Reconocimiento y Clasificación de bordes cerrados en Imágenes

Alumnos:

Gentile Fernando
Gonzalez Alejandro

Director:

Armando Di Giusti

INDICE

TABLAS Y FIGURAS

Capítulo I
Motivación y Objetivos

I.1. Motivación.

Actualmente muchas son las áreas de aplicación que hacen uso de lo que se denomina visión computarizada. Entre ellas tenemos el análisis de imágenes médicas, el tratamiento de imágenes satelitales, automatización industrial y la robótica.

La intención de tales sistemas es que la computadora brinde un soporte tecnológico a la interpretación o comprensión de una imagen, reconociendo los distintos objetos o clases de objetos que en ella se encuentran.

La investigación en esta área trata de llevar a la máquina a tener la misma habilidad que la que posee el ser humano en este aspecto. Pero, lejos aún del objetivo, los algoritmos actuales dan soluciones a problemas determinados (de alcance muy acotado) en los que se hace uso de alguna forma de conocimiento previo para la clasificación automática. Este conocimiento previo también puede ser "autoaprendido" por la máquina.

En este contexto un subtema importante de visión computarizada, es el reconocimiento y clasificación de figuras de bordes cerrados, que será el objetivo de esta tesina.

Interesa entonces estudiar los distintos métodos de representación y descripción de bordes cerrados en imágenes para su posterior clasificación, investigando particularmente los problemas en la extracción de características.

Estos problemas nos permiten analizar temas relacionados con el análisis de patrones y procesamiento de imágenes, desde la segmentación de la imagen hasta la clasificación de los distintos patrones que en ella se encuentran.

Por último se pueden analizar variantes de algoritmos tanto desde el punto de vista de exactitud en la clasificación como de eficiencia en el tiempo de ejecución.

I.2. Objetivo.

Estudiar el reconocimiento y clasificación de bordes cerrados en imágenes monocromáticas, poniendo énfasis en el problema de extracción de características y el empleo de las mismas para su clasificación.

Capítulo II
Temas estudiados

II.1. Introducción.

La tecnología actual ha permitido el desarrollo de la visión computarizada aplicable a diversos problemas tales como: reconocimiento de caracteres, análisis de imágenes médicas, automatización industrial, robótica, cartografía, práctica forense, imágenes satelitales, monitoreo remoto; donde los distintos sistemas de visión computarizada tratan de interpretar una imagen a través del procesamiento y análisis de la misma [JAI89]. En la figura II.1.1 se ven distintas aplicaciones.

	APLICACIONES	PROBLEMAS
1	Clasificación de Correo, lectura de etiquetas, lectura de texto.	Reconocimiento de caracteres
2	Detección de tumores, medidas y detección formas de órganos internos, análisis de cromosomas, conteo de células de sangre.	Análisis de Imágenes Médicas
3	Identificación de partes en líneas de ensamble, inspección de efectos y fallas.	Automatización Industrial
4	Reconocimiento e interpretación de objetos en la escena, control de movimiento y ejecución a través de retroalimentación visual.	Robótica
5	Construcción de mapas a partir de fotografías, síntesis de mapas climáticos.	Cartografía
6	Apareo de impresiones digitales y análisis de sistemas de seguridad automatizados.	Forense
7	Identificación y detección de blancos, sistemas de ayuda en helicópteros y aviones para el aterrizaje, asistencia en la conducción de vehículos a control remoto.	Imágenes de Radar
8	Análisis multiespectral de imágenes, predicción de tiempo, clasificación y monitoreo de ambientes urbano, marino de imágenes satelitales.	Sondeo remoto

Figura II.1.1 Tabla de aplicaciones y problemas relacionados con la visión computarizada.

Esta tarea de interpretación es también conocida como análisis de la escena o *análisis de patrones*. En este proceso la computadora analiza una imagen tratando de aislar distintos objetos, los cuales identificará y clasificará, mediante la extracción de características y un manejo heurístico de las mismas.

Básicamente el análisis de patrones involucra el estudio de tres temas:

- Segmentación de la imagen.
- Representación y descripción (Extracción de características).
- Técnicas de clasificación de patrones.

De esta manera una computadora podría diferenciar "inteligentemente" entre imágenes (patrones complejos) u objetos (patrones simples) que posean características

distintas y al mismo tiempo agrupar en familias de imágenes a aquellas que tengan características similares.

Muchas de las tareas que se realizan en el análisis de una imagen pueden ser llevadas a cabo exitosamente por el clásico *Sistema de Reconocimiento de Patrones* mostrado en la figura II.1.2.

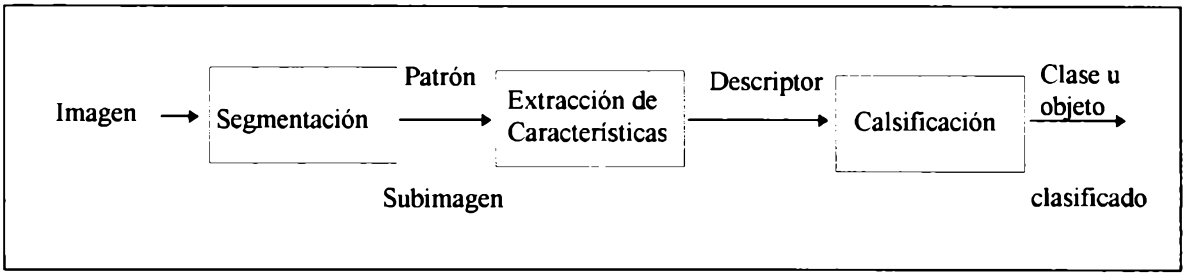


Figura II.1.2. Sistema de Reconocimiento de Patrones.

Estos sistemas reciben, en general, una imagen digital y generan como salida una interpretación de la misma analizando los distintos objetos que la componen. Dado que los tipos de imágenes y las interpretaciones a realizar sobre éstas pueden ser muy variadas, como se refleja en la figura II.1.1, lo que se trata de hacer es definir un sistema de reconocimiento de patrones específico para cada tipo de aplicación [NIE90].

Un sistema de reconocimiento de patrones posee tres funciones o etapas básicas, las cuales son orientadas a los distintos tipos de reconocimientos.

La imagen a analizar es tomada por una primer etapa denominada segmentación, la cual producirá como salida la imagen particionada o segmentada en subimágenes disjuntas que denominaremos patrones u objetos. El tipo de partición realizada por esta etapa dependerá de la clase de reconocimiento que se quiera hacer. Si por ejemplo el sistema pertenece a una aplicación que realiza reconocimiento de caracteres, es de esperar que cada objeto o subimagen, generado en la segmentación, contenga un simple carácter; mientras que si el sistema pertenece a una aplicación que trata de identificar los distintos tipos de suelo en una imagen satelital, es de esperar que cada objeto generado en la segmentación contenga regiones con tonos de grises similares.

En una segunda etapa cada una de las subimágenes que componen la salida de la segmentación son analizadas individualmente por la etapa de extracción de características donde para cada subimagen se genera un conjunto de características o descriptores. Este conjunto, es construido de forma tal que pueda describir de la mejor manera al objeto en cuestión. El tipo de características que formarán parte del patrón, dependerá, nuevamente, del tipo de reconocimiento que el sistema realice. La elección de cuáles características constituirán el patrón es básica e influirá en la tercer etapa de clasificación.

Nuevamente si tomamos los ejemplos anteriores del reconocimiento de caracteres y el análisis de imágenes satelitales, podemos suponer que en el primer caso nos va a interesar características como forma y ancho de los caracteres, mientras que en el segundo nos inclinaremos por características como color y textura.

Luego que se han segmentado los objetos de la imagen y se ha generado un patrón para cada uno, la tercer etapa, clasificación, arma distintas clases de patrones. Cada clase está formada por conjuntos de objetos cuyos descriptores son, de alguna manera, similares.

Por lo antedicho podemos ver que es muy ventajoso y hasta necesario orientar de antemano el sistema a algún tipo de reconocimiento en especial.

Nosotros, como detallamos en el inciso Objetivos, nos dedicaremos al reconocimiento de bordes cerrados sobre imágenes monocromáticas, por lo que estudiaremos sistemas orientados en ese sentido. Más específicamente investigaremos la segunda etapa que compone estos sistemas, evaluando las distintas clases de características, su utilidad, y su complejidad desde el punto de vista computacional.

El análisis de patrones en imágenes es un campo muy vasto dado que este puede tomar distintos caminos en el momento de definir qué tipo de patrones se está buscando.

Un patrón es una descripción cuantitativa o estructural de un objeto o de alguna otra entidad de interés en una imagen. En general un patrón está formado por una o más características; las que tratan de identificarlo o clasificarlo en forma unívoca, por lo que es obvio pensar que el tipo de características a extraer están firmemente relacionadas con el tipo de patrón a buscar.

En síntesis los sistemas de Reconocimiento de Patrones son diseñados para clasificar un patrón de entrada que puede ser una imagen o parte de una imagen, en varias categorías o clases de acuerdo a sus características, utilizando técnicas de procesamiento de imágenes.

II.2. Segmentación.

Generalmente lo primero que se realiza para analizar una imagen es segmentar.

La segmentación subdivide la imagen en sus partes constituyentes u objetos. El nivel al cual es realizada la subdivisión depende del problema que se está resolviendo. Es decir, la segmentación debería terminar cuando los objetos de interés en una aplicación hayan sido separados.

Por ejemplo: Tenemos imágenes aéreas obtenidas vía satélite en forma digitalizada. Dichas imágenes fueron tomadas sobre distintas rutas y caminos y necesitamos identificar diversos tipos de vehículos sobre un tramo de la ruta. El primer paso a realizar sobre la imagen es localizar el camino, dado que este es el objeto de mayor tamaño y marca un límite para nuestra segmentación, ya que nos evitará perder tiempo en la separación de objetos que se encuentran fuera del camino.

Una vez obtenido el camino debemos realizar un segundo paso dirigido ahora a la segmentación del contenido del camino. Nuestro interés está ahora centrado en los tipos de vehículos, los cuales pueden ser particionados según su tamaño. De esta forma los objetos son segmentados dentro de un rango de tamaños que se pueda corresponder con posibles vehículos.

En este ejemplo podemos observar que hay dos niveles de segmentación, y que esta finaliza cuando nuestros objetos de interés (es este caso vehículos) han sido aislados.

En general la segmentación es una de las tareas más difíciles en el procesamiento de imágenes. Este paso en el proceso determina el eventual éxito o falla del análisis. Por ese motivo debemos tomar extremo cuidado en la segmentación.

Básicamente el tipo de segmentación que se realice dependerá de la clase de patrón u objeto buscado. Por ejemplo: tenemos una imagen satelital de las costas de un continente y queremos diferenciar entre el continente y el océano. En este caso sería de conveniencia una segmentación orientada a regiones, para un posterior análisis basado en características como: textura y color.

En cambio, si la imagen es captada por un robot el cual tiene que diferenciar objetos según su forma, convendría una segmentación orientada a bordes, la cual simplificaría la información de la imagen, dejando resaltadas las propiedades morfológicas de los objetos que hay en la misma.

II.2.1 Algoritmos de Segmentación.

Los algoritmos de segmentación para imágenes monocromáticas [GON92] están generalmente basados en dos propiedades básicas de los niveles de grises:

- Discontinuidad.
- Similitud.

Segmentación basada en las discontinuidades de los niveles de grises.

Estos algoritmos recorren la imagen computando la intensidad de los niveles de grises de la misma; para así encontrar algún tipo de discontinuidad en dichos niveles.

Los tipos de discontinuidades más comunes, tales como puntos, líneas y bordes; son encontrados por ofrecer cambios abruptos en los valores de grises.

Muchas veces los bordes encontrados tienen algunas imperfecciones; debido a que el ruido original de la imagen ha producido una iluminación no uniforme de la misma. Ante esta situación es común realizar algún método de encadenamiento; que simplemente se basa en unir aquellos puntos del borde que han quedado aislados luego de haber utilizado el algoritmo de segmentación.

Segmentación basada en las similitudes de los niveles de grises.

Estos algoritmos a diferencia de los basados en discontinuidades se basan en trabajar con las similitudes de los niveles de grises de cada uno de los pixels y de su vecindario.

Estos algoritmos de segmentación particionan la imagen en forma completa dado que la unión de todas las regiones me devuelve la imagen original. Estas regiones deben tener todos sus puntos conectados y cada una de ellas debe cumplir la condición de ser desjunta dos a dos con cada una de las demás regiones.

Es decir nos preocupamos de la región en sí y su contenido, de esta forma cada región tiene una determinada propiedad y cada uno de los pixels dentro de ella deben cumplirla.

Dentro de los algoritmos que cumplen con las condiciones expuestas tenemos los que trabajan con el umbralamiento de acuerdo a determinados valores de grises, el crecimiento de regiones a través del agregado de pixels con determinadas propiedades y los que trabajan con la separación y fusión de regiones.

II.2.1.1 Detección de discontinuidades. Segmentación orientada a puntos, líneas y bordes.

Los tres tipos de discontinuidades en una imagen digital son: puntos, líneas y bordes.

En la práctica la forma más usual de buscar discontinuidades es correr una máscara a través de la imagen.

Una máscara se la representa como una matriz, por ejemplo de 3 X 3. La manera de pasar una máscara a través de la imagen es colocando su centro en el pixel superior izquierdo de la imagen y desplazarla a través de las filas y columnas de la misma. En cada posición calculamos la respuesta de la máscara como la suma de los productos de los coeficientes de la máscara, con los niveles de grises de los pixels sobre los cuales la máscara va siendo ubicada.

Ejemplo:

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Si tenemos esta máscara y la ubicamos sobre una porción de una imagen cuyos pixels tiene los siguientes niveles de grises $z_1..z_q$, entonces la respuesta de la máscara en ese punto es:

$$R = \sum_{i=1}^{q} w_i z_i$$

Estas matrices son también llamadas *filtros espaciales,* y pueden tener distintas formas y/o tamaños.

Según como se configuren los coeficientes de la matriz vamos a obtener distintas respuestas ante la presencia de puntos, líneas o bordes de la imagen filtrada.

A continuación veremos algunas matrices especializadas en la detección de puntos, líneas y bordes.

Detección de puntos

En general el tipo de matriz utilizada para la detección de puntos es aquella que tiene un valor positivo en el centro y negativo en su vecindario.

-1	-1	-1
-1	8	-1
-1	-1	-1

Utilizando la máscara decimos que un punto ha sido detectado en el lugar en el cual la máscara está centrada si |R| > T, donde T es un umbral no negativo y R es la respuesta de la máscara. [PRA78]

Esta formulación mide las diferencias pesadas entre el punto central y sus vecinos. La idea es que el nivel de gris de un punto aislado será lo suficientemente distinto del nivel de gris de sus vecinos.

Detección de líneas

Igual que en la detección de puntos existen ciertos tipos de matrices cuya respuesta es mayor ante la presencia de líneas.

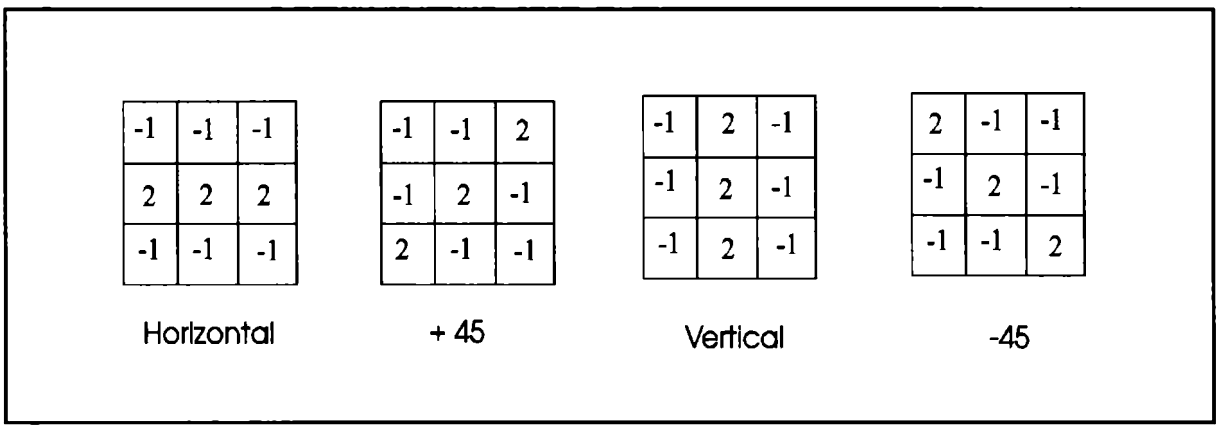


Figura II.2.1. Matrices utilizadas para detectar líneas.

Las matrices de la figura II.2.1 son un ejemplo de máscaras espaciales de 3X3 para la detección de líneas. Contienen una hilera de coeficientes positivos y el vecindario con coeficientes negativos. La orientación de línea a detectar dependerá de la orientación de la hilera en la máscara.

Detección de bordes

Un borde es el límite entre dos regiones con niveles de grises distintos, por lo tanto, si podemos encontrar en que lugar se produce el cambio en los niveles de grises, fácilmente aislaremos el borde. Una manera de hacer esto es ver la imagen 2-D como una función en el plano (x,y,z), de manera que x e y representen la ubicación del pixel y "z" su nivel de gris. [GON92]

Viendo una imagen en 3D (donde los ejes X,Y son la ubicación del pixel y el eje Z es el nivel de gris en esa ubicación) y haciendo un corte transversal de la misma, un borde puede ser visto de la siguiente manera.

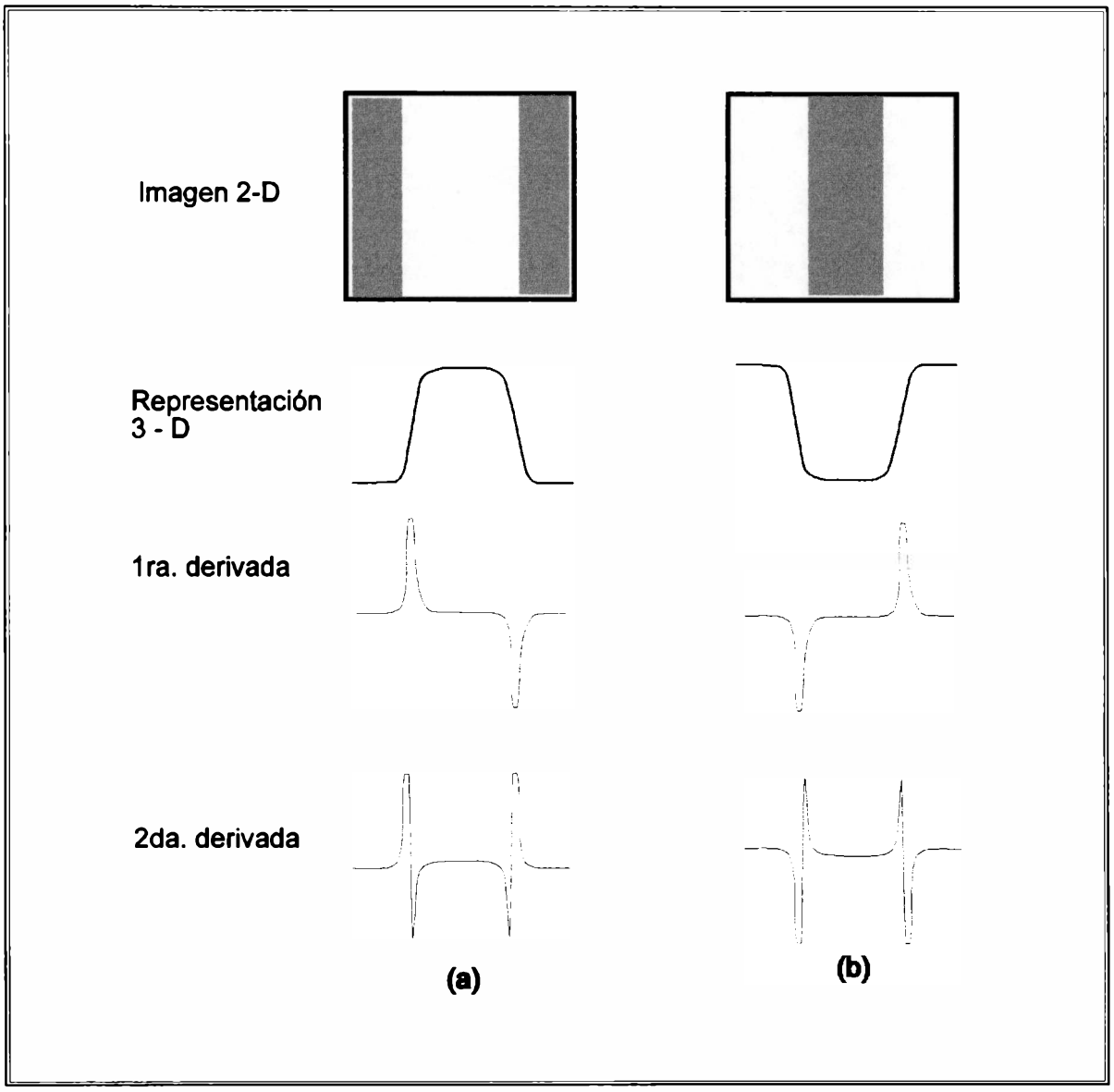


Figura II.2.2. Representación de un borde usando derivadas.

La figura II.2.2(a) muestra un ejemplo de una función 2-D y el corte transversal de su representación 3-D. En la primera derivada del corte transversal observamos que la misma es distinta de cero cuando se produce un cambio en los niveles de grises, positiva cuando el cambio es de negro a blanco y negativa de blanco a negro. Por lo anterior, la magnitud de la primera derivada puede ser usada para la detección de un borde. Con la segunda derivada vemos que es positiva cuando estamos del lado oscuro del borde y negativa del lado claro, por lo tanto nos es útil para saber si un pixel determinado pertenece al lado oscuro o al claro de un borde. Algo para tener en cuenta es que la segunda derivada se anula a mitad de la transición.

La primera derivada en cualquier punto de la imagen es obtenida calculando la magnitud del operador gradiente en ese punto, y la segunda derivada es obtenida con el operador Laplaciano.

Ahora veremos como podemos calcular el Gradiente y el Laplaciano de una imagen.

Operador Gradiente

El operador Gradiente de una imagen f(x,y) lo definimos como:

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{df}{dx} \\ \frac{df}{dy} \end{bmatrix}$$

este está basado en la computación de las derivadas parciales de f en cada pixel. Una de las mejores maneras de implementar las derivadas en forma digital es utilizando los operadores de Sobel los cuales calculan a G_x y G_y de la siguiente manera:

$$G_x = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

y

$$G_y = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

lo cual definen a las máscaras de la figura

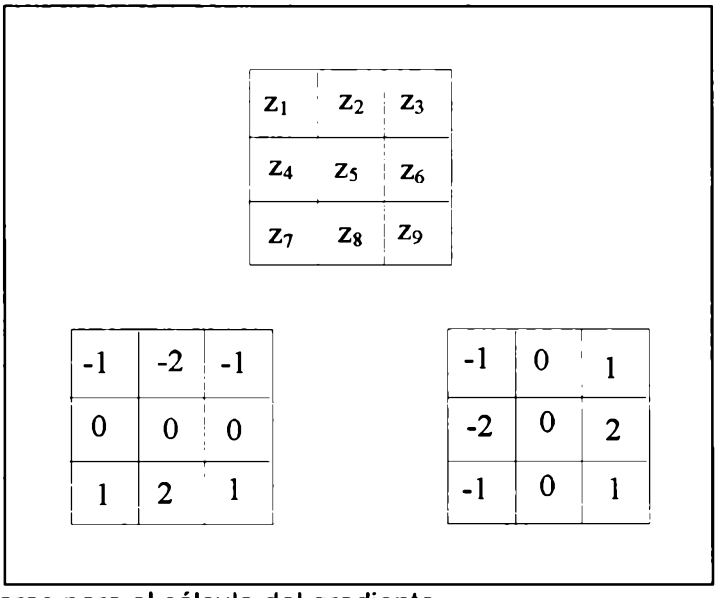


Figura II.2.3. Máscaras para el cálculo del gradiente.

Pasando estas máscaras por toda la imagen f(x,y) obtenemos la imagen gradiente del mismo tamaño que la original.

Operador Laplaciano

El Laplaciano de una función 2-D f(x,y) está definido como:

$$\nabla^2 f = \frac{d^2 f}{dx^2} + \frac{d^2 f}{dy^2}$$

de igual manera que con el gradiente, el Laplaciano se basa en las derivadas de f(x,y) y puede ser implementado en forma digital de varias maneras [PRA78]. Dada una máscara de 3x3, el cálculo del Laplaciano se reduce a:

$$\nabla^2 f = 4z_5 - (z_2 + z_4 + z_6 + z_8)$$

donde el requerimiento básico en la definición del Laplaciano digital, es que el coeficiente asociado con el pixel central debe ser positivo y los coeficientes asociados con la vecindad deben ser negativos.

Como vimos en la figura II.2.2, el Laplaciano [Gon92] no es comúnmente usado para la detección de bordes por varias razones como:

- Produce bordes dobles.
- Es muy sensitivo al ruido.

Por estas características el Laplaciano juega un papel secundario y es utilizado para establecer cuando un pixel cae sobre una zona oscura o clara de la transición.

II.3. Representación y descripción.

Se trata de seleccionar qué y cómo se guardará la información de cada objeto, extrayendo características que resulten en alguna información cuantitativa de interés o de reconocimiento básico para la diferenciación entre distintas clases de objetos. Dado que esta etapa recibe objetos en forma de conjuntos de pixels aparentemente sin ningún tipo de significado, lo que se trata de hacer es guardar en alguna estructura, no al objeto en su totalidad sino a una representación del mismo, que lo identifique en forma precisa y más abreviada. Es de esperar que dicha representación sea más útil en la computación de las características que lo describirán.

En la representación de un borde no hay pérdida de información dado que se trata del mismo objeto codificado de otra manera, por ejemplo: rúbrica. La representación del borde es para facilitar la extracción de alguna característica en especial. Por lo tanto el borde puede ser recuperado en su forma original.[SCH92].

En la descripción se tiene una o varias características que intentan identificar a un borde unívocamente, a este conjunto de características se lo denomina *patrón* y aunque identifican al borde, este no es recuperable a partir de las mismas.

Entonces vamos a trabajar con representaciones que estén orientadas a la forma y plantear si realmente se adaptan o no al tipo de reconocimiento que queremos llevar a cabo. [GON92].

Dentro de estas representaciones tenemos:

- ♦ Código Cadena.
- ♦ Rúbrica.
- ♦ Aproximación Poligonal.

II.3.1 - Representación.

En la etapa de representación se decide cómo se guardará la información de cada objeto segmentado. La representación elegida para el objeto trata de facilitar el procesamiento posterior del mismo. Como en nuestro análisis nos interesa la forma de los bordes, ciertas características como color o posición del pixel podrían no ser incluidas en dicha representación.

II.3.1.1 - Código Cadena.

Un código cadena representa un borde de una figura el cual se obtiene codificando los segmentos que hay entre pixels sucesivos [GON92] . Esta representación se basa en la cuatro-conectividad o en la ocho-conectividad de dichos segmentos. Para conseguir los pares de pixels se superpone el borde a una cuadrícula y se sigue el límite tomando los puntos de intersección de la cuadrícula más cercanos al borde. Luego se asigna un código de dirección a cada segmento que conecta cada par de puntos; quedando así generado el código cadena. La dirección de cada segmento es codificada usando el esquema mostrado en la siguiente figura:

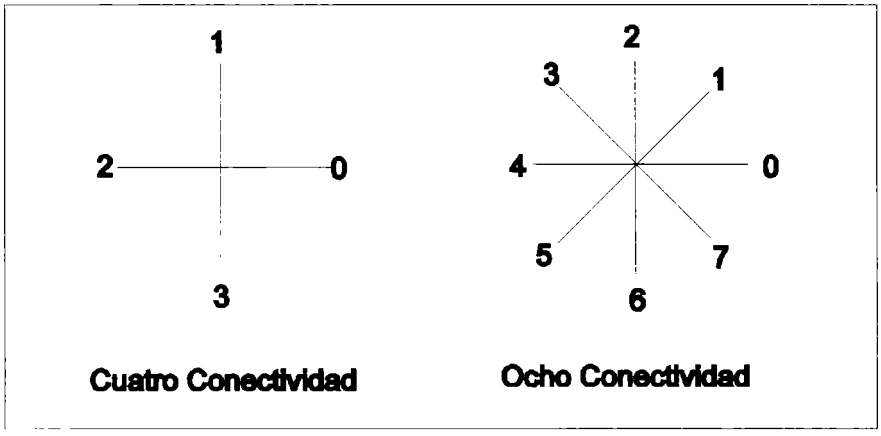


Figura II.3.1. Conectividad de los pixels.

La cuadrícula más detallada y "natural" que se pueden tomar son los mismos pixels de la pantalla pero esta genera un código cadena de n dígitos para un borde de n pixels o puntos, lo cual hace que nos quedemos con una representación muy grande. Por este

motivo, generalmente, se toma cuadrículas más grandes, las cuales tienen la función de muestrear el borde a una mayor frecuencia, obteniendo códigos cadena menos exactos pero más cortos.

Observando el código cadena obtenido en la figura II.3.2 usando 8-conectividad, vemos que el mismo depende de su punto de comienzo como también del sentido en que recorramos el borde.

No es lo mismo comenzar por el punto A que por el punto B. Suponiendo que el sentido de recorrido es el mismo, obtenemos los siguientes códigos:

Punto A: 4456676666022121223.
Punto B: 2234456676666022121.

Una manera de que los códigos cadena nos sean útiles es calcular el *mínimo,* el cual se obtiene tratando al código como una secuencia circular y haciendo desplazamientos hasta obtener el mínimo entero posible. Para el ejemplo anterior el mínimo sería:

Mínimo: 0221212234456676666.

como único código cadena para ese borde. Por lo tanto el mínimo nos resuelve el problema de la elección del número de comienzo.

Ahora bien, ¿qué pasa si calculamos el código cadena del mismo borde pero rotado q grados? es obvio que el código se verá afectado por esta rotación, por lo que una mejora a este procedimiento es normalizar el vector diferencia en vez del código cadena en sí. El vector diferencia se obtiene contando el número de direcciones que separan dos elementos adyacentes del código. Nótese que de esta manera el código obtenido es independiente de la rotación del objeto.

Por ejemplo:

Código Cadena:	30333
Diferencia:	13000

Otro punto a tener en cuenta es que para poder comparar 2 códigos cadenas, estos deben ser del mismo orden, por esto se entiende que deben haber sido calculados con cuadrículas del mismo tamaño.

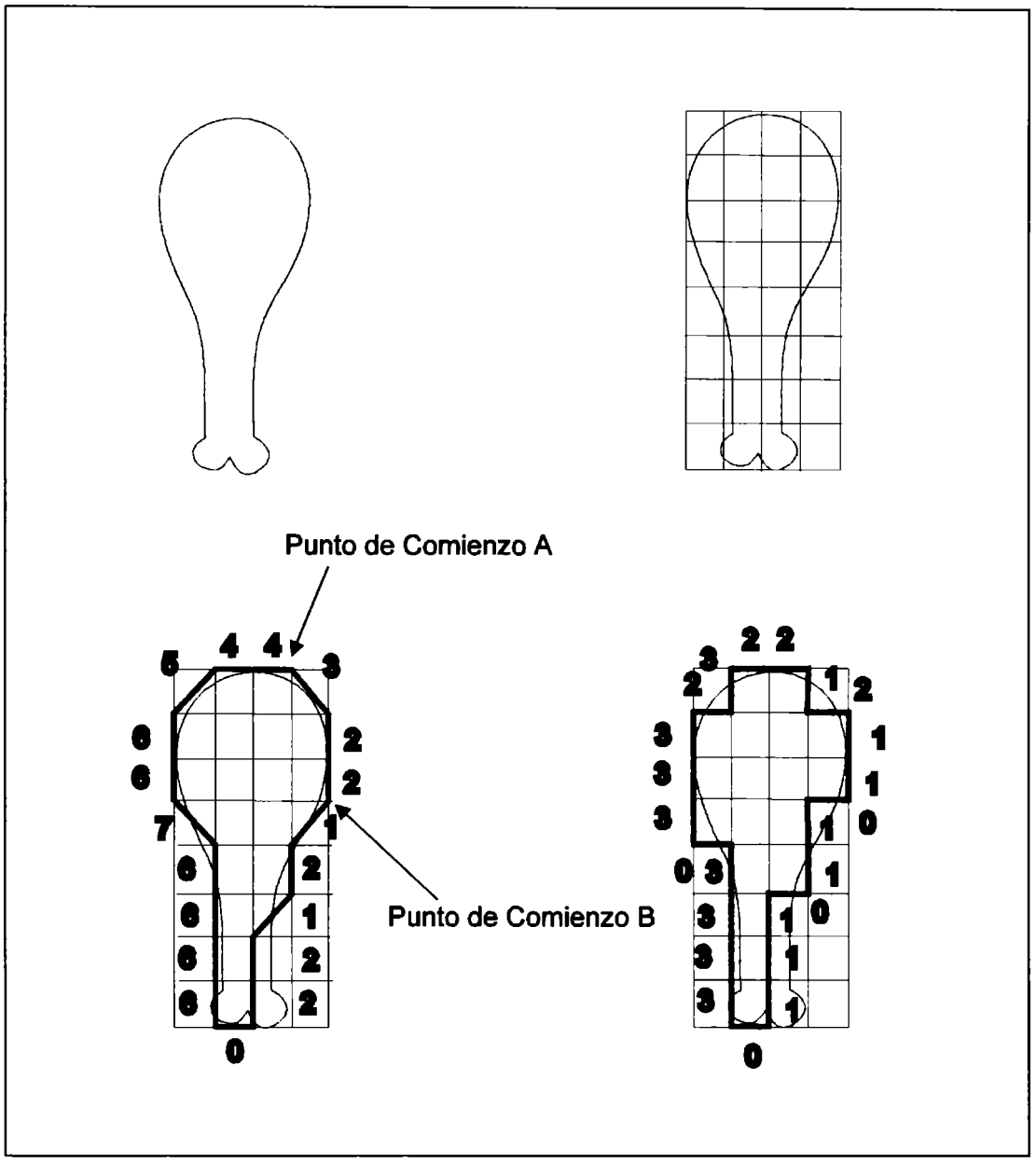


Figura II.3.2 - Formas de obtener el Código Cadena.

II.3.1.2 - Rúbrica.

Es una representación 1-D de un borde. Esta función puede ser generada de varias maneras:

la más simple es hacer una gráfica de la distancia desde el centroide al borde como una función del ángulo.

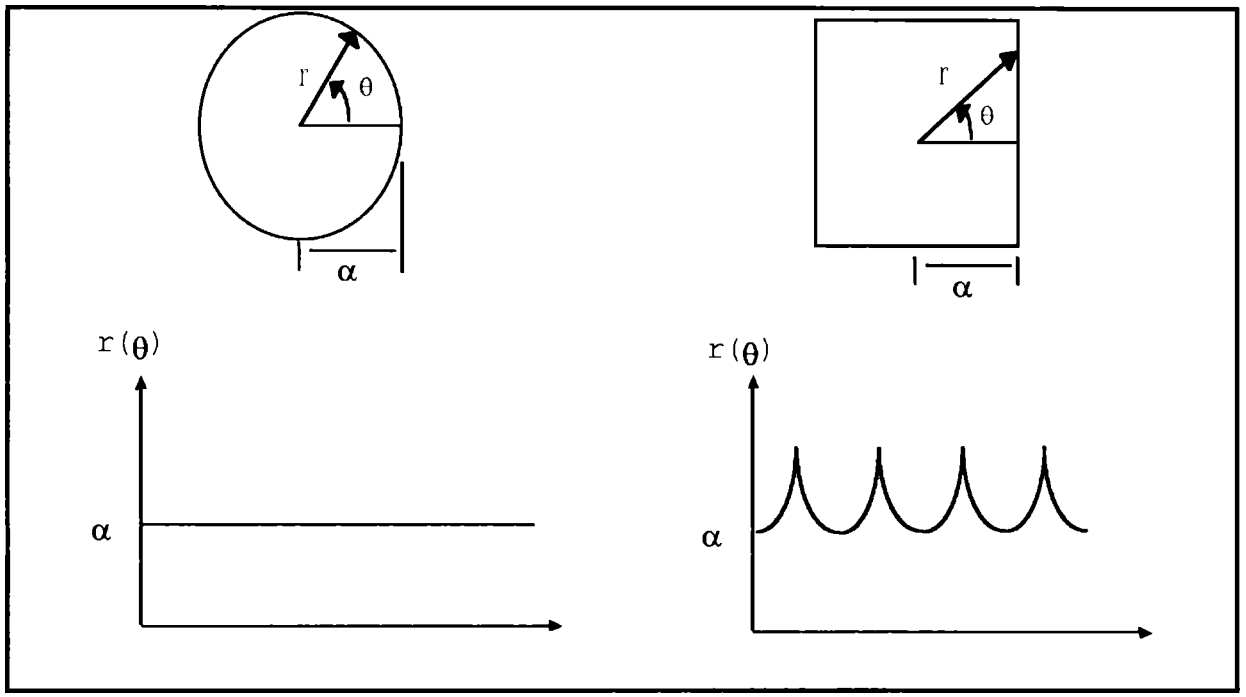


Figura II.3.2. Rúbrica de diferentes figuras, tomando la distancia del centro al borde.

El objetivo es el de reducir en una dimensión la representación, lo cual supone que facilita su tratamiento. Esta así calculada es invariante a la traslación aunque dependiente a la rotación y el escalamiento. [HUS91]. Para evitar la dependencia a la rotación, se genera la rúbrica comenzando siempre desde un mismo punto, por ejemplo: elegir el punto más lejano al centroide, si este es único, o elegir el punto sobre el eje mayor más lejano del centroide.

El cambio de tamaño en la figura genera cambios en la amplitud de la gráfica. Esto puede ser evitado dividiendo los valores de la función por su máximo y haciéndola variar entre [0,1].

Otra manera de generar la rúbrica es trazando una línea de referencia y dibujando la función que represente la medida del ángulo entre esta y la tangente en cada punto del borde.

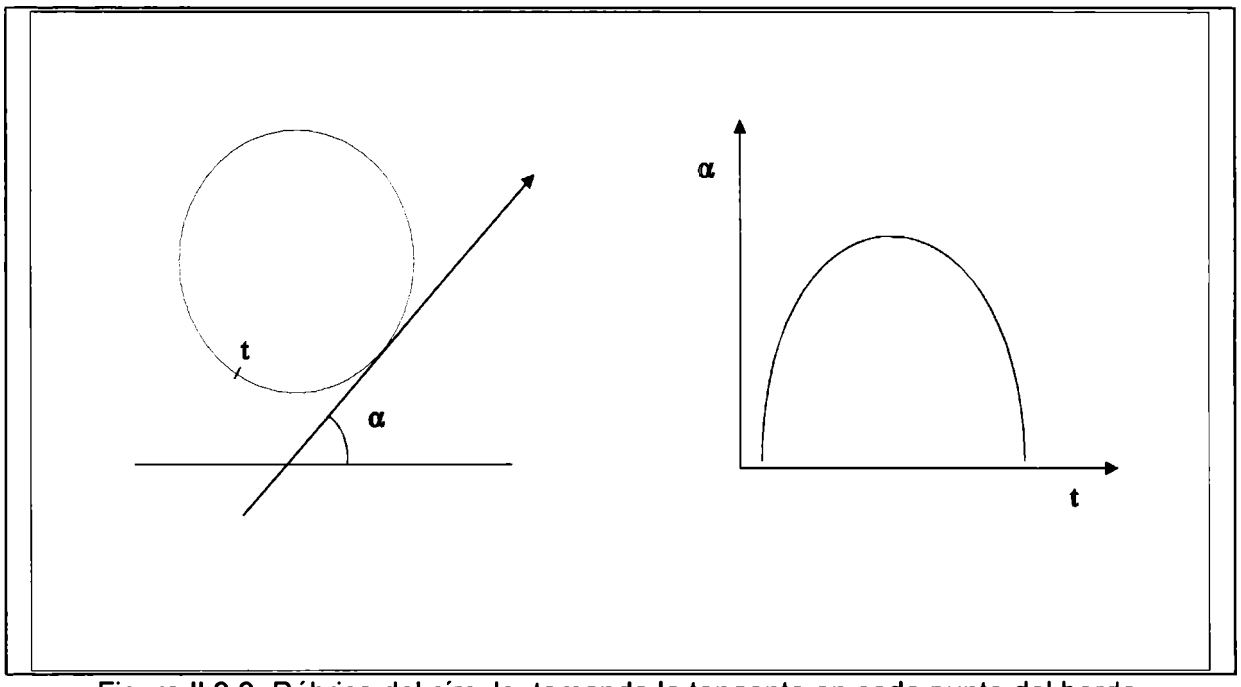


Figura II.3.3. Rúbrica del círculo, tomando la tangente en cada punto del borde.

II.3.1.3 - Aproximación poligonal.

Un borde digital puede ser aproximado, con bastante exactitud, por un polígono. Para una curva cerrada ,la aproximación es exacta cuando el número de segmentos en el polígono es igual al número de puntos en el borde. En la práctica, el objetivo es capturar la esencia de la forma del borde con la menor cantidad de segmentos posibles, lo cual no es una tarea trivial y consume mucho tiempo de computación.

Un procedimiento para encontrar el polígono de mínimo perímetro se ve en el siguiente ejemplo:

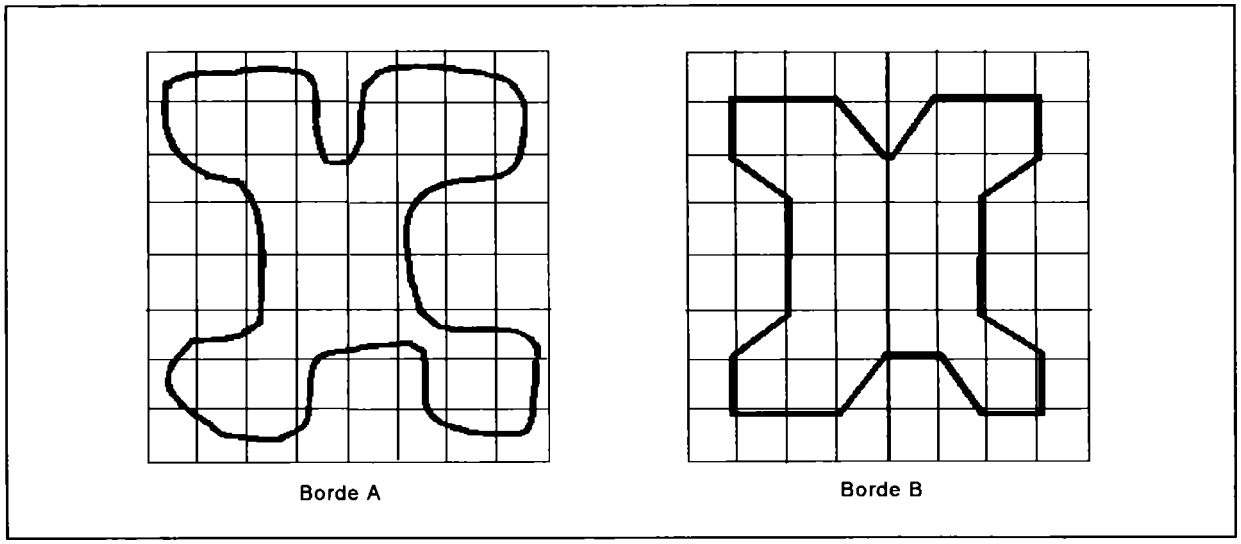


Figura II.3.4 Aproximación poligonal de un borde.

- Supongamos tener el borde A de la figura II.3.4
- Encerramos el borde por un conjunto de celdas concatenadas de manera que el borde quede entre dos "paredes".
- Suponiendo que el borde fuera una banda elástica y pudiera contraerse, entonces quedaría como el borde B de la figura II.3.4.

De esta manera podría reconstruir el borde a partir de los puntos que unen los segmentos los cuales son menos que los originalmente contenidos por el borde.

Otra manera de encontrar una aproximación poligonal del borde es subdividir el segmento sucesivamente en dos partes hasta que algún criterio sea satisfecho. Por ejemplo, un requerimiento podría ser que la máxima distancia perpendicular desde el borde hasta la línea que une los dos puntos no exceda un cierto valor prefijado.

En el siguiente ejemplo se ve reflejado este método:

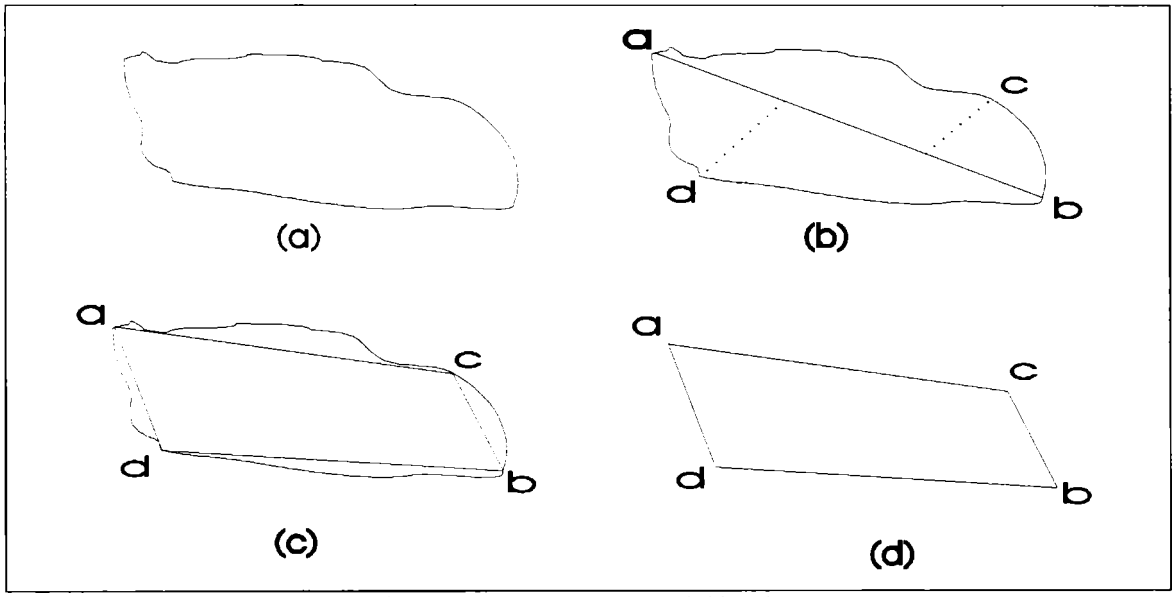


Figura II.3.5. Aproximación poligonal por subdivisión de segmentos.

El siguiente paso consiste en subdividir los dos nuevos segmentos. Para ello buscamos el punto más distante ,en forma perpendicular, a cada uno de los segmentos. En nuestro ejemplo serían los puntos: c correspondiente al segmento ab y el punto d correspondiente al segmento ba. Así el segmento ab quedó dividido en ac y cb, y el segmento ba en bd y da.

Dado que las rectas formadas cumplen con la condición de cercanía a los segmentos, no es necesario seguir subdividiéndolos, quedando la aproximación poligonal D de la figura II.3.5.

Pero no siempre ocurre esto y es necesario continuar con la subdivisión; por lo tanto debe ser buscado nuevamente el punto más distante a cada segmento y trazar las rectas que unan a esos puntos.

II.3.1.4 - Descriptores de Fourier.

Los descriptores de Fourier se generan a partir de las coordenadas de los puntos pertenecientes al borde [JAI89].

Supongamos tener un borde digitalizado en el plano xy compuesto por N puntos. Comenzando en un punto arbitrario (x_0,y_0); debemos encontrar los pares $(x_0,y_0),(x_1,y_1)..(x_{N-1},y_{N-1})$ recorriendo el límite en sentido antihorario. Estas coordenadas pueden ser expresadas de la forma $x(k)=x_k$ e $y(k)=y_k$ y así el borde pasaría a ser una secuencia de coordenadas de la forma

$$s(k) = [x(k), y(k)]; \text{ para } k = 0..N-1$$

Finalmente se representará cada coordenada como un número complejo donde x es la parte real e y la parte imaginaria [GON92]. Esto sería

$$s(k) = x(k) + j\, y(k)$$

Esta redefinición de puntos no afecta a la naturaleza del borde y ha reducido el problema de 2-D a 1-D.

Ahora podemos tomar la DFT (Discret Fourier Transform) de s(k) y nos da:

$$a(u)=\frac{1}{N}\sum_{k=0}^{N-1} s(k)\cdot e^{[-j2\pi uk/N]}$$

para u = 0.. N - 1. Los coeficientes complejos a(u) son llamados *descriptores de Fourier* del borde.

Si aplicamos la transformada inversa de Fourier a los a(u) obtendremos los s(k) originales, esto es:

$$s(k)=\sum_{u=0}^{N-1} a(u)\cdot e^{[j2\pi uk/N]}$$

para k = 0..N-1. Por definición de la transformada de Fourier, la aplicación de la misma no produce pérdida de información, por lo que se sigue teniendo el borde pero representado de otra manera; ahora en vez de tener N coordenadas enteras, tenemos N coordenadas complejas. A continuación veamos cuál es la ventaja de tener los N descriptores de Fourier en vez de los N puntos.

Supongamos que para reconstruir los s(k) con k = 0..N-1 no utilizamos todos los a(u), sino sólo una cantidad C < N. De esta manera obtendríamos las siguientes aproximaciones de los s(k):

$$\hat{s}(k)= \sum_{u=0}^{M-1} a(u)\cdot e^{[j2\pi uk/N]}$$

para k = 0..N-1. Aunque solamente C términos son usados para obtener cada componente de $\hat{s}(k)$; k sigue variando entre 0..N-1. Esto significa que el mismo número de puntos existe en el borde aproximado, pero no se utilizan tantos términos para reconstruir cada punto. Obviamente ante la eliminación de descriptores, la reconstrucción del borde no es exacta, pero lo que se paga en exactitud se gana en la disminución de puntos que se necesitan almacenar. Si el número de puntos en un borde es muy grande, C es generalmente seleccionado como un entero que sea potencia de 2 para que el algoritmo de la FFT (Fast Fourier Transform) pueda ser usado y así acelerar la computación de los descriptores. Recordemos que de acuerdo a las propiedades de la transformada de Fourier, las componentes de frecuencias altas son responsables de los detalles más finos, mientras que las componentes de frecuencias bajas determinan la forma global. Por lo tanto al reconstruir los s(k) con menos descriptores, lo que estamos perdiendo son los detalles de la forma del borde y quedándonos con su forma global.

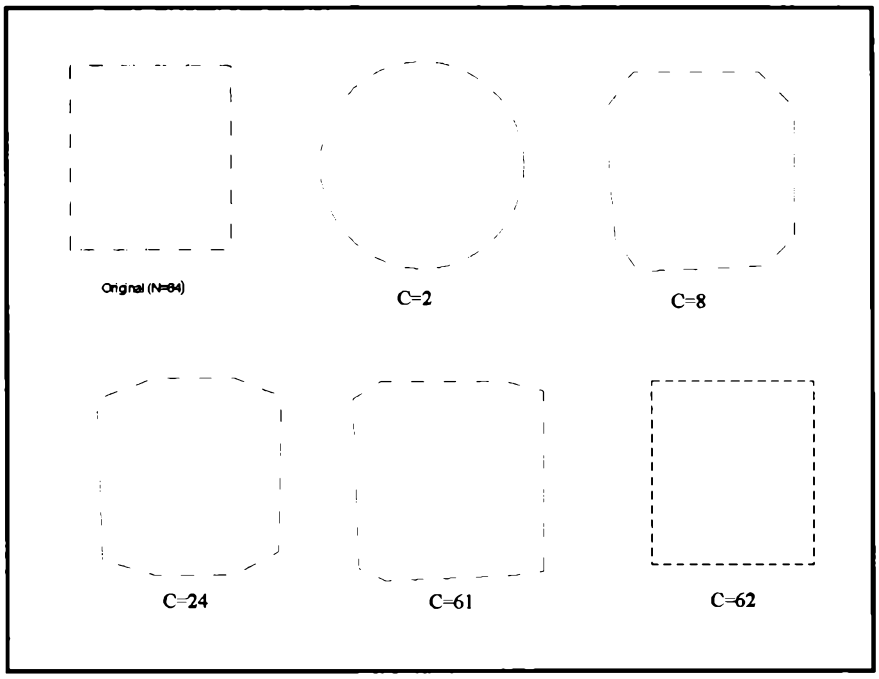


Figura II.3.6. Ejemplos de reconstrucciones de Descriptores de Fourier para varios valores de C.

Como ejemplo podemos ver la figura II.3.6 donde originalmente tenemos un borde cuadrado de 64 puntos (N = 64), el cual se lo descompone en sus 64 descriptores, según el

proceso antes descripto y luego se lo trata de reconstruir calculando los s(k) tomando diferentes valores para C. Comparando los bordes resultantes para los distintos C, observamos que:

- Para C < 8 el borde obtenido tiene forma de círculo.
- Con 8 < C < 56 se tiene un borde que se asemeja a un cuadrado con vértices redondeados.
- A partir de C = 56 hay un cambio notable en las esquinas, las cuales muestran sus puntos como rompiendo la secuencia.
- En C = 61 las curvas se transforman en rectas.
- Finalmente en C = 64 se recupera el borde original.

Así podemos ver que los coeficientes de bajo orden son capaces de rescatar lo grueso de la figura; pero muchos más términos de alto orden son necesarios para describir adecuadamente las características de la forma, tales como esquinas y líneas rectas.

El inconveniente que existe con estos descriptores es que son sensibles a la traslación, rotación, punto de comienzo o escalamiento del borde a tratar. Pero esto puede ser salvado y ser utilizados para compararla similitud entre diversos bordes.

Supongamos que a(k) y b(k) son descriptores de dos bordes u(n) y v(n) respectivamente, entonces si

$$d(\mu_0,\alpha,\theta_0,n_0) = min\{\sum_{n=0}^{N-1}\left(u(n) - \alpha v(n-n_0)^{\theta_0} - u_0\right)^2\}$$

es pequeña se puede decir que u(n) y v(n) son similares. Los parámetros $\mu_0,\alpha,\theta_0,n_0$ son elegidos para minimizar los efectos de traslación, escalamiento, punto de comienzo y rotación.

Si u(n) y v(n) son normalizados de manera que $\sum u(n) = \sum v(n) = 0$, entonces para un cierto n_0 la distancia anterior es mínima cuando:

$$u_0 = 0$$

$$\alpha = \frac{\sum_k c(k)\cos\left(\psi_k + k_\phi + \phi_0\right)}{\sum_k \left[b(k)\right]^2}$$

y

$$tan\theta_0 = \frac{\sum_k c(k) sin(\psi_k + k_\phi)}{\sum_k c(k)\cos(\psi_k + k_\phi)}$$

donde $a(k)b(k) = c(k)e^{j\psi k}$, $\phi = \frac{-2\pi n_0}{N}$ y $c(k)$ es un real. Estas ecuaciones nos dan α y ϕ_0, con los cuales la distancia mínima queda expresada de la siguiente forma:

$$d = min_\theta\left[d(\theta)\right] = min_\phi\{\sum_k\left[a(k) - \alpha b(k)e^{\left[j(k_\phi + \theta_0)\right]}\right]^2\}$$

La distancia $d(\theta)$ puede ser calculada para cada $\phi = \phi(n_0)$, $n_0 = 0..N-1$ y así se obtiene el mínimo d. El valor d es una medida útil para estimar la similitud de dos figuras.

II.3.2 -Descripción.

La etapa de descripción recibe la representación de un borde, la cual se analiza con el fin de obtener aquellas propiedades relevantes para su posterior clasificación. Cada una de estas propiedades o características recibe el nombre de *descriptor* y el proceso de obtenerlas recibe el nombre de e*xtracción de característica.*

En general para almacenar las distintas características de un objeto se suele utilizar, como estructura básica, vectores.

En cuanto a la extracción de características, se deben almacenar aquellas que mejor describan al patrón, lo cual dependerá del tipo de reconocimiento a realizar.

Por ejemplo: en el caso de análisis de fotos satelitales donde se quiere hacer el reconocimiento de distintos tipos de suelos (campos de trigo, bosques, ciudades) algunas características relevantes podrían ser color y textura, mientras que tamaño y forma no servirían para este tipo de descripción.

II.3.2.1 Extracción de Características.

Una vez que una imagen ha sido segmentada y representada en objetos discretos de interés, el próximo paso en el proceso de análisis de imágenes es medir las características individuales de cada objeto. Para describir un objeto se pude tener una o varias características las cuales pueden ser comparadas con medidas conocidas para la clasificación del objeto entre distintas clases. Algunos de los tipos de características que podemos extraer son: brillo, color, textura, forma, número de vértices, redondez, etc.

La clave es elegir y extraer características que tengan las siguientes propiedades:

- Computacionalmente factibles.
- Relacionadas con el tipo de reconocimiento que se desea llevar a cabo.

- Que reduzcan los datos a una cantidad manejable de información, sin pérdida de información vital

Lo más importante para aplicar los diversos métodos de reconocimiento de figuras cerradas en una imagen digital, es considerar la información de los límites y bordes de dicha figura.

Habitualmente el patrón que queremos clasificar puede sufrir ciertas distorsiones como traslación, rotación, escalamiento (cambio de tamaño), etc., por lo tanto se deseará que los descriptores del patrón sean invariantes a estos cambios.

Problemas como estos son los que nos obligan a hacer una buena selección de las características que usaremos para construir los patrones.

II.3.2.2 Características.

- Características de Amplitud:

Una de las características de las imágenes es la medida de la amplitud en términos de su luminosidad u otras unidades. Esta medida pude ser tomada de puntos específicos o sobre su vecindad. Una medida clásica es la luminosidad promedio.

- Características del Histograma:

Dado el histograma $P(b) \approx N(b)/M$, donde *M* es la cantidad de pixels en una ventana centrada en (x,y), y *N(b)* es el número de pixels de amplitud b en esa ventana. Algunas características son:

⇒ Promedio:

$$\bar{b} = \sum_{b=0}^{L-1} bP(b)$$

⇒ Varianza:

$$\sigma_b^2 = \sum_{b=0}^{L-1} (b - \bar{b})^2 P(b)$$

⇒ Energía:

$$b_N = \sum [P(b)]$$

⇒ Inclinación: (Skewness)

$$b_s = \frac{1}{\sigma_b^3} \sum_{b=0}^{L-1} (b - \overline{b})^3 P(b)$$

⇒ Entropía:

$$b_E = \sum_{b=0}^{L-1} P(b) \log_2 [P(b)]$$

♦ Medidas de la forma o geométricas:

Las medidas de la forma son medidas físicas dimensionales que caracterizan la apariencia de un objeto. La lista de dichas medidas puede hacerse muy larga, una operación particular de análisis de una imagen utiliza un subconjunto de posibles medidas. En las aplicaciones de análisis de imágenes la meta es utilizar la menor cantidad de medidas necesarias para caracterizar a un objeto adecuadamente, de manera que no resulte ambiguamente clasificado. [BAX94]

Entre las medidas citamos las siguientes:

⇒ Perímetro.

Es la medida de la longitud del borde de un objeto. Para computar esto adecuadamente, donde un pixel del borde toca su vecino vertical u horizontalmente, la distancia entre pixels es 1 unidad. Donde un pixel toca a su vecino en diagonal, la distancia al pixel es raíz cuadrada de 2, o 1.414 unidades.

⇒ Área.

El área es computada como el número de pixels interiores, incluidos los del borde. El resultado es el tamaño del objeto, y usualmente no incluye las áreas agujereadas.

⇒ Proporción del Área al Perímetro.

Es una medida de la redondez del objeto, dando un valor entre 0 y 1. Si la proporción es igual a 1, el objeto es un círculo perfecto, si decrece de 1 el objeto se aleja de su forma circular. La redondez se calcula como:

$$\frac{4*Pi*Area}{Perímetro^2}$$

⇒ Eje Mayor.

Es la recta que une los 2 puntos pertenecientes al borde más distantes entre sí, cabe acotar que para la computación del eje mayor hay que calcular la distancia confrontando todos los puntos del borde.

⇒ Longitud del Eje Mayor.

Es la distancia entre los puntos finales del Eje Mayor. Se calcula:

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

donde los puntos (x_1,y_1); (x_2,y_2) son los puntos finales del eje mayor.

⇒ Ángulo del Eje Mayor.

Es el ángulo entre el eje mayor y el eje x de la imagen. El ángulo varía entre 0° y 360°. El resultado es una medida de orientación del objeto.

$$\tan^{-1}\frac{(y_1 - y_1)}{(x_2 - x_1)}$$

donde los puntos (x_1,y_1);$(x_2 ;y_2)$ son los puntos finales del eje mayor.

⇒ Eje Menor.

Son los puntos finales (x,y) de la línea más corta que puede ser dibujada a través del objeto, manteniendo la perpendicularidad con el eje mayor. Estos puntos son encontrados computando las distancias entre cada combinación de pixels del borde y nos quedamos con el par de longitud máxima.

⇒ Excentricidad.

Proporción de las longitudes de los Ejes Mayor y Menor.

Excentricidad = Longitud Eje Menor /Longitud Eje Mayor

⇒ Área de la caja borde o rectángulo base.

La calculamos de la siguiente manera:

Área Caja = Longitud Eje Menor X Longitud Eje Mayor

⇒ Rectángulo base.

El rectángulo base es el menor de los rectángulos que encierra al borde, el mismo está alineado con la orientación del borde. Una vez que la orientación θ ha sido encontrada usamos la siguiente transformación:

$$\alpha = x\cos\theta + y\,\mathrm{sen}\,\theta$$
$$\beta = x\,\mathrm{sen}\,\theta + y\cos\theta$$

Sobre los puntos del borde y buscamos los $\alpha max, \alpha min, \beta max, \beta min$ .Con ellos podemos calcular la longitud de sus lados:

$$\mathrm{largo} = \alpha max - \alpha min$$
$$\mathrm{ancho} = \beta max - \beta min$$

Y así poder calcular la proporción: (largo*ancho/área); donde área es la del borde.

Invarianza de las medidas.

Cuando hablamos de la Invarianza en las medidas nos referimos a ciertas medidas que son insensibles a varianzas particulares. Cuando una medida es insensible a una varianza, el valor no cambiará en respuesta a ella; así la medida es invariante.

- Características basadas en momentos:

⇒ Definición

Para una imagen digital el momento de orden (p+q) es definido como:

$$m_{pq} = \sum_x \sum_y x^p y^q f(x,y) \text{para p,q = 0,1,2....}$$

El teorema de unicidad de Papoulis(1965) establece que si f(x,y) es continua a trozos y tiene valores distintos de cero en una parte finita del plano xy, los momentos de todos los órdenes existen y la secuencia de momentos m_{pq} determina unívocamente a f(x,y).

Los momentos centrales pueden ser expresados:

$$\mu_{pq} = \sum\sum (x - \bar{x})^p (y - \bar{y})^q f(x,y)$$

donde: $\bar{x} = m_{10}/m_{00}$ $\bar{y} = m_{01}/m_{oo}$

Esto lo hace invariante a la traslación.

Los momentos centrales normalizados, denotados:

$\eta pq = \mu pq/\mu_{00}^{t}$ donde $t = (p+q)/2$, para p + q = 2,3...

lo hace invariante al escalamiento.

El siguiente conjunto de siete momentos pueden ser derivados del segundo y tercer momento; y son invariantes ante escalamiento, traslación y rotación:

$$\phi_1 = \eta_{20} + \eta_{02}$$

$$\phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta^2_{11}$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{30})^2$$

$$\phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$

$$\phi_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})\left[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2\right]$$

$$+(3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})\left[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\right]$$

$$\phi_6 = (\eta_{20} - \eta_{02})\left[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\right] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})$$

$$\phi_7 = (3\eta_{21} - \eta_{30})(\eta_{30} + \eta_{12})\left[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} - \eta_{03})^2\right] +$$

$$(3\eta_{21} - \eta_{30})(\eta_{21} - \eta_{03})\left[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} - \eta_{03})^2\right]$$

⇒ Centro de Masa:

$$\bar{n} = 1/N \sum_{(m,n)\in R} mi \qquad \bar{m} = 1/N \sum_{(m,n)\in R} ni$$

donde R es la región y N la cantidad de pixels.
El momento central de orden (p,q) se transforma en:

$$\mu_{pq} = \sum_{(m,n)\in R} (m-\bar{m})^p (n-\bar{n})^q$$

⇒ Excentricidad:

$$\varepsilon \underset{=}{\Delta} \left(\mu_{2,0} - \mu_{0,2}\right)^2 + 4\mu_{1,1} / \text{área}$$

♦ Características de la forma:

⇒ Shape Number (Número de figura)

Una característica que se puede extraer a un borde para la clasificación de su forma es el *número de figura*, dado que es invariante a distorsiones tales como: escalamiento, traslación y rotación. El número de figura de orden n está basado en un código cadena 4-direccional y se lo define como la primer diferencia de menor magnitud . El orden n en el número de figura está referido a la cantidad de dígitos que tendrá la representación.

Para esto tomamos el eje mayor y el eje menor y construimos el rectángulo base del borde donde el eje mayor es la recta que une los dos puntos más distantes entre sí; mientras que el eje menor es perpendicular a este y es lo suficientemente largo como para que junto al eje mayor formen una caja que contenga al borde. En la práctica, para un deseado orden, encontramos el rectángulo de orden n cuya excentricidad (promedio del eje mayor con el eje menor) mejor se aproxime a la del rectángulo básico y utilizamos este nuevo rectángulo para establecer el tamaño de la grilla.

Aunque la primer diferencia es computada tratando al código cadena como una secuencia circular resultando así invariante a la rotación, en general el borde codificado depende de la orientación de la grilla con la cual se muestrea.

Una forma de evitar esto es normalizar la grilla, por ejemplo ubicándola siempre a lo largo del eje mayor.

Observemos el ejemplo de la figura II 3.7:

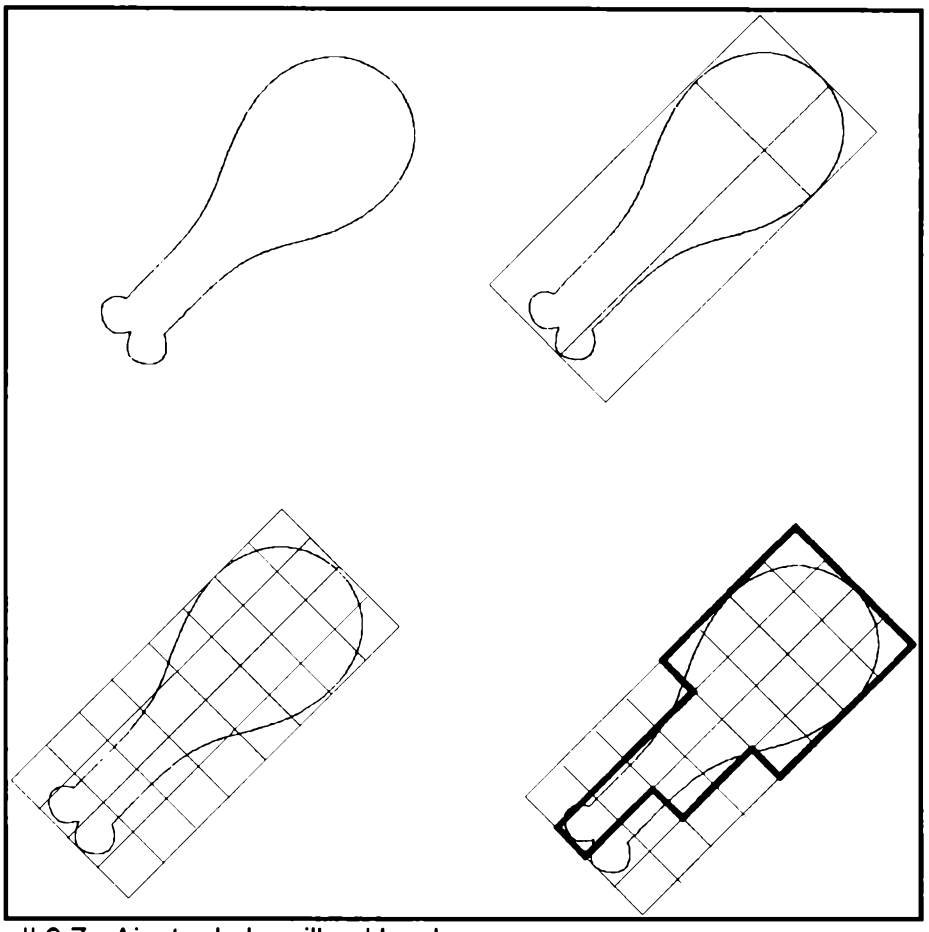


Figura II.3.7 - Ajuste de la grilla al borde.

A medida que se incrementa el número de orden veremos que el patrón que forma el código cadena se va pareciendo cada vez con mayor exactitud al borde que representa.

⇒ Rúbrica:

Una rúbrica es una representación funcional de un límite y puede ser generada de varias maneras. Una de las más simples es dibujar la distancia desde el centro hasta el borde como una función del ángulo. Sin tener en cuenta cómo la rúbrica es generada, la idea básica es reducir la representación del límite a una función 1-D, la cual es, presumiblemente, más fácil de describir que el límite original en 2-D .

Las rúbricas generadas por la interpretación recién descripta son invariantes a la traslación, pero ellas dependen de la rotación y escalamiento. La normalización con respecto a la rotación puede realizarse encontrando una manera de elegir el mismo punto de partida para generar la rúbrica, sin tener en cuenta la orientación de la figura. Una forma de hacer esto es elegir el punto de partida como el punto más alejado del centro, si ocurre que este es único e independiente de la rotación. Otra manera es elegir el punto sobre el eje mayor más alejado del centro.

Basándose en las suposiciones de uniformidad en el escalamiento con respecto a ambos ejes , y que el muestreo es tomado a intervalos iguales de q, los cambios en el tamaño de una figura resultan ser cambios en los valores de amplitud de la

correspondiente rúbrica. Una manera simple de normalizar este resultado es escalar todas las funciones de forma que siempre se expandan en el mismo rango de valores ,digamos [0,1]. La principal ventaja de este método es la simplicidad, pero tiene la seria desventaja que el escalamiento de la función entera depende de solamente dos valores: el mínimo y el máximo. Si las figuras son ruidosas, esta dependencia puede ser una fuente de error de un objeto a otro.

Una interpretación más robusta, pero también más intensiva computacionalmente es dividir cada muestra por la varianza de la rúbrica, suponiendo que la varianza no es cero o tan pequeña que cree dificultades computacionales. El uso de la varianza produce un factor de escalamiento variable que es inversamente proporcional a los cambios en tamaño.

Cualquiera que sea el método utilizado, la idea básica es remover la dependencia en el tamaño pero preservar la forma fundamental de la función.

Distancia versus ángulo no es la única forma de generar una rúbrica. Por ejemplo el límite puede ser seguido y el ángulo entre una línea tangente al límite y una línea de referencia, dibujado como una función de posición a lo largo del límite. La rúbrica resultante, aunque bastante diferente de la curva $r(\theta)$, traería información acerca de características básicas de la figura. Por ejemplo los segmentos horizontales en la curva se corresponderían a líneas rectas a lo largo del límite, dado que el ángulo tangente sería constante allí. Una variación de este método es usar la llamada *función de densidad de pendientes* como una rúbrica. Esta función es simplemente un histograma de los valores de los ángulos tangentes. Como un histograma es una medida de concentración de valores, la función de densidad de pendientes responde fuertemente a las secciones del límite con ángulos tangentes constantes y ángulos que varían rápidamente reflejando esquinas u otras inflexiones puntiagudas.

II.4. Reconocimiento y clasificación.

Normalmente, varias características simbólicas tales como líneas y áreas son lo primero que se determina en la imagen. Estas entonces se comparan con características similares asociadas con modelos almacenados para encontrar coincidencias al contemplar objetos específicos.

El análisis de la imagen es un proceso de descubrir, identificar y entender los patrones que son relevantes para el desarrollo de una determinada tarea basada en imágenes. Una de las principales metas del análisis de imágenes por computadora es dotarla de capacidades similares, en cierto sentido, a las capacidades humanas de entendimiento. Por ejemplo: en un sistema para leer imágenes documentos tipeados en forma automática, los patrones de interés son los caracteres alfanuméricos, y el objetivo es alcanzar una exactitud de reconocimiento de caracteres que sea lo más cercana posible a la capacidad humana para tal tarea.

Así un sistema automatizado de análisis de imágenes debería ser capaz de exhibir cierto grado de inteligencia. El concepto de inteligencia es algo vago, en particular con referencia a una máquina. Sin embargo no es difícil conceptualizar varios tipos de comportamiento generalmente asociados a inteligencia.

Algunas características serían:

1) La habilidad de extraer información pertinente de un fondo de detalles irrelevantes.

2) La capacidad de aprender de ejemplos y generalizar ese conocimiento de manera que pueda ser aplicado a diferentes circunstancias.
3) La habilidad de hacer inferencias de información incompleta.

Sistemas de análisis de imágenes con estas características pueden ser diseñados e implementados para medios ambientes operacionales limitados. Sin embargo aún no se sabe a ciencia cierta como dotar a estos sistemas con un nivel de performance tan cercano a las capacidades humanas en cuanto a sus funciones de análisis de imágenes. En realidad la mayoría de los sistemas actuales están basados en formulaciones heurísticas trabajadas para resolver problemas específicos; como ser: lectura de documentos apropiadamente formateados.

Los problemas de este estilo son altamente especializados y tienen poca extensión, o nada, a otros problemas. Así es que las actuales teorías y limitaciones de implementación en el campo del análisis de imágenes implican soluciones que son altamente dependientes del problema.

II.4.1. Métodos de Reconocimiento.

En general antes de embarcarnos en la clasificación de objetos dentro de una imagen, debemos tener algún conocimiento acerca de qué estamos buscando.

En cualquiera de los casos en los cuales esté basada nuestra búsqueda, el proceso de clasificación involucra comparar un conjunto de características o descriptores de un objeto con algún criterio preestablecido. Este proceso se realiza en tres pasos: primero, determinamos las características que deseamos usar para clasificar el objeto; segundo, acordamos tolerancias, estableciendo cuan cerca deben estar las medidas de las características para poder realizar el apareo; y tercero, creamos grupos de clasificación, categorías, o clases, a las cuales un objeto será asignado dependiendo de la comparación de sus características.

Generalmente se seleccionan aquellas medidas que son computacionalmente más fáciles de derivar, y más tarde de comparar con el objeto que está siendo analizado.

Hay diversos métodos de reconocimiento. Básicamente podemos agruparlos en

- Métodos de decisión teórica.
- Métodos estructurales.

Los *métodos de decisión teórica* se basan en representar patrones en forma de vectores y luego buscar aproximaciones para agrupar y asignar vectores de patrones en diferentes clases de patrones. Algunos de los métodos usados son los clasificadores de mínima distancia y apareo de plantillas.

En el *reconocimiento estructural* los patrones son representados en forma simbólica (tales como strings o árboles), y los métodos están basados en apareo simbólico o en modelos que los tratan como sentencias de un lenguaje artificial.

II.4.1.1 Métodos de Decisión Teórica

Los métodos de decisión teórica están basados en el uso de *funciones de decisión.* Sea $x = (x_1,x_2,..,x_n)^T$ la representación de un vector patrón n-dimensional. Para M clases de patrones w_1, w_2,..,w_n el problema básico en estos métodos es encontrar M funciones de decisión $d_1(x)$, $d_2(x)$,..,$d_n(x)$ con la propiedad de que si un patrón x pertenece a la clase w_i, entonces

$$d_i(x) > d_j(x) \quad j = 1,2,..,M;\ j<>i$$

En otras palabras, un patrón desconocido x pertenece a la clase i-ésima, si substituyendo x en todas las funciones de decisión, di(x) produce el valor numérico más grande.

El *límite de decisión* que separa la clase w_i de w_j está dado por los valores de x para los cuales di(x) = dj(x) o, equivalentemente, para los valores de x que satisfacen la ecuación

$$di(x) - dj(x) = 0$$

En la práctica el límite de decisión entre dos clases se calcula con la función diq(x) = di(x) - dj(x) = 0. Así dij(x) > 0 para patrones de la clase wi, y dij(x) < 0 para patrones de la clase wj.

- Clasificador de mínima distancia

Supongamos que cada clase de patrón es representada por un vector prototipo o vector promedio:

$$m_i = \frac{1}{N} \sum_{x \in w_j} x \qquad j = 1,2,..,M$$

donde N es el número de vectores patrón de la clase wj y la sumatoria es calculada sobre esos vectores. Una manera de determinar la pertenencia a una clase de un vector patrón desconocido x es asignar este a la clase de su prototipo más cercano. Usando la distancia Euclidiana para determinar su cercanía se reduce el problema de computar las medidas de distancia:

$$Dj(x) = \| x - mj \| \qquad j = 1,2,..,M$$

donde $\|a\| = (a^T a)^{1/2}$. Luego asignaremos x a la clase wj si Dj(x) es la menor distancia. No es difícil demostrar que es equivalente a evaluar las funciones:

$$d_j(x) = x^T m_j - \frac{1}{2} m_j^T m_j \qquad j = 1,2,..,M$$

y asignar x a la clase w_i si dj(x) produce el valor numérico más grande.

Por lo anterior se deduce el límite decisión como:

dij(x) = di(x) - dj(x)
= xT (mi - mj) - ½ (mi - mj)T (mi - mj) = 0

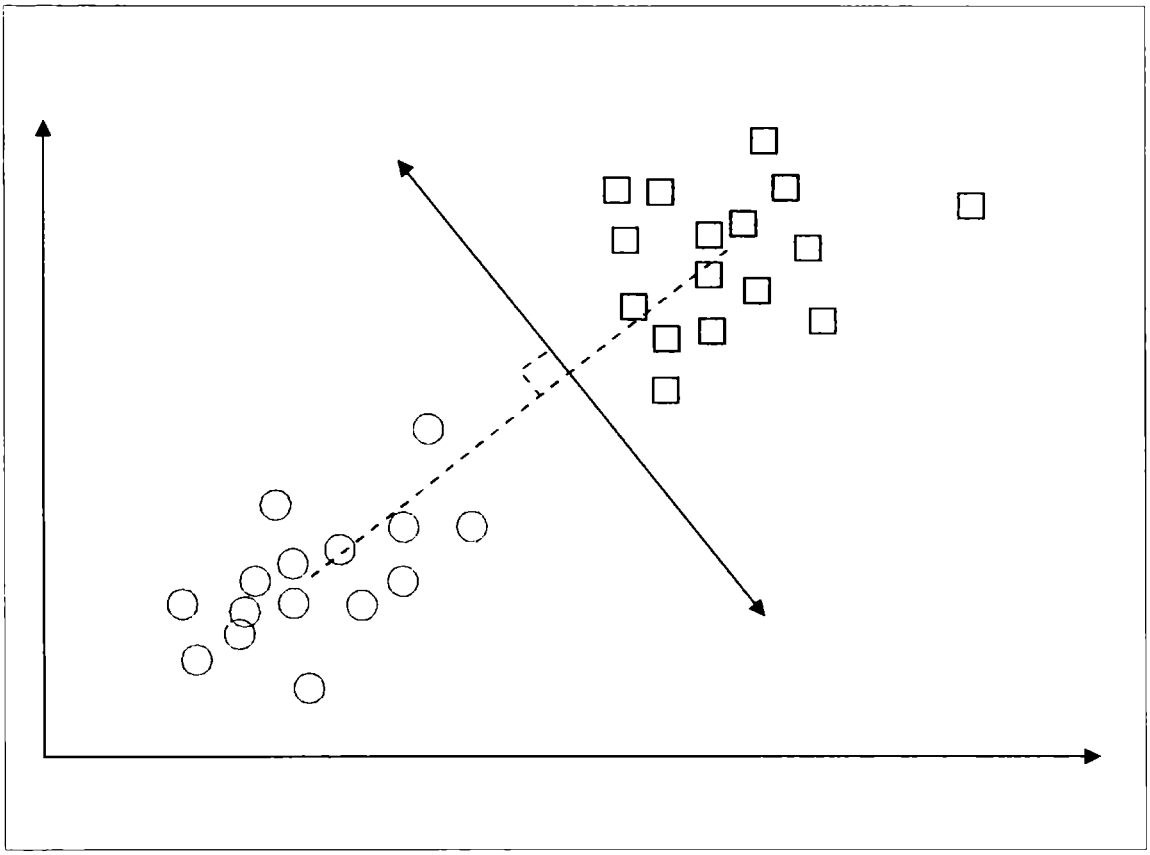


Figura II.4.1 - Ejemplo de un Clasificador de Mínima Distancia.

En la figura II.4.1 se observa la distribución de los vectores para distintos objetos, agrupándose en dos posibles clases.

- Apareo de Plantillas (Template Matching):

Unas de las herramientas más importantes de la detección de un objeto en una imagen es el apareo de plantillas, en el cual una réplica de un objeto de interés es comparada con todos los objetos desconocidos en la imagen. Si el apareo de plantilla entre

un objeto desconocido y la plantilla es casi igual, el objeto desconocido es llamado como el objeto plantilla.

Como un ejemplo simple de este proceso consideremos la figura II.4.2.

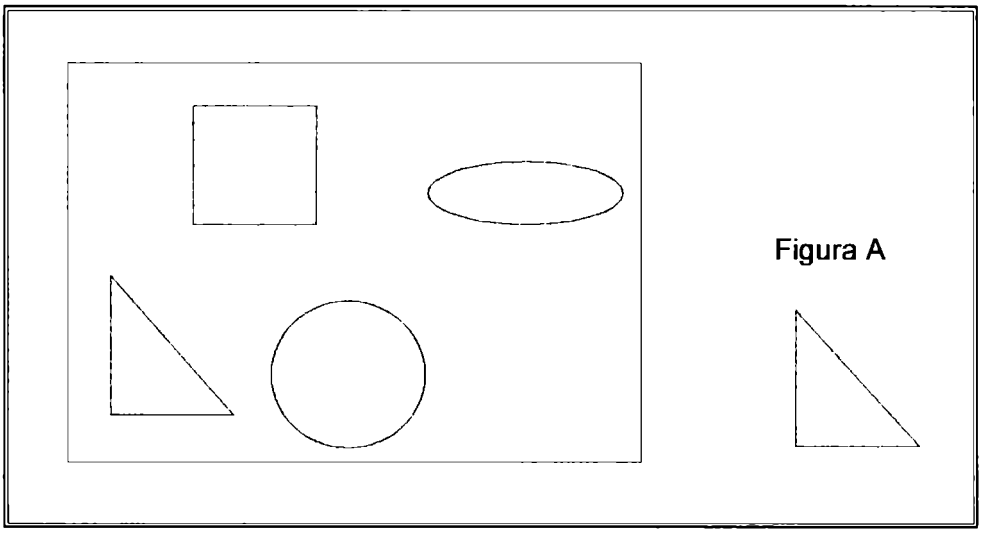


Figura II.4.2. Imagen original y plantilla a buscar.

En este ejemplo el objetivo es detectar la presencia y ubicación de triángulos en la imagen. La figura A contiene una simple plantilla para la localización de triángulos equiláteros. El largo de los lados es elegido como compromiso entre la ubicación exacta y el tamaño invariante de la plantilla. En la operación la plantilla es secuencialmente deslizada sobre la imagen y la región común entre la plantilla y la imagen es comparada por similitud.

El apareo de plantilla raramente es exacto por:

- Ruido en la imagen.
- Efectos de escalamiento, rotación, u otras distorsiones...
- Incertidumbre a priori de como es el objeto a ser detectado.

Es obvio que por la metodología usada por este algoritmo sólo un patrón idéntico al de la plantilla, será reconocido en forma exitosa.

II.4.1.2 Métodos Estructurales

- Clases de Patrones

Una *clase* de patrón es una familia de patrones que comparten algunas características comunes.

Los patrones pueden ser generados de numerosas formas. La clave está en seleccionar la o las medidas sobre las cuales basarse, pues será de influencia directa sobre la performance del sistema de análisis de la imagen.

Estas técnicas producen clases de patrones caracterizados por información cuantitativa.

La figura II.3.2 muestra un ejemplo de generación de un vector de patrón. En este caso elegimos describir cada objeto por su rúbrica, donde obtenemos una función 1-D. Muestreando esta función a intervalos de q; es decir $q_1,q_2,...,q_n$; podemos obtener diversos vectores patrones de la forma x1 = r(q_1); x2 = r(q_2); ...; Xn = r(qn).

- Apareo de Número de Figura

Entre los métodos de reconocimientos podemos ver los que trabajan con las relaciones estructurales inherentes a la forma del patrón.

Por ejemplo se puede realizar el apareo de los números de figura. Podemos utilizar un concepto similar a la distancia mínima, donde el *grado de similitud k,* entre dos bordes de figuras, A y B, es definido como el mayor orden para el cual dos números de figura todavía coinciden.

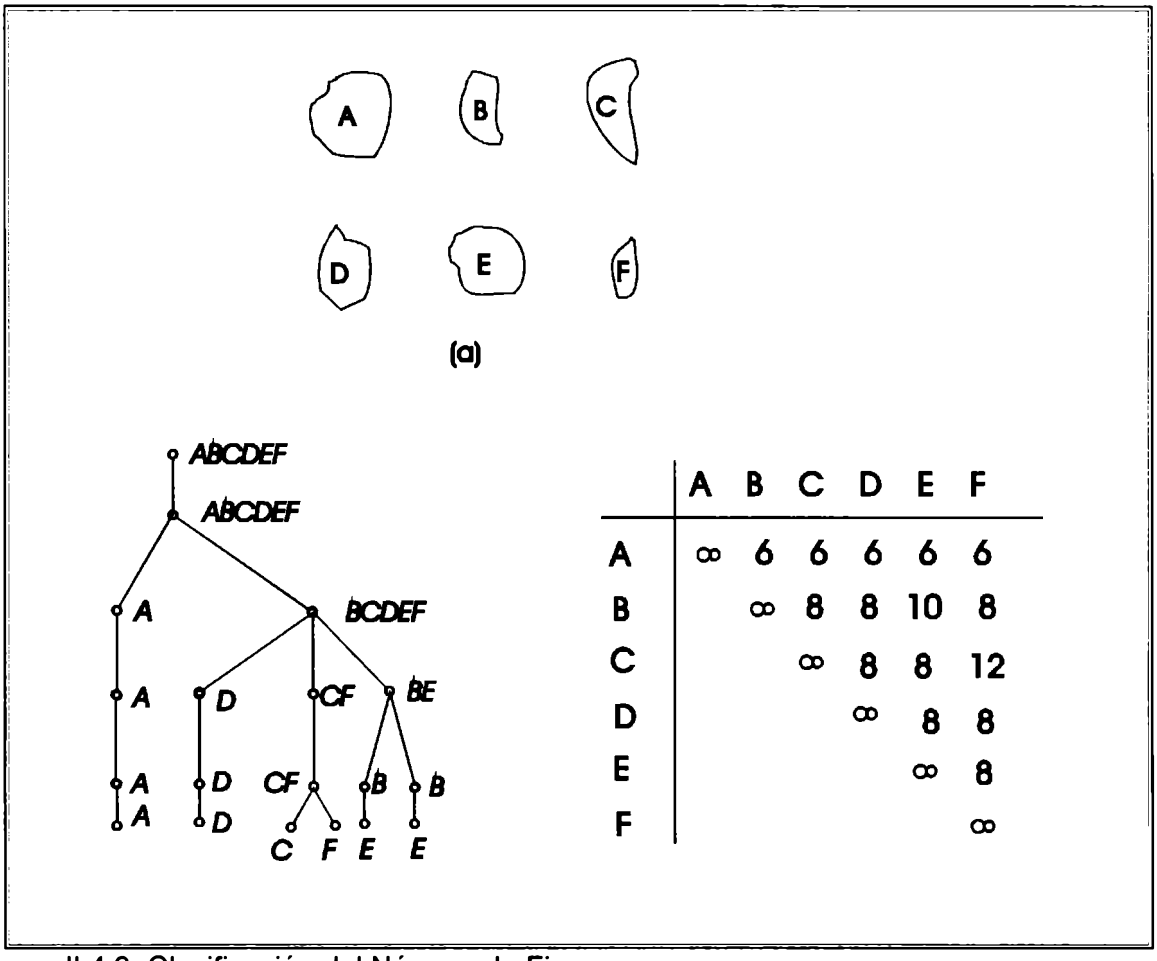
La *distancia* entre dos figuras A y B se define como la inversa del su grado de similitud:

$$D(A,B) = 1/k$$

Esta distancia satisface las siguientes propiedades:

$D(A,B) >= 0$
$D(A,B) = 0$ si solo so $A = B$
$D(A,C) <= máx[D(A,B), D(A,B)]$

Supongamos tener una forma F y queremos encontrar su apareo más cercano a un conjunto de otras formas (A,B,C,D, y E), como se ve en la figura II.4.3 (a). La búsqueda puede ser visualizada con un árbol de similitud (figura II.4.3 (b)). La raíz del árbol se corresponde al menor nivel de similitud posible, el cual en el ejemplo es 4. Las formas son idénticas en el grado 8, a excepción de la forma A, cuya similitud con respecto a las otra figuras es 6. Continuando hacia abajo en el árbol encontramos que las figuras C y F aparean en forma única, teniendo el mayor grado de similitud que cualquiera de cualquier otra forma. La misma información puede ser resumida en una *matriz de similitud,* como se muestra en la figura II.4.3(c).



	A	B	C	D	E	F
A	∞	6	6	6	6	6
B		∞	8	8	10	8
C			∞	8	8	12
D				∞	8	8
E					∞	8
F						∞

Figura II.4.3. Clasificación del Número de Figura

- Apareo de Strings

Supongamos que dos límites de región, A y B están codificados como strings (código cadena, número de figura, etc), denotados a1,a2,...an y b1,b2,...,bn respectivamente. Sea M el número de apareos entre dos strings, donde hay un apareo en la k-ésima posición si $a_k = b_k$. El número de símbolos que no aparean es :

$$Q = \max(|A|,|B|) - M$$

donde |A| es la longitud (número de símbolos) en el string A.

Una simple medida de similitud entre A y B es el promedio:

$$R = M/Q = M/\max(|A|,|B|) - M$$

Dado que el apareo se hace símbolo a símbolo el punto de comienzo en cada límite es importante. Cualquier método que normalice al string mejorará el tiempo computacional

dado que de lo contrario, sin la normalización, el algoritmo consistiría en comenzar en un punto arbitrario en cada string, rotar uno de los strings y calcular R para cada rotación.

Capítulo III
Algoritmos Seleccionados

III.1 Consideraciones iniciales.

En este capítulo detallamos algunos de los métodos, tanto de representación como de descripción, que desde nuestro punto de vista son los más convenientes para el tipo de reconocimiento que hemos planteado.

Para seleccionar los métodos tuvimos en cuenta que los bordes a clasificar pueden tener algunas distorsiones que no deberían estar reflejadas en su descripción. Dada una figura cualquiera esta puede presentarse rotada, escalada o trasladada.

En la figura III.1 se pueden ver estas transformaciones:

(a) Patrón original.
(b) Patrón rotado.
(c) Patrón escalado.
(d) Patrón trasladado.

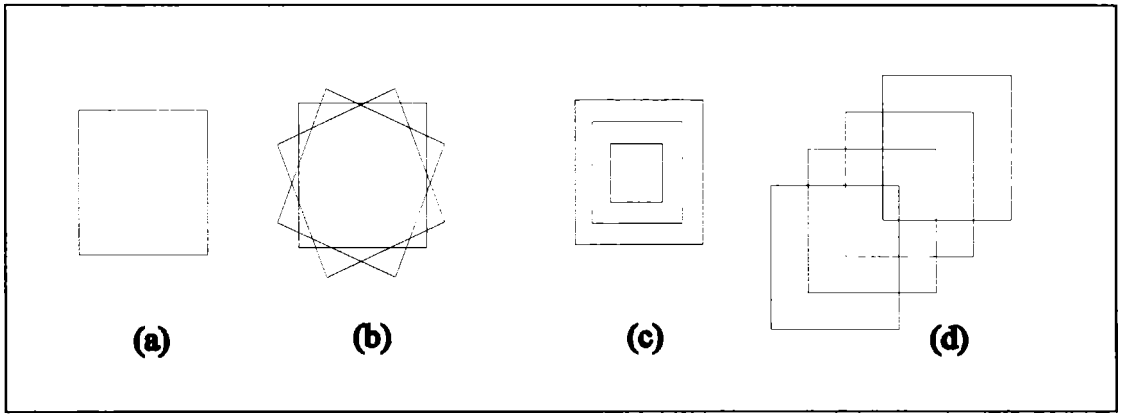


Figura III.1. Algunas distorsiones de un patrón.

Es obvio que no tiene mucha utilidad una característica de un patrón que varíe ante estas distorsiones.

Por ejemplo: supongamos tener un bitmap conteniendo la figura de un cuadrado de borde negro sobre un fondo blanco y que nuestro sistema de reconocimiento de patrones tiene como descriptor de "cuadrado" las siguientes características:

- Cantidad de lados: 4.
- Perímetro: 10.
- Excentricidad (relación entre el eje mayor y el eje menor): 1.

Dado que el cuadrado de nuestra imagen tiene un perímetro *x* distinto de 10, nuestro sistema no lo reconocerá como tal. De aquí deducimos que la característica perímetro no es un buen descriptor para este tipo de reconocimiento.

Se puede decir que el cuadrado de la imagen sufre una distorsión con respecto al patrón base.

Problemas como estos son los que nos obligan a hacer una buena selección de las características que usaremos para construir los descriptores de los patrones. Dichas características deben ser invariantes a la traslación, rotación y escalamiento.

III.2 Los algoritmos.

Para la etapa de descripción tenemos las siguientes posibles características a extraer:

- características de amplitud.
- características del histograma.
- medidas de la forma.
- momentos.
- shape number.
- características sobre la rúbrica.

Las dos primeras se basan sobre el análisis de características en imágenes con niveles de grises, con lo cual quedan descalificadas para nuestro propósito.

En general todas las medidas de la forma que hemos estudiado se relacionan con el tamaño del borde a analizar, con lo cual obtenemos un descriptor que no es invariante al escalamiento y por consiguiente tampoco nos servirían.

Por otro lado el método de los momentos, shape number y características de la rúbrica parecen ser los más adecuados dado que en teoría son independientes a las distorsiones antes vistas.

Por lo anterior seleccionaremos, para cada algoritmo, la representación que mejor se adapte y así los evaluaremos tratando de expresar el tipo de complejidad en la implementación, los tiempos y el uso de memoria. Un análisis más preciso, ya sobre ejemplos probados lo veremos en el próximo capítulo.

III.2.1 Algoritmo del shape number

En este caso es necesario trabajar con una representación basada en el código cadena, dado que el método se genera a partir del mismo.

Los pasos a seguir serían los que ilustra la figura III.2.

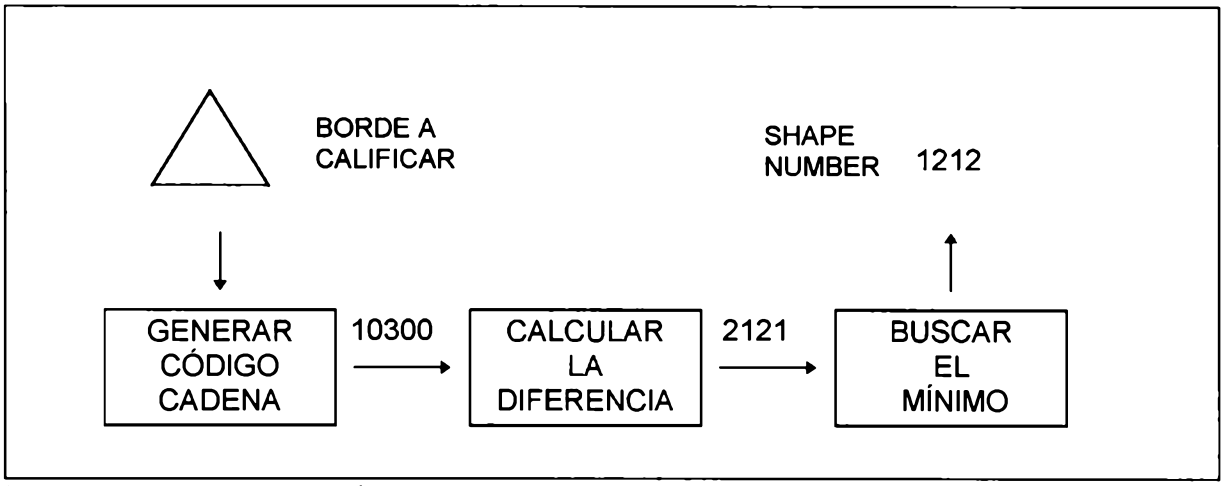


Figura III.2 - Pasos en el cálculo del *shape number*.

En forma más detallada tenemos el siguiente pseudocódigo:

- Obtener el código cadena:
  - Calcular eje mayor.
  - Calcular eje menor.
  - Generar el rectángulo base y la cuadrícula según el nº de orden.
  - Extraer los puntos de la cuadrícula que forman el código cadena.
- Obtener la diferencia.
- Obtener la mínima diferencia (shape number).

Evaluando los pasos a seguir observamos que la obtención del código cadena es la parte más compleja del algoritmo, por lo que consumirá mayor tiempo de computación; y la más delicada, dado que en definitiva el éxito del método se basa en ella.

Supongamos tener un borde del cual queremos obtener su nº de figura. Los pasos a seguir serían los siguientes.

Calculo del eje mayor

Lo primero a calcular será su eje mayor, para lo cual obtendremos las distancias entre todos los puntos del borde. Este procedimiento es altamente costoso en cuanto a tiempo se refiere, y es proporcional al tamaño o cantidad de puntos que forman el borde. Un problema que se tiene en este paso es la posibilidad de que algunos bordes pueden tener más de 1 eje mayor. De ser así se puede obtener cuadrículas con diferentes orientaciones con las cuales generaríamos códigos cadena imposibles de comparar. Recordemos que el borde codificado depende de la orientación de la grilla y que la normalización de ésta, ubicándola siempre a lo largo del eje mayor, más el cálculo de la diferencia, tratan de resolver el problema de la rotación.

Analicemos la figura III.3 donde en particular tenemos un borde con dos ejes mayores con los cuales se generan dos códigos cadenas donde uno es el reflejo del otro.

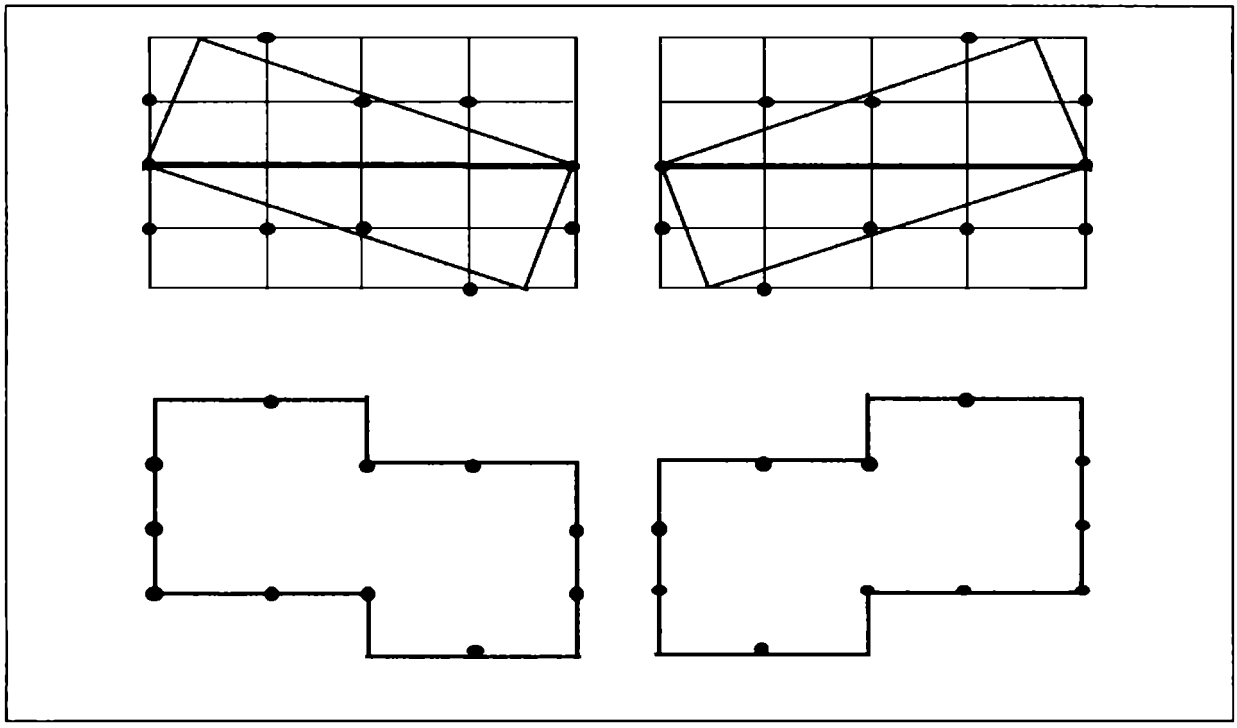


Figura III.3 Problemas en el shape number.

Calculo del eje menor

Una vez obtenido el eje mayor (figura III.4), el borde nos queda dividido en dos segmentos. Para la búsqueda del eje menor recorremos los segmentos y para cada uno hallamos el punto más lejano en forma perpendicular al eje mayor, para lo que implementamos un procedimiento que calcula la distancia entre un punto y una recta.

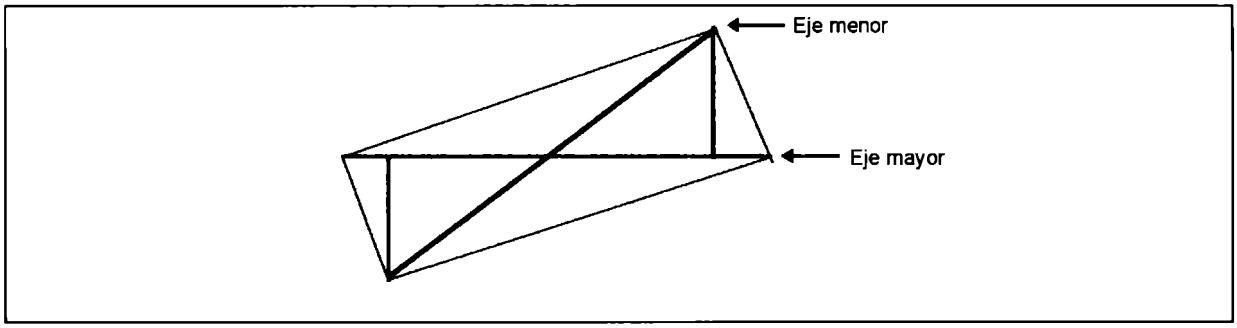


Figura III.4 - Calculo del eje menor.

Estos dos puntos serán los que definirán al eje menor, como se ven en la figura III.4.

Generación del rectángulo base y la cuadrícula según el número de orden

Ya con los dos ejes, y buscando algunas líneas paralelas, podremos encontrar el rectángulo base el cual encierra a la figura en su totalidad y forma el borde de nuestra retícula.

Hasta aquí la complejidad y los tiempos del algoritmo dependen del tamaño del borde, esto es, la cantidad de puntos que lo forman, pero de ahora en más también influirá el nº de orden del nº de figura que se quiera calcular. A mayor nº de orden la cuadrícula será más fina y contendrá más puntos de intersección. Además hay que tener en cuenta que estos puntos tienen coordenadas reales, lo cual hace que a medida que el nº de orden crezca, la utilización de la memoria se ve directamente afectada. En la implementación manejamos una matriz conteniendo tantas coordenadas reales como nº de orden sea el nº de figura a calcular.

Extracción de los puntos de la cuadrícula que forman el código cadena

En este punto tenemos principalmente dos estructuras. Un arreglo conteniendo los puntos del borde y una matriz conteniendo los puntos que representan la cuadrícula.

Paso seguido, para cada punto del borde, calculamos las distancias a todos los puntos de la cuadrícula, quedándonos con el punto más cercano. Así nos quedamos con aquellos puntos de la cuadrícula que en definitiva forman el código.

III.2.2 Método de los Momentos.

Este método plantea una serie de cálculos sobre las coordenadas de los puntos pertenecientes a una borde.

Pueden calcularse los momentos espaciales si se toma la figura como región o los momentos geométricos si tomamos solamente el borde. Estos últimos serán los que evaluaremos.

De algorítmica sencilla, el método reduce su performance al instante de evaluar el tiempo que consume, y esto debido a la complejidad de sus cálculos y a los varios niveles de anidamiento que poseen los mismos. Inclusive la acumulación de resultados parciales a lo largo del cálculo de los momentos invariantes, hace que el recurso memoria cobre importancia.

III.2.3 Rúbrica.

Una de las representaciones más sencillas de calcular y normalizar para que resulte invariante a las distorsiones, por nosotros antes planteadas, es la rúbrica de un borde.

La rúbrica normalizada como explicamos anteriormente puede basarse en función de la distancia del centro al borde, o en función del ángulo de la tangente en cada punto.

Considerándola como una función 1-D se puede extraer medidas tales como: cantidad de puntos, cantidad de máximos, cantidad de mínimos locales y absolutos, varianza, media, área, etc., los cuales pueden ser agrupados en una vector y tomados como una descripción del borde.

El algoritmo de este método plantea el cálculo de la rúbrica del borde y la extracción de características a la misma.

Cálculo de la rúbrica

Evaluado los dos métodos posibles para el cómputo de la rúbrica hemos optado por el que se basa en la distancia del centro a los puntos del borde, dado que los cálculos a primera vista, parecen ser más sencillos y por lo tanto emplear menos tiempo de computación.

El desarrollo de este algoritmo presenta el siguiente pseudocódigo:

- El primer paso es ubicar el centro de la figura, el cual se lo define como la intersección de las diagonales del rectángulo base.

  Para calcular el rectángulo base debemos encontrar el eje mayor y el eje menor.

  El primero lo obtendremos calculando las distancias entre todos los puntos del borde. Este procedimiento es altamente costoso en cuanto a tiempo se refiere, y es proporcional al tamaño o cantidad de puntos que forman el borde. Un problema que se tiene en este paso es la posibilidad de que algunos bordes pueden tener más de un eje mayor. De ser así se puede obtener cuadrículas con diferentes orientaciones con las cuales generaríamos códigos cadena imposibles de comparar.

  Una vez obtenido el eje mayor, el borde nos queda dividido en dos segmentos. Así la búsqueda del eje menor la realizamos recorriendo estos segmentos y para cada uno hallamos el punto más lejano perpendicular al eje mayor. Para esto implementamos un procedimiento que calcula la distancia entre un punto y una recta.

  Ya con los dos ejes y buscando algunas líneas paralelas podremos encontrar el rectángulo base el cual encierra a la figura en su totalidad.

- Al momento de calcular la rúbrica dos cosas importantes a tener en cuenta son el punto de comienzo y en que sentido se recorre el borde, de lo contrario dos rúbricas para un mismo borde podrían diferir. Una solución a esto sería elegir, por ejemplo, como punto de comienzo a uno de los puntos del eje mayor y seguir el borde siempre en sentido antihorario.

  La teoría del método de la rúbrica propone sacar la distancia del centro a todos los puntos del borde, aunque llevado a la práctica, es conveniente tomar un subconjunto de los mismos. En nuestro caso definimos un ángulo α y tomamos un punto del borde cada vez que la pendiente de la recta que une el centro al punto varía α grados de la pendiente que unía al centro con el punto anteriormente tomado.

  Para la elección del ángulo α debemos considerar que un mayor ángulo generará una rúbrica de pocos puntos que no reflejará las características básicas del borde, mientras que un menor ángulo producirá muchos puntos que complicarán los cálculos y redundarán en información acerca de las características del borde.

  Una vez elegido el ángulo, recorremos el borde y obtenemos un vector de distancias que representa la función rúbrica, a la que nosotros hemos elegido extraerle el promedio, el área y la varianza, características con la cual formaremos un descriptor del borde.

  Teóricamente el método no implica una gran complejidad algorítmica, y en cuanto al uso de la memoria no parece ser este un punto preocupante.

Para la evaluación de los resultados, dadas las características del método de la rúbrica, hemos optado por usar el método llamado clasificador de mínima distancia.

Capítulo IV
Resultados Obtenidos

IV.1 Consideraciones iniciales.

Hemos realizado distintas pruebas sobre un conjunto de imágenes evaluando los siguientes puntos:

- Complejidad.
- Tiempo de Ejecución.
- Eficiencia.

Cada imagen contiene un solo borde y los mismos están afinados a un pixel con el objetivo de enfocarnos en los problemas de representación y clasificación.

Las figuras son bordes cerrados simples tales como círculos, cuadrados, triángulos y otras figuras no geométricas, las cuales rotamos y escalamos para testear la eficiencia del algoritmo.

Los algoritmos los hicimos en Turbo Pascal 7.0 para D.O.S. y fueron corridos para su testeo sobre una PC Pentium de 100 Mhz. con 8 Mb. de memoria RAM.

IV.2 Shape Number

Para la evaluación de este método la idea era utilizar lo que se denomina apareo de número de figura, pero nos topamos con un inconveniente, la digitalización de la imagen. Esta no sólo nos impidió usar el método de clasificación, sino que también tuvo incidencia en los resultados parciales al extraer los números de figura de los bordes.

Teníamos un concepto básico de la teoría el cual cambió al ser llevado a la práctica:

A un orden bajo los números de figura de distintos bordes eran iguales, y a medida que el orden crecía se iban diferenciando, mientras que para bordes iguales los números de figura se mantenían iguales hasta órdenes grandes. Basado en este concepto el método del apareo del número de figura era válido, dado que este buscaba hasta que orden k los números de figura de dos bordes se mantenían iguales. El método de clasificación concluía que si k era lo suficientemente grande los bordes eran iguales, de lo contrario no.

La digitalización de los bordes trajo como consecuencia que bordes iguales, sólo diferenciados por alguna distorsión (traslación, rotación, escalamiento), dieran códigos cadena levemente distintos y por consiguiente números de figura distintos. Para casos como este, de dos bordes iguales, la digitalización trajo los siguientes problemas:

- Códigos cadenas semejantes:

El código cadena para un borde se hallaba recorriendo el mismo y tomando los puntos de la cuadrícula más cercanos a él. De esta manera el borde de la figura IV.1 podría generar como código cadena cualquiera de estas opciones:

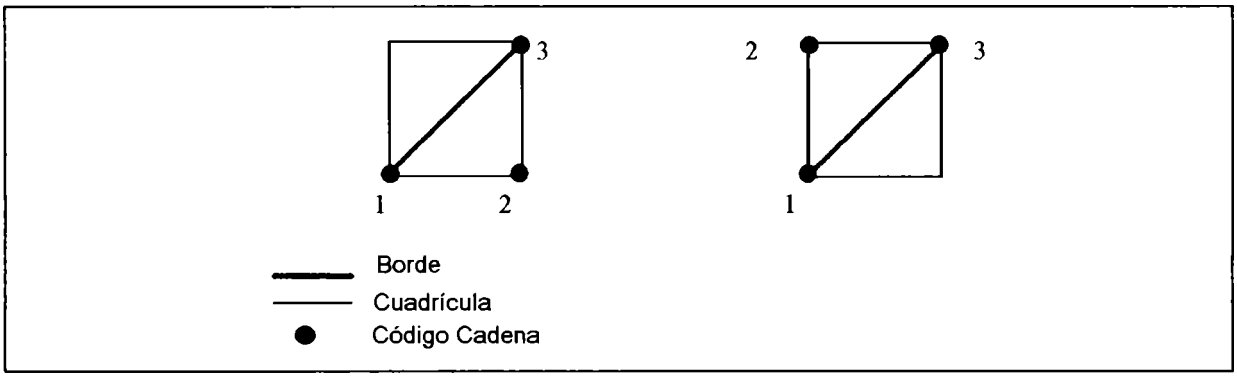


Figura IV.1 - Problema de códigos cadena semejantes.

Esto se debe a que la digitalización hace que el borde sufra pequeñas distorsiones, las cuales afectan al código cadena y al número de figura, lo cual nos conducirá a tener dos números de figura distintos para el mismo borde.

- Códigos de distinto orden:

A veces es posible que el número de figura de un borde a orden *k* dé un número de más de *k* dígitos. Esto depende de la forma del borde y del tamaño de la cuadrícula. Supongamos tener dos bordes, A y B. El borde A es un cuadrado y el borde B es el mismo cuadrado pero rotado 45°. En teoría si el cuadrado A tiene un número de figura de k + 1 dígitos a orden *k,* el cuadrado B también debería tenerlo. En la práctica, y por consecuencia de la digitalización, esto a veces no sucede, y por lo tanto nos encontramos con bordes iguales que tienen códigos cadenas de distintos órdenes los cuales no son comparables.

Dado estos dos problemas el método *apareo de número de figura* no tenía sentido dado que dos bordes iguales podrían tener códigos cadena distintos a orden bajo. Por lo tanto realizamos la evaluación calculando el número de figura para dos bordes desde el orden 8 hasta el orden 50, y contando el porcentaje de éxitos. Los resultados fueron los siguientes:

Bordes comparados: CIRC2 CIRC3	
ORDEN	RESULTADO
8	iguales
10	iguales
12	iguales
14	iguales
16	iguales
18	iguales
20	iguales
22	iguales
24	iguales
26	distintos
28	iguales
30	iguales
32	iguales
34	iguales
36	distintos
38	iguales
40	iguales
42	distintos
44	iguales
46	ordenes distintos
48	ordenes distintos
50	ordenes distintos

Bordes comparados: PLANO_A1 PLANO_A2	
ORDEN	RESULTADO
8	iguales
10	iguales
12	iguales
14	iguales
16	iguales
18	iguales
20	iguales
22	iguales
24	iguales
26	iguales
28	iguales
30	iguales
32	iguales
34	iguales
36	iguales
38	iguales
40	iguales
42	iguales
44	iguales
46	iguales
48	iguales
50	iguales

RESUMEN	
IGUAL	16
DISTINTOS	3
DISTINTO ORDEN	3
SIMILITUD	84%

RESUMEN	
IGUAL	22
DISTINTOS	0
DISTINTO ORDEN	0
SIMILITUD	100%

Nota: para el cálculo de la *similitud* no se cuentan los casos de distinto orden.

Bordes comparados: ROMBO7 ROMBO8	
ORDEN	RESULTADO
8	distintos
10	distintos
12	distintos
14	distintos
16	distintos
18	distintos
20	distintos
22	distintos
24	distintos
26	distintos
28	distintos
30	distintos
32	distintos
34	distintos
36	distintos
38	distintos
40	distintos
42	distintos
44	distintos
46	distintos
48	distintos
50	distintos

Bordes comparados: CUA6 CUA2	
ORDEN	RESULTADO
8	distintas
10	iguales
12	iguales
14	iguales
16	iguales
18	iguales
20	iguales
22	iguales
24	distintas
26	distintos
28	iguales
30	iguales
32	iguales
34	distintas
36	iguales
38	iguales
40	distintas
42	distintos
44	iguales
46	iguales
48	iguales
50	distintas

RESUMEN	
IGUAL	0
DISTINTOS	22
DISTINTO ORDEN	0
SIMILITUD	0%

RESUMEN	
IGUAL	15
DISTINTOS	7
DISTINTO ORDEN	0
SIMILITUD	68%

Nota: para el cálculo de la *similitud* no se cuentan los casos de distinto orden.

Bordes comparados: TRIANG2 TRIANG1	
ORDEN	RESULTADO
8	iguales
10	iguales
12	distintas
14	iguales
16	iguales
18	distintas
20	distintas
22	iguales
24	distintas
26	distintas
28	distintas
30	distintas
32	distintas
34	distintas
36	distintas
38	distintas
40	distintas
42	distintas
44	distintas
46	distintas
48	distintas
50	distintas

Bordes comparados: PLANO_B1 PLANO_B3	
ORDEN	RESULTADO
8	iguales
10	ordenes distintos
12	ordenes distintos
14	iguales
16	iguales
18	iguales
20	distintas
22	ordenes distintos
24	distintas
26	ordenes distintos
28	ordenes distintos
30	distintas
32	distintas
34	ordenes distintos
36	distintos
38	distintas
40	ordenes distintos
42	ordenes distintos
44	ordenes distintos
46	ordenes distintos
48	ordenes distintos
50	ordenes distintos

RESUMEN	
IGUAL	5
DISTINTOS	17
DISTINTO ORDEN	0
SIMILITUD	23%

RESUMEN	
IGUAL	4
DISTINTOS	6
DISTINTO ORDEN	12
SIMILITUD	40%

Nota: para el cálculo de la *similitud* no se cuentan los casos de distinto orden.

Como se puede ver los resultados no dieron lo bueno que se esperaban aunque creemos que el algoritmo de este método puede ser perfeccionado dado que se tienen acotados estos dos problemas puntuales.

IV.3 Momentos.

El método de los momentos geométricos no nos dio los resultados tan auspiciosos que la teoría anunciaba. Aunque los cálculos a hacer eran relativamente sencillos, y teniendo en cuenta que aquí la digitalización del borde también juega en desmedro del método, los valores de los momentos entre bordes iguales, no nos dieron lo esperado.

Por otro lado, tuvimos problemas de desbordamiento de memoria al calcular algunos momentos parciales con algunos bordes.

IV.4 Rúbrica.

A continuación presentamos algunos resultados utilizando una primera aproximación del método de clasificación mínima distancia.

La idea básica del método de clasificación de mínima distancia es tomar el vector descripción de cada borde y tratarlo como un punto en el espacio n-dimensional, tomando como objetos de una misma clase aquellos que estén a una distancia relativamente cercana entre sí.

Por tal motivo en una primera utilización de este método calculamos la distancia entre todos los vectores descripción. Los resultados son los que se muestran en la Tabla 1.

Como se puede ver las distancias entre objetos del mismo tipo dieron relativamente pequeñas comparadas con las distancias entre objetos de distintos tipos. Por ejemplo en la primer tabla que comparamos el borde CIRC vemos que la distancia a los otros círculos es aproximadamente cero, mientras que la distancia las figuras de otro tipo es no menor de 3,574 aproximadamente.

Utilizando más adecuadamente el método de clasificación hemos "entrenado" al sistema de la siguiente manera:

Ingresamos un conjunto de imágenes conteniendo cuadrados, a los cuales el sistema extrajo el vector característica, para luego generar el vector promedio que representaría a la clase cuadrado. Luego hicimos lo mismo para el resto de las figuras. De esta manera el sistema quedó entrenado para diferenciar entre siete distintos objetos: cuadrados, triángulos equiláteros, triángulos isósceles, rombos, círculos y las figuras no geométricas que denominaremos PLANO_A y PLANO_B.

Luego presentamos al sistema distintas figuras no ingresadas en la etapa de entrenamiento, esperando que el sistema las clasifique en alguna de las figuras conocidas.

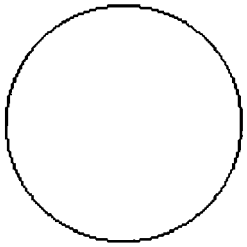
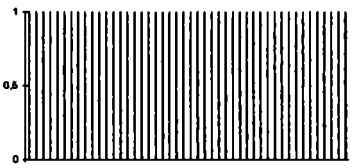
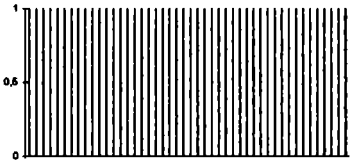
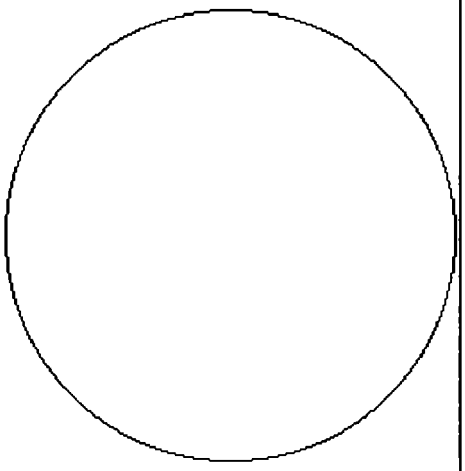
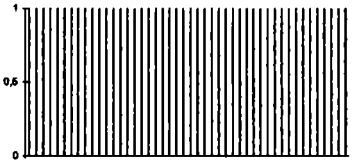
A continuación mostramos los resultados obtenidos:

	CIRC1	CIRC2	CIRC3	CIRC4	CUA1	CUA2	CUA3	CUA4	PLANO_A1	PLANO_A2	PLANO_B1
CIRC1											
CIRC2	1.45										
CIRC3	0.00	1.20									
CIRC4	0.00	1.45	0.00								
CUA1	3947.36	2174.87	3848.80	3947.36							
CUA2	3584.20	1944.11	3492.56	3584.20	0.00132						
CUA3	3574.57	1937.97	3483.11	3574.57	0.00144	0.00					
CUA4	3524.36	1906.31	3433.87	3524.36	0.00245	0.00	0.00				
PLANO_A1	31861.20	22830.21	31400.34	31861.20	1131.63	1259.99	1267.34	1286.22			
PLANO_A2	29687.75	21135.22	29250.34	29687.75	957.18	1070.36	1076.87	1093.56	0.00323		
PLANO_B1	67684.53	51232.12	66860.47	67684.53	4791.68	5214.39	5231.97	5294.41	78.53	101.68	
PLANO_B2	66257.49	50081.54	65446.90	66257.49	4610.82	5020.53	5037.74	5098.25	68.88	90.00	0.00
PLANO_B3	63964.65	48233.05	63175.69	63964.65	4321.17	4710.19	4726.74	4784.19	55.49	73.61	0.00380
PLANO_B4	62063.55	46745.22	61335.50	62106.35	4105.56	4477.88	4494.05	4548.99	44.79	60.55	0.0249
TRIANG1	149404.08	118758.45	147893.24	149404.07	18975.37	20220.93	20258.07	20442.85	2291.81	2541.83	249.06
TRIANG2	112438.51	878445.89	111219.02	112438.41	11664.99	12528.89	12555.74	12684.16	972.11	1099.51	51.75
TRIANG3	115876.01	907044.69	114628.71	115876.01	12300.51	13198.94	13226.91	13360.40	1059.23	1196.51	60.63
ROMBO1	346281.30	287783.90	343441.02	346281.13	71181.07	74518.86	74616.81	75108.21	16639.47	17842.71	4936.37
ROMBO2	338165.65	280722.90	335375.49	338165.65	68714.32	71965.42	72060.86	72539.58	15832.45	16990.06	4613.50
ROMBO3	335668.93	278553.07	332894.34	335668.93	67961.65	71185.76	71280.54	71755.28	15573.83	16717.52	4511.21
ROMBO4	314028.82	259766.51	311390.04	314028.82	61519.33	64510.68	64599.28	65039.86	13454.37	14478.69	3690.25
ISOCELE1	246666.62	201951.38	244486.25	246666.62	43422.17	45652.17	45735.28	46060.92	6702.05	7376.26	1439.99
ISOCELE2	229129.69	186798.08	227062.28	229129.69	38522.93	40576.33	40650.37	40950.95	5707.11	6293.78	1109.03
ISOCELE3	204481.65	165701.92	202583.78	204481.65	32221.83	34017.42	34082.59	34345.64	4268.90	4739.92	707.59
ISOCELE4	24357.31	199297.32	241436.63	243597.31	42556.52	44755.48	44837.05	45158.28	6518.02	7176.63	1377.47

	PLANO_B2	PLANO_B3	PLANO_B4	TRIANG1	TRIANG2	TRIANG3	ROMBO1	ROMBO2	ROMBO3	ROMBO4	ISOCELE1	ISOCELE2	ISOCELE3
CIRC1													
CIRC2													
CIRC3													
CIRC4													
CUA1													
CUA2													
CUA3													
CUA4													
PLANO_A1													
PLANO_A2													
PLANO_B1													
PLANO_B2													
PLANO_B3	0.00052												
PLANO_B4	0.0081	0.00053											
TRIANG1	274.32	317.03	364.69										
TRIANG2	59.54	73.07	89.51	3.48									
TRIANG3	69.44	84.70	103.00	2.31	0.000377								
ROMBO1	5165.29	5543.78	5916.75	446.91	1255.39	1143.52							
ROMBO2	4831.30	5191.53	5547.21	393.72	1138.80	1034.92	0.000422						
ROMBO3	4725.38	5079.72	5429.65	378.15	1103.87	1002.40	0.00126	0.00					
ROMBO4	3874.74	4180.56	4483.69	259.15	828.17	746.84	0.121	0.0402	0.0261				
ISOCELE1	1523.80	1668.84	1801.71	110.15	336.75	296.92	71.43	62.06	58.69	37.09			
ISOCELE2	1179.90	1302.13	1417.18	54.49	206.52	178.73	77.34	65.82	61.94	36.50	0.0762		
ISOCELE3	758.45	846.74	930.77	25.17	107.39	90.85	145.62	125.67	119.11	73.97	1.29	0.142	
ISOCELE4	1458.89	1599.72	1729.25	99.03	311.68	274.06	72.28	62.56	59.11	36.87	0.00	0.0376	0.947

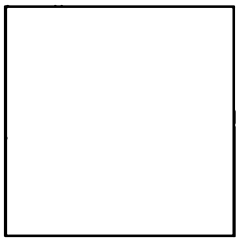
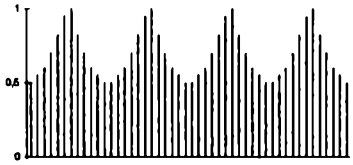
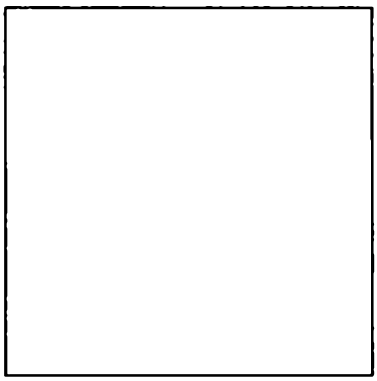
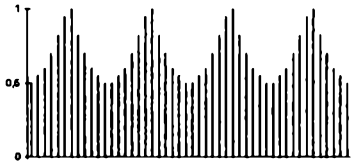
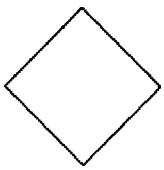
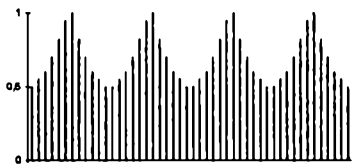
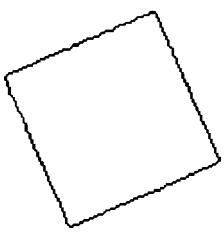
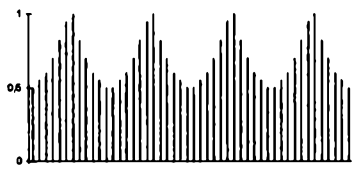
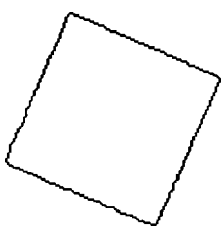
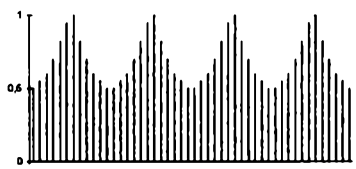
Tabla 1 - Distancias entre las figuras -

Bordes de entrenamiento de la clase Círculo:

Borde	Rúbrica	Vector descriptor
		(0.00034, 0.996, 39.86)
		(0.013, 0.969, 38.79)
		(0.00020, 0.994, 39.79)

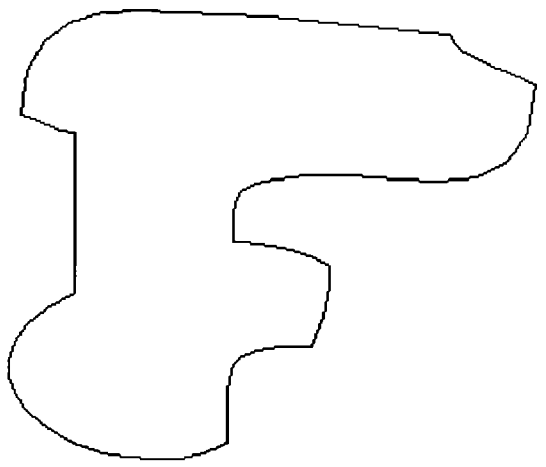
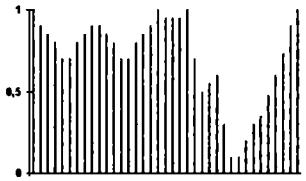
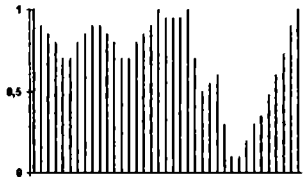
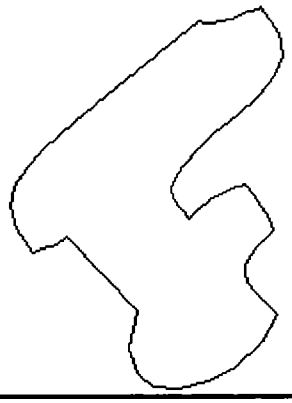
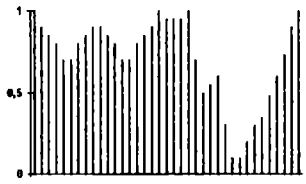
Vector promedio de la clase círculo (0.004, 0.986, 39.473)

Bordes de entrenamiento de la clase Cuadrado:

Borde	Rúbrica	Vector descriptor
		(0.31, 0.996, 39.86)
		(0.31, 0.79, 31.82)
		(0.288, 0.798, 3.93)
		(0.307, 0.803, 32.12)
		(0.297, 0.803, 32.13)

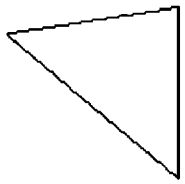
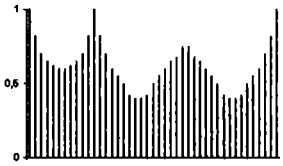
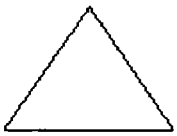
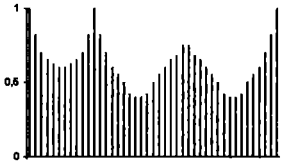
Vector promedio de la clase Cuadrado: (0.302, 0.801, 32.043)

Bordes de entrenamiento de la clase Plano_A:

Bordes	Rúbrica	Vector descriptor
		(2.55, 0.59, 23.93)
		(2.59,0.60, 24.02)
		(2.70, 0.60, 24.30)

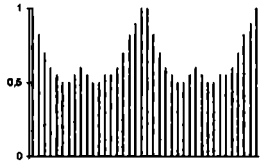
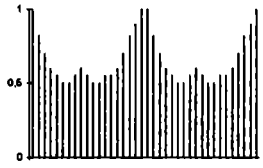
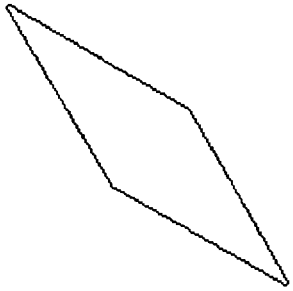
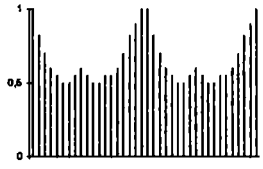
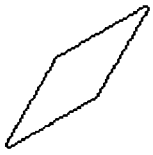
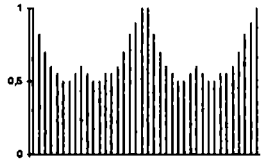
Vector promedio de la clase Plano_A: (3.045 - 0.674 - 26.971)

Bordes de entrenamiento de la clase Triángulo:

Bordes	Rúbrica	Vector descriptor
		(1.12, 0.50, 20.23)
		(1.28, 0.53, 21.45)

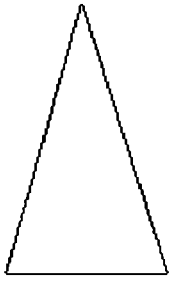
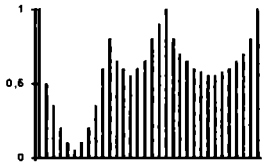
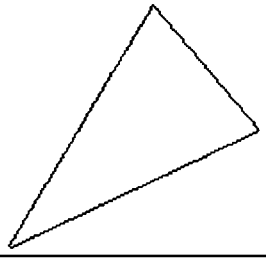
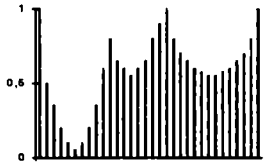
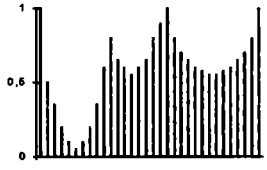
Vector promedio de la clase Triángulo: (1.187, 0.523, 20.903)

Bordes de entrenamiento de la clase Rombo:

Bordes	Rúbrica	Vector descriptor
		(1.32, 0.39, 15.74)
		(1.36, 0.39, 15.69)
		(1.34, 0.39, 15.78)
		(1.39, 0.40, 16.23)

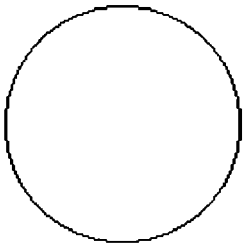
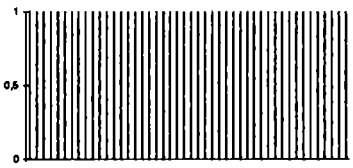
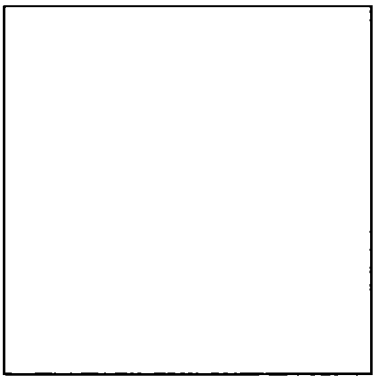
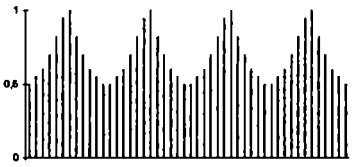
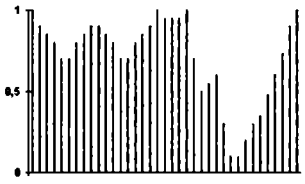
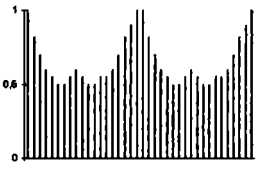
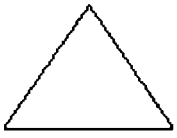
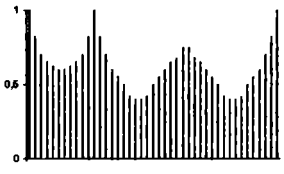
Vector Promedio de la clase Rombo: (1.353, 0.395, 15.806)

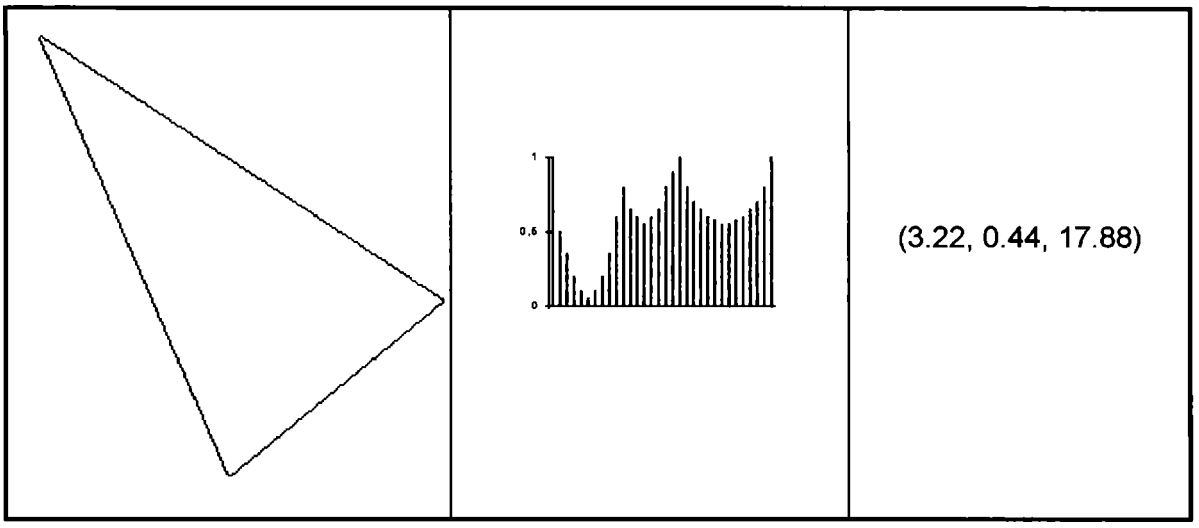
Bordes de entrenamiento de la clase Isósceles:

Bordes	Rúbrica	Vector descriptor
		(3.28, 0.44, 17.81)
		(2.89, 0.45, 18.17)
		(2.82, 0.47, 18.78)

Vector promedio de la clase Isósceles: (3.001, 0.456, 18.261)

Bordes clasificados por el algoritmo

		(0.00034, 0.996, 39.86)
		(0.31, 0.79, 31.82)
		(2.62, 0.60, 24.17)
		(1.36, 0.39, 15.67)
		(1.25, 0.53, 21.45)



En general el tiempo utilizado por el algoritmo, desde que se ingresa el borde ya segmentado hasta que este es clasificado, ronda los 30 segundos.

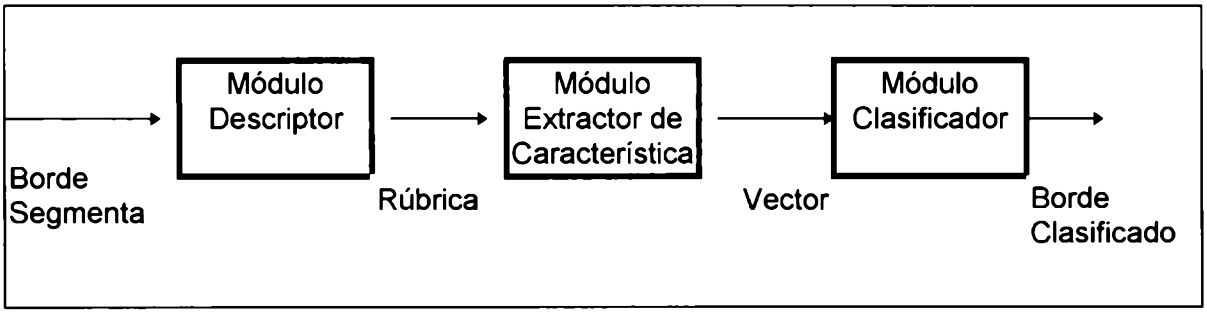


Figura IV.2. Procedimientos utilizados en el Algoritmo.

La figura IV.2 muestra los procedimientos específicos que fueron implementados. Podemos aclarar que el tiempo utilizado se distribuye de la siguiente manera:

90% Módulo Descriptor.
10 % Módulos de extracción y Clasificación.

Esta utilización del tiempo es obvia dado que el módulo descriptor calcula la rúbrica recorriendo el borde y sacando la distancia desde el punto central a todos sus puntos. La mayor parte del tiempo es usada en recorrer el borde. Por otro lado los módulos de extracción y clasificación hacen cálculos simples sobre valores de tipo reales.

En cuanto a la eficiencia del algoritmo, por lo menos en los casos estudiados, obtuvimos respuestas favorables en todos ellos.

Nos hemos planteado la posibilidad de paralelización para mejorar el tiempo de ejecución, el cual es de suma importancia en el momento de llevarlo a una aplicación real como puede ser el reconocimiento de formas a través de visión robótica. [HUS91]

En el caso que estamos planteando es obvio que el módulo descriptor es el más indicado a ser paralelizado, de alguna manera, dado el tiempo relativo que este consume. Por ejemplo, se podría calcular la distancia del punto central a todos los puntos del borde al mismo tiempo, lo cual reduciría en forma considerable el tiempo de ejecución, haciéndose más importante a medida que los bordes a clasificar crecen en tamaño.

De todas maneras la posibles mejoras a ser implementadas, ya sea paralelización o alguna otra técnica, dependen de los algoritmos elegidos, por eso en nuestro trabajo de grado planteamos otros métodos para el reconocimientos de bordes cerrados los cuales presentan otras características tanto en la manera de encarar el problema como en los resultados obtenidos.

Existen en la actualidad diversos lenguajes y plataformas de hardware que permiten realizar la paralelización de estos algoritmos con el objetivo de mejorar el tiempo de ejecución. Por ejemplo utilizando una arquitectura DSP. [ACO96].

Capítulo V
Conclusiones y Líneas de trabajo

Hemos analizado la complejidad, eficiencia y tiempo de ejecución de cada uno de los procedimientos involucrados en este algoritmo, mostrando los resultados obtenidos.

Hemos concluido que un algoritmo de reconocimiento general de formas es algorítmicamente complejo y por lo tanto es necesario especificar el problema.

El algoritmo del *shape number* tuvo problemas con la digitalización del borde, haciendo que el método no resulte eficiente.

El algoritmo de la *rúbrica* resultó ser capaz de reconocer y clasificar diversas formas con éxito, aunque no ignoramos que el mismo y sus resultados están muy ligados al tipo de reconocimiento a realizar.

Los inconvenientes encontrados en los algoritmos estudiados no son críticos, y creemos que un análisis pormenorizado de los mismos puede llevar a algoritmos más completos, y eficaces, basados en los anteriores.

Resultaría interesante estudiar la paralelización de los algoritmos de reconocimientos de bordes tal como los que hemos planteado, el cual es una de las líneas de trabajo en el LIDI.

Bibliografía.

[BAX94] "Digital Image Processing"
Gregory A.Baxes.
John Wiley, 1994.

[DOU87] "Matrix Structured Image Processing"
E. R. Dougherty, C.R. Giadina.
Prentice-Hall Inc., 1987

[FOL90] "Computer Graphics"
J. Foley, A. van Dam, S. Feiner, J. Hughes.
Addison-Wesley Publishing Comp., 1990.

[GEV87] "Máquinas Inteligentes"
William M. Gevarter.
Ediciones Díaz Santos 1987.

[GON92] "Digital Image Processing"
Rafael Gonzalez.
Addison-Wesley 1992.

[HEE91] "Parallel Algorithms in Computacional Science"
D. W. Heermann, A. N. Burkitt.
Springer-Verlag, 1991.

[HUS91] "Digital Image Processing practical applications of parallel processing techniques".
Zahid Hussain.
Ellis Horwood Series,1991

[IEEE] Colección de "Computer Graphics and Applications", IEEE.

[IEEE] Colección de "IEEE Transactions on Computers", IEEE.

[JAI89] "Fundamentals of Digital Image Processing".
Anil K. Jain.
Prentice Hall Inc., 1989.

[NIE90] "Pattern Analysis and Understanding"
Heinrich Niemann.
Springer-Verlag 1990.

[PAO89] "Adaptative Pattern Recognition and Neuronal Networks"
Yoh-Han-Pao.
Addison-Wesley, 1989.

[PRA78] "Digital Image Processing".
W. Pratt.
Wiley, 1978.

[RIM92] "Supercharged bitmapped graphics"
Rimmer.
McGraw-Hill, 1992

[SCH92] "Pattern Recognition"
R. Schalkoff.
Wiley, 1992.

[ZAV91] "Computer Architecture for Machine Perception"
B. Zadovique, P. Wendel (Ed).
Camp91, 1991

[ACO96] "Evaluación de Algoritmos de Procesamiento Paralelo sobre DSP y multi-DSP"
L. Acosta Burllaile - A. Cosentino - M. De Andrea.

[ACO95] "Algoritmo paralelizable para el reconocimiento de patrones de figuras geométricas".
L. Acosta Burllaile - A. Cosentino - M. De Andrea.
Second International Congress of Information Engineering 1995.

[HAR92] "Computer and Robot Vision"
Robert M. Haralick - Linda Shapiro
Adisson Wesley, 1992.