UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

RECONOCIMIENTO ROBUSTO Y EN TIEMPO REAL DE GESTOS

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL EN COMPUTACIÓN

JONG BOR LEE FERNG

PROFESOR GUÍA:
SR. JAVIER RUIZ DEL SOLAR

MIEMBROS DE LA COMISIÓN:
SRA. NANCY HITSCHFELD KAHLER
SR. NELSON BALOIAN TATARYAN

SANTIAGO DE CHILE
ENERO 2010

RESUMEN DE LA MEMORIA
PARA OPTAR AL TITULO DE
INGENIERO CIVIL EN COMPUTACIÓN
POR: JONG BOR LEE FERNG
FECHA: 21/01/2010
PROF. GUIA: Sr. JAVIER RUIZ DEL SOLAR

"RECONOCIMIENTO ROBUSTO Y EN TIEMPO REAL DE GESTOS"

El presente trabajo tiene como fin el desarrollo de un sistema de reconocimiento de gestos dinámicos aplicable a interfaces humano-computador. El reconocimiento es puramente visual, con imágenes obtenidas de una cámara web convencional. Un sistema de visión computacional previamente desarrollado permite extraer la posición de la cara y las manos.

El trabajo se desglosa en dos objetivos. El primer objetivo es revisar y mejorar dicho sistema de visión computacional para alcanzar una mayor robustez en el *tracking* (seguimiento) de los movimientos corporales del usuario. El segundo objetivo consiste en proponer y estudiar un método de reconocimiento de gestos dinámicos caracterizados por la trayectoria de la mano.

Para abordar el primer objetivo, se ponen a prueba varias modificaciones del sistema de tracking actual: restricción de la zona de tracking a la zona con piel y movimiento, extracción de nuevas características del objeto seguido, actualización en línea de dichas características, y búsqueda del objeto seguido a partir de múltiples puntos iniciales en la imagen.

En cuanto al segundo objetivo, se observa que el reconocimiento de gestos dinámicos es un problema de reconocimiento de patrones espacio-temporales. Como tal, está sujeto a las variaciones de velocidad y geometría con que los usuarios realizan los gestos, lo cual hace necesario el uso de métodos estadísticos que den cuenta de tales variaciones. En este trabajo, se propone un método novedoso que evita el análisis temporal explícito, al reducir la secuencia de manos detectadas a estadísticas puramente geométricas. Estas estadísticas son suministradas continuamente a un conjunto de clasificadores Naïve Bayes, que entregan probabilidades de ejecución de cada gesto conocido por el sistema; al encontrarse un máximo local en dicha probabilidad, se declara la posible presencia de un gesto, el cual es validado mediante la comparación con plantillas. Se usan algunas reglas sencillas para abordar el problema de subgestos (reconocimiento errado de un gesto en vez de otro que lo contiene) y se aplican heurísticas como examinar la velocidad de la mano para determinar si el usuario desea dar por terminado el gesto o no.

Los resultados de los experimentos muestran una mejoría pequeña del rendimiento del tracking gracias a los cambios introducidos. En tanto, el sistema obtiene una tasa de reconocimiento del $81,7\,\%$ al reconocer dígitos dibujados con el puño en el aire, desempeño que podría ser mejorado desarrollando un mejor sistema de tracking de la mano y usando métodos geométricos más sofisticados que los propuestos en este trabajo. La velocidad de funcionamiento es de 6 a 7 cuadros por segundo, suficiente para su uso en tiempo real. El sistema también ha sido puesto a prueba con el robot de servicio Bender, con gestos diseñados especialmente para controlarlo, obteniendo una tasa de reconocimiento de $78,5\,\%$.

Agradecimientos

Agradezco a mi profesor guía Javier Ruiz del Solar, por su dedicación para la elaboración de este trabajo y por las oportunidades dadas.

Agradezco también a todos quienes me han acompañado en estos años en la universidad, y en especial, durante el último año, a mis compañeros en el laboratorio de Robótica.

Finalmente, agradezco a mi familia y le dedico con cariño esta obra.

Índice General

Índice de figuras

Capítulo 1

Introducción

La robótica se ocupa del diseño y construcción de robots, que son agentes artificiales capaces de realizar tareas desenvolviéndose con autonomía en el ambiente en que se encuentran.

Las aplicaciones más conocidas y exitosas de la robótica en la actualidad se encuentran en la industria manufacturera, donde los robots cumplen tareas tales como soldado, pintura, ensamblaje, empaque, inspección y prueba de piezas; un ejemplo notable es la industria automotriz. En contraste, una línea de investigación más reciente y menos madura es la de los robots sociales, capaces de sostener una interacción natural con personas. Esto incluye la percepción de la presencia de humanos, el respeto a su espacio personal, la capacidad de navegar en ambientes caseros, la capacidad de reconocer y observar convenciones sociales, y la capacidad de recibir y ejecutar órdenes.

En este contexto, la interacción del robot es con usuarios no necesariamente especializados, por lo que es preciso contar con un medio de comunicación natural e intuitivo. Así, se concibe la posibilidad de comunicarse con el robot mediante gestos. Un gesto es una forma de comunicación no verbal en que se usan movimientos corporales para transmitir un mensaje, en oposición a la comunicación verbal, basada en el habla. Los gestos son una parte primordial de la comunicación no verbal entre humanos. En algunos casos, es más natural usar gestos que comunicarse únicamente mediante el habla, por ejemplo, al transmitir información espacial. En ambientes ruidosos o donde no se pueden transmitir sonidos, los gestos son la única opción para comunicarse; ejemplo de esto es la comunicación de señales de tránsito y el lenguaje de señas.

Un gesto forma parte de una intención comunicativa que engloba varias formas de expresión, incluyendo el habla y las expresiones faciales. Puede observarse que las personas gesticulan incluso cuando no pueden ser vistas por su interlocutor (por ejemplo, al hablar por teléfono). Sin embargo, en este trabajo, se pondrá el foco únicamente en el reconocimiento de gestos, sin considerar la totalidad del proceso comunicativo.

Se hará la distinción entre gestos estáticos y gestos dinámicos. Los gestos estáticos se caracterizan por una pose de una parte del cuerpo; por ejemplo, empuñar la mano o hacer el signo de la paz. Por otro lado, los gestos dinámicos son aquellos que se caracterizan por el movimiento de una parte del cuerpo. Ejemplos de gestos dinámicos son mover el brazo en

señal de despedida y hacer una reverencia.

En este trabajo, consideraremos un tipo de gesto dinámico más limitado, que se caracteriza por la trayectoria espaciotemporal de la mano. Por ejemplo, una persona puede describir, con su mano, la forma de los números del 0 al 9 en el aire. Para reconocer este tipo de gestos, es necesario obtener información sobre la posición de la mano a través del tiempo. Esto implica, en primer lugar, detectar la presencia de la mano en una imagen y, en segundo lugar, hacer seguimiento o tracking[1] de la posición de la mano a medida que ésta se mueve. Un sistema de reconocimiento de gestos debe analizar las sucesivas coordenadas de la mano para encontrar si efectivamente se trata de un gesto y, en tal caso, a qué gesto (de un conjunto predefinido) corresponde el movimiento del usuario.

Esta memoria describe un sistema de reconocimiento de gestos dinámicos basado en imágenes. En primer lugar, se describe un sistema de visión computacional desarrollado en trabajos pasados, cuyo objetivo es obtener la posición de la cabeza y las manos del usuario. Se exponen las mejoras hechas a este sistema con miras al objetivo de reconocer gestos dinámicos. En segundo lugar, se describe el sistema de reconocimiento de gestos, que basa su funcionamiento en métodos estadísticos (aprendizaje de máquinas) y de reconocimiento de patrones. Este sistema es capaz de segmentar los gestos (extraer su cuadro inicial y final) y clasificarlos en línea. Finalmente, se exponen resultados experimentales y una discusión de éstos.

1.1. Objetivos

1.1.1. Objetivos generales

El objetivo de la memoria es mejorar y extender un sistema de reconocimiento de gestos dinámicos basado en un sistema de visión computacional que permite extraer, con suficiente precisión, datos acerca del individuo que se encuentra a la vista de la cámara (zonas de piel, posición de manos y cara, etc.).

Se trata de un sistema que se encuentra parcialmente desarrollado (el alumno trabajó en su desarrollo entre diciembre del 2008 y enero del 2009, con resultados documentados en dos publicaciones [23, 24]).

Las mejoras y extensiones deben permitir al sistema funcionar de manera robusta y en tiempo real.

1.1.2. Objetivos específicos

1. Estudio y modificación del sistema de visión computacional actual [31].

    a) Estudio de componentes de detección de cara, piel, manos y fondo.

[1] En este trabajo, se usará el término "tracking".

b) Organización (reescritura) del código. El sistema había sido concebido modularmente, pero esto no se reflejaba en el código. Lograr facilidad de comprensión del código.

c) Optimización. Se pretende hacer funcionar el sistema en tiempo real.

d) Mejora en la efectividad. Es importante lograr que la detección y tracking de cara y manos funcionen bien, pues la detección de gestos dinámicos se basa en el buen funcionamiento de estos módulos.

2. Revisión y mejora del módulo de reconocimiento de gestos dinámicos.

   a) Documentación de la estructura del reconocedor de gestos.

   b) Mejora de la efectividad del reconocedor de gestos.

1.2. Justificación

La comunicación mediante gestos dinámicos constituye una opción importante dentro de las posibilidades de interacción humano-computador debido a que introduce mayor naturalidad en dicha interacción, considerando que el lenguaje corporal forma una parte importante de la comunicación humana. Varios trabajos realizados en esta área exigen al usuario llevar equipamiento especial para la adquisición de datos, como guantes, que pueden ser costosos y además incomodan al usuario. Por lo tanto, los métodos basados únicamente en visión suponen un avance significativo tanto en términos de costo como en facilidad de uso. Sin embargo, el uso de visión implica dificultades de obtención y manejo de datos, los cuales resultan más imprecisos que los entregados por los equipos especializados. Por esto, el reconocimiento robusto de gestos dinámicos, puramente visual y que sea capaz de sobreponerse a estas imprecisiones, es un área en desarrollo al cual esta memoria hace una contribución.

El método propuesto es novedoso dentro de la literatura, por lo que no se conocen bien sus ventajas y limitaciones, de modo que esta memoria debería servir como punto de partida para la comprensión de estos aspectos. Concretamente, buena parte de los métodos que se encuentran en la literatura intentan hacer corresponder las secuencias de datos obtenidos con modelos de estados. Como un gesto puede ser ejecutado con velocidad y énfasis variables, al hacer la correspondencia, se permite saltar o repetir un estado. En cambio, el método que se desarrolla en esta memoria intenta condensar una secuencia de datos en un conjunto de atributos geométricos (un vector de características); este vector es entregado a un clasificador, el cual encuentra el gesto más probable dados dichos atributos. Este enfoque convierte al método en un miembro de los llamados "métodos basados en apariencia". Este tipo de método es inusual en el reconocimiento de gestos basados solamente en trayectorias, por lo que hace falta investigar con más profundidad los alcances de tal manera de procesar los datos.

El trabajo desarrollado será usado en el robot Bender [1], desarrollado en el Laboratorio de Robótica del departamento de Ingeniería Eléctrica, que es un proyecto pionero en Latinoamérica en el área de robots de servicio. Este robot ha servido como plataforma para desarrollar y probar algoritmos de varios tipos, como detección y reconocimiento de caras, detección de objetos, control de brazos robóticos y navegación. Bender ha participado en las

RoboCup desde el año 2007, siendo de los pocos participantes latinoamericanos, y ha ganado en dos ocasiones el premio a la innovación [21].

Capítulo 2

Antecedentes

2.1. Aprendizaje de máquinas

El aprendizaje de máquinas busca diseñar algoritmos capaces de mejorar su rendimiento en base a la experiencia [43]. Esta área de estudio encuentra aplicación en problemas donde los modelos existentes son deficientes, por lo que no es posible implementar algoritmos especializados, y donde sin embargo hay abundancia de datos reales [4].

Es común encontrar tareas que son fáciles para los seres humanos, pero que son difíciles de explicar y por lo tanto no tienen una solución algorítmica eficaz. La comprensión del lenguaje hablado, el reconocimiento de la letra manuscrita y la identificación de caras se cuentan entre estas habilidades, así como el reconocimiento de gestos; en todos estos problemas, el costo de obtener datos de instancias reales es bajo, por lo que el aprendizaje de máquinas aparece como opción viable.

Los algoritmos de **aprendizaje supervisado** tienen como fin construir una función $f : X \to Y$ a partir de un conjunto de entradas y salidas conocidas (x, y), llamado conjunto de entrenamiento. Cuando f es una función continua, se trata de un problema de regresión; en cambio, si f es una función discreta, se trata de un problema de clasificación.

En un problema de clasificación, los elementos del conjunto Y se interpretan como clases o categorías a las que pertenecen los elementos del conjunto X. Por lo tanto, la función f se llama un clasificador. El reconocimiento de gestos puede modelarse como un problema de clasificación en que el conjunto de entrenamiento consta de pares (x, y), donde x es un gesto (representado por un conjunto de atributos numéricos) e y la clase del gesto.

2.1.1. Clasificadores Naïve Bayes

Los clasificadores Naïve Bayes usan el teorema de Bayes para deducir la probabilidad de que un objeto pertenezca a una clase; la regla de clasificación consiste en asignar a cada objeto la clase cuya probabilidad a posteriori sea máxima. Esta idea será explicada en detalle a continuación.

Según el teorema de Bayes, si A es un evento de probabilidad $P(A)$ y se cuenta con la información adicional de que ha ocurrido un evento B, es posible mejorar la estimación de la probabilidad de A según la regla

$$P(A|B) = \frac{P(A)P(B|A)}{P(B)}$$

$P(A|B)$ es llamada la probabilidad a posteriori. $P(A)$ es la probabilidad a priori, $P(B|A)$ es la verosimilitud, y $P(B)$ es la evidencia.

En un problema de clasificación, el objetivo es determinar si un objeto x pertenece a una clase c. Como punto de partida, se toma la probabilidad de que un objeto cualquiera pertenezca a la clase c, $P(C = c)$. Esta es la estimación a priori, que no depende de x, y es la mejor que se puede hacer sin tomar en cuenta al objeto mismo. Usando el teorema de Bayes, dicha estimación puede ser mejorada tomando en cuenta a x:

$$P(C = c|X = x) = \frac{P(C = c)P(X = x|C = c)}{P(X = x)} \tag{2.1}$$

Con este planteamiento, clasificar el objeto x consiste en elegir la clase c tal que $P(C = c|X = x)$ sea máximo. Para hacer factible el cálculo del lado derecho de (2.1) hace falta hacer algunas suposiciones.

Si X tiene una representación vectorial $X = (X_1, X_2, \ldots, X_n)$, el evento $X = x$ equivale a $X_1 = x_1 \wedge X_2 = x_2 \wedge \ldots \wedge X_n = x_n$, y por lo tanto

$$\begin{aligned} P(X = x|C = c) &= P(X_1 = x_1 \wedge X_2 = x_2 \wedge \ldots \wedge X_n = x_n|C = c) \\ &= \prod_i P(X_i = x_i|C = c) \end{aligned} \tag{2.2}$$

donde en la última ecuación se hace una suposición de independencia que permite descomponer la probabilidad del evento conjunto en un producto de probabilidades.

Esta suposición simplifica los cálculos, ya que las probabilidades $P(X_i = x_i|C = c)$ se pueden estimar a partir de los datos de entrenamiento. Si X_i es una variable discreta, $P(X_i = x_i|C = c)$ se estima como la frecuencia del valor $X_i = x_i$ entre los elementos de clase c, dividido por el total de elementos de clase c. En cambio, si X_i es una variable continua, se supone una distribución Gaussiana, con lo cual

$$P(X_i = x_i|C = c) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2.3}$$

donde μ y σ se estiman a partir de las instancias del conjunto de entrenamiento pertene-

cientes a la clase c, usando, por ejemplo, la media y varianza muestral.[1]

La cantidad $P(C = c)$ se estima como la cantidad de ejemplares de entrenamiento de clase c dividido por el total de ejemplares.

Finalmente, la regla de clasificación consiste en elegir la clase c que maximiza la cantidad

$$P(C = c|X = x) \propto P(C = c) \prod_i P(X_i = x_i|C = c) \tag{2.4}$$

donde el factor $P(X = x)$, por ser un factor de normalización que no cambia dado un x a clasificar, no es relevante para establecer el orden de estas cantidades, y por lo tanto se omite. Si fuera necesario obtener probabilidades en el intervalo $[0, 1]$, lo usual es calcular el lado derecho de (2.4) para todas las clases y al final normalizar dividiendo por la suma.

Al centro de Naïve Bayes se encuentra la suposición de independencia que permite escribir la ecuación (2.2). Esta suposición rara vez tiene sustento en la realidad, pues es difícil escoger variables X_i que no tengan correlaciones entre sí. Sin embargo, se ha demostrado que Naïve Bayes exhibe un buen rendimiento a pesar de lo fuerte de esta suposición [26].

2.2. Métodos y conceptos de visión computacional

2.2.1. Tracking y el algoritmo mean-shift

Dada una muestra $x_1, \ldots, x_n$ de una distribución de probabilidad f, es posible calcular una *estimación por kernel* (kernel density estimator) de f escogiendo una función de ponderación K_H y sumando las contribuciones de cada punto según la expresión

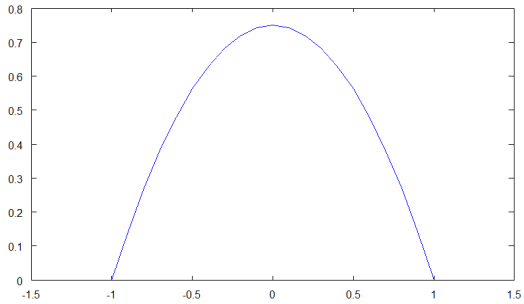
$$\hat{f}(x) = \frac{1}{n} \sum_{i=1}^{n} K_H(x - x_i) \tag{2.5}$$

La función K_H, llamada kernel, debe ser acotada y simétrica con respecto al origen. La figura 2.1 muestra algunos ejemplos de kernels unidimensionales.

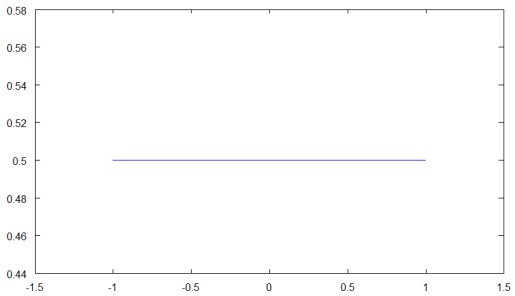
El algoritmo mean-shift, en su forma más general, es un método que permite encontrar los máximos de una función con la forma de $\hat{f}$. La búsqueda de estos máximos se realiza ascendiendo por el gradiente; la forma de la función $\hat{f}$ permite encontrar una fórmula iterativa eficiente para llevar a cabo este procedimiento. La descripción detallada del algoritmo y sus múltiples aplicaciones en visión computacional se encuentran [14].

En esta memoria interesa la aplicación del algoritmo mean-shift para el tracking o se-

[1] La fórmula (2.3), en rigor, no es correcta, pues la probabilidad de que una variable aleatoria sea igual a un número real dado es nula. Véase [36], donde se sigue una exposición similar a la de esta sección, para una justificación de esta ecuación.



(a) Kernel de Epanechnikov



(b) Kernel uniforme

Figura 2.1: Ejemplos de kernels.

guimiento de objetos. El problema del tracking consiste en extraer la posición de un objeto móvil a lo largo de un video. Yilmaz et al. han realizado un estudio general del tema en [70]. La aplicación de mean-shift para el tracking es objeto de estudio en [15]; el resto de esta sección ofrece una síntesis de los aspectos relevantes de dicho trabajo para el desarrollo de la memoria.

En el tracking con mean-shift, el objeto a trackear debe ser localizado en un cuadro inicial y representado mediante una función de densidad de probabilidad (fdp) $\hat{q}$. En el cuadro siguiente, una subimagen candidata a contener el objeto tendrá una posición y y estará caracterizada por una fdp $\hat{p}(y)$. En la práctica, las fdp tienen representaciones discretas en forma de histograma, de modo que:

$$\text{Modelo:} \qquad \hat{q} = \{\hat{q}_u\}_{u=1\ldots m} \qquad \sum_{u=1}^{m} \hat{q}_u = 1$$

$$\text{Candidato:} \quad \hat{p} = \{\hat{p}_u(y)\}_{u=1\ldots m} \quad \sum_{u=1}^{m} \hat{p}_u = 1$$

Las regiones candidatas en la imagen son normalizadas a un tamaño estándar y sus pixeles son ponderados por un kernel $k(x)$, de modo que los pixeles centrales tienen más importancia que los del borde; esto otorga mayor robustez, puesto que los pixeles del borde suelen estar afectados por interferencias del fondo y oclusiones.

Los histogramas $\hat{q}$ y $\hat{p}$ son conteos de una característica de los pixeles. Típicamente, esta característica es la tripleta de color RGB cuantizada, pero de ningún modo es ésta la única característica que puede usarse. En cualquier caso, si $b : R^2 \rightarrow \{1 \ldots m\}$ es la función que asocia al pixel en posición x_i su índice $b(x_i)$ en el histograma, se puede expresar el modelo $\hat{q}_u$ como

$$\hat{q}_u = C \sum_{i=1}^{n} k(\|x_i\|^2)\delta[b(x_i) - u] \tag{2.6}$$

donde δ es el delta de Kronecker y C resulta de imponer la condición $\sum_{u=1}^{m} \hat{q}_u = 1$. Del mismo modo, los candidatos tienen la expresión

$$\hat{p}_u(y) = C_h \sum_{i=1}^{n_h} k\left(\left\|\frac{y - x_i}{h}\right\|^2\right) \delta[b(x_i) - u] \tag{2.7}$$

donde el parámetro h, llamado ancho de banda, regula la importancia que el kernel da a los pixeles x_i centrados en y. C_h es un factor de normalización que resulta de imponer $\sum_{u=1}^{m} \hat{p}_u(y) = 1$.

El candidato idóneo es aquel que sea más similar al modelo; una función $\hat{\rho}(y) \equiv \rho[\hat{p}(y), \hat{q}]$ medirá esta similitud. Una elección posible para ρ es el coeficiente de Bhattacharyya

$$\hat{\rho}(y) = \rho[\hat{p}(y), \hat{q}] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(y)\hat{q}_u}$$

Geométricamente, este valor es el coseno del ángulo entre los vectores $(\sqrt{\hat{p}_1(y)}, \ldots, \sqrt{\hat{p}_m(y)})$ y $(\sqrt{\hat{q}_1}, \ldots, \sqrt{\hat{q}_m})$; cuanto mayor sea su valor, menor es el ángulo y mayor es la similitud entre los histogramas. Luego, el problema de elegir un candidato se reduce a encontrar el valor y^* que maximiza $\hat{\rho}(y)$. El desarrollo de Taylor de $\hat{\rho}(y)$ alrededor de $\hat{p}_u(\hat{y}_0)$ es

$$\rho[\hat{p}(y, \hat{q})] \approx \frac{1}{2} \sum_{u=1}^{m} \sqrt{\hat{p}(y_0)\hat{q}_u} + \frac{1}{2} \sum_{u=1}^{m} \hat{p}_u(y) \sqrt{\frac{\hat{q}_u}{\hat{p}(y_0)}}$$

que, tras manipular y reemplazar la expresión de la ecuación (2.7), entrega

$$\rho[\hat{p}(y, \hat{q})] \approx \frac{1}{2} \sum_{u=1}^{m} \sqrt{\hat{p}(y_0)\hat{q}_u} + \frac{C_h}{2} \sum_{i=1}^{n_h} w_i k\left(\left\|\frac{y - x_i}{h}\right\|^2\right) \tag{2.8}$$

esto es, una estimación por kernel k con pesos w_i dados por

$$w_i = \sum_{u=1}^{m} \sqrt{\frac{\hat{q}_u}{\hat{p}(y_0)}} \delta[b(x_i) - u]$$

Al asimilar el segundo término de (2.8) con (2.5), se puede aplicar mean-shift para encontrar un máximo de $\hat{\rho}(y)$, y por lo tanto para determinar la nueva posición y^* del objeto.

En síntesis, mean-shift es un algoritmo de ascenso por el gradiente que encuentra una aplicación en el tracking de objetos, puesto que es capaz de encontrar eficientemente un máximo en la función de similitud entre el objeto trackeado y los candidatos, los cuales son representados como histogramas.

2.2.2. Textura

El de textura es un concepto amplio, según puede verse en la discusión presentada por Tuceryan y Jain [58]. Dicho trabajo expone la idea de textura como una variación en la intensidad de una imagen que presenta un patrón reconocible. En esta memoria, interesa el uso de la textura como característica capaz de discriminar un objeto de interés —la mano— del resto de la imagen, de modo de facilitar el tracking.

En este contexto, se usan los *patrones binarios locales* (local binary patterns, LBP) introducidos por Ojala et al. en [46]. Dicha obra considera la textura como una propiedad de un pixel en relación a sus vecinos. Un pixel tiene 8 vecinos; al compararlos con el pixel central, se les asigna un valor igual a 1 si tienen una intensidad mayor y 0 de lo contrario. Estos dígitos son concatenados en un orden preestablecido, y el número binario resultante caracteriza la textura del pixel (figura 2.2).

Con esta definición, LBP adopta 256 valores diferentes. Si se rota una imagen, el valor de LBP de cada pixel cambiará; para remover este efecto, se establece la equivalencia entre los LBP que puedan ser obtenidos mediante rotaciones circulares de los bits, lo cual disminuye el número de valores distintos a 36 (ver figura 2.3).

Cada pixel en una imagen tendrá un valor LBP distinto. En consecuencia, es posible agregar la textura LBP a la tripleta de colores que caracteriza cada pixel, y estas cuatro características pueden ser consideradas en el histograma usado en el tracking por mean-shift (sección 2.2.1).

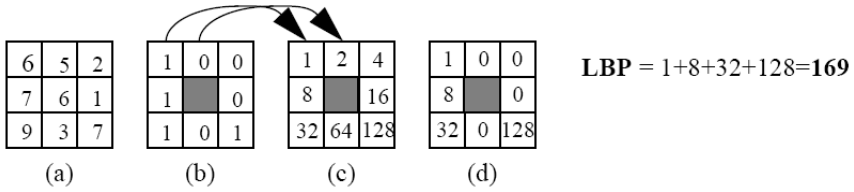


Figura 2.2: Cálculo de LBP. La figura (a) muestra la intensidad de un pixel y sus vecinos. En (b), a los vecinos cuyo valor es mayor o igual al pixel central se les asigna un bit 1, mientras que el resto toma el valor 0. Estos bits se hacen corresponder con las potencias de 2 de la figura (c), resultando la figura (d). Finalmente, el LBP es la suma de los pixeles en (d). Figura tomada de [48].

2.3. Sistema de visión computacional (SVC)

El sistema de visión computacional (SVC) está a cargo de detectar y trackear caras y manos en una secuencia de imágenes, las cuales pueden provenir de un video o una cámara. La figura 2.4 muestra un diagrama de las componentes del SVC. Este sistema es producto de investigaciones anteriores y se encuentra descrito en [23]. El desarrollo de este sistema al momento de aplicarlo al reconocimiento de gestos estáticos se encuentra documentado en [32].

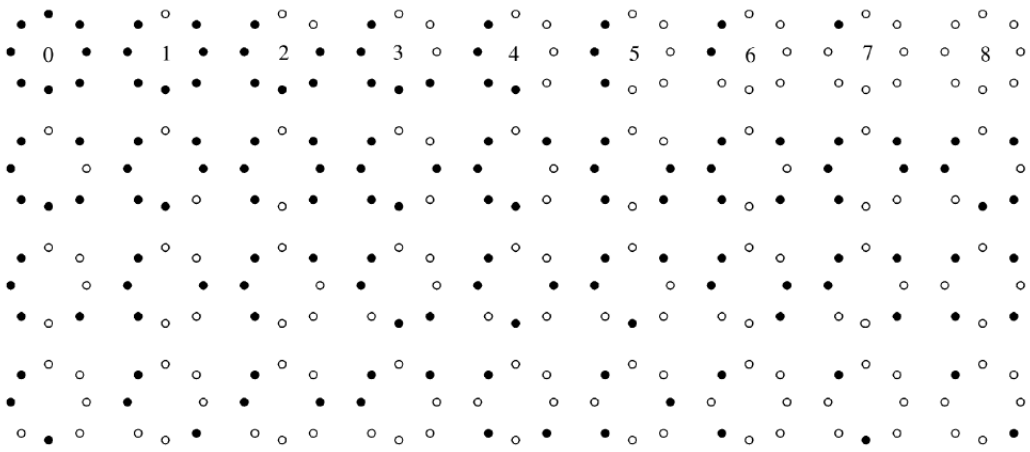


Figura 2.3: Valores de LBP, considerando invarianza a rotación. Figura tomada de [46].

A continuación se presentan los módulos de este sistema, siguiendo la exposición hecha en [23].

2.3.1. Detección y tracking de cara

El sistema de detección de caras se encuentra descrito en [61]. En breve, este módulo está implementado como una cascada anidada de clasificadores boosted (*nested cascade of boosted classifiers*). La técnica de boosting busca crear un clasificador fuerte a partir de un conjunto de clasificadores débiles, es decir, clasificadores basados en reglas simples que tienen en general un rendimiento pobre, pero que son mejores que una clasificación aleatoria. El conjunto de estos clasificadores se organiza en una cascada de complejidad creciente que permite descartar rápidamente las subimágenes que no contienen caras, lo cual resulta en tiempos de procesamiento bajos y alta precisión en la clasificación.

Dado que el algoritmo de detección es demasiado costoso para ser aplicado repetidamente en tiempo real, una vez detectada la cara, se efectúa el seguimiento o tracking de ésta mediante el algoritmo mean-shift (sección 2.2.1). Este algoritmo es inicializado con la subimagen de la cara detectada, y usa un histograma de color RGB cuantizado a 16 niveles (4 bits) como vector de características.

Teniendo a la vista el objetivo de detectar la mano, la información de las dimensiones de la cara es usada para delimitar una región de la imagen que con gran probabilidad contiene piel de la mano derecha. Las coordenadas (x_R,y_R) de la esquina superior izquierda de esta región, su ancho w_R y su alto h_R están dados por:

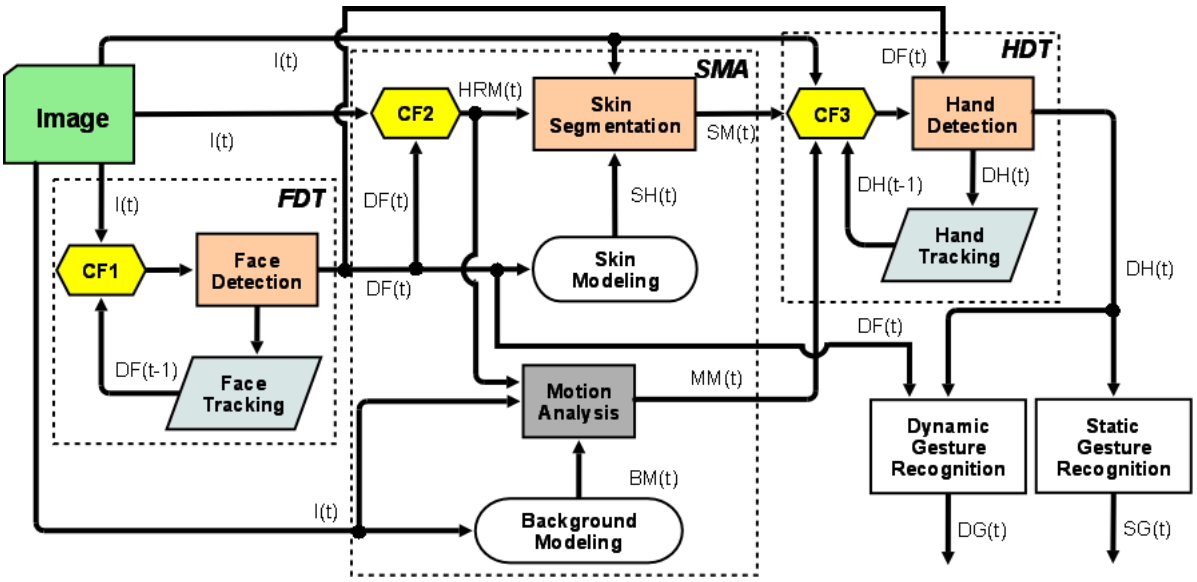


Figura 2.4: El sistema se compone de los módulos FDT (detección y tracking de caras), SMA (análisis de piel y movimiento) y HDT (detección y tracking de manos). $I(t)$ es la imagen obtenida de la cámara, $DF(t)$ y $DH(t)$ las caras y manos detectadas. $BM(t)$ la imagen del fondo, $MM(t)$ la máscara de movimiento, $SH(t)$ el histograma de piel, $HRM(t)$ la región de búsqueda de la mano, $SM(t)$ la máscara de piel y movimiento. $DG(t)$ y $SG(t)$ son los gestos dinámicos y estáticos detectados en el instante t.

$$
\begin{aligned}
x_R &= max(0, x_f - 3w_f) \\
y_R &= min(I_h, y_f + 2h_f) \\
w_R &= min(I_w - x_R, 1.5w_f) \\
h_R &= min(I_h - y_R, 3h_f)
\end{aligned}
\tag{2.9}
$$

donde x_f, y_f son las coordenadas de la esquina superior izquierda de la cara, w_f y h_f son su ancho y alto respectivamente, I_w es el ancho de la imagen e I_h es la altura de la imagen.

El sistema vuelve a realizar una detección después de una cierta cantidad de cuadros haciendo tracking (entre 10 y 30). Esto permite recuperarse de casos de tracking fallido y posibilita el ingreso de un nuevo usuario al campo visual de la cámara.

2.3.2. Segmentación de piel y análisis de movimiento

El módulo de segmentación de piel y análisis de movimiento (SMA) tiene como objetivo determinar regiones de la imagen donde buscar la mano. Esta región ya está delimitada por las ecuaciones (2.9). La información de piel y movimiento permite restringir aún más esta región. La salida de este módulo es, justamente, una máscara de piel y movimiento.

Segmentación de piel

La piel es segmentada usando el algoritmo skindiff [22]. Este algoritmo consta de dos fases: una fase de clasificación pixel por pixel, y una fase de difusión. En la fase de clasificación se determina si un pixel tiene o no una alta probabilidad de ser piel usando un modelo de piel G_t. En la fase de difusión se hace la decisión definitiva de si un pixel es o no de piel usando información de los vecinos. El modelo de piel G_t es un histograma de color que es actualizado continuamente con la información del color de la cara, según la regla

$$G_t = G_{t-1}\alpha + \hat{G}_{cara(t)}(1-\alpha)$$

donde $\hat{G}_{cara(t)}$ es estimado a partir de la cara del cuadro actual y α determina la ponderación dada a cada modelo. Hacer esta actualización permite adaptar el modelo a distintas condiciones de iluminación y a la variabilidad del color de piel entre distintas personas.

Análisis de movimiento

El análisis de movimiento tiene como objetivo obtener una máscara de movimiento, esto es, una representación de las partes de la imagen que se encuentran en movimiento.

El análisis de movimiento está basado en la técnica de substracción de fondo (*background substraction*). Esta técnica considera el fondo como aquellos pixeles que no cambian su valor de un cuadro al siguiente; el efecto de eliminar estos pixeles del cuadro actual es obtener una imagen de movimiento (es decir, una imagen que muestra solo aquellas partes que han cambiado considerablemente de un cuadro al siguiente).

Para conseguir esto, el algoritmo mantiene un modelo del fondo representado como una imagen, la cual es actualizada cuadro a cuadro haciendo un promedio ponderado entre el modelo almacenado (peso 0,9) y el cuadro actual (peso 0,1). La substracción de fondo se lleva a cabo como una resta en valor absoluto entre el modelo de fondo y el cuadro actual. Aplicando un umbral, es posible obtener una máscara binaria de movimiento.

2.3.3. Detección y tracking de manos

Puesto que las manos son objetos altamente deformables, son difíciles de identificar automáticamente en una imagen. Para sobreponerse a esta dificultad, se establece la convención de que el usuario muestra su mano en una forma predefinida para permitir su detección (haciendo el gesto estático "puño"); una vez detectada, se efectúa su tracking mediante el algoritmo mean-shift. La figura 2.5 ilustra este proceso. Usando esta convención, se elimina el problema de detectar la mano en una pose arbitraria. Además, esta disposición permite mayor eficiencia, ya que los algoritmos de detección son demasiado caros como para ser aplicados cuadro a cuadro, mientras que mean-shift es un algoritmo comparativamente rápido.

Tal como con en la detección de caras (sección 2.3.1), la detección de puños usa las

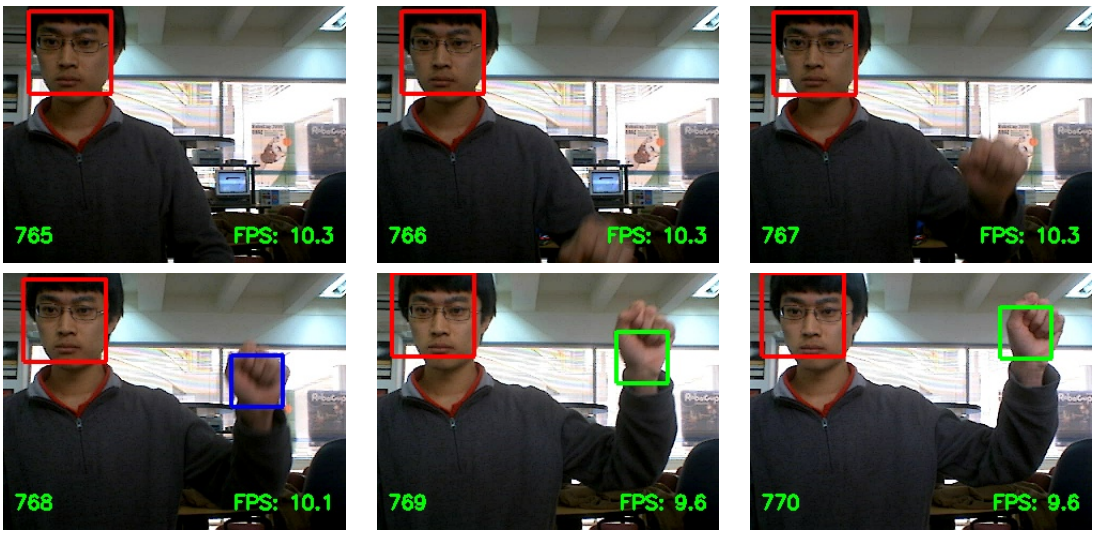


Figura 2.5: Como convención, se usa el gesto "puño" para inicializar el seguimiento de la mano. La detección se marca con un recuadro azul (cuadro 768) y el tracking con recuadros de color verde (cuadros 769 y 770).

cascadas anidadas de clasificadores boosted documentadas en [61]. La detección del puño se repite periódicamente cada cierto número de cuadros.

El tracking de la mano resulta ser más difícil que el de la cara, ya que se trata de objetos más pequeños, deformables y movedizos. Por esto, además de la característica de color RGB, cuantizada a 4 bits, se agrega una característica de borde cuyo valor es 1 ó 0 dependiendo de si la diferencia de intensidad del pixel con respecto a alguno de sus vecinos supera cierto umbral. Por lo tanto, cada pixel queda caracterizado por una tupla $(r, g, b, borde)$. El uso de borde obtiene una tasa de tracking exitoso del 89 %, mientras que el tracking basado solo en color alcanza un 60 %.

2.4. Reconocimiento de gestos dinámicos

En el ámbito de detección de gestos corporales, los trabajos más antiguos se remontan a principios de la década de 1990; como ejemplo, están los artículos de Yamato et al. [67] y Darrell y Pentland [20], motivados por aplicaciones como la interacción humano-computador y la vigilancia automática.

El primer paso en el reconocimiento de gestos es la adquisición de datos. Hay sistemas que usan sensores portables, como guantes. Stiefmaier et al. [57] usan una red de sensores inerciales para identificar las acciones del portador (figura 2.6a). Unzueta et al. [59] usan un dispositivo Wii (figura 2.6b). Ward et al. [64] usan un acelerímetro 3-D que se lleva en la muñeca. Los dispositivos portables tienen la ventaja de entregar datos precisos, pero son costosos e invasivos. Esto hace atractivos a los métodos basados en visión, menos costosos y más cómodos para el usuario, pero donde la adquisición de datos se vuelve ostensiblemente más difícil.

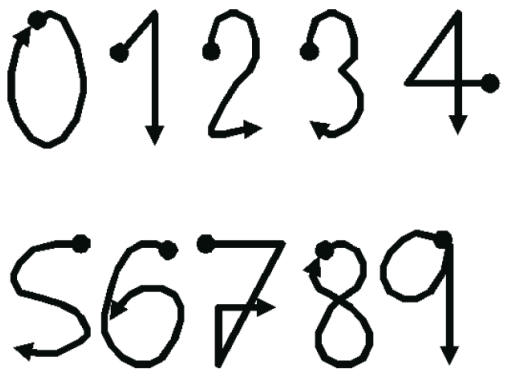
(a) Sensores inerciales (tomado de [57])

(b) Wii (tomado de [59])

Figura 2.6: Ejemplos de sensores que el usuario debe manipular o llevar puestos.

Gesture	Command	Command Description
	Last	Go to last slide.
	First	Go to first slide.
	Next	Go to next slide.
	Previous	Go to previous slide.
	Start	Start presentation.
	Bye	Quit PowerPoint™.
	White	Fill screen with white color.
	Black	Fill Screen with black color.
	Hidden	Show hidden slide.
	Stop	End presentation.

(a) Gestos de Lee y Kim, tomado de [39]



(b) Gestos de Unzueta et al., tomado de [59]

Figura 2.7: Gestos basados en trayectorias.

La gama de gestos cuyo reconocimiento ha sido estudiado es amplia. Los más simples son gestos basados en trayectorias, como los de Lee y Kim [39], usados para dar órdenes a un programa de diapositivas, o los dígitos trazados en el aire [3, 29, 59]. La figura 2.7 muestra ejemplos de gestos basados puramente en trayectorias. Varios trabajos están dedicados al reconocimiento de lenguaje de señas para sordomudos [10, 38, 68] como los mostrados en la figura 2.8. Una vastísima cantidad de trabajos estudia gestos "naturales" hechos con brazos y manos, como apuntar, agitar, saludar, gesticular al hablar, etc. [16, 18, 20, 40, 44, 51, 53, 54, 56, 66, 69], como los de la figura 2.9. Finalmente, varios trabajos se ocupan de la detección de gestos de cuerpo completo, incluyendo movimientos deportivos [52, 67] (figura 2.10), danza [27, 37] (figura 2.11), y acciones humanas en general [11, 12] (figura 2.12).

Pueden distinguirse dos familias de métodos de reconocimiento de gestos dinámicos: los métodos basados en modelos de estado, y los métodos basados en apariencia.

Los **métodos basados en modelos de estados** modelan cada gesto como una secuencia de estados, y el proceso de reconocimiento consiste en hacer una correspondencia entre el modelo y la secuencia de datos reales. En el proceso de hacer esta correspondencia se permite

parear un cuadro con un mismo estado más de una vez, o bien saltarse estados, con el fin de tomar en cuenta que un mismo gesto puede ser ejecutado con distintas velocidades y distintos énfasis. Las instancias más notables de esta familia de técnicas son Dynamic Time Warping [19], Hidden Markov Models [50], Continuous Dynamic Programming [3, 47] y Conditional Random Fields [62].

Los Hidden Markov Models (HMM) en particular se han convertido en la técnica predominante de los sistemas de reconocimiento de gestos dinámicos. El uso de este modelo surgió en la comunidad de reconocimiento de voz, donde ha obtenido buenos resultados. La principal ventaja de los HMMs es su estructura matemática rica y bien estudiada en conjunto con algoritmos igualmente bien conocidos para el entrenamiento y la evaluación de modelos.

El sistema desarrollado en esta memoria se basa en una estrategia distinta, que consiste en calcular características geométricas y cinemáticas que caracterizan en su totalidad al movimiento y que son independientes de la longitud y velocidad del gesto ejecutado. Por lo tanto, este método es más cercano a los **métodos basados en apariencia** o *appearance-based methods*. Hay varios ejemplos de esta familia de técnicas. Cui y Weng [17] representan cada mano detectada como un conjunto sin procesar de pixeles, normalizados a cierto tamaño. Una secuencia de tales manos es entregada a un clasificador, el cual entrega el gesto reconocido. Esta técnica trata de encontrar estadísticamente las características que mejor permiten discriminar entre un gesto y otro, mientras que en esta memoria se definen tales características explícitamente.

Yang et al. [68] representan una secuencia de cuadros como una secuencia de las transformaciones afines necesarias para transformar regiones correspondientes entre cuadros consecutivos, y luego entregan esta representación a una red neuronal con retraso temporal (*Time Delay Neural Network*). Cutler y Turk [18] calculan el flujo óptico (*optical flow*) de cada cuadro y luego segmentan el resultado en "masas de movimiento" ("motion blobs"). Cada cuadro se caracteriza por el número de *motion blobs*, el movimiento de dichos blobs, y su posición. Un sistema basado en reglas infiere el gesto ejecutado. Davis y Bobick [11] proponen un método que usa la llamada *motion history image*, que representa el movimiento espaciotemporal en imágenes, evitando así el análisis temporal explícito. En [54] se discute un algoritmo parecido.

En el reconocimiento en línea de gestos, los gestos se encuentran incluidos en una secuencia de otros movimientos no necesariamente significativos. El comienzo y el final de un gesto son desconocidos y determinarlos es un problema importante llamado segmentación de gestos (*gesture segmentation*). El enfoque más simple para resolver este problema consiste en usar información de bajo nivel como la velocidad de la mano, su aceleración y sus variaciones angulares [9,38,59,63]. Algunas obras proponen el uso de indicadores explícitos de finalización de gesto, como sacar la mano del alcance de la cámara o ejecutar un gesto predefinido [42]. La principal dificultad que surge al hacer este tipo de segmentación es que entrega datos poco confiables, pues es normal que la mano haga pausas o tenga cambios de curvatura bruscos durante la ejecución de un gesto; a la inversa, es posible que al principio o final de un gesto no haya variaciones importantes de velocidad ni curvatura, y que por lo tanto el sistema no se percate del inicio o fin de un gesto. En definitiva, usar variables de bajo nivel obliga al usuario a marcar explícitamente el comienzo y el final de cada gesto, limitando la usabilidad.

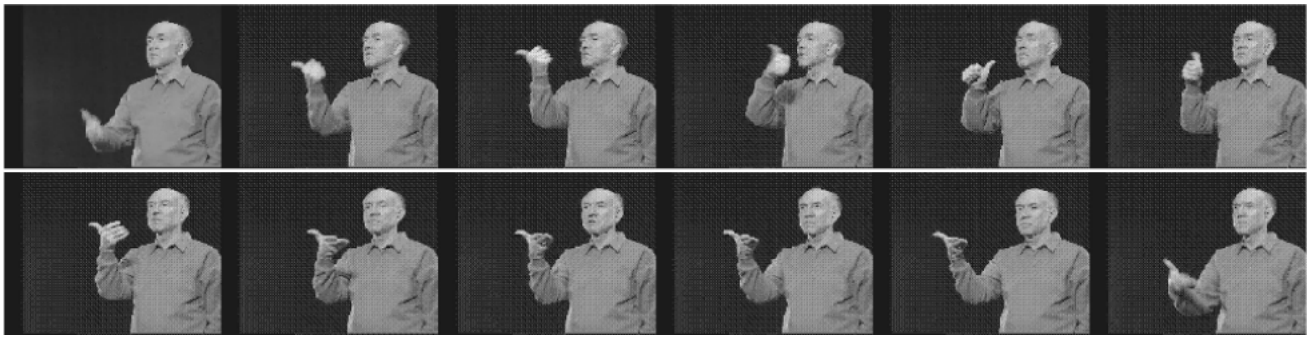


Figura 2.8: Lenguaje de señas estadounidense. Tomado de [68].

Figura 2.9: Movimientos con los brazos ("agitar ambas manos", "agitar mano derecha", "levantar mano derecha", "izquierda abajo derecha arriba"). Tomado de [40].

Por esto, una segunda familia de algoritmos trata la segmentación y el reconocimiento como aspectos de un mismo problema. La segmentación se realiza buscando, en la secuencia de datos entrantes, aquellas subsecuencias que entregan puntajes altos al ser comparados con los modelos gestuales. Por ejemplo, algunos métodos basados en HMMs encuentran candidatos a puntos de segmentación (iniciales o finales) aplicando un umbral a la probabilidad gestual [25, 39] o bien modelando explícitamente un "modelo de basura" o "modelo de relleno" que se hace corresponder con los movimientos sin significado que ocurren entre gesto y gesto. Los métodos basados Dynamic Time Warping, Continuous Dynamic Programming y Conditional Random Fields se caracterizan por entregar medidas de similitud o probabilidad, por lo que también son aptos para esta forma de segmentación.

En nuestro sistema, la segmentación y el reconocimiento son ejecutados simultáneamente por medio de encontrar subsecuencias de cuadros que tengan alta probabilidad de contener un gesto. Por otro lado, también se usa información de bajo nivel como la velocidad de la mano y la no detección de la mano —probablemente debida a que ésta no se encuentra visible— para dar un gesto por terminado y reiniciar el proceso de reconocimiento.

Figura 2.10: Secuencia deportiva, tomada de [67].

Figura 2.11: Secuencias de danza estudidadas en [27].

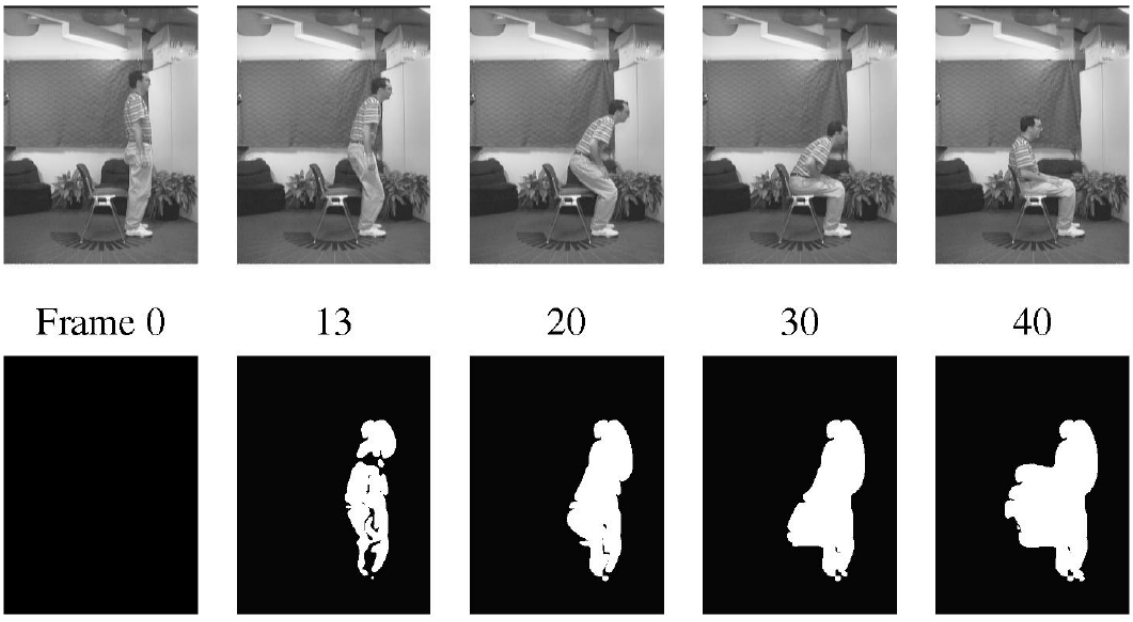


Figura 2.12: La acción de sentarse. Tomado de Bobick et al. [11].

2.5. Otros antecedentes en el reconocimiento de patrones

2.5.1. Reconocimiento de caracteres

El reconocimiento de caracteres es un problema de reconocimiento de patrones que busca traducir imágenes de textos manuscritos o disponibles solo en papel en representaciones manipulables por computador. Un estudio general del tema se encuentra en [49].

El reconocimiento de caracteres es de interés para esta memoria en cuanto se trata de un problema parecido al reconocimiento de gestos dinámicos. En ambos casos, se trata de clasificar una figura, por lo que hay un razonamiento geométrico de por medio. El reconocimiento de caracteres puede hacerse en línea o fuera de línea; la diferencia entre ambas modalidades está en la disponibilidad de información para efectuar el reconocimiento: el reconocimiento en línea puede razonar sobre la secuencia ordenada de puntos por donde pasa el dispositivo de escritura, mientras que el reconocimiento fuera de línea solo puede razonar sobre la forma resultante. En este sentido, el reconocimiento de gestos es más similar al reconocimiento en línea, lo cual no impide que las técnicas de reconocimiento fuera de línea puedan ser de utilidad. En efecto, en esta memoria se saca provecho de ambos tipos de información.

2.5.2. Medidas de similitud entre formas geométricas

Se ha diseñado una gran variedad de funciones de similitud para objetos como figuras geométricas e imágenes [60]. En esta memoria, una medida de similitud de interés es la distancia de Hausdorff. Dadas dos figuras en el espacio, la distancia de Hausdorff es, informalmente, la mayor distancia que un punto está forzado a recorrer para ir de una figura a la otra.

Dados dos conjuntos finitos y no vacíos de puntos F y G en $\mathbb{R}^2$ y una distancia d, la distancia de Hausdorff D entre F y G es

$$\begin{aligned} D(F,G) &= \text{máx}(h(F,G), h(G,F)) \\ \text{donde } h(F,G) &= \max_{f\in F}(\min_{g\in G} d(f,g)) \end{aligned} \tag{2.10}$$

Esta distancia puede aplicarse en particular a figuras marcadas sobre imágenes binarias. En este caso, los conjuntos de puntos son los pixeles que pertenecen a la figura y la función de distancia puede ser la euclidiana o bien puede quedar definida por un conjunto de vecindades cuadradas (figura 2.13) o romboidales (figura 2.14).

La distancia de Hausdorff ha sido aplicada en reconocimiento de caracteres, reconocimiento facial y búsqueda de patrones en imágenes, véase por ejemplo [7, 35, 71].

3	3	3	3	3	3	3
3	2	2	2	2	2	3
3	2	1	1	1	2	3
3	2	1	0	1	2	3
3	2	1	1	1	2	3
3	2	2	2	2	2	3
3	3	3	3	3	3	3

Figura 2.13: Vecindades cuadradas. Los números indican la distancia al centro, marcado con cero.

6	5	4	3	4	5	6
5	4	3	2	3	4	5
4	3	2	1	2	3	4
3	2	1	0	1	2	3
4	3	2	1	2	3	4
5	4	3	2	3	4	5
6	5	4	3	4	5	6

Figura 2.14: Vecindades romboidales. Los números indican la distancia al centro, marcado con cero.

2.5.3. Comparación de secuencias temporales

Un problema fundamental en el reconocimiento de patrones temporales es la velocidad y énfasis variables que pueden presentar dos secuencias distintas. Esta variabilidad hace que una comparación entre secuencias no sea un procedimiento obvio. Para sobreponerse a esta dificultad, se hace necesario establecer algún tipo de correspondencia o calce entre secuencias que tome en cuenta dicha variabilidad. Una solución a este problema es el Dynamic Time Warping, el cual está estrechamente relacionado con el concepto de distancia de edición (conocido también como distancia de Levenshtein), que será descrito en esta sección.

La distancia de edición cuantifica la diferencia entre dos secuencias $A = \{a_1, \ldots, a_n\}$ y $B = \{b_1, \ldots, b_m\}$ (no necesariamente del mismo largo). Para lograr esto, se plantea el problema de transformar A en B mediante la menor cantidad posible inserciones, eliminaciones y sustituciones. El número mínimo necesario de tales operaciones se llama el costo o distancia de edición.

El cálculo de la distancia de edición procede como sigue. Se designa con A_i el prefijo de A cuya longitud es i, y se define análogamente B_j. Se identifica el subproblema de transformar un prefijo de A en un prefijo de B. Si $d(i, j)$ es el costo de transformar A_i en B_j, es fácil identificar los costos base $d(i, 0) = i$ y $d(0, j) = j$. Además, $d(i, j)$ puede calcularse recursivamente según la expresión:

$$d(i,j) = \begin{cases} d(i-1, j-1) & \text{si } a_i = b_j, \\ 1 + \text{mín}(d(i-1,j), d(i,j-1), d(i-1,j-1)) & \text{si no} \end{cases} \tag{2.11}$$

donde $d(i-1, j)$ da cuenta del costo de inserción, $d(i, j-1)$ del costo de eliminación, y $d(i-1, j-1)$ del costo de sustitución.

La aplicación repetida de esta fórmula permite calcular $d(i, j)$ para múltiples valores de i y j hasta llegar a $d(n, m)$, que es la distancia de edición entre A y B. La secuencia de inserciones, eliminaciones y sustituciones que dan origen a este valor se puede obtener volviendo sobre los pasos del algoritmo, examinando cómo se obtiene el mínimo en cada aplicación de la ecuación (2.11).

Capítulo 3

Sistema de visión computacional (SVC)

El sistema de visión computacional (SVC) tiene como fin detectar caras y manos en una imagen. Este sistema fue desarrollado en el contexto del reconocimiento de gestos estáticos (memoria de Hardy Francke [32]) y se encuentra descrito en la sección 2.3.

3.1. Mejoras necesarias para el reconocimiento de gestos

Al momento de abordar la memoria se contaba con un detector facial altamente confiable (tasas sobre el 90 %) [61], un módulo de segmentación de piel adaptivo de buen funcionamiento, basado en [22], y un módulo de análisis de movimiento basado en un algoritmo sencillo y confiable como la substracción de fondo, descrito en la sección 2.3.2.

Buena parte de los módulos descritos están al servicio del tracking de la mano, que es crucial para hacer una correcta detección de gestos dinámicos. Las tasas de tracking correcto eran del 89 % en la base de datos ALON-EASY (sección 4.2.1) usando características de color y borde (véase la sección 2.3.3). La tasa de tracking correcto limita automáticamente la tasa de reconocimiento de gestos, por lo que se hizo necesario mejorar el rendimiento del tracking de la mano.

En esta sección se describen distintas medidas que buscan mejorar el tracking de la mano.

3.1.1. Tracking con movimiento y piel

La máscara de piel y movimiento (MPM), que es la salida del módulo de segmentación de piel y análisis de movimiento (SMA, sección 2.3.2), se puede usar para mejorar el tracking.

Para esto, se superpone la MPM al cuadro actual. Como resultado, las partes de la imagen

que no son piel ni están en movimiento quedan de color negro. Si se inicializa correctamente el tracking, que basa su funcionamiento en histogramas de color, la ventana de tracking quedará restringida a las zonas de piel y movimiento, donde se espera que esté la mano. El costo adicional de realizar este procedimiento es mínimo, pues se trata de información que ya estaba disponible para otro fin (la detección eficiente de la mano).

Según muestra la figura 3.1, el uso de esta técnica disminuye la posibilidad de confusión de la mano con objetos del fondo que tengan color similar, los cuales, si son estáticos, siempre quedarán ennegrecidos. Sin embargo, no impide la confusión de la mano con el brazo cuando el usuario lleva los brazos descubiertos, puesto que el brazo, por ser móvil y de color piel, no forma parte de las zonas ennegrecidas.

3.1.2. Evaluación de nuevas características para tracking

El algoritmo mean-shift basa su funcionamiento en la comparación de histogramas: aquel del objeto trackeado y aquellos de las subimágenes candidatas. El histograma es una distribución de probabilidad de algún atributo numérico de los pixeles, y debe representar fielmente una subimagen para que el tracking funcione. Los histogramas de color son la elección más natural; sin ir más lejos, los experimentos llevados a cabo en el trabajo original de Comaniciu et al. [15] usan esta descripción.

En múltiples trabajos posteriores al de Comaniciu, uno de los tópicos explorados es el uso de otras características, entre éstas: textura, bordes e intensidad (véase [6]).

En esta memoria se considera el uso de características de textura. La textura de un pixel es una descripción asignada a este según la pertenencia a un patrón de intensidades en conjunto con sus vecinos (véase la sección 2.2.2).

Integrar cualquier característica al histograma implica hacer consideraciones de eficiencia. El histograma registra un conteo de las características tomadas en conjunto, por lo que agregar una característica que toma n valores distintos tiene como consecuencia multiplicar el tamaño del histograma —y por lo tanto, el tiempo de procesamiento— en un factor de n.

En este sentido, la característica de borde es la más económica, puesto que tiene 2 valores posibles (0 ó 1, dependiendo de si el pixel es o no de borde). Los valores RGB de los canales de color, al estar cuantizados a 4 bits, entregan 16 valores distintos cada uno. En cambio, añadir una característica como la textura LBP (sección 2.2.2), que asocia uno de 36 valores a cada pixel, tiene un costo mucho mayor, y éste no puede atenuarse mediante cuantización, puesto que los valores entregados por LBP no pertenecen a un continuo, por lo que eliminar bits lo haría indistinguible de otros valores que tienen significados muy distintos.

La característica de textura fue probada, pero no incrementó significativamente el rendimiento del tracking (incluso lo empeoró en algunos casos), mientras que aumentó el tiempo de procesamiento necesario. Por lo tanto, no fue considerada en la implementación final.

3.1.3. Actualización del modelo

Al centro del algoritmo mean-shift está la comparación de histogramas. Usualmente, el objeto a seguir queda caracterizado por el histograma calculado en la primera detección. Sin embargo, si el objeto es deformable o sufre cambios de iluminación durante su movimiento, dicho histograma inicial deja de tener vigencia y es de esperar que el tracking falle. Por lo tanto, parece conveniente actualizar el modelo inicial con el histograma actual, lo cual será eficaz siempre que el tracking funcione bien en primer lugar.

Algunos experimentos sencillos demuestran que reemplazar el histograma modelo por el de la ventana actual no trae buenos resultados si se hace cuadro a cuadro. Tampoco trae buenos resultados hacer un promedio ponderado entre el modelo actual y la ventana de tracking más reciente, si se hace en todos los cuadros. Un tercer enfoque, consistente en actualizar el histograma modelo cada n cuadros, demuestra tener mejores resultados. Este último enfoque es el que finalmente se usa en el sistema implementado.

3.1.4. Múltiples puntos iniciales

Un problema común en los algoritmos de optimización basados en ascenso por el gradiente es el de converger a un máximo local en vez de uno global. Para evitar esto, una técnica usual es la búsqueda de varios máximos locales a partir de distintos puntos iniciales (elegidos aleatoriamente). El algoritmo se hace converger para cada punto de partida y se elige finalmente el mejor resultado de entre los máximos encontrados.

En vez de puntos de partida aleatorios, es interesante también la idea de predecir el movimiento a partir de los últimos cuadros, de modo de comenzar la búsqueda en los puntos donde, a juzgar por la tendencia del movimiento, debería encontrarse la mano. Sin embargo, algunos experimentos sencillos realizados en casos de tracking fallido, en los cuales la ventana de tracking se distrae con un elemento de color similar como el brazo (un máximo local), demuestran que la predicción de movimiento no necesariamente incrementa la efectividad del tracking, y que aún habría que tomar otras medidas para evitar que el algoritmo mean-shift considere como similares el puño con otras zonas de piel. Por lo tanto, el sistema final no contempla el uso de múltiples puntos iniciales para el tracking.

3.1.5. Detección y tracking de la mano simultáneos

Un problema sutil de la detección seguida de tracking es el siguiente: puesto que la detección cuesta mucho más tiempo que el tracking, es posible que, tras una detección exitosa, en el tiempo que toma hacer esta detección, la mano se haya movido tanto que el algoritmo mean-shift, que trabaja sobre el cuadro siguiente a la detección, sea incapaz de converger a la nueva posición (se debe tomar en cuenta que, en tiempo real, se pierden cuadros de la cámara debido al tiempo de procesamiento). Por ser un algoritmo de ascenso por el gradiente, mean-shift solo convergerá si en la posición inicial la función de similaridad se encuentra en una cuesta ascendente donde el gradiente apunta a un máximo local. La zona de la imagen donde se cumple esta condición es llamada cuenca de atracción (*basin of attraction*).

Para subsanar este problema, en vez de hacer una nueva detección incondicionalmente después de cierta cantidad de cuadros, el sistema realiza una detección en paralelo a lo largo de varios cuadros —usando poco tiempo de procesamiento, y por lo tanto permitiendo la recepción más seguida de cuadros. Si se detecta una mano, y esta coincide con los datos del tracking, se corrobora que el tracking es correcto, por lo que no es necesario hacer una nueva detección. En cambio, si se detecta la mano y ésta no coincide con el tracking actual, se procede a hacer una nueva detección desde cero. Que el tracking no coincida con la detección se debe normalmente a que el tracking ha fallado completamente, por lo que esta es una decisión correcta.

Si no se detecta la mano, se supone que el tracking está realizándose correctamente, y por lo tanto no se realiza una detección. Desde luego, es posible que dicha suposición no sea correcta. Así y todo, si en realidad el tracking fuese erróneo, el sistema es capaz de recuperarse si el tracking converge a un punto fijo de la imagen, pues se agrega la condición de reiniciar el procedimiento de tracking completo si se detecta que el tracking de la mano ha estado inmóvil por suficiente tiempo y que el coeficiente de Bhattacharyya (sección 2.2.1) tiene un valor bajo (en esta aplicación se usa un umbral de 0.5).

3.2. Organización modular del software

El sistema desde siempre fue concebido modularmente, pero esto no siempre estuvo reflejado en el código fuente mediante la separación de módulos distintos en distintos archivos o clases, dificultando la mantenibilidad y facilidad de comprensión del código.

La tarea de implementar el diseño del sistema tuvo necesariamente un carácter de refactorización, puesto que se trabajó sobre código existente. Se decidió usar una organización orientada a objetos, la cual ofrece modularidad y flexibilidad para modificaciones futuras mediante el uso de herencia, polimorfismo y patrones de diseño. La delimitación conceptual entre variables públicas y privadas facilita la comprensión del código y enfatiza el carácter modular del diseño al hacer explícita la distinción entre lo que forma parte del funcionamiento interno de un módulo y lo que ofrece a los otros módulos —su interfaz. Usar orientación a objetos no tiene costo adicional a la mera separación de funciones en archivos, y en definitiva se considera un mejor enfoque.

Originalmente, el código estuvo estructurado en torno a varias variables y funciones globales. En miras a modularizarlo, se procedió a identificar cada variable y función con una de las siguientes áreas:

1. Detección y tracking de cara
2. Segmentación de piel
3. Análisis de movimiento
4. Detección y tracking de cara

Luego, se creó una clase para cada una de las áreas mencionadas, convirtiendo variables globales en miembros de alguna de las clases y funciones globales en métodos. Cada clase ofrece una interfaz minimalista. Mediante la lectura de la firma de los métodos es posible saber qué datos de entrada necesita cada módulo y cuáles son los datos de salida. Esto hace explícita las dependencias y evita que cualquier función pueda usar cualquier dato disponible globalmente, mejorando la legibilidad y la posibilidad de predecir del comportamiento del programa.

En la práctica, no fue necesario organizar la parte de detección de piel, que ya estaba en una clase; no así con los otros tres módulos. La figura 3.2 presenta un diagrama de clases simplificado del estado actual del sistema. La clase `ImageProcessor` actúa como coordinador de la aplicación, proveyendo las entradas correctas a las clases `SkinDetect`, `FaceDetectionAndTracking`, `HandDetectionAndTracking` y `BackgroundProcessing`.

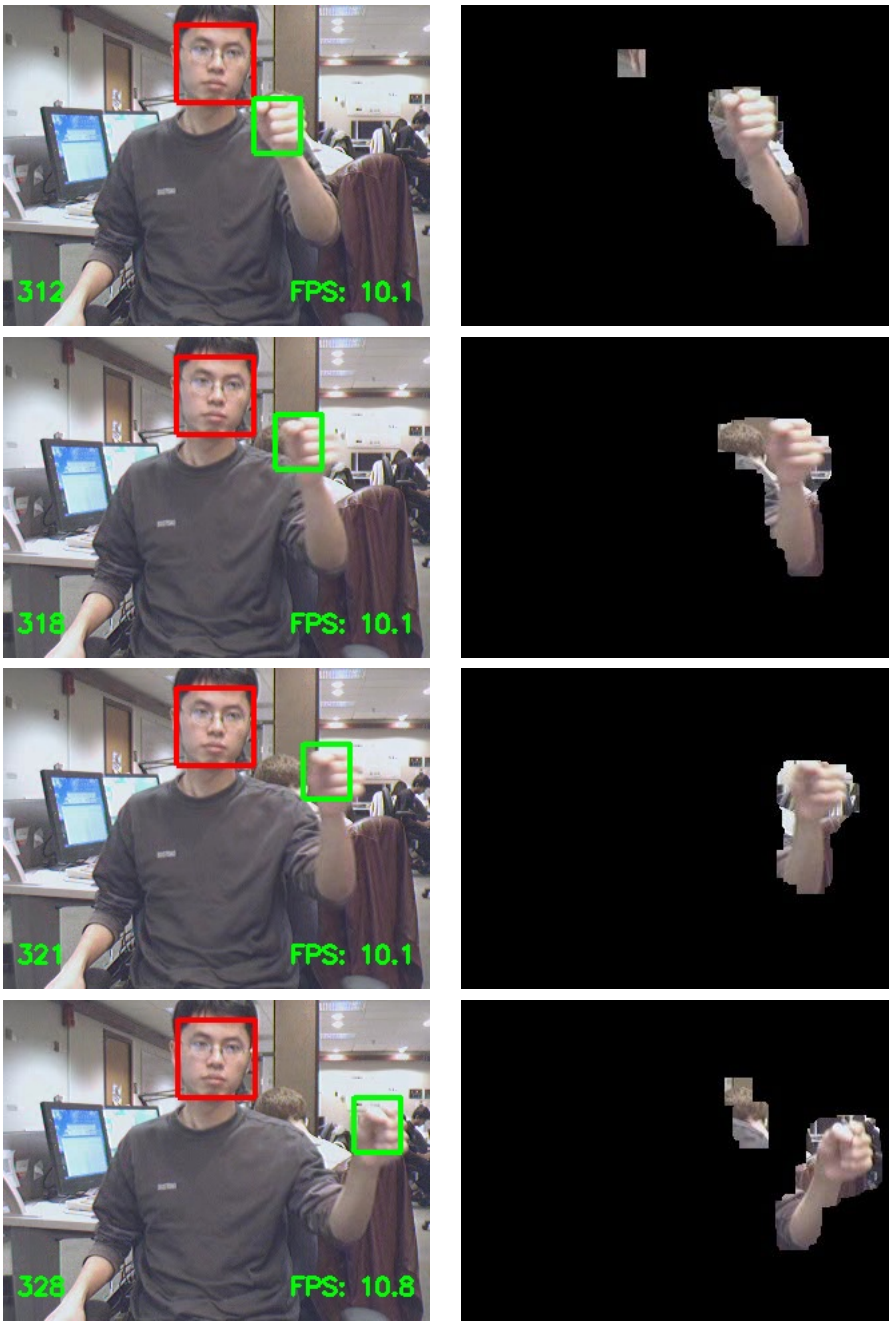


Figura 3.1: Ejemplos de uso de piel y movimiento para disminuir el área de tracking. La mano corre el riesgo de confundirse con la cabeza de la persona que se encuentra detrás. El análisis de movimiento descarta dicha zona después de pocos cuadros.

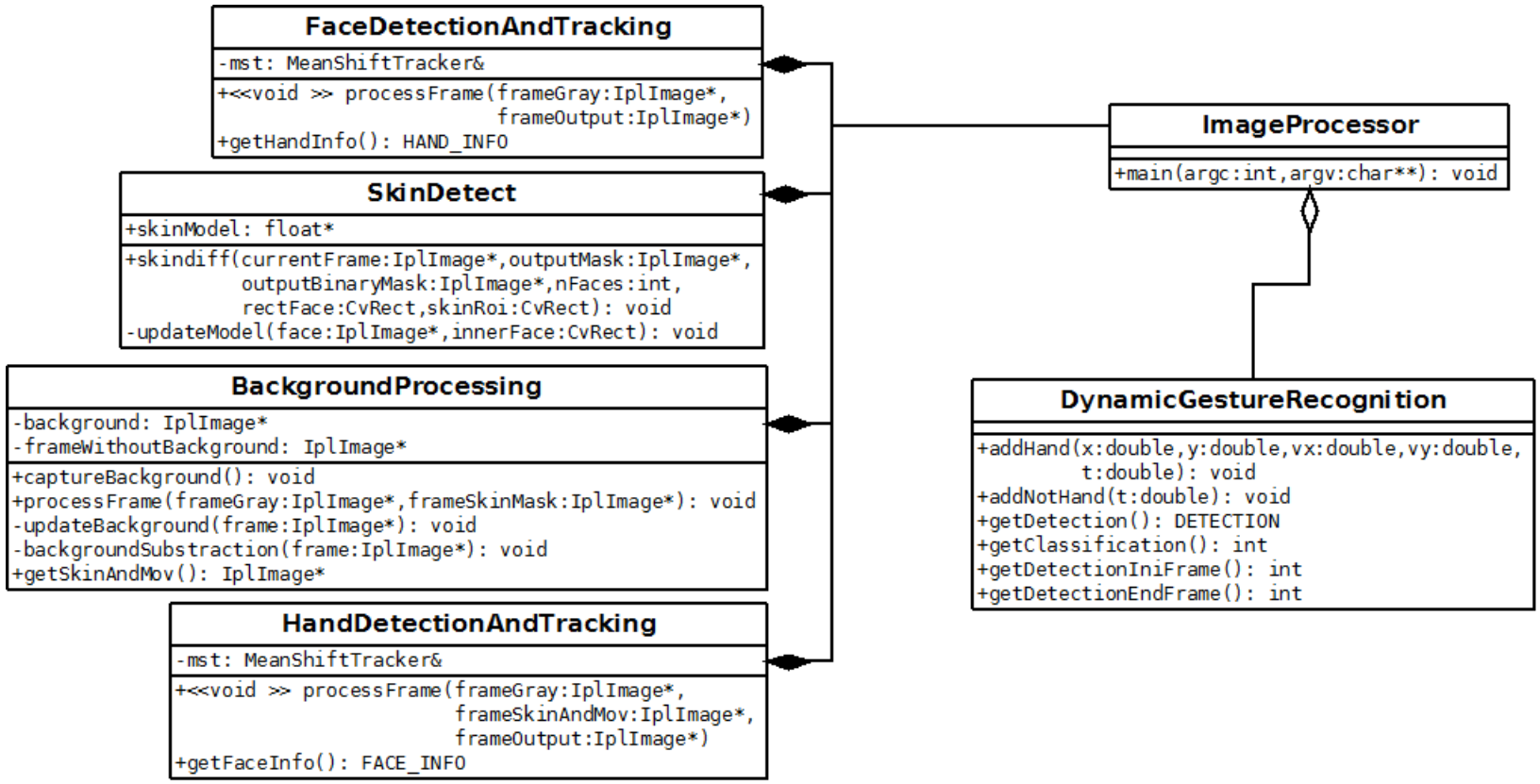


Figura 3.2: Diagrama de clases para el sistema de visión computacional (SVC). Aparecen los cuatro módulos de visión implementados como clases y se incluyen los métodos y variables más representativos de cada uno. `ImageProcessor` actúa como coordinador de la aplicación, proveyendo las entradas correctas a cada clase.

Capítulo 4

Sistema de reconocimiento de gestos

4.1. Visión general

El objetivo del sistema de reconocimiento de gestos es detectar e identificar los gestos que realiza un usuario moviendo la mano frente a una cámara a medida que estos son ejecutados.

Los gestos pertenecen a un conjunto predefinido de tamaño N_g y quedan identificados únicamente por la trayectoria que realiza la mano, no por la pose de ésta.

La solución propuesta se basa en la aplicación continua de clasificadores binarios. Hay un clasificador binario por cada gesto G, el cual entrega la probabilidad de ejecución de G. Cuando, a lo largo de varios cuadros, esta probabilidad es suficientemente alta, se declara la existencia de un candidato c, asociado a G e identificado por un cuadro inicial y final. Este candidato es sometido a pruebas de verificación antes de ser admitido en una lista de candidatos. Analizando información de bajo nivel (velocidad de la mano), el sistema realiza hipótesis sobre las intenciones del usuario (¿está realizando un gesto, retirando la mano o manteniéndola quieta?) y determina el momento en que debe elegir un gesto de la lista de candidatos para declararlo como el definitivo.

4.2. Entrenamiento

4.2.1. Datos de entrenamiento y prueba

Para entrenar el sistema, se escogieron los gestos definidos por Alon et al. [3], que son los dígitos Graffiti usados en dispositivos Palm (figura 4.1) dibujados en el aire como se ilustra en la figura 4.2.

La base de datos de entrenamiento (ALON-TRAINING en esta memoria) se encuentra disponible en línea[1] y consta de 30 videos, cada uno de los cuales muestra a una persona

[1] http://cs-people.bu.edu/athitsos/digits/

haciendo los gestos del 0 al 9, sumando un total de 30 instancias de cada gesto. En total participan 10 personas; cada una aparece en 3 videos. Los usuarios llevan un guante verde cuyo único fin es facilitar la extracción de la trayectoria para el entrenamiento: en el uso real, los usuarios no llevan guantes. El cuadro inicial y final de cada gesto es también parte de la base de datos.

En el mismo trabajo se publican dos bases de datos de prueba, llamadas "easy" y "hard". La base de datos "easy" (ALON-EASY en esta memoria) consta de 30 videos donde 10 usuarios distintos ejecutan los gestos del 0 al 9 sobre un fondo relativamente estático y con los brazos descubiertos (figura 4.3). En tanto, la base de datos "hard" (ALON-HARD en esta memoria) consta de 14 videos donde 7 usuarios distintos ejecutan los gestos del 0 al 9 sobre un fondo dinámico (con personas moviéndose constantemente) y los brazos descubiertos (figura 4.4).

Figura 4.1: Dígitos Graffiti. Imagen tomada de [3].

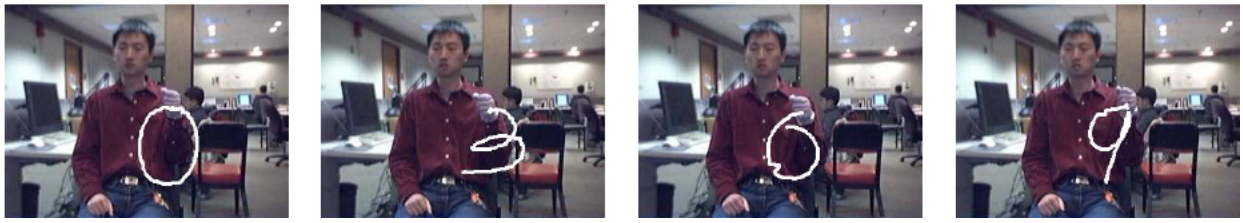


Figura 4.2: Muestra de la ejecución de los gestos. Imagen tomada de [3].

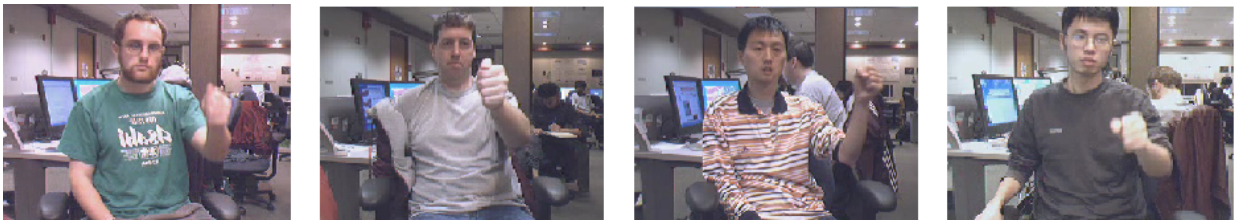


Figura 4.3: Muestras de la base de datos ALON-EASY.

Figura 4.4: Muestras de la base de datos ALON-HARD. Imagen tomada de [3].

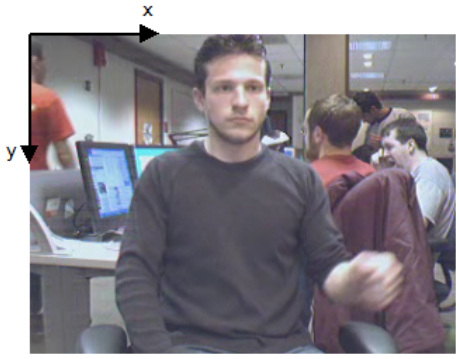


Figura 4.5: Sistema de coordenadas

4.2.2. Extracción de características

Preprocesamiento

Suponiendo detección y tracking correctos tanto de la cara como de la mano, el módulo de gestos dinámicos cuenta con los siguientes datos:

- Las coordenadas de la caja que encierra la cara (x_0^f, y_0^f) (esquina superior izquierda), (x_1^f, y_1^f) (esquina inferior derecha)
- Las coordenadas de la caja que encierra la mano (x_0^h, y_0^h) (esquina superior izquierda), (x_1^h, y_1^h) (esquina inferior derecha)

(Según la convención en aplicaciones de procesamiento de imágenes, el sistema de coordenadas es el ilustrado en la figura 4.5)

Para el reconocimiento de gestos dinámicos, es suficiente considerar la mano como un punto móvil, ignorando su altura y anchura. Por otro lado, para obtener invarianza a rotación y escala, se normalizan las coordenadas de la mano midiéndolas con respecto al centro de la cara y dividiendo por el tamaño de ésta (así, el desplazamiento de la mano es medido en unidades-cara, en vez de pixeles, y resulta independiente de la distancia del usuario a la cámara). A partir de estas coordenadas puntuales y normalizadas (x, y) se estima la velocidad de la mano (v_x, v_y) y se obtiene un vector (x, y, v_x, v_y, t) que contiene las coordenadas de la mano, su velocidad, y el instante t de obtención de los datos.

Específicamente, se hacen los siguientes cálculos para obtener (x, y):

$$\begin{aligned}
\text{Centro de la cara: } x^f = \frac{x_0^f + x_1^f}{2},\ y^f = \frac{y_0^f + y_1^f}{2} \\
\text{Tamaño de la cara: } L^f = \frac{(x_1^f - x_0^f) + (y_1^f - y_0^f)}{2} \\
\text{Centro de la mano: } x^h = \frac{x_0^h + x_1^h}{2},\ y^h = \frac{y_0^h + y_1^h}{2} \\
\text{Coordenadas definitivas de la mano: } x = \frac{x^h - x^f}{L^f},\ y = \frac{y^h - y^f}{L^f}
\end{aligned} \tag{4.1}$$

La velocidad de la mano es estimada a partir de cuadros consecutivos: $v_x = \frac{x - x_{prev}}{t - t_{prev}}$, $v_y = \frac{y - y_{prev}}{t - t_{prev}}$, donde el subíndice *prev* remite al dato del cuadro previo.

Teniendo en cuenta esta representación, el movimiento de la mano es una secuencia de vectores de la forma

$$S = \{(x(t), y(t), v_x(t), v_y(t), t)\}_{t=t_0}^{t_1}$$

Características para una secuencia de puntos

Para posibilitar el uso de clasificadores, se formula el problema de reducir una secuencia como S a un vector de características, esto es, un conjunto de estadísticas suficientes para describir un gesto, de modo de declarar su existencia y distinguirlo de otros gestos.

Esta sección presenta una discusión de las diversas características consideradas, muestra sus ventajas y desventajas, y finalmente escoge un conjunto de características, el cual es usado en definitiva en el resto de la memoria.

Las características más simples, que asimismo son las más burdas, son las que aparecen en el cuadro 4.1. Véase también la ilustración de algunas de estas características en la figura 4.7.

Algunas de estas características permiten medir, con más o menos precisión, el tamaño del gesto (por ejemplo: `DELTA_X`, `DELTA_Y`, `AVE_R`, `CH_AEXT`, `CH_PEXT`), por lo que pueden ser útiles para excluir secuencias demasiado grandes o demasiado pequeñas.

Algunas de estas características dan una noción de la distribución espacial de los puntos. Tal es el caso de `AVE_ANGLE`, `SKEW_X`, `SKEW_Y` y las covarianzas.

Finalmente, algunas de estas características no tienen una interpretación geométrica clara (`AVE_VX`, `AVE_VX`) y algunas probablemente no son relevantes, e incluso pueden ser engañosas (`AVE_X`, `AVE_Y`).

Para la evaluación de estas características se considera su capacidad de discriminar entre gestos distintos: interesa que gestos distintos tengan características distintas y que gestos

Nombre	Descripción
DELTA_X	Diferencia entre mínima y máxima coordenada en el eje x
DELTA_Y	Diferencia entre mínima y máxima coordenada en el eje y
AVE_X	Promedio de las coordenadas x
AVE_Y	Promedio de las coordenadas y
AVE_VX	Promedio de velocidades v_x
AVE_VY	Promedio de velocidades v_y
AVE_R	Promedio de las distancias de los puntos a (AVE_X,AVE_Y)
AVE_ANGLE	Ángulo polar promedio con respecto a (AVE_X,AVE_Y)
CH_AEXT	Área de la cerradura convexa
CH_PEXT	Perímetro de la cerradura convexa
COV00, COV01, COV02, COV03, COV11, COV12, COV13, COV22, COV23, COV33	Componentes de la matriz de covarianza de los vectores (x, y, v_x, v_y, t) (COV00= $\mathrm{cov}(X, X)$, COV23= $cov(V_X, V_Y)$, etc.)
SKEW_X	Asimetría estadística (*skewness*) de las coordenadas x
SKEW_Y	Asimetría estadística (*skewness*) de las coordenadas y

Cuadro 4.1: Características simples

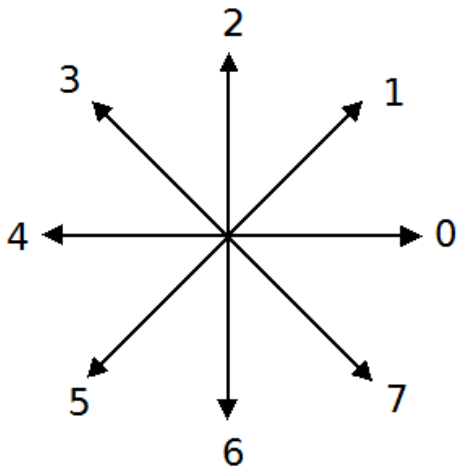


Figura 4.6: Códigos de dirección

parecidos tengan características parecidas. Una propiedad que habla bien del poder discriminativo de un conjunto de características es la *reconstructibilidad*, esto es, cuán factible es reconstruir el gesto a partir dichas características.

Si solo se consideran las características del cuadro 4.1, es claro que varias secuencias de puntos distintas pueden dar origen a un mismo conjunto distinto de características. Sin ir más lejos, un mismo conjunto de puntos puede ser interpretado como secuencias distintas al recorrer los puntos en distintos órdenes, sin embargo, las distintas trayectorias tendrán la misma altura, anchura, covarianza, etc.

Este razonamiento lleva a buscar características que capturen mejor lo que interesa de una trayectoria descrita en el espacio. El cuadro 4.2 presenta algunas características que cumplen estos requisitos.

La imagen binaria (cuyas componentes se llaman `HIST2Dij`), es la que más fielmente captura la forma general de cada gesto. Para construir esta característica, se marcan los puntos de la secuencia (x, y, v_x, v_y, t) dentro de una grilla o cuadriculado (en esta memoria, de 8×8) y luego se interpola entre puntos consecutivos, para lo cual se puede usar cualquiera de los algoritmos conocidos en computación gráfica para dibujar líneas en un cuadriculado (en este caso, por sencillez y rapidez, se usa el algoritmo de Bresenham [13]).

Un aspecto que la imagen binaria no considera es el orden en que se recorren los puntos. Una línea horizontal trazada de izquierda a derecha es un gesto distinto a una línea horizontal trazada de derecha a izquierda (y la distinción es importante si cada gesto representa una orden para moverse en una dirección distinta). Por esto, se introducen las características de códigos de dirección `SLOPE00,SLOPE01,...`. Éstas corresponden a las direcciones del movimiento en distintos instantes de la trayectoria. Concretamente, se elige una cantidad n_{slope} ($n_{slope} = 10$ en esta memoria) y se toman $n_{slope} + 1$ puntos de la trayectoria, de modo que el primer y último punto coincidan con el primero y último de la trayectoria, y la distancia entre puntos consecutivos sea aproximadamente igual. El i-ésimo valor `SLOPEi` se obtiene como la pendiente entre los puntos i e $i + 1$, discretizada usando el código más parecido de la figura 4.6.

Nombre	Descripción
`HISTX00,HISTX01,...`	Histograma de puntos en el eje x. Este histograma se obtiene dividiendo el eje x en k partes, comprendidas entre la mínima y máxima coordenada x, y luego calculando el porcentaje de puntos que caen en cada franja.
`HISTY00,HISTY01,...`	Análogo del anterior en el eje y.
`HIST2D00,HIST2D01,...`	Imagen binaria. Se superpone una grilla de dimensiones fijas sobre el *bounding-box* del gesto y se marca en cada celda si la trayectoria pasa o no por ella.
`SLOPE00,SLOPE01,...`	Códigos de dirección. Se toma cierta cantidad c de puntos en la secuencia (equiespaciadamente) y se mide la pendiente de la recta que va de cada punto al siguiente. La pendiente se codifica usando un número del 0 al 7, según el diagrama 4.6, dando origen a esta característica.

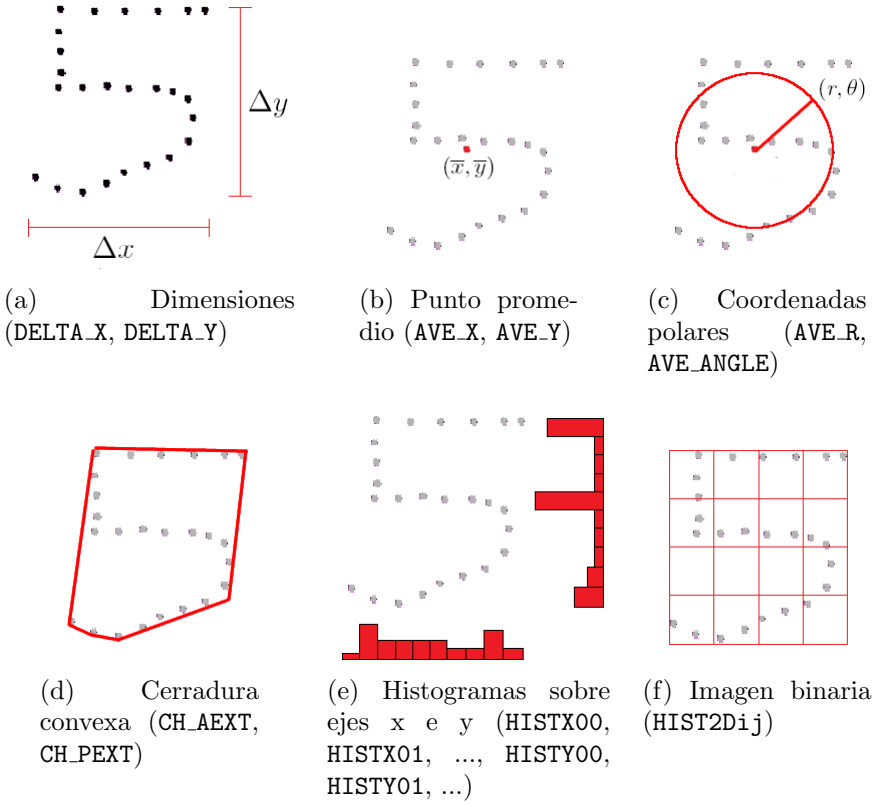
Cuadro 4.2: Características más descriptivas

La decisión final de cuáles características usar considera los criterios explicados de tamaño, forma general y dirección instantánea de la trayectoria para elegir tres conjuntos de características:

Tamaño	`DELTA_X`, `DELTA_Y`
Imagen binaria	`HIST2Dij`
Códigos de dirección	`SLOPE00,SLOPE01,...`

La razón para elegir estos tres conjuntos de características es que cada uno da cuenta aspectos distintivos del gesto de manera complementaria. La imagen binaria da una representación que facilita la reconstrucción del gesto, pero no permite saber en qué orden se hizo la trayectoria. Al contrario, los códigos de dirección permiten conocer el orden de ejecución de la trayectoria, pero, como en algunos casos no son suficientes para distinguir entre gestos (como el 0 y el 6). Finalmente, es necesario tener alguna medida del tamaño de los gestos, que no existe en ninguna de los dos conjuntos de características anteriores.

Puede verse que la selección de características es, hasta ahora, puramente manual. En los objetivos (sección 1.1) se plantea la necesidad de encontrar características descriptivas usando métricas existentes. Sin embargo, dicho objetivo tenía en mente las características del cuadro 4.1, que eran las usadas al principio de este trabajo. En el desarrollo de la memoria, se llegó a la conclusión de que lo realmente necesario era idear un nuevo conjunto de características, y el resultado es el expuesto en el cuadro anterior.



(a) Dimensiones (`DELTA_X`, `DELTA_Y`)

(b) Punto promedio (`AVE_X`, `AVE_Y`)

(c) Coordenadas polares (`AVE_R`, `AVE_ANGLE`)

(d) Cerradura convexa (`CH_AEXT`, `CH_PEXT`)

(e) Histogramas sobre ejes x e y (`HISTX00`, `HISTX01`, ..., `HISTY00`, `HISTY01`, ...)

(f) Imagen binaria (`HIST2Dij`)

Figura 4.7: Algunas de las características consideradas

Normalización

Uno de los factores de éxito en los problemas de reconocimiento de caracteres es la normalización [33], entendida como una transformación geométrica que intenta igualar la apariencia de caracteres similares. Sin embargo, al aplicar este procedimiento se corre el riesgo de introducir distorsiones y de suprimir rasgos distintivos de un caracter. Este problema es abordado en [41].

El uso de una imagen binaria asimila el problema de detección de gestos al de reconocimiento de caracteres, haciendo necesaria también la normalización. Para minimizar las distorsiones, se toman en cuenta las advertencias de Liu et al. [41], quienes sostienen la importancia de normalizar según la razón de aspecto del caracter. En particular, aquellos caracteres que tienen menores razones de aspecto tienen mayor propensión a resultar deformados.

En esta memoria, la normalización de una secuencia queda definida por la elección del rectángulo a partir del cual se calculará la imagen binaria. La calidad de la normalización variará dependiendo de dónde se centre este cuadriculado y qué dimensiones tenga. La idea más simple es usar un rectángulo ceñido a la figura, pero esto introduce deformaciones en gestos angostos como el número 1. Por lo tanto, se adopta la regla de usar un rectángulo ceñido solo para las subsecuencias que tengan una razón de aspecto suficientemente grande (por ejemplo, 0,35). Para las subsecuencias con razón de aspecto menor, se usa un cuadrado cuyo lado es igual al lado de medida mayor del rectángulo ceñido respectivo. Véase la figura

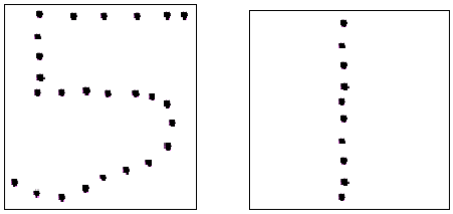


Figura 4.8: Elección de un rectángulo para el cálculo de la imagen binaria. El rectángulo ceñido al 5 tiene una forma cuadrada. En cambio, el rectángulo ceñido al 1 es demasiado angosto y la imagen binaria quedaría irreconocible, por lo que se opta por usar un cuadrado cuyo lado es igual a la altura del 1.

4.8

4.2.3. Clasificadores

Una vez definidas las características, se procedió a evaluar el rendimiento de distintos clasificadores. En primera instancia, se planteó el problema de entrenar un clasificador multiclase, capaz de discriminar entre los 10 gestos. Para esto, se evaluaron los clasificadores Naïve Bayes, SVM y C4.5. Los resultados de la clasificación fueron muy altos, incluso cuando se usaron las características consideradas más burdas según la discusión de la sección 4.2.2.

En base a estos resultados, publicados en [24], se decidió el uso de clasificadores Naïve Bayes, por ser los más rápidos y tener las mejores tasas de clasificaciones correctas. Además, Naïve Bayes calcula probabilidades $P(C|S)$ que permiten tener grados de certeza de si la secuencia S es un gesto de clase C; tales probabilidades permiten, por lo tanto, encontrar los momentos de verdadera ejecución de un gesto y descartar las secuencias que no contienen ningún gesto.

La capacidad de Naïve Bayes de entregar probabilidades contrasta con el hiperplano separador de SVM y el árbol de decisión C4.5, los cuales, por su naturaleza, se decidirán por una clase cualquiera sea su entrada. Sin embargo, el clasificador Naïve Bayes multiclase tampoco escapa por completo a este defecto, ya que las probabilidades $P(C|S)$, con S fijo, deben sumar 1, por lo que necesariamente hay al menos una clase con probabilidad no nula y un máximo.

Esto lleva a formular el problema de construir un clasificador capaz de decidir de qué clase de gesto es su entrada pero también capaz de abstenerse si su entrada no es ningún gesto. Otra manera de formular este problema es permitir una clase que sea "no gesto". La solución propuesta en esta memoria consiste en construir múltiples clasificadores binarios, uno por cada gesto, cada uno de los cuales es entrenado para discriminar un gesto dado del resto de los gestos. Si la entrada es una secuencia sin significado, se espera que ninguno de los clasificadores entregue un resultado positivo.

El clasificador binario especializado en el gesto g es entrenado con las instancias del gesto g como ejemplares positivos y con el resto de los ejemplares de entrenamiento como

	G0	NONG0
G0	30	10
NONG0	0	260

Correctos: 290 (96.67 %)
Incorrectos: 10 (3.33 %)

	G1	NONG1
G1	30	0
NONG1	0	270

Correctos: 300 (100 %)
Incorrectos: 0 (0 %)

	G2	NONG2
G2	30	1
NONG2	0	269

Correctos: 299 (99.67 %)
Incorrectos: 1 (0.33 %)

	G3	NONG3
G3	30	2
NONG3	0	268

Correctos: 298 (99.33 %)
Incorrectos: 2 (0.67 %)

	G4	NONG4
G4	30	0
NONG4	0	270

Correctos: 300 (100 %)
Incorrectos: 0 (0 %)

	G5	NONG5
G5	30	11
NONG5	0	259

Correctos: 289 (96.33 %)
Incorrectos: 11 (3.67 %)

	G6	NONG6
G6	30	14
NONG6	0	256

Correctos: 286 (95.33 %)
Incorrectos: 14 (4.66 %)

	G7	NONG7
G7	30	0
NONG7	0	270

Correctos: 300 (100 %)
Incorrectos: 0 (0 %)

	G8	NONG8
G8	30	1
NONG8	0	269

Correctos: 299 (99.67 %)
Incorrectos: 1 (0.33 %)

	G9	NONG9
G9	29	2
NONG9	1	268

Correctos: 297 (99 %)
Incorrectos: 3 (1 %)

Figura 4.9: Rendimiento de los clasificadores. La evaluación se realiza mediante validación cruzada con 10 rondas. Para el clasificador i hay 30 ejemplares positivos del gesto i (etiquetados como `Gi`) y 270 negativos (etiquetados como `NONGi`). Cada matriz de confusión muestra en sus columnas las etiquetas reales y en sus filas la predicción del clasificador. Debajo de cada matriz se consignan los porcentajes de clasificaciones correctas e incorrectas. Puede observarse que los ejemplares positivos son en su gran mayoría clasificados correctamente, pero que algunos ejemplares negativos también aparecen como positivos.

ejemplares negativos. En ALON-TRAINING, cada clasificador tiene 30 ejemplares positivos y 270 negativos. La figura 4.9 muestra el rendimiento de los clasificadores entrenados para los gestos del 0 al 9. En tanto, la figura 4.10 muestra los parámetros del clasificador Naïve Bayes para un gesto en particular.

Cabe acotar que descomponer un clasificador multiclase en clasificadores binarios es un problema conocido. La descomposición expuesta es del tipo *one-against-all* o uno contra todos. Una alternativa es la descomposición *all-pairs* (todos los pares), que entrena un clasificador por cada par de clases posible, ignorando los ejemplares que no pertenezcan a ninguna de las dos clases. Allwein et al. [2] ofrecen una descripción general del problema.

```
                Class
Attribute          G6   NONG6
                 (0.1)  (0.9)
============================
DELTA_X
  mean           1.6555  1.472
  std. dev.      0.2238 0.5029
  weight sum         30    270
  precision      0.0072 0.0072

DELTA_Y
  mean             2.22 2.1783
  std. dev.      0.1982 0.3722
  weight sum         30    270
  precision      0.0072 0.0072

HIST2D00
  0                27.0  153.0
  1                 5.0  119.0
  [total]          32.0  272.0

HIST2D01
  0                23.0   90.0
  1                 9.0  182.0
  [total]          32.0  272.0

HIST2D02
  0                14.0   68.0
  1                18.0  204.0
  [total]          32.0  272.0

[...]

HIST2D77
  0                11.0  125.0
  1                21.0  147.0
  [total]          32.0  272.0

SLOPE00
  0                11.0   76.0
  1                 7.0   30.0
  2                 1.0   12.0
  3                 1.0   43.0
  4                 1.0   42.0
  5                 1.0   12.0
  6                 1.0   35.0
  7                15.0   28.0
  [total]          38.0  278.0

SLOPE01
  0                 6.0   63.0
  1                 1.0    7.0
  2                 1.0    2.0
  3                 1.0   16.0
  4                 1.0   73.0
  5                 1.0   10.0
  6                 2.0   53.0
  7                25.0   54.0
  [total]          38.0  278.0

[...]

SLOPE09
  0                24.0   40.0
  1                 1.0   36.0
  2                 1.0   22.0
  3                 1.0    5.0
  4                 1.0   62.0
  5                 1.0    2.0
  6                 1.0   85.0
  7                 8.0   26.0
  [total]          38.0  278.0
```

Figura 4.10: Una muestra de los parámetros del clasificador (acá, para el gesto 6) según el formato de Weka [65]. Para las variables continuas (`DELTA_X`, `DELTA_Y`) se calcula la media y la desviación estándar en las clases gesto y no-gesto. En el caso de las variables discretas (el resto), se calcula la distribución de los posibles valores en cada clase (gesto/no gesto).

4.3. Reconocimiento en línea

Si bien los clasificadores de la sección 4.2.3 obtienen muy altas tasas de reconocimiento sobre gestos segmentados, esto no es suficiente para garantizar el reconocimiento en línea, donde no se conoce ni el comienzo ni el final de los gestos. A continuación, se describe el procedimiento usado para buscar, por medio de estos clasificadores, las ocurrencias de un gesto en una secuencia arbitraria de posiciones de la mano.

4.3.1. Estructura del reconocedor en línea

El sistema de reconocimiento de gestos es un módulo que recibe continuamente información de la posición de la mano del usuario. Su salida, también continua, señala si se ha detectado un gesto y, en caso de que se haya detectado, cuál gesto es y entre qué cuadros fue ejecutado.

El algoritmo de reconocimiento se divide en tres fases, las cuales se aplican por cada cuadro entrante:

- En la etapa de **generación de candidatos**, se aplican de manera continua los clasificadores binarios a subsecuencias temporales del movimiento de la mano. Al encontrarse subsecuencias con alto puntaje, se reconoce la existencia de un candidato.
- En la etapa de **evaluación de candidatos**, se verifica la plausibilidad de los candidatos entrantes y se eliminan aquellos candidatos que están contenidos dentro de otros.
- Finalmente, en la etapa de **evaluación de intencionalidad**, se analiza la velocidad actual de la mano para predecir la intención actual del usuario, ya sea continuar haciendo un gesto o darlo por terminado. En virtud de esta información se decide dar por reconocido un gesto o bien continuar con la generación de candidatos.

En las siguientes secciones se detallan estas tres fases del algoritmo, cuyo esquema general está ilustrado en la figura 4.11.

4.3.2. Generación de candidatos

El módulo de generación de candidatos realiza un análisis de subsecuencias y como resultado extrae candidatos a gestos (caracterizados por una clase de gesto, un cuadro inicial y un cuadro final).

Este módulo almacena los vectores (x, y, v_x, v_y, t) de los últimos N cuadros (por ejemplo, $N = 50$). Si llega el $(N + 1)$-ésimo cuadro, se elimina el más antiguo.

Dado un gesto g, un cuadro inicial i y cuadro final f, es posible calcular, mediante los clasificadores entrenados, la probabilidad $P^g_{i,f}$ de que haya ocurrido g en la subsecuencia

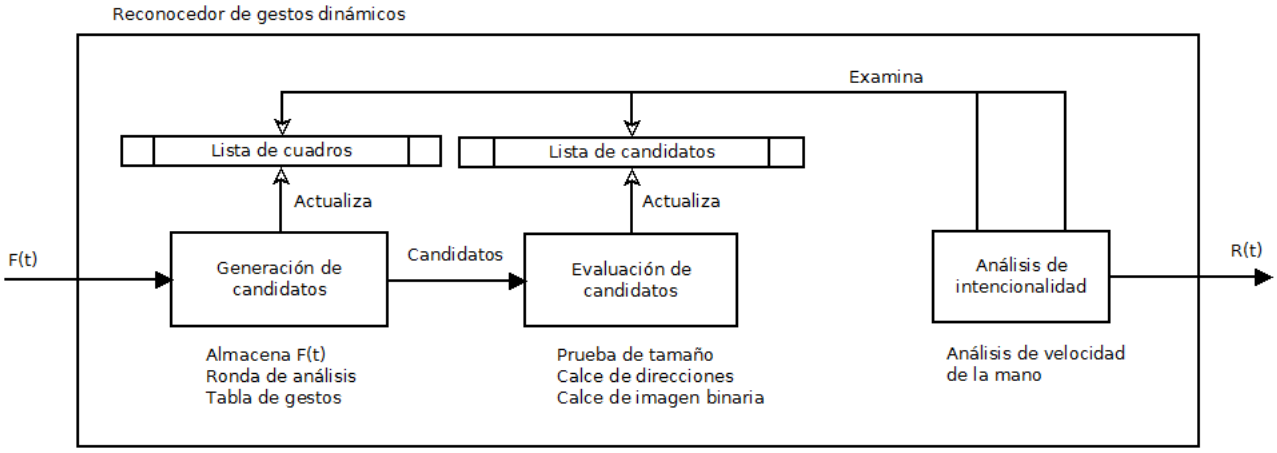


Figura 4.11: Diagrama general. $F(t)$ es la información del cuadro entrante y $R(t)$ es la información de reconocimiento resultante (un bit que indica si ha ocurrido un gesto y, si es pertinente, qué gesto ha sido ejecutado y entre cuáles cuadros). El módulo de generación de candidatos actualiza la lista de cuadros y realiza análisis sobre él. Los candidatos resultantes son usados por el módulo de evaluación de candidatos para actualizar la lista de candidatos. Finalmente, el módulo de análisis de intencionalidad examina los datos obtenidos para hacer una decisión de si ha ocurrido un gesto o no.

delimitada por i y f. Luego, el problema de encontrar la ocurrencia de un gesto g se puede plantear como el problema de encontrar —en línea— máximos de la función de dos variables $(i, f) \to P^g_{i,f}$. La búsqueda es en línea ya que a solo a medida que van llegando cuadros es posible saber la forma de esta función. La búsqueda de esto máximos se hace resumiendo la información obtenida dado el último cuadro, y luego combinando esta información con la de cuadros anteriores según se detalla a continuación.

Suponiendo una numeración correlativa de los cuadros entrantes, sea s el número del primer cuadro almacenado y c el del último cuadro entrante. Tras almacenar c, comienza una **ronda de análisis**, donde se examinan todas las subsecuencias que terminan con c (salvo aquellas demasiado cortas, de largo menor que un valor N_{min} fijado de antemano). Cada subsecuencia es convertida en un vector de características, que luego es entregado a cada uno de los N_g clasificadores. El resultado es un conjunto de valores $P^g_{i,c}$ donde i varía entre s y $c - N_{min}$. La ronda de análisis está ilustrada en la figura 4.12.

Como la secuencia completa es de largo $O(N)$, habrá $O(N)$ subsecuencias analizadas que se convertirán en $O(N)$ probabilidades por gesto y $O(NN_g)$ probabilidades en total. Para condensar esta gran cantidad de datos, se toma el máximo M^g_c de cada una de las N_g secuencias de probabilidades $\{P^g_{i,c}\}^{c-N_{min}}_{i=s}$. En la figura 4.12, esto corresponde a tomar el máximo de cada fila (posiblemente usando una ventana móvil para lograr mayor robustez). Como regla, si hay una secuencia de varios máximos idénticos, se elige el que tenga el cuadro inicial más posterior. Este máximo, de por sí, representa la mejor hipótesis de la ejecución del gesto g, suponiendo que este ha terminado en el último cuadro.

Sin embargo, no es suficiente que el puntaje sea alto después de un cuadro: es necesario que este sea consistentemente alto durante varios cuadros. Por lo tanto, cada máximo M^g_c, es llevado a la **tabla de gestos** (figura 4.13), donde se mantienen, por cada gesto, los máximos

de los últimos k cuadros, es decir, M^g_{c-k+1}, M^g_{c-k+2}, ..., M^g_c. Dentro de esta tabla se mantiene, además, el promedio móvil de estos valores PM^g_c, el cual en definitiva representa la mejor predicción de que se haya ejecutado el gesto g dado el último cuadro. En esta memoria, se escogió $k = 5$ para el tamaño de la tabla de gestos.

Si el puntaje PM^g_c supera un umbral (prefijado y distinto para cada gesto), se declara la existencia de un candidato para el gesto g, el cual se asocia con el cuadro inicial y final del mayor de los máximos M^g_j $(j = c - k + 1, \ldots, c)$.

4.3.3. Evaluación de candidatos

Verificación de candidatos

Los clasificadores Naïve Bayes tienden a arrojar muchos falsos positivos, incluso tras pasar por el filtrado de la tabla de gestos. Por esto, el resultado de la tabla de gestos debe verse solo como una primera etapa de generación de candidatos. Una segunda etapa descarta aquellos candidatos que no son capaces de pasar pruebas más rigurosas.

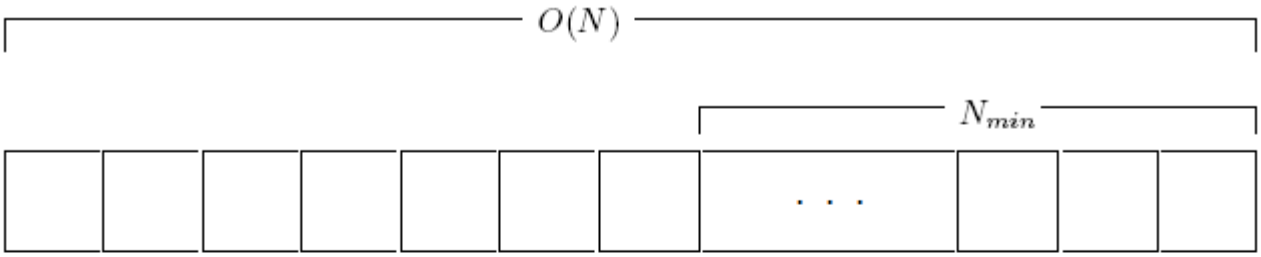
Las pruebas elegidas para verificar los candidatos consideran las mismas características que el vector de características: tamaño, dirección instantánea de la trayectoria y forma global (sección 4.2.2). Mientras que el clasificador Naïve Bayes hace una evaluación general de los méritos de una subsecuencia, las pruebas que se describen a continuación buscan un calce más preciso.

Estas pruebas son aplicadas en cascada, deteniéndose su aplicación cuando una de ellas logra descartar la validez de una subsecuencia:

1. **Prueba de tamaño**: se espera que la altura y anchura del gesto se encuentren dentro de ciertas desviaciones estándar de los gestos del conjunto de entrenamiento. Esta prueba impide que gestos demasiado pequeños sean reconocidos como legítimos, especialmente cuando, por azar, oscilaciones pequeñas de la mano coinciden con un gesto existente.

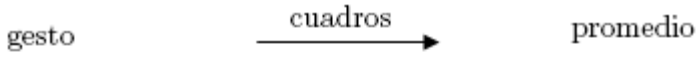
2. **Calce de direcciones**: por cada gesto se determina, de antemano y a partir de los parámetros de los clasificadores binarios, una plantilla que representa al gesto como una lista de códigos de direcciones (véase la figura 4.14). Cada candidato se compara con la plantilla correspondiente mediante una distancia de edición con pesos (sección 2.5.3). Una idea similar es usada en [57], también con el fin de reconocer gestos. Aquellos candidatos cuya distancia a la plantilla sea mayor que cierto umbral son rechazados.

3. **Calce de imagen**: como en el caso de la prueba anterior, se extrae una plantilla de los parámetros de cada clasificador tomando para cada pixel el valor más frecuente. Estas plantillas son mostradas en la figura 4.15. Los candidatos son comparados con las plantillas mediante la distancia de Hausdorff (sección 2.5.2), y son descartados si dicha distancia es mayor que cierto umbral.



0	0.00	0.00	0.00	0.00	0.00	0.00	0.00
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.00	0.00	0.00	0.00	0.00	0.00	0.00
3	0.05	0.96	0.99	0.67	0.03	0.10	0.02
4	0.00	0.00	0.00	0.01	0.17	0.81	0.31
5	0.00	0.01	0.00	0.00	0.00	0.00	0.00
6	0.01	0.00	0.00	0.02	0.00	0.00	0.00
7	0.00	0.00	0.00	0.00	0.00	0.00	0.00
8	0.00	0.00	0.00	0.00	0.00	0.00	0.00
9	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Figura 4.12: El sistema guarda hasta N cuadros. Cada vez que llega un nuevo cuadro, se analizan todas sus subsecuencias que tengan al menos N_{min} cuadros. Por cada gesto g se genera una serie de probabilidades $\{P^g_{i,c}\}_{i=s}^{c-N_{min}}$ (en la figura, cada $P^g_{i,c}$ va debajo del cuadro de inicio i correspondiente). Los máximos en este puntaje (marcados en cada fila) indican una alta probabilidad de ejecución del gesto correspondiente.



0	0.00	0.00	0.00	0.00	0.00	0.00
1	0.98	0.01	0.00	0.00	0.20	0.24
2	0.00	0.00	0.00	0.00	0.00	0.00
3	0.00	0.01	0.03	0.86	0.99	0.38
4	0.00	0.00	0.00	0.02	0.81	0.17
5	0.00	0.01	0.00	0.00	0.00	0.00
6	0.00	0.00	0.00	0.00	0.02	0.00
7	0.00	0.00	0.00	0.00	0.00	0.00
8	0.00	0.00	0.00	0.00	0.00	0.00
9	0.00	0.00	0.00	0.00	0.00	0.00

Figura 4.13: Tabla de gestos. Para examinar los datos recolectados a lo largo de varios cuadros, se almacenan, por cada gesto, los últimos k máximos $\{M^g_j\}_{j=c-k+1}^{c}$ (la tabla corresponde a $k = 5$). El valor de la última columna es el promedio móvil PM^g_c, que es el que finalmente se usa para tomar la decisión de declarar la existencia de un candidato.

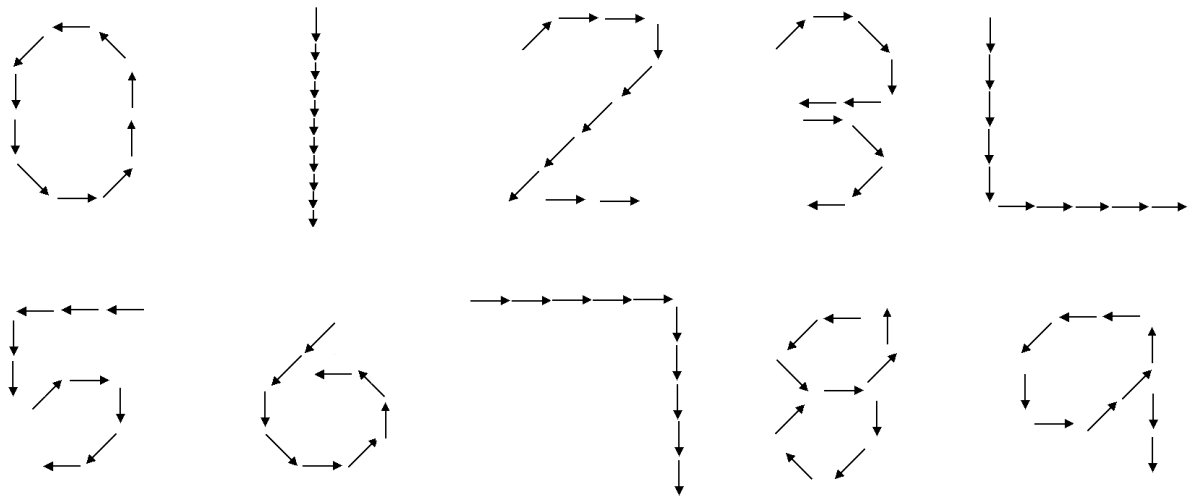


Figura 4.14: Plantillas de códigos de direcciones. Su representación interna es una cadena de códigos de dirección (números enteros) como aquellos de la figura 4.6. Estas plantillas son construidas a partir de los parámetros del respectivo clasificador (ver figura 4.10) tomando la dirección que es más frecuente en cada etapa del movimiento.

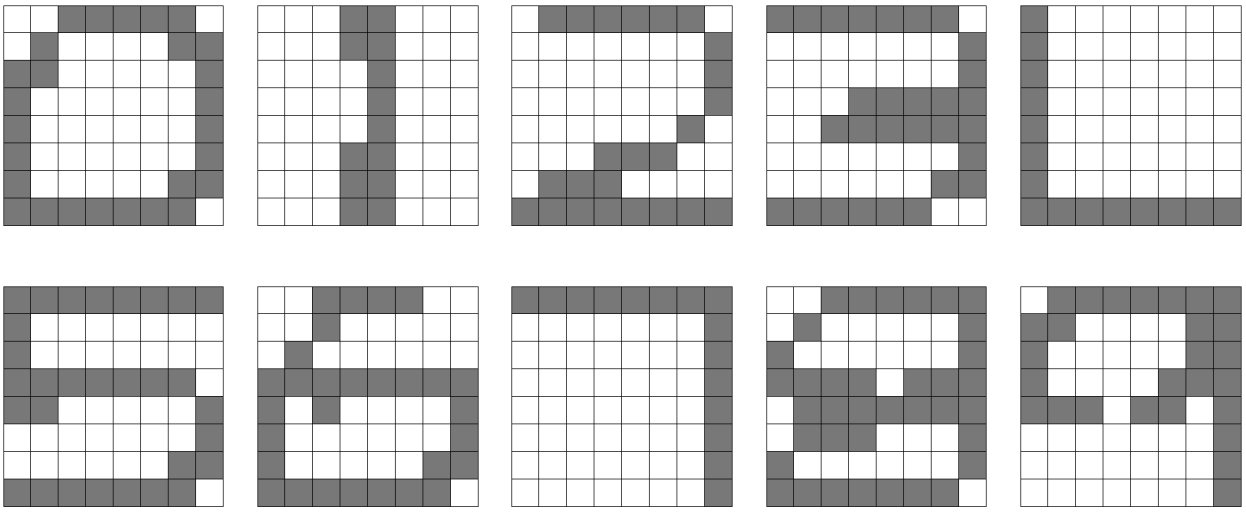


Figura 4.15: Plantillas de imágenes binarias. Estas plantillas son construidas a partir de los parámetros del respectivo clasificador (ver figura 4.10) tomando, para cada pixel, el valor más frecuente en el conjunto de entrenamiento.

Inserción de un nuevo candidato y el problema de subgestos

Un candidato que ha pasado por las pruebas de verificación queda asociado con el puntaje

$$\frac{1}{(1+\mathrm{DE})(1+\mathrm{DH})} \tag{4.2}$$

donde DE es la distancia de edición calculada en la prueba de calce de direcciones y DH es la distancia de Hausdorff calculada en la prueba de calce de imagen. El puntaje asociado al clasificador Naïve Bayes, en cambio, es descartado (estudios [8] afirman que este puntaje suele estar mal calibrado, es decir, que su valor tiende a estar muy por encima o por debajo de su valor real, fenómeno debido a la suposición ingenua de independencia entre las variables que está al centro de Naïve Bayes).

En el sistema desarrollado, como regla, por cada gesto se mantiene como máximo un candidato, de modo que si ya existe un candidato del mismo tipo de gesto, solo se mantiene el de mejor puntaje (calculado con la fórmula (4.2)).

El trabajo de Alon [3] llama la atención sobre el problema de subgestos: un gesto g puede ser reconocido erróneamente en vez de un G que lo contiene, y que era la verdadera intención del usuario. Ejemplos son el 5 (subgesto de 8) y el 1 (subgesto de 4 y 7).

Para abordar el problema de subgestos, un candidato entrante C es comparado con cada candidato existente C_i. Si C está contenido en algún C_i, C no será agregado. En caso contrario, C será agregado como candidato y se realiza una segunda iteración para revisar si C contiene a algún C_i. En tal caso, C_i es eliminado. Finalmente, C es agregado a la lista de candidatos.

A diferencia del trabajo de Alon, la relación "estar contenido" se refiere únicamente a la inclusión temporal, no a la inclusión geométrica como se hace en dicho trabajo. Es decir, en esta memoria se descarta cualquier candidato que se encuentre *temporalmente* contenido dentro de otro candidato (incluso si geométricamente no son subgestos), mientras que en el trabajo de Alon se descarta cualquier candidato que *geométricamente* sea subgesto de otro (incluso si no tienen una relación de inclusión temporal). En dicho trabajo existe, además, una regla adicional que prohibe que cualquier par de candidatos coincidan temporalmente (esto incluye cualquier coincidencia temporal, no tan solo inclusiones).

En definitiva, en esta memoria hay estrictamente menos reglas para el manejo de candidatos que en Alon. Esto se considera una necesidad, puesto que el sistema desarrollado acá no existen mecanismos que permitan inferir la relación de inclusión geométrica como en Alon et al.

4.3.4. Evaluación de intencionalidad del usuario

Una vez que un candidato pasa las pruebas de verificación descritas, su validez queda comprobada y por lo tanto el sistema podría declararlo oficialmente como el gesto reconocido,

borrar los cuadros almacenados y reiniciar el proceso de reconocimiento. Usando este criterio de término, no sería necesaria la mantención de una lista de candidatos.

La razón de usar una lista de candidatos es la existencia de gestos que se parecen unos a otros y, en particular, la existencia de subgestos. Si el usuario hace el gesto 1, puede ser que su intención sea hacer un 1, o bien completar un 4 haciendo una línea horizontal adicional. Tras hacer el 1, el sistema no tiene información para decidir cuál es la intención del usuario, por lo que lo mejor que puede hacer es tomar nota del 1 que ha sido reconocido hasta el momento y luego esperar.

Yendo más lejos con el mismo ejemplo, podría ocurrir que el usuario haga una línea horizontal para completar un número 4. Al aparecer este candidato, el algoritmo de manejo de subgestos eliminaría el candidato 1 y lo reemplazaría por el 4, pero aún así no podría declarar inmediatamente este gesto como el definitivo, pues el 4 es subgesto del 5.

Por lo tanto, la mantención de la lista de candidatos reconoce el hecho de que no se pueden tomar decisiones apresuradas basadas únicamente en el último cuadro para realizar el reconocimiento de gestos. Sin embargo, aún falta definir un criterio para decidir en qué momento se tomará un candidato de la lista para declararlo oficialmente como el gesto detectado por el sistema.

Un primer criterio puesto a prueba consiste en esperar una cantidad de cuadros w cada vez que llega un nuevo candidato. Una vez que han transcurrido w cuadros, se elige el mejor candidato y se reinicia el reconocimiento. Si llega un nuevo candidato, la espera de w cuadros comienza de nuevo.

El problema con este criterio es que la cantidad w de cuadros antes de la declaración es fija. Si hay un gesto G que posee un subgesto g, el subgesto g aparecerá primero como candidato y el éxito del reconocimiento dependerá de que G aparezca como candidato dentro de los w cuadros disponibles. No se considera si la intención del usuario es seguir haciendo un gesto, retirar la mano, o tener la mano quieta en espera del reconocimiento.

Para tomar en cuenta esta intencionalidad del usuario, se usa como indicador la velocidad de la mano, en base a la cual se mantiene en una variable f el "cuadro de declaración", es decir, el cuadro en que el sistema elegirá al mejor gesto. Al aparecer un candidato, el cuadro de declaración es actualizado a $f = m + w$ donde m es el cuadro final del candidato y w es un número de cuadros de espera fijo.

Luego, se evalúa la velocidad actual de la mano. Si esta es demasiado alta o demasiado baja, esto se toma como señal de que el gesto ha terminado, por lo que se adelanta el cuadro de declaración restando un valor positivo a f.

En cambio, si no se detecta ninguna de estas dos condiciones, se considera que el usuario tiene intenciones de completar un gesto (es decir, que el candidato a gesto actual es solo un subgesto). Para impedir la declaración del gesto, se agrega un valor positivo a f. Este valor puede ser igual a 1, de modo mantener alejado el cuadro de declaración a una distancia constante, o puede ser mayor que 1, de modo de aumentar esta distancia.

Una vez que el número del cuadro actual es mayor o igual que f, se elige al candidato con

el mejor puntaje, se borra la lista de candidatos, la lista de cuadros almacenada y la tabla de gestos de modo de reiniciar el proceso de reconocimiento.

Se elige w pensando en dar suficiente tiempo al movimiento de la mano para revelar su intención. Un tiempo de aproximadamente 1 segundo resulta adecuado para este fin, tomando en cuenta que los gestos más largos duran unos 3-4 segundos. Para las pruebas en video, a 30 cuadros por segundo, un valor de $w = 20 - 30$ resulta satisfactorio, mientras que las pruebas en tiempo real, funcionando a menos cuadros por segundo, necesitarán valores menores.

4.3.5. Complejidad

El costo del algoritmo completo está dominado por la fase de generación de candidatos. La fase de evaluación de candidatos se ejecuta esporádicamente, solo cuando aparecen candidatos, mientras que la fase de evaluación de intencionalidad se limita a mantener la velocidad de la mano y sumar o restar un valor al cuadro de declaración.

La complejidad de una ejecución del algoritmo de generación de candidatos es $O(N(C_{carac} + C_{clasif}))$ donde C_{carac} es el costo de extracción de características y C_{clasif} es el costo de clasificación del vector de características en los N_g clasificadores, mientras que el factor N da cuenta de la iteración sobre la secuencia completa de datos.

El costo del cálculo de características está dominado por la construcción la imagen binaria, que implica recorrer $O(N)$ puntos para estamparlos sobre el cuadriculado y luego unirlos. La extracción de códigos de dirección implica igualmente un recorrido de la secuencia de $O(N)$ puntos en busca de aquellos equidistantes. Luego, el costo C_{carac} es $O(N)$.

Por tratarse de clasificadores Naïve Bayes, la evaluación de un vector de características toma tiempo igual al número de características. Considerando además que hay tantos clasificadores como gestos, si N_c es el número de características, C_{clasif} es $O(N_g N_c)$.

En conclusión, la complejidad teórica del algoritmo es $O(N(N + N_g N_c))$.

Optimización

Para mejorar el rendimiento de la generación de candidatos, es preciso proceder de manera incremental, aprovechando lo mejor posible los cálculos de iteraciones pasadas para hacer nuevos cálculos. Al clasificar múltiples subsecuencias durante una ronda de análisis, las diferencias entre las características de una subsecuencia y la siguiente (obtenida removiendo un cuadro) pueden ser mínimas, y por lo tanto el resultado de aplicar Naïve Bayes también será muy parecido.

Idealmente, la extracción de características durante la ronda de análisis debería basarse en un único cálculo inicial más una actualización en tiempo $O(1)$ en las iteraciones siguientes, en vez un cálculo desde cero cada vez. Entonces, C_{carac} sería $O(1)$ amortizado. Por otro lado, la clasificación podría seguir el mismo principio incremental, haciendo una clasificación inicial y luego actualizando en tiempo constante el resultado de la clasificación basándose solamente

en aquellos términos del vector de características que cambien, con lo cual C_{clasif} sería $O(N_g)$ amortizado. En resumen, la complejidad podría llegar a ser $O(NN_g)$, es decir, un cálculo de la tabla desplegada en la figura 4.12 hecho de la manera más eficiente posible.

Un análisis de la ejecución del programa mediante un profiler revela que, en efecto, la clasificación mediante Naïve Bayes es uno de los ítems más costosos, por lo que el esfuerzo de optimización se centra en este aspecto.

Para mejorar el rendimiento de la clasificación, la clave es observar que buena parte de los cálculos usados en la evaluación de una subsecuencia pueden ser reutilizados en la siguiente subsecuencia (obtenida removiendo un cuadro), puesto que entre los respectivos vectores de características existirán muy pocas diferencias, a veces ninguna.

Más detalladamente, se observa que, al clasificar un vector de características S, se realiza el cálculo

$$P(C_i|S) = KP(C_i)\prod_{j=1}^{n} P(S_j|C_i)$$

donde K es la constante de normalización. Tomando logaritmo, lo cual es habitual al hacer este cálculo, ya que se evita el *underflow* de punto flotante debido a la multiplicación de muchos números pequeños, se obtiene la expresión

$$\log(P(C_i|S)) = \log(P(C_i)) + \sum_{j=1}^{n} \log(P(S_j|C_i)) + K' \tag{4.3}$$

donde K' es constante. Al evaluar $\log(P(C_i|S))$, es necesario calcular los términos $\log(P(C_i))$ y $\log(P(S_j|C_i))$ para $j = 1 \ldots n$. Si en una evaluación subsiguiente se clasifica un vector de características S' (parecido a S), muchas de las componentes de S' serán iguales a las de S, y por lo tanto los términos $\log(P(S'_j|C_i))$ serán también los mismos. En consecuencia, los únicos términos que habrá que calcular para obtener el nuevo valor de la expresión (4.3) serán aquellos que difieran entre S y S'.

El algoritmo que aprovecha estas observaciones es inicializado con los siguientes datos:

```
// prevScores guarda los resultados del clasificador para cada clase
// Es decir, acá se almacenará log(P(C_i|S)) con S = último vector evaluado
// (inicialmente se asigna log(P(C_i)))
para todo i (clase)
    prevScores[i] = log(elementos de clase i/total conjunto entrenamiento)

// prevTerms almacena los términos log(P(S_j|C_i)) de la sumatoria
// (inicialmente no se ha clasificado ningún vector, y se pone en 0)
para todo i (clase), j (característica)
    prevTerms[i][j] = 0

// prevVect guarda las componentes del último vector clasificado
// (inicialmente no existe, por lo que se usa NaN - not a number)
para todo j (característica)
    prevVect[j] = NaN
```

Luego, al analizar una nueva secuencia S (probablemente parecida a la almacenada en `prevVect`), basta realizar los siguientes cálculos:

```
// Inicializar probs, donde se guarda el resultado,
// con los puntajes de la iteración anterior
para todo i (clase)
    probs[i] = prevScores[i]

// Actualizar probabilidades
para todo j (característica), i (clase)
    si S[j] == prevVect[j]
        // si no ha cambiado un término
        // no hace falta modificar probs[i]
    sino
        // ha cambiado un término

        // calcular nuevo término
        sea tmp = log(P(S[j]|C[i]))

        // quitar término antiguo y agregar nuevo
        probs[i] -= prevTerms[i][j]
        probs[i] += tmp

        // guardar nuevo término
        prevTerms[i][j] = tmp

// Guardar valores para próxima iteración:

// Guardar valores del vector recién clasificado
para todo j (característica)
    prevVect[j] = S[j]

// Guardar resultados de la última clasificación
para todo i (clase)
    prevScores[i] = probs[i]
```

Los valores almacenados en `probs[i]` son iguales a $\log(P(C_i|S))$ salvo por una constante aditiva, por lo que, para obtener las verdaderas probabilidades, solo falta aplicar la exponencial y luego normalizar (dividiendo por la suma).

La aplicación de este algoritmo efectivamente disminuye el tiempo de ejecución de los clasificadores, principalmente debido al ahorro de operaciones de punto flotante repetidas y al ahorro de accesos a memoria (los parámetros de Naïve Bayes, almacenados en una tabla tridimensional, resultan, según el análisis con profiler, costosos de acceder). Sin embargo, no es inmediato hacer un análisis de si realmente disminuye la complejidad teórica del algoritmo o no.

El tiempo de ejecución sería aún menor si no fuera necesario calcular el vector de características de nuevo, es decir, si se pudieran actualizar eficientemente las características (imagen binaria y códigos de dirección) al eliminar un cuadro de la secuencia, sin recorrer

toda la secuencia de nuevo. Sin embargo, según el profiler, el cálculo de características ocupa un tiempo despreciable en comparación con otras funciones, lo cual hace no prioritaria su optimización.

4.4. Detalles de implementación

El sistema está implementado en lenguaje C++ en un ambiente de desarrollo Microsoft Visual Studio. Se usó la librería de visión computacional OpenCV[2] para varias tareas relacionadas con la manipulación de imágenes, incluyendo la adquisición de éstas desde la cámara, la conversión y procesamiento de imágenes intermedias y el despliegue de los resultados en la pantalla. La figura 4.16 muestra la interfaz gráfica, la cual despliega el estado actual del tracking de cara y mano, así como el estado actual de la detección de gestos.

El sistema corre a 5-7 fps en el robot Bender [1], cuya central de procesamiento es un tablet PC con sistema operativo Windows XP, 2GB de RAM y procesador Intel Core 2 Duo U7600 de 1.20 GHz. Las imágenes usadas son de 320×240 pixeles y se obtienen de una webcam común (Philips SPC900NC).

[2]http://opencv.willowgarage.com/wiki/

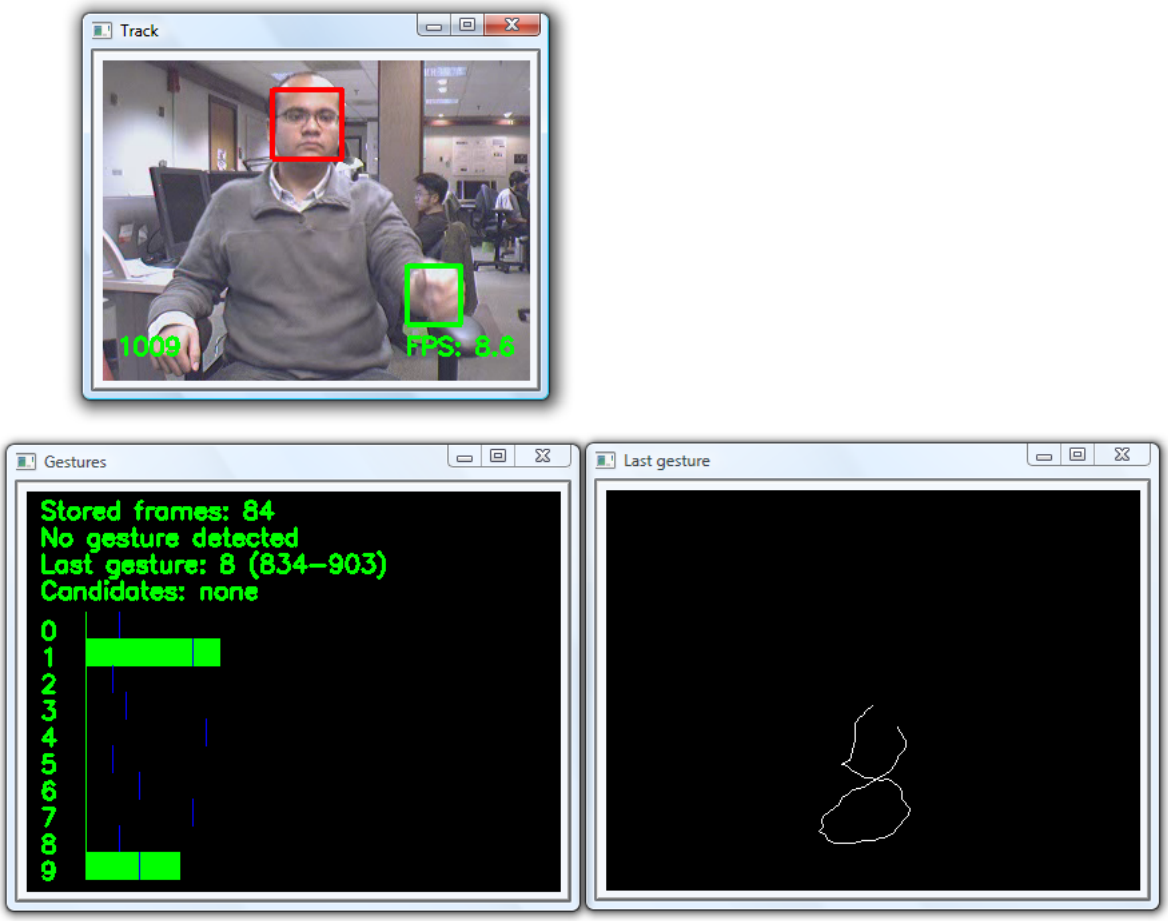


Figura 4.16: Interfaz gráfica. La ventana “Track” muestra la cara y la mano detectadas. La ventana “Gestures” muestra información del estado de la detección más la tabla de gestos con los umbrales. La ventana “Last gesture” muestra la forma del último gesto detectado.

Capítulo 5

Resultados y discusión

5.1. Descripción de los experimentos

El sistema desarrollado ha sido puesto a prueba en dos bases de datos de gestos dinámicos. La primera base de datos es ALON-EASY, descrita en la sección 4.2.1, que contiene 30 videos de 10 personas distintas dibujando los dígitos Graffiti en el aire. Las personas llevan mangas cortas, el fondo es estático la mayor parte del tiempo y la mano es retirada después de cada gesto.

La segunda base de datos es GESTOS-BENDER, que contiene comandos diseñados específicamente para el robot. Esta base de datos consta de 15 videos de 5 personas distintas dibujando los gestos: derecha (RIGHT), izquierda (LEFT), arriba (UP), abajo (DOWN), vuelta anti-horaria (CCW), vuelta horaria (CW), saludo (WAVE), visto bueno (CHECK) y cruz (CROSS) (véase la figura 5.1).

Las condiciones en GESTOS-BENDER son similares a las de ALON-EASY: el fondo es estático, los usuarios usan mangas cortas y retiran la mano tras cada gesto. Esta base de datos fue grabada en una situación de interacción realista con el robot: los usuarios estaban situados frente a Bender, a la vista de la cámara que éste tiene instalada en la cabeza, y podían ver la imagen captada por la cámara en el tablet PC que tiene Bender en el torso. Los participantes de la grabación recibieron mínimas instrucciones sobre la velocidad y tamaño necesario para permitir el reconocimiento correcto de los gestos, y no hubo un ensayo previo antes de grabar los videos, de los cuales se preservó, en su gran mayoría, el primer intento. La figura 5.2 muestra imágenes de GESTOS-BENDER.

A continuación, se presentan los resultados del reconocimiento de gestos para las base de datos ALON-EASY y GESTOS-BENDER. Para cada base de datos se ofrece, en primer lugar, un resumen de los resultados del tracking. Luego, se presentan los resultados del reconocimiento de gestos y una discusión de éstos. El presente capítulo termina con una breve discusión integrada de los resultados.

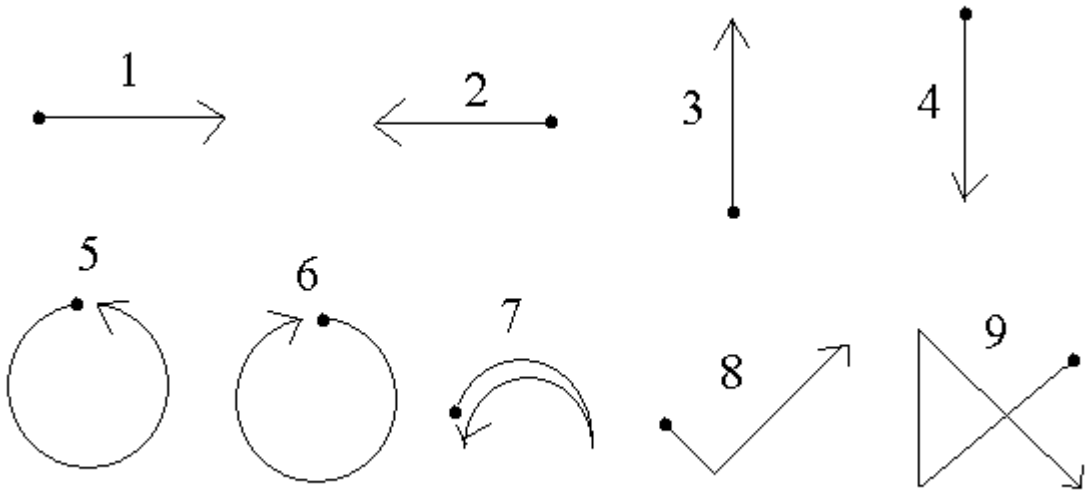


Figura 5.1: Gestos diseñados para controlar al robot Bender: 1. derecha (RIGHT), 2. izquierda (LEFT), 3. arriba (UP), 4. abajo (DOWN), 5. vuelta anti-horaria (CCW), 6. vuelta horaria (CW), 7. saludo (WAVE), 8. visto bueno (CHECK), 9. cruz (CROSS).

Figura 5.2: Muestras de la base de datos GESTOS-BENDER

5.2. Resultados para la base de datos ALON-EASY

5.2.1. Resultados del tracking

Puesto que en ALON-EASY no se sigue la convención de inicializar el tracking con el gesto "puño", este paso debe ser hecho manualmente solo para efectos de la evaluación. La decisión de si el tracking es correcto o no queda a criterio de un evaluador humano.

El cuadro 5.1 entrega el detalle de los resultados del tracking para los 300 gestos de ALON-EASY. El nombre de cada video remite al usuario que participa en él. Los videos de un mismo usuario son grabados en condiciones similares de iluminación, con el usuario en la misma posición y a la misma distancia a la cámara. El cuadro 5.2 entrega un resumen del rendimiento por gesto, mientras que el cuadro 5.3 es un resumen global.

Los resultados entregados corresponden a los mejores obtenidos de entre las pruebas hechas. Se usa mean-shift con un histograma RGB cuantizado a 4 bits y una característica de pixel de borde, aplicado sobre la máscara de piel y movimiento y con actualización del histograma modelo cada 10 cuadros. El uso de características de textura LBP fue considerado, pero descartado por tener un impacto considerable en la eficiencia sin aportar mejoras proporcionales en la calidad del tracking. También se consideró el uso de múltiples puntos de partida (sección 3.1.4), pero este enfoque disminuyó la estabilidad del movimiento trackeado y en general empeoró el rendimiento.

5.2.2. Resultados de reconocimiento de gestos

De acuerdo con Lee y Kim [39], hay tres tipos de errores en los que puede incurrir un sistema de reconocimiento continuo de gestos:

1. de inserción: cuando el sistema declara un gesto pero ningún gesto real coincide temporalmente con él.
2. de eliminación: cuando el usuario realiza un gesto y el sistema no realiza ningún reconocimiento.
3. de sustitución: cuando el sistema clasifica erróneamente un gesto.

La tasa de detección se define como la razón entre los gestos correctamente reconocidos y el total de gestos realizados:

$$\text{Tasa de detección} = \frac{\text{gestos correctamente reconocidos}}{\text{total de gestos realizados}}$$

Esta cantidad toma en cuenta los gestos sustituidos y eliminados, pero no aquellos insertados. Puesto que la inserción de un gesto hace perder credibilidad al reconocimiento, se define un índice que dé cuenta de este efecto:

Video	Gestos									
	0	1	2	3	4	5	6	7	8	9
angshuman1	o	o	o	o	o	o	o	o	o	o
angshuman2	o	o	o	o	o	o	o	o	o	o
angshuman3	o	o	o	o	o	o	o	o	o	o
ashwin1	o	o	o	o	o	o	o	o	o	o
ashwin2	o	o	o	o	o	o	o	o	o	o
ashwin3	o	o	o	o	o	o	o	o	o	o
ben1	x	o	x	o	o	o	o	o	o	o
ben2	x	o	o	o	o	o	x	o	o	o
ben3	x	x	o	o	o	o	o	o	o	o
jared1	x	o	o	o	o	o	o	o	o	o
jared2	x	o	o	o	o	o	o	o	o	o
jared3	x	o	o	o	o	o	o	o	o	o
jingbin1	o	o	x	o	o	o	o	o	o	o
jingbin2	o	o	o	o	o	o	o	o	o	o
jingbin3	o	o	o	o	o	o	o	o	x	o
john1	x	o	o	o	o	o	x	o	o	o
john2	o	x	o	o	o	o	o	o	o	o
john3	o	o	o	o	o	o	x	o	o	o
joni1	o	o	o	o	o	o	o	o	o	o
joni2	o	o	o	o	o	o	o	o	o	o
joni3	o	o	o	o	o	o	o	o	o	o
murat1	o	o	o	o	o	o	o	o	o	o
murat2	o	o	o	o	o	o	o	o	o	o
murat3	o	o	o	o	o	o	o	o	o	o
quan1	x	o	o	x	x	o	x	o	o	o
quan2	o	o	o	o	o	o	x	o	o	o
quan3	x	o	o	o	o	o	o	o	o	o
tian1	x	o	x	o	o	o	o	o	o	o
tian2	o	o	o	o	o	o	o	o	o	x
tian3	x	o	x	o	o	o	o	o	o	o

Cuadro 5.1: Resultados detallados del tracking en ALON-EASY. Cada fila representa un video y cada columna representa un gesto. Con "o" se marca el tracking exitoso, con "x" el tracking fallido.

Resultado	Gestos									
	0	1	2	3	4	5	6	7	8	9
correctos (o)	19	28	26	29	29	30	25	30	29	29
incorrectos (x)	11	2	4	1	1	0	5	0	1	1
% correctos	63.3	93.3	86.7	96.7	96.7	100	83.3	100	96.7	96.7

Cuadro 5.2: Resumen de resultados del tracking por gesto, base de datos ALON-EASY.

Resultado	Cantidad	Porcentaje
correctos (o)	274	91.3 %
incorrectos (x)	26	8.7 %

Cuadro 5.3: Resumen global para ALON-EASY. Hay un 91.3 % de instancias correctas de tracking.

Gesto	Total	Correcto	Insertado	Eliminado	Sustituido	% detección	% confiabilidad
0	30	13	0	2	15	43.3	43.3
1	30	29	2	1	0	96.7	90.6
2	30	19	0	5	6	63.3	63.3
3	30	29	0	1	0	96.7	96.7
4	30	30	0	0	0	100	100
5	30	25	0	1	4	83.3	83.3
6	30	26	0	0	4	86.7	86.7
7	30	24	0	0	6	80	80
8	30	27	0	2	1	90	90
9	30	23	0	0	7	76.7	76.7
Total	300	245	2	12	43	81.7	81.7

Cuadro 5.4: Resultados del reconocimiento de gestos en ALON-EASY.

$$\text{Confiabilidad} = \frac{\text{gestos correctamente reconocidos}}{\text{total de gestos realizados} + \text{número de gestos insertados}}$$

El cuadro 5.4 da un resumen de los errores del sistema y las tasas de detección y confiabilidad.

Este cuadro se puede derivar de la matriz de confusión (cuadro 5.5), siempre y cuando no exista un caso de múltiples detecciones erróneas durante un gesto (por ejemplo, durante la ejecución de un 6, detectar un 1 y después un 0). Esta representación de los resultados es usada también en la obra de Alon et al. [3]. Para construir esta matriz, el sistema asocia cada gesto detectado con el gesto real más cercano. Luego, se aumenta en 1 la entrada correspondiente en la matriz. Usando el mismo criterio de Alon, una detección es correcta cuando los cuadros de la detección cubren al menos el 50 % del gesto real y viceversa.

5.3. Discusión para la base de datos ALON-EASY

Al comparar los resultados obtenidos en ALON-EASY con los de Alon et al., el resultado es mejor que la tasa de detección del 75.6 % obtenido en condiciones comparables (un solo candidato a mano), pero no supera el mejor resultado de dicho trabajo (94.6 % al usar 4, 5 ó 6 candidatos a mano).

El examen de la matriz de confusión entrega varias ideas de los tipos de errores en que incurre el reconocimiento de gestos. Hay problemas de confusión geométrica (0 reconocido

	0	1	2	3	4	5	6	7	8	9	insertados
0	13	0	0	0	0	0	1	0	0	0	0
1	5	29	0	0	0	0	3	6	0	2	2
2	0	0	19	0	0	0	0	0	0	0	0
3	0	0	2	29	0	2	0	0	0	1	0
4	2	0	0	0	30	0	0	0	0	4	0
5	0	0	0	0	0	25	0	0	1	0	0
6	8	0	0	0	0	0	26	0	0	0	0
7	0	0	3	0	0	0	0	24	0	0	0
8	0	0	1	0	0	2	0	0	27	0	0
9	0	0	0	0	0	0	0	0	0	23	0
eliminados	2	1	5	1	0	1	0	0	2	0	

Cuadro 5.5: Matriz de confusión para ALON-EASY. Las columnas corresponden a las etiquetas reales de los gestos de prueba, mientras que las filas indican el resultado del reconocimiento, si lo hubo. Los gestos que no son detectados por el sistema se marcan en la fila "eliminados", mientras que la columna "insertados" da cuenta de reconocimientos del sistema que no corresponden a ningún gesto real.

como 6), problemas de subgestos reconocidos en vez del supergesto (1 reconocido en vez del 0, 6, 7 ó 9 que lo contenía; 4 en vez de 0 ó 9; 5 en vez de 8; 7 en vez de 2), y problemas de supergestos reconocidos en vez de subgestos, cuando el usuario, al retirar la mano, termina describiendo una figura que se parece a un gesto existente (un 2 que se convierte en 3, un 5 que se convierte en 8). Finalmente, hay número de gestos que no son reconocidos del todo (sufren del error de eliminación).

Esta sección ofrece un examen de los factores que explican el rendimiento del sistema.

5.3.1. Errores en el tracking

Un factor importante en la tasa de detección del sistema es el tracking correcto. En efecto, uno de los problemas que encuentran Alon et al. en otros sistemas de reconocimiento de gestos es su dependencia en un tracking confiable de la mano. En su opinión, esta meta es difícil de alcanzar, por lo que proponen sustituir este requerimiento por otro más simple de alcanzar, cual es un módulo que entrega zonas de la imagen candidatas a contener la mano. El algoritmo propuesto por ellos ha sido diseñado expresamente para aprovechar esta información.

A pesar de esto, en esta memoria se supone un tracking confiable. El algoritmo mean-shift basado en características de color y borde da resultados positivos en alrededor del 91 % de los casos. La mayoría de los fallos se debe a confusiones con el fondo o con el propio brazo del usuario, cuando este usa mangas cortas, como en la figura 5.3. Se trata de tareas de tracking que un humano podría realizar perfectamente. El uso de color y borde es insuficiente, y es probable que se puedan evitar errores considerando más información, ya sean características especializadas o información contextual (por ejemplo, razonamiento sobre el cuerpo humano).

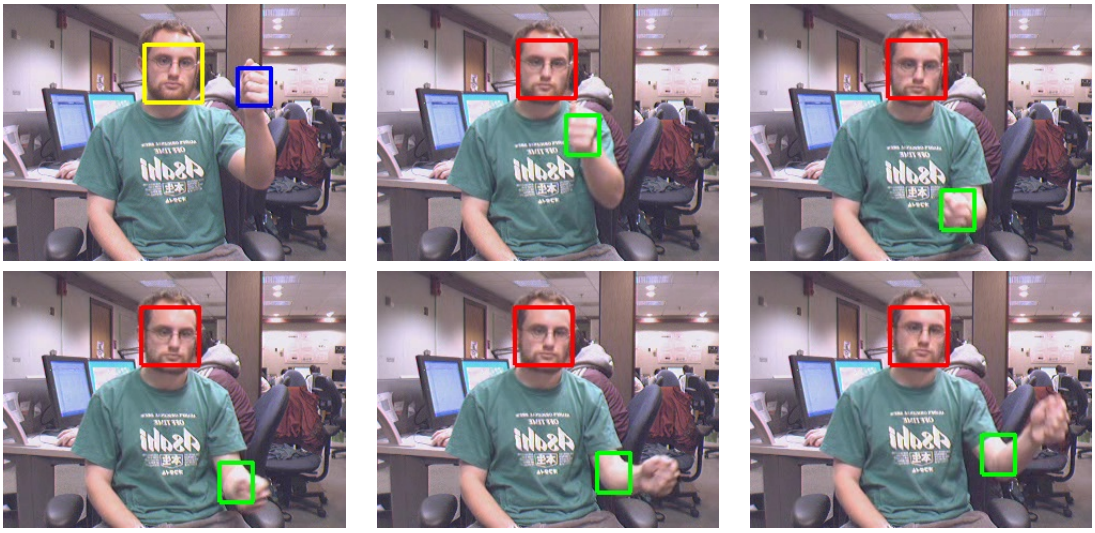


Figura 5.3: Ejemplo de fallo del tracking. El usuario describe un cero, pero el tracking se distrae con el brazo.

La mejora con respecto al tracking antes de abordar la memoria es marginal, considerando la gran cantidad de cambios hechos.

Considerando que el reconocimiento de gestos es exitoso en 245 casos y que el tracking es exitoso en 274 casos, la tasa de reconocimiento sin considerar fallos por tracking es $\frac{245}{274} = 89.4\,\%$.

5.3.2. Errores en el reconocimiento de gestos

El uso de plantillas para verificar la plausibilidad de un gesto es necesario puesto que la etapa de generación de candidatos entrega una gran cantidad de falsos positivos. Sin embargo, es sabido que el uso de plantillas, si bien tiene la ventaja de necesitar un solo ejemplar de entrenamiento, tiene la desventaja de no tomar en cuenta la variabilidad entre distintas ejecuciones de un gesto. El sistema desaprovecha la información de variabilidad implícita en el conjunto de entrenamiento al condensar todos los datos en plantillas, y las medidas de distancia usadas (distancia de edición y Hausdorff) no son lo suficientemente flexibles para detectar gestos que presentan variaciones importantes pero que aún son instancias válidas del gesto. Véanse, por ejemplo, las figuras 5.4 (un falso negativo) y 5.5 (un falso positivo).

Algunas propuestas para atacar este problema son: (a) la mejora del algoritmo de normalización, usando transformaciones proyectivas como en [45] (b) el uso de razonamiento topológico (es decir, invariante bajo deformaciones) aplicado a imágenes binarias [30, 34] y (c) el uso de algoritmos de recuperación de información que permitan obtener de los datos de entrenamiento aquellos gestos con el mayor parecido (por ejemplo, la recuperación de los vecinos más cercanos en el conjunto de entrenamiento más un esquema de suma de votos para decidir cuál gesto es el más parecido).

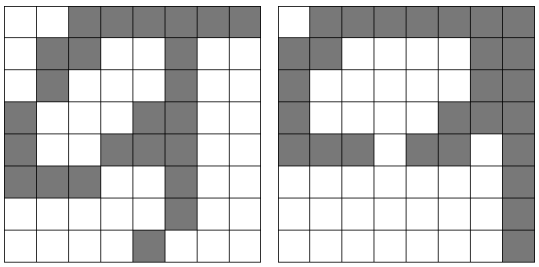


Figura 5.4: Comparación de un gesto real del usuario (izquierda) con la plantilla oficial. El usuario no cierra con exactitud el 9, dejando una línea horizontal que se asoma. El gesto pasa la prueba de calce de direcciones, pero no pasa el calce de imágenes binarias. Sin embargo, el gesto es reconocible como un 9 y debería ser aceptado como tal por el sistema.

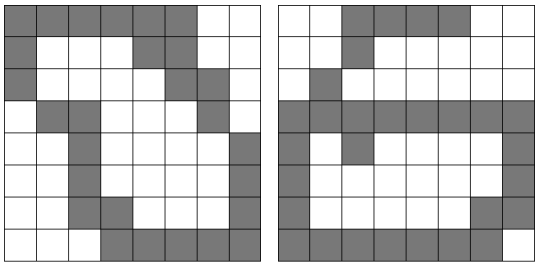


Figura 5.5: El usuario ejecuta un cero, pero por imprecisiones del tracking se obtiene la figura de la izquierda, la cual, comparada con la plantilla del 6 (derecha) mediante la distancia de Hausdorff, resulta estar dentro del umbral de aceptación pese a que no tiene las mismas propiedades topológicas.

5.3.3. Criterio de término de gesto

Una de las principales dificultades con que se encuentra el algoritmo propuesto es la de decidir cuándo dar por terminado el reconocimiento, es decir, decidir el momento en que se escogerá un candidato para declararlo como el definitivo.

En Alon et al. [3], se elige un gesto g como el oficialmente reconocido si todas las detecciones parciales en curso han comenzado después del cuadro final de g. Es decir, se han descartado todos los candidatos que coinciden temporalmente con g (y g mismo no ha sido descartado). Con "detecciones parciales" se denomina a los calces que el sistema ha detectado hasta el momento entre la secuencia de manos y los prefijos de modelos gestuales existentes. Este criterio tiene sentido en el contexto de dicho trabajo, puesto que los gestos están modelados como secuencias de estados y el algoritmo mantiene continuamente un calce entre las posiciones detectadas de la mano y los prefijos de cada gesto.

En cambio, este mismo criterio no tiene sentido en el método propuesto en esta memoria, donde no hay una manera de averiguar si una secuencia dada es prefijo de un gesto. Por esto, se sigue dependiendo, como en otros trabajos, de variables de bajo nivel como la velocidad de la mano para realizar el reconocimiento. Hay que aclarar que la segmentación en sí no depende de la velocidad (como en otros trabajos), pero sí se necesita examinar dicha variable para determinar, mediante reglas adicionales, cuál es la verdadera intención del usuario (continuar haciendo un gesto o terminarlo). Desafortunadamente, la velocidad de la mano resulta un indicador poco confiable, pues es susceptible a variaciones casuales del movimiento de la mano.

Esto explica algunos errores en que se declara un subgesto en vez de un supergesto: una vez que aparece el subgesto como candidato, y antes de completar el supergesto, el sistema detecta que la mano ha aumentado o disminuido considerablemente su velocidad y adelanta el cuadro de declaración, pero en muchos casos esta variación es casual, y no puede ser modelada únicamente con un par de umbrales (para velocidades bajas y altas). Del mismo modo, el usuario podría no variar la velocidad de su mano una vez terminado el gesto, por lo que el proceso de reconocimiento supone que existe la intención de completar un supergesto, por lo que queda en espera, y el resultado es el reconocimiento de un falso supergesto (como al reconocer un 8 en vez de un 5, debido a un trazo diagonal adicional cuya verdadera intención era poner la mano en posición para dibujar un nuevo gesto).

Una estrategia para mejorar el razonamiento con subgestos consiste en extraer las relaciones de subgestos e inferir reglas de término para cada candidato posible. La observación clave es que hay gestos que no son subgestos de ningún otro gesto, como el 8. Por lo tanto, si aparece el 8 como candidato, no hace falta esperar que el usuario complete ningún gesto. En cambio, el 5, que es subgesto del 8, estaría asociado a la siguiente regla: si aparece el candidato 5, y tras el cuadro final del 5 la mano no sube, se da por terminado el reconocimiento, declarando al 5 como gesto reconocido. Esta regla es válida puesto que el 8 es el único supergesto del 5.

La principal objeción contra este método es que, en el sistema desarrollado, la extracción de relaciones de subgestos debe ser hecha manualmente, por las razones ya explicadas: no hay una manera obvia de trabajar con subtrayectorias de los gestos. Por esta misma razón,

Video	Gestos								
	RIGHT	LEFT	UP	DOWN	CCW	CW	WAVE	CHECK	CROSS
isao1	o	o	o	o	o	o	o	o	o
isao2	o	o	o	o	o	o	o	o	x
isao3	o	o	o	o	x	o	x	o	o
jongbor1	o	o	o	o	o	o	o	o	o
jongbor2	o	o	o	o	o	o	o	o	o
jongbor3	o	o	o	o	o	o	o	o	o
mauricio1	o	o	o	o	o	o	o	o	o
mauricio2	o	o	o	o	o	o	o	o	o
mauricio3	o	o	o	o	o	o	o	o	o
paul1	o	o	o	o	o	o	x	o	o
paul2	o	o	o	o	o	o	o	o	o
paul3	o	o	o	o	o	o	o	o	o
rodrigo1	o	o	o	o	o	o	o	o	o
rodrigo2	o	o	o	o	o	o	o	o	o
rodrigo3	o	o	o	o	o	o	o	o	o

Cuadro 5.6: Resultados detallados del tracking en GESTOS-BENDER. Cada fila representa un video y cada columna representa un gesto. Con "o" se marca el tracking exitoso, con "x" el tracking fallido.

la extracción de las reglas también sería manual.

5.4. Resultados para la base de datos GESTOS-BENDER

Los resultados para GESTOS-BENDER se presentan en el mismo formato que los de ALON-EASY.

Los parámetros y ajustes usados en esta prueba son los mismos que los usados con ALON-EASY, es decir, para el tracking, se usa el algoritmo mean-shift con un histograma RGB cuantizado a 4 bits y una característica de pixel de borde, aplicado sobre la máscara de piel y movimiento y con actualización del histograma modelo cada 10 cuadros.

5.4.1. Resultados del tracking

El cuadro 5.6 entrega el detalle de los resultados del tracking para los 150 gestos de GESTOS-BENDER. El nombre de cada video remite al usuario que participa en él. Los videos de un mismo usuario son grabados en condiciones similares de iluminación, con el usuario en la misma posición y a la misma distancia a la cámara. El cuadro 5.7 entrega un resumen del rendimiento por gesto, mientras que el cuadro 5.8 es un resumen global.

Resultado	Gestos								
	RIGHT	LEFT	UP	DOWN	CCW	CW	WAVE	CHECK	CROSS
correctos (o)	15	15	15	15	14	15	13	15	14
incorrectos (x)	0	0	0	0	1	0	2	0	1
% correctos	100	100	100	100	93.3	100	86.7	100	93.3

Cuadro 5.7: Resumen de resultados del tracking por gesto en GESTOS-BENDER.

Resultado	Cantidad	Porcentaje
correctos (o)	131	97 %
incorrectos (x)	4	3 %

Cuadro 5.8: Resumen global para GESTOS-BENDER. Hay un 97 % de instancias correctas de tracking.

5.4.2. Resultados del reconocimiento de gestos

El cuadro 5.9 da un resumen de los errores del sistema y las tasas de detección y confiabilidad, mientras que el cuadro 5.10 muestra la matriz de confusión. Se remite al lector a la sección 5.2.2 para explicaciones sobre estos cuadros.

5.5. Discusión para la base de datos GESTOS-BENDER

El tracking resulta muy efectivo en GESTOS-BENDER (una tasa del 97 %), lo cual encuentra explicación en que el fondo en estos videos es estático y uniforme y los usuarios están cerca de la cámara y mantienen el puño frontal en la medida de lo posible.

Un factor que dificulta el reconocimiento en GESTOS-BENDER es la sencillez de los gestos (especialmente los gestos "izquierda", "derecha", "arriba" y "abajo"), que frecuentemente aparecen en el movimiento de la mano entre gestos —dando lugar a gestos insertados— o co-

Gesto	Total	Correcto	Insertado	Eliminado	Sustituido	% detección	% confiabilidad
RIGHT	15	14	0	1	0	93.3	93.3
LEFT	15	15	0	0	0	100	100
UP	15	15	1	0	0	100	93.8
DOWN	15	12	2	3	0	80	70.6
CCW	15	12	0	1	2	80	80
CW	15	11	0	2	2	73.3	73.3
WAVE	15	8	0	6	1	53.3	53.3
CHECK	15	14	0	0	1	93.3	93.3
CROSS	15	5	0	0	10	33.3	33.3
Total	135	106	3	13	16	78.5	76.8

Cuadro 5.9: Resultados del reconocimiento de gestos en GESTOS-BENDER (véase una explicación en la sección 5.2.2).

	RIGHT	LEFT	UP	DOWN	CCW	CW	WAVE	CHECK	CROSS	ins.
RIGHT	14	0	0	0	0	0	0	0	0	0
LEFT	0	15	0	0	0	0	0	0	0	0
UP	0	0	15	0	0	0	0	0	10	1
DOWN	0	0	0	12	2	2	1	1	0	2
CCW	0	0	0	0	12	0	0	0	0	0
CW	0	0	0	0	0	11	0	0	0	0
WAVE	0	0	0	0	0	0	8	0	0	0
CHECK	0	0	0	0	0	0	0	14	0	0
CROSS	0	0	0	0	0	0	0	0	5	0
elim.	1	0	0	3	1	2	6	0	0	

Cuadro 5.10: Matriz de confusión para GESTOS-BENDER, análogo al cuadro 5.5.

mo subgestos. El problema de subgestos está cubierto, pero el problema de inserción requiere una regla adicional, cual es eliminar aquellos candidatos trazados a alta velocidad (según un umbral encontrado experimentalmente), puesto que las inserciones generalmente se deben a movimientos que hace el usuario para posicionar o retirar la mano, y estos movimientos suelen ser mucho más rápidos que el movimiento durante los gestos. Tomando estas medidas, se consigue un mínimo de detecciones insertadas (3 inserciones de los gestos "arriba" y "abajo", comparados con 21 inserciones que se obtenían antes de agregar esta regla).

Los gestos con peor tasa de reconocimiento son "saludo" (WAVE), que sufre varias eliminaciones, y "cruz" (CROSS), que no suele aparecer como candidato, y en cambio se reconoce su subgesto "arriba" (UP).

Tal como en ALON-EASY, los errores en GESTOS-BENDER pueden agruparse en tres categorías:

- Errores en el tracking: que en este caso no son muchos.
- Errores en el reconocimiento de gestos: lo cual incluye errores en la segmentación misma del gesto, gestos que no son suficientemente grandes, que son demasiado rápidos o que ni siquiera caben en la ventana de N cuadros donde se realiza la generación de candidatos (este suele ser el caso del gesto "saludo", que tiende a ser largo).
- Errores en el criterio de término de gesto: debidos sobre todo a bajas de la velocidad de la mano que el sistema interpreta como señales de fin de gesto.

La tasa de detección considerando solo los ejemplares con tracking exitoso es de $\frac{106}{131} = 80.9\,\%$. Este porcentaje es menor que el $89.4\,\%$ conseguido en ALON-EASY. Esta diferencia se explica probablemente en que los usuarios en GESTOS-BENDER realizan los gestos por primera vez, sin ser expertos en el sistema y sin ensayo previo. Esto necesariamente lleva a gestos que no son hechos con la velocidad ideal ni con una forma óptima para el reconocimiento (sin dejar de ser gestos reconocibles para un observador humano).

5.6. Discusión integrada de los resultados

Los experimentos sobre la base de datos ALON-EASY revelan las principales deficiencias del sistema:

- Un tracking que presenta fallos al distraerse con objetos de color similar o simplemente al divergir.
- Errores de calce geométrico que ocasionan tanto falsos positivos como falsos negativos.
- Un criterio de término basado puramente en la velocidad de la mano, que resulta susceptible a variaciones casuales del movimiento de ésta.

Estas causas de error se encuentran descritas en la discusión de los resultados de ALON-EASY (sección 5.3).

La base de datos GESTOS-BENDER tiene algunas diferencias fundamentales con ALON-EASY. En GESTOS-BENDER, el tracking resulta mucho más efectivo. Los gestos, geométricamente, son menos complejos, pero los usuarios de GESTOS-BENDER no los han practicado tanto como los de ALON-EASY, y la variabilidad en la velocidad de ejecución resulta mucho mayor, por lo que un aspecto como el tamaño de la ventana de búsqueda, que en ALON-EASY no resultaba importante, en GESTOS-BENDER pasa a serlo. Los errores debidos que el gesto no cabe en la ventana de búsqueda se consideran subsanables ya que, durante el uso real, dichos gestos resultarán eliminados, y el usuario podrá intentar de nuevo dibujar el gesto a una velocidad algo mayor.

Aparte de este aspecto, en GESTOS-BENDER, tal como en ALON-EASY, se encuentran errores por tracking (aunque mucho menos), errores de calce geométrico y errores por mala elección del momento de término (especialmente debidos a bajas velocidades de la mano durante la ejecución del gesto).

Las pruebas con GESTOS-BENDER revelan además que los gestos sencillos suponen una dificultad adicional en cuanto es fácil que sufran del error de inserción, por lo que se toman medidas adicionales que descartan las trayectorias hechas con velocidades muy altas.

Capítulo 6

Conclusiones

En esta memoria se llevó a cabo el desarrollo de un método novedoso en el ámbito del reconocimiento de gestos dinámicos.

El método propuesto intenta prescindir de hacer correspondencias temporales explícitas entre secuencias, y en cambio hace calces geométricos entre plantillas obtenidas del conjunto de entrenamiento (como la imagen binaria y los códigos de dirección) y las secuencias entrantes.

Este enfoque ofrece posibilidades interesantes de trabajar en la geometría de los gestos, pero al costo de hacer más difícil el análisis de la información que podría llamarse "secuencial" o "parcial"; concretamente, el sistema no ofrece un mecanismo para averiguar si el usuario lleva ejecutado el prefijo de un gesto. Esto tiene una repercusión al momento de analizar las intenciones del usuario, y por lo tanto introduce defectos en el reconocimiento.

El análisis geométrico continuo es implementado mediante la aplicación de clasificadores Naïve Bayes. Existen trabajos que incorporan la variable temporal explícitamente a Naïve Bayes [5]; en cambio, en este trabajo, se usan clasificadores Naïve Bayes tradicionales.

El sistema analiza la secuencia de posiciones de la mano y es capaz de segmentar exitosamente los gestos mediante un análisis como el de la generación de candidatos, basado en la búsqueda de máximos locales de la probabilidad de un gesto y la mantención de promedios móviles a lo largo de varios cuadros. Considerando los defectos de Naïve Bayes (el hacer una suposición fuerte de independencia y el no estar calibrado) y la sencillez de los cálculos usados, resulta interesante que la segmentación sea correcta.

6.1. Cumplimiento de objetivos

En relación a los objetivos, puede decirse que se ha cumplido la meta de desarrollar un sistema de reconocimiento de gestos dinámicos. El sistema ha sido puesto a prueba en bases de datos de gestos y también en uso real. Este último uso resulta algo más problemático, pues se necesita un período de aprendizaje de las convenciones y limitaciones del sistema.

Así y todo, se estima que su rendimiento es suficiente para su uso real.

También se cumplió con el objetivo de estudiar el sistema de visión computacional y organizar su código. Sin embargo, no pudo cumplirse con cabalidad con el objetivo de aumentar la efectividad del tracking, pues, pese a la gran cantidad de ideas propuestas, la mejora fue mínima. En cuanto a optimizar los algoritmos de visión (detección de caras y puños), debe mencionarse que el código de estos algoritmos es el producto de varios trabajos pasados y no está en el alcance inmediato de esta memoria entenderlo e intervenirlo.

6.2. Limitaciones

El sistema exige la inicialización del tracking de la mano usando el puño frontal y sin rotar. Esto puede dificultar la ejecución de gestos que tienen su punto inicial alejado de la cara del usuario, donde es difícil no rotar el puño. El movimiento de la mano no necesita ser exageradamente lento, pero sí hay que cuidar de no hacer movimientos bruscos durante el gesto que hagan fallar el tracking. El uso de mangas cortas también pone dificultades al tracking, que en algunos casos empieza a seguir el brazo.

Otra limitación es la dependencia en variables de bajo nivel para dar por terminado el proceso de reconocimiento. Si bien la segmentación en sí no necesita que la mano haga pausas al principio y al final del gesto, todavía es necesario hacer alguna variación significativa en el movimiento de la mano (aumentar su velocidad, disminuirla o bien retirar la mano del campo visual de la cámara) para dar por terminado el proceso de reconocimiento.

El sistema reconoce solamente gestos caracterizados por trayectorias espaciales de la mano. En el contexto de interacción humano-robot, interesa a largo plazo tener gestos con otras partes del cuerpo (especialmente la cabeza, los brazos, los pies y los hombros) que involucren tanto pose como movimiento.

6.3. Trabajo futuro

Queda como tema de investigación el cómo hacer un tracking robusto de la mano. Las ideas propuestas en la sección 3.1, como la evaluación de nuevas características, la actualización del modelo, la inicialización en múltiples puntos y la predicción del movimiento, no son en ningún caso ideas nuevas, pero probablemente necesitan algo más de dedicación que la que pudo entregarse en el desarrollo de este trabajo. También valdría la pena investigar las técnicas de tracking más recientes, basadas en aprendizaje en línea de la apariencia del objeto seguido, como [55]. Queda abierta la pregunta de si es posible desarrollar un tracking de alto rendimiento y funcionamiento en tiempo real, o si el enfoque de Alon et al. de considerar múltiples candidatos es más realista.

En lo que concierne al reconocimiento de gestos en sí, para mejorar las tasas de detección, el trabajo futuro debería centrarse en dos aspectos: mayor robustez ante la variabilidad de los gestos, y mejor análisis de intencionalidad del usuario.

La mayor robustez ante la variabilidad podría lograrse usando herramientas geométricas más sofisticadas como las explicadas en la sección 5.3.2. Yendo más lejos, convendría investigar cuál es el verdadero aporte del uso de una imagen binaria al reconocimiento de gestos, ya que los códigos de dirección parecen contener toda la información necesaria para identificar un gesto, si se toman suficientes muestras. Es posible que la imagen binaria sea útil al añadir mayor robustez, al permitir la extracción de menos códigos de dirección, o que en realidad sea prescindible.

Por otro lado, el análisis de intencionalidad debería ser más robusto que el mero examen de la velocidad de la mano. El cuadro de declaración (descrito en la sección 4.3.4) tiene la propiedad de ser adaptivo a la velocidad de la mano, pero aún hay muchas variables que deben ser fijadas manualmente, como el valor de espera inicial y los umbrales de la velocidad. El razonamiento sobre gestos incompletos como en Alon et al. parece ineludible; sin embargo, no hay una manera obvia de realizar este análisis en el sistema desarrollado.

En el contexto más amplio de interacción humano-computador, interesa que esta interacción sea lo más natural y cómoda posible para el usuario. Hay una gran cantidad de trabajos que han analizado el movimiento de cuerpo completo, combinando diversos gestos dinámicos en una única arquitectura. La literatura al respecto es abundante. Solo como ejemplo, véanse [11, 28].

No debe perderse de vista que, a diferencia de formas de interacción humano-computador más tradicionales como el teclado y el mouse, las formas de comunicación como los gestos y el habla están sujetos a muchos más errores, por lo que necesariamente tendrán que estar insertos en un marco de trabajo que considere la interacción en su totalidad, integrando distintas formas de comunicación (el habla, los gestos, etc.) y añadiendo mecanismos de manejo de la ambigüedad, la incerteza y los errores.

Referencias

[1] Sitio oficial de Bender, http://bender.li2.uchile.cl/.

[2] E. L. Allwein, R. E. Schapire, and Y. Singer. Reducing multiclass to binary: a unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1:113–141, 2001.

[3] J. Alon, V. Athitsos, Q. Yuan, and S. Sclaroff. A unified framework for gesture recognition and spatiotemporal gesture segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(9):1685–1699, 2008.

[4] E. Alpaydin. *Introduction to Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, October 2004.

[5] H. H. Avilés-Arriaga and L. E. Sucar. Dynamic Bayesian networks for visual recognition of dynamic gestures. *Journal of Intelligent and Fuzzy Systems*, 12(3-4):243–250, 2002.

[6] A. Babaeian, S. Rastegar, M. Bandarabadi, and M. Rezaei. Mean shift-based object tracking with multiple features. *Southeastern Symposium on System Theory*, 0:68–72, 2009.

[7] E. Baudrier, G. Millon, F. Nicolier, and S. Ruan. A fast binary-image comparison method with local-dissimilarity quantification. In *ICPR '06: Proceedings of the 18th International Conference on Pattern Recognition*, pages 216–219, Washington, DC, USA, 2006. IEEE Computer Society.

[8] P. N. Bennett. Assessing the calibration of Naive Bayes' posterior estimates. Technical report, School of Computer Science, Carnegie Mellon University, 2000.

[9] M. Bhuyan, D. Ghosh, and P. Bora. Continuous hand gesture segmentation and co-articulation detection. In *ICCVGIP06*, pages 564–575, 2006.

[10] N. D. Binh, E. Shuichi, and T. Ejima. Real-time hand tracking and gesture recognition system. *ICGST International Journal on Graphics, Vision and Image Processing, Special Issue on Biometrics*, 06:31–39, 2006.

[11] A. F. Bobick and J. W. Davis. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):257–267, 2001.

[12] G. R. Bradski and J. W. Davis. Motion segmentation and pose recognition with motion history gradients. *International Journal of Machine Vision and Applications*, 13(3):174–184, 2002.

[13] J. E. Bresenham. Algorithm for computer control of a digital plotter. *IBM System Journal*, 4(1):25–30, 1965.

[14] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:603–619, 2002.

[15] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):564–575, 2003.

[16] A. Corradini. Dynamic time warping for off-line recognition of a small gesture vocabulary. In *RATFG-RTS '01: Proceedings of the IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems (RATFG-RTS'01)*, page 82, Washington, DC, USA, 2001. IEEE Computer Society.

[17] Y. Cui and J. Weng. Appearance-based hand sign recognition from intensity image sequences. *Appearance-based hand sign recognition from intensity image sequences*, 78(2):157–176, 2000.

[18] R. Cutler and M. Turk. View-based interpretation of real-time optical flow for gesture recognition. In *FG '98: Proceedings of the 3rd. International Conference on Face & Gesture Recognition*, page 416, Washington, DC, USA, 1998. IEEE Computer Society.

[19] T. J. Darrell, I. A. Essa, and A. P. Pentland. Task-specific gesture analysis in real-time using interpolated views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(12):1236–1242, 1996.

[20] T. J. Darrell and A. P. Pentland. Recognition of space-time gestures using a distributed representation. Technical report, MIT Media Laboratory Vision and Modeling Group, 1993.

[21] J. R. del Solar, M. Correa, M. Mascaró, F. Bernuy, S. Cubillos, I. Parra, R. Verschae, R. Riquelme, and J. Galaz. Uchile Homebreakers 2009 team description paper. *RoboCup Symposium*, 2009.

[22] J. R. del Solar and R. Verschae. Robust skin segmentation using neighborhood information. In *ICIP*, pages 207–210, 2004.

[23] J. R. del Solar, R. Verschae, M. Correa, J.-B. Lee-Ferng, and N. Castillo. Real-time hand gesture recognition for human robot interaction. *RoboCup Symposium 2009*, 2009.

[24] J. R. del Solar, R. Verschae, J. Lee-Ferng, and M. Correa. Dynamic hand gesture recognition for human robot interaction. *Latin American Robotics Symposium*, 2009.

[25] J. W. Deng and H. T. Tsui. An HMM-based approach for gesture segmentation and recognition. In *ICPR '00: Proceedings of the International Conference on Pattern Recognition*, page 3683, Washington, DC, USA, 2000. IEEE Computer Society.

[26] P. Domingos and M. Pazzani. On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, 29(2-3):103–130, 1997.

[27] Q. Dong, Y. Wu, and Z. Hu. Gesture segmentation from a video sequence using greedy similarity measure. In *ICPR '06: Proceedings of the 18th International Conference on Pattern Recognition*, pages 331–334, Washington, DC, USA, 2006. IEEE Computer Society.

[28] S. Eickeler, A. Kosmala, and G. Rigoll. Hidden Markov model based continuous online gesture recognition. In *In Int. Conference on Pattern Recognition (ICPR*, pages 1206–1208, 1998.

[29] M. Elmezain, A. Al-Hamadi, J. Appenrodt, and B. Michaelis. A hidden Markov model-based isolated and meaningful hand gesture recognition. *World Academy of Science, Engineering and Technology*, 2008.

[30] S. Faisan, N. Passat, V. Noblet, R. Chabrier, and C. Meyer. Topology preserving warping of binary images: Application to atlas-based skull segmentation. In *MICCAI '08: Proceedings of the 11th international conference on Medical Image Computing and Computer-Assisted Intervention - Part I*, pages 211–218, Berlin, Heidelberg, 2008. Springer-Verlag.

[31] H. Francke, , J. R. del Solar, and R. Verschae. Real-time hand gesture detection and recognition using boosted classifiers and active learning. *Lecture Notes in Computer Science 4872 (PSIVT 2007)*, pages 533–547, 2007.

[32] H. Francke. Diseño y construcción de interfaz humano robot utilizando gestos realizados con las manos, Noviembre 2007. Memoria para optar al Título de Ingeniero Civil Electricista, Universidad de Chile.

[33] A. Güdesen. Quantitative analysis of preprocessing techniques for the recognition of handprinted characters. *Pattern Recognition*, 8(4):219 – 227, 1976.

[34] S. B. Gray. Local properties of binary images in two dimensions. *IEEE Transactions on Computers*, 20(5):551–561, 1971.

[35] D. P. Huttenlocher, G. A. Klanderman, and W. A. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Transactions in Pattern Analysis and Machine Intelligence*, 15(9):850–863, 1993.

[36] G. H. John and P. Langley. Estimating continuous distributions in Bayesian classifiers. *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, pages 338–345, 1995.

[37] K. Kahol, P. Tripathi, and S. Panchanathan. Automated gesture segmentation from dance sequences. In *In Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition, Seoul, Korea*, pages 883–888, 2004.

[38] W. Kong and S. Ranganath. Automatic hand trajectory segmentation and phoneme transcription for sign language. In *8th IEEE Int'l Conference on Automatic Face and Gesture Recognition*, Washington, DC, USA, 2008. IEEE Computer Society.

[39] H.-K. Lee and J. H. Kim. An HMM-based threshold model approach for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10):961–973, 1999.

[40] H. Li and M. Greenspan. Continuous time-varying gesture segmentation by dynamic time warping of compound gesture models. *HAREM 2005: International Workshop on Human Activity Recognition and Modelling*, pages 35–42, 2005.

[41] C.-L. Liu, M. Koga, H. Sako, and H. Fujisawa. Aspect ratio adaptive normalization for handwritten character recognition. In *ICMI '00: Proceedings of the Third International Conference on Advances in Multimodal Interfaces*, pages 418–425, London, UK, 2000. Springer-Verlag.

[42] S. Malassiotis and M. G. Strintzis. Real-time hand posture recognition using range data. *Image and Vision Computing*, 26(7):1027–1037, 2008.

[43] T. M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

[44] P. Morguet and M. Lang. Spotting dynamic hand gestures in video image sequences using hidden Markov models. In *In: ICIP*, pages 193–197. Kluwer, 1998.

[45] G. Nagy. Normalization techniques for handprinted numerals. *Communications of the ACM*, 13(8):475–481, 1970.

[46] T. Ojala, M. Pietikäinen, and T. Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002.

[47] R. Oka. Spotting method for classification of real world data. *The Computer Journal*, 41(8):559–565, 1998.

[48] M. Pietikäinen, T. Ojala, and Z. Xu. Rotation-invariant texture classification using feature distributions. *Pattern Recognition*, 33(1):43 – 52, 2000.

[49] R. Plamondon and S. N. Srihari. On-line and off-line handwriting recognition: A comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):63–84, 2000.

[50] L. R. Rabiner. *A tutorial on hidden Markov models and selected applications in speech recognition*, pages 267–296. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

[51] A. Ramamoorthy, N. Vaswani, S. Chaudhury, and S. Banerjee. Recognition of dynamic hand gestures. *Pattern Recognition*, 36(9):2069–2081, 2003.

[52] M.-C. Roh, B. Christmas, J. Kittler, and S.-W. Lee. Gesture spotting for low-resolution sports video annotation. *Pattern Recognition*, 41(3):1124–1137, 2008.

[53] Y. Sato and T. Kobayashi. Extension of hidden Markov models to deal with multiple candidates of observations and its application to mobile-robot-oriented gesture recognition. *Pattern Recognition, International Conference on*, 2:20515, 2002.

[54] C. Shan, Y. Wei, X. Qiu, and T. Tan. Gesture recognition using temporal template based trajectories. *Pattern Recognition, International Conference on*, 3:954–957, 2004.

[55] S. Stalder, H. Grabner, and L. van Gool. Beyond semi-supervised tracking: Tracking should be as simple as detection, but not simpler than recognition. *On-line Learning for Computer Vision Workshop*, 2009.

[56] T. Starner, J. Auxier, D. Ashbrook, and M. Gandy. The gesture pendant: A self-illuminating, wearable, infrared computer vision system for home automation control and medical monitoring. In *Proceedings of 4th IEEE International Symposium on Wearable Computing*, Atlanta, GA, 2000.

[57] T. Stiefmeier, D. Roggen, and G. Tröster. Gestures are strings: efficient online gesture spotting and classification using string matching. In *BodyNets '07: Proceedings of the ICST 2nd international conference on Body area networks*, pages 1–8, ICST, Brussels, Belgium, Belgium, 2007. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[58] M. Tuceryan and A. K. Jain. *Texture analysis*, pages 235–276. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 1993.

[59] L. Unzueta, O. Mena, B. Sierra, and Ángel Suescun. Kinetic pseudo-energy history for human dynamic gestures recognition. In *AMDO '08: Proceedings of the 5th international conference on Articulated Motion and Deformable Objects*, pages 390–399, Berlin, Heidelberg, 2008. Springer-Verlag.

[60] R. C. Veltkamp. Shape matching: Similarity measures and algorithms. *International Conference on Shape Modeling and Applications, SMI 2001*, pages 188–197, 2001.

[61] R. Verschae, J. R. del Solar, and M. Correa. A unified learning framework for object detection and classification using nested cascades of boosted classifiers. *Machine Vision and Applications*, 19(2):85–103, 2008.

[62] S. B. Wang, A. Quattoni, L.-P. Morency, D. Demirdjian, and T. Darrell. Hidden conditional random fields for gesture recognition. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1521–1527, Washington, DC, USA, 2006. IEEE Computer Society.

[63] T.-S. Wang, H.-Y. Shum, Y.-Q. Xu, and N.-N. Zheng. Unsupervised analysis of human gestures. In *PCM '01: Proceedings of the Second IEEE Pacific Rim Conference on Multimedia*, pages 174–181, London, UK, 2001. Springer-Verlag.

[64] J. A. Ward, P. Lukowicz, and G. Tröster. Gesture spotting using wrist worn microphone and 3-axis accelerometer. In *sOc-EUSAI '05: Proceedings of the 2005 joint conference on Smart objects and ambient intelligence*, pages 99–104, New York, NY, USA, 2005. ACM.

[65] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, second edition, 2005.

[66] S.-F. Wong and R. Cipolla. Continuous gesture recognition using a sparse Bayesian classifier. In *ICPR '06: Proceedings of the 18th International Conference on Pattern Recognition*, pages 1084–1087, Washington, DC, USA, 2006. IEEE Computer Society.

[67] J. Yamato, J. Ohya, and K. Ishii. Recognizing human action in time-sequential images using hidden Markov model. In *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR '92., 1992 IEEE Computer Society Conference on*, pages 379–385, 1992.

[68] M. H. Yang, N. Ahuja, and M. Tabb. Extraction of 2d motion trajectories and its application to hand gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8):1061–1074, 2002.

[69] R. Yang and S. Sarkar. Coupled grouping and matching for sign and gesture recognition. *Computer Vision and Image Understanding*, 113(6):663–681, 2009.

[70] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Computing Surveys*, 38(4):13, 2006.

[71] C. Zhao, W. Shi, and Y. Deng. A new Hausdorff distance for image matching. *Pattern Recognition Letters*, 26(5):581 – 586, 2005.