Método para la segmentación y reconocimiento simultáneo de ademanes

por

Harold Andres Vasquez Chavarria

Tesis sometida como requisito parcial para obtener el grado de

Maestro en Ciencias en el Área de Ciencias Computacionales en el

Instituto Nacional de Astrofísica, Óptica y Electrónica

Supervisada por:

Dr. Luis Enrique Sucar Succar, INAOE
Dr. Hugo Jair Escalante Balderas, INAOE

Resumen

Los ademanes son una forma natural de comunicación entre las personas, y por lo tanto también lo son para la interacción humano computadora (HCI). Aunque existen diversas técnicas para reconocer ademanes, asumen en general que se encuentran ya segmentados, por lo que no se ha resuelto aún el problema de segmentación.En este trabajo se propone un método para abordar ambas tareas a la vez, es decir, un método de segmentación y reconocimiento simultáneo de ademanes, usando modelos ocultos de Markov (HMM). Este método está basado en un esquema novedoso de exploración de la secuencia de vídeo denominado múltiples ventanas de tamaño dinámico. Esto consiste en varias ventanas superpuestas, que comienzan en puntos distintos de la secuencia y van aumentando su tamaño a medida que se va capturando al usuario con un Kinect. En cada instante de crecimiento de la ventanas, se obtienen predicciones de cada una de ellas, considerados como votos para cierto ademán. Cuando hay una mayoría absoluta entre las ventanas hacia cierto ademán, se espera el punto donde deja de ser unánime esta decisión y dicho punto se considera como el final del ademán. Una vez detectado ese punto final (segmentación), se arroja como resultado el ademán que obtuvo la votación mayoritaria hasta dicho momento (reconocimiento). El método propuesto se aplicó a comandar un robot de servicio, mediante la captura de información de los movimientos de el usuario con un sensor Kinect. Los resultados de los experimentos arrojaron un 82.76 % de ademanes bien segmentados, con una holgura alrededor del punto final de 15 cuadros. Para efectos de nuestra aplicación de comandar robots, esta holgura representa apenas medio segundo, que es suficientemente bajo e imperceptible en tiempo real. De igual modo, el reconocimiento es muy bueno cuando se logra segmentar el ademán, obteniéndose un 89.58 % de precisión. Una de las principales ventajas de esta propuesta, es que no es necesario el uso de un modelo HMM para no ademán, contrario a lo que se hace en otros trabajos. De igual modo, no es necesario un pose indicativo por parte del usuario para saber cuando empieza y termina un ademán.

Abstract

Gestures are a natural way of communication between people, hence also are for Human Computer Interaction (HCI). While there are many techniques to recognize gestures, these generally assume that the gestures are already segmented, so the problem of segmentation has not yet been solved. In this work we propose a method for addressing both tasks at once, ie. a method for simultaneous segmentation and recognition of gestures using Hidden Markov Models (HMM). This method is based on a novel video-stream exploration scheme called multi-size dynamic windows. This consists of multiple overlapping windows, which start at different points in the sequence and are increasing in size while capturing user information with a Kinect. At each growing moment of the windows, predictions are obtained from each one of them, that we considered as votes to certain gesture. When there is a majority decision between the windows to a certain gesture, it is expected a point where the unanimous decision finish and that point is considered the end of the gesture. Once detected that endpoint (segmentation), is given as result the gesture that won the majority vote until such moment (recognition). The proposed method was applied to command a service robot, by capturing information of the user movements with a Kinect sensor. The results of the experiments showed a 82.76 % of gestures well segmented, with a boundary relaxation around the final point of 15 frames. For purposes of our application to command robots, this gap represents only half from a second, which is sufficiently low and undetectable in real time. Similarly, the recognition is very good when the gesture is segmented, giving a accuracy of 89.58 % . One main advantages of this proposal is that it is not necessary use a HMM for not gesture, contrary to what is done in others works. Similarly, do not need a pose user to indicate when a gesture starts and ends.

Agradecimientos

Primero a Dios, que sin él nada es posible. Luego a mis asesores por el gran apoyo que me han brindado y de los cuales aprendí mucho. A mis compañeros de clase, que me han brindado esa conocida hospitalidad Mexicana y me extendieron la mano cuando la necesité. A México por abrirme sus puertas y mostrarme una tierra llena de oportunidades y bellezas. A mi país Venezuela por darme la oportunidad de hacer lo que más me gusta: aprender cada día algo nuevo.

Dedicatorias

Mi esposa e hijas que soportaron mi abandono por largos periodos, para brindarme el espacio necesario para culminar esta tesis; merecen esta dedicatoria indudablemente. Mi mamá, papá y hermanas que a lo lejos me apoyaron para que alcanzará esta meta, aquí los honro con este producto de mi esfuerzo.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

En este capítulo se presentan los orígenes del problema a abordar en este trabajo de investigación. Para tal fin, se detalla la motivación que llevo a ahondar sobre este tema, cuál es el problema que se desea resolver, los objetivos que se van a alcanzar, la solución que se propone para resolver el problema y los principales aportes que generaron esta investigación.

1.1. Motivación

En la actualidad existe una tendencia de usar ademanes (señales con las brazos y manos) para dar instrucciones a diferentes tipos de dispositivos; como por ejemplo, operación de videojuegos, manejo de interfaces de sistemas operativos y programas en general, operación de equipos remotos, instrucciones a robots, etc. Para los fines antes descritos, se han propuesto trabajos donde la persona debe llevar puesto un tipo de traje especial que detecta los movimientos del cuerpo o en el mejor de los casos, usar pequeños dispositivos en las manos (guantes, mandos a distancia inalámbricos, dedales, etc.) (Correa et al., 2010; Kim et al., 2007; Raheja et al., 2010; Yang et al., 2006). Sin embargo, estas propuestas son en muchos casos incomodas y poco naturales para una persona. De aquí que existe una tendencia mayor a hacer uso de técnicas de visión por computadora para lidiar con el problema antes descrito. Es por ello que actualmente es muy popular el uso del Kinect; aparato desarrollado y comercializado por Microsoft®para operar su consola de video juego Xbox 360. En particular, en el caso de detección de gestos ha sido una herramienta ampliamente usada por sus beneficios: información de profundidad, no hay problemas de ambientes ruidosos (fondos con muchos objetos que pueden provocar falsos positivos en la detección y seguimiento de mano o cara), lidia muy bien con problemas de luminosidad (sólo se afecta en ambientes exteriores muy soleados), existen muchas librerías para ser usadas en distintos sis-

temas operativos que ahorran tiempo de programación, se dispone de un esqueleto del cuerpo humano que contribuye significativamente en la clasificación de gestos, etc. (Goodrich et Schultz., 2007; Mitra et Acharya., 2007).

Por otra parte, siempre ha sido un desafío el lograr operar un robot usando medios de comunicación más naturales, como por ejemplo voz y/o ademanes (Goodrich et Schultz., 2007). En el caso particular de ademanes, se han utilizado cámaras convencionales y algunas más sofisticadas, para estudiar el movimiento del usuario y clasificarlo en algún tipo de instrucción que entienda y ejecute el robot. Sin embargo, en los últimos tiempos se han orientado estos trabajos al uso del Kinect con todos sus potenciales. Muchos trabajos de investigación aprovechan este avance para clasificar las imágenes o videos de acuerdo a sus intereses (Goodrich et Schultz., 2007; Waldherr et al., 2000).

Una ventaja principal de poder operar un robot con ademanes, usando en este caso un Kinect, es la naturalidad que brinda este medio de comunicación a una persona cualquiera. Podemos apreciar como el ser humano siempre usa movimientos de las manos, cabeza o incluso del cuerpo entero, para expresar ideas o necesidades. Incluso esto permite poder instruir al robot en ambientes muy ruidosos donde el sólo usar la voz no es del todo suficiente. Otra ventaja de esto es la posibilidad de que personas con cierta discapacidad puedan hacer uso de un robot asistencial para las mismas. Un ejemplo inmediato es una persona muda que pueda operar un robot que lo asista, dando instrucciones al mismo por medio de ademanes. Entre las partes del cuerpo que se pueden usar para mostrar un ademán, se puede mencionar la cabeza, la cara (en especial para mostrar sentimientos o estados de ánimo), los brazos, las manos, las piernas y combinación de algunos o todos los anteriores (todo el cuerpo). Uno de los más estudiados y más cómodos de usar por una persona son los brazos. Esto porque no se necesita tanto esfuerzo para expresar una orden al robot y los brazos permiten un sin número de posibles ademanes que se pueden formar. Es importante señalar que aunque aquí se menciona los brazos como los más estudiados, en realidad lo que se sigue es el movimiento de la mano en un intervalo de tiempo determinado, por lo que finalmente se habla de ademanes de mano. Otro aspecto a destacar es que, por razones de simplificación, la mayoría de los estudios de ademanes de mano, sólo se concentran en una de ellas, en particular el de la mano derecha (Goodrich et Schultz., 2007; Mitra et Acharya., 2007).

En este trabajo nos enfocamos en la detección e identificación de ademanes con las manos, haciendo uso del Kinect, para dar instrucciones a un robot. Este problema presenta los siguientes retos:

1. Identificar el inicio y fin de un ademán en una secuencia de video continuo, problema conocido como segmentación.

2. Extraer los datos de interés de las imágenes (cuadros de video) de acuerdo a la técnica de reconocimiento que se desea utilizar.

3. Considerar variedad de estatura, estilos y velocidad de movimientos de los usuarios.

4. Desarrollar un método de reconocimiento efectivo para realizar la tarea de identificar el movimiento de la persona, de acuerdo a un ademán previsto. Aquí se debe considerar precisión y velocidad de respuesta; este último debido a que se piensa en un sistema en tiempo real.

5. Disponer de secuencias de video de cada ademán para el entrenamiento del modelo de reconocimiento. Incluso, sería deseable poder usar un clasificador que pueda ser entrenado con un solo ejemplo de cada ademán.

6. Incorporar el método de reconocimiento al robot para que pueda identificar el ademán que está señalando el usuario y se comporte según lo programado.

7. Determinar una manera fácil de incorporar nuevos ademanes para instruir al robot.

Por otra parte, también existen otras variantes al problema que pueden ser considerados, como por ejemplo: el uso de las dos manos en vez de una, movimiento del robot o del usuario o de ambos al momento de hacer los ademanes, posición adversa (no frontal) del usuario ante el Kinect, entre otros. Entre todos los desafíos planteados anteriormente, este trabajo se enfoca en resolver el primer problema, esto es, determinar el punto de inicio y fin de un ademán, también conocido como segmentación del video para cada ademán. Dicho problema radica en la dificultad de establecer posturas al usuario para identificar los dos puntos (inicio y fin), dado que contradice la forma natural de interactuar entre personas; y por otra parte no es posible establecer tiempos fijos para cada ademán, dado que los mismos pueden variar de acuerdo a la complejidad de cada uno y al estilo de ejecución de un mismo usuario. En la mayoría de los trabajos relacionados con este tema, los ademanes usados en el entrenamiento y posterior prueba, son separados de la secuencia completa de video de forma manual (Arriaga et al., 2011; Kumar et al., 2010). Esto no es del todo útil al momento de querer dar varias instrucciones distintas seguidas al robot. Por lo tanto, se desea desarrollar un sistema capaz de identificar uno o más ademanes en una secuencia de video, usando el Kinect para dar instrucciones a un robot.

El problema que se propone investigar en el presente trabajo es la segmentación de videos por cada ademán, usando el Kinect, de un usuario realizando uno o más ademanes con una sola mano, de tal manera que se puedan dar órdenes a un robot.

1.2. Problema

En el campo de la robótica existe una rama dedicada al diseño y construcción de robots de servicio. Estos son robots cuyas funcionalidades principales están avocadas a asistir a personas ya sea en el hogar o cualquier otro ambiente donde necesiten ayuda personal (Aracil et al., 2008). Para este tipo de robots es necesaria una interacción natural con los seres humanos, ya que de esta forma se asegura una convivencia más placentera entre el robot y su dueño. De aquí que se estén realizando grandes esfuerzos para dotar a estos robots de mecanismos de interacción con el humano cada vez más comunes para ellos, tales como conversación por voz, detección de estados de ánimos o de peligro en la persona, seguimiento de órdenes dadas por ademanes, entre otros (Goodrich et Schultz., 2007). Todo esto indica que dotar de estas cualidades a un futuro robot casero completamente funcional, es un tema importante, por lo que muchos investigadores están avocados a esta tarea. Además, los ademanes también son útiles para cualquier modo de comunicación, como las interacciones humano computadora (HCI).

Por otra parte, el uso de ademanes para dar instrucciones a un robot, es un campo de investigación muy amplio (Goodrich et Schultz., 2007; Otero et al., 2006; Van den Bergh et al., 2011). Entre todas las posibles partes del cuerpo humano que se pueden usar para esto, una de las más estudiadas es la mano, es decir, hacer seguimiento del movimiento de la posición de la mano en un intervalo de tiempo determinado, para identificar el ademán y la consiguiente instrucción para el robot. Esto sin considerar qué forma está mostrando la mano. Una herramienta tecnológica ampliamente usada para lograr identificar ademanes es el Kinect, el cual brinda grandes beneficios (Goodrich et Schultz., 2007; Waldherr et al., 2000), tal como detección y seguimiento de la persona, dotación de información de los movimientos del cuerpo del usuario, integración con la mayoría de los sistemas computacionales, bajo costo, entre otros.

Además de lo anterior, se debe hacer uso de uno o más modelos de reconocimiento para identificar el o los ademanes. Entre los más usados por la literatura actual se encuentran los modelos ocultos de Markov (*Hidden Markov Models* - HMM), redes bayesianas temporales (*Temporal Bayessian Network* - BNT), campos aleatorios condicionales (*Conditional Random Field* - CRF), filtrado de partículas y condensación, máquinas de estados finito (*Finite State Machines* - FSM), máquinas de soporte vectorial (*Support Vector Machine* - SVM), redes neuronales con retropropagación (*BackPropagation Neural Network* - BPN), árboles de decisión, entre otros (Arriaga et al., 2011; Otero et al., 2006; Li., 2010).

Ahora bien, este campo de investigación aún presenta varios retos por solventar. Uno de ellos tiene que ver con el hecho de poder determinar de forma automática cuándo inicia y cuándo termina cada ademán. Esto porque un usuario puede necesitar mostrar al robot varios ademanes distintos, cada uno seguido del otro, para poder dar instrucciones continuas. Así mismo, el usuario no siempre está dando órdenes por medio de ademanes, por lo que el robot debe estar atento

a que en cualquier momento puede iniciarse esta acción por parte de la persona.

Ante esta problemática, varios trabajos (Raheja et al., 2010; Mitra et Acharya., 2007) proponen utilizar una postura fija del usuario que se usa como punto de inicio y fin para cada ademán; regularmente, la persona debe estar erguida de pie y con las manos abajo. Esta propuesta presenta el inconveniente de no brindar al usuario un comportamiento más natural para interactuar con el robot. Otra solución usada en pocos trabajos, es la fijación de un tiempo determinado para cada ademán, lo que da indicios de los intervalos de tiempo para segmentar el video (Mitra et Acharya., 2007). Sin embargo, no es del todo correcto asumir esto, dado que, dependiendo del ademán y del usuario, su duración en tiempo es variable. Algunos otros trabajos tratan de automatizar la segmentación, pero usando reconocedores para No-ademanes (Li et Greenspan., 2011; Kim et al., 2007). Sin embargo, determinar qué es un no ademán no es del todo trivial, dado que esto podría tratarse de infinitos movimientos posibles, lo cual no permite un adecuado entrenamiento para cualquier reconocedor. Una última propuesta, es la de segmentar el video manualmente, esto es, capturar secuencias de videos separadas para cada ademán; lo cual evidentemente no representa una solución aplicable al desenvolvimiento normal que debería tener un robot casero en su entorno de acción (Mitra et Acharya., 2007).

1.3. Objetivos

En esta sección se muestran los objetivos a alcanzar en este trabajo de investigación.

1.3.1. Objetivo General

Diseñar un método de segmentación y reconocimiento de ademanes simultáneo con las manos, que obtenga una precisión comparable a un reconocimiento con segmentación manual.

1.3.2. Objetivos específicos

1. Desarrollar un método para segmentación de ademanes.

2. Determinar la configuración más adecuada del modelo de segmentación de ademanes propuesto, que maximice la precisión de segmentación y reconocimiento.

3. Implementar un método para el reconocimiento de gestos segmentados.

4. Evaluar la precisión de segmentación y reconocimiento de ademanes en videos por parte del método desarrollado.

5. Implementar el método de reconocimiento propuesto en un robot de servicio.

1.4. Solución propuesta

De acuerdo a lo antes descrito, se plantea un esquema basado en múltiples ventanas dinámicas que se combinan con un esquema de votación, como se ilustra en la Figura 1.1

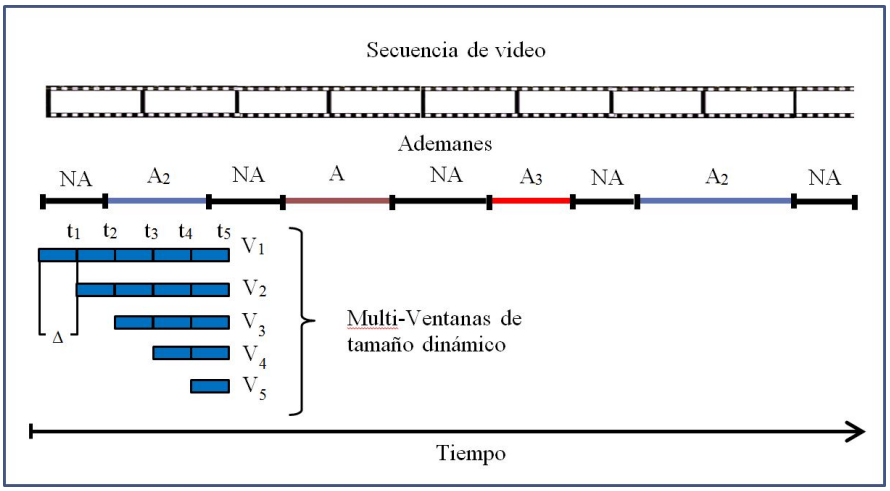


Figura 1.1: Esquema de Múltiples ventanas superpuestas de tamaño dinámico. En cada instante de tiempo t_i se tienen i ventanas y $i \times M$ probabilidades, que permitirán recabar mejor información para encontrar los límites del ademán.

Supongamos que se tiene una secuencia de video donde se tiene un usuario realizando varios ademanes A_j del vocabulario definido, con $1 \leq j \leq M$, donde $M = |A|$. Entre cada par de ademanes, el usuario no realiza ningún ademán conocido (NA). Por otra parte, se tiene intervalos de tiempo t_i. Cada t_i tiene una duración de tiempo igual a Δ. En el instante de tiempo t_1 se crea una ventana v_1 de tamaño Δ frames. Al final de esta ventana, se toman las características del segmento delimitado por v_1 y se envían a M modelos HMMs, con lo que se tiene M probabilidades. Cada probabilidad indica el nivel de predicción que se tiene para cada ademán. Seguidamente, en el instante de tiempo t_2 se incrementa el tamaño de v_1 en Δ frames y se crea una nueva ventana v_2 de tamaño inicial Δ frames. Con esto, para la ventana v_1 se tienen M nuevas probabilidades y para la ventana v_2 se tienen las primeras M probabilidades en el instante de tiempo t_2. De este modo, en cada t_i se va incrementando el tamaño de las ventanas existentes (tamaño dinámico) y se van creando nuevas ventanas (multi-ventanas). Con lo anterior, se cumple dos criterios:

1. En el instante de tiempo t_i se tienen i ventanas.

2. Para cada ventana, en el instante de tiempo i, se tienen i puntos con m probabilidades.

De todas las m probabilidades de cada ventana, una de ellas es mayor; con lo que se considera como predicción de la ventana (voto) el ademán que corresponda a esa probabilidad. De este modo, cada ventana aportará votos a ciertos ademanes y en cierto momento un ademán en particular obtendrá una mayoría de votos, suficiente para determinar que está ejecutándose ese ademán. En el instante en que esta mayoría cesa, se considera dicho punto como el final del ademán, con lo que se puede indicar a este como el ademán reconocido.

Toda la información recabada, permitirá determinar el cuadro más cercano al punto final del ademán y luego poder identificar cuál ademán acaba de ejecutarse. Esto se hace aplicando una estrategia de voto simple de las ventanas, es decir, según el ademán que cada ventana va identificando en cada t_i, se irá acumulando los votos que va obteniendo cada ademán en ese instante de tiempo. Si en algún punto de la secuencia algún ademán deja de tener votos mayoritarios (votos superiores a un δ), se dice que estamos en el punto final de un ademán, por lo que se segmenta y se reconoce.

Se reportan resultados de experimentos donde se pudo obtener una tasa de 82.76 % de segmentación y 89.58 % de reconocimiento. Estos valores son suficientemente aceptables, considerando que usando segmentación manual, se obtiene un 82.76 % de precisión de reconocimiento.

1.5. Contribuciones

Las principales contribuciones de esta tesis son las siguientes:

1. Un método de segmentación y reconocimiento automático simultáneo de ademanes, que no requiere el uso de un modelo de No-ademán, ni tampoco la obligación del usuario de adoptar un pose particular para indicar el fin e inicio de los ademanes.

2. La integración del método a un robot de servicio para comandarlo usando ademanes.

3. Una base de datos con información completa de varios ademanes, que puede usar cualquier equipo de desarrollo de robots de servicio.

1.6. Organización de la tesis

El resto del documento se compone de un marco teórico en el capítulo 2, donde se dan a conocer conceptos de ademán, modelos ocultos de Markov (HMM) y Kinect; una revisión del estado del arte en el capítulo 3; una explicación detallada del método de segmentación y reconocimiento automático simultáneo en el capítulo 4, los experimentos realizados y los resultados obtenidos en el capítulo 5 y finalmente, en el capítulo 6, se muestran las conclusiones y el trabajo futuro.

Capítulo 2

Marco teórico

En el presente capítulo se describen una serie de conceptos relacionados con lo desarrollado de esta investigación. Primero se introduce temas relacionados al reconocimiento de ademanes: concepto, reconocimiento y segmentación; luego se detalla el reconocedor utilizado para identificar dichos ademanes, que para este trabajo fueron los modelos ocultos de Markov(HMM) y se finaliza con una breve descripción del Kinect, que es el sistema de visión que aporta los datos que necesita los HMMs para llevar a cabo su trabajo de clasificación.

2.1. Segmentación y reconocimiento de ademanes

En la lengua española existen dos términos para denotar el término en inglés de "*gesture*", y según las costumbres dialécticas de los países latinoamericanos, cada término puede tener significados distintos. Según (Real Academia Española., 2001) gesto es "movimiento del rostro, de las manos o de otras partes del cuerpo con que se expresan diversos afectos del ánimo". Por otra parte, un ademán es "movimiento o actitud del cuerpo o de alguna parte suya, con que se manifiesta un afecto del ánimo"(Real Academia Española., 2001).

Aunque de lo anterior se puede concluir un uso indistinto de ambos términos, en este trabajo solo usaremos el término ademán, debido a que en muchos países el término gesto está más ligado a expresiones de la cara y podría generar confusiones. Además, este trabajo de investigación se enfocará en ademanes con las manos, es decir, movimientos de los brazos para manifestar una orden.

Los ademanes pueden ser usados en muchos campos de aplicación (Sebastien., 2002): robótica, realidad virtual aumentada, interfaces multi-modales, juegos, co-

reografías, control de computadoras, entre muchos más. Esto debido a la naturalidad que brindan a las personas para interactuar con dispositivos, tal como cuando dos personas se comunican y complementan o acompañan su interacción con movimientos de manos.

Adicionalmente, los ademanes tienen variaciones espacio-temporales (Li., 2010). Estos es, dos usuarios distintos realizan el mismo ademán en forma y duración distinta (variabilidad intra-sujeto). Aun peor, un mismo usuario puede realizar el mismo ademán varias veces, y cada una de estas repeticiones pueden variar en la forma o trayectoria que sigue y en la duración de realización (Variabilidad inter-sujeto). Esta es una de las varias complicaciones que presenta el encontrar los límites de un ademán y posteriormente identificar dicho ademán.

2.1.1. Segmentación de ademanes

En todo sistema de interacción humano computadora (*Human Computer Interaction*, HCI), el usuario puede realizar algunos movimientos para lograr dar las instrucciones necesarias a un computador y así obtener los resultados deseados. En el caso de este trabajo, se tiene un sistema de interacción humano robot (*Human Robot Interaction*, HRI), donde de forma análoga a lo antes descrito, un usuario puede instruir al robot por medio de ademanes. Sin embargo, bajo este escenario, el usuario realiza muchos movimientos de manos y no necesariamente todos corresponden a ademanes contemplados para dar órdenes al robot. Es decir, sea un conjunto de ademanes $A = \{a_i \setminus$ "a_i *es un movimiento de manos correspondiente a un ademan establecido en el diccionario*" $\}$ y un conjunto de instrucciones $I = \{i_i \setminus$ "i_i *es un comando establecido en el robot que indica al mismo a realizar ciertas acciones*" $\}$, entonces entre estos dos conjuntos se define un función como:

$$f(x) : A \to I, t.q. \forall y \in I : \exists! x \in A \setminus f(x) = y \tag{2.1}$$

Por lo tanto, es necesario delimitar los movimientos que correspondan al conjunto A para aplicar la función f y obtener respuesta del robot de servicio. Esto se conoce como segmentación de ademanes, es decir, encontrar los puntos de inicio y fin, en una secuencia continua de movimientos, que corresponden a la ejecución de un ademán. Esta tarea puede resultar muy difícil de realizar (Kahol et al., 2003), debido a la variabilidad temporal de los ademanes; es decir, una ademán puede tener duraciones de tiempos de ejecución muy variadas. Además, puede ser muy subjetivo determinar los límites de un ademán, dependientes de la secuencia de ademanes que se ejecute y es imposible enumerar todos los posibles ademanes (Kahol et al., 2003). En el capítulo 3 revisamos el estado del arte en segmentación de ademanes.

2.1.2. Reconocimiento de ademanes

Otro aspecto a considerar cuando se trabaja en HCI o HRI con ademanes, una vez hecha la segmentación, es el de asociar a cuál ademán corresponden los movimientos realizados por el usuario, para hacerlo corresponder con la instrucción a ejecutar por el que recibe la orden. Esto se conoce como reconocimiento o en algunos escritos también lo consideran clasificación. En este trabajo lo denominamos como reconocimiento de ademanes. Existen un número considerable de técnicas, unas mejores que otras, para lograr este cometido. Sin embargo, entre todas, los HMM's resultan ser uno de los más adecuados (Li., 2010), debido a su capacidad de enfrentar las variabilidades espacio-temporal que sufren los ademanes. En la siguiente sección se da una introducción a estos modelos.

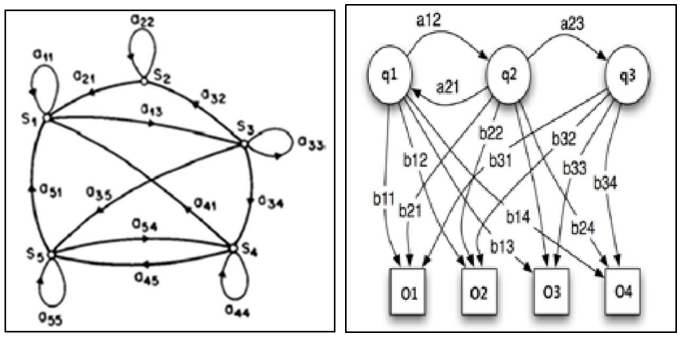
2.2. Modelos ocultos de Markov

Un Modelo Oculto de Markov (*Hidden Markov Model*, HMM)(Rabiner., 1989) puede ser visto como un tipo especial de red bayesiana dinámica (*Dynamic Bayesian Networks*, DBN) y a menudo se utiliza para codificar la estructura que queda implícita en la representación de la DBN (Koller et Friedman., 2009). El nombre se debe aparentemente a L. P. Neuwirth (Poritz., 1988).

Este modelo usa métodos estocásticos (aleatorio o probabilístico) y es un modelo probabilista paramétrico, por lo que se define con un conjunto finito de números reales. Posee dos componentes principales: una cadena de estados de Markov finita y un conjunto finito de distribuciones de probabilidad de salida. A diferencia de una cadena de Markov donde en cada estado corresponde a un evento observable, en los HMMs no ocurre esto. Por lo tanto, en los HMMs las observaciones son una función probabilista de los estados; es decir, el modelo es un doble proceso estocástico embebido, con un proceso estocástico subyacente que no es observable (oculto), pero que puede ser observado sólo a través de otro conjunto de procesos estocásticos que produce la secuencia de observaciones (Rabiner., 1989). En la Figura 2.1 se pueden apreciar ejemplos de estos modelos.

Por otra parte, los HMMs tienen un gran abanico de usos posibles (MacDonald et Zucchinni., 1997) tales como en reconocimiento de voz, genética y bioquímica, comportamientos de mercados, predicción de clima, biomedicina, etc. De aquí que en los últimos años haya habido un alto crecimiento en el número de publicaciones, que muestran los resultados de aplicar estos modelos a distintos problemas de la vida real. Uno de estos problemas es el de reconocimiento de ademanes tanto para HCI como para HRI; donde de igual modo, se puede encontrar una gran variedad de investigaciones usando este tipo de modelos para identificar ademanes.

A continuación se presentan los elementos necesarios para especificar un HMM, los problemas básicos que se plantean con los HMMs y que sirven para poder entender cómo se utilizan los mismos para distintos tipos de aplicaciones. También



(a) Cadena de Markov de 5 estados

(b) HMM de 3 estados y cuatro símbolos de observaciones

Figura 2.1: Ejemplos comparativos entre una cadena de Markov y un HMM. En la cadena de Markov no hay probabilidades de símbolos de observaciones de salida. Un HMM posee una cadena de Markov para los estados ocultos y otra para las observaciones.

se describen tipos de HMMs, consideraciones a ser tomadas en cuenta cuando se implementan estos modelos y una breve revisión de las herramientas y/o librerías disponibles en distintos lenguajes de programación para usar e implementar los HMMs.

2.2.1. Elementos de un HMM

Se pueden distinguir en un HMM un conjunto de estados $S = s_1, s_2, \cdots, s_N$, en un instante de tiempo t se tiene un estado denominado q_t y una observación O_t, de un conjunto de símbolos $V = v_1, v_2, \cdots, v_M$.

Un HMM de N estados puede ser descrito por una estructura $\lambda = (A, B, \Pi)$ (Rabiner., 1989), donde:

1. A_{NxN} es la matriz de transición de estados, donde cada $a_{ij} \in A$, es:

$$a_{ij} = P[q_{t+1} = s_j \mid q_t = s_i], 1 \leq i, j \leq N \tag{2.2}$$

   donde $P[q_{t+1} = s_j \mid q_t = s_i]$ es la probabilidad de que en el tiempo $t + 1$ el modelo se encuentre en el estado s_j dado que en el tiempo previo t se encontraba en el estado s_i.

2. B_{NxM} es la matriz de emisión de M símbolos de observaciones posibles, también conocido como distribución de probabilidad de los símbolos de ob-

servación, donde $b_{ij} \in B$, es:

$$b_{ij} = P[O_t = v_j \mid q_t = s_i], 1 \leq i \leq N, 1 \leq j \leq M \tag{2.3}$$

donde $O_t = v_j$ indica que la observación O asume el valor del símbolo v_j en el instante t.

3. Π_N es un vector de la distribución de probabilidades iniciales, es decir, indica la probabilidad de que un estado i sea inicial, por lo que cada $\pi_i \in \Pi$, es:

$$\pi_i = P[q_1 = s_i], 1 \leq i \leq N \tag{2.4}$$

Dada una secuencia de observaciones $O_1^T = O_1 O_2 \ldots O_T$, y un modelo $\lambda = (A, B, \Pi)$, se puede obtener la probabilidad de emisión de dicha secuencia de observación O_1^T dado el modelo λ, es decir, $P(O \mid \lambda)$.

2.2.2. Problemas básicos de un HMM

Para que un HMM pueda ser aplicable para resolver problemas de la vida real, se debe responder tres preguntas (Rabiner., 1989):

1. Dada la secuencia de observación O y el modelo λ, cómo obtener eficientemente $P(O \mid \lambda)$, lo que también se conoce como evaluación o reconocimiento.

2. Dada la secuencia de observación O y el modelo λ, obtener la secuencia de estados $Q = q_1 q_2 \cdots q_T$ más probables o que mejor "explique"las observaciones, también conocido como secuencia óptima.

3. Cómo ajustar los parámetros de λ para maximizar $P(O \mid \lambda)$, lo que se conoce también como entrenamiento.

Para efectos prácticos, este trabajo sólo necesita responder las preguntas 1 y 3; es decir, sólo se necesita entrenar el modelo y posteriormente usarlo para reconocer. Por lo tanto, sólo se muestra la respuesta a estas dos preguntas. La respuesta a la segunda pregunta se puede encontrar de modo claro en (Ferguson., 1980).

Reconocimiento.

Existe un método directo, que es sumando la probabilidad conjunta sobre todas las posibles secuencias de estados, esto es:

$$P(O) = \Sigma_Q P(O, Q_i) \tag{2.5}$$

Sin embargo, esto amerita $2T \times N^T$ operaciones donde T es el tamaño de la observación, lo cual es computacionalmente costoso. Por esta razón, existe un método iterativo, llamado algoritmo *Forward*, mostrado en el Algoritmo 2.1.

Algoritmo 2.1: Algoritmo de *Forward* para clasificación de una secuencia de observaciones dado un modelo λ

---

1 Inicialización:

$$\alpha_1(i) \leftarrow \pi_i b_i(O_1), 1 \leq i \leq N$$

2 Inducción:

$$\alpha_{t+1}(j) \leftarrow \left[\sum_{i=1}^{N} \alpha_t(j) a_{ij}\right] b_j(O_{t+1}), 1 \leq t \leq T-1, 1 \leq j \leq N$$

3 Terminación:

$$P(O \mid \lambda) \leftarrow \sum_{i=1}^{N} \alpha_t(i)$$

El algoritmo de *Forward* se basa en la idea de ir evaluando en paralelo la probabilidad de estados dadas las observaciones para cada tiempo. Para esto se define la variable *Forward*, como:

$$\alpha_t(i) = P(O_1 O_2 O_3 \cdots O_t, S_t = q_i) \tag{2.6}$$

es decir, la probabilidad de una secuencia parcial de observaciones y que llegue a cierto estado.

Con el Algoritmo 2.1 ahora se tienen $N^2 \times T$ operaciones, lo cual es mucho menor al método directo. Por esta razón, en este trabajo se usa este método para la clasificación de los ademanes una vez entrenados los modelos, pero con unas pequeñas variaciones que se detallan más adelante.

Entrenamiento.

El algoritmo más usado para entrenar un modelo HMM, es el de Baum-Welch o el equivalente *Expectation-Maximitation*-EM. Lo que se quiere es estimar (no es posible obtener valores exactos) son los parámetros del modelo λ que maximicen la probabilidad de una serie de observaciones de entrenamiento dadas. Este método se muestra en Algoritmo 2.2.

Para entender mejor dicho algoritmo, primero se define la variable *backward* como:

Algoritmo 2.2: Algoritmo de Baum-Welch para estimar los parámetros de un HMM

1 Probabilidades iniciales. Número de veces en el estado i en $t = 1$:

$$\pi_i \leftarrow \gamma_1(i)$$

2 Probabilidades de transición. Número de transiciones de i a j, entre el número de veces en i:

$$a_{ij} \leftarrow \frac{\sum_t \xi_{ij}(t)}{\sum_t \gamma_i(t)}$$

3 Probabilidades de salida. Número de veces en el estado j y observar k, entre el número de veces en ese estado:

$$b_{ij} \leftarrow \frac{\sum_{t,0=k} \gamma_j(t)}{\sum_t \gamma_j(t)}$$

$$\beta_t(i) = P(O_{t+1}O_{t+2}O_{t+3}\cdots O_T, S_t = q_i) \tag{2.7}$$

Dicha ecuación 2.7 se puede calcular iterativamente como:

$$\beta_t(i) = \sum_j \beta_{t+1}(j)a_{ij}b_j(O_{t+1}) \tag{2.8}$$

siendo $\beta_T(j) = 1$
Con (2.6) y (2.8), se puede definir la siguiente variable:

$$\gamma_t(i) = P(S_t = q_i \mid O) = \frac{\alpha_t(i)\beta_t(i)}{\sum_i \alpha_t(i)\beta_t(i)} \tag{2.9}$$

Finalmente, se define la variable ξ como:

$$\xi_t(ij) = P(S_t = q_i, S_{t+1} = q_j \mid O) = \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{\sum_i \sum_j \alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)} \tag{2.10}$$

De este modo, con las ecuaciones (2.9) y (2.10) se puede construir el Algoritmo 2.2. Se inicia con ciertos valores al azar y se va mejorando iterativamente, sin garantizar encontrar un óptimo global, por lo que es un estimador de máxima

verosimilitud.

2.2.3. Tipos de HMMs

Dependiendo de posibles restricciones que se impongan a los parámetros A, B y Π de un modelo λ, se pueden obtener distintas estructuras de un HMM.

Totalmente conexas

Un HMM totalmente conexo tiene la propiedad de que cada estado puede ser alcanzado desde cualquier otro en un número finito de pasos (Rabiner., 1989). En el ejemplo de la Figura 2.2, con $N = 4$ cada a_{ij} es positivo, es decir, cada estado es alcanzado directamente desde los otros.

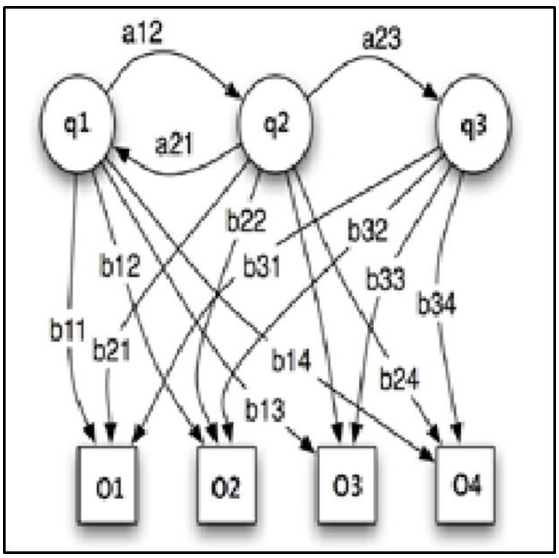


Figura 2.2: Modelo gráfico de un HMM tipo ergódico con transiciones entre todos los estados

Left-to-Right HMM

Son usados comúnmente en reconocimiento de voz y son llamadas también modelos Bakis (Levinson E. et al., 1983). Estas poseen las siguientes características:

1. La primera observación es producida mientras la cadena de Markov está en un estado especial llamado estado inicial, denotado con q_1.

2. La última observación es generado mientras la cadena de Markov está en un estado especial llamado estado final, denotado por q_N.

3. Una vez una cadena de Markov deja un estado, ese estado no puede ser visitado nuevamente más adelante.

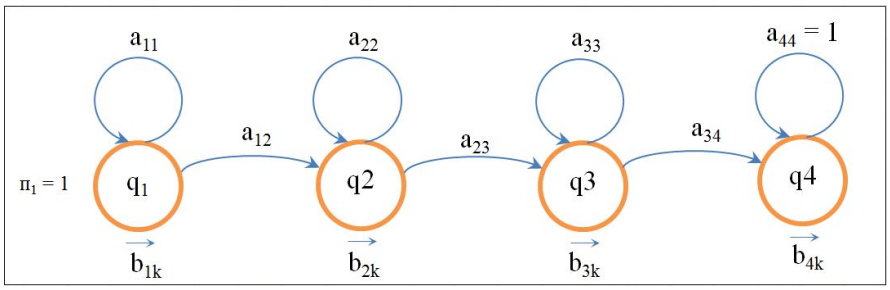


Figura 2.3: Modelo gráfico de un HMM tipo Left-to-Right con transiciones sólo entre estados continuos

Un modelo gráfico de este tipo especial de HMM puede ser apreciado en la Figura 2.3.

De lo anterior se puede deducir que para cumplir la primera condición, el conjunto $\Pi = (1, 0, 0, \cdots, 0)$ es fijo, es decir, no se reestima en todo el entrenamiento. La segunda condición puede satisfacerse, haciendo:

$$B_{iT} = \begin{cases} 1 & \text{para j=N} \\ 0 & \text{en otro caso} \end{cases} \tag{2.11}$$

Finalmente, la condición tres puede satisfacerse en el algoritmo Baum-Welch inicializando $a_{ij} = 0$ para todo $j < i$ u otra combinación de índices para transiciones no permitidas para este tipo de modelo. Es decir, se puede condicionar $a_{ij} = 0$ para todo $j > i + \Delta$, con lo que para el modelo de la Figura 2.3, $\Delta = 1$, con lo que se especifica que no puede hacer transiciones de más de un estado. Para $\Delta = 2$ (transiciones de no más de dos estados) se obtiene un modelo como el de la Figura 2.4.

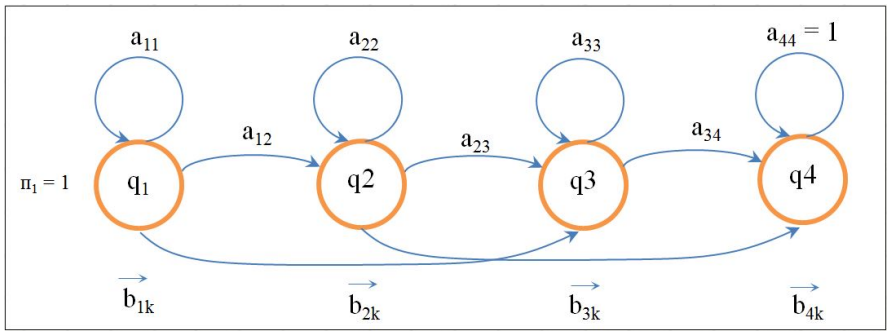


Figura 2.4: Modelo gráfico de un HMM tipo Left-to-Right con transiciones entre máximo dos estados

Para este trabajo de investigación se usa este tipo particular de modelo HMM, con $\Delta = 1$, dado que el reconocimiento de ademanes tiene un comportamiento muy parecido al reconocimiento de voz. Es decir, en ambos casos la señal tiene un comportamiento muy parecido, donde desde su inicio hasta su final la palabra o

ademan lleva una secuencia sin retorno.

2.2.4. Cuestiones de implementación de los HMMs

Aunque se puede pensar que los algoritmos de entrenamiento (Baum-Welch) y reconocimiento (forward) pueden ser implementados tal como se mostraron, en realidad debe considerarse algunos aspectos de implementación.

Underflow

El termino underflow se refiere al hecho cuando un número es demasiado pequeño para la capacidad de números que puede ser almacenado por el computador. En dicho caso, dicho número se marca de una manera especial para indicar que esta fuera del rango del conjunto de números que puede existir en la memoria.

Ahora bien, la estimación de parámetros del HMM con Baum-Welch y obtención de probabilidad de una observación dada con el algoritmo de *forward*, pueden ocasionar problemas de *underflow*. (Levinson E. et al., 1983) afirma que para cualquiera de los dos algoritmos mencionados anteriormente, para entrenamiento y reconocimiento, se requiere la evaluación de las variables $\alpha_t(i)$ y $\beta_i(i)$ para $1 \leq t \leq T$ y $1 \leq i \leq N$, con lo que cuando $T \to \infty$, entonces $\alpha_t(i) \to 0$ y $\beta_i(i) \to 0$ en forma exponencial. De lo anterior se desprende que, con el número adecuado de observaciones para entrenar un modelo y calcular sus probabilidades, originará *underflow* en cualquier computador tradicional. Por esta razón, se ha propuesto una modificación a estos algoritmos, con un método de escalamiento de los cálculos de dichas variables. Por ejemplo, para la variable α, se propone multiplicarla por un factor de escalamiento C_t, tal que:

$$C_t = \left[\sum_{i=1}^{N} \alpha_t(i) \right]^{-1} \tag{2.12}$$

de modo que $\sum_{i=1}^{N} C_t \alpha_t(i) = 1$ para $1 \leq t \leq T$. Con esto, el Algoritmo 2.1 se puede reescribir como se muestra en el Algoritmo 2.3.

Se puede apreciar en dicho algoritmo, en la fase de terminación, que el resultado no es sólo $P(O \mid \lambda)$, sino más bien $\log(P(O \mid \lambda))$. Por lo tanto, si se desea obtener la probabilidad original, se debe calcular:

$$P(O \mid \lambda) = e^{\log P(O|\lambda)/10} \tag{2.13}$$

donde, dado que $\log(P(O \mid \lambda))$ genera números negativos muy alejados del cero, al aplicar la exponencial, puede producir números muy pequeños o incluso cero. De ahí que primero se haga la división por 10.

Adicionalmente, para M observaciones distintas (M clases), se debe escalar el valor de las probabilidades de los M modelos HMMs, para que la suma de todos ellos sea 1. Esto simplemente se obtiene de la siguiente forma:

$$P(O \mid \lambda)_i = \frac{P(O \mid \lambda)_i}{\sum_{i=1}^{M} P(O \mid \lambda)_i} \tag{2.14}$$

Algoritmo 2.3: Algoritmo *Forward* escalado para evitar *underflow*

1 Inicialización:

$$\alpha_1(i) \leftarrow \pi_i b_i(O_1), 1 \leq i \leq N$$

$$C_1 \leftarrow \sum_{i=1}^{N} \alpha_1(i)$$

$$\alpha_1(i) \leftarrow \alpha_1(i)/C_1, 1 \leq i \leq N$$

2 Inducción:

$$\alpha_{t+1}(j) \leftarrow \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}), 1 \leq t \leq T-1, 1 \leq j \leq N$$

$$C_{t+1} \leftarrow \sum_{i=1}^{N} \alpha_{t+1}(i)$$

$$\alpha_{t+1}(j) \leftarrow \alpha_{t+1}(j)/C_{t+1}, 1 \leq j \leq N$$

3 Terminación:

$$\log P(O \mid \lambda) \leftarrow \sum_{i=1}^{T} \log(C_t(i))$$

Probabilidades cero

Para garantizar buenos resultados en el proceso de entrenamiento, es importante contar con un buen número de observaciones de entrenamiento. Si no es suficiente, se puede originar valores de probabilidad cero al momento de realizar las pruebas con nuevas observaciones. Esto ocurre especialmente en los modelos HMM tipo *Left-Right*, porque la naturaleza transitoria de los estados en el modelo, sólo permite un pequeño número de observaciones para cualquier estado (hasta que una transición es hecha a un estado sucesor) (Rabiner., 1989). Otra solución posible, es restringir algunos valores de los parámetros (solo A y B) del modelo a entrenar para que no sean cero, dependiendo del tipo de HMM que se desea. Por lo tanto, se puede maximizar $P(O \mid \lambda)$ sujeto a las restricciones $a_{ij} \geq \varepsilon > 0$ y $b_{ij} \geq \varepsilon > 0$ (Levinson E. et al., 1983). Finalmente, una técnica ampliamente

usada es el suavizamiento de las probabilidades de emisión, esto es, redistribuir las probabilidades existentes cuando se tiene alguna con valor cero, dando un valor pequeño a estas últimas. En (Boodidhi., 2011) se pueden encontrar varias técnicas para hacer esto.

Estimaciones iniciales de los parámetros

Aunque el algoritmo de Baum-Welch nos permite reestimar los parámetros del modelo que corresponda a un máximo local de la función de probabilidad, una pregunta común es cómo escoger las estimaciones iniciales de los parámetros del modelo, tal que el máximo local corresponda en realidad al máximo global de la función de probabilidad.

La experiencia ha mostrado que la solución más práctica para esto es simplemente una estimación inicial aleatoria (sujeto a restricciones de valores estocásticos y distintos de cero) o uniforme de los valores de A y Π (Rabiner., 1989).

Para el caso del parámetro B, se ha evidenciado que buenas estimaciones iniciales son provechosas para casos de símbolos discretos. Dichas estimaciones iniciales pueden ser obtenidas de distintas formas (Rabiner., 1989), como por ejemplo segmentación de la máxima verosimilitud de las observaciones con promedios.

2.2.5. Herramientas disponibles

Para hacer uso de los HMM's, existe una buena variedad de herramientas disponibles de forma gratuita:

1. *Kevin Murphy's Toolbox*: esta caja de herramientas está pensada para usarse en Matlab y en realidad ya prácticamente fue abandonado su soporte y actualización desde el 2005, dado que se mudó este proyecto al que se describe a continuación (Murphy, Kevin., 1998).

2. *Probabilistic Modeling Toolkit for Matlab/Octave (PMTK)*: Se encuentra en su tercera versión y en su creación y mantenimiento participaron Matt Dunham, Kevin Murphy y otras personas. No solo está diseñado para HMM's, sino también para otra gran variedad de modelos probabilísticos (Dunham, Matt and Murphy, Kevin., 2010).

3. *Hidden Markov Model Toolkit (Htk3)*: aunque fue pensado originalmente para reconocimiento de voz, puede usarse para la amplia gama de aplicaciones de los HMM's. Consiste en un conjunto de librerías y herramientas en lenguaje C (Cambridge University Engineering Department (CUED)., 1993).

4. *General Hidden Markov Model Library(GHMM)*: esta implementado en lenguaje C e incluye un conjunto de *"Wrappers"* para crear interfaces con *Python*. Es distribuido bajo la licencia LGPL. Fue desarrollado por el grupo de Bioinformática de la Universidad de Rutgers (Schliep, Alexander., 2010).

Posee una herramienta gráfica para crear y editar HMM's, llamada *Hidden Markov Model editor* (HMMEd). En la Figura 2.5 se muestra una captura de pantalla de su interfaz.

5. *Java Hidden Markov Model (Jahmm)*: como se lee, esta es una implementación en Java de los HMM's, esta licenciado bajo la GPL, sin embargo actualmente está mudando su sitio web y van a adoptar el licenciamiento *Berkeley Software Distribution* (BSD). Provee una interfaz gráfica y de línea de comandos (Francois, Jean-Marc., 2005).

6. *Tapas Kanungo's Hidden Markov Model Toolkit*: es uno de lo más usados. Está escrito en lenguaje C y entre los varios programas que pone a disposición en el campo de la inteligencia computacional, se encuentra uno para los HMM's (Kanungo, Tapas., 2013).

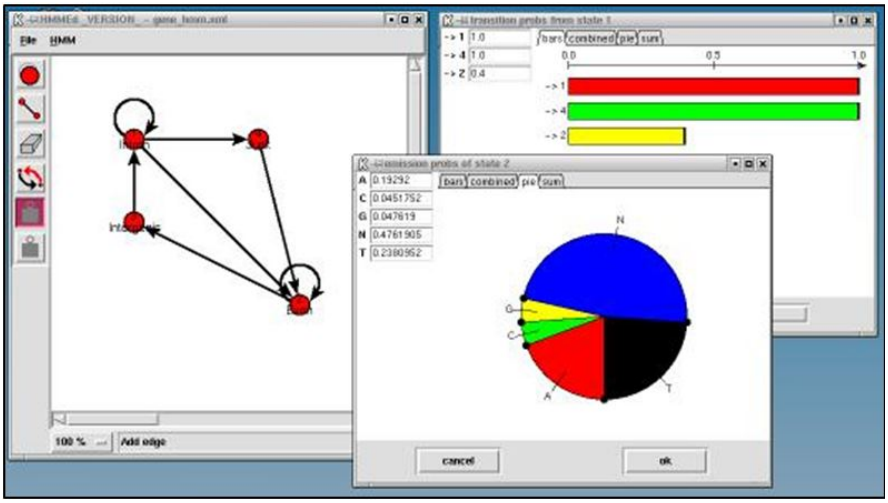


Figura 2.5: Captura de pantalla del editor de HMM de *General Hidden Markov Model Library* (GHMM) (Schliep, Alexander., 2010)

Para finalizar esta sección, debemos considerar ahora que estos reconocedores necesitan datos de entrada (observaciones) para realizar su trabajo. En nuestro caso, dichos datos esta relacionados con los movimientos de las manos y por ende se necesita un sistema de visión que aporte dicha información. El sistema usado en este trabajo es el Kinect, dada sus ventajas descritas en la siguiente sección y ademas porque el robot Sabina ya posee dichos equipos instalados en su hardware.

2.3. Kinect

Aunque inicialmente este dispositivo fue pensado en su lanzamiento (4 de noviembre de 2010) para ser usado exclusivamente en la consola de video juego

Xbox de Microsoft, rápidamente se migró a otros usos. Uno de estos es lo referente a HCI, dado las ventajas que ofrece como su bajo costo, gran variedad de librerías disponibles para su uso, buena segmentación del usuario en ambientes ruidosos, etc.

Una de las bondades que ofrece el Kinect es el referente al esqueleto, como se muestra en la Figura 2.6, donde se puede disponer de las coordenadas (x, y, z) de 20 uniones (*joints*) del cuerpo humano. Parte de esta información es utilizada para extraer las características que necesitan los HMMs para poder hacer su tarea de clasificación. En el apéndice **??** se aporta una descripción más detallada de este sensor.

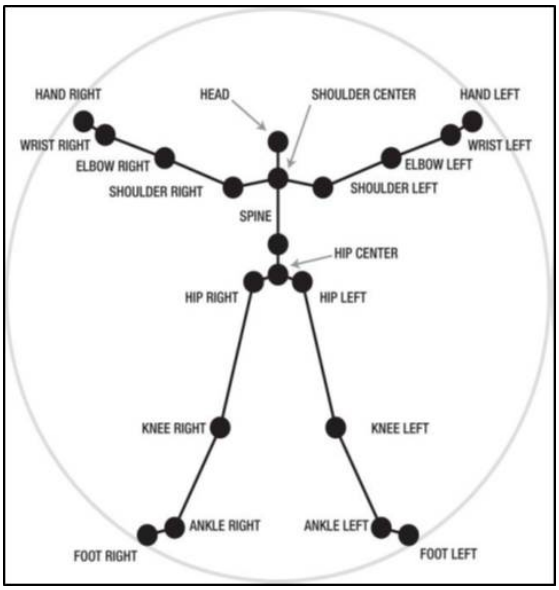


Figura 2.6: Esqueleto que se puede obtener del Kinect donde se dispone de las coordenadas (x, y, z) de 20 uniones del cuerpo humano (Jarrett et James., 2012)

2.4. Resumen

En este capítulo se mencionaron los conceptos base de ademanes, se explicó las teorías elementales de los Modelos Ocultos de Markov para entender su construcción y uso, y se cerró con una breve descripción del Kinect.

En el capítulo que sigue, se presenta una revisión del estado del arte sobre los temas relacionados con este trabajo de investigación.

Capítulo 3

Trabajo relacionado

Dado que el reconocimiento y segmentación de ademanes ha tenido un fuerte auge en la última década, existe un número importante de publicaciones que muestran algunos resultados relacionados con este trabajo de investigación. A continuación se presentan los resultados de la revisión de algunos de estos trabajos, que de una u otra manera, contribuyeron a esta investigación.

Primero se muestra los principales métodos usados para identificar los ademanes, y luego se estudian las propuestas que existen sobre las características más adecuadas a extraer de la secuencia de video de acuerdo al método de reconocimiento usado. Sigue un resumen del conjunto de ademanes usados en algunos trabajos para comandar robots de servicio y se finaliza el capítulo con la revisión de las propuestas que existen para segmentar y reconocer ademanes de forma simultanea.

3.1. Método de Reconocimiento

Aunque la propuesta de este trabajo de investigación es la segmentación automática, esto es casi imposible de lograr sin un reconocedor bien entrenado; y para tener un reconocedor de esta forma, es imperativo entrenarlo con ademanes ya segmentados, es decir, con ademanes segmentados manualmente (Song et al., 2012; Kahol et al., 2003). De esto se desprende la necesidad de identificar dos elementos importantes para llevar a buen término este trabajo: un reconocedor y el conjunto de características para dicho reconocedor.

En relación a los reconocedores a usar, hay un gran número de trabajos que usan los HMM's o sus variantes para reconocer los gestos (Nguyen-Duc-Thanh et al., 2012), también debemos considerar otro grupo de reconocedores usados.

En (Rehr et al., 2010; Arriaga et al., 2011) se propone el uso de redes bayesianas dinámicas (Dynamic Bayesian Networks-DBN) en contraposición con los HMM's

argumentando mayor precisión en la clasificación. Por otra parte, en (Doliotis et al., 2011) se propone el uso de DTW's para hacer corresponder un nuevo ademán con los ya registrados para reconocer.

En este trabajo se utilizó los HMM's como reconocedores, debido a la amplia bibliográfica que muestra su efectividad y resultados aceptables cuando se usa con los ademanes. Sin embargo, el método propuesto se podría extender para ser utilizado con otro tipo de reconocedor, lo cual esta fuera del alcance de este trabajo y se recomienda como trabajo futuro.

3.2. Características

Existe en la literatura un gran número de propuestas distintas para considerar las mejores características a utilizar para el reconocimiento de ademanes. La selección de éstas puede alterar considerablemente la precisión del reconocimiento. La mayoría de los trabajos usan la mano como parte del cuerpo que mejor representa los ademanes (Arriaga et al., 2011; Brethes et al., 2004; Correa et al., 2010; Goodrich et Schultz., 2007; Kumar et al., 2010; Mitra et Acharya., 2007).

En (Arriaga et al., 2011) se distinguen dos tipos de atributos: de movimiento y de postura. Se propone usar tres atributos para el movimiento de la mano y cuatro para la postura. Los atributos de movimiento son cambio de área en la mano ($\Delta area$), cambio en el eje x (Δx) o cambio en el eje y (Δy). Cada atributo puede tomar tres valores posibles $+, -, 0$ que indica incremento, decremento y sin cambio; en referencia al área o posición de la mano en el cuadro del video anterior.

Los atributos de postura son llamados forma, derecha, arriba y torso. Para forma, se tienen tres valores posibles: "+"si la mano está en posición vertical, "-"si está en horizontal y "0" en otro caso. Los siguientes tres atributos son valores binarios en caso de que la mano esté a la "derecha"de la cabeza,"arriba"de la cabeza o si está al frente del "torso".

Algo similar a lo descrito anteriormente se usa en (Rehr et al., 2010). Primero se obtiene el contorno de la imagen de la mano $I_m(x, y, t)$ para el instante t, y luego se calcula el centro de gravedad $\Lambda_t = (cx_t, cy_t)^T$ utilizando para ello el momento de la imagen dado por:

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q I_m(x, y, t) \tag{3.1}$$

con lo que el centro de gravedad es dado por:

$$cx_t = \frac{\mu_{10}}{\mu_{00}} \qquad cy_t = \frac{\mu_{01}}{\mu_{00}} \tag{3.2}$$

De este modo, con la Ecuación 3.2, se extraen las características relativas a posición $\Delta\Lambda_t$, de la siguiente forma:

$$\Delta\Lambda_t = \Lambda_t - \Lambda_{t-1} \forall t \subseteq 1, \cdots, n, \Lambda_{t=0} = 0 \tag{3.3}$$

Para las características relativas a la forma, que sea invariante a escala, traslación y rotación; se usa el momento de *Hu* (Hu., 1962) con lo que se obtiene el vector $\xi_t = (hu1_t, hu2_t, hu3_t, hu4_t, hu5_t, hu6_t, hu7_t)^T$. Finalmente, se combinan los vectores Λ_t y ξ_t en un sólo vector de características de las manos ζ_t.

En (Yan et al., 2012) se obtiene primero los ángulos en distintos tiempos de los hombros y codos del usuario. Por lo tanto, se tiene un vector de la forma:

$$q_{t\,t=1}^{N} = q_{t,l1}, q_{t,l2}, q_{t,l3}, q_{t,l4}, q_{t,r1}, q_{t,r2}, q_{t,r3}, q_{t,r4}{}_{t=1}^{N} \tag{3.4}$$

donde $(q_{t,l1}, q_{t,l2}, q_{t,l3})$ son los ángulos de inclinación (*pitch*), derrape (*yaw*) y redoble (*roll*) del hombro izquierdo, $q_{t,l4}$ es el ángulo de derrape para el codo izquierdo. De manera similar se aplica para $q_{t,r1}, q_{t,r2}, q_{t,r3}$ y $q_{t,r4}$.

De la ecuación (3.4) se crea una matriz $q_{t,j}{}_{t=1}^{N}$ donde $j = l1, l2, l3, l4, r1, r2, r3r4$, representa valores de ángulos para una unión en diferentes tiempos. Luego con una transformada de Fourier rápida los coeficientes de las primeras M frecuencias bajas de $q_{t,j}{}_{t=1}^{N}$ son obtenidas como:

$$c_{k,j}{}_{k=1}^{2M} = [c_{1,j}, c_{2,j}, \cdots, c_{2M,j}] \tag{3.5}$$

Considerando todos los valores de las uniones, el vector de características en el dominio de frecuencia se convierte en:

$$C_f = [c_{k,l1}{}_{k=1}^{2M}, c_{k,l2}{}_{k=1}^{2M}, \cdots, c_{k,r4}{}_{k=1}^{2M}]^T \tag{3.6}$$

De todos los trabajos antes mencionados, el propuesto por (Arriaga et al., 2011) es el más simple en contraposición con los otros. Además, las otras propuestas están muy ligadas al sistema de visión usado, contrario a las características propuestas por (Arriaga et al., 2011). Finalmente, estas características pueden ser muy fáciles de extraer del modelo del esqueleto dado por el Kinect.

Por las razones antes expuestas, en este trabajo se toman algunas de las ideas propuestas por (Arriaga et al., 2011). Se considera dos grupos de características de acuerdo al tipo de experimento que se este realizando. En el primer grupo sólo se extrae variaciones espaciales de las manos (trayectoria), tomando en cuenta sus coordenadas en 3D que son otorgadas por el Kinect. En el segundo grupo se toma en cuenta la trayectoria junto con la posición de la mano respecto a un punto de referencia, que en este caso dicho punto es el centro de los hombros, que podríamos simplificarlo como cuello.

3.3. Vocabulario de ademanes

Otro aspecto importante antes de poder alcanzar nuestro objetivo, es el determinar cuáles ademanes son los más adecuados para poder comandar un robot de servicio. En la Tabla 3.1 se muestra un resumen de los ademanes usados para comandar robots de servicio en algunos trabajos.

Tabla 3.1: Conjunto de ademanes considerados por algunos trabajos para comandar robots

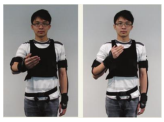
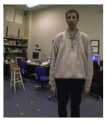
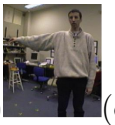
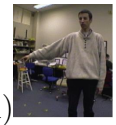
Trabajo	Ademanes	Imagen
(Yan et al., 2012)	Derecha	
	Adelante	
	Atrás	
	Alto	
	Velocidad. Movimientos rápidos (lentos), mayor (menor) velocidad.	
(Waldherr et al., 2000)	Alto: Imagen 'b'	(a)  (b) 
	Seguir: Movimientos de la manos de arriba hacia abajo. Imágenes 'd' y 'e'	(c)  (d) 

Tabla 3.1 – Continuación de la tabla anterior

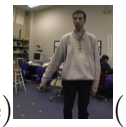
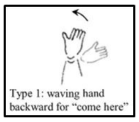
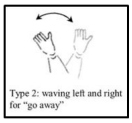
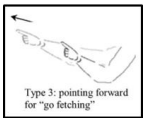
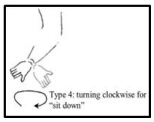
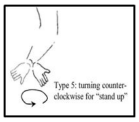
Trabajo	Ademanes	Imagen
	Señalamiento vertical: desde el pose 'a', mueve la mano a la posicion 'f', lo mantiene un momento y regresa al pose 'a'	(e)  (f) 
	Señalamiento bajo: desde el pose 'a', la persona apunta un objeto en el piso como en 'e', y reotrna la pose 'a'	
(Zhu et Sheng., 2011)	Ven aquí	
	Vete	
	Buscar	
	Sentarse	
	Levantarse	
(Arriaga et al., 2011)	Ven	
	Atención	

Tabla 3.1 – Continuación de la tabla anterior

Trabajo	Ademanes	Imagen
	Detener	
	Derecha	
	Izquierda	
	Mirar a la izquierda	
	Mirar a la derecha	
	Waving-hand	
	Apuntar	
	Posición inicial y final	

Se puede apreciar que algunos ademanes pertenecen a más de un grupo, por lo que podrían considerarse para este trabajo. Por lo tanto, considerando los comandos que se desea ejecute el robot en un principio básico, se definieron un conjunto de 5 ademanes que son detallados más adelante.

3.4. Segmentación automática

Una vez aclarado qué reconocedor y conjunto de características son los más adecuados para nuestra investigación, se debe disponer del conocimiento sobre los esfuerzos que existen para alcanzar el mismo cometido aquí planteado; es decir, lograr una segmentación y reconocimiento simultáneo de ademanes.

(Song et al., 2012) proponen un esquema donde se considera los movimientos de la mano y cuerpo para segmentación e interpretación de ademanes contínuamente. Para lograr esto, primero estiman un modelo 3D de la parte superior del cuerpo, usando las imágenes capturadas de un sistema de cámaras estéreo y mejorando su seguimiento con filtrado de partículas (*Particle Filter*). Esto último, obliga al usuario a una pose de inicio en forma de la letra T, es decir, parado y con las manos extendidas a los lados (pose "T") para inicializar el seguimiento, lo cúal puede no ser cómodo para el mismo. Luego detecta las manos y su forma usando histogramas de gradientes orientados (*Histogram of Oriented Gradients*, HOG) y un clasificador multiclase SVM (*Suport Vector Machine*). Con las características de cuerpo y manos extraídas, usan un *Latent-Dynamic Conditional Random Field* (LDCRF) y filtrado multicapa para segmentar y etiquetar los ademanes continuamente. Para los experimentos, usaron la base de datos NATOPS (*The Naval Air Training and Operating Procedures Standardization*), constituido por 24 ademanes. Se usó *K-fold cross validation*, donde el conjunto de datos de prueba y validación contenían 2400 ejemplos y el conjunto de entrenamiento 4800, cada conjunto con ejemplos realizados por usuarios distintos. De este modo, se obtuvo un 75.37 % de precisión para el conjunto de prueba y 86.35 % para el conjunto de validación. Por lo tanto, la mayor desventaja de esta propuesta, es la obligación que tiene el usuario de tomar cierto pose para poder inicializar el sistema y posteriormente para indicar el final y comienzo de cada ademán.

(Kahol et al., 2003) proponen un algoritmo para segmentación conformado por tres pasos: derivación de la segmentación espacial, derivación de la actividad del segmento y detección de los límites del ademán. Los dos primeros pasos sólo sirven para determinar las características a usar en el tercer paso, donde se realiza la segmentación usando un clasificador bayesiano simple. Las características están relacionadas con un modelo jerárquico del cuerpo humano y sus movimientos, por lo que se calcula el momento, la energía kinetica y la fuerza. Para esto, se usaron la masa, la velocidad y aceleración de cada segmento del cuerpo en estudio. Para las pruebas, se dispuso de 25 secuencias cortas (de 2 a 3 minutos) de bailes y actividades cotidianas. Cinco observadores realizaron una segmentación manual de 15 de estas secuencias como valor de ground truth y se aplicó la estrategia propuesta a 20 secuencias. Con esto, se obtuvo en promedio un 87.9 % de precisión en la segmentación. Este trabajo no muestra una estrategia de reconocimiento.

(Li et Greenspan., 2011) proponen tres métodos de segmentación y detección simultánea. Todos los métodos usan un mismo modelo llamado "Modelo de gesto". La mayor diferencia entre los tres, es la manera como localizan el punto final

del ademán. El primer método combina una técnica de detección de movimiento y una búsqueda multi-escala explícita para encontrar el comienzo y fin de un ademán. La técnica de detección de movimiento usa un modelo de No-ademán para encontrar el punto de inicio, donde cada cuadro es emparejado a este modelo y la puntuación obtenida es comparada a un umbral predefinido. Cuando cae por debajo de este umbral, se indica que está comenzando un ademán. La búsqueda multi-escala explícita es empleada ahora para encontrar el mejor emparejamiento entre los segmentos de movimientos. Si un segmento de movimiento es más corto que la máxima escala s_{max} de un modelo de gesto dado, una medida de similitud es evaluada entre estos movimientos en la correspondiente escala; en otro caso, el movimiento es comprimido a la máxima escala y la comparación es realizada a la máxima escala. En el caso de los otros dos métodos, estos mejoran la eficiencia y precisión de este primero, empleando Programación Dinámica (*Dynamic Programming*, DP) y una extensión de *Dynamic Time Warping* (DTW). El modelo de gesto propuesto se obtiene calculando la distancia normalizada de todo el contorno del sujeto, durante varios cuadros de tiempo; con lo cual se construye una imagen en escala de grises considerada como una firma del gesto efectuado. Sin embargo, a pesar de algunos intentos infructuosos, este modelo no es muy robusto a la variabilidad temporal de los ademanes. Además, se propone un modelo de no-ademán, con el sujeto sin hacer ningún movimiento. Usan las tasas de medición del reconocimiento de voz: sustitución (sub), borrado (del) e inserción (ins). La combinación de estas medidas en una sola se puede apreciar en la Tabla 3.2. Con esta medida, reportan una tasa de reconocimiento de 88.9 %, 91.4 % y 96.4 % para los métodos I, II y III respectivamente. Un aspecto importante que se menciona en este trabajo, es sobre la dificultad para compararse con trabajos similares, dado que estos no están disponibles para ser probados. Por otra parte, la propuesta de un modelo de no-ademán con la persona sin moverse, no necesariamente representa este tipo de casos, es decir, un No-ademán puede ser cualquier movimiento distinto a los contenidos en los ademanes. Otro aspecto no muy adecuado es que en las capturas de ademanes continuos, el sujeto fue impedido de realizar pausas entre dichos ademanes, lo cual no es muy natural.

En (Kim, et al., 2007) se plantea la necesidad de encontrar el punto de inicio y fin de un ademán en la secuencia hacia adelante. Esto porque las técnicas que buscan hacia atrás el punto de inicio luego de encontrado el punto final, provocan un retardo para sistemas de tiempo real. Para lograr lo que plantean, calculan lo que llaman diferencia de probabilidad de observación competitiva (*Competitive Differential Observation Probability*, CDOP), que usan las probabilidades de ademanes y No-ademán. Los puntos donde cambia de signo el CDOP se toman como puntos de inicio y fin de un ademán. Luego determinan el ademán entre estos puntos, usando un HMM acumulativo, junto con una ventana deslizante. Esto lo aplican para controlar luces y cortinas de una casa. Con lo anterior aseguran obtener un 95.42 % de reconocimiento. El gran inconveniente de esta propuesta es la construcción del modelo HMM para no-ademán, el cual no es trivial de realizar,

por la infinidad de movimientos posibles para lograrlo.

3.5. Resumen

Para tener un panorama general del estado del arte, se presenta en la Tabla 3.2 una comparación entre los principales trabajos que realizan segmentación y reconocimiento de ademanes, destacando algunas de sus principales características.

Tabla 3.2: Cuadro comparativo del estado del arte en segmentación y reconocimiento simultáneo de ademanes. Las siglas N.E. significan No Especifican.

Característica	**(Song et al., 2012)**	**(Kahol et al., 2003)**	**(Li et Greenspan., 2011)**	**(Kim et al., 2007)**
Tasa Segmentación	N.E.	87.9 %	N.E.	95.71 %
Tasa Reconocimiento	75.37 %	N.E.	96.4 %	95.42 %
Medida usada	*F1 Score*	Precisión	$Rec.Rate = 100\,\% * \left(1 - \frac{Sub+Del+Ins}{N.Ademanes}\right)$	Para segmentación: Num. de puntos de (inicio y fin) bien segmentados entre el total, usando una holgura de 2 unidades de tiempo. Para reconocimiento: Num. de ademanes correctamente detectados entre el total.

Tabla 3.2 – Continuación de la tabla anterior

Característica	**(Song et al., 2012)**	**(Kahol et al., 2003)**	**(Li et Greenspan., 2011)**	**(Kim et al., 2007)**
Clasificador	*Latent-Dynamyc Conditional Random Field* (LDCRF)	Clasificador Bayesiano Simple	DP y DTW	HMM Acumulativo
Sistema de Visión	Cámara estéreo *Bumblebee2*	Ninguno. Se usaron marcas de cuerpo	*Point Grey FireFly Camera y Point Grey DragonFly camera*	Tres cámaras CNB-AN202LCCD
Datos	*NATOPS dataset*	25 secuencias cortas de bailes y acciones cotidianas	Ocho ademanes efectuados por cinco sujetos. Para entrenamiento, tres sujetos efectuaron 30 repeticiones con tiempos similares	Ocho ademanes de 480 secuencias (60 para cada ademán)

Tabla 3.2 – Continuación de la tabla anterior

Característica	**(Song et al., 2012)**	**(Kahol et al., 2003)**	**(Li et Greenspan., 2011)**	**(Kim et al., 2007)**
Segmentación	Ventana deslizante temporal con filtro multicapas	Red Bayesiana Simple para encontrar mínimos locales de los movimientos de acuerdo a límites de ademanes dados por observadores humanos.	Usan un modelo de No-ademán para encontrar los límites.	Usan un modelo de No-ademán par encontrar los límites

De la Tabla 3.2 se puede concluir que:

1. Solo uno de los trabajos reporta tasas de segmentación y reconocimiento. Los restantes no especifican (N.E.) alguno de ellos.

2. No existe una medida de segmentación y reconocimiento de uso común para todos los trabajos.

3. Ninguno de los trabajos usan el Kinect como sistema de visión.

4. Hay una gran diversidad en los datos usados en los experimentos.

Finalmente, en términos generales las propuestas aquí mostradas usan un modelo de no-ademán o usan posturas indicativas hechas por los usuarios para segmentar y posteriormente reconocer. El requerimiento de cierta postura al inicio y fin de un ademán no es adecuada para una interacción natural con un robot. Además, contar con un conjunto finito de posibles movimientos para un modelo de no-ademán es muy difícil. De aquí que surja la propuesta presentada en este

trabajo, para lograr esta segmentación y reconocimiento simultáneo, sin requerir posturas específicas de inicio y fin de ademán, ni modelos de no-ademán.

De este modo, en el siguiente capítulo se mostrará a detalle el método propuesto para lograr este fin.

Capítulo 4

Segmentación y reconocimiento simultáneo de ademanes

En este capítulo se describe el método propuesto para lograr identificar los puntos de inicio y fin de un ademán (segmentación), y al mismo tiempo identificar cuál ademán se ejecutó (reconocimiento). Primero se introduce una descripción general del método y luego se explica la técnica de exploración de la secuencia de video, denominada múltiples ventanas superpuestas de tamaño dinámico. Posteriormente se da una explicación detallada del método propuesto, las características extraídas para el reconocedor, la manera de obtener las probabilidades de una secuencia de observación, los esquemas de votación usados, la determinación de los límites del ademán y algunas consideraciones finales del método.

4.1. Introducción

Primero, se asume que se puede extraer de la secuencia dada por el Kinect o cualquier otro sensor, la información necesaria para ser usada en nuestro método (características). El primer reto que se afronta es cómo ir extrayendo dichas características de la secuencia para ir entregándoselas a un reconocedor. Para esto se piensa en pedazos de dicha secuencia que inicialmente llamaremos segmentos. Con el segmento de características y el reconocedor, se pueden crear votos de los mismos hacia un ademán en particular. Luego, estos votos pueden dar lugar a un ademán vencedor por un tiempo determinado, que en algún momento deja de serlo. En dicho instante, se infiere que el ademán terminó su ejecución, con lo que se puede segmentar su punto final, suficiente para que el método reconozca a dicho ademán.

La Figura 4.1 muestra el diagrama general del método propuesto. En dicho

diagrama se puede apreciar que se debe disponer de una técnica de exploración de la secuencia de video que genere segmentos del mismo para poder entregárselos a un módulo generador de votos. Este último determina las decisiones tomadas por cada segmento sobre el cuál es supuestamente el ademán que se encuentra contenido en él. Dentro de dicho módulo hay una parte encargada de extraer las características del segmento, que son entregadas a un reconocedor, quién a su vez identifica el supuesto ademán. Con los votos obtenidos, se va preguntando en qué instante ocurre una decisión mayoritaria de los segmentos existentes y posteriormente se verifica cuando dichas decisiones dejan de ser mayoría. En este último instante, se presume que los segmentos están indicando que un ademán acaba de terminar su ejecución, por lo que podemos identificar el punto final del mismo. De este modo, se procede a segmentar y reconocer el ademán que se ejecutó.

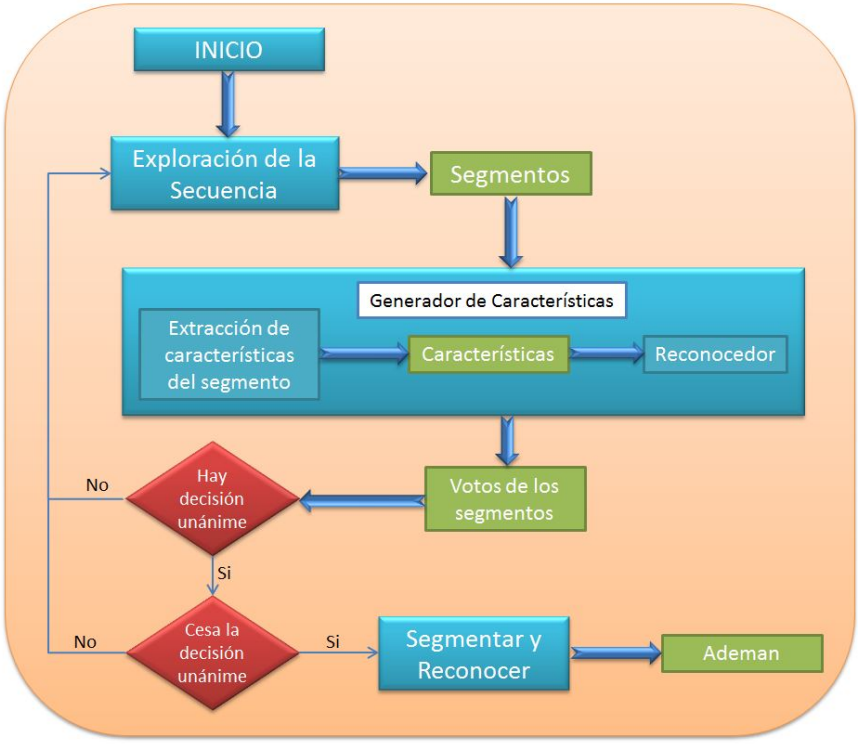


Figura 4.1: Diagrama general del método propuesto. Se generan segmentos de la secuencia los cuales se analizan con los modelos basados en HMMs y se generan sus correspondientes votos. Cuando las decisiones de los segmentos deja de ser mayoría, se detecta el punto final de una ademán y luego se puede reconocer.

A continuación, se describe con más detalle todo el proceso aquí introducido.

4.2. Múltiples ventanas superpuestas de tamaño dinámico

Para poder entender mejor el método de segmentación y reconocimiento simultáneo propuesto, se debe entender primero la técnica de exploración de la secuencia de video dada por el Kinect, donde la persona está ejecutando los ademanes.

Se debe idear un mecanismo de exploración de la secuencia que sea suficientemente eficiente para cubrirlo y además que permita delimitar la zona donde se encuentra el ademán. Esto último porque un reconocedor arroja mejores resultados si la información que se le suministra está libre de señales de ruido y sólo corresponde a la señal a identificar.

Una de las técnicas más usadas para hacer esta exploración, es con una ventana deslizante de tamaño fijo. Sin embargo, la mayor limitante de este esquema es que es muy poco probable que la ventana suministre exclusivamente la secuencia de observación completa del ademán. Incluso si el tamaño de esta ventana se acerca a la de la duración del ademán, su posicionamiento en la secuencia no necesariamente coincidirá con el ademán a buscar, como se puede apreciar en la Figura 4.2, donde ninguna de las t_i coincide exactamente con la duración del ademan A_2.

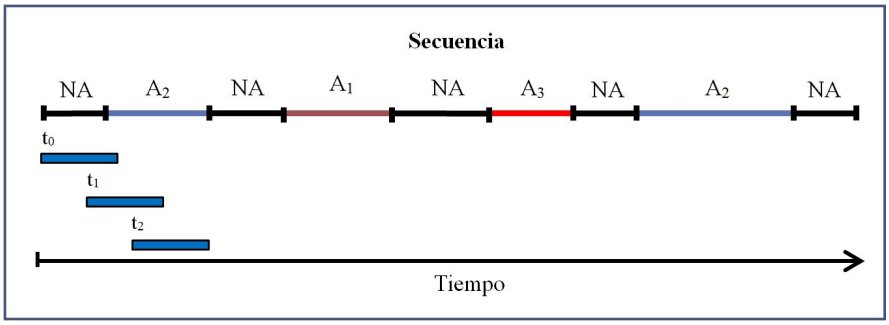


Figura 4.2: Esquema de ventana deslizante de tamaño fijo. En el instante t_0 se crea la ventana y esta cubre parte de No-ademán (NA) y parte del ademán 2 (A_2). En el instante t_1 se desliza la ventana a la derecha y aquí cubre parte de A_2 y un nuevo NA. Por lo tanto, la ventana pocas veces puede cubrir exactamente un ademán.

Por lo tanto, una solución más idónea sería utilizar varias ventanas superpuestas, con la esperanza de que alguna de ellas se ajuste al ademán. Además, en vez de ir deslizando las ventanas, podríamos mejor mantener en una posición fija las mismas, pero se va incrementando su tamaño.

Con base en las ideas antes expuestas, proponemos un esquema de varias ventanas (multi-ventanas), superpuestas y de crecimiento dinámico. Con este esquema se amplía el espacio de búsqueda de las fronteras del ademán, usando varias

ventanas y, adicionalmente, cada una tiene una cobertura distinta del espacio de búsqueda.

En la Figura 4.3 se muestra un esquema de esta propuesta, donde puede apreciarse que cada ventana tiene un punto de inicio distinto en la secuencia. Por lo tanto, en t_1, una vez transcurrido Δ tiempo (normalmente medido en cuadros), se crea una ventana v_1 de tamaño Δ. Posteriormente, en t_2 se amplía el tamaño de la ventana v_1 en Δ cuadros y se crea una nueva ventana v_2. Repitiendo este procedimiento en cada t_i, se llega a que en t_4 se tienen 4 ventanas, donde la ventana v_1 tiene un tamaño de $4.\Delta$, v_2 tiene un tamaño de $3.\Delta$, v_3 un tamaño de $2.\Delta$ y v_4 un tamaño de Δ cuadros.

Esto brinda la posibilidad de cubrir mejor la secuencia y disponer de una ventana que se ajuste relativamente bien al espacio donde ocurre el ademán. En la Figura 4.3, se aprecia que la ventana v_2 es la que mejor se ajusta al ademán A_3 en el instante de tiempo t_3. Por lo tanto, este esquema propuesto proporciona mayor posibilidad de encontrar los puntos de inicio y fin del ademán y en consecuencia, permitiría reconocer con mayor precisión qué ademán se ejecutó.

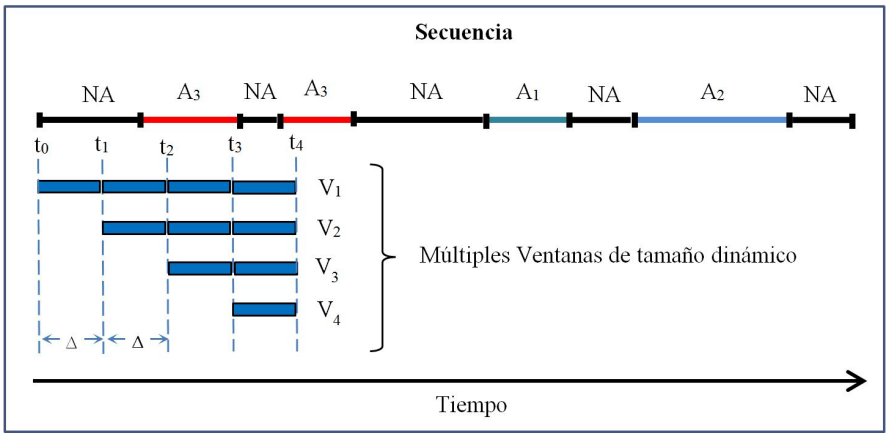


Figura 4.3: Esquema de múltiples ventanas de crecimiento dinámico para explorar de manera más eficiente el espacio de búsqueda de los puntos iniciales y finales de los ademanes.

Ahora bien, una vez que se dispone de estas ventanas, ¿cómo se puede usar la información que ellas proveen? La respuesta a esta pregunta se encuentra en la siguiente sección, donde se explica a detalle el método de segmentación y reconocimiento simultáneo.

4.3. Segmentación y reconocimiento simultáneo

Para que la técnica de múltiples ventanas superpuestas de tamaño dinámico pueda usarse para segmentar y reconocer ademanes simultáneamente, se debe disponer de predicciones de cada una de ellas sobre cuál ademán supuestamente se

está ejecutando en el espacio que ésta cubre. Por esta razón, éstas deben combinarse con un buen reconocedor y uno de los más usados en el estado del arte en el reconocimiento de ademanes, es el modelo oculto de Markov (HMM).

Asumimos que los HMMs serán inicialmente entrenados con ejemplos de ademanes segmentados manualmente, y por lo que serán capaces de identificar mejor el ademán en cuestión, en aquellas ventanas que mejor se ajusten al intervalo donde empieza y termina el ademán. Por lo tanto, teniendo un modelo HMM por cada ademán, de cada ventana se puede obtener tantas probabilidades por ademán y posteriormente, la mayor probabilidad corresponde a un voto que aporta la ventana al ademán correspondiente de esa probabilidad. Con los votos de las ventanas, se pueden tomar las decisiones de las mismas para identificar los puntos de inicio y fin de un ademán. Para ello lo que se requiere es identificar primero el punto final, es decir, lo más relevante es saber cuándo termina de ejecutarse un ademán, y posteriormente reconocerlo. Bajo este criterio, usando las votaciones de las ventanas en cada instante t_i, se debe identificar cuando un ademán acaba de terminar de ejecutarse.

Para lograr esto, se va revisando en cada instante t_i, la cantidad de votos asignados a cada ademán. Cuando algún ademán k supera un cierto umbral δ (medido en porcentaje), se puede decir que la mayoría de las ventanas están de acuerdo en que se está ejecutando dicho ademán. Posteriormente cuando los votos de ese ademán k decaen por debajo de δ, se puede inferir que el ademán k terminó de ejecutarse, dado que las ventanas ya no están de acuerdo en sus decisiones. Es entonces cuando se segmenta en dicho punto y se puede arrojar como resultado del reconocimiento, que se ejecutó el ademán k.

Por lo tanto, se propone un esquema de votación de múltiples ventanas para identificar el punto final del ademán y simultáneamente reconocer el mismo. Con lo antes descrito, se propone el Algoritmo 4.1.

En el algoritmo anterior, la variable t (paso 2 y 23) va registrando el tiempo (cuadro) donde se encuentra actualmente la señal de entrada (video del Kinect) y además indica el número de ventanas que existen en cada instante. La variable *band* (pasos 3, 12 y 14) permite identificar cuando se encuentra el punto final del ademán. Cuando los votos de un ademán superan el umbral δ, esta variable es verdadero (pasos 11 y 12) y posteriormente, cuando los votos bajan, si previamente nos encontrábamos con votos mayoritarios (paso 12) entonces es cuando se decide segmentar y reconocer (paso 15).

Los pasos 4, 19, 20 y 22 corresponden al método de múltiples ventanas superpuestas de tamaño dinámico. Específicamente, en el paso 4, primero se generan tantas ventanas hasta un cuadro de Inicio (*Frame* de Inicio, FI). Esto porque no vale la pena hacer una búsqueda para segmentar al comienzo de la secuencia, donde se espera sea una zona de no-ademán. Esto porque para nuestra aplicación destino de comandar un robot, es prácticamente imposible que el inicio de reconocimiento de ademanes por parte del robot, coincida exactamente con el inicio de ademán por parte del usuario. Por lo tanto, dependiendo del valor del

Algoritmo 4.1: Método de segmentación y reconocimiento simultáneo. Dentro del algoritmo se van creando las ventanas superpuestas y se va incrementando sus tamaños. Se usan dichas ventanas junto con modelos HMMs para segmentar el punto final de un ademán y simultáneamente reconocerlo.

```
1  while Exista señal de entrada do
2      t ← 0;
3      band ← falso ;
4      t ← Generar ventanas hasta FI;
5      while t < FP do
6          foreach cada ventana v_i existente do
7              Extraer características de v_i ;
8              Obtener las M probabilidades de v_i;
9              Determinar el ademán ganador de v_i y acumular el voto para el
               ademán;
10         end
11         if Num. de voto de un ademán > δ then
12             band ← verdadero;
13         else
14             if band then
15                 Segmentar y reconocer;
16                 Salir del While;
17             end
18         end
19         foreach cada ventana v_i existente do
20             Incrementar en Δ el tamaño de v_i ;
21         end
22         Generar nueva ventana;
23         t ← t + 1
24     end
25     Reiniciar;
26 end
```

parámetro *FI*, se crearán tantas ventanas que coincidirá con el tiempo t donde nos encontremos en la secuencia.

Por otra parte, en caso de no encontrarse con votos mayoritarios (superiores a δ), se debe tener una condición de parada. Esto se aplica en el paso 5, donde se tiene un cuadro de Parada (*Frame* de Parada, FP) de seguridad, por lo que cuando se supera el valor de este parámetro, se "reinicia" todo el proceso (paso 25); es decir, se eliminan las ventanas y se va a un nuevo ciclo, siempre que se tenga señal de entrada del Kinect (paso 1).

Los pasos 6, 16 y 19 son auto explicativos. Los pasos 7, 8, 9 y 15 merecen especial atención y por ende se explican con más detalle en las siguientes secciones, en ese mismo orden.

4.4. Extracción de características

De toda la información aportada por el Kinect (video RGB, video de profundidad, esqueleto y audio), solo se usará la concerniente a la posición de las manos en el esqueleto. Adicionalmente, como Arriaga et al (2011) reporta, es conveniente no solo usar la posición de las manos en términos de su tendencia cinemática, sino también la posición de las mismas en referencia a alguna parte del cuerpo. Por esta razón, también se considera la posición de las manos en referencia al cuello (que se obtiene del esqueleto): arriba-abajo o derecha-izquierda del mismo.

Se tienen dos posibles escenarios de características a considerar: sólo información de movimiento de las manos o movimiento de la manos junto con posición relativa. Veamos en detalle cada escenario.

4.4.1. Información de Movimiento

Supongamos que se dispone de las coordenadas (x, y, z) de cada mano. Entonces se puede crear una 6-tupla de valores discretos en el conjunto $0, 1$ por cada cuadro i con $i > 0$, de la secuencia capturada con el Kinect. Para cada i-ésimo cuadro de la secuencia, se tiene una *6-tupla* $T_i = (t_1^i, \cdots, t_6^i), i > 0$, tal que:

$$t_j^i = \begin{cases} 0 & t_j^i \leq t_j^{i-1} \\ 1 & t_j^i > t_j^{i-1} \end{cases} \quad 1 \leq j \leq 6 \tag{4.1}$$

Donde $1 \leq j \leq 3$ corresponde a las coordenadas (x, y, z) de la mano izquierda y $4 \leq j \leq 6$ corresponde a las coordenadas de la mano derecha. Es decir, cuando hay un incremento en alguna coordenada de alguna mano, se obtiene un 1 y en caso contrario se anota un 0.

4.4.2. Información de Movimiento y Postura

Para este escenario consideramos una sola mano con información de movimiento y postura, aunque se podría extender a considerar las dos manos.

Una vez obtenidas las coordenadas (x, y, z) de la mano derecha y las coordenadas (xc, yc) del centro de los hombros, que para efectos prácticos, lo simplificaremos como cuello; se crea una *5-tupla* de valores discretos en el conjunto $0, 1$ por cada frame i, con $i > 0$ de la secuencia capturada con el Kinect. Por lo tanto, para cada *i-ésimo* cuadro, se tiene una *5-tupla* $T_i = (t_1^i, \cdots, t_5^i), i > 0$, tal que:

$$t_j^i = \begin{cases} 0 & t_j^i \leq t_j^{i-1} \\ 1 & t_j^i > t_j^{i-1} \end{cases} \quad 1 \leq j \leq 3 \tag{4.2}$$

Donde $1 \leq j \leq 3$ corresponde a las coordenadas (x, y, z) de la mano derecha; es decir, se aplica el mismo criterio de cambio de dirección (cinemática) del caso anterior. Para los j restantes $(4 \leq j \leq 5)$, se aplica la posición respecto al cuello, considerando sólo las coordenadas (x, y) de la mano derecha (suponemos que no hay ademanes que se realicen por detrás del cuello, todos son frente al cuerpo). Por lo tanto, para estos dos componentes de la *5-tupla* se tiene que si la mano está a la derecha del cuello, su valor es 1 y 0 en caso contrario y si la mano está arriba del cuello, su valor es 1 y 0 en caso contrario. Esto es:

$$t_j^i = \begin{cases} 0 & t_{jm}^i \leq t_{jc}^i \\ 1 & t_{jm}^i > t_{jc}^i \end{cases} \quad 4 \leq j \leq 5 \tag{4.3}$$

donde t_{jc}^i corresponde a la coordenada "xc" ó "yc" del cuello, y t_{jm}^i corresponde a la coordenada "x" ó "y" de la mano derecha, en ese mismo orden y para un mismo frame j.

Como se observa, estos dos conjuntos de características son muy sencilla de extraer del esqueleto otorgado por el Kinect y además permiten discretizar los valores continuos para poder usarlos en los HMMs.

4.5. Obtención de las Probabilidades

Como ya se mencionó anteriormente, los modelos HMMs han sido ampliamente usados en tareas de reconocimiento de ademanes, y por esta razón se consideran en este trabajo de investigación. La manera de utilizar los HMMs para el reconocimiento de ademanes es como sigue. Suponiendo que se tienen M ademanes contemplados, entonces se deben crear y entrenar M modelos HMMs distintos, uno para cada ademán. Cada HMM es capaz de reconocer ademanes de una categoría. De este modo, se tienen λ_i conjuntos de parámetros, uno para cada ademán, con $1 \leq i \leq M$. Con esto se puede obtener M probabilidades $P(O \mid \lambda_i)$ distintas de una misma observación. Por lo tanto, para determinar el ademán que repre-

senta la observación O, se selecciona el índice i que arroja la mayor probabilidad, es decir, se calcula:

$$ademan_i = argmax_{1 \leq i \leq M} \left[P(O \mid \lambda_i) \right] \tag{4.4}$$

Supongamos que O son las características antes explicadas, provistas por cada ventana que exista en un tiempo determinado t_i, con lo que el tamaño de O varía de acuerdo a como varía el tamaño de cada ventana. Además, en un tiempo t_i cualquiera, con i ventanas existentes, se tendrán $i * M$ probabilidades, pero cada ventana v_i arrojará un ademán ganador de acuerdo a lo que se explicó anteriormente.

Por otra parte, el tipo de modelo HMM usado aquí es de la forma *Left-Right* o Bakis. Por lo tanto, siempre se empieza en un solo estado inicial, siempre se avanza hacia la derecha o al mismo estado, con transiciones entre estados contiguos, sin retornos y cuando se llega al último estado, no regresa a ningún otro.

Para el ajuste de sus parámetros $\lambda_i = (A, B, \Pi)$, se usó el algoritmo de Baum-Welch, teniendo cuidado en la inicialización de los valores iniciales de estos, para evitar un *underflow*. Para las inferencias, se utilizó el algoritmo de *Forward*, e igualmente para evitar *underflow*, se le aplicó escalamiento interno.

De igual modo, los modelos HMM tienen otros parámetros que ajustar para lograr un adecuado entrenamiento. Dichos parámetros y sus valores para todos los modelos HMM por igual, fueron:

1. Número de estados igual a 3.
2. Número de iteraciones de parada en caso de no superar el umbral de error igual a 200.
3. Valor del umbral de error a alcanzar (tolerancia) = 1×10^{-5}.

Estos valores se obtuvieron empíricamente luego de varias simulaciones. Con base en lo anterior, se dispone de M modelos HMM para inferir qué ademán representa una observación cualquiera y además, pueden ser usados para la segmentación y reconocimiento automático.

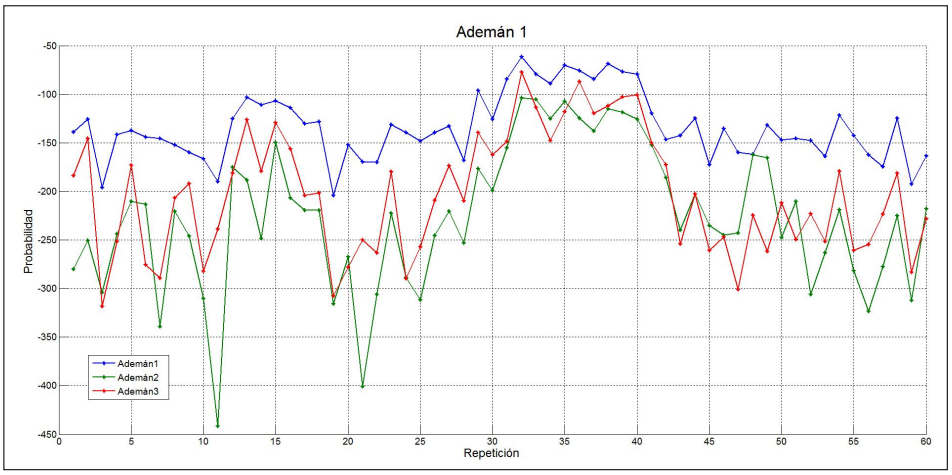
Otro asunto importante a destacar es que cuando se aplica el escalamiento interno al *Forward* lo que realmente se obtiene es $r_i^1 = \log(P(O \mid \lambda_i)), 1 \leq i \leq M$, como se puede apreciar en la Figura 4.4(a), donde se muestrá estos valores para 60 ejemplos de un mismo ademán 1. Por lo tanto, se debe realizar la siguiente transformación al resultado anterior:

$$r_i^2 = e^{\left(\frac{r_i^1}{10}\right)} \tag{4.5}$$

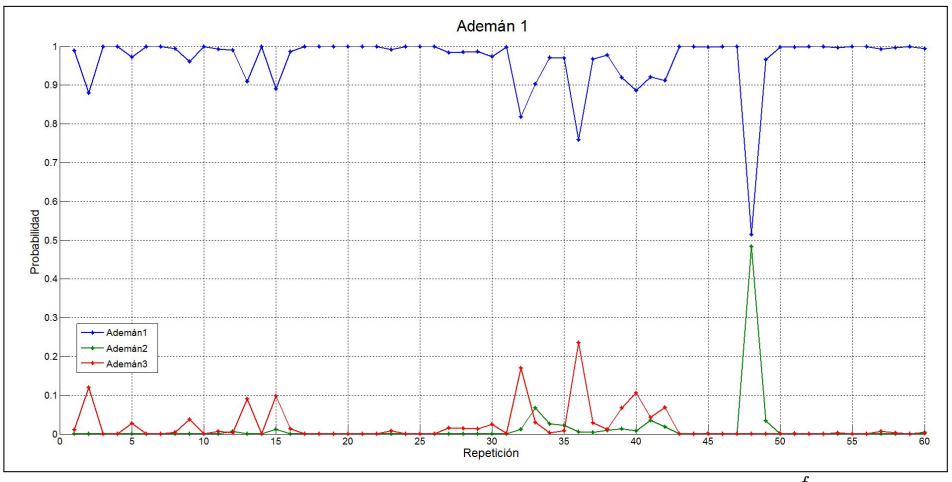
Además, se debe escalar los resultados obtenidos en la Ecuación (4.5) para llevarlos al intervalo $[0, 1]$ y que sumen 1 las probabilidades. Esto se logra de la siguiente forma:

$$r_i^f = \frac{r_i^2}{\sum_{i=1}^{M} r_i^2} \tag{4.6}$$

De esta forma, se obtienen valores de probabilidad como los que se muestran en la Figura 4.4(b), que son más apropiados para nuestro método.



(a) Probabilidades originales del *Forward* escalado (r_i^1)



(b) Probabilidades del *Forward* escalado transformadas(r_i^f)

Figura 4.4: Ejemplo de estimación de probabilidades para diferentes ventanas considerando 3 ademanes (1, 2, 3). (a) Resultados del *Forward* con escalamiento interno sin transformación (b) Probabilidades obtenidas luego de transformar y normalizar la salida del *Forward*

4.6. Esquema de Votación

Como se viene detallando en las secciones anteriores, para poder segmentar y posteriormente reconocer cada ademán en una secuencia continua, se necesita considerar las decisiones que arroja cada ventana de nuestro esquema de ventanas superpuestas de tamaño dinámico. Las decisiones de las ventanas obtenidas a través de las probabilidades de los HMMs como se explicó en la sección anterior se denominan "votaciones simples".

Una posible variante para el voto simple antes descrito, es el voto ponderado. La razón de esto viene del hecho de que en un instante t_i disponemos de i ventanas, de las cuales una o algunas de ellas se ajustarán mejor al intervalo del ademán, por lo que esta(s) ofrecerán información más certera del tipo de ademán existente. Se ideó un mecanismo de pesado de los votos de las ventanas, que recompensa a las ventanas que se acerquen mejor al tamaño esperado del ademán. Se contemplan tres posibles esquemas de pesado:

1. Usar el valor de la mayor probabilidad que arroja la ventana en el instante de tiempo t_i, es decir, el voto para el ademán es la probabilidad que este obtuvo en el HMM. Esto porque se espera que las ventanas más cercanas al intervalo del ademán, arrojen probabilidades más altas para el ademán que reconoce, en comparación con las demás ventanas.

2. Calcular la distribución de los tiempos de duración de los ademanes en los datos (de entrenamiento y pruebas), y en base a esto determinar los pesos que den mayor valor a las ventanas con tamaño cercanas al promedio de duración de los ademanes. En la Figura 4.5 se muestra un ejemplo de lo que se explica aquí.

3. Una combinación de los dos pesos anteriores. En este caso, multiplicando la probabilidad del ademán más probable de la ventana en el instante t_i por el peso correspondiente a su tamaño en ese mismo instante de tiempo.

En el siguiente capítulo se probarán estos cuatro esquemas de votos: voto simple y los tres tipos de votos pesados; para determinar cuál arroja mejores resultados para el método propuesto.

4.7. Determinación del punto final del ademán

Como ya se explicó, en el instante t_i en que el porcentaje de votos del ademán más votado cae por debajo de δ después de haber tenido una mayoría de votos, se considera ese t_i como el punto final del ademán. Sin embargo, si se toma exactamente t_i (en el punto) como el punto donde se tomará la decisión de cuál ademán se ejecutó, se puede estar en un momento donde no hay un ademán ganador, es

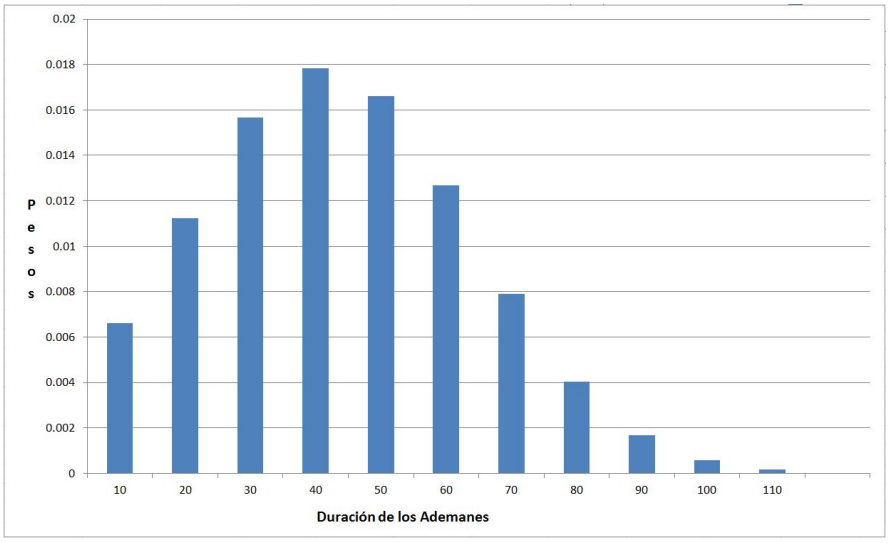


Figura 4.5: Ejemplo de pesos a aplicar a los votos de las ventanas según su tamaño. Las ventanas entre 30 y 50 frames tendrán mayor peso, mientras que las más pequeñas o muy grandes tendrán menor peso en su voto.

decir, se puede estar en un empate de dos o más ademanes. Aquí se podría decidir por el ademán que indique la ventana que está en el medio, esto es, la ventana $i/2$, con redondeo hacia arriba por ejemplo. Esto porque se espera que dicha ventana este mejor ajustada al intervalo donde se ejecutó el ademán. Este caso se muestra en la Figura 4.6(a).

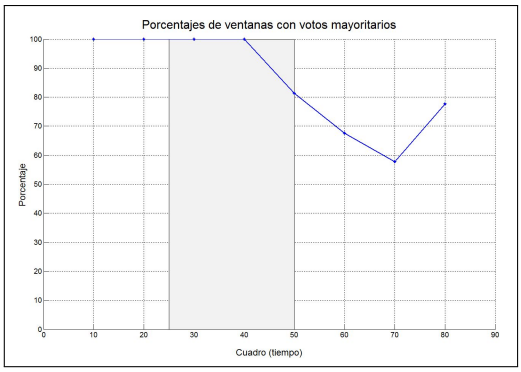
Una variante que resuelve el problema anterior, es segmentar en t_{i-1} (un paso atrás), con lo que ya nos encontraríamos en el instante en que existía un voto mayoritario (incluso podría ser decisión unánime, 100 % de los votos para un ademán) de las ventanas por un ademán k y entonces es fácil arrojar dicho ademán como resultado del reconocimiento. Este caso se presenta en el ejemplo de la Figura 4.6(b).

Otra posible variante, que se probó para verificar su efectividad, es segmentar en t_{i+1} (un paso adelante), con lo que la tarea de reconocimiento se podría encontrar en la misma situación que cuando se segmenta en el punto t_i.

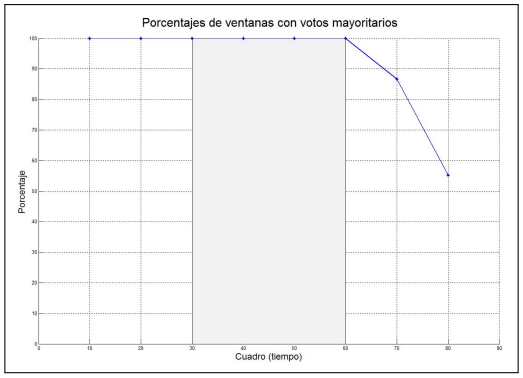
En el siguiente capítulo se muestran resultados de simulaciones comparando estos 3 esquemas para determinar el punto final del ademán.

4.8. Consideraciones Finales

Hasta ahora se ha descrito en detalle el método propuesto para segmentar y reconocer un ademán simultáneamente. Con el esquema de múltiples ventanas explicado, se van obteniendo i votos en cada instante de tiempo t_i, con lo que, al encontrar un instante donde las ventanas apoyan mayoritariamente un ademán k



(a) En este ejemplo conviene segmentar en t_i (en el punto)



(b) En este ejemplo conviene segmentar en t_{i-1} (punto atrás)

Figura 4.6: Ejemplos de porcentajes de votos mayoritarios. La zona en gris es donde ocurre el ademán y el resto son zonas de No-ademán. La linea azul es el porcentaje de votos mayoritario para un ademán (a) En el cuadro (*frame*) 50 la decisión de las ventanas deja de ser unánime, por lo que se considera éste como el punto final del ademán. (b) En este caso, es en el cuadro 70 donde las ventanas no están de acuerdo, pero el verdadero punto final del ademán está en el cuadro 60, por lo que conviene segmentar un paso atrás

en particular y posteriormente esta decisión cambia (las ventanas están indecisas), se presume el final del ademán k (segmentación) y se reconoce a este como el que acaba de ejecutarse.

Ahora bien, del Algoritmo 4.1 aún quedan varios asuntos por atender para el método aquí propuesto:

1. El porcentaje (umbral δ) de coincidencias de votos para tomarlos como mayoritarios y que permita identificar el punto final del ademán.

2. El cuadro de Inicio (FI) donde se empieza la búsqueda del punto final del ademán.

3. Hasta qué punto (Frame de Parada - FP) se debe buscar el punto final del ademán, es decir, cuál será nuestro frame de parada de seguridad.

Por otra parte, como se describe en el siguiente capítulo, para medir la precisión de segmentación del punto final del ademán, es conveniente otorgar un espacio de holgura alrededor del punto de segmentación que arroja este método y considerar si el punto final real se encuentra en dicha zona. Por lo tanto, supongase que el punto final real del ademán es pf_r, pero el método aquí propuesto determina que el punto final es pf_s, entonces se tiene:

$$dif = \mid pf_r - pf_s \mid \tag{4.7}$$

con lo que si $dif \leq holgura$ entonces el punto pf_s se considera como un acierto.

Los tres asuntos antes descritos, generan parámetros a ajustar para el método propuesto, además del tipo de voto y punto de segmentación. Decidimos encontrar los valores óptimos de los mismos de forma empírica, para lo cual se deben realizar un alto número de experimentos para lograr este cometido. Afortunadamente, esto sólo es necesario para entrenar el método. El modelo con los mejores parámetros es utilizado en tiempo real para segmentar y reconocer simultáneamente.

4.9. Resumen

En el presente capítulo se detalló el método de segmentación y reconocimiento simultáneo propuesto. Para este cometido, primero se explicó la técnica de exploración usada de la secuencia de observaciones, denominado múltiples ventanas superpuestas de tamaño dinámico. Luego se especificó cómo se usan dichas ventanas para identificar el punto final del ademán que se ejecutó. Finalmente, se describe como se combinan los resultados de varias ventanas mediante diferentes esquemas de pesado para reconocer el ademán.

Se dispone de un método de segmentación (solo el punto final) y reconocimiento simultáneo, usando solamente las probabilidades arrojadas por los modelos HMMs de cada ademán y las ventanas. Algo muy importante del método propuesto es que, a diferencia de muchos otros trabajos, no se necesita un modelo HMM para No-ademán, el cual es muy complicado de obtener. Esto porque un No-ademán es un conjunto infinito de movimientos posibles del usuario, que impiden lograr un entrenamiento adecuado de ese modelo HMM. Otro aporte importante de este método, es la ausencia de un pose que deba asumir el usuario para indicar la separación de cada ademán ejecutado, es decir, no se requiere que el usuario asuma cierto pose (ej. pose en forma de T) cuando termina y va a empezar cada ademán, que ayude a la segmentación de los mismos.

Por otra parte, el método aquí propuesto es fácilmente generalizable a cualquier reconocedor, dado que solo se necesita las decisiones que éste aporte para cada ventana que se tenga en cualquier momento. Del mismo modo se pueden cambiar las características extraídas de la secuencia de video para alimentar el reconocedor, el sistema de visión usado y el conjunto de datos.

Una vez decidido los valores óptimos de los parámetros y el esquema de pesado de los votos de las ventanas más idóneo, se dispone de un método eficaz, sencillo y poco costoso computacionalmente, para hacer una segmentación del punto final del ademán y simultáneamente reconocer el ademán que acaba de terminar de ejecutarse, en una secuencia continua de movimientos de los brazos de una persona.

En el siguiente capítulo mostramos los resultados de la evaluación experimental del método descrito en este capítulo.

Capítulo 5

Experimentos y resultados

En el presente capítulo se presentarán los experimentos realizadas y los resultados alcanzados en cada uno de ellas. Se presentan tres tipos de experimentos, unos en condiciones controladas, otros bajo el entorno del robot y otros en secuencias con varias ademanes. Antes de describir los experimentos, se detalla el entorno de los mismos, la configuración de los datos usados y las técnicas de evaluación de los resultados.

5.1. Entorno de los experimentos

Los experimentos fueron realizados con el uso de dos equipos computacionales distintos, uno para cada fase de los mismos. Para la captura de los ademanes con el Kinect, se usó una portátil Utrabook, procesador i5-3427U 1.8 GHz, 6 Gb de memoria RAM, 120Gb de SSD y Windows 7. Para la realización de los experimentos, se usó una computadora de escritorio, con procesador i7-2600K 3.4 GHz, 24 Gb de memoria RAM, 250 Gb de SSD, tarjeta de video NVIDIA GeForce GT 430 y Windows 8.

Para los experimentos en el entorno del robot, se utilizó un robot PatrolBot con un anillo de sonares, dos ruedas principales y cuatro de apoyo, dos motores con codificadores, un laser SICK LMS200, una video cámara Canon VCC5, un micrófono direccional SHURE SM81, altavoces, un PC integrado, dos portátiles (uno con Linux y otro con Windows), un brazo 6M Katana y dos Kinects (ver Figura 5.7).

Además, para la captura de los ademanes su usó NuiCapture V1.4, bajo Windows 7 y el SDK Kinect para Windows V1.6. Para la realización de los experimentos con los datos capturados, se usó Matlab R2010a V7.10.0.499 y la caja de herramientas estadística (Statistics Toolbox).

5.2. Configuración de los datos

En este trabajo de investigación es necesario disponer de datos de ademanes segmentados para el entrenamiento de los HMMs, así como de secuencias de varios ademanes no segmentados para pruebas.

Para esto se usó el Kinect junto con una herramienta de captura que permitiera identificar visualmente donde empieza y donde termina un ademán, para segmentarlo manualmente. Esta no es una tarea trivial porque algunos ademanes, por su naturaleza cinemática, tienen movimientos que hacen un poco difícil esta tarea. Se prepararon dos tipos de datos por separado: de entrenamiento y de prueba.

Para los datos de entrenamiento, a cada usuario se le solicitaba que repitiera n veces un mismo ademán en la misma sesión de grabación. Esto se repetía para cada ademán por cada usuario, hasta obtener la cantidad de datos suficientes para poder entrenar los modelos HMMs.

Para los datos de prueba, se le solicitó al usuario que ejecutara cada ademán en forma aleatoria en una misma sesión de grabación. La única restricción que se le impuso, es que realizara al menos una vez cada ademán, pero no necesariamente en forma consecutiva. Se realizaron al menos dos sesiones de grabación por usuario para recabar un número suficiente de repeticiones de cada ademán.

Nótese que el número de repeticiones de los datos de entrenamiento es un valor **n** fijo e igual para cada ademán. En cambio, el número de repeticiones de los datos de prueba para cada ademán no es igual. Esto porque no se tiene control del número de repeticiones que ejecutará cada usuario de cada ademán. Esta decisión se deja al libre albedrío del usuario.

Una vez capturados los datos de entrenamiento y prueba, estos se dispusieron como se aprecia en la Figura 5.1, para ser usados en cada caso según se aprecia en dicha gráfica. Además, tanto los datos de entrenamiento como los de prueba fueron segmentados en el punto de inicio y punto final de cada ademán, lo cual llamaremos **segmentación manual**. Por otra parte, además en los datos de prueba se aisló cada ademán, pero dejando zonas de no ademanes a cada lado del ademán. Se procuró determinar el punto medio entre el fin de un ademán y el inicio del siguiente, de modo tal que cada ademán tuviese una zona de no ademán equitativo y a esto lo llamaremos **sin segmentación**.

5.3. Evaluación

Para evaluar la calidad de entrenamiento de los modelos HMMs y para determinar el valor de cota superior que puede alcanzar nuestro método de segmentación y reconocimiento automático, se usó la técnica de validación cruzada dejando uno por fuera (*Leave-One-Out Cross-Validation* - LOOCV). Básicamente, esta técnica es una validación cruzada de K grupos (*K-Fold Cross-Validation*), donde el va-

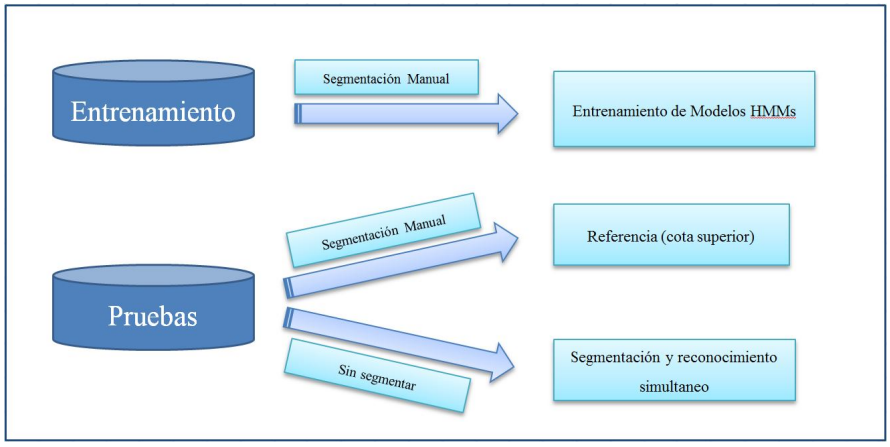


Figura 5.1: Configuración de los datos. Los datos de entrenamiento segmentados manualmente se usan para estimar los parámetros de los modelos HMMs. Los datos de pruebas segmentados manualmente se usan para determinar la tasa de reconocimiento de referencia (cota superior) y sin segmentar para la segmentación y reconocimiento automático

lor de k es n, siendo n el número de datos que se dispone. Este procedimiento se realizó con los datos de entrenamiento y con los datos de prueba, con segmentación manual.

Por otra parte, para evaluar el método aquí propuesto, no existe en el estado del arte un consenso sobre las medidas más adecuadas para determinar la calidad de un método de segmentación y reconocimiento simultáneo. Por esta razón, se propone el uso de dos medidas ampliamente conocidas y utilizadas en aprendizaje computacional y reconocimiento de patrones: cobertura (*recall*) y precisión (*precision*).

Cobertura se refiere a la proporción entre el número de datos de interés (correctos) obtenidos y el total de datos de interés que se desea obtener. Esto es:

$$Cobertura = \frac{tp}{tp + fn} \tag{5.1}$$

donde tp *(true positive)* son los datos correctos recuperados y fn *(false negative)* son los datos correctos no recuperados o ausentes.

Para este trabajo, la cobertura se aplicó para medir la eficiencia de segmentación del método en estudio y se calcula de la siguiente forma:

$$Cobertura = \frac{\#ademanes - segmentados - correctamente}{total - de - ademanes - de - prueba} \tag{5.2}$$

Es decir, se desea segmentar correctamente todos los ejemplos de prueba con un ademán contenido, por lo que la cobertura será el porcentaje de ejemplos que se logren segmentar "relativamente" bien de ese total. El término "relativamente"

se usa debido a que no es necesaria una segmentación precisa en el punto donde termina el ademán. Basta con disponer de cierta holgura que sea aceptable para poder identificar el ademán en cuestión, que sería el objetivo final más relevante. El valor de esta holgura se varía, de acuerdo a la exigencia de precisión que deseamos en la segmentación; es decir, un valor de holgura pequeño exigirá mayor precisión en la segmentación.

Precisión se refiere a la proporción entre el número de datos de interés (correctos) obtenidos y el total de datos que se obtuvieron (correctos o incorrectos). Esto es:

$$Precisión = \frac{tp}{tp + fp} \tag{5.3}$$

donde fp *(false positive)* son los datos incorrectos recuperados.

Para este trabajo se usó una variación de esta medida para medir la calidad de reconocimiento del método propuesto. Esto porque ésta propuesta solo aplica el reconocimiento cuando ya se tiene segmentado correctamente el ademán en cuestión. Por lo tanto, esto se calcula de la siguiente forma:

$$Precisión = \frac{\#ademanes - reconocidos - correctamente}{total - de - ademanes - segmentados - correctamente} \tag{5.4}$$

Por otra parte, es bien conocido que existe un compromiso entre estos dos valores; es decir, en la mayoría de las técnicas de recuperación de información, es difícil tener altos valores en ambas medidas, dado que están relacionados de forma inversa. En casos donde la cobertura alcanza altos valores, la precisión es baja y de igual modo, en forma inversa. Por esta razón, existen medidas que indican un posible balance entre estos dos valores. Entre las más usadas, se encuentra la medida-f (*f-measure*), la cual proporciona un indicativo de compromiso entre la cobertura y la precisión. Su fórmula es:

$$F_\beta = (1 + \beta^2).\frac{precision.recall}{(\beta^2.precision) + recall} \tag{5.5}$$

El valor de β más usado es de 1, con lo que se obtiene el promedio harmónico, y la fórmula anterior queda:

$$F_1 = 2.\frac{precision.recall}{precision + recall} \tag{5.6}$$

Todas estas medidas (recall, precision y f-measure) tiene valores en el intervalo $[0, 1]$, donde 1 es el mejor valor posible.

Se utilizarán estas medidas para determinar qué tan bueno es el método propuesto en estudio con los valores de los parámetros usados.

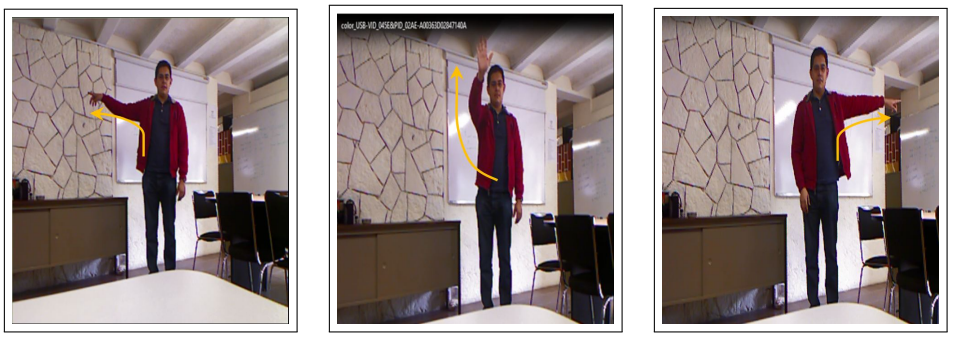
5.4. Experimentos 1: Escenario controlado

El objetivo de este primer experimento es evaluar el método propuesto en un escenario controlado con pocos ademanes. En particular se desea analizar el efecto de los diferentes parámetros del método en las medidas de evaluación. A continuación se detallan los experimentos para verificar que el método de segmentación y reconocimiento simultáneo funciona con un conjunto de ademanes sencillos, en un ambiente controlado (sin el robot) y fuera de línea. Se presentan los resultados obtenidos al variar el tamaño de crecimiento de la ventana (Δ), el umbral para voto mayoritario (δ), cuadro de inicio (FI), el cuadro de parada (FP) y el tipo de segmentación (punto atrás, en el punto y punto adelante); los cuales se analizan al final de la sección.

5.4.1. Datos utilizados

Los ademanes contemplados en esta fase, fueron solo tres y no necesariamente pensados para formar parte del conjunto final de ademanes para instruir al robot.

Los ademanes utilizados son: cruce a la derecha, cruce a la izquierda y alto. Imágenes ilustrando la trayectoria esperada y punto final de cada ademán pueden apreciarse en la Figura 5.2.



(a) Cruce a la derecha (b) Alto (c) Cruce a la izquierda

Figura 5.2: Conjunto de ademanes usados para los experimentos iniciales. Se muestra en la imagen el punto final de cada ademán y una trayectoria aproximada (flecha amarilla)

Como se explicó en la sección 5.2, se prepararon dos grupos de datos: los de entrenamiento con segmentación manual y los de prueba con segmentación manual y sin segmentar.

Con la configuración del HMM descrita en la sección 4.5 se procedió a realizar la captura de los ademanes antes descritos. Se capturaron 60 repeticiones de cada ademán, con tres usuarios distintos (20 repeticiones por usuario). Con estos

ademanes con segmentación manual, se entrenarán tres HMMs, uno para cada ademán.

5.4.2. Resultados con segmentación manual

Tanto a los datos de entrenamiento como a los de prueba se les aplicó segmentación manual. Esto es, usando una herramienta gráfica, se visualizó el punto de inicio y final de cada ademán y se separó dicho segmento de la secuencia completa. Dado que este trabajo solo fue realizado por una persona, no se tienen problemas de concordancia y fiabilidad entre observadores. En términos intra observador, se procuró mantener cierta concordancia sobre dónde se encuentran los puntos de inicio y fin de cada ademán y realizar una segmentación mas o menos homogénea en los dos conjuntos de datos.

Ahora bien, en esta sección se presentan los resultados del entrenamiento de los modelos HMM y para verificar la calidad de reconocimiento de los mismos, se aplicó LOOCV, con lo que se obtuvo la matriz de confusión de la Tabla 5.1.

Tabla 5.1: Matriz de confusión de las pruebas de *LOOCV* con tres ademanes con segmentación manual para los datos de entrenamiento

Matriz de Confusión			
Ademán	1	2	3
1: Cruce a la derecha	58	1	1
2: Alto	0	57	3
3: Cruce a la izquierda	0	0	60

Con lo anterior se verifica que la tasa de reconocimiento de los HMMs para estos ademanes es de 97.2 %. Por lo tanto, se verifica que los modelos son efectivos para reconocer ademanes segmentados manualmente.

Por otra parte, se realizó una prueba análoga con los ademanes de prueba, segmentados manualmente. Esto se hizo para disponer de un valor de referencia (cota superior) para verificar que el método de segmentación y reconocimiento automático arroje resultados cercanos a este valor. Se espera que el método de segmentación y reconocimiento simultáneo tenga una tasa de reconocimiento menor a la segmentación manual, por lo que si se acerca a esta última se considera un buen resultado

Se usó los modelos ya entrenados con los datos de entrenamiento y se evalúan los datos de prueba con segmentación manual. Esto arrojó los resultados de la Tabla 5.2.

Con lo que la tasa de precisión fue de 85.96 %. Es importante acotar que el porcentaje de reconocimiento se redujo debido a que estos datos de pruebas fueron capturados en ocasiones distintas a los datos de entrenamiento. A pesar de

Tabla 5.2: Matriz de confusión de las pruebas de *LOOCV* con los datos de prueba con segmentación manual y modelos entrenados con los datos de entrenamiento.

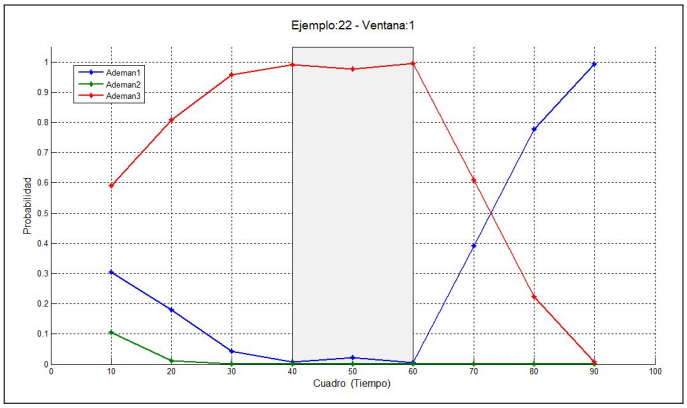
Matriz de Confusión			
Ademán	1	2	3
1: Cruce a la derecha	15	1	2
2: Alto	1	16	1
3: Cruce a la izquierda	0	3	18

que son los mismos usuarios, se evidencia la variabilidad espacio-temporal de los ademanes. Con estos resultados se procede a evaluar el método de segmentación y reconocimiento simultáneo propuesto, dado que se tiene garantía de que tiene una tasa de reconocimiento aceptable para ademanes con segmentación manual. Además, se tiene un valor de referencia con el cual comparar el método de segmentación y reconocimiento automático. Es decir, se espera alcanzar un valor cercano a 85.96 % de reconocimiento con nuestro método.

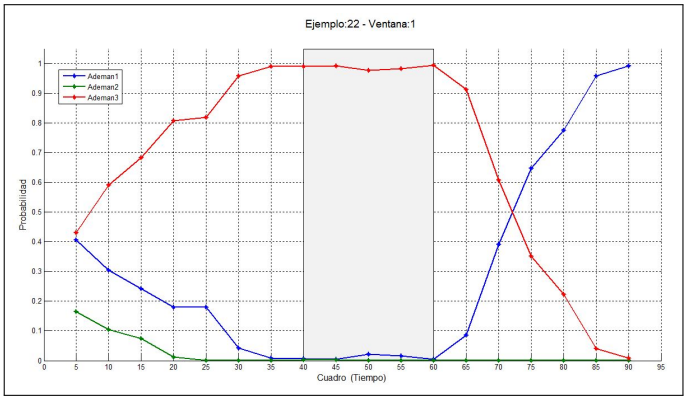
5.4.3. Resultados para segmentación y reconocimiento simultáneo

En este experimento se utilizó solamente el esquema de votación simple, analizando la efectividad del método al variar los siguientes parámetros: tamaño de crecimiento de la ventana (Δ), el umbral para voto mayoritario (δ), cuadro de inicio (FI), el cuadro de parada (FP) y el tipo de segmentación (punto atrás, en el punto y punto adelante). Se capturaron varias secuencias de ademanes combinados. De estas secuencias se generaron ejemplos con un sólo ademán y zonas de no ademán a ambos lados. Se obtuvieron 57 ejemplos, de los cuales 18 correspondían al ademán 1, 18 al ademán 2 y 21 para el ademán 3. En promedio, los ejemplos contienen ademanes de 37 cuadros de duración. El ademán más rápido fue de 15 cuadros y el más lento de 50 cuadros.

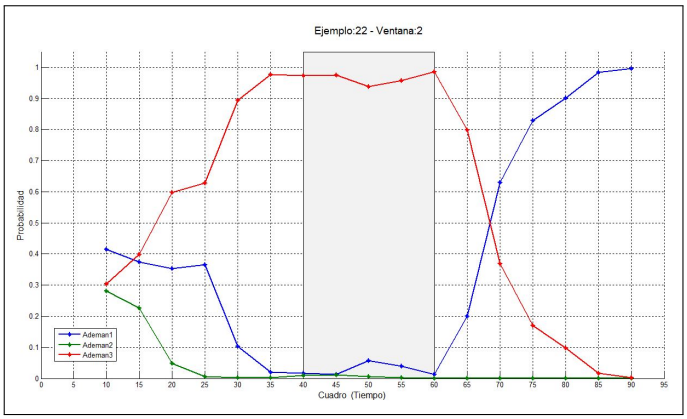
Un asunto a resolver antes de aplicar el método de segmentación y reconocimiento automático, es el relacionado con el tamaño de crecimiento de las ventanas (valor de Δ) descrito en la sección 4.2. Para esto, se observaron los valores de las probabilidades en estos ejemplos, para $\Delta = 5$ y para $\Delta = 10$. En la Figura 5.3 se muestra el resultado de uno de los ejemplos en los dos casos. En la Figura 5.3(a) se tiene un $\Delta = 10$ y sólo se muestra la primera ventana (v_1) que es la que cubre la secuencia desde el cuadro 0. En las Figuras 5.3(b) y 5.3(c), el valor de Δ es 5, y se muestran las primeras ventanas v_1 y v_2 dado que, al igual que la ventana anterior, prácticamente cubren toda la secuencia, con la diferencia de que v_2 empieza en el frame 5.



(a) Ventana 1 con $\Delta = 10$



(b) Ventana 1 con $\Delta = 5$



(c) Ventana 2 con $\Delta = 5$

Figura 5.3: Probabilidades de los 3 ademanes (cada HMM) para el mejor ejemplo. Solo se muestra las probabilidades de la ventana 1 y 2, que cubre todo el ejemplo. En (a) $\Delta = 10$ y comienza en el cuadro 0, en (b) $\Delta = 5$ y comienza en el cuadro 0 y en (c) $\Delta = 5$ y comienza en el cuadro 5

Tabla 5.3: Resultados de cobertura y precisión para el experimento 1, variando δ, FI y FP.

δ	FI	FP	Cobertura	Precisión
100	20	60	61.4	65.72
		70	56.14	68.76
		80	52.63	70.00
	30	60	**70.18**	67.50
		70	63.16	69.44
		80	59.65	**70.60**
	40	60	61.4	62.87
		70	49.12	64.29
		80	45.61	69.24
90	20	60	61.4	65.72
		70	56.14	68.76
		80	52.63	70.00
	30	60	**70.18**	67.50
		70	63.16	69.44
		80	59.65	**70.60**
	40	60	61.4	62.87
		70	49.12	64.29
		80	45.61	69.24
80	20	60	57.89	69.70
		70	54.39	70.97
		80	54.39	**70.97**
	30	60	**68.42**	69.23
		70	61.40	68.58
		80	61.40	68.58
	40	60	59.65	64.71
		70	47.37	62.95
		80	47.37	66.67

De lo anterior se desprende que el resultado de la clasificación no cambia significativamente en cualquiera de los valores de Δ. Es decir, para ambos valores de crecimiento de las ventanas, el resultado de clasificación es similar en cada cuadro, con una pequeña variación en la ventana 2, donde en los cuadros 10 y 70 son otros ademanes los ganadores, pero esto no afecta nuestro método de segmentación y reconocimiento automático. Para otros ejemplos y valores de Δ se obtuvieron resultados parecidos. Por lo tanto, se decidió realizar todas los experimentos que siguen con $\Delta = 10$.

Otro parámetro que se puede fijar es la tasa de votos mayoritarios (umbral ó δ) desde donde se empezará la búsqueda del punto final del ademán (segmentación). Para esto se realizaron varios experimentos moviendo este valor, el cuadro de inicio (FI) y el cuadro de parada (FP). Los resultados se muestran en la Tabla 5.3 donde se resaltan en negrita los mejores.

De lo anterior se puede observar que los mejores resultados se obtienen para valores de umbrales de 100 % (decisión unánime de las ventanas) y de 90 %. Además, para estos dos valores de δ, los resultados son exactamente los mismos. Por lo tanto, podemos fijar el valor del umbral δ en 100 % y verificar las mejoras que podemos alcanzar variando los otros parámetros.

Hasta ahora, los parámetros que debemos optimizar en nuestro método son

FI, FP y el punto de segmentación. Además, se puede verificar los resultados que se obtendrían si se varía la holgura. Como se mostró en la Tabla 5.3, el conjunto de valores usados para $FI = 20, 30, 40$ y para $FP = 60, 70, 80$. Estos valores se consideraron de acuerdo a las estadísticas obtenidas de los ejemplos de pruebas, donde se midió el promedio del punto de inicio de los ademanes y el promedio de duración de toda la secuencia de los ejemplos.

Para holgura se consideraron los valores $5, 10, 15, 20$; es decir, se partió desde una precisión alta en la segmentación (holgura de 5), hasta una relajación en lo máximo posible (holgura de 20). Una holgura mayor a 20, aunque podría mejorar la tasa de segmentación (cobertura), no mejoraría la de precisión, porque la detección del punto final no sería tan buena para identificar el ademán en cuestión. Esto se puede evidenciar en la Tabla 5.4, donde se fijó FI = 30, FP = 60 y el punto de segmentación (PS) un paso adelante (t_{i+1}). En la misma se evidencia que a mayor holgura, mayor cobertura, pero menor precisión. Para efectos de nuestra aplicación (comandar robots de servicio), es preferible mayor seguridad sobre cuál ademán se está ejecutando (precisión), más que cuándo termina de ejecutarse dicho ademán (cobertura). Sin embargo, si se tiene una baja tasa de cobertura, obligaría al usuario a repetir muchas veces el ademán, hasta conseguir una buena segmentación, que por ende ocasiona una mejor precisión.

Tabla 5.4: Resultados para varios valores de holgura, fijando FI = 30, FP = 60 y PS = t_{i+1}.

Holgura	**Cobertura**	**Precisión**
5	26.32 %	79.98 %
10	50.88 %	68.97 %
15	56.14 %	68.76 %
20	73.68 %	66.66 %

Una vez establecidos los dominios de valores de los parámetros que faltan por optimizar, se realizaron experimentos exhaustivos de todos ellos para verificar en cuáles se obtenían mejores resultados así como la sensitividad de los resultados respecto a estos parámetros.

Para visualizar mejor lo resultados, se separaron en tres grandes grupos, de acuerdo al parámetro de punto de segmentación y se combinaron los demás parámetros FI y FP. Además se muestra la calidad de la cobertura variando la holgura.

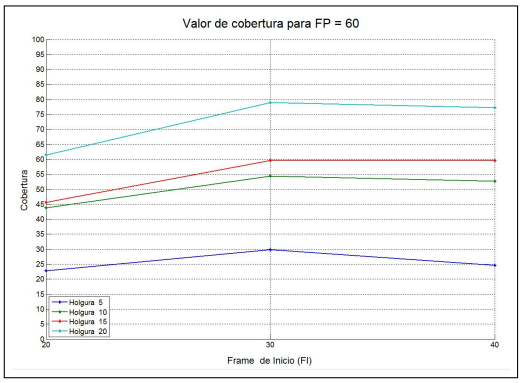
Segmentación un punto atrás.

En estos experimentos se considera como punto final t_{i-1}, es decir, un paso atrás de donde la decisión de las ventanas ya no es unánime. Los resultados se resumen en la Tabla 5.5.

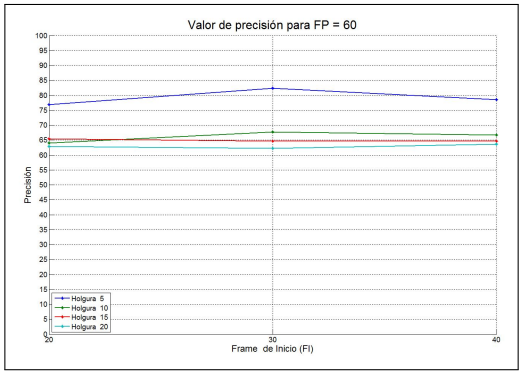
Puede observarse que cuando FP es 60, se obtienen mejores resultados de cobertura. Para visualizar mejor esto, se muestran gráficas de estos datos, fijando

Tabla 5.5: Resultados del método de segmentación y reconocimiento automático, segmentando un paso atrás. En negrilla se resalta los mejores resultados.

FI	**FP**	**Holgura**							
		5		10		15		20	
		Cober.	Prec.	Cober.	Prec.	Cober.	Prec.	Cober.	Prec.
20	60	22.81	76.92	43.86	64.00	45.61	65.38	61.40	62.86
	70	21.05	83.33	42.11	66.67	42.11	66.67	59.65	61.76
	80	21.05	83.33	40.35	69.57	40.35	69.57	59.65	61.76
30	60	**29.82**	82.35	**54.39**	67.74	**59.65**	64.71	**78.95**	62.22
	70	29.82	88.24	52.63	70.00	54.39	67.74	71.93	65.85
	80	28.07	87.50	50.88	72.41	52.63	70.00	71.93	65.85
40	60	24.56	78.57	52.63	66.67	59.65	64.71	77.19	63.64
	70	22.81	92.31	45.61	69.23	49.12	67.86	71.93	63.41
	80	22.81	**92.31**	42.11	**75.00**	45.61	**73.08**	70.18	**67.50**



(a) Cobertura con FP = 60 y segmentando un paso atras



(b) Precisión con FP = 60 y segmentando un paso atras

Figura 5.4: Gráficas de cobertura (a) y precisión (b), fijando FP en 60 y segmentado un paso atrás. Se va variando los valores de FI y de holgura.

FP en 60, como se aprecia en la Figura 5.4.

De la figura anterior, puede observarse que, como se esperaba, a mayor holgura mejor cobertura, pero menor precisión y en sentido inverso, a menor holgura, mayor precisión. Otro aspecto a resaltar, es que en ambos casos, se obtienen máximos locales cuando FI = 30, por lo que podría decirse que éste es un buen valor para este parámetro, al igual que 60 para FP.

Segmentación en el punto.

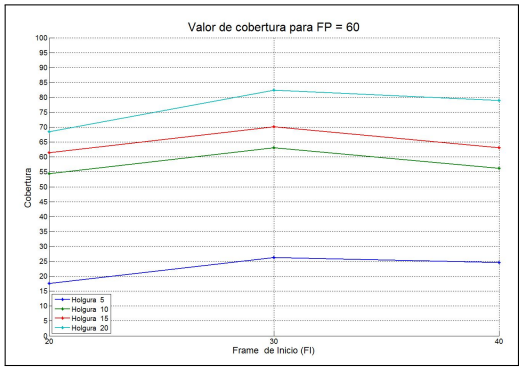
En estos experimentos se considera como punto final t_i, es decir, en el punto donde la decisión de las ventanas ya no es unánime. Los resultados se resumen en la Tabla 5.6.

Del mismo modo que en el caso anterior, los mejores resultados están cuando FP = 60. Si repetimos las gráficas, obtenemos lo que se muestra en la Figura 5.5.

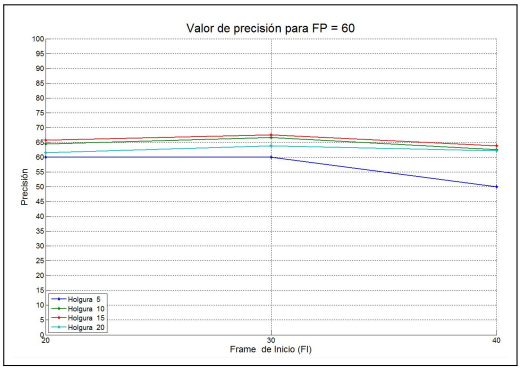
La mayor diferencia con respecto al caso anterior, es que para una holgura

Tabla 5.6: Resultados del método de segmentación y reconocimiento automático, segmentando en el punto. En negrilla se resalta los mejores resultados.

FI	**FP**	**Holgura**							
		5		10		15		20	
		Cober.	Prec.	Cober.	Prec.	Cober.	Prec.	Cober.	Prec.
20	60	17.54	60.00	54.39	64.52	61.40	65.71	68.42	61.54
	70	15.79	**66.67**	52.63	66.67	57.89	66.67	66.67	60.53
	80	14.04	62.50	50.88	**68.97**	56.14	68.75	66.67	60.53
30	60	**26.32**	60.00	**63.16**	66.67	**70.18**	67.50	**82.46**	63.83
	70	21.05	58.33	61.40	65.71	64.91	67.57	75.44	**65.12**
	80	17.54	50.00	59.65	67.65	63.16	**69.44**	75.44	**65.12**
40	60	24.56	50.00	56.14	62.50	63.16	63.89	78.95	62.22
	70	17.54	50.00	49.12	60.71	52.63	63.33	73.68	59.52
	80	15.79	44.44	47.37	66.67	50.88	68.97	70.18	65.00



(a) Cobertura con FP = 60 y segmentando en el punto



(b) Precisión con FP = 60 y segmentando en el punto

Figura 5.5: Gráficas de cobertura (a) y precisión (b), fijando FP en 60 y segmentado en el punto. Se va variando los valores de FI y de holgura

baja, no es mejor la precisión. Sin embargo, se repite el hecho de que los mejores resultados son para un FI = 30.

Segmentación un punto adelante.

En estos experimentos se considera como punto final t_{i+1}, es decir, en un punto adelante del punto donde la decisión de las ventanas ya no es unánime. Los resultados se resumen en la Tabla 5.7.

Nuevamente, repetimos las mismas gráficas fijando FP en 60, con lo que obtenemos lo que se muestra en la Figura 5.6.

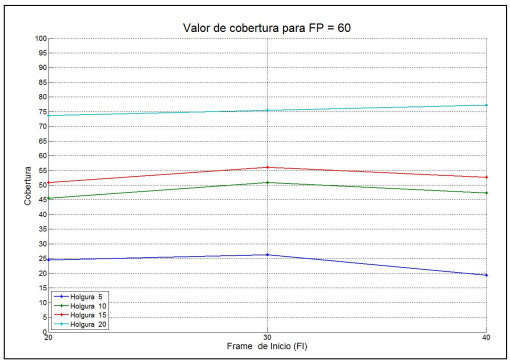
Nuevamente la precisión mejora cuando la holgura es pequeña y los mejores resultados son para FI = 30

5.4.4. Conclusiones

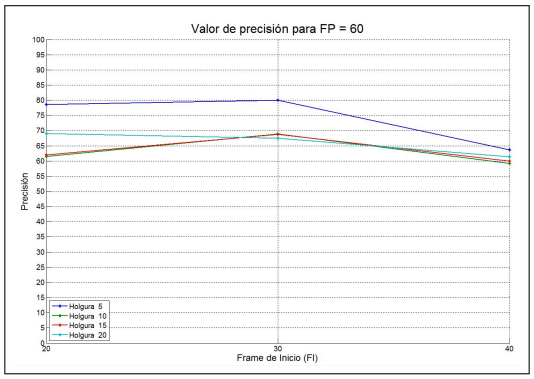
De los experimentos anteriores podemos concluir lo siguiente:

Tabla 5.7: Resultados del método de segmentación y reconocimiento automático, segmentando un paso adelante. En negrilla se resalta los mejores resultados.

FI	**FP**	**Holgura**							
		5		10		15		20	
		Cober.	Prec.	Cober.	Prec.	Cober.	Prec.	Cober.	Prec.
20	60	24.56	78.57	45.61	61.54	50.88	62.07	73.68	**69.05**
	70	21.05	**83.33**	42.11	58.33	45.61	57.69	71.93	63.41
	80	19.30	81.82	38.60	59.09	42.11	58.33	71.93	63.41
30	60	**26.32**	80.00	**50.88**	**68.97**	**56.14**	**68.75**	**75.44**	67.44
	70	19.30	81.82	42.11	62.50	43.86	64.00	68.42	58.97
	80	15.79	77.78	38.60	63.64	40.35	65.22	68.42	58.97
40	60	19.30	63.64	47.37	59.26	52.63	60.00	77.19	61.36
	70	10.53	66.67	33.33	47.37	35.09	50.00	71.93	48.78
	80	10.53	66.67	31.58	55.56	33.33	57.89	68.42	53.85



(a) Cobertura con FP = 60 y segmentando un punto adelante



(b) Precisión con FP = 60 y segmentando un punto adelante

Figura 5.6: Gráficas de cobertura (a) y precisión (b), fijando FP en 60 y segmentado un paso adelante. Se va variando los valores de FI y de holgura

1. Con los resultados obtenidos en este escenario controlado, se puede ampliar el número de ademanes a usar, pensando ya en comandar un robot de servicio.

2. Los mejores resultados se obtienen, en general, al considerar FI=30, FP=60 y segmentando un paso atrás.

3. Se puede observar el compromiso entre cobertura y precisión. Por ejemplo, para coberturas en el orden de 22 % se logran precisiones de más de 90 %; pero para coberturas mayores a 50 %, la precisión es menor al 75 %. La mejor combinación dependerá de la aplicación.

4. Se observa que en general el método no es muy sensible a los parámetros, en particular con FI y FP.

Para el experimento 2 se incluyen, además de los parámetros anteriores, las opciones de pesado de las ventanas. Para esto se definió un nuevo conjunto de ademanes pensados ya en comandar el robot, lo cual es lo que se muestra en la siguiente sección.

5.5. Experimentos 2: Entorno de un robot de servicio

A continuación se describen los experimentos realizados usando el robot de servicio Sabina (Aviles-Arriaga et al., 2009) del laboratorio de Robótica del INAOE. Es importante aclarar que el método propuesto se evaluó en un ambiente simulado para un robot de servicio (similar a una casa) y con los sensores del robot. El método no fue integrado en el robot debido a restricciones técnicas (implementación del método en C++ e integración a los otros módulos de software). Para esta fase se consideró un conjunto más amplio de ademanes orientados a comandar un robot de servicio, además se cambiaron las características a extraer del esqueleto aportado por el Kinect y se aplicó votación pesada.

5.5.1. Datos utilizados

Para esta fase de los experimentos se realizaron nuevas capturas, usando el Kinect superior del robot de servicio Sabina y en un entorno simulado de casa, donde se entrena a dicho robot, tal como se puede apreciar en la Figura 5.7.

Además, se definió un nuevo conjunto de ademanes, tal como se aprecia en la Figura 5.8. Este conjunto se diseñó luego de una revisión del estado del arte sobre ademanes para comandar robots de servicio y de acuerdo a las instrucciones que se necesitan dar a Sabina para su desenvolvimiento en las competencias nacionales e internacionales de Robocup @Home (Chen et al., 2013) que se realizan anualmente.

Como se puede observar, todos los ademanes pueden ser ejecutados con una sola mano, en este caso se realizaron con la mano derecha. Por lo tanto, las características para los modelos HMM contempladas en estos experimentos cambian. Esto no sólo por considerarse una sola mano, sino también porque se desea experimentar con información de posición en relación a una parte fija del cuerpo y poder verificar si existe mejora en la clasificación. Por lo tanto, las características usadas fueron la diferencia cinética y de posición, tal como se especificó en la sección 4.4.2.

Los ademanes fueron capturados por 6 usuarios, todos ellos estudiantes pertenecientes al laboratorio de robótica y diestros. Se les mostró a cada usuario dos repeticiones de video de cada ademan para que tuviesen conocimiento de cómo

Figura 5.7: Robot Sabina del Laboratorio de Robótica del INAOE. Con el Kinect ubicado en la parte más alta de Sabina, se capturan los ademanes hechos por el usuario.

ejecutarlos. Ahora bien, debido a la falta de disponibilidad total de los usuarios utilizados para las capturas de estos experimentos, el número de repeticiones que se dispone no son iguales para todos los ademanes, tanto para los datos de entrenamiento como para los de prueba. En la Tabla 5.8 se muestra el número de repeticiones para los datos de entrenamiento para cada ademán.

Tabla 5.8: Número de repeticiones capturadas por ademán para entrenamiento entre los 6 usuarios.

Ademán		**Repeticiones por usuario**						**Total**
		1	2	3	4	5	6	
1	Atención	20	10	10	10	10	10	**70**
2	Alto	20	10	10	10	10	10	**70**
3	Apuntar	20	10	10	10	10	10	**70**
4	Acercarse	20	10	10	10	10	10	**70**
5	Alejarse	20	0	10	10	10	10	**60**

Se puede apreciar que: se utilizaron 6 usuarios distintos para estas capturas, con el usuario 1 se pudo capturar 20 repeticiones de todos los ademanes y con el usuario 2 no se pudo capturar repeticiones del ademán 5. Sin embargo, los datos son suficientes para evaluar la efectividad del método propuesto.

En relación a los datos de prueba, el número de ejemplos por clase de ademán se muestran en la Tabla 5.9.

(a) Atencion (b) Alto

(c) Apuntar (d) Acercarse

(e) Alejarse

Figura 5.8: Conjunto de ademanes considerados para esta fase de los experimentos. Sólo se muestra en la imagen el punto final de cada ademán, y una trayectoria aproximada (flecha amarilla).

Para los datos de prueba, el número de ejemplos no es igual para todos los ademanes y además, la duración promedio de los ademanes fue de 32 cuadros. Como para estos experimentos se quiere utilizar votaciones pesadas, entonces se debe calcular los pesos que se usarán de acuerdo al tamaño de la ventana y las probabilidades posteriores de cada modelo. Para el peso por tamaño se obtuvieron estadísticas del tiempo de ejecución (tamaño de la ventana), generándose la gráfica que se ilustra en la Figura 5.9.

De acuerdo a dicha gráfica se asigna los pesos a las diferentes ventanas, de forma de dar mayor peso a ventanas de tamaños más probables y menos peso a ventanas de tamaños menos probables, los pesos se muestran en la Tabla 5.10

Con estos datos se realizó el experimento 2 cuyos resultados se describen a continuación.

Tabla 5.9: Número de ejemplos de prueba por ademán.

Ademán		Total
1	Atención	10
2	Alto	14
3	Apuntar	16
4	Acercarse	10
5	Alejarse	8

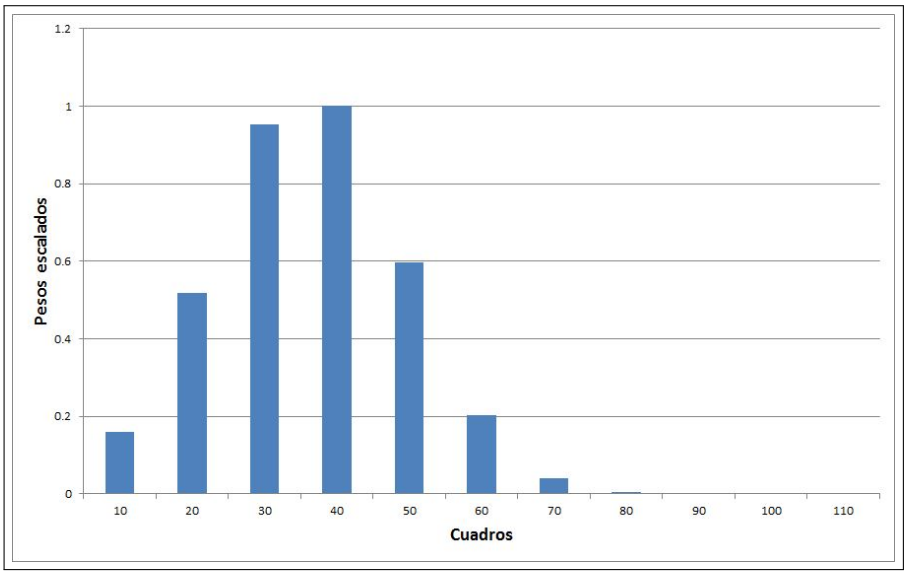


Figura 5.9: Gráfica de pesos por duración de los ademanes

5.5.2. Resultados para segmentación manual

Tal como en los experimentos anteriores, aquí se aplicó *LOOCV* a los datos de entrenamiento, para comprobar la validez de los modelos HMM. Dado que se tienen 5 ademanes, entonces se entrenaron 5 modelos con la misma configuración tratada hasta ahora. La matriz de confusión resultante de esto se muestra en la Tabla 5.11.

Con lo anterior se verifica que la tasa de reconocimiento de los HMMs para estos ademanes es de 88.53 %. Aunque es una tasa menor a los de los experimentos anteriores, se puede decir que es aceptable considerando el número de ademanes y el poco número de repeticiones disponibles.

De igual modo, se aplicó *LOOCV* a los datos de prueba con segmentación manual, usando los modelos entrenados con los datos de entrenamiento antes descritos. La matriz de confusión obtenida se presenta en la Tabla 5.12.

Con lo que se obtuvo 82.76 % de reconocimiento. Este valor, de igual modo como en el caso anterior, será nuestro valor de referencia (cota superior) para

Tabla 5.10: Pesos a usarse para ponderar los votos de acuerdo a la longitud de la ventana en cuestión.

Tamaño	Pesos
10	0.160123651
20	0.517871335
30	0.953681544
40	1
50	0.597051424
60	0.202973151
70	0.039289867
80	0.004330496
90	0.000271775
100	9.71174E-06
110	1.97606E-07

Tabla 5.11: Matriz de confusión de las pruebas de *LOOCV* con cinco ademanes con segmentación manual, para los datos de entrenamiento.

Matriz de Confusión					
Ademán	1	2	3	4	5
Atención	70	0	0	0	0
Alto	1	55	1	8	5
Apuntar	0	0	70	0	0
Acercarse	1	0	0	65	4
Alejarse	7	4	0	8	41

evaluar el método de segmentación y reconocimiento automático.

5.5.3. Resultados para segmentación y reconocimiento simultáneo

En forma análoga a los experimentos anteriores, se evalúa la cobertura y precisión del método para diferentes valores de los parámetros. Además, se consideraron tres estrategias distintas de pesado, a saber:

1. Pesado por tamaño de la ventana. De acuerdo al tamaño de la ventana en el instante t_i, se acumula el peso correspondiente para el ademán ganador en ese instante t_i (según Tabla 5.10). Así por ejemplo, una ventana en t_1 con tamaño de 10 cuadros, aportará 0,160123651 al ademán que resulte reconocido. Luego en t_2, esta misma ventana tendrá un tamaño de 20 cuadros, por lo que aportará 0,517871335 al ademán ganador en ese instante, que incluso podría ser el mismo del instante anterior.

2. Pesado por probabilidad del ademán. Se toma la mayor probabilidad pos-

Tabla 5.12: Matriz de confusión para las pruebas de *LOOCV* con cinco ademanes con segmentación manual para los datos de prueba.

Matriz de Confusión					
Ademán	1	2	3	4	5
Atención	8	0	0	2	0
Alto	4	9	0	1	0
Apuntar	0	0	14	0	2
Acercarse	0	0	0	10	0
Alejarse	1	0	0	0	7

terior de los modelos para cada ventana, y ese valor de probabilidad se va acumulando para ese ademán en cada instante t_i. Usando el mismo ejemplo del caso anterior, nuestra ventana aportará en cada t_i al ademán reconocido, el valor de probabilidad que arrojó el modelo j HMM que lo dio como ganador ($P(O \mid \lambda_j)$). Por lo tanto, aquí no se usa la Tabla 5.10.

3. Pesado combinado. Se multiplica el peso correspondiente al tamaño de la ventana en el instante t_i, (según Tabla 5.10) por el valor del modelo de mayor probabilidad de dicha ventana. Ese valor se va acumulando para cada ademán en cada instante t_i. De igual modo que en los casos anteriores, en el instante t_1, para un ademán j que resultó reconocido por el modelo HMM λ_j, se acumula $0{,}160123651 \times P(O \mid \lambda_j)$.

Por lo tanto, se tiene tres estrategias de pesado distintas, y para cada una, se debe variar los valores de los parámetros tal como en los experimentos anteriores.

Por otra parte, de acuerdo a la experiencia alcanzada de los experimentos iniciales, se decidió cambiar el conjunto de valores de los parámetros, con lo que se tiene: $FI = 20, 30, 40$, $FP = 50, 60, 70, 80$ y $Holgura = 5, 10, 15$. Se puede apreciar entonces que se incluye un valor más en FP para verificar si con esto mejora la segmentación y además se elimina el valor de 20 en la holgura, dado que es un valor muy amplio para la precisión de segmentación que se desea. Adicionalmente, es importante acotar que los ejemplos de prueba aquí utilizados solo tienen un ademán en cada uno, rodeados por zonas de no-ademanes a cada lado. Pruebas con secuencias con varios ademanes se muestra en la sección 5.6.

Por lo tanto, a continuación se muestra las mismas tres tablas anteriores (una por cada punto de segmentación) para cada estrategia de pesado antes descrito.

Estrategia de pesado 1:

a. Segmentación un punto atrás: Se muestra los resultados en la Tabla 5.13

Tabla 5.13: Resultados del método de segmentación y reconocimiento automático, segmentando un paso atrás y pesado 1. En negrilla se muestran los mejores resultados.

FI	**FP**	**Holgura**					
		5		10		15	
		Cober.	Prec.	Cober.	Prec.	Cober.	Prec.
20	50	20.69	**91.67**	39.66	**86.96**	51.72	83.33
	60	25.86	86.67	46.55	81.48	55.17	84.38
	70	25.86	80.00	46.55	77.78	56.90	81.82
	80	32.76	84.21	46.55	77.78	55.17	81.25
30	50	29.31	88.24	50.00	82.76	67.24	82.05
	60	32.76	89.47	60.35	85.71	75.86	88.64
	70	36.21	85.71	60.35	82.86	75.86	86.36
	80	**37.93**	86.36	58.62	82.35	72.41	85.71
40	50	29.31	88.24	51.72	83.33	70.69	82.93
	60	36.21	90.48	**63.79**	86.49	**79.31**	**89.13**
	70	37.93	86.36	62.07	83.33	79.31	86.96
	80	37.93	86.36	58.62	82.35	75.86	86.36

Se repite el comportamiento anterior, a mayor holgura, mayor cobertura pero menor precisión. Los mejores resultados están en la banda de FI = 40 y FP 60.

b. Segmentación en el punto: Se muestra los resultados en la Tabla 5.14

Tabla 5.14: Resultados del método de segmentación y reconocimiento automático, segmentando en el punto y pesado 1. En negrilla se muestran los mejores resultados.

FI	**FP**	**Holgura**					
		5		10		15	
		Cober.	Prec.	Cober.	Prec.	Cober.	Prec.
20	50	20.69	**91.67**	43.10	80.00	55.17	78.13
	60	27.59	87.50	53.45	74.19	58.62	76.47
	70	25.86	80.00	51.72	70.00	60.35	74.29
	80	31.03	83.33	50.00	68.97	58.62	70.59
30	50	29.31	88.24	48.28	82.14	65.52	81.58
	60	34.48	90.00	62.07	83.33	74.14	86.05
	70	36.21	85.71	58.62	79.41	72.41	83.33
	80	36.21	85.71	55.17	78.13	68.97	80.00
40	50	29.31	88.24	50.00	82.76	68.97	82.50
	60	**37.93**	90.91	**65.52**	**84.21**	**77.59**	**86.67**
	70	37.93	86.36	60.35	80.00	75.86	84.09
	80	36.21	85.71	55.17	78.13	72.41	80.95

Los resultados no mostraron una mejora significativa y de nuevo los buenos resultados están en la banda de FI = 40 y FP 60.

c. Segmentación un punto adelante: Se muestra los resultados en la Tabla 5.15

De nuevo, no hubo una mejora significativa para ambas medidas y la mayoría de los buenos resultados están en la banda de FI = 40 y FP 60.

Para esta estrategia de pesado, el punto de segmentación no influye de manera

Tabla 5.15: Resultados del método de segmentación y reconocimiento automático, segmentando un paso adelante y pesado 1. En negrilla se muestran los mejores resultados.

FI	**FP**	**Holgura**					
		5		10		15	
		Cober.	Prec.	Cober.	Prec.	Cober.	Prec.
20	50	27.59	**93.75**	44.83	**92.31**	65.52	84.21
	60	34.48	85.00	55.17	84.38	68.97	82.50
	70	32.76	78.95	53.45	80.65	68.97	80.00
	80	37.93	81.82	51.72	80.00	65.52	78.95
30	50	31.03	88.89	50.00	86.21	70.69	82.93
	60	36.21	85.71	63.79	86.49	79.31	86.96
	70	37.93	81.82	60.35	82.86	75.86	84.09
	80	37.93	81.82	56.90	81.82	70.69	82.93
40	50	31.03	88.89	50.00	86.21	70.69	82.93
	60	**39.66**	86.96	**65.52**	86.84	**79.31**	**86.96**
	70	39.66	82.61	60.35	82.86	75.86	84.09
	80	37.93	81.82	55.17	81.25	70.69	82.93

significativa en los resultados. Sin embargo, se puede observar que en todos los casos los mejores resultados son para FI = 40 y FP 60. Verifiquemos ahora la siguiente estrategia de pesado, usando las probabilidades posteriores de los modelos HMMs como peso de los votos.

Estrategia de pesado 2:

a. Segmentación un punto atrás: Se muestra los resultados en la Tabla 5.16

Tabla 5.16: Resultados del método de segmentación y reconocimiento automático, segmentando un paso atrás y pesado 2. En negrilla se muestran los mejores resultados.

FI	**FP**	**Holgura**					
		5		10		15	
		Cober.	Prec.	Cober.	Prec.	Cober.	Prec.
20	50	20.69	91.67	39.66	82.61	51.72	80.00
	60	27.59	**93.75**	48.28	85.71	56.90	87.88
	70	27.59	87.50	50.00	82.76	60.35	85.71
	80	34.48	90.00	50.00	82.76	58.62	85.29
30	50	29.31	88.24	50.00	79.31	67.24	79.49
	60	34.48	90.00	62.07	86.11	77.59	88.89
	70	37.93	90.91	63.79	86.49	79.31	89.13
	80	**39.66**	91.30	62.07	86.11	75.86	88.64
40	50	29.31	88.24	51.72	80.00	70.69	80.49
	60	37.93	90.91	**65.52**	**86.84**	81.03	89.36
	70	39.66	91.30	65.52	86.84	**82.76**	**89.58**
	80	39.66	91.30	62.07	86.11	79.31	89.13

b. Segmentación en el punto: Se muestra los resultados en la Tabla 5.17

Tabla 5.17: Resultados del método de segmentación y reconocimiento automático, segmentando en el punto y pesado 2. En negrilla se muestran los mejores resultados.

FI	**FP**	**Holgura**					
		5		10		15	
		Cober.	Prec.	Cober.	Prec.	Cober.	Prec.
20	50	20.69	91.67	43.10	76.00	55.17	75.00
	60	27.59	**93.75**	53.45	77.42	58.62	79.41
	70	25.86	86.67	53.45	74.19	62.07	77.78
	80	31.03	88.89	51.72	73.33	60.35	77.14
30	50	29.31	88.24	48.28	78.57	65.52	78.95
	60	34.48	90.00	62.07	83.33	74.14	86.05
	70	36.21	90.48	60.35	82.86	74.14	86.05
	80	36.21	90.48	56.90	81.82	70.69	85.37
40	50	29.31	88.24	50.00	79.31	68.97	80.00
	60	**37.93**	90.91	**65.52**	**84.21**	**77.59**	**86.67**
	70	37.93	90.91	62.07	83.33	77.59	86.67
	80	36.21	90.48	56.90	81.82	74.14	86.05

c. Segmentación un punto adelante: Se muestra los resultados en la Tabla 5.18

Tabla 5.18: Resultados del método de segmentación y reconocimiento automático, segmentando un paso adelante y pesado 2. En negrilla se muestran los mejores resultados.

FI	**FP**	**Holgura**					
		5		10		15	
		Cober.	Prec.	Cober.	Prec.	Cober.	Prec.
20	50	27.59	**93.75**	44.83	**88.46**	65.52	81.58
	60	34.48	90.00	53.45	87.10	67.24	84.62
	70	32.76	84.21	53.45	83.87	68.97	82.50
	80	37.93	86.36	51.72	83.33	65.52	81.58
30	50	31.03	88.89	50.00	82.76	70.69	80.49
	60	36.21	85.71	62.07	86.11	77.59	86.67
	70	37.93	86.36	60.35	85.71	75.86	86.36
	80	37.93	86.36	56.90	84.85	70.69	85.37
40	50	31.03	88.89	50.00	82.76	70.69	80.49
	60	**39.66**	86.96	**63.79**	86.49	**77.59**	**86.67**
	70	39.66	86.96	60.35	85.71	75.86	86.36
	80	37.93	86.36	55.17	84.38	70.69	85.37

Nuevamente en esta estrategia de pesado, se nota que los mejores resultados se concentran en la banda de FI = 40 y FP = 60. El punto de segmentación no altera mucho los resultados.

Estrategia de pesado 3:

a. Segmentación un punto atrás: Se muestra los resultados en la Tabla 5.19

Tabla 5.19: Resultados del método de segmentación y reconocimiento automático, segmentando un paso atrás y pesado 3. En negrilla se muestran los mejores resultados.

FI	**FP**	**Holgura**					
		5		10		15	
		Cober.	Prec.	Cober.	Prec.	Cober.	Prec.
20	50	24.14	85.71	39.66	78.26	51.72	76.67
	60	29.31	**88.24**	46.55	81.48	55.17	84.38
	70	29.31	82.35	46.55	77.78	56.90	81.82
	80	36.21	85.71	46.55	77.78	55.17	81.25
30	50	32.76	84.21	50.00	75.86	67.24	76.92
	60	36.21	85.71	60.35	82.86	75.86	86.36
	70	39.66	86.96	60.35	82.86	75.86	86.36
	80	41.38	87.5	0 58.62	82.35	72.41	85.71
40	50	32.76	84.21	53.45	77.42	70.69	75.61
	60	39.66	86.96	**65.52**	**84.21**	**79.31**	**86.96**
	70	**41.38**	87.50	63.79	83.78	79.31	86.96
	80	41.38	87.50	60.35	82.86	75.86	86.36

b. Segmentación en el punto: Se muestra los resultados en la Tabla 5.20

Tabla 5.20: Resultados del método de segmentación y reconocimiento automático, segmentando en el punto y pesado 3. En negrilla se muestran los mejores resultados.

FI	**FP**	**Holgura**					
		5		10		15	
		Cober.	Prec.	Cober.	Prec.	Cober.	Prec.
20	50	22.41	84.62	43.10	72.00	55.17	71.88
	60	29.31	**88.24**	53.45	74.19	60.35	77.14
	70	27.59	81.25	51.72	70.00	62.07	75.00
	80	32.76	84.21	50.00	68.97	60.35	71.43
30	50	31.03	83.33	48.28	75.00	65.52	76.32
	60	36.21	85.71	62.07	80.56	75.86	84.09
	70	37.93	86.36	58.62	79.41	74.14	83.72
	80	37.93	86.36	55.17	78.13	70.69	80.49
40	50	29.31	82.35	50.00	75.86	68.97	75.00
	60	**37.93**	86.36	**65.52**	**81.58**	**79.31**	**84.78**
	70	37.93	86.36	60.35	80.00	77.59	84.44
	80	36.21	85.71	55.17	78.13	74.14	81.40

c. Segmentación un punto adelante: Se muestra los resultados en la Tabla 5.21

Igual como en las otras estrategias, los mejores resultados se concentran en la banda de FI = 40 y FP = 60 y el punto de segmentación no altera mucho los resultados.

5.5.4. Conclusiones

De los experimentos anteriores podemos concluir lo siguiente:

Tabla 5.21: Resultados del método de segmentación y reconocimiento automático, segmentando un paso adelante y pesado 3. En negrilla se muestran los mejores resultados.

FI	**FP**	**Holgura**					
		5		10		15	
		Cober.	Prec.	Cober.	Prec.	Cober.	Prec.
20	50	27.59	**87.50**	43.10	**84.00**	62.07	75.00
	60	34.48	85.00	53.45	83.87	67.24	79.49
	70	32.76	78.95	51.72	80.00	67.24	76.92
	80	37.93	81.82	50.00	79.31	63.79	75.68
30	50	31.03	83.33	48.28	78.57	67.24	74.36
	60	36.21	80.95	62.07	83.33	77.59	82.22
	70	37.93	81.82	58.62	82.35	74.14	81.40
	80	37.93	81.82	55.17	81.25	68.97	80.00
40	50	31.03	83.33	48.28	78.57	67.24	74.36
	60	**39.66**	82.61	**63.79**	83.78	**77.59**	**84.44**
	70	39.66	82.61	58.62	82.35	74.14	83.72
	80	37.93	81.82	53.45	80.65	68.97	82.50

1. Los resultados obtenidos permiten asegurar que el método propuesto es efectivo para realizar la segmentación y reconocimiento simultáneo de ademanes, sin la necesidad de un modelo de no-ademán y poses de marcas de inicio y fin de ademanes.

2. Los mejores resultados se obtienen, en general, al considerar FI=40, FP=60 y sin importar el punto de segmentación. Esto corresponde a los promedios de duración de los ademanes mostrados en la Figura 5.9.

3. Las estrategias de pesado y el punto de segmentación no aporta significativas diferencias en los resultados obtenidos. Finalmente, las tres estrategias de pesado otorgan relevancia a las ventanas que mejor se ajustan al intervalo del ademán, que es lo que se pretendía.

4. Al igual que en los experimentos anteriores, sigue habiendo un compromiso entre cobertura y precisión.

5. La estrategia de pesado y el conjunto de características considerados, mejoraron los resultados significativamente.

5.5.5. Análisis

Dado que lo que se desea es instruir al robot de servicio con el conjunto de los cinco ademanes descrito en los segundos experimentos, los resultados de estos son los que mayor atención ameritan. Se requiere un compromiso entre cobertura y precisión, por lo que se necesita medir esto para poder encontrar la combinación de parámetros más idónea a usar en nuestro método de segmentación y reconocimiento simultáneo a implementar en Sabina. Por lo tanto, se muestran gráficas de

la medida-f aplicado a todos los experimentos realizados para cada estrategia de pesado. En total, por cada estrategia, se tienen 108 experimentos al ir combinando los valores considerados de los parámetros y la holgura. Es por esta razón, que las gráficas de la Figura 5.10 poseen en su abscisa, un índice por cada combinación de los mismos. Para una descripción detallada del conjunto de parámetros para cada experimento, véase la tabla en el Anexo A. Además, en el Anexo B se muestra esta misma gráfica pero discriminada para cada valor de holgura.

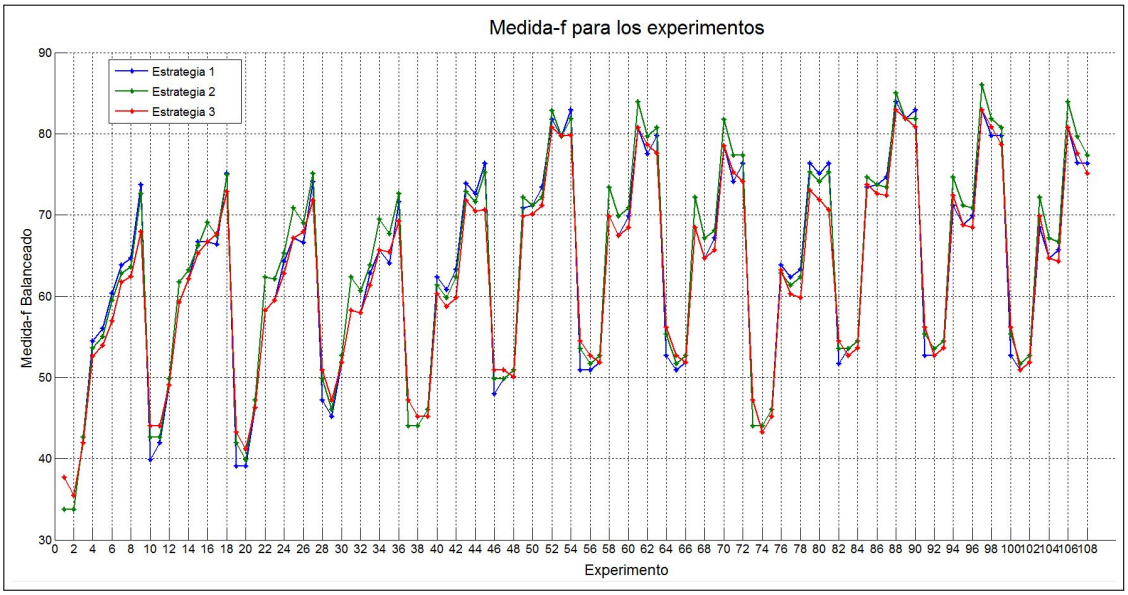


Figura 5.10: Valor de medida-f para las tres estrategias de pesado, para cada una de los 108 experimentos donde se combinan los distintos valores de los parámetros FI, FP, punto de segmentación y se va cambiando la holgura

De los experimentos, se obtuvo que la mayor cobertura corresponde al experimento 97 con 82.76 % y el resultado con la mayor precisión corresponde al experimento 3 con 93.75 %. Sin embargo, este último tiene una cobertura de 27.58 %.

Coincidencialmente, de los resultados de la Figura 5.10, se puede ver que el mejor valor de la medida-f se alcanza en el experimento 97 (mejor cobertura). Los parámetros y valores respectivos son FI = 40, FP = 70, Holgura = 15 y Tipo Segmentación = punto atrás y estrategia de pesado 2 (Pesado basado en la probabilidad del ademán), obteniéndose los siguientes resultados:

Cobertura = 82.76 % y Precisión = 89.58 %.

Otras combinaciones de parámetros que dan buenos resultados son las siguientes:

1. FI = 40, FP = 70, Holgura = 15, Tipo Segmentación = punto atrás y estrategia 1(Pesado basado en tamaño del ademán) y 3 (pesado con probabilidad*pesos): Cobertura = 79.31 % y Precisión = 86.96 %.

2. FI = 40, FP = 60, Holgura = 15, Tipo Segmentación = punto atrás y estrategia 2(Pesado basado en la probabilidad del ademán): Cobertura = 81.03 % y Precisión = 89.36 %.

3. FI = 40, FP = 80, Holgura = 15, Tipo Segmentación = punto atrás y estrategia 2(Pesado basado en la probabilidad del ademán): Cobertura = 79.31 % y Precisión = 89.13 %.

En el caso de una holgura muy cerrada como de 5 cuadros, los mejores resultados obtenidos fueron de 41.38 % para cobertura y 87.50 % de precisión, correspondiente a FI = 40, FP = 70, segmentando un punto atrás y estrategia de pesado 3 (combinación de la 1 y 2).

Por lo tanto, para un holgura de 15 cuadros, sorprendentemente la precisión de este experimento (89.58 %) supera a la de la segmentación manual (82.76 %) en 6.82 %. Dado que la medida de precisión obtenida en estos experimentos es en base a los ademanes bien segmentados, es posible que ocurra este fenómeno. Por otra parte, la tasa de cobertura es exactamente igual a la precisión en la segmentación manual, lo cual también se considera una coincidencia. Como ya se mencionó, una holgura de esta dimensión no afecta el uso de los ademanes para comandar un robot, dado que se trata de medio segundo antes o después del punto final real del ademán.

Consideramos que estos resultados son prometedores para llevarlos a la práctica en el comando de robots de servicio por ademanes, proveyendo de una precisión cercana al 90 % con una cobertura de más de 80 %. Estas medidas permite al robot reconocer con mas certeza cuándo (cobertura) y cuál (precisión) ademan esta ejecutando el usuario.

5.6. Experimentos 3: Experimentos en secuencias con varios ademanes

Para estos experimentos se tomaron los 58 ademanes segmentados de los experimentos 2, donde cada ejemplo tenía un solo ademan y zonas de no ademan a ambos lados; y se juntaron por usuario. Por lo tanto, se crearon secuencias por usuario conteniendo el numero de ademanes mostrado en la Tabla 5.22.

Con lo anterior, se aplicó el método de segmentación y reconocimiento automático aquí propuesto, con los siguientes valores de parámetros:

1. Holgura = 5, 10, 15

2. FI = 20, 30, 40

3. FP = 50, 60, 70, 80

Tabla 5.22: Número de ademanes por usuario. Cada secuencia representa un usuario.

Usuario	Núm. de ademanes
1	17
2	8
3	8
4	16
5	9

4. Tipo de segmentación: Punto atrás y en el punto

Obsérvese que no se segmento un punto adelante porque no es fácil de implementar para este tipo de experimento y además que las simulaciones de los experimentos 1 y 2 muestran que este tipo de segmentación arroja los peores resultados. De igual modo que las simulaciones del experimento 2, se usó las tres estrategias de pesado (solo pesos, solo probabilidades y pesos×probabildiades).

Con todo lo anterior, al igual que como se mostró en la sección 5.5.5, se gráfica la medida-f para cada estrategia de pesado, combinando cada parámetro y la holgura, obteniéndose 72 experimentos y los resultados mostrados en la Figura 5.11.

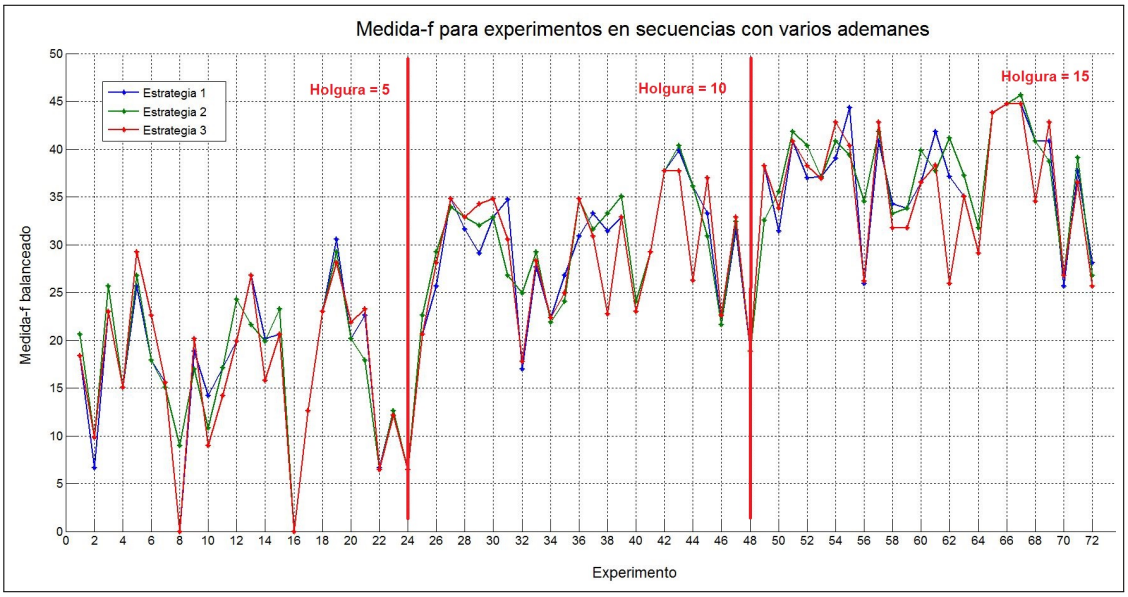


Figura 5.11: Valor de medida-f para las tres estrategias de pesado, para cada una de los 72 experimentos donde se combinan los distintos valores de los parámetros y la holgura, aplicado a las 5 secuencias con varios ademanes. Los primeros 24 experimentos poseen una holgura de 5, los experimentos del 25 al 48 una holgura de 10 y los restantes una holgura de 15.

De los resultados anteriores se observa que el mejor resultado es la simulación 67 (Holgura = 15). Los parámetros de este experimento son FI = 40, FP = 60 y tipo segmentación = punto atrás, usando la estrategia de pesado 2, es decir, solo con el valor de probabilidad. El valor de cobertura y precisión fue de 36.21 % y 61.90 % respectivamente.

El mejor resultado para las holguras de 5, es para el experimento 19, con valores de parámetros de FI = 40, FP = 60 y tipo de segmentación = punto atrás. Aquí se obtiene para cobertura 20.69 % y precisión 58.33 %. La estrategia de pesado de este experimento es el 1, es decir, solo pesos. Sin embargo, la estrategia 2 de este mismos experimento 19, arroja 18.97 % y 63.64 % para cobertura y precisión respectivamente, la cual es preferible dado que no se necesita calcular pesos y tiene mejor precisión y no es mucho la diferencia en cobertura con respecto al anterior.

5.7. Resumen

En esta capítulo se mostraron los resultados de los experimentos realizados para verificar la validez del método propuesto para segmentar y reconocer ademanes simultáneamente. Se determinó de forma empírica los mejores valores para los parámetros de nuestro método. Para esto, se realizaron tres tipos de experimentos. Unos iniciales con pocos ademanes, esto con el fin de determinar que si el método funciona bajo este escenario, entonces podría aplicarse a escenarios mas complejos. Es precisamente esto último el segundo tipo de experimentos realizados, donde se amplió el conjunto de ademanes a cinco, mas complejos y usando el robot de servicio del laboratorio de robotica del INAOE, denominada Sabina. Sin embargo, las secuencias de estos segundos experimentos sólo contenían un ademan cada uno. Por esa razón, se llevó a cabo el tercer grupo de experimentos donde se agregó mas ademanes por secuencia e incluso cada secuencia corresponde a un usuario distinto.

A pesar de no haber obtenido tan buenos resultados en el primer grupo de experimentos, en el segundo se obtuvieron resultados suficientemente aceptables para la aplicación aquí considerada (comandar robots con ademanes). En estos se obtuvo un 82.76 % de cobertura y un 89.58 % de precisión. Ya con secuencias mas completas estos valores descendieron a 36.21 % para cobertura y 61.90 % para precisión.

De los resultados obtenidos se puede desprender lo siguiente:

1. Los mejores valores para Δ es 10 y para el umbral δ es 100. En relación al tamaño de crecimiento de las ventanas Δ, este valor se fijó de acuerdo a lo obtenido en las gráficas de la Figura 5.3. Para el caso del umbral δ, se mostró en la Tabla 5.3 que se obtiene los mismos o peores resultados con otros valores.

2. El mejor punto para segmentar es un paso atrás. Esto verifica lo que se presumía, dado que en dicho punto aún se tenía la decisión unánime de las ventanas.

3. En términos generales, los mejores valores para FI y FP son muy cercanos en todos los experimentos, $(30, 60)$ para (FI, FP) en el experimento 1, $(40, 70)$ en el experimento 2 y $(40, 60)$ en el 3. Estos valores están muy relacionados con los promedios de duración en cuadros de los ademanes en cada caso.

4. El segundo conjunto de características (movimiento y posición) aportan mejores resultados. La información adicional de posición aporta facilidad de discriminación entre los ademanes a los modelos HMMs.

5. Es mejor usar votación pesada que simple. Con el voto pesado se da mayor valor a los votos de las ventanas que mejor se ajustan al intervalo del ademán, las cuales deberían tener una decisión más certera del tipo de ademán que se ejecutó.

6. De las tres estrategias de pesado consideradas, el correspondiente a la probabilidad posterior de los modelos HMMs, es la que en términos generales aportó mejores resultados. Esto simplifica el entonado de nuestro método dado que ya no sería necesario estimar pesos de acuerdo a la duración de los ademanes.

Capítulo 6

Conclusiones y Trabajo Futuro

Encontrar de forma automática el punto de inicio y fin de un ademán, acción conocida como segmentación, es un problema abierto. No se tiene aún una propuesta suficientemente confiable que otorgue dichos puntos lo más exacto posible. Adicionalmente, una vez se logre dicha segmentación, se desea conocer cuál fue el ademán que se ejecutó, acción conocida como reconocimiento. Por lo tanto, se desea cumplir con ambas acciones (segmentación y reconocimiento) a la vez. Así pues, existen pocos trabajos de investigación que se han avocado a lograr la segmentación y reconocimiento simultáneo de un ademán, entre una secuencia de video, donde la persona está realizando muchos movimientos. Estos trabajos en general sufren de dos limitaciones importantes:

- Requieren de una posición preestablecida para el inicio y fin de cada ademán, lo cual no es natural.
- Requieren de un modelo de no-ademán, que es difícil de construir.

Por otra parte, una de las maneras más naturales de comunicación entre la persona y un robot destinado a apoyarlo en las actividades de la casa (robot de servicio) es por medio de ademanes. Esto toma mayor importancia si la persona sufre de alguna discapacidad del habla.

Es por las razones antes expuestas que en el presente trabajo se desarrolló un método de segmentación y reconocimiento simultáneo de ademanes para comandar un robot de servicio. El método propuesto no requiere de posiciones fijas de inicio y fin del ademán, ni de un modelo de no-ademán. Las simulaciones realizadas presentan resultados aceptables que dan indicio que puede funcionar suficientemente bien a la hora de integrarlo en algún robot de servicio, en específico, en el robot del equipo de Markovito, del INAOE.

A continuación se resume el método propuesto, se listan las principales conclusiones y se dan algunas ideas para los siguientes pasos en esta investigación.

6.1. Resumen

El método de segmentación y reconocimiento simultáneo de ademanes aquí propuesto esta basado en una técnica de exploración de la secuencia de video denominada múltiples ventanas de tamaño dinámico. Con esto, se tienen varios segmentos cubriendo distintas zonas de la secuencia. Cada segmento (ventana) aporta una decisión sobre cuál ademán considera que se ejecuta en su dominio. Tomando las decisiones (votos) de todas las ventanas es posible determinar el punto final del ademán, que corresponde al momento cuando los votos mayoritarios de estas para un ademán particular, cae debajo de un umbral.

Una vez determinado el punto más cercano al final del ademán, se puede arrojar la decisión sobre cual ademán se ejecutó, con base en los votos mayoritarios que se tenían de las ventanas. Con todo lo anterior, se realizaron simulaciones que dieron buenos resultados, con tasas superiores al 80 % tanto para segmentación como para reconocimiento.

6.2. Conclusiones

El método de segmentación y reconocimiento simultáneo de ademanes para comandar un robot de servicio propuesto en este trabajo de investigación, nos permite llegar a las siguientes conclusiones:

1. La técnica de exploración de la secuencia de video propuesta aquí, denominada múltiples ventanas de tamaño dinámico (MVTD), contribuyó significativamente en el diseño final del método de segmentación y reconocimiento simultáneo de ademanes. Sin esta técnica, no se dispondría de la información suficiente para lograr el cometido fijado en esta investigación.

2. A diferencia de otras propuestas, en este método no se necesita ni un modelo HMM de no ademán ni una postura de ruptura por parte del usuario. Es decir, en ningún momento se necesitó un modelo HMM para movimientos distintos a los ademanes contemplados. Tampoco se le exigió al usuario que tomara una postura determinado para poder identificar cuando termina de ejecutar un ademán y empieza uno nuevo.

3. Los votos pesados y las características cinemáticas combinadas con posición relativa, aportaron incremento en las medidas de calidad usadas para segmentación y reconocimiento.

4. Como era de esperarse, el mejor punto a considerar como el final del ademán, es el anterior al momento en que los votos de las ventanas dejan de ser unánimes. Esto porque era el momento justo en que aún se estaba ejecutando

dicho ademán y es donde se tiene la mejor información para decidir cuál ademán acaba de terminar de ejecutarse.

5. Con el método propuesto se logró alcanzar una tasa de 82.76 % de cobertura y 89.58 % de precisión con una holgura de 15 cuadros, lo cual es suficiente comparada al 82.76 % que se puede alcanzar si se hiciese una segmentación manual. Con una holgura de 5 cuadros los mejores resultados obtenidos fueron de 41.38 % para cobertura y 87.50 % de precisión.

6. Los parámetros del método propuesto son poco sensibles a cambios en sus valores. Esto facilita la fijación de sus valores para mejorar la segmentación y reconocimiento.

7. Se creó una base de datos de ademanes ideal para ser usada por toda la comunidad científica que desee avocarse al estudio de la problemática aquí tratada.

8. Los resultados obtenidos en secuencias completas se redujeron drásticamente para cobertura y precisión. Es necesario realizar cambios al método propuesto para mejorar estos valores.

9. Aun falta por probar el método aquí propuesto con un conjunto de ademanes más amplio, especialmente con ademanes similares.

10. Aunque los ademanes son un medio de comunicación de uso muy común, existen otros más cotidianos y expresivos, como por ejemplo la voz.

11. Se generaron dos publicaciones en congresos internacionales: uno aceptado en el *Workshop on Ubiquitous Data Mining (UDM)* de la *23rd. International Joint Conference on Artificial Intelligence* (IJCAI 2013), realizado en Beijing, China; y otro en revisión en el *16th International Conference on Advanced Robotics* (ICAR 2013), a realizarse en Montevideo, Uruguay.

En general, los experimentos realizados arrojaron resultados que evidenciaron coherencia en las hipótesis asumidas al momento de proponer el método de segmentación y reconocimiento simultáneo propuesto. Un ejemplo de esto, fue la presunción de que segmentando un paso atrás debería dar mejores resultados y efectivamente en todos los experimentos, se comprobó dicha asunción.

6.3. Trabajo Futuro

A pesar de los buenos resultados alcanzados en esta propuesta, se idearon modificaciones y adiciones que podrían mejorar lo antes descrito. Por lo tanto, se proponen los siguientes trabajos futuros:

1. Cambiar la configuración de los HMMs aquí tratados, incluso se puede tener configuraciones distintas para cada ademán. Esto porque según algunos resultados de otros trabajos, puede ser beneficioso este esquema para mejorar la tasa de reconocimiento.

2. Experimentar con variaciones en las características extraídas del Kinect, tal como vectores de dirección entre codo y mano, matrices de rotación de los codos, ángulos formados entre brazo y tronco, forma de la mano, etc. Esto para obtener mas precisión de los movimientos de las manos y lograr una mejor distinción entre los ademanes, mas aún entre aquellos muy parecidos.

3. Probar el método de segmentación y reconocimiento aquí propuesto con otro reconocedor, como por ejemplo, redes bayesianas dinámicas o redes neuronales temporales; y así poder verificar la extensibilidad del método con cualquier otro reconocedor y verificar la mejora o no de la tasa de reconocimiento.

4. Aunque las técnicas de votos pesados dieron buenos resultados, se podrían usar otros mecanismos de votación distintos a los aquí usados, como por ejemplo voto acumulativo, de aprobación, plural con eliminación, de borda, entre otros; en aras de comprobar si pueden obtenerse mejores resultados.

5. Combinar esta modalidad de comunicación por medio de ademanes con otra, por ejemplo, voz. Esto se conoce como gestos multimodales. Hay que tener en cuenta que no necesariamente el comando de voz debe coincidir con la orden dada con las manos. Por ejemplo, se puede dar la orden por voz al robot de buscar y con la mano se está señalando qué se debe buscar.

6. Aplicar variaciones al método propuesto, como por ejemplo no eliminar las ultimas ventanas creadas, usar tamaños de crecimiento distintos, usar secuencia estimada de ademanes para predecir, etc. Todo lo anterior con la finalidad de mejorar los resultados en secuencias completas y mas complejas.

7. El método aquí propuesto puede ser aplicado a otras áreas de interés, como por ejemplo, los videos juegos.

Apéndice A

Tabla de parámetros de los experimentos

En la Tabla A.1 se detalla la combinación de parámetros y holgura para las 108 simulaciones realizados para cada estrategia de pesado. La primera columna corresponde al indice del experimento, sigue Cuadro de Inicio (FI), el Cuadro de Parada (FP), la Holgura y la última columna es el tipo de segmentación, donde:

1. Es segmentación un paso atrás.
2. Es segmentación en el punto.
3. Es segmentación un paso adelante.

Tabla A.1: Combinación de valores de los parámetros y holgura de las 108 simulaciones realizados para cada estrategia de pesado.

Num.	FI	FP	Holg.	Seg.
1	2	5	5	1
2	2	5	5	2
3	2	5	5	3
4	2	5	10	1

Tabla A.1 – Continuación de la tabla anterior

Num.	FI	FP	Holg.	Seg.
5	2	5	10	2
6	2	5	10	3
7	2	5	15	1
8	2	5	15	2
9	2	5	15	3
10	2	6	5	1
11	2	6	5	2
12	2	6	5	3
13	2	6	10	1
14	2	6	10	2
15	2	6	10	3
16	2	6	15	1
17	2	6	15	2
18	2	6	15	3
19	2	7	5	1
20	2	7	5	2
21	2	7	5	3
22	2	7	10	1
23	2	7	10	2
24	2	7	10	3
25	2	7	15	1
26	2	7	15	2
27	2	7	15	3
28	2	8	5	1
29	2	8	5	2
30	2	8	5	3

Tabla A.1 – Continuación de la tabla anterior

Num.	FI	FP	Holg.	Seg.
31	2	8	10	1
32	2	8	10	2
33	2	8	10	3
34	2	8	15	1
35	2	8	15	2
36	2	8	15	3
37	3	5	5	1
38	3	5	5	2
39	3	5	5	3
40	3	5	10	1
41	3	5	10	2
42	3	5	10	3
43	3	5	15	1
44	3	5	15	2
45	3	5	15	3
46	3	6	5	1
47	3	6	5	2
48	3	6	5	3
49	3	6	10	1
50	3	6	10	2
51	3	6	10	3
52	3	6	15	1
53	3	6	15	2
54	3	6	15	3
55	3	7	5	1
56	3	7	5	2

Tabla A.1 – Continuación de la tabla anterior

Num.	FI	FP	Holg.	Seg.
57	3	7	5	3
58	3	7	10	1
59	3	7	10	2
60	3	7	10	3
61	3	7	15	1
62	3	7	15	2
63	3	7	15	3
64	3	8	5	1
65	3	8	5	2
66	3	8	5	3
67	3	8	10	1
68	3	8	10	2
69	3	8	10	3
70	3	8	15	1
71	3	8	15	2
72	3	8	15	3
73	4	5	5	1
74	4	5	5	2
75	4	5	5	3
76	4	5	10	1
77	4	5	10	2
78	4	5	10	3
79	4	5	15	1
80	4	5	15	2
81	4	5	15	3
82	4	6	5	1

Tabla A.1 – Continuación de la tabla anterior

Num.	FI	FP	Holg.	Seg.
83	4	6	5	2
84	4	6	5	3
85	4	6	10	1
86	4	6	10	2
87	4	6	10	3
88	4	6	15	1
89	4	6	15	2
90	4	6	15	3
91	4	7	5	1
92	4	7	5	2
93	4	7	5	3
94	4	7	10	1
95	4	7	10	2
96	4	7	10	3
97	4	7	15	1
98	4	7	15	2
99	4	7	15	3
100	4	8	5	1
101	4	8	5	2
102	4	8	5	3
103	4	8	10	1
104	4	8	10	2
105	4	8	10	3
106	4	8	15	1
107	4	8	15	2
108	4	8	15	3

Apéndice B

Gráficas de medida-f por tipo de holgura

En este apéndice se muestra los mismos resultados de la gráfica de medida-f del segundo grupo de simulaciones, pero discriminado por los distintos valores de holgura. De esta forma se tiene en la Figura B.1 una holgura de 5, en la Figura B.2 una holgura de 10 y en la Figura B.3 una holgura de 15.

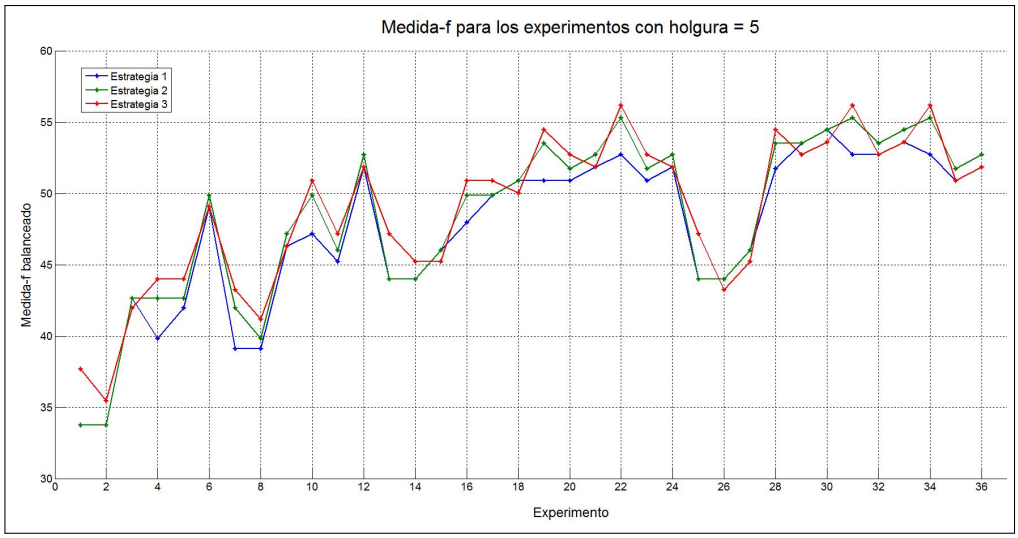


Figura B.1: Valores de la medida-f con un valor de holgura de 5 para el segundo grupos de simulaciones, donde se prueba las tres estrategias de pesado.

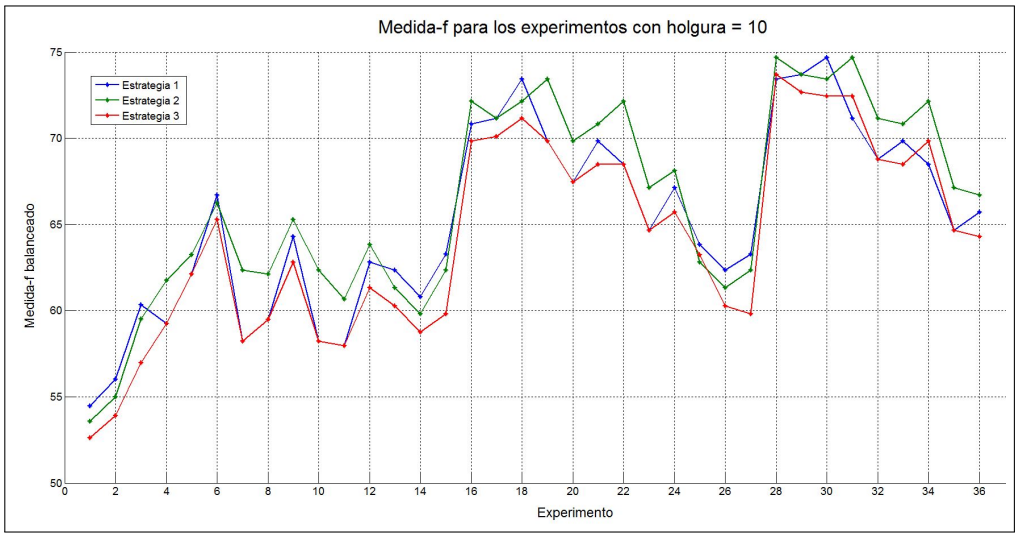


Figura B.2: Valores de la medida-f con un valor de holgura de 10 para el segundo grupos de simulaciones, donde se prueba las tres estrategias de pesado.

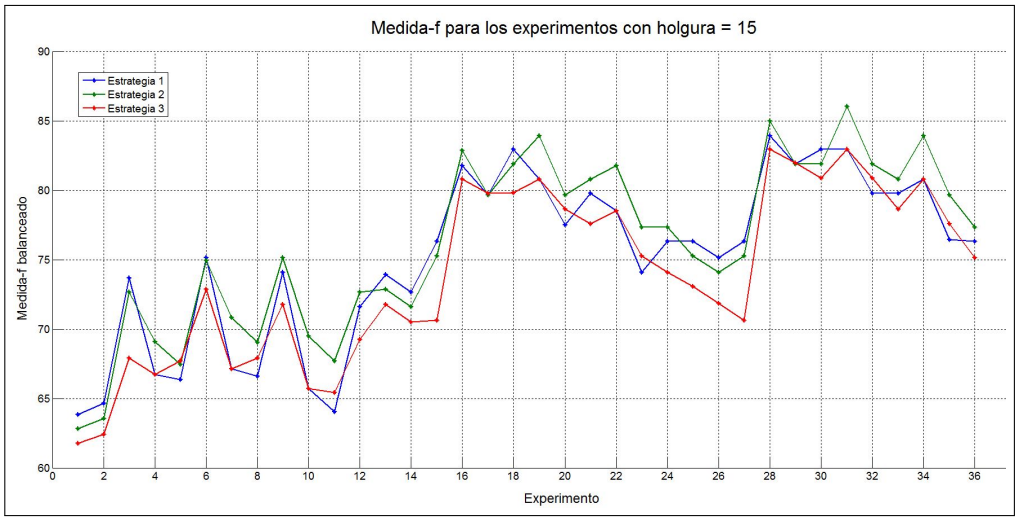


Figura B.3: Valores de la medida-f con un valor de holgura de 15 para el segundo grupos de simulaciones, donde se prueba las tres estrategias de pesado.

Bibliografía

Aracil R., Balaguer C., et Armada M. (2008). Robots de servicio. *Revista Iberoamericana de Automática e Informática Industrial*, 5(2):6–13.

Arriaga H. A., Succar L. S., Durán C. M., et Cortés L. P. (2011). A comparison of dynamic naive bayesian classifiers and hidden markov models for gesture recognition. *Journal of Applied Research and Technology*, 9(1):81–102.

Aviles-Arriaga H., Sucar L., Morales E., Vargas B., et Corona E. (2009). Markovito: A Flexible and General Service Robot. En Springer Berlin / Heidelberg editores, *Design and Control of Intelligent Robotic Systems*, volume 177/2009 of *Studies in Computational Intelligence*, pp. 401–423. Springer.

Boodidhi S. (2011). *Using Smoothing Techniques to Improve the Performance of Hidden Markov Model*. Master of Science in Computer Science. Digital Cholarship UNLV, University of Nevada, Las Vegas.

Brethes L., Menezes P., Lerasle F., et Hayet J. (2004). Face tracking and hand gesture recognition for human-robot interaction. En *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, volume 2, pp. 1901–1906 Vol.2.

Cambridge University Engineering Department (CUED) (1993). http://htk.eng.cam.ac.uk/. Consultado en Abril de 2013.

Chen X., Stone P., Sucar L. E., et van der Zant T. (2013). *RoboCup 2012: Robot Soccer World Cup XVI*. Springer Berlin Heidelberg.

Correa M., Ruiz-del Solar J., Verschae R., et Castillo J. L.-F. (2010). Real-time hand gesture recognition for human robot interaction. *RoboCup 2009: Robot Soccer World Cup XIII*, 5949(1):46–57.

Doliotis P., Stefan A., McMurrough C., Eckhard D., et Athitsos V. (2011). Comparing gesture recognition accuracy using color and depth information. En *Proceedings of the 4th International Conference on PErvasive Technologies Related to Assistive Environments*, PETRA '11, pp. 20:1–20:7, New York, NY, USA. ACM.

Dunham, Matt and Murphy, Kevin (2010). https://code.google.com/p/pmtk3/. Consultado en Abril de 2013.

Ferguson J. D. (1980). Hidden markov analysis: An introduction. En *IDA-CRD, The Symposium on the Applications of Hidden Markov Models to Text and Speech*.

Francois, Jean-Marc (2005). https://code.google.com/p/jahmm/.

Goodrich M. A. et Schultz A. C. (2007). Human-robot interaction: A survey. *Foundations and Trends in Human-Computer Interaction*, 1(3):203–275.

Hu M.-K. (1962). Visual pattern recognition by moment invariants. *Information Theory, IRE Transactions on*, 8(2):179–187.

Jarrett W. et James A. (2012). *Beginning Kinect Programming with the Microsoft Kinect SDK*. Apress.

Kahol K., Tripathi P., Panchanathan S., et Rikakis T. (2003). Gesture segmentation in complex motion sequences. En *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, volume 2, pp. II–105–8 vol.3.

Kanungo, Tapas (2013). http://www.kanungo.com/.

Kim D., Song J., et Kim D. (2007). Simultaneous gesture segmentation and recognition based on forward spotting accumulative hmms. *Pattern Recognition*, 40(11):3012–3026.

Koller D. et Friedman N. (2009). *Probabilistic Graphical Model*. The MIT Press.

Kramer J., Burrus N., Echtler F., Herrera D., et Parker M. (2012). *Hacking the Kinect*. Apress.

Kumar P., N.R.V.Praneeth, et Sudheer.V (2010). Hand and finger gesture recognition system for robotic application. *International Journal of Computer Communication and Information System (IJCCIS)*, 2(1):266–269.

Levinson E. S., Rabiner R. L., et Sondhi M. M. (1983). An introduction to the application of the theory of probabilistic functions of a markov process to automatic speech recognition. *Bell Sys. Tech. Journal*, 62(4):1035–1074.

Li H. (2010). *Model-Based Segmentation and Recognition of Continuous Gestures*. Tesis doctoral, Queen's University, Kingston, Ontario, Canada.

Li H. et Greenspan M. (2011). Model-based segmentation and recognition of dynamic gestures in continuous video streams. *Pattern Recognition*, 44(8):1614 – 1628.

MacDonald I. et Zucchinni W. (1997). *Hidden Markov and Other Models for Discrete Valued Time Series*. Chapman and Hall.

Mitra S. et Acharya T. (2007). Gesture recognition: A survey. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 37(3):311–324.

Murphy, Kevin (1998). http://www.cs.ubc.ca/ murphyk/Software/HMM/hmm.html.

Nguyen-Duc-Thanh N., Lee S., et Kim D. (2012). 2-stage hidden markov model in gesture recognition for human robot interaction. *International Journal of Advanced Robotic Systems: Human Robot Interaction*, 9(1):1–10.

Otero N., Knoop S., Nehaniv C., Syrdal D., Dautenhahn K., et Dillmann R. (2006). Distribution and recognition of gestures in human-robot interaction. En *Robot and Human Interactive Communication, 2006. ROMAN 2006. The 15th IEEE International Symposium on*, pp. 103–110.

Poritz A. (1988). Hidden markov models: a guided tour. En *Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988 International Conference on*, pp. 7–13 vol.1.

Rabiner L. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

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

Real Academia Española (2001). www.rae.es/. Consultado en Abril de 2013.

Rehr T., Theissing N., Bannat A., Gast J., Arsic D., Wallhoff F., Rigoll G., Mayer C., et Radig B. (2010). Graphical models for real-time capable gesture recognition. En *Image Processing (ICIP), 2010 17th IEEE International Conference on*, pp. 2445–2448.

Schliep, Alexander (2010). http://www.ghmm.org/.

Sebastien M. (2002). Gestures for multi-modal interfaces: A review. En *Idiap*.

Song Y., Demirdjian D., et Davis R. (2012). Continuous body and hand gesture recognition for natural human-computer interaction. *ACM Trans. Interact. Intell. Syst.*, 2(1):5:1–5:28.

Van den Bergh M., Carton D., de Nijs R., Mitsou N., Landsiedel C., Kuehnlenz K., Wollherr D., Van Gool L., et Buss M. (2011). Real-time 3d hand gesture interaction with a robot for understanding directions from humans. En *RO-MAN, 2011 IEEE*, pp. 357–362.

Waldherr S., Romero R., et Thrun S. (2000). A gesture based interface for human-robot interaction. *Autonomous Robots*, 9(2):151–173.

Yan R., Tee K.-P., Chua Y., Li H., et Tang H. (2012). Gesture recognition based on localist attractor networks with application to robot control [application notes]. *Computational Intelligence Magazine, IEEE*, 7(1):64–74.

Yang H.-D., Park A.-Y., et Lee S.-W. (2006). Human-robot interaction by whole body gesture spotting and recognition. En *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 4, pp. 774–777.

Zhu C. et Sheng W. (2011). Wearable sensor-based hand gesture and daily activity recognition for robot-assisted living. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 41(3):569–573.