UNIVERSIDAD NACIONAL AUTÓNOMA DE MEXICO

DIVISIÓN DE ESTUDIOS DE POSGRADO - FACULTAD DE INGENIERÍA

RECONOCIMIENTO DE GESTOS CLAVE PARA INTERACTUAR CON UN ROBOT MÓVIL

T E S I S

QUE PARA OBTENER EL GRADO DE
MAESTRO EN INGENIERÍA (ELÉCTRICA)

P R E S E N T A

WOLF SCHAARSCHMIDT

DIRECTOR

DR. JESÚS SAVAGE CARMONA

CUIDAD UNIVERSITARIA, MEXICO, D.F., AGOSTO 2004

Agradecimientos

A la UNAM y los profesores de la División de Estudios de Posgrado de la Facultad de Ingeniería, que compartieron conmigo sus conocimientos y experiencias.

Al Dr. Jesús Savage Carmona por su amistad, por el apoyo y los conocimientos que me brindo en la realización de esta tesis.

A los integrantes del jurado, por aceptar formar parte de mi jurado:

Dr. Boris Escalante Ramirez

Dr. Jesús Savage Carmona

Dr. Pablo Pérez Alcazar

Dr. Abel Herrero Camacho

Dr. Rogelio Alcántara Silva

Resumen

El objetivo de esta tesis es el reconocimiento de un número de gestos clave, utilizando sólo una cámara de video. La cámara de video graba imágenes de los movimientos del usuario. Se filtra el ruido de fondo. Usando un modelo de la piel humana (de tipo Caucásico), se extraen tres regiones que representan la cara del usuario y ambas manos. Usando el algoritmo de K-medias, se calculan los centros de masa de estas tres regiones. Las posiciones de la cabeza y las manos son constantemente calculadas; si hay algún movimiento entre dos cuadros, las diferencias entre dichas posiciones no son cero. Así se obtienen tres vectores de movimiento. Usando la cuantización vectorial (VQ), este espacio de movimiento se codifica en un 'codebook' relativamente pequeño de menos de mil movimientos. Éstos movimientos codificados sirven como símbolos de observación para varios modelos ocultos de Markov (HMMs). Se obtiene como resultado el mejor camino con la probabilidad máxima dadas las observaciones, asi se pueden reconocer visualmente los siguientes gestos clave que hace una persona enfrente de un robot: 'saludar con la izquierda', 'saludar con la derecha', 'aplaudir', 'levantarse', 'sentarse', 'caminar a la izquierda' y 'caminar a la derecha'.

Contenido

Lista de Figuras

Capítulo 1

Introducción

Para aumentar su utilidad, los robots móviles deben observar a los seres humanos y entender lo que ellos están haciendo. Con suerte, ellos deberían deducir la intención del ser humano y ofrecerle ayuda según sus posibilidades. Como un reconocedor general de acciones humanas no parece factible, esta tesis se concentra en reconocer ciertos gestos claves e ignorar los otros movimientos, inspirado por una analogía con las palabras claves que reconoce un sistema de reconocimiento de voz.

Recientemente, muchos robots humanoides han sido desarrollados principalmente por varias empresas en Japón, pero también en algunas universidades europeas. A diferencia de los robots de automatización de las fábricas, los robots humanoides se diseñan para interactuar con los seres humanos. Estos robots deben entender lo que las personas alrededor de ellos están haciendo y lo que ellos quieren lograr. Para ser realmente útiles, estos robots no pueden (solamente) dirijirse por control remoto, como en un videojuego - ellos deben tomar alguna iniciativa, o por lo menos ser capaces de tomar acciones complejas en respuesta a una orden simple. Ellos deben tener en cuenta el estado de su ambiente y de las personas alrededor de ellos al obedecer una orden. Ellos deben tener un poco de comprensión de qué es lo que las personas típicamente intentan lograr en situaciones diferentes (algún tipo de sentido común) - además de ser los observadores hábiles.

Sólo los robots más avanzados pueden caminar realmente en dos pies; la mayoría de los otros confía en la locomoción de ruedas. Entonces la mayoría de ellos se confinará a una casa doméstica o a la oficina. Así ellos se expondrán a un número limitado de personas y tendrán la oportunidad de aprender su rango normal de conducta. Probablemente el robot humanoide más famoso es *Asimo* (vea la figura 1.1), alardeando las habilidades ambulantes de dos pies avanzadas. Claro que su costo alto puede impedir su adopción como un sirviente doméstico en el futuro cercano. Japón fue el primero en reconocer la necesidad de desarrollar los robots de servicio para ayudar a su población envejecida, ya que el cambio demográfico es más rápido allí que en cualquier parte en el mundo - para 2015, un 25% de la población tendrá una edad de 65 años o mas.

Figura 1.1: Asimo (Honda, Japón)

Mientras el desarrollo del hardware de los robots está progresando rápidamente, el desarrollo de software de funcionamiento y de modelos de utilidad no está avanzando a la misma velocidad. El robot *Tmsuk cuatro* (vea la figura 1.2), por ejemplo, se controla totalmente mediante un joystick complejo que opera por teléfono móvil - le falta cualquier iniciativa, igual como su hermano mayor *Tmsuk cinco* que se opera como un exo-esqueleto.

El robot de servicio al aire libre *WorkPartner* (vea la figura 1.3) es un ejemplo muy interesante que combina inteligencia artificial y mando del operador directo a través de una variedad de interfaces: el reconocimiento de voz, la medida directa de movimientos del operador con potenciómetros, el reconocimiento del gestos por cámara y apuntando con un indicador de láser. Es deseable poder interactuar con estos robots utilizando gestos naturales.

El objetivo de esta tesis es el reconocimiento de un número de gestos clave, utilizando sólo una cámara de video.

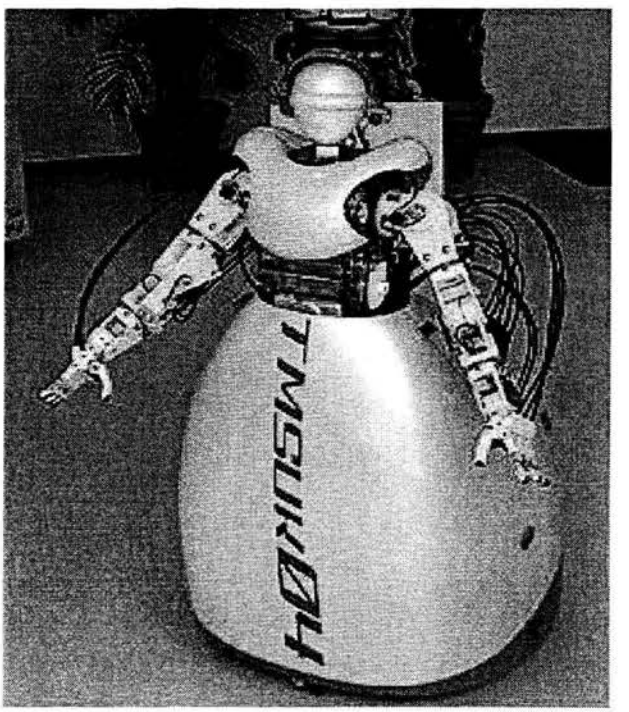

Figura 1.2: Tmsuk cuatro (TMSUK, Japón)

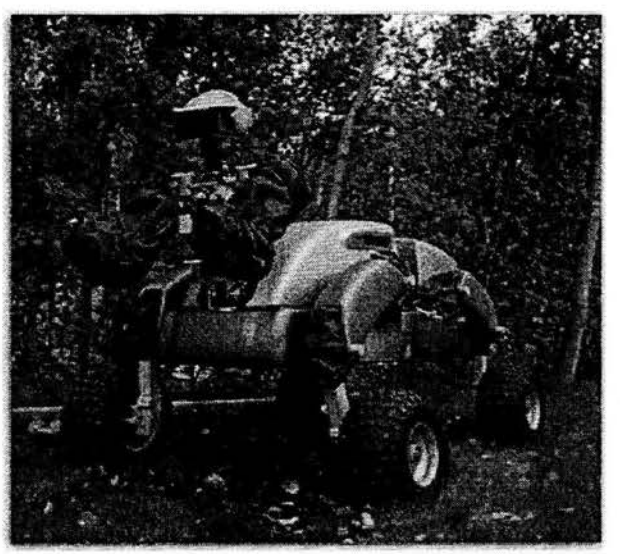

Figura 1.3: WorkPartner (Helsinki University of Technology, Finlandia)

Capítulo 2

Un robot de servicio doméstico: Wakamaru

Con *Wakamaru*, Mitsubishi Heavy Industrias quiso desarrollar "un robot que es amistoso a las personas y útil en casa" cuyo propósito es "proponer un nuevo estilo de vida, conviviendo con un robot en casa que permita hacer la vida más conveniente y enriquecida". Más que la mayoría de las otras compañías, ellos enfatizan la importancia del robot interactuando y comunicandose con las personas (vea la figura 2.1). Se planea introducir este robot al mercado japonés en 2004 por un precio de aproximadamente $15,000\ US\$$.

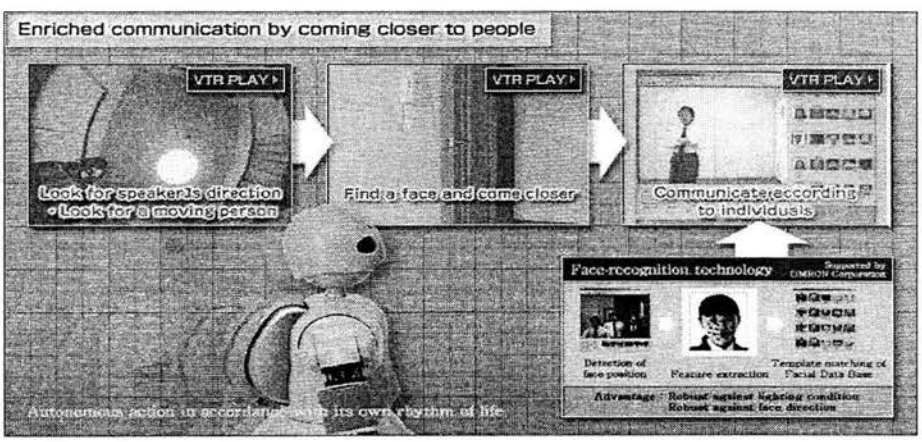

Figura 2.1: Wakamaru (Mitsubishi Heavy Industries, Japón)

La siguiente imagen retrata una de las situaciones importantes que también se cubre en esta tesis: Wakamaru espera que la familia llegue a casa. Cuando ellos llegan y le saludan, él también les saluda (vea la figura 2.6). Otra situación: Wakamaru reconoce que la familia está desayunando (vea la figura 2.2) y ofrece

traer más comida (vea la figura 2.3).

Figura 2.2: Wakamaru en la mesa de desayuno

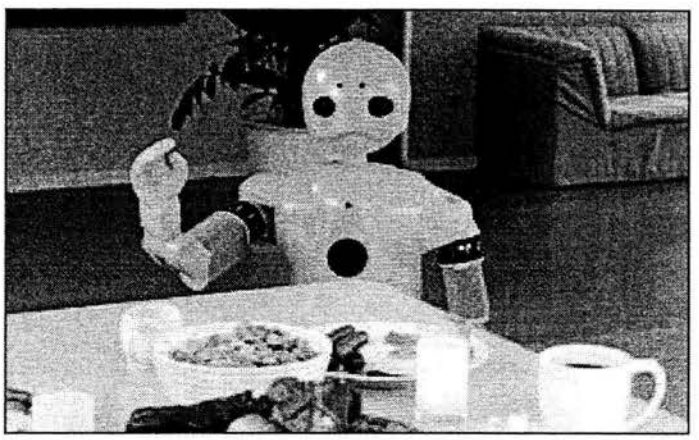

Figura 2.3: Wakamaru ofreciendo traer mas comida

Wakamaru navega observando los patrones ópticos en el techo y así siempre sabe en qué cuarto está (vea la figura 2.4). Eso es importante para interpretar las acciones de las personas alrededor de él, cuyo significado puede depender del cuarto en que ellas se encuentren. Observando a las personas alrededor de él y reconociendo sus intenciones, Wakamaru tiene el potencial para volverse un robot verdaderamente útil (vea la figura 2.5).

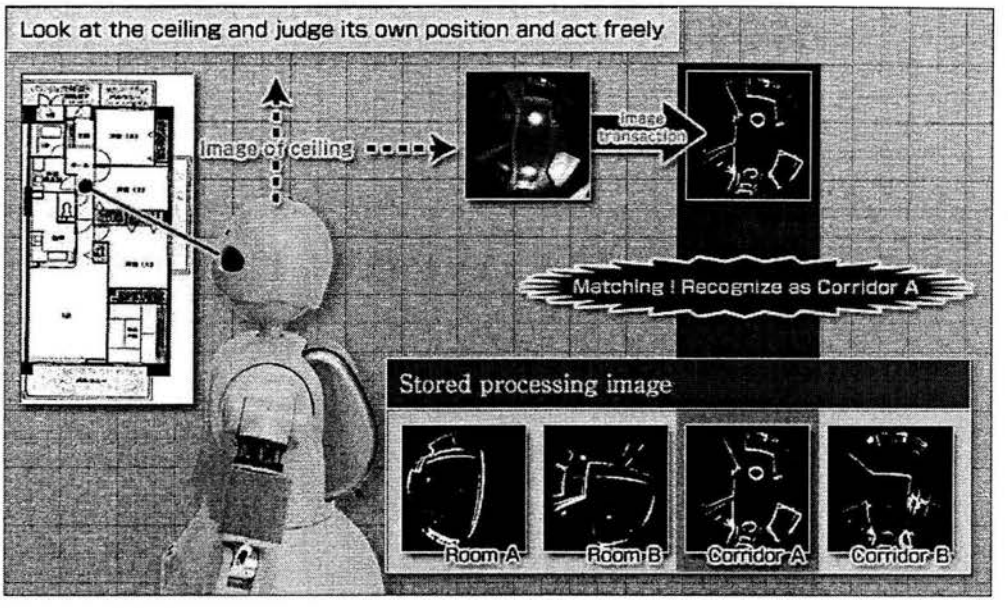

Figura 2.4: Wakamaru navegando la casa

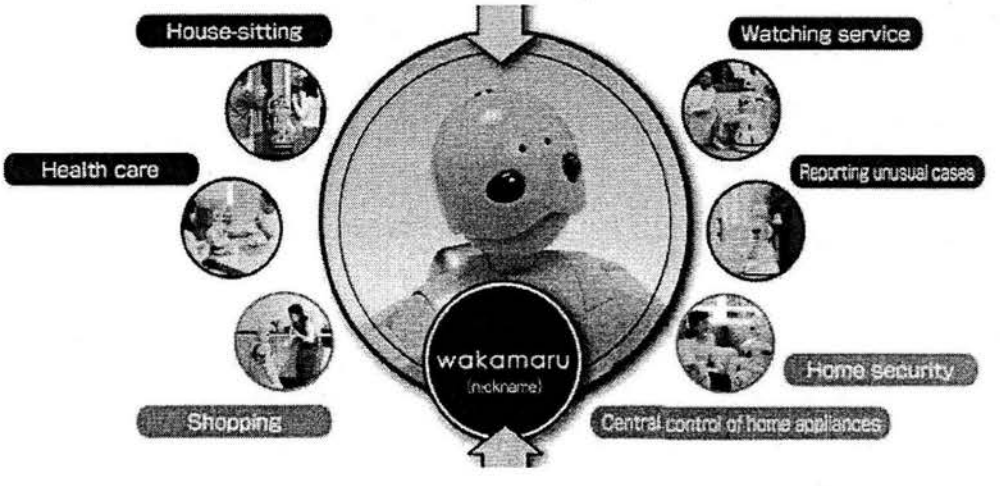

Figura 2.5: Wakamaru - un robot versátil

Figura 2.6: Wakamaru saludando a la familia

Capítulo 3

Vision global del sistema

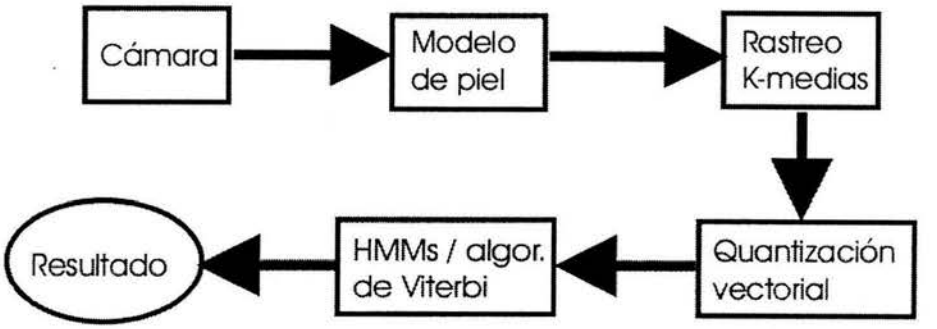

Figura 3.1: Diagrama de bloques: vision global del sistema

Una cámara de video con una lente de ángulo ancho graba imágenes de los movimientos del usuario. Las imágenes se capturan con una videograbadora y se filtra el ruido de fondo. Usando un modelo de la piel humana (de tipo Caucásico), se extraen tres regiones que representan la cara del usuario y ambas manos. Usando el algoritmo de K-medias, se calculan los centros de masa de estas tres regiones. Otras estadísticas, como el área y la forma, también son calculadas. Éstas no se usan actualmente, pero pueden servir para extensiones futuras al análisis de movimiento en tres dimensiones. Las posiciones de la cabeza y las manos son constantemente calculadas; si hay algún movimiento entre dos cuadros, las diferencias entre dichas posiciones no son cero. Así se obtienen tres vectores de movimiento. Por simplicidad, se hace referencia a estos vectores como 'el vector de movimiento'. Dependiendo de la resolución de la cámara, el espacio de vectores de movimiento es potencialmente muy grande. Entonces, se requiere alguna simplificación. Usando la cuantización vectorial

(VQ), este espacio de movimiento se codifica en un 'codebook' relativamente pequeño de menos de mil movimientos. Éstos movimientos codificados sirven como la base para el análisis extenso. Utilizando varios modelos ocultos de Markov (HMMs), se generan hipótesis de movimientos de los códigos de VQ. Para imponer un orden a estas hipótesis, ellas alimentan entonces a un meta-HMM llamado 'la gramática' de movimientos. La gramática decide el camino mejor a través del espacio de secuencias de movimientos, usando el algoritmo de Viterbi. Se obtiene como resultado el mejor camino, con la probabilidad máxima, dadas las observaciones, asi se pueden reconocer visualmente acciones que hace una persona enfrente de un robot. La gramática de movimientos representa los siguientes estados: sentándose, sentado, levantándose, caminando.

3.1 Escogiendo a una tecnología

Existen muchas opciones diferentes para capturar el movimiento humano, entre las mas populares estan 'visión natural', 'visión con ayuda' u 'otros medios técnicos'. En la primera opción, la 'visión natural', los movimientos de los seres humanos son observados por una o más cámaras, sin cualquier marca especial en la persona. En la segunda opción, 'visión con ayuda', algunas partes del cuerpo son marcadas con colores o patrones especiales para facilitar su reconocimiento por la computadora. La última opción, 'otros medios técnicos', incluye todos los otros dispositivos para determinar posición o configuración, como los sensores magnéticos de Polhemus, los datagloves, palancas 3-D de mando y similares.

De la primera a la tercera opción, la fiabilidad aumenta, pero también los requisitos del recurso y la molestia del usuario. En esta tesis, se escogió la primera opción: los movimientos de miembros sin marcas son capturados usando una sola cámara. Varios métodos existen para extraer el movimiento humano de una secuencia de imágenes capturadas por una sola cámara.

Uno de los acercamientos más simples involucra las imágenes de diferencia, es decir, simplemente substrayendo un cuadro de imagen de un cuadro anterior. Asumiendo que la iluminación es constante, todas las partes de la imagen que no cambiaron se cancelarán exactamente, dejando una diferencia de cero. Sólo en las áreas de movimiento la imagen tiene una diferencia no cero (positiva o negativa). La figura 3.2 muestra un movimiento lateral de la mano, bajo condiciones realistas; es decir con ruido. Como las imágenes reales siempre contienen ruido, se utiliza un cierto umbral de movimiento por área para seleccionar un rectangulo de 'área de interés'.

Puesto que el poder de esta técnica es bastante limitado, su uso parece restringido a seleccionar una área de interés para ser procesada por los métodos más poderosos. Todavía, para una aplicación de tiempo real, puede ser importante limitar el área sujeta a procesos más extensos por algún medio computacionalmente barato. Uno de los métodos más poderosos, pero también computacionalmente mas caros se llama 'flujo óptico'. Empezando con dos cuadros diferentes, involucra establecer las correspondencias entre pixeles que se han movido, y de esto, obtener los vectores de movimiento. La asunción sub-

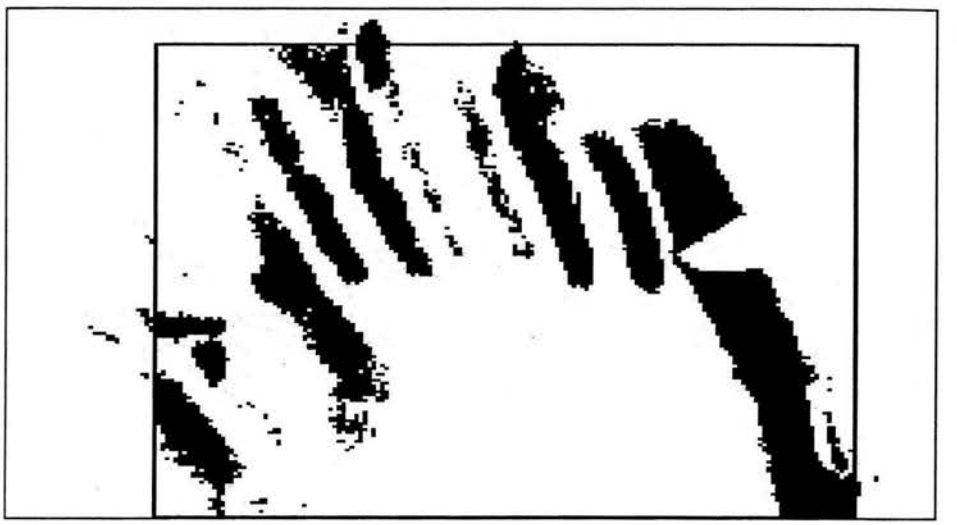

Figura 3.2: Imagen de diferencia : movimiento de la mano

yacente es que ambos cuadros contienen los mismos objetos, pero posiblemente en posiciones diferentes.

La figura 3.3 muestra el resultado de un experimento que involucra un movimiento lateral de la mano derecha. El algoritmo de flujo óptico en este caso descubre un campo de vectores de movimiento más diagonal y no homogeneo. En algunos otros experimentos, los vectores de movimiento descubiertos eran ortogonales al movimiento real. Cabe aclarar que los algoritmos de flujo ópticos se evaluan casi siempre usando secuencias de cuadros totalmente sintéticas, en lugar de reales (vea [1]). Por esta falta de fiabilidad, no se usó esta técnica en la tesis.

Aunque se decidió no usar una señal artificial en los miembros humanos, la piel humana es naturalmente de un color bastante distinto y no compartido por muchos objetos artificiales. Por consiguiente, se utiliza el color de piel para habilitar el rastreo de las manos humanas y la cara. Para distinguir fielmente entre piel y no piel, un modelo de piel fue desarrollado. Los movimientos descubiertos por el rastreador de K-medias son procesados por el cuantizador vectorial y entonces analizados usando los modelos ocultos de Markov.

3.2 Trabajo relacionado

Barron, Fleet & Beauchemin [1] han usado secuencias de imagenes sintéticas para evaluar el comportamiento de varios algoritmos de flujo óptico. Baumberg & Hogg [2] usaron imágenes de diferencia y siluetas para rastrear el movimiento de las personas. Cutler & Turk [3] usaron flujo óptico y crecimiento de regiones para reconocer seis gestos diferentes. Heap & Samaria [5] usaron modelos de-

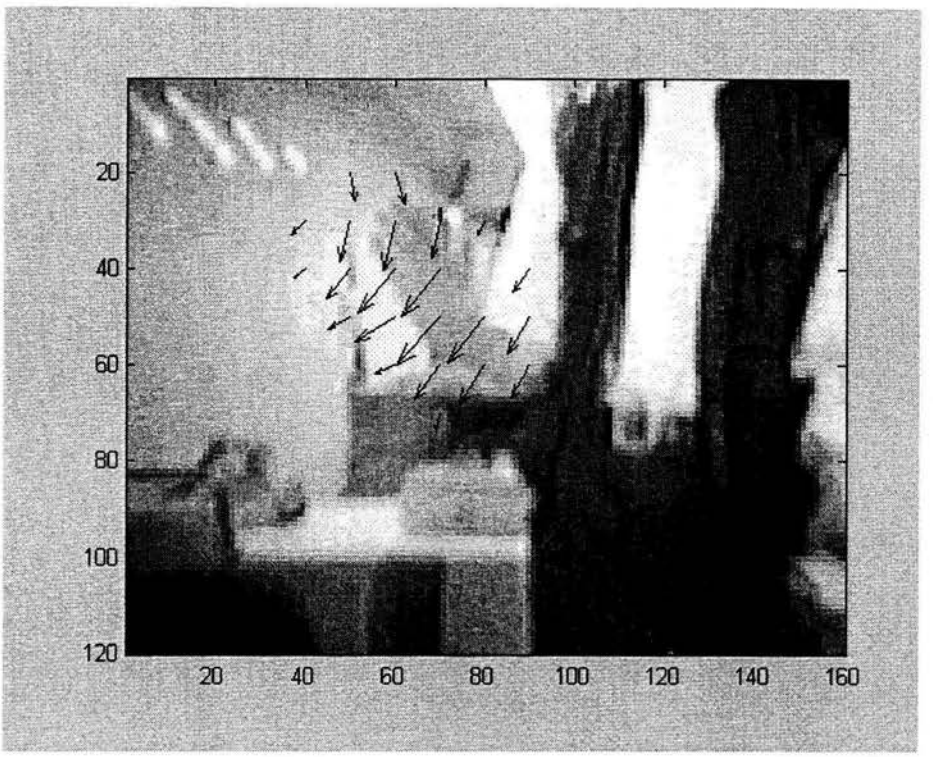

Figura 3.3: Flujo óptico: movimiento de la mano

formables de forma activa, también conocidos como 'serpientes inteligentes', para habilitar el rastreo en tiempo real y reconocimiento de gestos. Las 'serpientes inteligentes' sólo funcionan bien cuando se empieza con una estimación inicial muy buena que en su sistema se proporciona por un algoritmo genético. Ellos dependen de las fuentes de illuminación adicionales en situaciones específicas. Huang & Pavlovic [6] revisaron varios métodos para el modelado de gestos de mano. Liu & Lovell [8] compararon el comportamiento del algoritmo de Baum-Welch y lo de Viterbi entrenando HMMs para reconocer gestos de la mano. McKenna & Gong [9] usaron imágenes de diferencia y trayectorias de evento para reconocer algunos gestos simples de la mano. Ong, McKenna & Gong [10] usaron un sensor de Polhemus puesto en la cabeza para generar una base de datos de vistas y rastrear la cabeza, con el fin de deducir la intención. Starner & Pentland [12] usaron guantes colorados para facilitar el reconocimiento del idioma de signos americano, utilizando modelos ocultos de Markov.

Capítulo 4

Rastreo

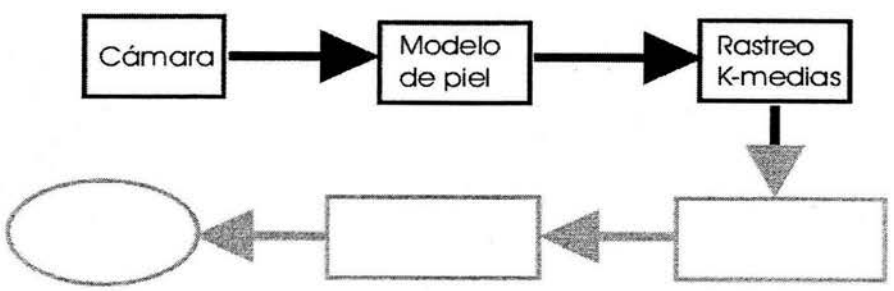

Figura 4.1: Diagrama de bloques de rastreo

4.1 La derivación del modelo de piel

El propósito del modelo de piel es distinguir la piel de los elementos no piel en la imagen. Primero, para generar el modelo de piel, se toma una muestra de la piel (en este ejemplo, el tipo de la piel es Caucásico). La muestra de la piel es una fotografía cortada en el modelo de color RGB (vea la Figura 4.2).

La colección de (principalmente) pixeles de piel en el espacio RGB se muestra en la Figura 4.3. Los valores de RGB estan entre 0 y 255. Analizando cada par de dimensiones (R-B, R-G, B-G), se puede derivar un conjunto de desigualdades. Estas desigualdades corresponden a planos que delimitan los pixeles de piel en el espacio de RGB.

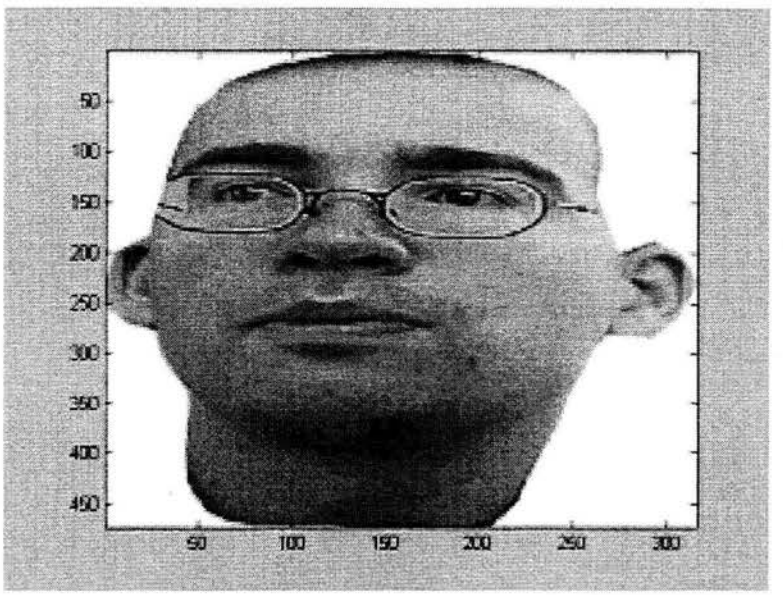

Figura 4.2: Muestra de piel

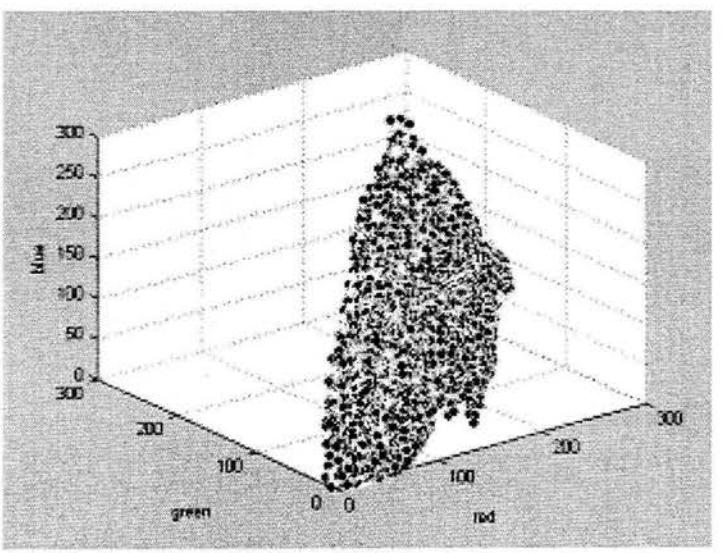

Figura 4.3: Pixeles de piel en el espacio de color RGB

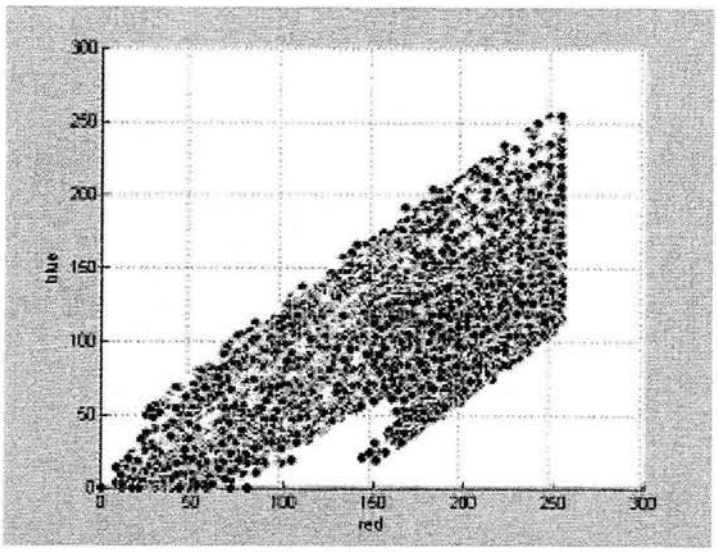

Figura 4.4: Subespacio RB

Analizando las dimensiones rojo contra el azul, se derivan las desigualdades siguientes (vea la Figura 4.4):

$$B < R < B + 150 \tag{4.1}$$

Analizando las dimensiones rojo contra el verde, se derivan las desigualdades siguientes(vea la Figura 4.5):

$$G < R < G + 150 \tag{4.2}$$

Analizando las dimensiones verde contra el azul, se derivan las desigualdades siguientes(vea la Figura 4.6):

$$B - 25 < G < B + 50 \tag{4.3}$$

Otra suposición es que si la iluminación cambia, las tres dimensiones cambian proporcionalmente. Si ése es el caso, el modelo permanecerá válido a través de un rango amplio de condiciones de iluminación.

4.2 Enmascaramiento de artefactos

Como se usa una cámara de video normal, la imagen resultante es de calidad relativamente baja. En particular, cambios rapidos entre partes blancas y negras de la imagen tienden a generar artefactos coloreados que podrían confundirse

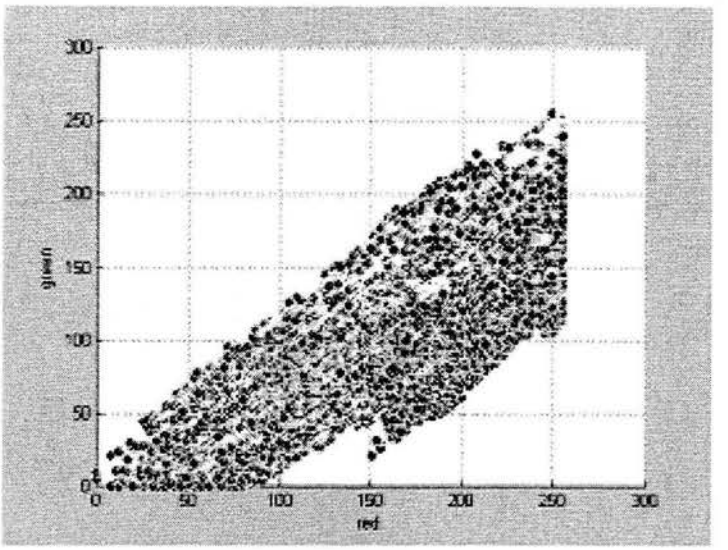

Figura 4.5: Subespacio RG

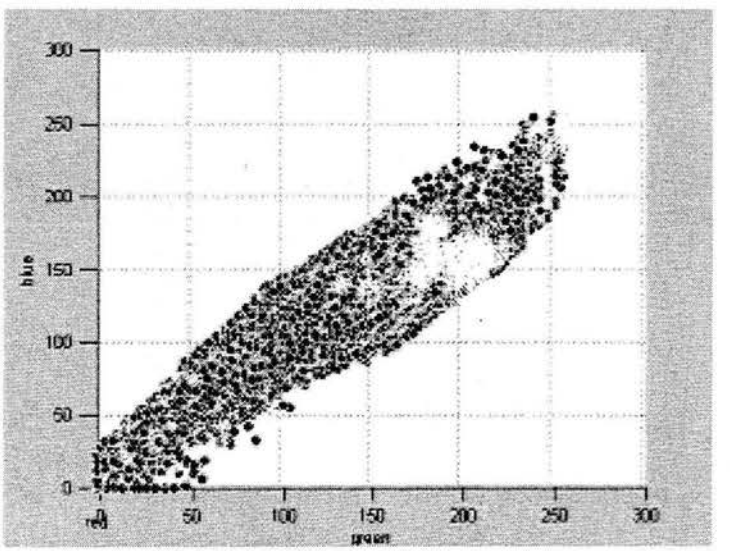

Figura 4.6: Subespacio GB

con la piel. También, en un cuarto desordenado, algunos objetos podrían tener colores similares a la piel. Por consiguiente, se requiere bloquear artefactos de la imagen. Así en una fase de calibración inicial, se graba una imagen del ambiente. De esta imagen, se calcula una máscara de bloqueo a partir de los pixeles clasificados como piel (aunque no lo sean). En lo siguiente, estos pixeles bloqueados nunca se consideran como piel. Ademas, pixeles aislados con color de piel se consideran artefactos, y entonces no se los toma en cuenta.

4.3 K-medias

K-medias es una técnica para realizar una clasificación multidimensional de una colección de elementos (normalmente vectoriales) en un número predeterminado de K regiones. En muchos casos, una interpretación geométrica es posible: K puntos sirven como puntos de atracción para los elementos vectoriales que se agrupan alrededor del punto más cercano a ellos. Esos puntos de K no son fijos, pero son calculados de manera reiterativa como la media de los elementos del agrupamiento asociado a ellos. Así se deriva el nombre del algoritmo, K-medias. El proceso tiene que ser inicializado con K puntos dados, normalmente de una manera que cubre el espacio vectorial de alguna manera regular. El algoritmo termina cuando alcanza un punto fijo o después de un número predeterminado de reiteraciones. Esquemáticamente, el algoritmo para asociar los N vectores con los K medias es así:

$$\text{Escoge } K \text{ centros iniciales } z_1(1), z_2(1), \ldots, z_K(1) \text{ de } X_1(1), X_2(1), \ldots, X_K(1) \tag{4.4}$$

$$\text{Asigna } x \text{ al agrupamiento } X_i(t) \text{ si } \| x - z_i(t) \| \leq \| x - z_j(t) \|, \quad 1 \leq j \leq K, \quad i \neq j \tag{4.5}$$

$$\text{Compute nuevos centros: } z_i(t+1) = \frac{1}{N_i} \sum_{x \in X_i(t)} x, \quad 1 \leq i \leq K \tag{4.6}$$

$$\text{Si } z_i(t+1) = z_i(t), \quad 1 \leq i \leq K, \quad \text{termina; sino, vuelve a paso 4.5} \tag{4.7}$$

En la aplicación real, el centroide, el área y la variación horizontal tanto como vertical de cada agrupamiento es calculada. Actualmente, sólo se evalúan los movimientos de los centroides en dos dimensiones.

4.4 Rastreando cabeza y manos

La cabeza y manos del usuario se rastrean continuamente usando el algoritmo de K-medias descrito en la sección anterior. Como la posición inicial del usuario es desconocida, las tres medias rastreadas se inicializan al centro de la pantalla. En la primera iteración, los centroides se ponen en el centro de masa de las tres regiones de la piel que representan la cabeza y las manos. Para lograr un rendimiento alto, sólo una iteración de K-medias es computada por cada

cuadro. En la práctica, la velocidad de procesamiento de cuadros es tan alta que el algoritmo puede rastrear al usuario sólo con un retraso muy pequeño.

Durante el rastreo información adicional es calculada. El área de las tres regiones también se graba. Las diferencias en el área podrían servir como señales que indicarían los movimientos en una tercera dimensión 'hacia adelante - hacia atrás', mientras los cambios en la variación pudieran indicar movimientos rotatorios, sobre todo de las manos. La información del área y de la variación no se evalúan en el sistema actual, pero podrían servir como una base para extensiones futuras.

La figura 4.7 muestra tres agrupamientos de pixeles de piel que representan la cabeza y manos del usuario. Los tres son marcados por rectángulos computados cuyo centro corresponde al centroide de cada agrupamiento y de quien la extensión horizontal y vertical corresponde a la variación respectiva del agrupamiento. La figura 4.8 muestra el cuadro siguiente dónde la mano izquierda del usuario se ha movido. Como se usa una cámara de video normal, el entrelazamiento de dos medio-cuadros, asignado por las normas de televisión, causa un fenómeno indeseable llamado 'efecto de peine'. Como los dos medio cuadros (uno que consiste en las líneas pares del cuadro completo, el otro de las líneas impares) se graban en momentos diferentes, los bordes verticales de todos los objetos moviendose horizontalmente se vuelven serradas, formando un patrón de peine. Por esta razón, sería deseable poder usar una cámara de rastreo progresivo.

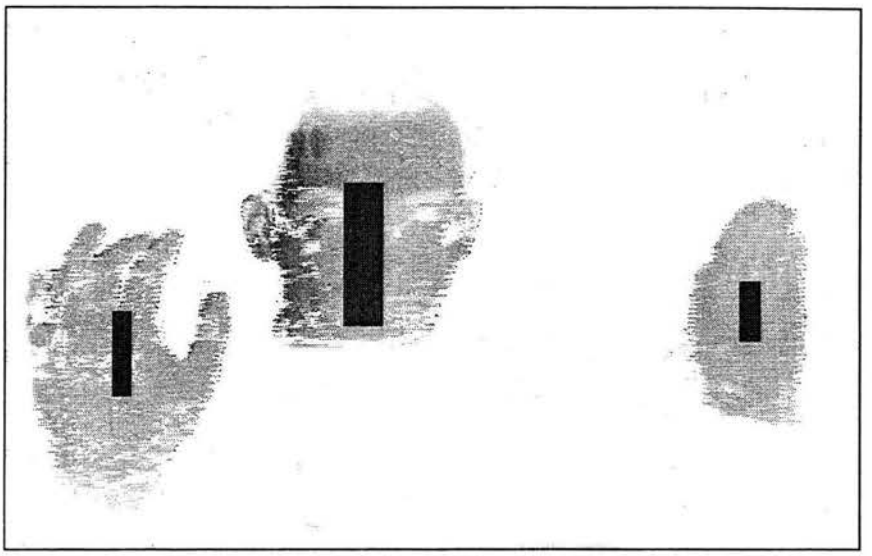

Figura 4.7: Rastreo - empezando saludo

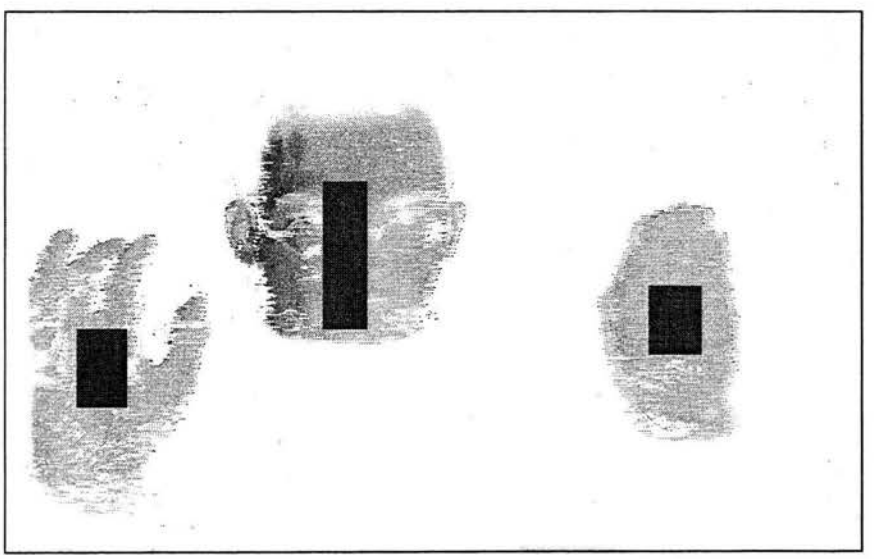

Figura 4.8: Rastreo - terminando saludo

Capítulo 5

Reconcimiento de las acciones

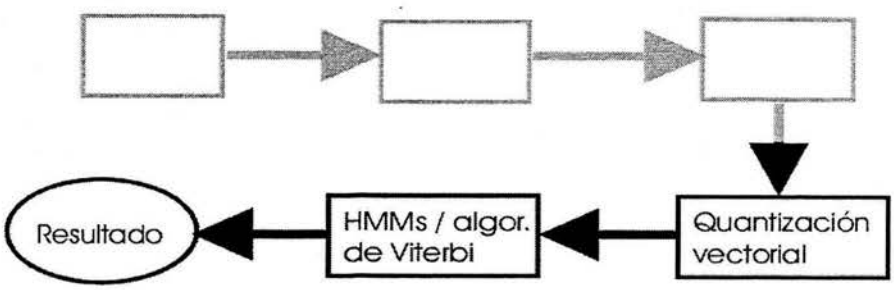

Figura 5.1: Diagrama de bloques de reconocimiento de acciones

5.1 Cuantización vectorial (VQ)

Rastreando los movimientos humanos, el interés no se centra tan solo en un cuadro sino en la diferencia entre dos cuadros. En una imagen de una resolución de $1000 \cdot 1000$ pixeles, un solo punto tiene un millón de movimientos posibles. Así un vector de tres puntos moviendose independientemente corresponde a 10^{18} posibles movimientos. Aunque el movimiento humano es evidentemente más restringido, el número de movimientos todavía es grande. Obviamente, se requiere alguna simplificación para que la evaluación sea computacionalmente factible.

La cuantización normalmente implica alguna pérdida de resolución, por ejemplo cuando se digitaliza un valor analógico en una representación digital de resolución fija. Lo mismo pasa al proyectar un valor de una escala más fina a una escala más tosca. En un ejemplo de la una dimensión, un movimiento horizontal que podría estar entre 1 y 1000 pixeles podría categorizarse en un conjunto de 10 categorías, con categoría uno significando "movimiento entre 1 y 100 pixeles" hasta la categoría 10 que significa "movimiento entre 901 y 1000 pixeles". En la cuantización vectorial, cada categoría se representa por un 'codeword' (palabra clave). El conjunto de todos los codewords forma el 'codebook' (libro de códigos).

La extensión crucial en la cuantización vectorial es que ya no se pueden ordenar linealmente los objetos a cuantizar, pero pueden ser multidimensionales. Un algoritmo como el de K-medias se utiliza para aprovecharse del orden inherente en el conjunto de datos vectoriales para establecer las categorías. Para crear las categorías, primero se debe determinar el tamaño deseado K del codebook. Entonces se corre al algoritmo de K-medias con los datos de muestra para calcular los K medias. Cada agrupamiento que se encontró es representado por una categoría y codificado con un codeword.

En la fase de la aplicación, cada vector de movimiento observado se compara con todas las categorías. Se escoge la categoría que mejor representa al vector de movimiento y su codeword se usa en la codificación de este movimiento. En esta aplicación, los vectores de todos los posibles movimientos de $<$ *mano izquierda*, *cabeza*, *mano derecha* $>$ se representan por un codebook de tamaño $9^3 = 729$ codewords. Las ocho direcciones principales se codifican usando el esquema presentado en la figura 5.2.

Los codewords forman los símbolos de observación para el procesamiento subsecuente por los modelos ocultos de Markov.

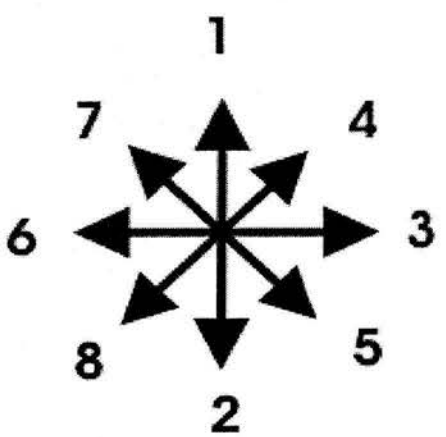

Figura 5.2: Esquema de codificación para cuantización vectorial

5.2 Modelos ocultos de Markov (HMMs)

Un modelo oculto de Markov (HMM de sus siglas en inglés Hidden Markov Model) es un modelo matemático que contiene una variable aleatoria oculta que representa el estado y una variable aleatoria observable que representa los símbolos de la observación. Las probabilidades de las transiciones entre estados de la variable oculta son conocidas, así como las probabilidades de los símbolos generados por la variable observable que son condicionales en el estado de la variable oculta. Para una introducción más detallada a modelos ocultos de Markov, vea [11] cuya nomenclatura se adopta aquí.

5.3 La definición de un modelo oculto de Markov

Un HMM tiene dos parámetros principales: El número de estados, N, y el número de distintos símbolos de observación, M. Estos implican unas N^2 probabilidades de transición entre estados y unas $M \cdot N$ probabilidades de observación, así como N probabilidades de estados iniciales. Algunas de estas probabilidades pueden ser cero (pero por razones de implementación, se ponen valores de un epsilon muy pequeño).

Así un HMM λ se define como una tupla: $\lambda = (A, B, \pi)$ con la distribución de probabilidad de transición entre estados A, distribución de probabilidad de símbolos de observación B y distribución de probabilidad de estados iniciales π; q_t es la variable (oculta) de estado en el momento t, considerando que $S_1 ... S_N$ son los N estados diferentes. $v_1 \cdots v_M$ son los M diferentes símbolos de observación.

Sea la matriz de probabilidades de transiciones entre estados

$$A = \{a_{ij}\} \text{ donde } a_{ij} = P(q_{t+1} = S_j \mid q_t = S_i), 1 \leq i, j \leq N. \tag{5.1}$$

y la matriz de probabilidades de símbolos

$$B = \{b_j(k)\} \text{ donde } b_j = P(v_k \text{ en } t \mid q_t = S_j), 1 \leq j \leq N, 1 \leq k \leq M. \tag{5.2}$$

$$\text{y el vector inicial } \pi = \{\pi_i\} \text{ donde } \pi_i = P(q_1 = S_i), 1 \leq j \leq N. \tag{5.3}$$

Ahora dada una secuencia de observación O:

$$O = O_1 O_2 \ldots O_T \tag{5.4}$$

y un HMM $\lambda = (A, B, \pi)$, queremos calcular la probabilidad $P(O \mid \lambda)$; es decir la probabilidad de la secuencia de observación, asumiendo que fue producida por

el HMM λ. Para hacer esto, debemos de tener en cuenta cada posible secuencia Q de estados de longitud T:

$$Q = q_1 q_2 \dots q_T \tag{5.5}$$

Entonces la probabilidad de la observación O dado un Q particular y un HMM λ es:

$$P(O \mid Q, \lambda) = \prod_{t=1}^{T} P(O_t \mid q_t, \lambda) \tag{5.6}$$

o en términos de probabilidades de observación de símbolos:

$$P(O \mid Q, \lambda) = b_{q_1}(O_1) \cdot b_{q_2}(O_2) \cdots b_{q_T}(O_T) \tag{5.7}$$

De otro lado, la probabilidad de una secuencia particular Q dado λ es:

$$P(Q \mid \lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \dots a_{q_{T-1} q_T} \tag{5.8}$$

La probabilidad conjunta de O y Q es:

$$P(O, Q \mid \lambda) = P(O \mid Q, \lambda) \cdot P(Q \mid \lambda) \tag{5.9}$$

Entonces usando el hecho (particionamiento de un A):

$$P(B) = \sum_i P(B, A_i) = \sum_i P(B \mid A_i) \cdot P(A_i) \tag{5.10}$$

sumando sobre todas las posibles secuencias Q conseguimos:

$$P(O \mid \lambda) = \sum_{all\ Q} P(O \mid Q, \lambda) \cdot P(Q \mid \lambda) \tag{5.11}$$

5.4 El procedimiento hacia adelante - hacia atrás

Como el número de posibles secuencias Q es exponencial en T, no es factible calcular separadamente todos caminos Q y entonces sumarlos. Un método más eficaz, con tiempo de ejecución en el orden de $O(N^2T)$, es el 'procedimiento hacia adelante - hacia atrás' que funciona por inducción sobre secuencias de observación parciales.

Definimos una variable hacia adelante $\alpha_t(i)$ como sigue:

$$\alpha_t(i) = P(O_1 O_2 \dots O_t, q_t = S_i \mid \lambda) \tag{5.12}$$

es decir, la probabilidad de observar los t primeros símbolos y de acabar en estado S_i en el momento t. El procedimiento tiene tres pasos:

$$\alpha_1(i) = \pi_i b_i(O_1),\ 1 \leq i \leq N \tag{5.13}$$

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] \cdot b_j(O_{t+1}),\ 1 \leq t \leq T-1,\ 1 \leq j \leq N \tag{5.14}$$

$$P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{5.15}$$

Como complemento, se define un $\beta_t(i)$ dirigido hacia atrás como sigue:

$$\beta_t(i) = P(O_{t+1} O_{t+2} \ldots O_T \mid q_t = S_i, \lambda) \tag{5.16}$$

es decir, la probabilidad de la secuencia de observación del momento $t+1$ hasta el fin, empezando en estado S_i en el momento t:

$$\beta_T(i) = 1,\ 1 \leq i \leq N \tag{5.17}$$

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} \cdot b_j(O_{t+1}) \cdot \beta_{t+1}(j),\quad t = T-1, T-2, \cdots, 1,\ 1 \leq i \leq N \tag{5.18}$$

Podemos definir la probabilidad γ de estar en estado S_i en el momento t como:

$$\gamma_t(i) = P(q_t = S_i \mid O, \lambda) \tag{5.19}$$

y calcularlo usando α y β:

$$\gamma_t(i) = \frac{\alpha_t(i) \cdot \beta_t(i)}{P(O \mid \lambda)} = \frac{\alpha_t(i) \cdot \beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i) \cdot \beta_t(i)} \tag{5.20}$$

Entonces, γ nos dice el estado individualmente más probable para cada t, cuando adquiere el valor máximo. Por otra parte, estos estados tomados juntos podrían formar una secuencia no posible. Para encontrar la secuencia de estados Q que maximiza $P(Q \mid O, \lambda)$, necesitamos un método diferente llamado el Algoritmo de Viterbi.

5.5 El algoritmo de Viterbi

El algoritmo de Viterbi intenta encontrar el camino óptimo, es decir, el más probable, a través del espacio de estados, dadas las observaciones. En general, hay un número exponencial de caminos, por lo cual la búsqueda exhaustiva no es una opción. Entonces, se utiliza la programación dinámica. El algoritmo de Viterbi funciona por inducción sobre la mejor secuencia de estados hasta el

tiempo t. Se guarda la probabilidad más alta de una secuencia parcial hasta el tiempo t, que acaba en S_i, en la variable $\delta_t(i)$, y el estado anterior en $\psi_t(i)$:

$$\delta_t(i) = \max_{q_1 q_2 \cdots q_{t-1}} P(q_1 q_2 \ldots q_t = i, O_1 O_2 \ldots O_t \mid \lambda) \tag{5.21}$$

$$\delta_{t+1}(j) = \left[\max_{q_1 q_2 \cdots q_{t-1}} \delta_t(i) a_{ij} \right] \cdot b_j(O_{t+1}) \tag{5.22}$$

El algoritmo consiste en los pasos siguientes:

$$\delta_1(i) = \pi_i b_i(O_1), \ 1 \leq i \leq N \tag{5.23}$$

$$\psi_1(i) = 0, \ 1 \leq i \leq N \tag{5.24}$$

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] \cdot b_j(O_t), \ 2 \leq t \leq T-1, \ 1 \leq j \leq N \tag{5.25}$$

$$\psi_t(j) = \arg \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}], \ 2 \leq t \leq T-1, \ 1 \leq j \leq N \tag{5.26}$$

$$P^* = \max_{1 \leq i \leq N} [\delta_T(i)] \tag{5.27}$$

$$q_T^* = \arg \max_{1 \leq i \leq N} [\delta_T(i)] \tag{5.28}$$

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \ t = T-1, T-2, \cdots, 1 \tag{5.29}$$

Como la probabilidad de un camino largo a través del espacio de estados implica el producto de varias probabilidades pequeñas, los números involucrados pueden volverse bastante pequeños. Entonces, es aconsejable usar una representación logarítmica para evitar los problemas numéricos como el 'underflow' (números que se hacen tan pequeños que la computadora no puede distinguirlos del cero).

5.6 Entrenamiento de los HMMs

Para un HMM ligeramente complejo, un número muy grande de probabilidades de transición, así como de probabilidades de símbolos condicionales, tiene que ser establecido. Éstas probabilidades normalmente no se conocen a priori y son bastante difíciles de determinar. Entonces, el HMM tiene que ser entrenado presentándole un conjunto representativo de observaciones, para que pueda deducir aproximaciones de las probabilidades requeridas. El algoritmo típicamente presentado en la literatura para lograr esta estimación de parámetros es el algoritmo de Baum-Welch, también conocido como el método de la maximización de espectaciones.

Desafortunadamente, el algoritmo de Baum-Welch, incluso en el formulario logarítmico, es numéricamente muy inestable sin las medidas extensas de estabilización. Por consiguiente, el algoritmo de Viterbi presentado en el último capítulo también se utiliza para entrenar a HMMs. Para aumentar la eficacia del entrenamiento, es útil tener algunas estimaciones a priori de las probabilidades de transición y de los símbolos, pero eso no es estrictamente necesario.

El procedimiento es utilizar el HMM existente para evaluar una serie de observaciones. Para cada secuencia de observaciones, el mejor camino a través de espacio de estados, dado el HMM actual, es calculado. Entonces se cuentan todas las transiciones y observaciones de símbolos. De estas frecuencias, se pueden calcular las probabilidades empíricas. Estas probabilidades empíricas se usan para actualizar el HMM. Las observaciones siguientes serán clasificadas con mas precisión por el HMM mejorado.

Este ciclo de clasificar las observaciones y actualizar el HMM se repite varias veces. Para una serie suficientemente larga de observaciones, las probabilidades calculadas se acercarán a las frecuencias condicionales reales.

5.7 Evaluando el entrenamiento con el algoritmo de Baum-Welch

Para evaluar el algoritmo de Baum-Welch se utiliza el arreglo siguiente: Un cierto HMM predeterminado, llamado el 'HMM dado', fue empleado para producir secuencias de símbolos según su modelo inherente $\lambda_{given} = (A_g, B_g, \pi_g)$. Entonces un nuevo HMM, llamado el 'HMM aprendiz', fue construido con el mismo número de estados y símbolos, pero con todas las probabilidades de transición y probabilidades de símbolos inicialmente uniformes, $\lambda_{learn} = (A_l, B_l, \pi_l)$. Observando a las secuencias O_j generadas por el HMM dado, el HMM aprendiz debe adaptar sus parámetros A_l, B_l, π_l para maximizar la probabilidad $P(O_j \mid \lambda_{learn})$. La probabilidad promedio $P(O_j \mid \lambda_{given}) = .004516$ se toma como un valor de referencia. A medida que $P(O_j \mid \lambda_{learn})$ se acerca a este valor de referencia, podría decirse que el HMM aprendiz está 'aprendiendo' los parámetros del HMM dado (o por lo menos algunos parámetros compatibles). Un parámetro p se actualiza usando un nuevo valor v y un parámetro η segun la fórmula:

$$p_{new} = p_{old} + \eta \cdot (v - p_{old}) \tag{5.30}$$

Así cuando $\eta = 1$, p_{old} es completamente reemplazado por v, y si $\eta = 0$, no hay ningún cambio. El proceso de aprendizaje se repite por un cierto número de iteraciones i, entonces la probabilidad promedio $P(O_j \mid \lambda_{learn})$ se evalua sobre una serie de 1000 observaciones.

Probabilidades usando Baum-Welch

η / i	1	2	$\cdots$	10
.1	.001723	.000886		.000811
.2	.000983	.001459		.000924
.3	.001085	.000286		.000996
.4	.000394	.000893		(diverge)
.5	.000853	.000487		(diverge)
.6	.00127	.001128		(diverge)
.7	.000268	(diverge)		(diverge)
.8	.000096	(diverge)		(diverge)
.9	.00081	(diverge)		(diverge)

Como puede verse de la tabla, las probabilidades promedio observadas no se acercan al valor de referencia. Con valores crecientes para η y para i, el algoritmo tiende a divergir. El mismo espacio de parámetros se muestra en la figura 5.3, donde la divergencia es marcada por el cero.

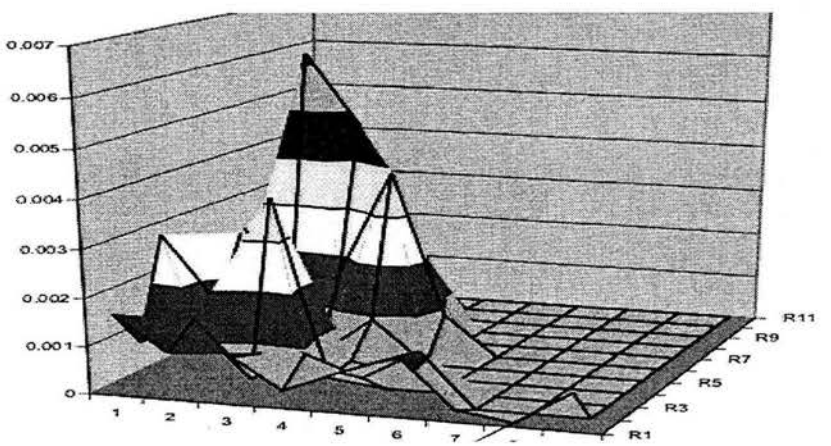

Figura 5.3: Espacio de parametros de Baum-Welch

Por lo anterior, el entrenamiento de HMMs que usa el algoritmo de Baum-Welch se abandonó y un procedimiento de entrenamiento basado en el algoritmo de Viterbi se seleccionó a cambio.

5.8 Evaluando el entrenamiento con el algoritmo de Viterbi

El procedimiento que utiliza el algoritmo de Viterbi para entrenar un HMM es como sigue: el arreglo de HMM dado y HMM aprendiz es como en el caso anterior. Ahora para cada secuencia de observaciones O_j, el algoritmo de Viterbi se usa para determinar el camino mejor a través del HMM aprendiz, dado su modelo actual $\lambda_{learn} = (A_l, B_l, \pi_l)$. En un contador de frecuencia son guardados el estado inicial determinado, todas las transiciones y los símbolos observados en cada estado. Esto se repite para las i secuencias de observación. Ahora las nuevas probabilidades v son calculados de las frecuencias observadas f por la fórmula:

$$v \propto \frac{f}{i} \tag{5.31}$$

Entonces los parámetros del HMM aprendiz se actualizan según la fórmula 5.30, dependiendo como antes del valor de η. La variación sistemática de los valores i entre 1 y 30, así como η entre .1 y .9 resulta en la figura 5.4. Es ventajoso seleccionar un número pequeño de iteraciones y un valor de η mayor que .4. Las probabilidades promedio, más allá de este umbral para η, generalmente se acercan al valor de referencia .004516, incluso a veces superándolo. Esto significa que el HMM aprendiz está aprendiendo un conjunto adecuado de parámetros, que resulta en probabilidades promedio $P(O_j \mid \lambda_{learn})$ altas. Uno de los aspectos más importantes de usar el algoritmo de Viterbi es que, a diferencia del procedimiento de Baum-Welch, nunca diverge.

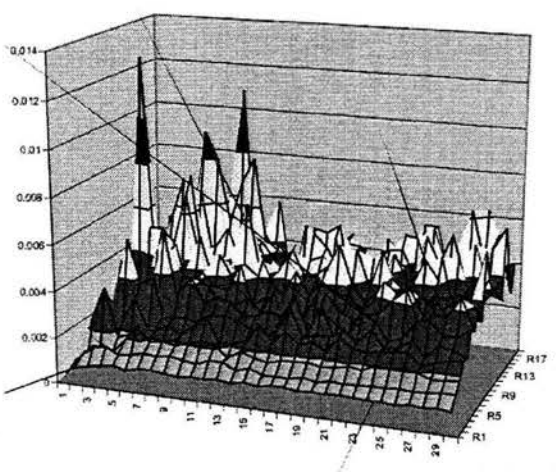

Figura 5.4: Espacio de parametros de Viterbi

5.9 La gramática de movimientos

Es más fácil de clasificar los movimientos en el contexto que en el aislamiento. Por consiguiente, una 'gramática de movimiento' humano se impone en la secuencia de movimientos observados. La gramática captura las probabilidades de un movimiento que sigue a otro, así como prohibiendo algunas secuencias, como ir de un estado 'sentado' a un estado 'caminando' sin atravesar el estado 'levantándose'. En esta aplicación, la gramática se lleva a cabo como un meta-HMM, que es un HMM cuyos estados son representados por otros HMMs.

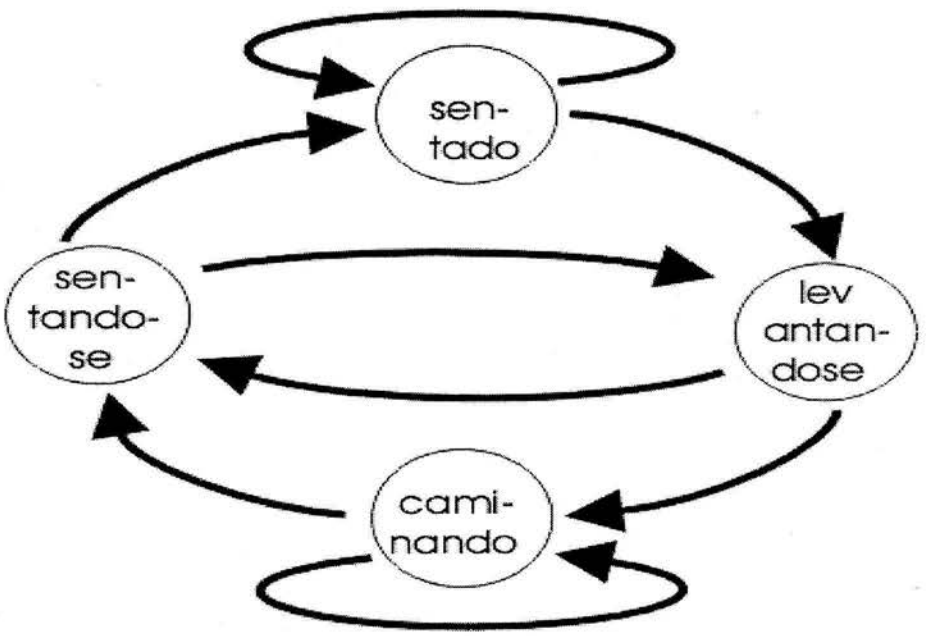

Figura 5.5: El HMM de gramática

La gramática prohibe las transiciones del estado 'caminando' al 'sentado' y viceversa sin ir a través del estado 'sentándose' y 'levantándose', respectivamente.

Capítulo 6

Sistema completo

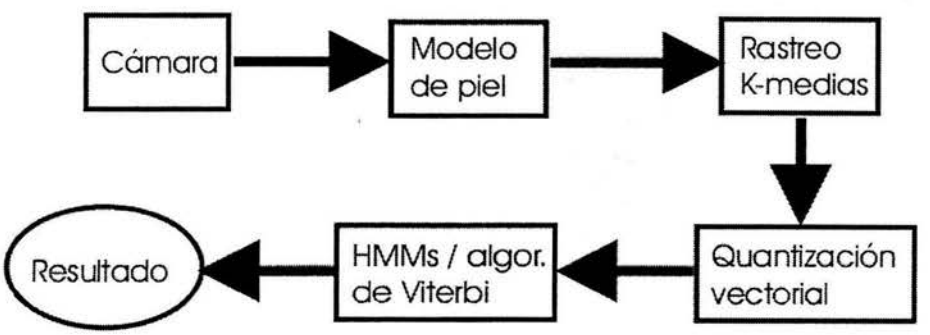

Figura 6.1: Diagrama de bloques del sistema completo

La capturadora de video convierte la imagen de la cámara en un bitmap RGB con una resolución de $768 \cdot 480$ pixeles y 8 bits de resolución de color. Así, cada pixel representa tres bytes de datos. El filtro de máscara filtra ciertos pixeles, por ejemplo los artefactos de la cámara. Los pixeles restantes se evalúan con respecto al modelo de piel. Se agrupan esos pixeles clasificados como piel usando el algoritmo de K-medias.

La figura 6.2 muestra ocho pixeles clasificados como piel, así como dos cruces que marcan los centroides de dos objetos en el cuadro anterior. Cada pixel se asigna al centroide más cerca de él. Entonces la nueva posición de un centroide es calculada como la posición media de todos los pixeles asignados a él. Para cada centroide, la diferencia entre la posición vieja y la nueva define un vector de movimiento; en este ejemplo, sería $(-3, 0)$ en ambos casos.

El sistema rastrea tres objetos, a saber la cabeza y dos manos, en tiempo real. Así, la diferencia entre dos cuadros da lugar a un conjunto de tres vectores

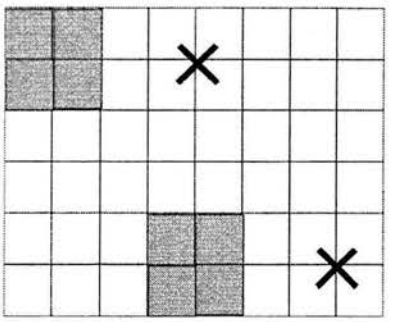

Figura 6.2: Ejemplo de K-medias

bidimensionales. Este conjunto se llama 'el vector de movimiento' por simplicidad. Para facilitar la evaluación del vector de movimiento, su complejidad será reducida por la cuantización vectorial. De cada vector bidimensional original, sólo la dirección se retiene y se cuantiza en una de las ocho direcciones principales del compás (como el norte, norte-este, este etc.). La magnitud sólo se usa para un umbral: debajo de un cierto límite, el vector se pone a cero. Así con un umbral de dos, el vector de movimiento $((4,0), (-3,-3), (1,1))$ es cuantizado en el codeword (este, sur-oeste, cero).

Los codewords realmente son enteros de tres digitos. En el ejemplo de la figura 6.3, la persona esta moviendo su mano derecha a la derecha y su mano izquierda a la izquierda, mientras la cabeza no mueve. Utilizando el esquema de codificación (ver figura 5.2), resulta la tupla $< 6, 0, 3 >$ para $<$ *mano izquierda*, *cabeza*, *mano derecha* $>$ y entonces el codeword 603.

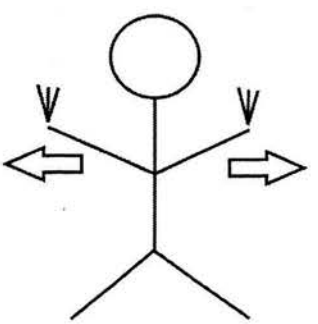

Figura 6.3: Persona saludando con dos manos

En el ejemplo siguiente asumimos por simplicidad que un codeword sólo puede ser uno del conjunto:

$$\{0, 1, 2, 3, 4, 5\} \tag{6.1}$$

donde 0 significa 'no movimiento', 1 significa 'la mano izquierda a la izquierda', y 3 significa 'la mano izquierda a la derecha'. El cuantizador vectorial produce la secuencia:

$$\text{"}0, 1, 3, 1, 3\text{"} \tag{6.2}$$

y queremos calcular la probabilidad de que esta secuencia fue producida por el HMM 'left_wave' que se muestra abajo:

HMM: left_wave

probabilidades iniciales:

estado	1	2	3	4	5
probabilidad	.9	.25	.25	.25	.25

probabilidades de transición :

desde / hasta	1	2	3	4	5
1	.1	.5	.2	.1	.1
2	.1	.1	.5	.2	.1
3	.1	.1	.1	.5	.2
4	.2	.1	.1	.1	.5
5	.5	.2	.1	.1	.1

probabilidades de símbolos:

estado / símbolo	'0'	'1'	'2'	'3'	'4'	'5'
1	.9	.1	0	0	0	0
2	0	.9	.1	0	0	0
3	0	0	.9	.1	0	0
4	0	0	0	.9	.1	0
5	0	0	0	0	.9	.1

Ahora seguimos los pasos del algoritmo de Viterbi:

$$\delta_1 = [.81, 0, 0, 0, 0] \tag{6.3}$$

$$\psi_1 = [0, 0, 0, 0, 0] \tag{6.4}$$

$$\delta_2 = [.0081, .3645, 0, 0, 0] \tag{6.5}$$

$$\psi_2 = [1, 1, 1, 1, 1] \tag{6.6}$$

$$\delta_3 = [0, 0, .018225, .06561, 0] \tag{6.7}$$

$$\psi_3 = [2, 2, 2, 2, 2] \tag{6.8}$$

$$\delta_4 = [.0013122, .0059049, 0, 0, 0] \tag{6.9}$$

$$\psi_4 = [4, 4, 4, 4, 4] \tag{6.10}$$

$$\delta_5 = [0, 0, .000295245, .001062882, 0] \tag{6.11}$$

$$\psi_5 = [2, 2, 2, 2, 2] \tag{6.12}$$

$$P^* = .001062882 \tag{6.13}$$

$$q_T^* = 4 \tag{6.14}$$

$$q^* = \{1, 2, 4, 2, 4\} \tag{6.15}$$

Así, la probabilidad que la secuencia "$0, 1, 3, 1, 3$"
se produjera por este HMM es:

$$P^* = .001062882 \tag{6.16}$$

y la secuencia de estados más probable es:

$$q^* = \{1, 2, 4, 2, 4\} \tag{6.17}$$

Capítulo 7

Implementación

El reconocimiento de gestos clave es distribuído en tres programas independientes que se comunican a través del mecanismo 'pipe' de Linux. Todos se llevan a cabo en el lenguaje *C* estándar que se compila bajo *Debian Linux Versión* 3.0. Ellos hacen uso de la biblioteca *my_grab* para la adquisición de la imagen por captura de video y la biblioteca de *Imlib* para el manejo de imagenes en general.

Una manera típica de ejecutar el procedimiento es la llamada siguiente:

$$gest \mid hmm \mid gram \tag{7.1}$$

donde *gest* lleva a cabo el rastreo de movimientos vía algoritmo de K-medias y cuantización vectorial, *hmm* lleva a cabo los HMMs del nivel de gestos y *gram* lleva a cabo el meta-HMM de la gramática conductual.

7.1 Rastreando movimientos

La principal estructura de datos en el programa de *gest* que logra el rastreo de movimiento se llama *g_object* y contiene la información sobre un objeto rastreado, como la posición, el área, la variación bidimensional y si el objeto es considerado válido o no:

```
long int xpos;
long int ypos;
long int area;
long int xsum;
long int ysum;
long int xvar;
long int yvar;
long int valid;
```

Los procedimientos principales son el filtro rapido, que filtra la imagen y encuentra los pixeles de piel, K-medias, que agrupa los pixeles de piel en tres

objetos empleando el algoritmo de los K-medias y la cuantización vectorial de los movimientos calculados. El programa asume la orientación de la cámara normal. Poniendo la bandera de compilador *vertical*, se supone que la cámara se ha girado 90 grados a la izquierda alrededor de su eje óptico, produciendo un formato de *'portrait'* en lugar del *'landscape'* usual.

7.2 HMMs de gestos

La principal estructura de datos en el programa *hmm* se llama *hmm_adt* e implementa el tipo de datos abstracto del 'modelo oculto de Markov':

```
char name[name_length];
prec init_prob[num_states];
prec trans_prob[num_states][num_states];
prec sym_prob[num_states][num_symbols];
prec init_prob_log[num_states];
prec trans_prob_log[num_states][num_states];
prec sym_prob_log[num_states][num_symbols];
```

El tipo *prec* simplemente es un alias para *long double*. Para aumentar la estabilidad numérica se usa una representación logarítmica de probabilidades siempre y cuando sea posible, particularmente para la multiplicación. Como la representación logarítmica no permite la suma directamente, hay conversiones frecuentes entre el formato logarítmico y normal. Como el logaritmo de cero es indefinido, la constante *almost_zero* se usa en lugar del cero.

Los procedimientos principales son el algoritmo de Viterbi, que calcula la probabilidad del camino más probable a través de un HMM y el camino mismo, este tambien se usa para entrenar HMMs. Hay algunos procedimientos auxiliares para poner un HMM a probabilidades iguales, para poner las probabilidades iguales para todos los parámetros excepto algunos y algunas rutinas que ayudan en la definición de HMMs grandes con muchos centenares de símbolos en una notación densa.

7.3 HMM de gramática

La principal estructura de datos en el programa *gram* se llama *m_hmm_adt* e implementa el tipo de datos abstracto del 'modelo oculto de Markov'. La diferencia principal con el *hmm_adt* anterior es que ahora los estados tienen nombres. Mientras en la actualidad sólo un meta-HMM está definido, podría haber varios en el futuro, cada uno representando una intención diferente. El meta-HMM que obtenga la mejor probabilidad se usaría para inferir la intención del usuario.

Capítulo 8

Evaluación y Conclusión

8.1 Experimentos y Resultados

Para evaluar el sistema, se utilizaron 8 HMMs con 6 estados, calibrados empiricamente, y secuencias de 4 observaciones. El primer HMM (1) absorbe todos los movimientos no considerados 'clave', los HMMs 2 hasta 8 reconocen los siguientes movimientos: 'saludar con la izquierda' (2), 'saludar con la derecha' (3), 'aplaudir' (4), 'levantarse' (5), 'sentarse' (6), 'caminar a la izquierda' (7), 'caminar a la derecha' (8). Estos ocho HMMs procesan una secuencia de cuatro codewords en paralelo (ver figura 8.1). El movimiento se clasifica segun el HMM con la probabilidad mas alta, calculada usando el algoritmo de Viterbi, el cual encuentra el mejor camino.

La cámara se puso encima del monitor y enfrente del usuario. Cada uno de los gestos clave se probó 20 veces. La matriz de confusión siguiente muestra la relación numérica entre movimientos reales (accion) y reconocidos (reconocido), utilizando el mismo esquema como arriba: 'movimiento ajeno' (1) hasta 'caminar a la derecha' (8). Por ejemplo, la accion 'saludar con la izquierda' (2) se reconoció correctamente 18 veces y 2 veces como 'caminar a la izquierda' (7).

matriz de confusión:

acción / reconocido	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
(1)	*	0	0	0	0	0	0	0
(2)	0	18	0	0	0	0	2	0
(3)	0	0	19	0	0	0	0	1
(4)	0	1	1	18	0	0	0	0
(5)	2	0	0	0	18	0	0	0
(6)	1	0	0	0	0	19	0	0
(7)	0	1	0	0	0	0	19	0
(8)	0	0	1	0	0	0	0	19

Así la precisión global está en el orden del 93%. Se puede distinguir la probabilidad promedio $P_{rec}(reconocer\ gesto\ X \mid acción\ gesto\ X) = 93\%$, que es la probabilidad de reconocer correctamente un dado gesto X, y la probabilidad promedio $P_{acc}(acción\ gesto\ X \mid reconocer\ gesto\ X) = 96\%$, que es la probabilidad de que, dado que se reconoció el gesto X, el usuario realmente hizo el gesto X. La gramática de movimientos puede corregir errores de clasificación en ciertas situaciones, por ejemplo cuando la persona esta sentada, pero se reconoció que la persona caminó. La gramática no se utilizó en esta prueba.

8.2 Conclusión

Se ha descrito un sistema para el reconocimiento de gestos humanos importantes, usando el algoritmo de K-medias para rastrear la cuantización vectorial de movimientos, y los modelos ocultos de Markov con el algoritmo de Viterbi para analizar los codewords de movimiento.

Se logro el objetivo, el reconocimiento de un número de gestos clave, utilizando sólo una cámara de video.

El sistema reconoce un conjunto restringido de movimientos humanos como levantarse, sentarse, caminar, saludar o aplaudir, con una exactitud razonable. Como un mecanismo de aprendizaje esta incluido en el sistema, la exactitud puede mejorarse entrenando más al sistema.

Mientras en la aplicación actual se analizan sólo los movimientos en dos dimensiones, las bases para extenderla a tres dimensiones ya están presentes.

8.3 Perspectiva

La motivación para este proyecto es un robot móvil que pueda observar a un ser humano, deducir sus intenciones y ofrecer ayuda para lograr sus metas en el momento y de forma apropiada.

En lugar de un sólo HMM de gramática general como en el sistema actual, uno podría imaginar varios que representaran casos diferentes de conducta intencional. Encontrando la que tenga mejor coincidencia entre esas gramáticas, el robot podría entender lo que el humano está intentando lograr.

Para ayudar el robot en su tarea, el sistema actual podría acoplarse con un sistema de reconociminento de voz, y podrían integrarse los resultados de ambos sistemas para reducir ambigüedad y la tasa de errores. Otros modos de entrada podrían usarse, como los guantes de datos, para darse cuenta de manera más exacta de los movimientos de los dedos, como en el caso de apuntar.

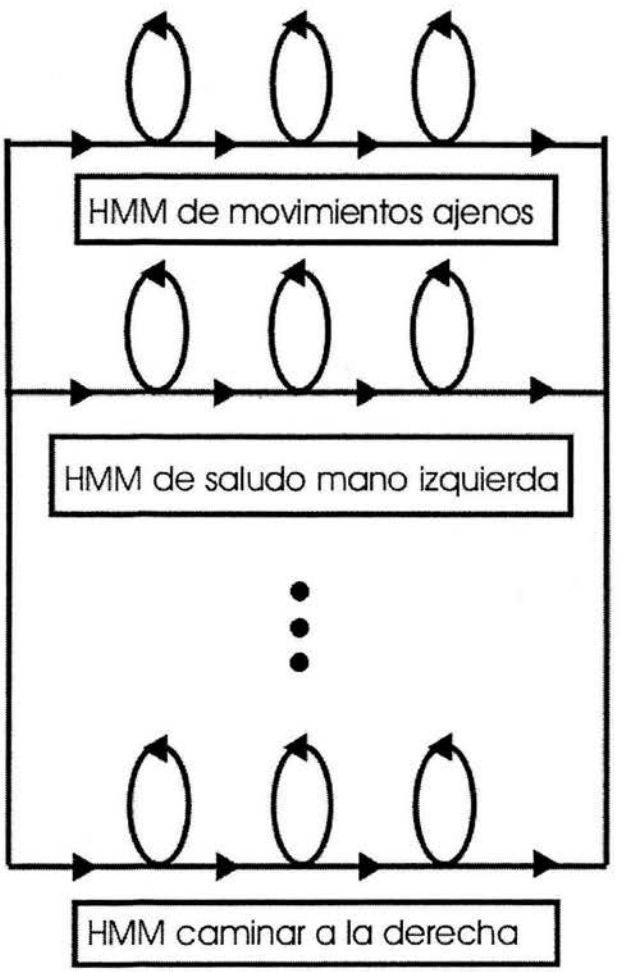

Figura 8.1: Ocho HMMs en paralelo

Appendice A

Glosario

Baum-Welch algorithm - algoritmo de Baum-Welch - algoritmo para entrenar HMMs

centroid - centroide - centro de gravedad de un objeto

codebook - libro de códigos - conjunto de códigos en cuantización vectorial

codeword - palabra de código - un código que representa a varios datos en cuantización vectorial

comb effect - efecto de peine - efecto indeseable que resulta del estandar de televisión y introduce artefactos de movimiento

frame grabber - tarjeta de captura - tarjeta que convierte señales analogicas de la cámara en información digital

frame rate - tasa de cuadros - numero de cuadros por segundo

Hidden Markov Model (HMM) - modelo oculto de Markov - modelo matemático para modelar procesos estocasticos

K-means algorithm - algoritmo de K-medias - algoritmo para agrupar objetos multidimensionales

optical flow algorithm - algoritmo de flujo óptico - algoritmo para detectar movimientos en secuencias de imágenes

Red-Green-Blue color model (RGB) - modelo de color RGB - modelo de color que consiste en los tres colores rojo, verde y azul

Viterbi algorithm - algoritmo de Viterbi - algoritmo para determinar la secuencia de estados mas probable en un HMM

Vector Quantization (VQ) - cuantización vectorial - método para representar un numero de objetos con un numero menor de códigos

Bibliografía

[1] Barron, J.L., Fleet, D.L. & Beauchemin, S.S. (1994), "Performance of Optical Flow Techniques", Int. Journal of Computer Vision 12:1, pp. 43-77

[2] Baumberg, A.M. & Hogg, D.C. (1993), "Learning Flexible Models from Image Sequences", Report 93.36, School of CS, University of Leeds

[3] Cutler, R. & Turk, M. (1998), "View-based Interpretation of Real-time Optical Flow for Gesture Recognition", Third IEEE International Conference on Automatic Face and Gesture Recognition, Nara, Japan

[4] Forsyth, D. A. & Ponce, J. (2003), "Computer Vision. A Modern Approach", Prentice Hall, New Jersey

[5] Heap, T. & Samaria, F. (1995) "Real-Time Hand Tracking and Gesture Recognitino Using Smart Snakes", Olivetti Research Limited

[6] Huang, T. S. & Pavlovic, V. I. (1995), "Hand Gesture Modeling, Analysis, and Synthesis", University of Illinois

[7] Jain, A. K. (1989), "Fundamentals of Digital Image Processing", Prentice Hall, New Jersey

[8] Liu, N. & Lovell, B. C. (2003), "Gesture Classification Using Hidden Markov Models and Viterbi Path Counting", Proceedings of the VIIth Conference on Digital Image Computing: Technique and Applications, Sydney

[9] McKenna, S. J. & Gong, S. (1998), "Gesture Recognition for Visually Mediated Interaction using Probabilistic Event Trajectories", University of Dundee / Westfield College

[10] Ong, E.-J., McKenna, S. & Gong, S. (1998), "Tracking Head Pose for Inferring Intention", University of Dundee / Westfield College

[11] Rabiner, L. & Juang, B.-H. (1993), "Fundamentals of Speech Recognition", Prentice Hall, New Jersey

[12] Starner, T. & Pentland, A. (1995) "Visual Recognition of American Sign Language using Hidden Markov Models", MIT Media Lab, MIT