios aleatorios usados a lo largo de los experimentos de parámetros, y probamos con todos los ejemplos de prueba asignados para ello. En la Tabla 5.4 se muestran los resultados obtenidos para *Dance*. Podemos ver que los valores de μ_m van avanzando en saltos de 15 unidades excepto en el rango 15-30 que es en donde encontramos un comportamiento interesante que explicaremos más adelante. Las filas 2 y 3 muestran la precisión promedio obtenida en el reconocimiento CA y SA respectivamente, la fila 3 contiene la desviación estándar de los datos de reconocimiento CA, las filas 4-6 muestran el porcentaje promedio, mínimo y máximo respectivamente (*i.e.* porcentaje de información del *gesto nuevo* que es necesaria para realizar el reconocimiento) y las filas 8 y 9 muestran el tiempo en milisegundos requerido para realizar la clasificación de un *gesto nuevo* CA y SA respectivamente. Primeramente mencionaremos que a partir de $\mu_m = 20$ la precisión del clasificador empieza a descender cada vez más, por lo que los valores $\mu_m \geq 20$ dejan de ser de interés. Es cierto que en los valores para μ_m mostrados en las columnas 1, 2 y 3 el clasificador mantiene su precisión, pero la diferencia está en el tiempo que tarda cada una de estas configuraciones en clasificar un *gesto nuevo*; si comparamos el tiempo que requiere el clasificador cuando $\mu_m = 10$ con el tiempo que requiere cuando $\mu_m = 17$, el segundo requiere 54 ms. menos CA y 73 ms. SA. Aunque el tiempo ahorrado en este experimento parece insignificante, veremos que este va creciendo conforme aumenta la cantidad de gestos. Finalmente debemos notar que el porcentaje mínimo y máximo se mantienen en las columnas 1-3 que son las de interés, mientras que el porcentaje promedio se reduce en un 3 %.

De este experimento podemos concluir que el umbral de movimiento ayuda

$\mu_m \rightarrow$	10	15	17	20	30	45	60	75	90	105	120
Precisión promedio CA	1.00	1.00	1.00	0.95	0.95	0.90	0.85	0.80	0.13	0.13	0.13
Precisión promedio SA	0.95	0.95	0.95	0.95	0.95	0.95	0.90	0.83	0.13	0.13	0.13
σ precisión promedio CA	0.00	0.00	0.00	0.07	0.07	0.10	0.10	0.07	0.00	0.00	0.00
Porcentaje promedio	0.43	0.41	0.41	0.43	0.42	0.41	0.47	0.43	0.82	0.82	0.82
Porcentaje mínimo	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.53	0.53	0.53
Porcentaje máximo	0.71	0.71	0.71	0.71	0.70	0.66	1.00	0.84	1.00	1.00	1.00
Tiempo prom. CA en ms.	688	690	634	579	598	551	507	374	45	33	34
Tiempo prom. SA en ms.	864	861	791	724	754	688	630	466	48	35	37

Tabla 5.4: Resultados obtenidos para los diferentes valores de μ_m en la base de datos *Dance*.

a nuestro método a realizar la clasificación más rápidamente, además, no todas las extremidades son necesarias para realizar el reconocimiento de los gestos. Aumentar demasiado el umbral de movimiento μ_m podría perjudicar la precisión del clasificador, por lo que hay que establecer los límites cuidadosa y moderadamente. Esta variable sí afecta la precisión obtenida en el reconocimiento de gestos SA.

Para la base de datos *Dance2*, utilizamos los mismos 10 diccionarios aleatorios de prueba usados a lo largo del análisis de parámetros para el método, y probamos con el conjunto de datos para prueba. En la Tabla 5.5 se muestran los resultados obtenidos con la misma organización de filas y columnas que para el experimento con *Dance*. En este experimento también podemos ver que si aumentamos demasiado el valor del umbral μ_m, la precisión del método se empieza a perder para los reconocimientos CA y SA. Al igual que con *Dance*, el tiempo requerido para realizar el reconocimiento es menor conforme aumentamos el umbral μ_m, sin embargo, en esta base de datos con menor número de gestos no es tan notorio este ahorro de tiempo. En este experimento también se conserva la precisión (no aumenta) del clasificador en los primeros valores de μ_m, lo que confirma que no es necesario contemplar todos los movimientos de las extremidades para realizar correctamente la clasificación.

En conclusión, la precisión del clasificador no aumentó sino que disminuyó en los casos en los que μ_m no fue bien elegida. En este experimento la cantidad de gestos es muy pequeña como para notar un ahorro significativo de tiempo en el reconocimiento de los gestos. Como mencionamos al inicio de este anexo, μ_m no es uno de los parámetros de nuestro método, sino es una variación del mismo, por lo que no elegimos el mejor valor obtenido para usarlo en los posteriores experimentos como hicimos con los parámetros de configuración.

$\mu_m \rightarrow$	15	30	45	60	75	90	105
Precisión promedio CA	0.84	0.81	0.76	0.81	0.71	0.25	0.25
Precisión promedio SA	0.98	0.98	0.95	0.95	0.79	0.25	0.25
σ precisión promedio CA	0.10	0.10	0.10	0.11	0.11	0.00	0.00

Porcentaje promedio	0.54	0.53	0.53	0.56	0.58	0.83	0.83
Porcentaje mínimo	0.30	0.30	0.30	0.30	0.30	0.52	0.52
Porcentaje máximo	1.00	1.00	1.00	1.00	1.00	1.00	1.00

Tiempo promedio CA en ms.	8	9	9	8	8	4	4
Tiempo promedio SA en ms.	11	13	12	11	11	6	6

Tabla 5.5: Resultados obtenidos para los diferentes valores de μ_m en la base de datos *Dance2*.

5.4. Experimentos con el conjunto de datos *MSR-Action3D*

Elegimos la base de datos *MSR-Action3D* para hacer algunos experimentos porque existen trabajos de detección de gestos SA que utilizaron este mismo conjunto de datos para realizar sus pruebas y reportaron algunos resultados con los que queremos comparar los resultados de nuestro método W. Li [2010], W. Jiang [2012]. El conjunto de datos *MSR-Action3D* está compuesto por gestos realizados por varias personas, lo que puede afectar el reconocimiento de nuestro método.

Conjunto de datos->	Completo		Por grupos		Por sujetos		Por sujetos y grupos	
	CA	SA	CA	SA	CA	SA	CA	SA
Ryoo [2011]				×				
W. Jiang [2012]		×						
Nuestro método SA	×	×	×	×	×	×	×	×

Tabla 5.6: Diferentes formas en las que se consideró el conjunto de datos *MSR-Action3D* para realizar las pruebas de la detección de gestos tradicional, de los trabajos presentados por Ryoo [2011], W. Jiang [2012]. Ryoo *et.al.* presentan resultados con la base de datos *MSR-Action3D* separada por grupos mientras que W. Jiang *et.al.* utilizaron la base de datos completa. Aquellas marcadas de color azul son las comparaciones que pudimos hacer de manera directa con los resultados obtenidos con nuestro método.

En la Tabla 5.6 indicamos los experimentos que realizamos con nuestro método utilizando la base de datos *MSR-Action3D* y los experimentos realizados por trabajos anteriores. Pudimos comparar los resultados obtenidos por W. Jiang [2012]

(usando todo el conjunto de datos) y los resultados obtenidos por W. Li [2010] (separando el conjunto de datos por grupos) con nuestros resultados (usando todo el conjunto de datos y separándolo por grupos, respectivamente). Estos resultados los veremos más adelante.

5.4.1. Experimentos con los parámetros del método

Debido a la gran cantidad de datos y de gestos que tiene esta base de datos, no realizamos todos los experimentos de la configuración de parámetros como lo hicimos con los conjuntos de datos *Dance* y *Dance2* (ver anexos). Lo que hicimos fue utilizar la mejor configuración obtenida de *Dance* y *Dance2* en la base de datos *MSR-Action3D*, además de otras configuraciones que se muestran en la Tabla 5.7. Hicimos este experimento ara observar cómo se comporta nuestro método al usar la mejor configuración de parámetros. En la Tabla 5.8 se muestran los resultados con las diferentes configuraciones. Las columnas 1-4 contienen los resultados obtenidos para cada una de la configuraciones. Recordemos que μ_m no es un parámetro de configuración sino una variación a nuestro método de reconocimiento, por lo que no se incluye en esta tabla.

	L	n_σ	*minPer*	*maxPer*	w
Configuración 1	2	2.0	50	80	8
Configuración 2	2	3.0	50	80	8
Configuración 3	2	3.0	70	80	8
Configuración 4	2	3.0	70	80	13

Tabla 5.7: Diferentes configuraciones usadas para probar los parámetros de configuración en *MSR-Action3D*.

En la columna 1 y fila 3, podemos ver que la mayoría de los gestos son clasificados después de que se ha ejecutado por lo menos el 35 % de su totalidad mientras que el porcentaje mínimo (fila 4) es de 14 %, esto nos indica que los gestos se están clasificando muy rápido. Al usar la Configuración 2 (columna 2) se aumenta ligeramente el número de desviaciones estándar para tomar una decisión por separación, vemos que el porcentaje promedio aumentó a 54 % mientras que el porcentaje mínimo continuó en 14 %, esto quiere decir que al aumentar el valor de γ, el porcentaje de información del gesto necesario para una clasificación aumenta. Al usar la Configuración 3 se elevó el parámetro *minPer*, como consecuencia el porcentaje mínimo aumentó hasta 20 %, al modificar el umbral inferior de decisión entonces el porcentaje mínimo de información de los gestos necesaria para el reconocimiento aumenta también. Finalmente, al usar la Configuración 4 (columna 4) en donde se aumentó el tamaño de la ventana, tenemos

menor cantidad de evaluaciones por lo que el tiempo de respuesta se reduce. En la Figura 5.5 podemos ver el tiempo en milisegundos (ms) necesario para clasificar un *gesto nuevo* CA y SA, en donde el tiempo de clasificación CA se redujo en más de 850 ms. al aumentar el tamaño de la ventana, y en el reconocimiento SA se redujo en más de 950 ms. Si comparamos la precisión promedio alcanzada con el reconocimiento SA y CA, sólo existe una diferencia de 1 % de precisión usando la Configuración 4, y aunque con la Configuración 3 alcanzamos la misma precisión en ambos reconocimientos, el tiempo que toma hacerlo es mucho más grande, es por esta razón que consideramos que la Configuración 4 es la mejor.

	Nuestro método				W. Jiang [2012]
	Conf1	Conf2	Conf3	Conf4	
Precisión promedio CA	0.31	0.34	0.35	0.34	/
Precisión promedio SA	0.35	0.35	0.35	0.35	0.88
Porcentaje promedio	0.35	0.54	0.59	0.63	/
Porcentaje mínimo	0.14	0.14	0.20	0.20	/
Porcentaje máximo	1.00	1.00	1.00	1.00	/
Tiempo prom. CA en ms.	3108.00	3624.00	3463.00	2583.00	/
Tiempo prom. SA en ms.	3434.00	3976.00	3794.00	2809.00	/

Tabla 5.8: Resultados obtenidos con el conjunto de datos *MSR-Action3D* para varias configuraciones.

En general, vimos el efecto causado en el método al ir modificando cada uno de los parámetros. Al final, la mejor configuración para el conjunto de datos *MSR-Action3D* no es el mismo que para *Dance* y *Dance2*, lo que quiere decir que la mejor configuración depende del tipo de gestos que conformen el conjunto de datos. También observamos que la estimación de la duración de los gestos en esta base de datos no es muy precisa, por ejemplo, al usar la Conf1 utilizamos $minPer = 50$; sin embargo el verdadero límite inferior que es el que se reporta en la Tabla 5.8 es de 14 % en promedio. Como consecuencia, aunque fijamos nuestro límite mínimo y máximo de clasificación, el clasificador se sale de estos rangos clasificando gestos antes y después de lo establecido. Además, la mejor precisión registrada con esta base de datos es del 35 % SA y 34 % CA, que está muy por debajo de la precisión reportada en W. Jiang [2012] que es del 88 %. La desventaja del método de W. Jiang *et.al.* es que requiere de un proceso complejo de entrenamiento y numerosas repeticiones del mismo gesto. En la siguiente sección realizamos la misma evaluación que realizaron los autores de W. Li [2010] con el conjunto de datos *MSR-Action3D*, para poder compararnos de una manera más

justa con sus resultados.

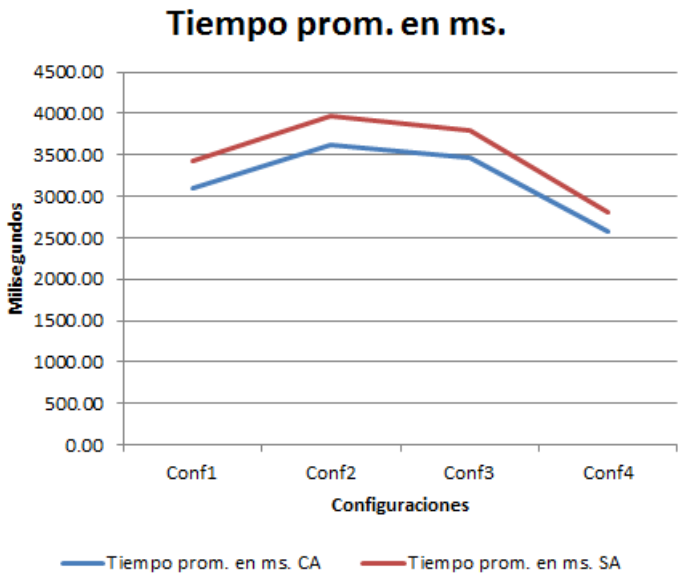


Figura 5.5: Milisegundos necesarios para el reconocimiento de un *gesto nuevo* CA y SA en *MSR-Action3D*.

5.4.2. Experimentos con *MSR-Action3D* dividida por sujetos

Nuestro método fue pensado para clasificar los gestos realizados por una sola persona, por lo que este experimento se realizó para medir la precisión de nuestro método en el conjunto de datos *MSR-Action3D* al separar los gestos por el sujeto que los realizó. Como mencionamos al inicio de este capítulo, los sujetos que participaron en la grabación de este conjunto fueron 10. Para este experimento, se usó la mejor combinación de parámetros de configuración encontrada, se dividieron por sujeto todos los gestos del conjunto de datos, se tomó el 30 % de los gestos de cada grupo y se generaron diferentes diccionarios aleatorios (un gesto de cada clase) y probamos cada uno de ellos con el 70 % de los gestos restantes de cada grupo. En la Tabla 5.9 se muestran los resultados obtenidos. en las filas 2 y 3 se muestra la precisión alcanzada para el reconocimiento SA y CA respectivamente, en la fila 4 se especifica el porcentaje del *gesto nuevo* que fue usado para lograr el reconocimiento; en la última columna se muestra la precisión promedio. Para obtener el 93 % de precisión con el reconocimiento SA, se requiere del 100 % del

gesto mientras que para lograr el 87 % de precisión con el reconocimiento CA, sólo se utilizó en promedio el 41 % de los gestos, esto quiere decir que la mayor parte de los gestos son reconocidos antes de que se haya ejecutado la mitad de éstos. Entre el reconocimiento SA y CA sólo existe una diferencia de 6 % en la precisión, mientras que en la diferencia de porcentaje de información necesaria hay una diferencia del 59 % (que es la cantidad de información que estamos ahorrando).

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	Prom.
Reconocimiento SA	0.95	0.94	0.86	0.88	0.95	0.97	0.97	0.98	0.87	0.95	0.93

Reconocimiento CA	0.85	0.69	0.81	0.88	0.95	0.88	1.00	0.90	0.82	0.90	0.87
Porcentaje del gesto	0.37	0.33	0.25	0.52	0.40	0.50	0.44	0.39	0.38	0.47	0.41

Tabla 5.9: Resultados del reconocimiento en el conjunto de datos *MSR-Action3D* dividido por sujetos.

Estos resultados muestran la viabilidad el método propuesto. Utilizando un escenario *one-shot*, obtenemos una precisión cercana al 90 % cuando evaluamos la misma cantidad de gestos pero tomando en cuenta solamente un sujeto (que era inicialmente la idea de este método). La diferencia entre la precisión obtenida SA y CA es de apenas 6 % y la cantidad de información necesaria para clasificar los gestos CA es de 41 % en promedio en contraste con el 100 % de información necesaria para la clasificación SA, en total estamos ahorrando el 56 %, por lo que la mayoría de los gestos se puede clasificar hasta después de conocer el 41 % de su totalidad. Acerca de los tiempos de reconocimiento, se hablará más adelante.

5.4.3. Experimentos por subgrupos

Otro experimento que realizamos con el conjunto de datos *MSR-Action3D* es el propuesto en W. Li [2010], en donde dividen en tres subgrupos el conjunto de datos, separando los gestos por diferente complejidad: AS1 AS2 y AS3. En la Figura 5.6 se muestran los gestos de cada uno de los grupos, donde es importante resaltar que entre los gestos de los subgrupos AS1 y AS2 existe cierta similitud, mientras que en AS3 se englobaron los gestos más elaborados y más disímiles entre ellos. Por ejemplo, en el grupo AS1 se encuentran los movimientos *high throw* y *hammer*, que son movimientos bastante parecidos; en AS2 están los movimientos *high arm wave* y *hand catch*, estos movimientos también tienen bastante parecido entre ellos; finalmente el grupo AS3 tiene movimientos como *pickup&throw* que no se parece a *Jogging* o *forward kick*, por lo que son gestos más disímiles entre ellos.

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

Figura 5.6: Los subgrupos creados a partir del conjunto de datos *MSR-Action3D*. Figura obtenida de W. Li [2010].

En la Tabla 5.10 se muestran los resultados obtenidos usando nuestro método en la base de datos *MSR-Action3D* dividida en subgrupos. En las columnas 1, 4 y 7 (Gestos) se enlistan los índices de los gestos que son parte de cada grupo, en las columnas 2, 5 y 8 (SA) están los resultados del reconocimiento sin anticipación (SA) para los tres grupos, mientras que en las columnas 3, 6 y 9 se presentan los resultados por cada gesto usando reconocimiento con anticipación (CA), y en el título de estas columnas se especifica qué porcentaje (en promedio) del *gesto nuevo* se utilizó para llevar a cabo el reconocimiento. Podemos observar que el mejor resultado lo obtuvimos en el subgrupo AS3, que es el que contiene los gestos más diferentes entre sí y se considera el más sencillo de clasificar, en donde la precisión del reconocimiento SA y CA son iguales y son los más altos de los tres, en el grupo AS2 la diferencia entre la precisión del reconocimiento SA y CA es de apenas 3 %, mientras que en el subgrupo AS1 obtuvimos mejor precisión en el reconocimiento CA, creemos que esto se debe a que en algunas de las grabaciones de esta base de datos, los gestos tienen al final algo de ruido en los datos, lo que perjudica al clasificador que utiliza toda la información de los mismos.

Comparándonos con los resultados obtenidos en W. Li [2010] (en donde usaron esta misma evaluación), en el grupo AS1, AS2 y AS3 obtuvimos una precisión promedio de 46 %, 44 % y 50 % en la clasificación CA respectivamente y una precisión de 42 %, 47 % y 50 % en la clasificación SA respectivamente, con un promedio total de 46 % (en ambos tipos de clasificaciones). Estos resultados también están por debajo de los reportados por W. Li [2010] *et.al.* que son del 72 %, 71 % y 74 % con un promedio total de 74 %. De nuevo hay que resaltar que nuestra clasificación está basada en un escenario de entrenamiento *one-shot* y de un reconocimiento anticipado, mientras que en W. Li [2010] los autores utilizan un gran número de repeticiones para el entrenamiento de su modelo y no soporta la clasificación de todos los gestos sin división, es por esta razón que hicieron su evaluación dividien-

	AS1			AS2			AS3		
	Gestos	SA	CA (52 %)	Gestos	SA	CA (52 %)	Gestos	SA	CA (68 %)
	2	0.19	0.25	1	0.06	0.06	6	0.33	0.33
	3	0.31	0.25	4	0.63	0.63	14	0.33	0.75
	5	0.50	0.44	7	0.19	0.19	15	0.75	0.17
	6	0.50	0.56	8	0.25	0.31	16	0.25	0.50
	10	0.38	0.50	9	0.44	0.38	17	0.50	0.58
	13	0.81	0.75	11	0.81	0.56	18	0.33	0.75
	18	0.38	0.63	12	0.56	0.56	19	0.67	0.50
	20	0.31	0.31	14	0.81	0.81	20	0.83	0.42
	Prom.	0.42	0.46	Prom.	0.47	0.44	Prom.	0.50	0.50
W. Li [2010]->	Prom.	0.93	/	Prom.	0.93	/	Prom.	0.96	/

Tabla 5.10: Resultados obtenidos en el reconocimiento de gestos separados en subgrupos como propusieron en W. Li [2010].

do todos los gestos en subgrupos. Finalmente, nuestro método fue pensado para detectar gestos realizados por un solo sujeto, por lo que en la próxima sección exploraremos los resultados de nuestro método si hacemos esta consideración.

5.4.4. Experimentos por subgrupos y sujetos

Finalmente realizamos un experimento en el que dividimos la base de datos *MSR-Action3D* por sujeto y por grupo, esto con la intención de saber qué precisión tiene nuestro método con la división de gestos propuesta en W. Li [2010], y usando un solo sujeto que es como fue considerado su funcionamiento desde un inicio. Para realizar este experimento, se dividieron los gestos por grupo y luego por sujeto, tomamos el 30 % de los datos de cada grupo y generamos aleatoriamente diversos diccionarios y evaluamos el resto de los gestos de cada grupo con cada uno de los diccionarios obtenidos. Los resultados se muestran en la Tabla 5.11, donde las columnas 1, 4 y 7 (Gestos) tienen el listado de los gestos que componen cada grupo, las columnas 2, 5 y 8 (SA) tienen la precisión obtenida en el reconocimiento SA, mientras que las columnas 3, 6 y 9 (CA) tienen la precisión alcanzada con el reconocimiento CA, y el porcentaje promedio necesario del *gesto nuevo*, se encuentra en el título de dichas columnas. En este experimento se obtuvieron los mejores resultados en el subgrupo AS1, en donde la precisión del reconocimiento SA es del 97 % y la del reconocimiento CA es del 95 %, utilizando solamente el 43 % de la información del *gesto nuevo*. También podemos observar que la mayoría de los gestos se clasifican antes de rebasar su 50 % de ejecución, y la mayor diferencia

entre la precisión obtenida con el reconocimiento SA y CA es del 3 %, *i.e.* con sólo el 50 % de la información de los gestos estamos clasificando casi con la misma precisión que al usar el 100 % de los gestos .

AS1		
Gestos	SA	CA(43 %)
2	1.00	1.00
3	1.00	0.96
5	0.96	0.90
6	0.92	0.94
10	1.00	0.95
13	0.93	0.87
18	1.00	0.98
20	0.98	1.00
Prom.	0.97	0.95

AS2		
Gestos	SA	CA(50 %)
1	0.85	0.81
4	0.94	0.92
7	0.93	0.91
8	0.92	0.88
9	0.78	0.73
11	1.00	0.95
12	0.98	0.95
14	1.00	1.00
Prom.	0.93	0.89

AS3		
Gestos	SA	CA(46 %)
6	1.00	1.00
14	1.00	1.00
15	0.95	0.93
16	0.90	0.88
17	0.88	0.88
18	0.97	0.92
19	0.95	0.83
20	1.00	1.00
Prom.	0.96	0.93

Tabla 5.11: Resultados del reconocimiento dividido en sujetos y subgrupos.

5.4.5. Experimentos con el umbral de movimiento (μ_m)

Como explicamos en la Sección4.3.2, una variación de nuestro método es agregar un umbral de movimiento a las secuencias de tiempo de los gestos, de esta manera aquellas extremidades que no se muevan lo suficiente no serán tomadas en cuenta en la clasificación. Para realizar este experimento, utilizamos la mejor configuración de parámetros obtenida en los experimentos de la Sección 5.4.1. Probamos con los valores 5, 10, 20 y 30 para el umbral de movimiento μ_m. Los resultados se muestran en la Tabla 5.12: en las columnas 1-4 se enlistan los resultados obtenidos en la precisión promedio CA (fila 1), precisión promedio SA (fila 2), el porcentaje promedio necesario del gesto para el reconocimiento CA (fila 3), el promedio mínimo y máximo registrados (filas 4 y 5 respectivamente), y los milisegundos necesarios para clasificar un gesto CA y SA (filas 6 y 7 respectivamente). Observamos que las duraciones máxima y mínima (en las filas 4 y 5) no cambiaron durante todo el experimento, mientras que la precisión promedio CA y SA van disminuyendo conforme aumentamos el umbral de movimiento μ_m. Inicialmente se realizaron pruebas con los valores 10, 20 y 30 para el μ_m, pero al ver que la precisión disminuía cuando μ_m aumentaba, decidimos probar con el valor $\mu_m = 5$. Hay que resaltar que cuando $\mu_m = 20$, el tiempo promedio necesario para clasificar un gesto disminuye en 881 ms. para el reconocimiento CA y 964 ms. para el reconocimiento SA sacrificando apenas el 1 % de precisión. Si comparamos los resultados obtenidos con $\mu_m = 10$ con los mejores resultados obtenidos usando $\mu_m = 0$ (ver Sección 5.4.1), vemos que ambos lograron una precisión del 35 %, pero

al usar μ_m el tiempo promedio requerido para la anticipación CA y SA es de 2278 y 2488 ms. respectivamente, mientras que sin usar μ_m son de 3463 y3794 respectivamente, por lo que al usar μ_m se reducen en 1185 y 1306 ms. respectivamente, que es un tiempo bastante considerable.

	$\mu_m = 5$	$\mu_m = 10$	$\mu_m = 20$	$\mu_m = 30$
Precisión promedio CA	0.35	0.35	0.34	0.32
Precisión promedio SA	0.35	0.35	0.34	0.32
Porcentaje promedio	0.64	0.64	0.64	0.65
Porcentaje mínimo	0.20	0.20	0.20	0.20
Porcentaje máximo	1.00	1.00	1.00	1.00
Tiempo promedio CA en ms.	2051.00	2278.00	1397.00	1328.00
Tiempo promedio SA en ms.	2233.00	2488.00	1524.00	1454.00

Tabla 5.12: Resultados aplicando el umbral de movimiento μ_m en la base de datos *MSR-Action3D*.

Entonces tenemos que la precisión al usar umbrales de movimiento μ_m en la base de datos *MSR-Action3D* no se incrementa, sin embargo, el tiempo requerido para reconocer un gesto se reduce más del 34 %.

5.5. Conclusiones

El método propuesto es capaz de reconocer gestos antes de que el usuario los ejecute por completo cuando las bases de datos incluyen gestos realizadas por una sola persona. La pérdida más grande registrada durante los experimentos fue de 13 % en términos de precisión para las bases de datos *Dance* y *Dance2*, conociendo entre el 45 % y 53 % de la información del gesto en promedio; para *Dance* y *Dance2* logramos un reconocimiento en tiempo real, esto equivale a realizar una anticipación entre el 47 % y 55 % en el tiempo de respuesta de reconocimiento en promedio. Esto no se cumple para la base de datos *MSR-Action3D*, pues el tiempo de reconocimiento para este conjunto de datos es muy lento debido al gran número de clases de gestos que contiene.

El tiempo de respuesta de nuestro método depende directamente del número de gestos conocidos; en nuestros experimentos, el tiempo necesario para lograr una clasificación de gestos tradicional es en promedio 8 veces más grande que el requerido para hacer una clasificación anticipada.

Una de las desventajas de nuestro método es su dificultad para manejar una gran cantidad de gestos. El tiempo de respuesta de nuestro método está direc-

tamente ligado al tamaño que tiene el diccionario, y entre más grande sea éste, mayor será el tiempo requerido para responder. Esto podría causar que el tiempo de respuesta superara el límite de tiempo que exige una clasificación en tiempo real, provocando que la respuesta del clasificador no esté lista a tiempo.

Otra desventaja es la pérdida de precisión cuándo los gestos son realizados por varias personas diferentes. Nuestro método ofrece resultados muy favorables cuando la persona que realizó los gestos que se encuentran en el diccionario es la misma que realiza los *gestos nuevos*; en cambio, cuando la persona que realiza los *gestos nuevos* es diferente a la persona que realizó los gestos del diccionario, la precisión se ve afectada.

Al agregar el umbral de movimiento a nuestro método no conseguimos aumentar la precisión, sin embargo, se pudo disminuir el tiempo de reconocimiento de un gesto requerido. Además se comprobó que no todas las extremidades son necesarias para realizar un reconocimiento correcto del gesto.

En comparación con el trabajo deW. Li [2010] que reporta una precisión de 72 %, 71 % y 79 % en su experimento con los gestos divididos en subgrupos (AS1 AS2 y AS3), y en comparación con W. Jiang [2012] que reporta una precisión total del 88 %, nuestros resultados son bajos, pues en nuestra evaluación por grupos alcanzamos una precisión de 42 % 47 % y 50 % SA y de 46 % 44 % y 50 % CA, mientras que en nuestra evaluación con todos los gestos obtuvimos una precisión del 35 % SA y 34 % CA. Aquí es muy importante resaltar que aunque estamos por debajo de sus resultados, nosotros utilizamos un escenario *one-shot* para el entrenamiento por lo que basamos nuestra clasificación en un solo ejemplo de cada gesto. Además los autores antes mencionados utilizan una clasificación de gestos tradicional, por lo que no hacen ninguna anticipación como nosotros. Finalmente, si separamos los resultados por sujeto, obtenemos una precisión de 86 % CA y 93 % SA.

En todos los experimentos realizados, nuestro método CA estuvo muy cerca de la precisión alcanzada por nuestro método SA, donde la mayor pérdida de precisión reportada comparando nuestros dos métodos es de 13 % (en el conjunto de datos *Dance2*, ver Sección 6.4) y donde es posible clasificar correctamente un gesto conociendo únicamente el 25 % del mismo. A pesar de que el tiempo promedio necesario para clasificar un gesto es alto (ver Sección 5.4.5), es posible paralelizar nuestro método, de tal manera que es posible reducir en gran medida estos tiempos; en los trabajos anteriores esto no es posible, ya que despues del entrenamiento de los SOM's la clasificación solamente consiste en la verificación del código disperso que se obtiene del gesto entrante por lo que la parte más pesada de estos métodos es la parte del entrenamiento y no es posible paralelizarlo.

5.5.1. Resumen

En este capítulo describimos las características de los conjuntos de datos que utilizamos para realizar nuestros experimentos. Para los conjuntos *Dance* y *Dance2*, describimos una serie de experimentos realizados para obtener la mejor configuración de parámetros propios del método; también describimos los experimentos realizados con las variaciones de nuestro método, *i.e.* aplicando la distancia con pesos y aplicado el umbral de movimiento u_m. Para el conjunto de datos *MSR-Action3D* se presentaron los experimentos realizados con la mejor configuración de parámetros obtenidas de los experimentos con *Dance* y *Dance2*; también se realizaron experimentos dividiendo los gestos del conjunto de datos original en varios subgrupos, por sujetos, y por grupos y sujetos la mismo tiempo. Se analizaron y se expusieron los resultados comprobando que es posible identificar correctamente un gesto sin contar con toda la información sobre éste, pero hay que cuidar el balance entre la anticipación y la precisión alcanzadas.

En la Tabla 5.13. se muestra un resumen de los resultados obtenidos para cada conjunto de datos, con y sin el umbral de movimiento. Logramos un reconocimiento CA en tiempo real para las bases de datos *Dance* y *Dance2* con una precisión del 85 % y 100 % respectivamente, mientras que para MSR-Action3D (con gestos de varias personas) no se logró el reconocimiento anticipado y la precisión alcanzada fue muy baja (34 %).

	Dance	*Dance2*	*MSR-Action3D*	*Dance* (μ_m)	*Dance2*(μ_m)	*MSR-Action3D*(μ_m)
Precisión promedio CA	0.85	1.0	0.34	1.0	0.84	0.35
Precisión promedio SA	0.98	0.95	0.35	0.95	0.98	0.35
σ precisión promedio CA.	0.00	0.07		0.00	0.1	
Porcentaje promedio	0.53	0.45	0.54	0.41	0.54	0.64
Porcentaje mínimo	0.30	0.25	0.14	0.25	0.30	0.20
Porcentaje máximo	0.87	0.91	1.00	0.71	1.0	1.00
Tiempo prom. CA en ms.	19.26	668	3624	634	8	2051

Tabla 5.13: Comparación de los mejores resultados obtenidos para cada conjunto de datos con y sin el umbral de movimiento (μ_m).

No incluimos una comparación directa con los resultados de trabajos anteriores porque ninguno de estos trabajos ha realizado anticipación con gestos similares a los utilizados en el presente trabajo. Todos los trabajos anteriores manejan gestos en los que se utilizan una o dos extremidades, y en muy pocos gestos realizan movimientos simultáneos de sus extremidades, por lo que no nos pareció una comparación justa. Además, las bases de datos de estos trabajos aún no son públicas,

por lo que los autores se negaron a ofrecernos los conjuntos de datos que utilizaron y no nos fue posible evaluar sus gestos con nuestro método. Finalmente decidimos comparar nuestro método de reconocimiento de gestos SA con otros trabajos ya existentes, es por esta razón que realizamos experimentos con el conjunto de datos *MSR-Action3D*; la desventaja de este conjunto de datos es que incluye gestos de más de un sujeto, y nuestro método funciona adecuadamente cuando los gestos pertenecen a una sola persona. En consecuencia, obtuvimos resultados muy bajos en la clasificación. Como alternativa, separamos este conjunto de datos por sujetos e hicimos una evaluación por separado de cada uno, tratando de hacer más justa la comparación. Como resultado de esta separación, nuestro método mostró una mejora notoria en sus resultados.

Capítulo 6

Conclusiones, aportaciones y trabajo futuro

En este capítulo resumimos el contenido de la presente tesis, enlistamos las contribuciones resultantes de este trabajo, proponemos ideas para mejorar este método en un futuro y recalcamos las aportaciones logradas.

6.1. Síntesis de la tesis

A lo largo de esta tesis, que está enfocada en reconocer anticipadamente gestos corporales, hemos examinado detalladamente los escasos trabajos existentes que abordan este mismo problema, que solamente son cuatro A. Mori [2006], M. Kawashima [2011, 2010, 2009]. En el Capítulo 3 describimos detalladamente cada uno de ellos y los diferentes algoritmos o modelos en los que están basados, así como también enfatizamos las diferencias entre estos trabajos y el nuestro. Principalmente, estas diferencias son las siguientes:

- Nuestro clasificador es capaz de reconocer gestos más complejos que otros trabajos, ya que somos capaces de detectar gestos de cuerpo completo en los que las extremidades del cuerpo se pueden mover simultáneamente. Los gestos que forman parte de nuestro diccionario están conformados por 4 extremidades y una *caja torácica* que se mueven simultáneamente en todos los gestos, comparados con los gestos que únicamente involucran 2 extremidades utilizadas en A. Mori [2006] con pocos movimientos de extremidades simultáneas; o los gestos de 4 extremidades de M. Kawashima [2011] en donde únicamente se usan 2 con muy pocos movimientos simultáneos; o con las 4 extremidades utilizadas en M. Kawashima [2010, 2009] en donde la

mayoría de sus gestos solo utilizan una extremidad, en algunas 2 extremidades simultáneas e incluye solo algunos movimientos con el uso de todas las extremidades.

- Propusimos un método de clasificación anticipada basado en DTW, que difieren de los trabajos existentes. Usar DTW nos permite reconocer gestos de la misma clase aunque hayan sido ejecutados en diferentes tiempos.

- Nuestro método basado en DTW opera bajo un modelo de aprendizaje con escenario *one-shot*, reduciendo en gran cantidad el número de repeticiones necesarias para que nuestro método pueda funcionar y no requiere de una fase de entrenamiento. Ofrece buenos resultados cuando los gestos de entrenamiento y prueba pertenecen a una sola persona.

Posteriormente explicamos detenidamente todo el proceso de detección anticipada de nuestro método basado en DTW, en el que los gestos conocidos y el *gesto nuevo* son divididos en secuencias de tiempo, estas secuencias de tiempo son evaluadas individualmente para cada gesto de manera que obtenemos una distancia por cada uno, misma que utilizamos para calcular la probabilidad que tiene cada gesto de ser la respuesta final. El proceso de comparación entre el *gesto nuevo* y los gestos conocidos se lleva a cabo a través de varias iteraciones, en donde por cada una de éstas el clasificador realiza una predicción parcial; cuando alguna de estas predicciones parciales cumple con ciertas condiciones, entonces se determina una respuesta final. Como variación del método, usamos un umbral de movimiento, *i.e.* no todas las secuencias de tiempo se toman en cuenta para la clasificación, sino que se descartan aquellas que no presenten mucho movimiento y que por tanto no son relevantes para el reconocimiento.

Describimos además en la sección de experimentos los diferentes conjuntos de datos con los que probamos nuestro método, en donde dos de ellos fueron construidos por nosotros mismos y otro fue tomado de un trabajo ya existente. Con estos conjuntos de datos pudimos comparar la precisión y tiempos obtenidos con nuestro método, aplicando el reconocimiento sin anticipación (SA) y reconocimiento con anticipación (CA).

6.2. Conclusiones

Es posible detectar correctamente un gesto sin conocer toda su información. Nuestro método es capaz de detectar gestos de baile hasta con un 80 % de anticipación con poca pérdida de precisión.

DTW es un buen algoritmo para realizar reconocimiento de gestos. Sus propiedades nos permiten clasificar gestos con diferente duración de manera aceptable. Propusimos un método basado en DTW para reconocer anticipadamente diferentes gestos de cuerpo completo, utilizando un escenario de aprendizaje *one-shot*.

Nuestro método funciona mejor cuando clasificamos los gestos realizados por un único sujeto. En los experimentos observamos que al usar los gestos de un único sujeto en la base de datos *MSR-Action3D*, la precisión aumentó un 52 % en reconocimiento CA y 58 % en reconocimiento SA.

El método es capaz de ofrecer buenos resultados cuando sólo se usa un gesto de entrenamiento. Además, el método probó ser robusto ante la selección de ejemplos para conformar el diccionario, mostrando un mejor rendimiento cuando éstos son efectuados por una sola persona.

El tiempo para clasificar un gesto depende directamente del número de gestos en el diccionario. Pudimos observar que entre más grande es el número de gestos en el diccionario, más alto es el tiempo de clasificación de nuestro método.

Además, somos los primeros en clasificar anticipadamente gestos de cuerpo completo. Mientras que otros trabajos que abordan el problema de reconocimiento anticipado clasifican gestos en los que sólo se mueve una extremidad a la vez, ya sea de la parte superior o inferior, nosotros reconocemos gestos complejos en los que las extremidades del cuerpo se muevan libre y simultáneamente. Nuestro conjunto de datos está compuesto de gestos de baile como explicamos en la Sección 5.1, y obtenemos resultados con precisión mayor al 90 %.

Sin embargo, en la fase de experimentación se puede observar cómo nuestro método realiza el reconocimiento de gestos cada vez más lento conforme el diccionario contiene un mayor número de gestos, por lo que nuestro método está limitado en cuanto al número de gestos con los que puede lograr una clasificación anticipada.

Aunque está planteado como trabajo futuro mejorar esta característica, nuestro método solamente es capaz de reconocer anticipadamente gestos realizados por una sola persona. Al entrenar el diccionario con los gestos de un usuario y querer reconocer los mismos gestos de otros usuarios, nuestro método no ofrece buenos resultados. Como se presentó en el capítulo de experimentos, la precisión se reduce hasta en un 50 % al usar un conjunto de gestos realizados por varios usuarios.

Nuestro método también presenta algunos problemas al calcular correctamente el final de los gestos entrantes, por lo que en algunos casos no ofrece ninguna respuesta antes de que el gesto se termine de ejecutar; esto quiere decir que no siempre puede ofrecer una clasificación anticipada y esto se debe, como ya se mencionó, a que no se estima adecuadamente el final del gesto de entrada ; también puede deberse a que el gesto de entrada es muy similar a varios gestos en el

diccionario, por lo que no puede tomar una decisión a tiempo.

6.3. Contribuciones

Propusimos un método que funciona bajo el escenario de entrenamiento *one-shot*, que es capaz de reconocer anticipadamente gestos corporales con una anticipación considerable sin pérdida importante de precisión. No es necesaria una fase de entrenamiento ni numerosas repeticiones de cada clase de gesto.

Nuestro método es capaz de reconocer anticipadamente gestos de cuerpo completo que incluyen movimiento libre y simultáneo de todas las extremidades del cuerpo. Hasta donde sabemos, sólo existen trabajos que clasifican gestos corporales que involucran movimientos de una o dos extremidades.

Nuestro conjunto de datos está compuesto de gestos de baile como explicamos en la Sección 5.1, y obtenemos resultados con precisión mayor al 90 %.

Como resultado de esta tesis el artículo *One-shot DTW-Based Method for Early Gesture Recognition* será publicado próximamente en el mes de noviembre de 2013, en el décimo octavo congreso iberoamericano en reconocimiento de patrones (*18th Iberoamerican Congress on Pattern Recognition CIARP 2013*) .

6.4. Trabajo Futuro

Como trabajo futuro deseamos agregar un método de segmentación de gestos al clasificador, de tal manera que se pueda detectar el inicio y el final de un gesto automáticamente, permitiendo así realizar una clasificación *on line*.

Además, podríamos paralelizar nuestro método para reducir aún más los tiempos de respuesta. El algoritmo puede ser modificado para comparar todos los gestos conocidos con el *gesto nuevo* de manera simultánea, en lugar de hacerlo de manera secuencial como hace actualmente.

También deseamos lograr automatizar el aprendizaje de los parámetros L, μ_m, $minPer$, $maxPer$, γ y w para evitar establecerlos experimentalmente. Debido a que es un problema de maximización se pueden emplear técnicas como algoritmos genéticos o recocido simulado para encontrar los mejores valores para estos parámetros.

Asimismo, nuestro modelo no considera la existencia de zonas en los videos en los que el usuario no esté realizando ningún gesto. Para lograr exitosamente una segmentación de gestos en un video, debemos considerar que el usuario podría detenerse un momento y que en ese periodo de tiempo no realizará ninguno de

los gestos conocidos, por lo que el clasificador debería ser capaz de detectar estas zonas de "no gesto", como comúnmente se conocen, y reconocerlas como tal.

Finalmente, es posible mejorar el mecanismo utilizado para estimar la duración que tendrá el *gesto nuevo* en ambos métodos. Es importante estimar lo más cercanamente posible esta duración, porque si no se hace correctamente, podría causar que el clasificador no ofrezca una respuesta a tiempo, *i.e.*, el gesto es ejecutado por completo y el clasificador no arroja ninguna respuesta.

Bibliografía

J. Davis A. Bobick. The recognition of human movement using temporal templates. *Transactions on Pattern Analysis and Machine Intelligence*, pages 257–267, 2002.

R. Kurazume R. Taniguchi-T. Hasegawa H. Sakoe A. Mori, S. Uchida. Early recognition and prediction of gestures. *International Conference on Pattern Recognition (ICPR)*, pages 560–563, 2006.

F. La Rosa C. Costanzo, G. Iannizzotto. Virtualboard: Real-time visual gesture recognition for natural human-computer interaction. *International Parallel and Distributed Processing Symposium*, pages 112–120, 2003.

H. Jun-Da C. Yao-Jen, C. Shu-Fang. A kinect-based system for physical rehabilitation. *A Pilot Study for Young Adults with Motor Disabilities (RDD)*, pages 2566–2570, 2011.

D. Sturges F. Riley. *Ingeniería mecánica estática*. Ferné Olsina, 1995.

S. Golestan F. Soltani, F. Eskandari. Developing a gesture-based game for deaf/mute people using microsoft kinect. *Complex, Sixth International Conference on Intelligent and Software Intensive Systems (CISIS)*, pages 491–495, 2012.

Z. Liu G. Yu, J. Yuan. Predicting human activities using spatio-temporal structure of interest points. *Proceedings of the 20th ACM International Conference on Multimedia*, pages 1049–1052, 2012.

P. Jangyodsuk B. Hammer-H.J. Escalante I. Guyon, V. Athitsos. Chalearn gesture challenge. *Computer Vision and Pattern Recognition*, pages 1–6, 2012.

U. Kumar P.B. Prasad J.L. Raheja, R. Shyam. Real-time robotic hand control using hand gestures. *Second International Conference on Machine Learning and Computing (ICMLC)*, pages 12–16, 2010.

I. Jolliffe. *Principal component analisis*. Springer, 2da edition, 2002.

K. Daijin K. Daehwan, S. Jinyoung. Simultaneous gesture segmentation and recognition based on forward spotting accumulative hmms. *18th International Conference on Pattern Recognition (ICPR)*, pages 1231–1235, 2006.

R. Taniguchi M. Kawashima, A. Shimada. Early recognition of gesture patterns using sparse code of self-organizing map. *Lecture Notes in Computer Science*, pages 116– 123, 2009.

R. Taniguchi M. Kawashima, A. Shimada. Early recognition based on co-occurrence of gesture patterns. *Lecture Notes in Computer Science*, pages 431–438, 2010.

R. Taniguchi H. Nagahara M. Kawashima, A. Shimada. Adaptive template method for early recognition of gestures. *17th Korea-Japan Joint Workshop Frontiers of Computer Vision (FCV)*, pages 1–6, 2011.

H. Hoppe M. Raptis, D. Kirovski. Real-time classification of dance gestures from skeleton animation. *Symposium on Computer Animation 2011*, pages 147–156, 2011.

A. Tinku M. Sushmita. Gesture recognition: a survey. *IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews*, 37(3):311–324, 2007.

Microsoft. Kinect for windows sdk, 2013a. URL http://msdn.microsoft.com/en-us/library/hh855347.aspx.

Microsoft. Microsoft developer network, 2013b. URL http://msdn.microsoft.com/en-us/library/hh438998.aspx.

R. Zaman N. Ibraheem. Survey on various gesture recognition technologies and techniques. *International Journal of Computer Applications*, 50(7):38–44, 2012.

OpenNI. The standard cuadrowork for 3d sensing, 2013.

M. Dhanalakshmi R. Anbarasi, R. Hemavathy. Deaf-mute communication interpreter. *International Journal of Scientific Engineering and Technology*, 2(5): 336–341, 2013.

J. Rocha. Skeltrack, 2013. URL https://github.com/joaquimrocha/Skeltrack.

M. Ryoo. Human activity prediction: early recognition of ongoing activities from streaming videos. *International Conference on Computer Vision (ICCV)*, pages 1036–1043, 2011.

P. Senin. Dynamic time warping algorithm review. 2008.

L. Smith. A tutorial on principal components analysis. 2002.

C. Dorai D. Rajan-T. Chua L. Chia T. Cham, J. Cai. Advances in multimedia modeling. *13th International Multimedia Modeling Conference*, page 797, 2007.

D. Prattichizzo V. Frati. Using kinect for hand tracking and rendering in wearable haptics. *World Haptics Conference*, pages 317–321, 2011.

W. Ying T. Junsong W. Jiang, L. Zicheng. Mining actionlet ensemble for action recognition with depth cameras. *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1290 – 1297, 2012.

Z. Liu W. Li, Z. Zhang. Action recognition based on a bag of 3d points. *International Workshop on Computer Vision and Pattern Recognition*, pages 9–14, 2010.

W. Zhu X. Jian-Feng, M. Xie. Driver fatigue detection based on head gesture and perclos. *International Conference on Wavelet Active Media Technology and Information Processing (ICWAMTIP)*, pages 128–131, 2012.

J.K. Aggarwal X. Lu, C. Chia-Chih. Human detection using depth information by kinect. *Computer Vision and Pattern Recognition*, pages 15–22, 2011.

T. Ishii K. Hashimoto-K. Katamachi K. Noguchi N. Kakizaki D. Cai Y. Dai, Y. Shibata. An associate memory model of facial expressions and its applications in facial expression recognition of patients on bed. *IEEE International Conference on Multimedia and Expo (ICME)*, pages 591 – 594, 2001.

Z. Zhengyou. Microsoft kinect sensor and its effect. *MultiMedia*, 19:4–10, 2012.

Anexos

En nuestro método basado en DTW existen varios parámetros que es necesario configurar para que funcione adecuadamente. Estos parámetros son los siguientes:

- Número de gestos tomados en cuenta para la toma de decisiones (L).
- Número de desviaciones estándar γ para la toma de decisiones.
- Umbral mínimo ($minPer$) para la toma de decisiones parciales y finales.
- Umbral de decisión forzada $maxPer$.
- Tamaño de la ventana w.

Realizamos un análisis de cada una de estas variables y la influencia que tienen en el funcionamiento de nuestro método. Para seleccionar la mejor configuración de parámetros llevamos a cabo una evaluación sistemática: primero probamos varios valores en un parámetro y fijamos el mejor, para posteriormente elegir otro parámetro y probarlo de nuevo con varios valores para elegir y fijar el mejor; hicimos esto hasta seleccionar el mejor valor obtenido para todos los parámetros. Sabemos que la elección de parámetros de manera secuencial puede afectar el comportamiento de los parámetros de forma individual, es por ello que especificamos en la Sección 6.4 que aplicar un método de optimización para la configuración de estos parámetros es considerado como trabajo futuro. Los parámetros independientes fueron analizados primero, y aquellos dependientes fueron analizados al final. En la siguiente sección explicamos los experimentos y exponemos los resultados obtenidos para el parámetro L.

Número de gestos para la toma de decisiones (L)

Como mencionamos en la Sección 4.3.3, L es el número de gestos que no tomamos en cuenta en el cálculo de la desviación estándar σ y el promedio μ (que sirven para calcular n_σ) debido a que su probabilidad de ser la respuesta es muy baja. L

es una variable discreta que puede tomar valores desde 1 hasta $R-1$, donde R es el número de clases que hay en el conjunto de datos. Para evaluar L en el conjunto de datos *Dance*, elegimos aleatoriamente 70 % de los ejemplos para entrenamiento y el restante 30 % para las pruebas. Con el conjunto de entrenamiento formamos aleatoriamente 5 diccionarios diferentes de gestos (donde cada diccionario tiene un ejemplo de cada clase de gesto), asegurándonos de usar por lo menos una vez cada una de las repeticiones disponibles por gesto. Este conjunto de diccionarios fue el mismo que usamos en todos los experimentos de parámetros realizados para el conjunto de datos *Dance*. Para todos los experimentos, aquellos valores que se quieren minimizar están marcados con línea continua en las gráficas, mientras que el valor que se quiere maximizar, con línea punteada.

Estos experimentos fueron realizados para determinar la influencia que tiene el parámetro L en nuestro método y comprobar si este parámetro debe variar dependiendo de la composición del conjunto de datos. Para finalizar, buscamos encontrar y fijar el valor de L con el que se obtuvieran mejores resultados.

Lo que hicimos fue probar todos los posibles valores de L usando los diccionarios de entrenamiento y el conjunto de datos de prueba, *i.e.* repetimos el experimento 5 veces, uno por cada diccionario obtenido, usando la siguiente configuración de parámetros: $\gamma = 2.0$, $maxPer = 95$, $minPer = 50$,$w = 5$. Elegimos estos valores por las razones siguientes: el tamaño de ventana w se eligió tomando en cuenta el número de cuadros promedio de los gestos en el diccionario, de tal forma que por cada gesto se llevara a cabo por lo menos 5 evaluaciones parciales a lo largo de la ejecución del *gesto nuevo*, que no son demasiadas evaluaciones como para saturar al clasificador ni tan pocas como para dejar pasar demasiada información antes de evaluar el *gesto nuevo*; *maxPer* se estableció así para tratar de obligar al clasificador a darnos una respuesta antes de que el gesto haya terminado, por ello elegimos un valor muy cercano a 100; $maxPer$ y γ se eligieron de esa manera porque en experimentos previos se detectó que estos valores ofrecían buenos resultados (no se incluyeron en el presente documento porque se consideran irrelevantes).

En la Tabla 6.1 se muestran los resultados. La primera fila indica el valor de L, la segunda fila indica en porcentaje el valor asignado a L; recordemos que el valor de L sólo puede tomar los valores desde 1 hasta $R-1$, por esta razón evaluamos L hasta el valor 7. La fila 3 muestra la precisión promedio obtenida con el reconocimiento con anticipación (CA) y la fila 4 indica la precisión alcanzada con el reconocimiento sin anticipación (SA). Las filas 5-7 muestran el porcentaje promedio, mínimo y máximo respectivamente, donde el porcentaje mínimo es el porcentaje de información más pequeño necesario para lograr el reconocimiento de un gesto y el porcentaje máximo es el porcentaje más grande de información

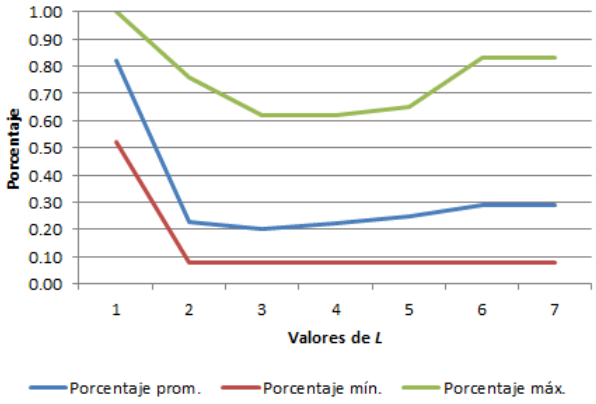
de un gesto necesario para el reconocimiento. El porcentaje promedio indica un estimado de la cantidad de información de un gesto que fue necesaria para realizar la clasificación.

L→	1	2	3	4	5	6	7
L en %→	12	25	37	50	62	75	87
Precisión prom. CA.	0.83±0.14	0.93±0.12	0.93±0.06	0.93±0.06	1.00±0.05	1.00±0.05	0.98±0.16
Precisión prom. SA.	0.95	0.95	0.95	0.95	0.95	0.95	0.95
Porcentaje prom.	0.81	0.44	0.41	0.40	0.41	0.41	0.41
Porcentaje mínimo	0.60	0.25	0.25	0.25	0.25	0.25	0.25
Porcentaje máximo	1.00	0.80	0.78	0.78	0.78	0.78	0.78

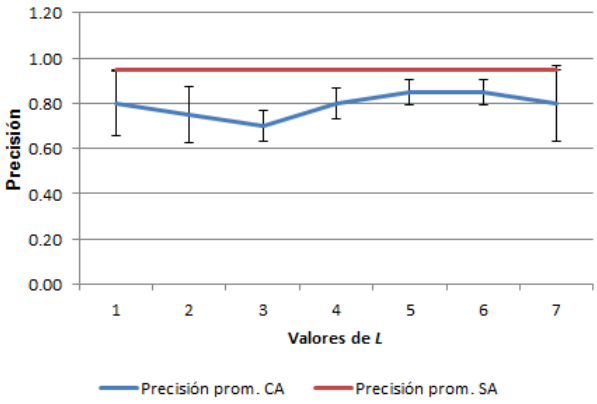
Tabla 6.1: Resultados obtenidos para los diferentes valores de L en la base de datos *Dance*.

En la Figura 6.1, se muestra más claramente el comportamiento de estos datos. Podemos observar que cuando la $L = 1$ el porcentaje máximo de gestos obtenida es de 100 % (ver Figura 6.1a), el porcentaje promedio está por encima del 80 % y el porcentaje mínimo por encima del 50 %. Esto quiere decir que hay gestos que no se alcanzan a anticipar, la mayoría de ellos se clasifica después del 80 % de su ejecución, y la clasificación más rápida fue después de que la mitad del gesto ya había sido ejecutada. Conforme va aumentando el valor de L, las líneas de comportamiento cambian, lo que buscamos es lograr un porcentaje mínimo, máximo y promedio lo más pequeños posible, mientras que buscamos maximizar la precisión obtenida. Debemos resaltar que hay un incremento notorio de la precisión cuando $L = 4$, alcanzando su valor máximo cuando $L = 6$; en este intervalo, observamos que el porcentaje mínimo se mantiene, mientras que el porcentaje promedio se reduce a su mínimo en $L = 3$ y alcanza su máximo en $L = 6$, mientras que el porcentaje máximo alcanza su mínimo cuando $L = 3$ y su máximo en $L = 6$ (si ignoramos el peor valor cuando $L = 1$). Cuando $L = 5$ ganamos mucha precisión (ver Figura 6.1b) y se presenta un ligero aumento en el porcentaje promedio y máximo, por lo que lo seleccionamos como el mejor valor para L, que es equivalente al 62 % de los gestos. La precisión obtenida con anticipación (CA) no es muy diferente de la precisión obtenida SA, la diferencia mínima entre estas es de 10 %.

Para el conjunto de datos *Dance2* realizamos la misma evaluación sistemática para establecer los mejores parámetros. Lo que hicimos fue seleccionar el 70 % de los gestos como entrenamiento y el 30 % restante como prueba; del conjunto de prueba extrajimos 10 diccionarios aleatorios diferentes (estos diccionarios sólo contienen un ejemplo de cada gesto), y utilizamos estos mismos diccionarios para



(a)



(b)

Figura 6.1: Precisión CA y SA, porcentaje máximo, mínimo y promedio alcanzados usando los diferentes valores de L en la base de datos *Dance*.

realizar todos los experimentos con parámetros. En este conjunto de datos se pudieron extraer más diccionarios de prueba, debido al mayor número de repeticiones por gesto.

Este experimento se realizó para fijar el valor de L con el que se obtienen los mejores resultados en el reconocimiento, para el conjunto de datos *Dance2*. El mejor valor obtenido será utilizado en experimentos posteriores. Los resultados obtenidos se muestran en la Tabla 6.2, los valores de L van desde 1 hasta 3, ya que sólo tenemos 4 gestos y $1 \leq L < R$. Los parámetros utilizados fueron los siguientes: $\gamma = 2.0$, $minPer = 50$,$maxPer = 95$, $w = 5$. Al igual que en la Tabla 6.1, la fila 2 contiene el valor de L en promedio, las filas 3 y 4 contienen la precisión promedio para el reconocimiento CA y SA respectivamente. Finalmente, las filas

5-7 muestran el porcentaje promedio, mínimo y máximo de información de un gesto necesaria para poder realizar el reconocimiento.

L→	1	2	3
L en %→	25	50	75
Precisión prom. CA.	0.86±0.20	0.86±0.07	0.86±0.13
Precisión prom. SA.	0.98	0.98	0.98
Porcentaje prom.	0.91	0.56	0.65
Porcentaje mín.	0.56	0.30	0.31
Porcentaje máx.	1.00	1.00	1.00

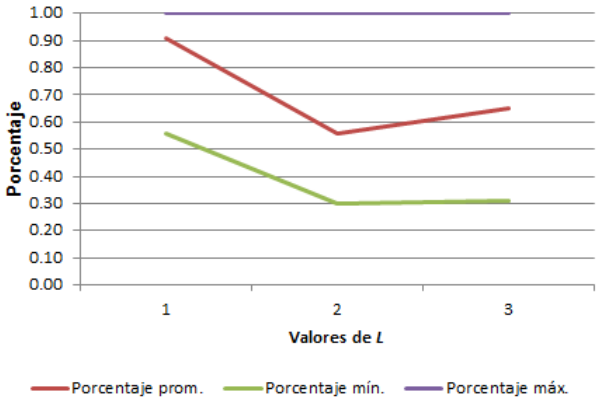
Tabla 6.2: Resultados obtenidos para los diferentes valores de L en la base de datos *Dance2*.

En la Figura 6.2 se puede apreciar mejor el comportamiento de las variables. La precisión promedio y el porcentaje máximo (ver Figura 6.2a) se mantuvieron durante los diferentes valores de L; cuando $L = 2$ el porcentaje máximo y promedio se reducen más que cuando $L = 1$ y $L = 3$, por lo que el mejor valor para L es 2 (correspondiente al 50 % de gestos). El porcentaje máximo cuando $L = 2$ es de 100 % por lo que hay algunos gestos que no se alcanzan a reconocer anticipadamente, el porcentaje mínimo de información necesaria para el reconocimiento es de 30 %, mientras que el porcentaje promedio es de 56 % (*i.e.* la mayoría de los gestos se clasifican antes de conocer el 56 % de su totalidad). Observemos que la precisión obtenida CA se acerca mucho a la precisión SA obtenida, donde ambas se mantuvieron igual a lo largo de todo el experimento.

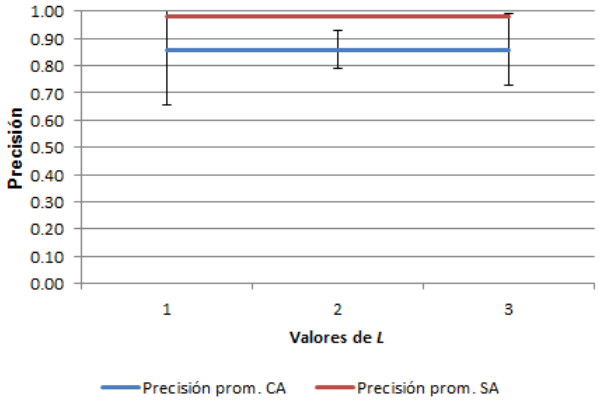
Si comparamos los resultados obtenidos para *Dance* y *Dance2* el mejor valor de L es parecido (ambos están cerca del 50 %, aunque esto puede variar en conjuntos de datos en los que los gestos tengan una duración muy diferente), mientras que la precisión obtenida fue de 100 % y de 86 % respectivamente. El porcentaje de información que nos ahorramos en *Dance* es del 59 % mientras que en *Dance2* es de 44 %. La diferencia entre estos resultados puede deberse al diferente número de gestos que hay en cada conjunto de datos y su diferente duración.

Umbral de separación para clasificación (γ)

Este experimento fue realizado para determinar el impacto que tiene este parámetro en nuestro método y fijar el valor de γ con el que se obtienen mejores resultados en la clasificación; una vez que este valor sea encontrado, será utilizado en experimentos siguientes. Recordemos que γ es un límite que n_σ debe superar para que el clasificador tome una decisión final, donde n_σ es el número de



(a)



(b)

Figura 6.2: Precisión CA. y SA., porcentaje máximo, mínimo y promedio alcanzados usando los diferentes valores de L en la base de datos *Dance2*.

desviaciones estándar que caben en la diferencia entre la probabilidad del mejor gesto y la probabilidad del segundo mejor gesto (ver Sección 4.3.3). El método de evaluación utilizado fue el mismo que en el experimento para el parámetro L, se utilizaron los mismos 5 diccionarios que fueron generados aleatoriamente para correr el experimento 5 veces, el valor de γ se fue variando de uno en uno. Los parámetros utilizados para este experimento fueron configurados como sigue: $L = 5$, $maxPer = 95$, $minPer = 50$, $w = 5$. En la Tabla 6.3 se muestran los resultados. Esta tabla contiene en la primera fila los valores asignados a γ, la fila 2 contiene la precisión promedio obtenida en el reconocimiento CA para los diferentes valores de γ y la fila 3 la precisión promedio del reconocimiento SA. Las filas 4-6 muestran el porcentaje promedio, máximo y mínimo respectivamente (*i.e.* el promedio del

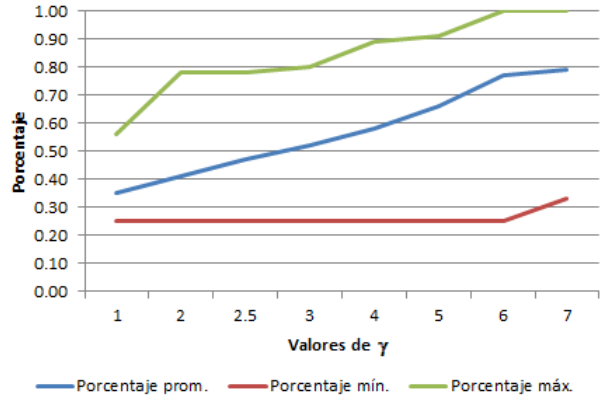
porcentaje de información de los gestos necesaria para la clasificación).

$\gamma \rightarrow$	1	2	2.5	3	4	5	6	7
Precisión promedio CA.	0.90	1.00	1.00	1.00	0.98	0.95	0.90	0.88
Precisión promedio SA.	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95
σ promedio CA	0.10	0.00	0.00	0.00	0.06	0.07	0.10	0.13
Porcentaje promedio	0.35	0.41	0.47	0.52	0.58	0.66	0.77	0.79
Porcentaje mínimo	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.33
Porcentaje máximo	0.56	0.78	0.78	0.80	0.89	0.91	1.00	1.00

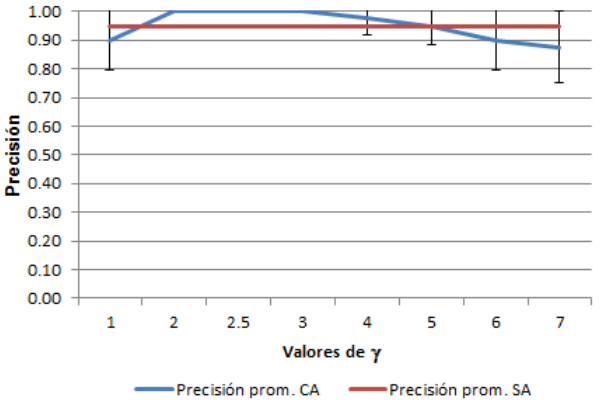
Tabla 6.3: Resultados obtenidos para los diferentes valores de γ en la base de datos *Dance*.

En la Figura 6.3 se observa mejor el comportamiento de nuestro método al variar γ. Podemos ver que el porcentaje mínimo mantiene su valor hasta que $\gamma = 7$ (ver Figura 6.3a), mientras que los porcentajes máximo y promedio siguen subiendo hasta que $\gamma = 7$. La precisión máxima es alcanzada cuando $\gamma = 2$ y $\gamma = 3$ (ver Figura 6.3b), por lo que incluimos un valor intermedio $\gamma = 2.5$, donde de igual manera se alcanzó una precisión del 100 %. Sin embargo, lo que buscamos es minimizar la cantidad de información del gesto necesaria para realizar el reconocimiento (porcentaje máximo, mínimo y promedio), esto se logra cuando $\gamma = 2$, por lo que ese es el mejor valor para γ. En este caso, entre más separación deba haber entre las probabilidades de los dos mejores gestos (es decir entre más grande sea γ), el clasificador tendrá que esperar cada vez más tiempo para tomar su decisión y es posible que las probabilidades de estos gestos no tengan una separación de probabilidades tan grande, por lo que se corre el riesgo de clasificar tardíamente el gesto. En la Figura 6.3a podemos ver como cuanto más aumenta el valor de γ, más información del gesto es necesaria.

Para el conjunto de datos *Dance2* se utilizaron los 10 diccionarios generados aleatoriamente como entrenamiento y se probó con el conjunto de datos de prueba descrito al inicio de este capítulo. Los resultados obtenidos con los diferentes valores de γ se muestran en la Tabla 6.4. Los parámetros utilizados fueron los siguientes: $L = 2$,$minPer = 50$,$maxPer = 95$, $w = 5$. De igual manera que en el experimento anterior, los valores de γ fueron incrementados de uno en uno incluyendo $\gamma = 2.5$ porque se obtuvieron resultados interesantes para los valores $\gamma = 2$ y $\gamma = 3$, esto nos sirvió para detallar más este intervalo de valores. En la fila 2 y 3 se muestran la precisión obtenida para el reconocimiento CA y SA respectivamente, la fila 3 contiene la desviación estándar de los datos de reconocimiento CA, finalmente las filas 5-7 muestran los porcentajes promedio, mínimo y máximo obtenidos.



(a)



(b)

Figura 6.3: Precisión CA y SA, porcentaje máximo, mínimo y promedio alcanzados usando los diferentes valores de γ en la base de datos *Dance*.

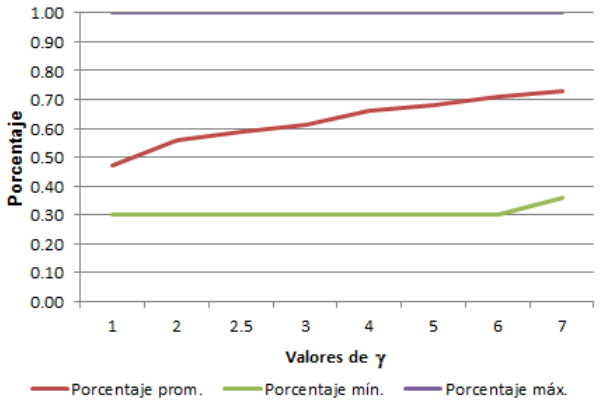
En la Figura 6.4, podemos ver mejor el comportamiento de las diferentes variables. Primero, el porcentaje máximo durante todos los valores de γ se mantiene en 100 % (ver Figura 6.4a) y el valor del porcentaje mínimo se mantiene igual hasta $\gamma = 7$. Aunque el valor más alto de precisión promedio se obtiene cuando $\gamma = 3$ (ver Figura 6.4b), cuando $\gamma = 2$ se obtiene un valor muy similar sin que el porcentaje promedio se eleve demasiado, por lo que el mejor valor para γ es 2.

Al comparar los resultados obtenidos para ambos conjuntos, vemos que el mejor valor para γ en ambos casos es 2, obteniendo una precisión máxima de 100 % y 86 % para *Dance* y *Dance2* respectivamente. En cuanto al porcentaje de información de los gestos necesario para el reconocimiento, para *Dance* es de 41 % mientras que para *Dance2* es de 56 %. Como mencionamos en los experimentos

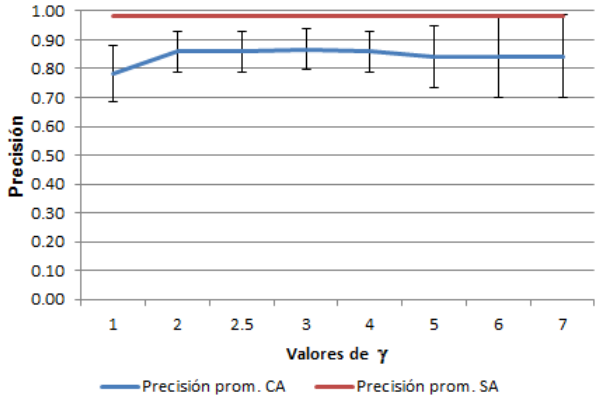
$\gamma \rightarrow$	1	2	2.5	3	4	5	6	7
Precisión promedio CA.	0.78	0.86	0.86	0.87	0.86	0.84	0.84	0.84
Precisión promedio SA.	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98
σ precisión promedio CA.	0.10	0.07	0.07	0.07	0.07	0.11	0.14	0.14

Porcentaje promedio	0.47	0.56	0.59	0.61	0.66	0.68	0.71	0.73
Porcentaje mínimo	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.36
Porcentaje máximo	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

Tabla 6.4: Resultados obtenidos para los diferentes valores de γ en la base de datos *Dance2*.



(a)



(b)

Figura 6.4: Precisión CA y SA, porcentaje máximo, mínimo y promedio alcanzados usando los diferentes valores de γ en la base de datos *Dance2*.

para el parámetro L, la duración, tipo y cantidad de gestos para cada conjunto de datos es diferente, la complejidad de ambas radica en diferentes aspectos (en un conjunto son pocos gestos y muchas repeticiones, y en otro conjunto son muchos gestos y pocas repeticiones). En ambos experimentos se puede ver cómo la precisión aumenta para los valores intermedios de γ y desciende en sus valores extremos, esto es porque si ponemos un valor muy pequeño de separación entre probabilidades, se tomará una decisión muy rápida sin la suficiente información sobre el gesto para hacerlo correctamente, mientras que si la separación entre probabilidades es muy grande sucede lo contrario. En la siguiente sección se presentan los experimentos realizados con el parámetro *minPer*.

Umbral mínimo para tomar decisiones (*minPer)*

Este experimento fue realizado para determinar el impacto que tiene este parámetro en nuestro método y para fijar el valor de *minPer* con el que obtenemos mejores resultados en el reconocimiento. Recordemos que *minPer* es el parámetro que le indica al clasificador cuál es el porcentaje del gesto que tiene que haberse ejecutado para que empiece a tomar decisiones parciales y finales (el valor que arroje los mejores resultados será utilizado en los siguientes experimentos). Al igual que en las Secciones 6.4 y 6.4, se utilizaron los 5 diccionarios aleatorios para repetir 5 veces el experimento, y se probó con el conjunto asignado para las pruebas. Los resultados se muestran en la Tabla 6.5. Aquí los valores de *minPer* van desde el 10 hasta el 100 porque *minPer* representa un porcentaje, los fuimos variando en saltos de 10 unidades para poder ver un cambio notorio en los resultados. Los valores de los parámetros fueron los siguientes: $L = 5$, $\gamma = 2.0$, $maxPer = 95$, $w = 5$. La fila 1 contiene los valores con los que se probó *minPer*, las filas están organizadas como en los experimentos con γ.

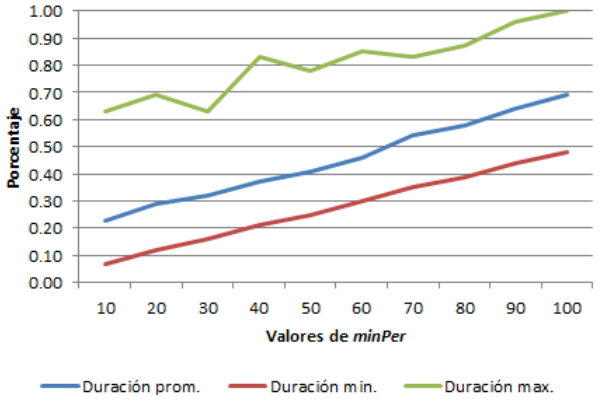
$minPer \rightarrow$	10	20	30	40	50	60	70	80	90	100
Precisión prom. CA.	0.83	0.93	0.95	0.98	1.00	0.98	0.98	1.00	0.98	0.95
Precisión prom. SA.	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95
σprecisión prom. CA.	0.07	0.11	0.11	0.06	0.00	0.06	0.06	0.00	0.06	0.07

Porcentaje prom.	0.23	0.29	0.32	0.37	0.41	0.46	0.54	0.58	0.64	0.69
Porcentaje mín.	0.07	0.12	0.16	0.21	0.25	0.30	0.35	0.39	0.44	0.48
Porcentaje máx.	0.63	0.69	0.63	0.83	0.78	0.85	0.83	0.87	0.96	1.00

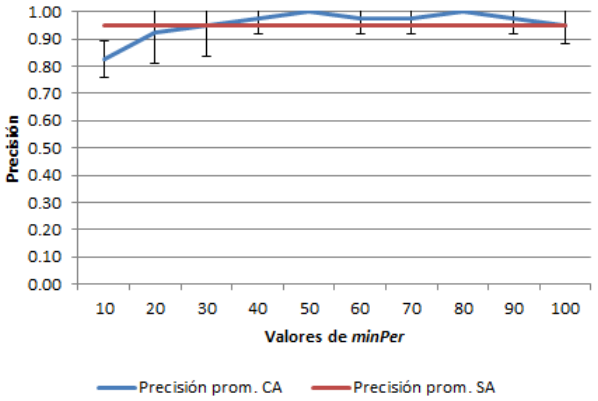
Tabla 6.5: Resultados obtenidos para los diferentes valores de *minPer* en la base de datos *Dance*.

En la Figura 6.5, se puede ver de manera más clara el comportamiento de

las variables. Podemos ver cómo el porcentaje máximo de la clasificación se ve directamente afectado al mover este parámetro, entre más grande sea *minPer*, mayor será el porcentaje mínimo de la clasificación. En este experimento, cuando $minPer = 50$ se alcanza la precisión máxima, y se vuelve a alcanzar cuando $minPer = 80$. Sin embargo, la primera opción es mejor porque el porcentaje promedio, mínimo y máximo son más pequeños.



(a)



(b)

Figura 6.5: Precisión CA y SA, porcentaje máximo, mínimo y promedio alcanzados usando los diferentes valores de *minPer* en la base de datos *Dance*.

Para el conjunto de datos *Dance2*, se usaron los 10 diccionarios generados aleatoriamente para probar y se probó con el conjunto de datos asignado para pruebas. En la Tabla 6.6 se muestran los resultados obtenidos. Las filas están organizadas como en los experimentos con γ. La forma de elegir los valores a evaluar de *minPer*, es la misma que en el experimento anterior. Los parámetros

utilizados para este experimento fueron los siguientes: $L = 2$,$\gamma = 2.0$,$maxPer = 95$, $w = 5$.

$minPer \rightarrow$	10	20	30	40	50	60	70	80	90	100
Precisión prom. CA.	0.70	0.68	0.76	0.79	0.86	0.86	0.90	0.92	0.93	0.94
Precisión prom. SA.	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98
σ precisión prom. CA.	0.08	0.10	0.12	0.10	0.06	0.08	0.07	0.08	0.08	0.07

Porcentaje prom.	0.34	0.37	0.43	0.50	0.56	0.57	0.66	0.72	0.76	0.77
Porcentaje mín.	0.13	0.17	0.21	0.26	0.30	0.39	0.43	0.47	0.52	0.52
Porcentaje máx.	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

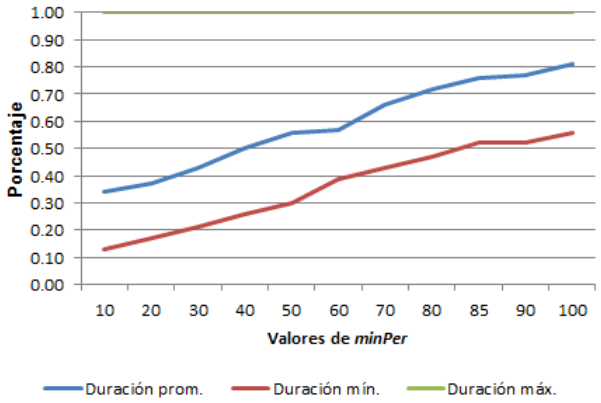
Tabla 6.6: Resultados obtenidos para los diferentes valores de *minPer* en la base de datos *Dance2*.

En la Figura 6.6 se muestra el porcentaje promedio, mínimo y máximo obtenidos durante el experimento. Podemos observar que la porcentaje máximo se mantiene durante todos los valores de *minPer* (ver Figura 6.6a); el valor de precisión promedio alcanza su valor máximo cuando $minPer = 100$ (ver Figura 6.6b), pero no podemos elegir este valor porque entonces no habría anticipación. El mayor incremento del porcentaje promedio inicia desde $minPer = 10$ hasta $minPer = 50$, en este último valor se obtiene el mayor valor también para la precisión promedio, además desde $minPer = 50$ hasta $minPer = 60$, el porcentaje promedio se incrementa muy ligeramente, y aunque el porcentaje mínimo se incrementa en mayor cantidad, aún está por debajo del 50 %, es por esta razón que el mejor valor para *minPer* es 50.

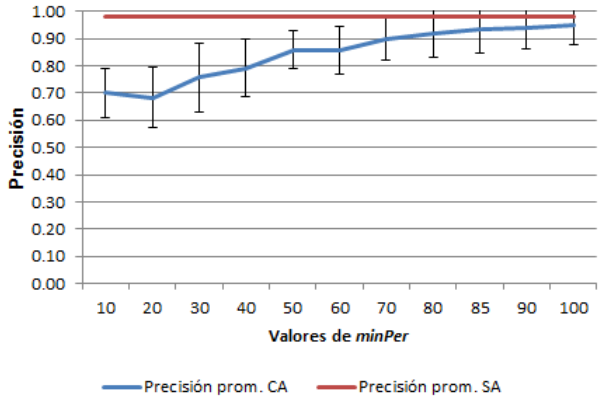
Comparando los resultados obtenidos para ambos conjuntos de datos, el mejor valor elegido para *minPer* en ambos casos es de 50, las precisiones máximas logradas al utilizar el valor de *minPer* elegido fueron de 30 % y 86 % respectivamente. Estas precisiones pueden subir una vez establecidos correctamente el resto de los parámetros. En la siguiente sección explicaremos los resultados obtenidos con el parámetro *maxPer*.

Umbral de decisión forzada (*maxPer)*

Recordemos que *maxPer* es el porcentaje del gesto que se debe haber ejecutado para que el clasificador arroje una respuesta forzada. Este experimento fue realizado para analizar el desempeño de nuestro método al variar el valor de *maxPer* y para fijar el valor de *maxPer* con el que se tienen mejores resultados en el reconocimiento (el mejor valor encontrado será utilizado en los siguientes experimentos).



(a)



(b)

Figura 6.6: Precisión SA y CA, porcentaje máximo, mínimo y promedio alcanzados usando los diferentes valores de *minPer* en la base de datos *Dance2*.

Se utilizaron los mismos 5 diccionarios generados aleatoriamente que fueron utilizados en los experimentos de las Secciones 6.4,6.4 y 6.4. El parámetro *maxPer* representa un porcentaje, por lo que probamos los valores desde el 10 hasta el 100, fuimos variando en saltos de 10 unidades para poder ver un cambio notorio en los resultados. Los valores de los parámetros fueron los siguientes: $L = 5$, $\gamma = 2.0$, $minPer = 50$, $w = 5$. Los resultados se muestran en la Tabla 6.7. Las filas están organizadas como en los experimentos con $minPer$.

En la Figura 6.7 se puede ver mejor el comportamiento de las variables. Los valores de las duraciones máxima, mínima y promedio, se mantienen hasta $maxPer = 40$ porque recordemos que el valor de $minPer = 50$, por lo que todas las decisiones se toman antes de que el 50 % de los gestos haya sido ejecuta-

$maxPer \rightarrow$	10	20	30	40	50	60	70	80	90	100
Precisión prom. CA.	0.90	0.90	0.90	0.98	0.98	0.98	0.98	1.00	1.00	1.00
Precisión prom. SA.	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95	0.95
σ precisión prom. CA.	0.10	0.10	0.10	0.06	0.06	0.06	0.06	0.00	0.00	0.00
Porcentaje prom.edio	0.35	0.35	0.35	0.36	0.37	0.39	0.40	0.40	0.40	0.41
Porcentaje mínimo	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25
Porcentaje máximo	0.56	0.56	0.56	0.56	0.56	0.60	0.66	0.78	0.78	0.90

Tabla 6.7: Resultados obtenidos para los diferentes valores de *maxPer* en la base de datos *Dance*.

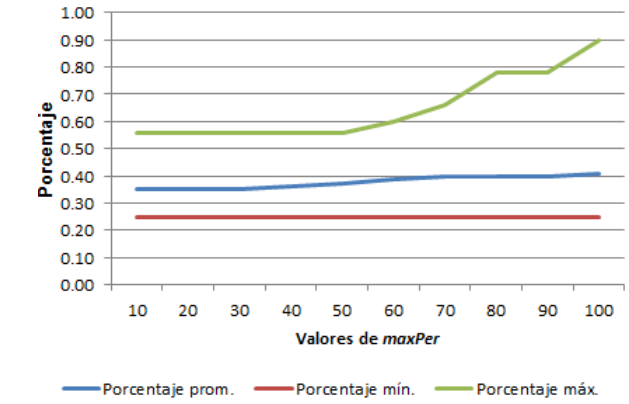
do. Vemos que aunque $maxPer = 100$, el porcentaje máximo registrado no llega al 100 %, esto es porque el porcentaje que ya fue ejecutado de los gestos es estimado y el que presentamos en las gráficas es el real, esta parte la mencionamos en el Capítulo 4.3.3. Desde $maxPer = 80$ hasta $maxPer = 100$, se alcanza la precisión más alta, pero cuando $maxPer = 80$, el porcentaje máximo y promedio son menores que cuando $maxPer = 100$, por lo que el mejor valor para *maxPer* es 80.

Para el conjunto de datos *Dance2* se utilizaron los 10 diccionarios generados aleatoriamente y se probó con el conjunto de datos destinado para ello.Los parámetros utilizados para este experimento fueron los siguientes: $L = 2$,$\gamma = 2.0$,$minPer = 50$, $w = 5$. En la tabla 6.8, se muestran los resultados obtenidos. El ordenamiento de las filas de esta tabla es el mismo que en el experimento anterior.

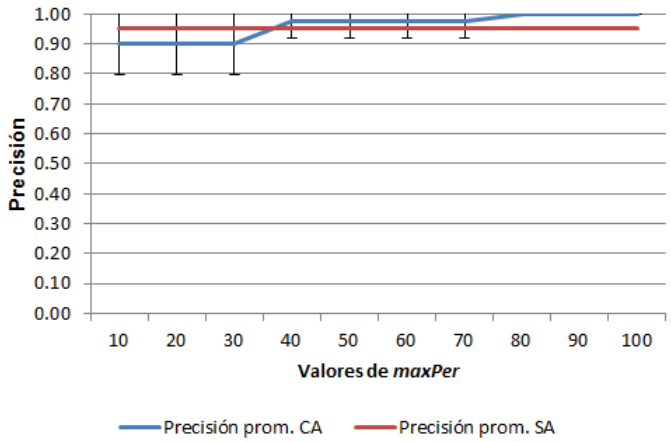
$maxPer \rightarrow$	10	20	30	40	50	60	70	80	90	100
Precisión prom. CA.	0.73	0.73	0.73	0.74	0.78	0.83	0.85	0.86	0.86	0.86
Precisión prom. SA.	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98
σ precisión prom. CA.	0.08	0.08	0.08	0.09	0.07	0.06	0.06	0.06	0.06	0.06
Porcentaje prom.	0.43	0.43	0.43	0.45	0.47	0.52	0.53	0.53	0.54	0.55
Porcentaje mín.	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30	0.30
Porcentaje máx.	0.60	0.60	0.60	0.75	0.80	0.87	0.87	1.00	1.00	1.00

Tabla 6.8: Resultados obtenidos para los diferentes valores de *maxPer* en la base de datos *Dance*2.

En la Figura 6.8 podemos ver más claramente el comportamiento del porcentaje máximo, mínimo y promedio obtenidos, así como también de la precisión promedio. Podemos observar que cuando $maxPer = 70$ el porcentaje máximo alcanza el 100 %, lo que quiere decir que hay algunos gestos que no se pueden clasificar con anticipación, mientras que cuando $maxPer = 60$ el porcentaje máximo



(a)

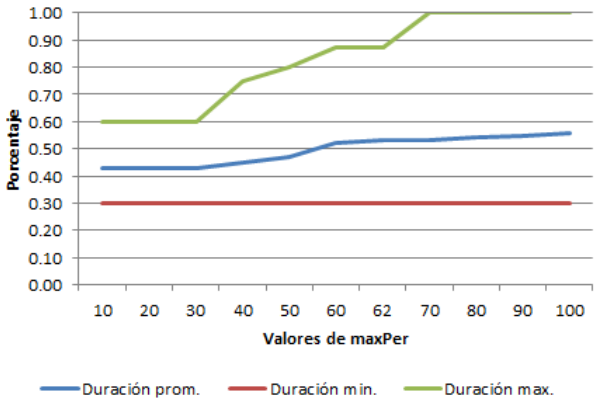


(b)

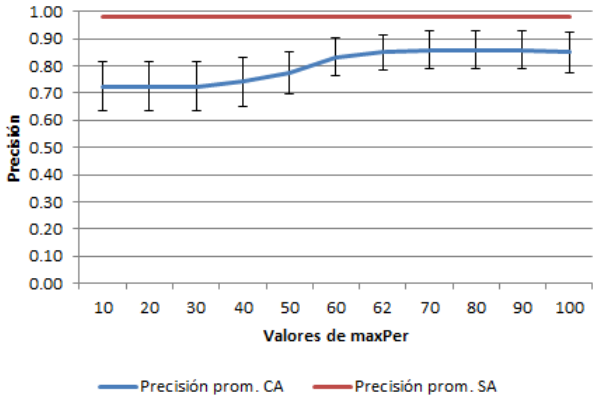
Figura 6.7: Precisión CA y SA, porcentaje máximo, mínimo y promedio alcanzados usando los diferentes valores de *maxPer* en la base de datos *Dance*.

alcanzado es de 82 %, es por esta razón que exploramos más a detalle este intervalo de valores de *maxPer*, encontrando que cuando su valor es 62, el porcentaje máximo se mantiene en 87 % mientras que si su valor es 63, sube a 100 %; sin embargo en el intervalo en el que $maxPer = 62$ y $maxPer = 70$, ni el porcentaje mínimo ni el porcentaje promedio se incrementan en gran medida, pero se obtiene un ligero aumento en la precisión promedio por lo que el mejor valor para *maxPer* es 62. Este valor es diferente del mejor valor obtenido para la base de datos *Dance*, pero en ambas pruebas podemos ver que entre más grande es *maxPer* más precisión alcanzamos y menos anticipación obtenemos, entonces debemos buscar un equilibrio entre estos dos factores a la hora de elegir nuestro valor de *maxPer*

para el conjunto de gestos elegido.



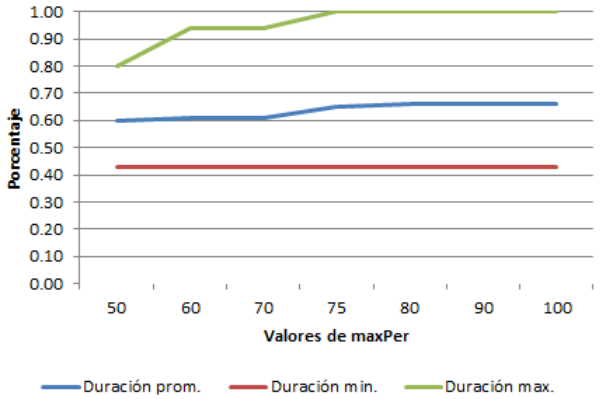
(a)



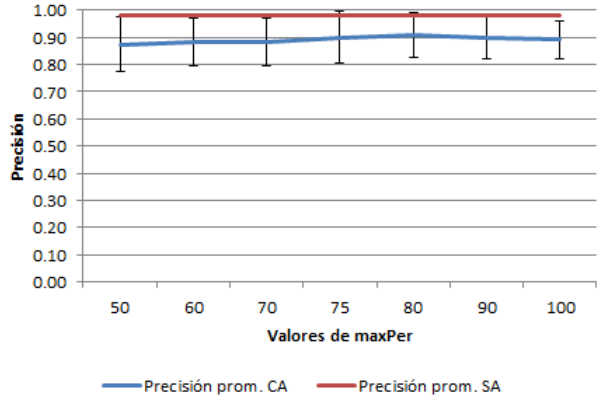
(b)

Figura 6.8: Precisión CA y SA, porcentaje máximo, mínimo y promedio alcanzados usando los diferentes valores de *maxPer* en la base de datos *Dance2*.

Ahora repetimos este experimento para $minPer = 70$ que es cuando la precisión en el experimento anterior rebasó el 90 %. En la Figura 6.9, se muestran los resultados obtenidos a partir de $maxPer = 50$. Cuando $maxPer = 80$ con $minPer = 70$ se alcanza una precisión del 91 %, y aunque el porcentaje máximo es de 100 %, la duración promedio es de 66 %, esto quiere decir que la mayoría de los gestos son clasificados alrededor de su 66 % de ejecución, donde el mínimo porcentaje registrado es de 43 %.



(a)

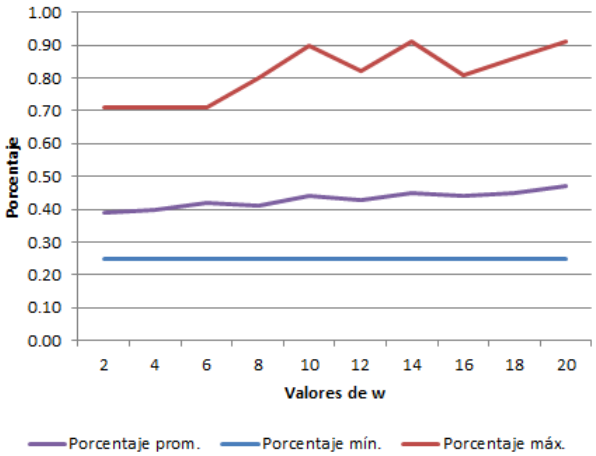


(b)

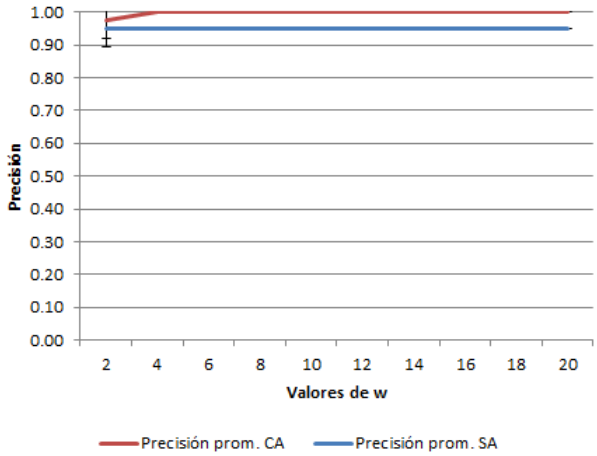
Figura 6.9: Precisión, porcentaje máximo, mínimo y promedio alcanzados usando diferentes valores de *maxPer* a partir de 50, y con un valor $minPer = 80$ en la base de datos *Dance2*.

Tamaño de ventana (w)

Realizamos el siguiente experimento para determinar la influencia que tiene este parámetro en nuestro método y para fijar el valor de w que nos arrojara los mejores resultados en la clasificación anticipada, en las dos bases de datos (el parámetro que arroje el mejor resultado será utilizado en experimentos posteriores). La configuración de parámetros utilizada para *Dance* fue la siguiente: $L = 5$, $\gamma = 2.0$, $minPer = 50$, $maxPer = 80$. Los resultados se muestran en la Figura 6.10. En este caso, el porcentaje máximo no llega hasta el 100 %, aún así observamos un incremento grande en el intervalo desde $w = 6$ hasta $w = 8$, pero también



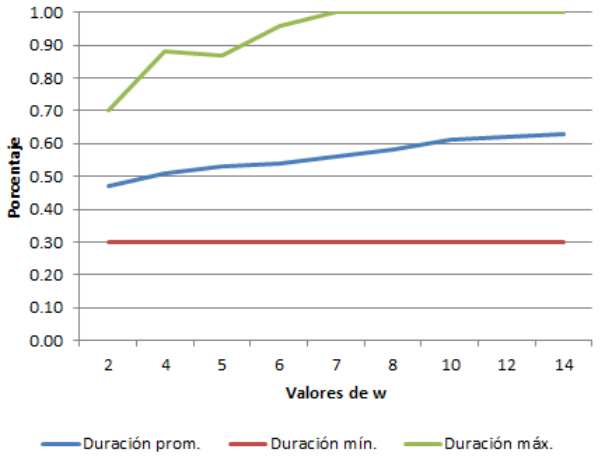
(a)



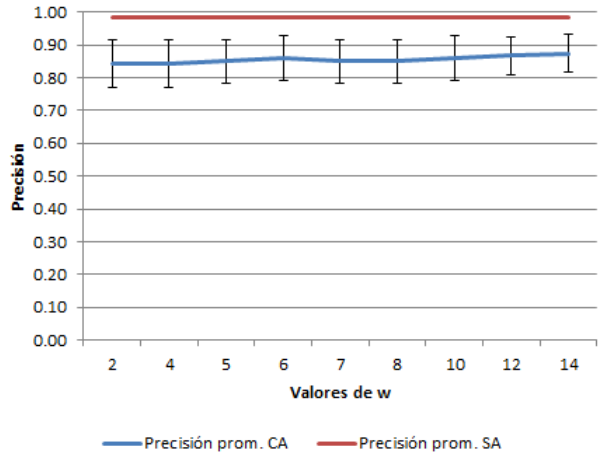
(b)

Figura 6.10: Precisión CA y SA. Porcentaje máximo, mínimo y promedio alcanzados usando diferentes valores de w, en la base de datos *Dance*.

un decrecimiento cuando $w = 12$ y $w = 16$. Cuando $w = 12$ al clasificador le toma alrededor de 700 milisegundos aproximadamente clasificar un gesto, *i.e.*, desde el momento en que el clasificador recibe los primeros cuadros hasta que arroja una decisión final. Sin embargo 700 milisegundos es demasiado tiempo si nuestro gesto tiene una duración menor; al aumentar el tamaño de la ventana este tiempo se reduce, es decir, cuando $w = 16$ el clasificador nos ofrece una respuesta en 600 milisegundos aproximadamente (ver Tabla 6.9), por lo que el clasificador podrá reconocer anticipadamente gestos con una duración mayor a 600 milisegundos.



(a)



(b)

Figura 6.11: Precisión CA y SA. Porcentaje máximo, mínimo y promedio alcanzados usando diferentes valores de w, en la base de datos *Dance2*.

Para *Dance2* se utilizó la siguiente configuración en el experimento: $L = 2$, $\gamma = 2.0$, $minPer = 50$, $maxPer = 62$. En la Figura 6.11 podemos ver la precisión promedio así como también el porcentaje promedio, máximo y mínimo obtenidos. Además agregamos duración en milisegundos promedio que se tarda nuestro método para clasificar un gesto (ver Tabla 6.9). Podemos ver que entre más grande sea el tamaño de la ventana, menos se tarda el clasificador en responder. Esto es porque el número de iteraciones y decisiones parciales se reduce, así como también el número de operaciones necesarias para ofrecer una respuesta. Sin embargo, si

dejamos crecer mucho la ventana, el método empieza a perder la posibilidad de detectar el gesto antes de que este sea terminado, como el número de cuadros que necesita dejar pasar es muy grande, es posible que para la siguiente iteración el gesto ya haya finalizado; entre más grande sea la ventana, menos refinada es la clasificación. Sin embargo, también podemos observar que el tamaño de la ventana no afecta en la precisión del método, únicamente en el porcentaje máximo en la que se reconocen los gestos. Los valores para $w \geq 70$ no son convenientes ya que el porcentaje máximo asciende hasta el 100 %. Cuando $w = 6$ se necesitan menos milisegundos para el reconocimiento que para $w = 5$, sin embargo, el porcentaje máximo crece mucho más para $w = 6$ que para $w = 5$ y en realidad los milisegundos necesarios para estos dos valores no son tan diferentes, por lo tanto el mejor valor para w es 5.

	2	4	5	6	7	8	10	12	14	16	18	20
Dance	2736.0	1670.0		1136.0		972.0	871.0	761.0	668.0	646.0	621.0	660.0
Dance2	31.3	22.3	19.3	18.9	17.3	16.2	15.9	15.1	12.8			

Tabla 6.9: Tiempo en ms. que le toma al clasificador reconocer un gesto. El tiempo fue tomado desde que inicia a hacer las decisiones parciales hasta que arroja una decisión final.