



Casa abierta al tiempo

UNIVERSIDAD AUTONOMA METROPOLITANA

**"DESCRIPCION Y MEDICION DEL
GRADO DE PARECIDO DE FORMAS
(OBJETOS) TRIDIMENSIONALES"**

TESIS QUE PRESENTA

ERNESTO BRIBIESCA CORREA

PARA LA OBTENCION DEL GRADO DE

DOCTOR EN CIENCIAS

MARZO, 1996

UNIVERSIDAD AUTONOMA METROPOLITANA-IZTAPALAPA

DIVISION DE CIENCIAS BASICAS E INGENIERIA

UNIDAD IZTAPALAPA

Av. Michoacán y La Purísima, Col. Vicentina, 09340 México, D.F. Tel.: 724-4600 TELEFAX: (5) 612 0885

Agradecimientos

Deseo agradecer a las siguientes instituciones: UAM-Iztapalapa, IIMAS-UNAM y CONACYT, las facilidades que me otorgaron para llevar a buen término mi programa doctoral. Mi agradecimiento al Dr. Ignacio Méndez Ramírez, por su apoyo decidido para la realización de esta tesis. En forma especial a mi asesor Dr. Richard G. Wilson, por su apoyo constante en la realización de mi trabajo doctoral. A mis sinodales: Dr. Eduardo Rivera Campo, Dr. Humberto Sossa Azuela, Dr. John Goddard Close y Dr. Gabriel Corkidi Blanco.

Al Dr. Eleuterio Castaño Tostado, Coordinador del Doctorado en Ciencias, por su confianza y ayuda en todos los problemas que se me presentaron. Al Ing. Bulmaro Cabrera Ruíz, por su apoyo constante. A la Lic. María Ochoa Macedo por sus comentarios para mejorar la presentación de este trabajo.

A mi madre, a su recuerdo, de quien siempre recibí apoyo y estímulo para mi formación profesional. A mi padre, de quien aprendí el valor del trabajo.

A mis hijos, quienes han sido fuente de motivación y superación.

Y a mi esposa, por su amor y comprensión.

INDICE

	Páginas
1 Introducción	5
2 Análisis de Forma en 2D	8
2.1 Descripción de contorno	8
2.2 Conceptos	10
2.2.1 Formas continuas y discretas	10
2.2.2 Conversión de una curva continua en una discreta	10
2.2.3 Independencia en traslación y rotación	12
2.2.4 Escala e independencia en el tamaño	12
2.3 Definiciones	16
2.4 Ejemplos del uso de la Notación de Curvatura Discreta	29
2.5 Representación de mallas 3D usando la Notación de Curvatura Discreta	34
2.6 Tamaño variable de segmento en función del cambio de dirección	40
2.7 La Notación de Curvatura Discreta como una gramática	40
2.7.1 Detección de características por medio de una gramática	41
2.7.2 Resultados	43
3 Descripción de Objetos 3D	48
3.1 Método de graficación de objetos representados por <i>voxels</i>	48
4 Invariantes en Traslación y en Rotación	52
4.1 Invarianza en traslación	52
4.2 Invarianza en rotación	53
4.2.1 Solución estándar	53
4.2.2 Representación por cuaternios	54

4.2.3 Los ejes mayor y menor	56
4.3 Invarianza en traslación y rotación usando máxima correlación	57
4.3.1 Invarianza bajo área	58
4.3.2 Máxima correlación	62
5 Voxelización	66
6 Transformación de un Objeto a Otro	72
6.1 El trabajo realizado para transformar un objeto a otro	72
6.2 Ejemplos de transformaciones de objetos	75
7 El Trabajo Desarrollado como Medida de Desemejanza entre Objetos	81
8 Conclusiones	85
8.1 Artículos derivados de esta tesis	85
Apéndice A: Información 3D Utilizada	87
Bibliografía	93

1 Introducción

El reconocimiento de la *forma* de los objetos es uno de los mayores retos en visión por computadora. La forma, a diferencia de otros parámetros (color, textura y contexto), es más difícil de cuantificar. Se ha avanzado en diferentes caminos para reconocer la forma de los objetos, entre los principales destacan: los patrones estructurales, los estadísticos y los sintácticos. El trabajo aquí presentado se basa principalmente en patrones estructurales, aunque en su etapa inicial se empezó a trabajar con patrones sintácticos, usando la notación de *curvatura discreta* [1] aplicada a una gramática [2].

Para resolver el problema del reconocimiento de la forma se hace necesario seleccionar una estructura geométrica adecuada para representar objetos o imágenes 3D. La representación utilizada en este trabajo se basa en celdas inmersas en arreglos tridimensionales, llamados *voxels*. Este tipo de representación consume una gran cantidad de almacenamiento si la resolución es alta, la cual crece en forma cúbica [3]. Sin embargo, a la fecha con la baja en costos de memoria y de dispositivos de almacenamiento, este tipo de notaciones son más usadas.

El primer programa de representación de objetos en 3D fue desarrollado por Roberts [4]. El principal objetivo era aceptar una imagen digitalizada de una escena de un poliedro y producir un dibujo lineal de esa escena. Un programa desarrollado por Guzmán [5] tomaba como dibujo lineal de entrada un poliedro y agrupaba regiones poligonales en conjuntos. Esos conjuntos permitieron tener una descripción aceptable de la escena.

Varios autores han usado diferentes técnicas para el reconocimiento de objetos 3D a partir de una vista 2D de los objetos [6]. En esos, a partir de una imagen de entrada, un conjunto de características o primitivas son extraídas. Requicha [7] y Besl [8] representan los sólidos por medio de sus fronteras o de las superficies que los encierran. Otros autores han usado esquemas de geometría sólida constructiva [9] y [10], donde los sólidos son representados como composiciones, por medio de operaciones de conjuntos

o de otros sólidos los cuales pueden tener movimientos rígidos. Soroka [11,12] y Brooks [13] presentan los cilindros generalizados como primitivas de volumen. Finalmente, Pentland [14] introduce los *superquadrics*.

En la tesis se presenta una *medida de semejanza* entre objetos tridimensionales. Así, el objeto es mapeado a una notación invariante bajo transformaciones geométricas afines: traslación y rotación. Los diferentes objetos a ser comparados también son normalizados a tener la misma cantidad de información para ser descritos, esto es, tener el mismo número de *voxels* cada uno y es llamado *invarianza en volumen*.

Cuando los diferentes objetos ya son invariantes en: traslación, rotación y volumen; se procede a realizar la transformación de un objeto A a otro B , esta transformación se realiza por medio del movimiento de *voxels* de uno al otro. Lo anterior genera una cantidad de trabajo desde el punto de vista de la Física, la cual es cuantificada y será la base para medir la semejanza. Los *voxels* que se van a mover son seleccionados de tal forma que el trabajo se minimiza. Por lo tanto, la *cantidad de trabajo* desarrollado para transformar un objeto a otro es directamente proporcional a la *medida de desemejanza* entre ambos. Así, objetos semejantes emplearán en su transformación de uno al otro una cantidad pequeña de trabajo. Mientras que objetos desemejantes consumirán una mayor cantidad de trabajo. Cuando dos objetos son idénticos la cantidad de trabajo desarrollado para transformar de uno al otro es cero. Así, la diferencia de forma entre dos objetos es obtenida contando el número de *voxels* a mover y cuantificando sus distancias de recorrido.

La investigación para analizar y describir formas binarias tomó tres caminos. Primero, se investigó como describir formas bidimensionales. La principal contribución fue la notación de curvatura discreta, la cual se presenta en el capítulo 2 y en la referencia [1]. Segundo, fue necesario investigar como representar objetos tridimensionales por medio de *voxels*. Los principales resultados de esta investigación se muestran en el capítulo 3 y los programas tomados del Apéndice A. Tercero, y lo que se considera

como el objetivo principal de este trabajo, se halló una forma original de medir el grado de parecido entre dos formas tridimensionales, basada en el concepto de trabajo (energía por espacio) necesario para mover ciertos *voxels* para transformar una forma a otra. Mientras más *voxels* se tengan que mover, más trabajo se realiza, y más diferentes son las formas. El análisis de esto se muestra en los capítulos del 4 al 7 y en el artículo *Measuring 3D Shape Similarity Using Progressive Transformations*, aceptado en la revista *Pattern Recognition*. Finalmente, en el capítulo 8 se presentan conclusiones.

2 Análisis de Forma en 2D

Varios autores han estado trabajando en métodos de reconocimiento de objetos en imágenes. Algunos se han enfocado a la descripción del contorno de formas, por medio del uso de las cadenas de Freeman [15], transformadas de ejes medios, descomposición de subconjuntos convexos, coordenadas polares [16], descomposición en vértices cóncavos; descomposición por agrupamientos, por ejes de espejo [17] y detectores de quiebres. Esos y otros métodos son analizados por Pavlidis [18]. Otros autores [19] han analizado la teoría de la codificación de la percepción de patrones para diferentes representaciones, Ballard and Brown [3] describen las principales estructuras geométricas. El uso de descriptores de Fourier para representar la frontera de una forma u objeto como una función periódica, la cual puede ser expandida como una serie de Fourier [20].

2.1 Descripción de contorno

Esta sección se basa en trabajos previos [1, 21 y 22], en relación a descripción de contorno y fueron parte de los antecedentes para el presente trabajo de tesis. Se propone una estructura geométrica llamada: *Notación de Curvatura Discreta* NCD. La NCD de una *curva* se obtiene colocando segmentos con longitud constante de línea recta sobre la curva (los extremos de los segmentos de línea recta tocando siempre la curva), y se calculan los cambios de dirección entre segmentos contiguos, escalados en un rango continuo de -1 a 1 .

La NCD es independiente de la traslación y de la rotación, esto debido a que los cambios de dirección alrededor de la curva son utilizados. Esta notación es unidimensional, esto es una característica importante, ya que formas con rasgos particulares, pueden ser generadas por una secuencia numérica conocida. También es posible hacer operaciones aritméticas entre formas. Al final de este capítulo se presenta la NCD con tamaño de segmento variable, como función de los cambios de dirección.

La NCD describe trayectorias o formas por medio del uso de elementos discretos en un espacio continuo. Usa secuencias numéricas llamadas *cadena*s [15], las cuales pueden generar formas de muchas características. La cadena es como una *semilla*, que contiene toda la información de la forma. Así es posible generar el universo discreto de todas las formas cerradas de 10 elementos a múltiplos de 60° .

La NCD es similar a otros códigos de cadenas, pero tiene diferencias importantes, Por ejemplo, las principales diferencias de esta notación con el código de Freeman [15] son las siguientes:

Características de las cadenas de Freeman

- 1.- Usan una cuadrícula.
- 2.- Son independientes de la traslación. También, de la rotación si se usa la derivada de Freeman (esto en múltiplos de 45°).
- 3.- El rango de direcciones va del 0 al 7 (cada 45°)
- 4.- La longitud del segmento de línea recta es de l en los elementos pares y de $l\sqrt{2}$ en los impares.

Características de la NCD

- 1.- No usa una cuadrícula.
- 2.- Es independiente de la traslación, de la rotación, y opcionalmente del tamaño.
- 3.- El rango de direcciones es ilimitado (varía continuamente entre -1 y 1); esto facilita la generación de secuencias numéricas. Los vértices siempre tocan la curva. Así, una mejor descripción de la curva es obtenida.
- 4.- La magnitud del segmento de línea recta (l) siempre es la misma para toda la forma.

Las diferencias entre las cadenas de Freeman y la notación propuesta pueden representar ventajas o desventajas, dependiendo del problema que se esté resolviendo.

2.2 Conceptos

El propósito en esta sección es el de presentar los conceptos y definiciones básicas de la NCD. Una importante suposición es que una *identidad* ha sido aislada de la imagen, llamada la *forma*, y está definida como el resultado de un procesamiento previo: formas compuestas de puntos discretos.

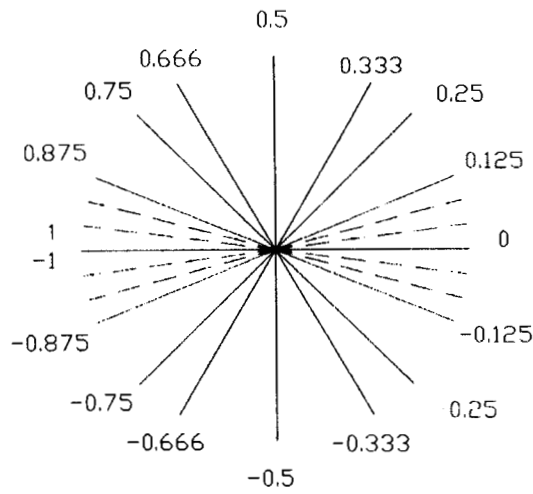
Usando la NCD los cambios de dirección varían continuamente de -1 a 1 . La Figura 1(a) ilustra el rango de los cambios de dirección y la 1(b) algunos ejemplos de éstos. Cualquier cambio de dirección entre $[-1, 1]$ es válido.

2.2.1 Formas continuas y discretas

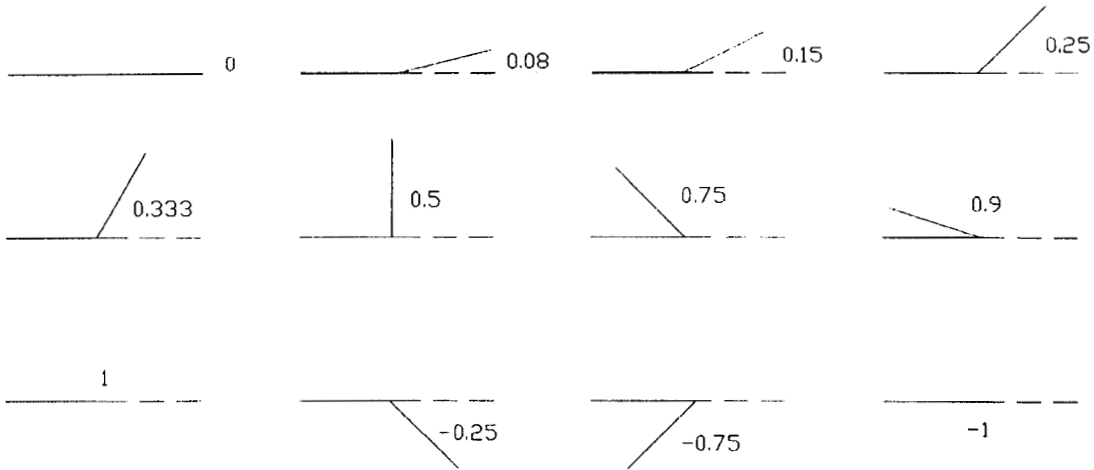
Una forma es *discreta* si la frontera de la región esta formada por segmentos rectilíneos. De otra manera, la forma es *continua*.

2.2.2 Conversión de una curva continua en una discreta

Para convertir una curva continua en una curva discreta, primero de debe seleccionar un segmento de línea recta. Una vez hecho esto se selecciona un punto de la curva como origen, si la curva es abierta se debe seleccionar un extremo. Posteriormente se hace coincidir un extremo de segmento con el origen seleccionado de la curva. Se hace girar hasta que el otro extremo toque la curva, minimizando la longitud de la curva entre los extremos; éste será el nuevo punto para el siguiente segmento, y así sucesivamente.



(a)



(b)

Figura 1. Cambios de dirección: (a) rangos de $[0, 1]$ y de $[0, -1]$: (b) Algunos ejemplos de cambios de dirección.

A continuación se calcula el ángulo entre los segmentos contiguos, normalizado a un rango de $[0, 1]$ para ángulos positivos y de $[0, -1]$ para ángulos negativos. Estos ángulos serán llamados *cambios de dirección*. La secuencia de cambios de dirección es la cadena que describe a la curva. La Figura 2 muestra la conversión de una curva continua a una discreta.

2.2.3 Independencia en traslación y rotación

Obviamente en la NCD cualquier forma es independiente de la *traslación*. Usando la NCD cualquier forma es independiente de la *rotación*, ya que se está calculando el cambio de dirección o derivada en cada unión de segmentos contiguos. La Figura 3 muestra un ejemplo de la independencia en rotación de la NCD, siempre y cuando se elija el mismo punto de inicio.

2.2.4 Escala e independencia en el tamaño

La escala de cualquier forma está dada por la *longitud del segmento de línea recta l*. En reconocimiento de formas, la independencia en el tamaño de un objeto o forma es importante, por ejemplo, para medir la semejanza entre formas [22]. La independencia en el tamaño de dos objetos se obtiene por la *normalización* de los perímetros de ambos, en un número fijo de elementos. Así, cada longitud de perímetro es dividida en el mismo número de elementos, y posteriormente comparada. La Figura 4 muestra la invarianza en tamaño. Note que el primer segmento de ambas formas es sólo un segmento de referencia para inicializar la cadena, no cuenta como parte de la cadena.

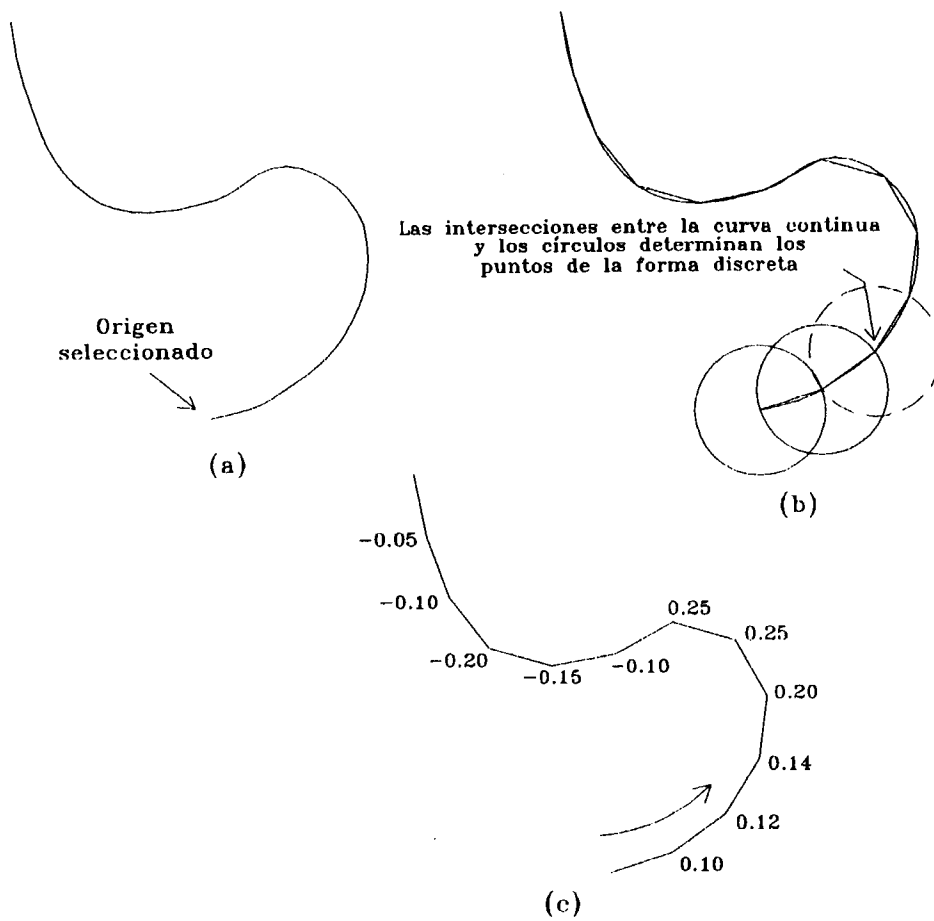


Figura 2. Conversión de una curva continua a una discreta: (a) curva continua con un origen seleccionado; (b) generación de círculos sobre la curva; (c) determinación de los cambios de dirección, los cuales definen la curva discreta.



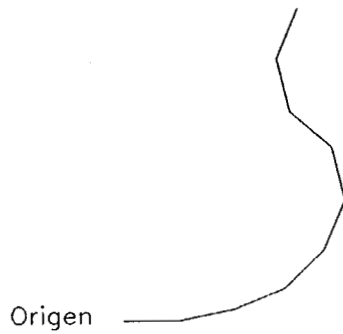
$$\frac{1}{8} \quad \frac{1}{8} \quad \frac{-1}{8} \quad \frac{-1}{6} \quad \frac{-1}{6} \quad \frac{-1}{6} \quad \frac{-1}{6} \quad \frac{-1}{6} \quad \frac{1}{2}$$



$$\frac{1}{8} \quad \frac{1}{8} \quad \frac{-1}{8} \quad \frac{-1}{6} \quad \frac{-1}{6} \quad \frac{-1}{6} \quad \frac{-1}{6} \quad \frac{-1}{6} \quad \frac{1}{2}$$

Figura 3. Independencia en rotación.

No. de elementos, $n=8$



Longitud de la curva, $P=8$
Longitud del segmento, $l=P/n$



$P=3.2$
 $l=(3.2)/8=0.4$

Cadenas:

$$\frac{1}{16} \frac{1}{16} \frac{1}{8} \frac{1}{8} \frac{1}{5} \frac{1}{5} \frac{-1}{5} \frac{-1}{5}$$

$$\frac{1}{16} \frac{1}{16} \frac{1}{8} \frac{1}{8} \frac{1}{5} \frac{1}{5} \frac{-1}{5} \frac{-1}{5}$$

Figura 4. Invarianza en tamaño usando la NCD.

2.3 Definiciones

Definición 1. Un *elemento* a_i de una cadena A es el cambio de dirección entre la dirección m_i y la dirección m_{i-1} , usando un segmento de línea recta de longitud l (l previamente definido). En otras palabras, el cambio de dirección en esa posición es la derivada. La Figura 5(a) muestra un elemento a_i .

Definición 2. Una *cadena* [15] es una secuencia ordenada de elementos, y está definida por:

$$A = a_1 a_2 a_3 \dots a_n = C_{i=1}^n a_i$$

donde n es el número de elementos, y C indica una secuencia de elementos. La Figura 5(b) muestra un ejemplo de una cadena.

Definición 3. La *longitud* L de una cadena es el producto del número de elementos por la magnitud del segmento de línea recta (l), y se representa por:

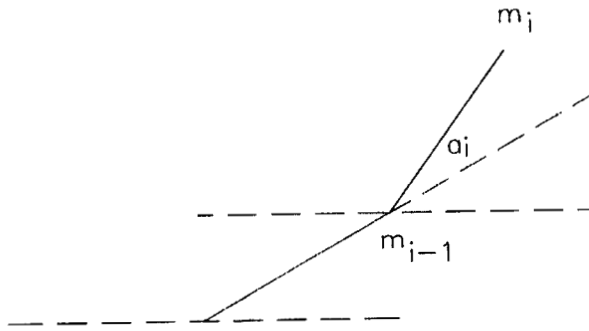
$$L = nl.$$

Así, la longitud de la cadena A de la Figura 5(b) es 12 para $l = 1$. Note que el primer segmento de la gráfica de la Figura 5(b) es sólo de referencia y que el paréntesis en la cadena no es una multiplicación, solamente indica el número de elementos iguales.

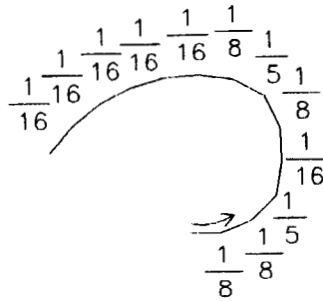
Definición 4. El *cambio de dirección acumulado* Acc de una cadena es la suma de todos sus elementos, y está definido por:

$$Acc = \sum_{i=1}^n a_i$$

Cualquier forma cerrada tiene un cambio de dirección acumulado de 2 o -2 . También un cambio acumulado mayor de 2 indica una forma con cruces internos o autointersecciones. Formas con cambio de dirección acumulado mayor de 1 pueden tener algún cruce. El valor de Acc para la cadena A de la Figura 5(b) es 1.275.



(a)



$$A = \frac{1}{8} \cdot \frac{1}{8} \frac{1}{5} \frac{1}{16} \frac{1}{8} \frac{1}{5} \frac{1}{8} \frac{1}{16} \frac{1}{16} \frac{1}{16} \frac{1}{16} \frac{1}{16}$$

$$A = 2 \left(\frac{1}{8} \right) \frac{1}{5} \frac{1}{16} \frac{1}{8} \frac{1}{5} \frac{1}{8} 5 \left(\frac{1}{16} \right)$$

$$n = 12 \text{ (No. de elementos)}$$

(b)

Figura 5. (a) Elemento a_i ; (b) ejemplo de una cadena de 12 elementos.

Definición 5. El *promedio de cambio de dirección* Am de una cadena es el promedio aritmético de todos los cambios de dirección, a lo largo de la cadena. El promedio de cambio de dirección es el acumulado dividido por el número de elementos, y está definido por la siguiente fórmula:

$$Am = \frac{1}{n} \sum_{i=1}^n a_i = \frac{1}{n} Acc$$

Definición 6. La *proyección de círculo discreto* Acp de una cadena es otra cadena con el mismo número de elementos manteniendo el valor del promedio de cambio de dirección en cada elemento, su descripción es:

$$Acp = C_{i=1}^n Am.$$

Cualquier cadena puede ser gráficamente representada usando la proyección de círculo discreto. La Figura 6(a) muestra la proyección de círculo discreto de la cadena A de la Figura 5(b).

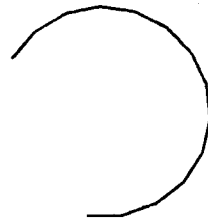
Definición 7. La *derivada* de una cadena

$$A = a_1 a_2 a_3 \dots a_n = C_{i=1}^n a_i,$$

es la cadena

$$A' = a_2 - a_1 \quad a_3 - a_2 \quad \dots \quad a_n - a_{n-1} = C_{i=2}^n (a_i - a_{i-1}).$$

La Figura 6(b) muestra la derivada de la cadena A .



$$A_{cp} = 12(0.106)$$

(a)



$$A' = 0 \frac{3}{40} \frac{-11}{80} \frac{1}{16} \frac{3}{40} \frac{-3}{40} \frac{-1}{16} 0 0 0 0$$

(b)

Figura 6. (a) La proyección de círculo discreto de la cadena A ; (b) la derivada de la cadena A .

Definición 8. Una *cadena nula* es una cadena que no contiene elementos. Se diferencia de la cadena cero, la cual sólo contiene elementos con valor cero.

Definición 9. Dos cadenas son *iguales* si hay igualdad entre cada uno de sus elementos ordenados; esto es,

$$C_{i=1}^n a_i = C_{i=1}^m b_i; \quad \text{si } m = n \text{ y } a_i = b_i; \text{ para toda } i$$

Definición 10. La *cadena simétrica* Asy de una cadena, es la misma con sus elementos ordenados en sentido contrario, esto es:

$$A = a_1 a_2 a_3 \dots a_n = C_{i=1}^n a_i$$

$$Asy = a_n \dots a_3 a_2 a_1 = C_{i=1}^n a_{n+1-i}$$

Generalmente hablando, la simetría entre cadenas se refiere a una simetría espejo, ya que no existen ejes de referencia. La Figura 7 muestra la cadena simétrica de A .

Definición 11. La *concatenación* de dos cadenas A y B representada por '//' es la cadena AB . El número total de elementos es la suma de los elementos de A (representados por m) y de los de B (representados por n), o sea $m + n$. El final de la cadena A coincide con el inicio de la cadena B . Por lo tanto la concatenación entre cadenas por lo general no es conmutativa, esto es,

$$C_{i=1}^n a_i // C_{i=1}^m b_i \neq C_{i=1}^m b_i // C_{i=1}^n a_i$$

Definición 12. La *cadena del círculo discreto* es una cadena de elementos iguales con un cambio de dirección acumulado de 2 ó -2 , esto es,

$$A = a_1 a_2 a_3 \dots a_n = C_{i=1}^n a_i$$

donde

$$a_1 + a_2 + a_3 + \dots + a_n = 2$$

y

$$a_i = a_j \quad \forall i, j \leq n$$

Por lo tanto, el valor numérico del elemento a_i está definido por:

$$a_i = \frac{2}{n}$$

Por ejemplo, si se desea generar un círculo discreto con 16 elementos, se necesita un cambio de dirección de $\frac{2}{16}$. En esta notación no es común usar el concepto de radio.

Definición 13. Los *ángulos internos* de una cadena son los ángulos suplementarios de cada dirección alrededor de la cadena. Dada una cadena: $A = a_1 a_2 a_3 \dots a_n = C_{i=1}^n a_i$ para $a_n > 0$ sus ángulos internos son:

$$b_i = \begin{cases} 1 - a_i & \text{si } a_i > 0; \\ -1 - a_i & \text{si } a_i < 0 \end{cases}$$

$$1 - a_1 \quad 1 - a_2 \quad 1 - a_3 \quad \dots \quad 1 - a_n$$

si el valor de un elemento es menor que cero, su ángulo interno será: $-1 + |a_i|$.

La *resultante* R de los ángulos internos de una cadena es la suma de todos los ángulos internos a lo largo de la cadena; esto es:

$$R = \sum_{i=1}^n (1 - |a_i|) \operatorname{sgn}(a_i)$$

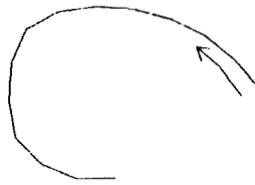
donde

$$\operatorname{sgn}(a_i) = \begin{cases} 1, & \text{si } a_i \geq 0; \\ -1, & \text{si } a_i < 0 \end{cases}$$

Para todos los círculos discretos se tiene $R = n - 2$. La Tabla 1 muestra los ángulos internos y resultantes para los círculos discretos (polígonos regulares).



$$A = 2\left(\frac{1}{8}\right) \frac{1}{5} \frac{1}{16} \frac{1}{8} \frac{1}{5} \frac{1}{8} 5\left(\frac{1}{16}\right)$$



$$A_{sy} = 5\left(\frac{1}{16}\right) \frac{1}{8} \frac{1}{5} \frac{1}{8} \frac{1}{16} \frac{1}{5} 2\left(\frac{1}{8}\right)$$

Figura 7. Cadena simétrica de A.








Círculo discreto	Perímetro	Cambio de dirección	Angulos internos	Resultante
	3	$1/1.5=0.666$	$0.5/1.5=0.333$	$3(0.333)=1$
	4	$1/2.0=0.500$	$1.0/2.0=0.500$	$4(0.500)=2$
	5	$1/2.5=0.400$	$1.5/2.5=0.600$	$5(0.600)=3$
	6	$1/3.0=0.333$	$2.0/3.0=0.666$	$6(0.666)=4$
	7	$1/3.5=0.285$	$2.5/3.5=0.714$	$7(0.714)=5$
	8	$1/4.0=0.250$	$3.0/4.0=0.750$	$8(0.750)=6$
	9	$1/4.5=0.222$	$3.5/4.5=0.777$	$9(0.777)=7$

Tabla 1. Angulos internos y resultantes para los círculos discretos (polígonos regulares).
Se nota que el divisor del cambio de dirección del círculo siguiente es incrementado en 0.5

Definición 14. La *integral* A^s de una cadena

$$A = a_1 a_2 a_3 \dots a_n = C_{i=1}^n a_i$$

es la cadena

$$A^s = a_1 \quad a_1 + a_2 \quad a_1 + a_2 + a_3 \quad \dots \quad a_1 + a_2 + a_3 + \dots + a_n = C_{i=1}^n \left(\sum_{j=1}^i a_j \right)$$

Por ejemplo, dada la cadena B :

$$B = 0.1 \quad 0.1 \quad 0.1 \quad 0.1 \quad 0.1$$

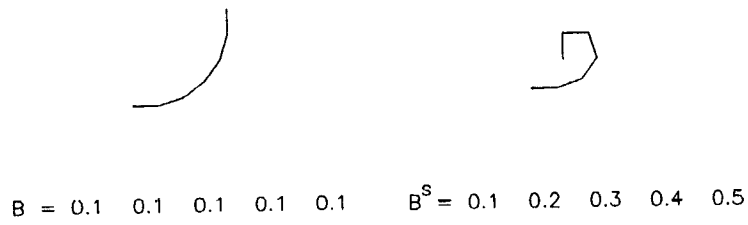
Su integral B^s de B es como sigue:

$$B^s = 0.1 \quad 0.2 \quad 0.3 \quad 0.4 \quad 0.5$$

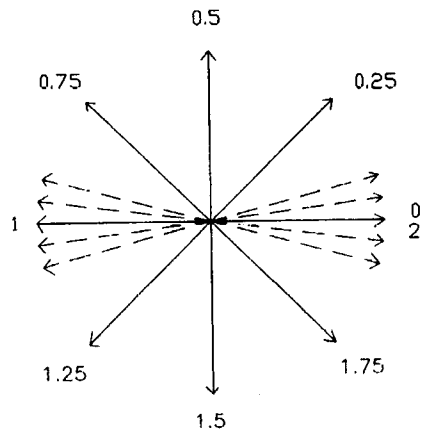
La Figura 8(a) muestra la cadena B y al 8(b) su integral.

La integral de la NCD genera un código de cadena, independiente solamente de la traslación. El rango de direcciones va de 0 a 2, usando números reales (el rango correspondiente en cadenas de Freeman va de 0 al 7, usando números enteros). La Figura 8(b) muestra las diferentes direcciones de la NCD. En forma contraria usando las direcciones mostradas en la Figura 8(b) en cada elemento de la integral de la cadena de B obtenemos la forma mostrada en la Figura 8(c).

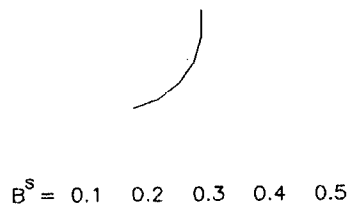
Note que la forma de la Figura 8(c) es igual a la cadena B (solamente sin el elemento de referencia). Esto es debido a que la integral y la derivada son operaciones opuestas. La integral de las cadenas genera un código similar a las cadenas de Freeman. La Figura 9 muestra las diferencias entre las cadenas del código de Freeman y las de la NCD.



(a)

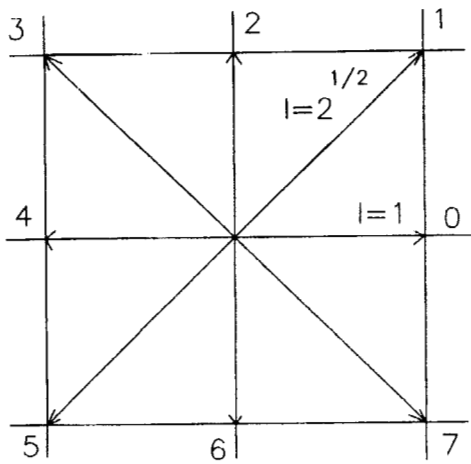


(b)



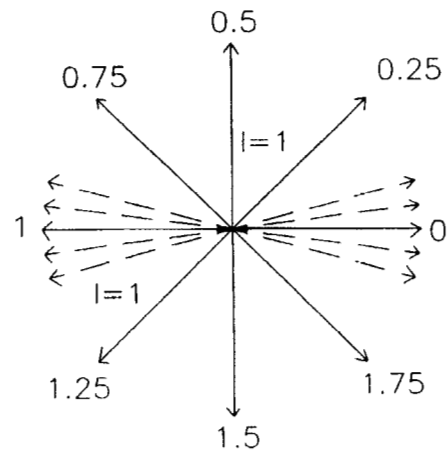
(c)

Figura 8. (a) Cadena B y su integral; (b) diferentes direcciones de la NCD; (c) código de cadena de la integral de B en la NCD.



$l=1$ para elementos pares
 $l=2^{1/2}$ para elementos nones

Cadenas de Freeman



$l=1$ para todos los elementos

Cadenas de la NCD

Figura 9. Diferencias entre las cadenas de Freeman y las cadenas de la notación de curvatura discreta.

Con $l = 1$ las componentes (incrementos x y y , respectivamente) para los dos códigos de cadena son como sigue:

Cadenas de Freeman			Integral de la NCD		
a_i	a_{x_i}	a_{y_i}	a_i	a_{x_i}	a_{y_i}
0	1	0	0	$\cos(\pi a_i)$	$\sin(\pi a_i)$
1	1	1	.	"	"
2	0	1	.	"	"
3	-1	1	.	"	"
4	-1	0	.	"	"
5	-1	-1	.	"	"
6	0	-1	.	"	"
7	1	-1	2	$\cos(\pi a_i)$	$\sin(\pi a_i)$

donde:

a_i es la dirección del código de cadena

a_{x_i} y a_{y_i} son los incrementos en los ejes x y y , respectivamente del plano de coordenadas rectangulares en cada dirección, con un segmento de línea recta igual a uno.

Lo anterior permite reconstruir las formas en el plano de coordenadas, esto se puede observar más claramente en el código de la *función cadena* (escrita en AutoLisp), que permite reconstruir formas en la NCD. El código de esta función es como sigue:

```
(defun cadena(/ p1 a b r n k)
  (graphscr)
  (setq p1 (getpoint "Punto inicial"))
  (setq r (getreal "Tamaño de segmento: "))
  (setq k (getint "No. de elementos: "))
  (setq b (getreal "Cambio de dirección: "))
  (setq a 0)
  (setq n 0)
```

```

(setq a (+ a b))
(while (< n k)
  (setq p2 (polar p1 (* a 3.141592654) r))
  (command "line" p1 p2 ""))
(setq b (getreal "Cambio de dirección: "))
(setq a (+ a b))
(setq p1 p2)
(setq n (+ n 1))
)
)

```

La *derivada de la cadena de Freeman* es obtenida substrayendo elemento por elemento de la cadena. La derivada de la cadena de Freeman se presentó por primera vez en el trabajo de números de forma de la referencia [22], y su objetivo principal es el tener un código de cadenas invariante en rotación. La Figura 10(a) muestra la derivada de las cadenas de Freeman. Su rango es de 0 a 4 y de 0 a -4 . Usando el concepto de cambio de dirección acumulado, el cambio para cualquier forma usando las derivadas de las cadenas de Freeman es de 8 o de -8 . La Figura 10(b) ilustra un ejemplo de la derivada de las cadenas y su cambio de dirección acumulado *Acc*, para una forma cerrada. Freeman presenta diferentes algoritmos para su código de cadenas, tales como:

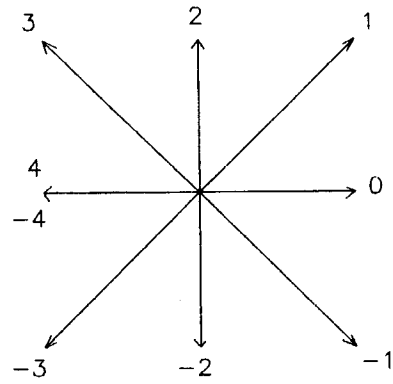
- a).- Determinación de ancho y altura
- b).- Determinación de residuos
- c).- Distancia entre dos puntos
- d).- Intersección de dos cadenas
- e).- Simetría
- f).- Cadenas simétricas
- g).- Integración

- h).- Cálculo del área cerrada
- i).- Determinación de momentos
- j).- Momentos de un eje diagonal
- k).- Centroides

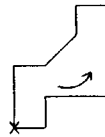
Considerando la similitud entre los dos códigos de cadenas y conociendo las relaciones entre sus componentes, no se hace necesario mostrar la solución de cada uno de estos algoritmos.

2.4 Ejemplos del uso de la Notación de Curvatura Discreta

Con el uso de la NCD es posible generar formas con características particulares a través de producir secuencias numéricas representadas por cadenas. La Figura 11 muestra ejemplos de algunas cadenas y sus representaciones geométricas. La notación utilizada en la Figura 11 es congruente a la notación usada por Freeman [15] y a la usada en la descripción de operadores relacionales, en lenguajes de programación y se indica de la siguiente manera: $C_{i=1}^{n,incr} a_i$ para $l = 1$, donde i es el elemento inicial, n el elemento final e $incr$ es el incremento de i , o $C_{i=1}^{l.e.,incr} a_i$ para $l = 1$, donde $l.e.$ es la expresión lógica. Esto es similar a las declaraciones DO y DO WHILE de algunos lenguajes de programación. En algunos ejemplos de la Figura 11 aparece un signo menos, el cual indica cambios de dirección negativos. La Figura 12 despliega parte de las posibles formas cerradas de 10 elementos y con cambios de dirección de $\frac{1}{3}$ ó $-\frac{1}{3}$. Esto se define como parte del *universo discreto de formas* de 10 elementos. La Figura 13 muestra una aplicación del uso de la NCD, que consiste en un problema de suavización de curvas. Primero, las cadenas de las curvas fueron divididas, truncadas y posteriormente multiplicadas, manteniendo el cambio de dirección original para cada cadena. Entre las características principales del la NCD es que de manera implícita ya es invariante en rotación, lo cual permite aumentar su número de aplicaciones, como posteriormente se menciona con el uso de técnicas gramaticales.



(a)



Cadenas de Freeman: 0 2 0 0 2 2 2 4 6 5 4 6 6

Derivada de las cadenas de Freeman: 2 -2 0 2 0 0 2 2 -1 -1 2 0 2

$$Acc = 2 - 2 + 0 + 2 + 0 + 0 + 2 + 2 - 1 - 1 + 2 + 0 + 2 = 8$$

(b)

Figura 10. (a) Derivada de las cadenas de Freeman; (b) un ejemplo de la derivada de las cadenas de Freeman y el cálculo del cambio de dirección acumulado Acc , para una forma cerrada.

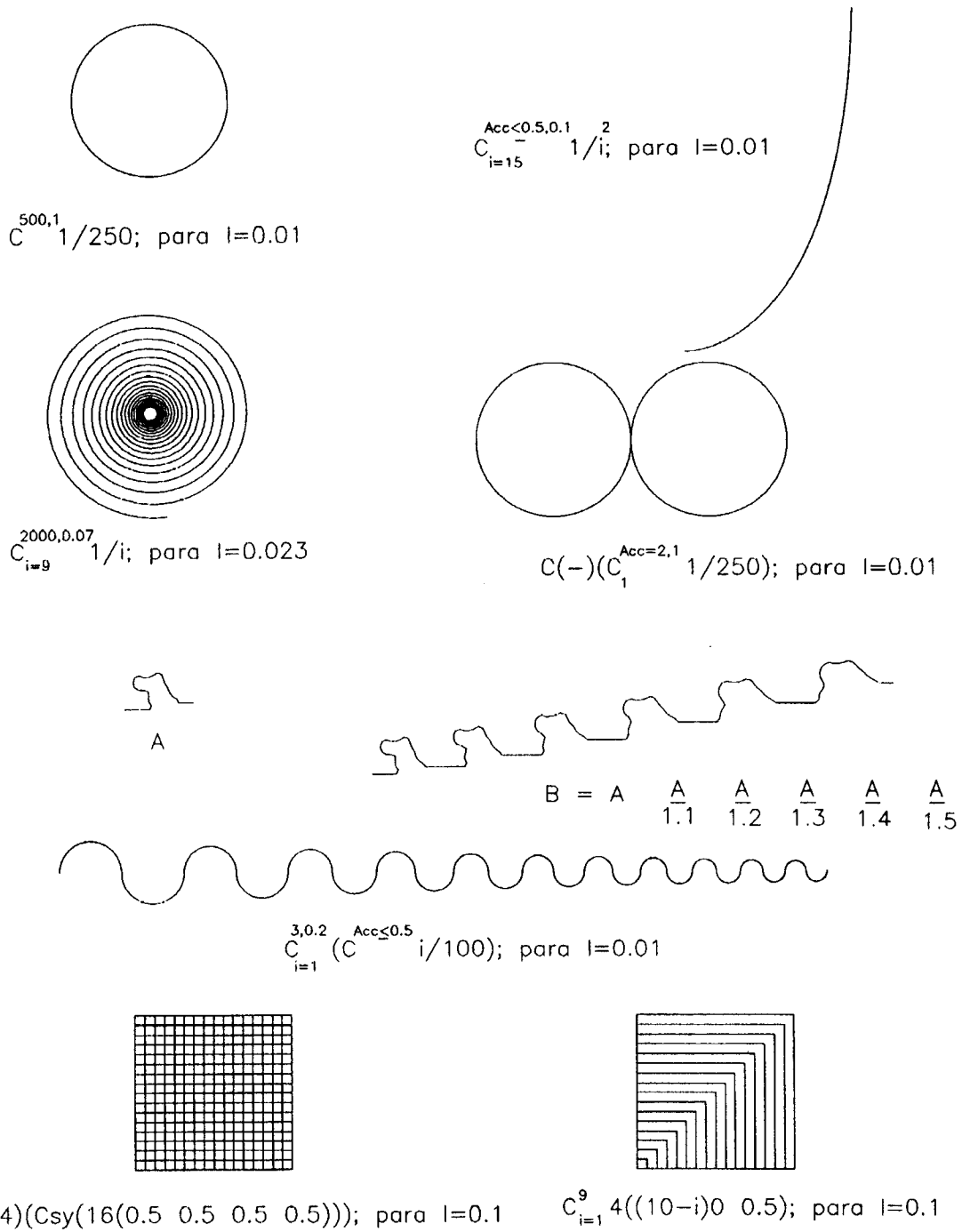


Figura 11. Ejemplos de cadenas con su representación geométrica.

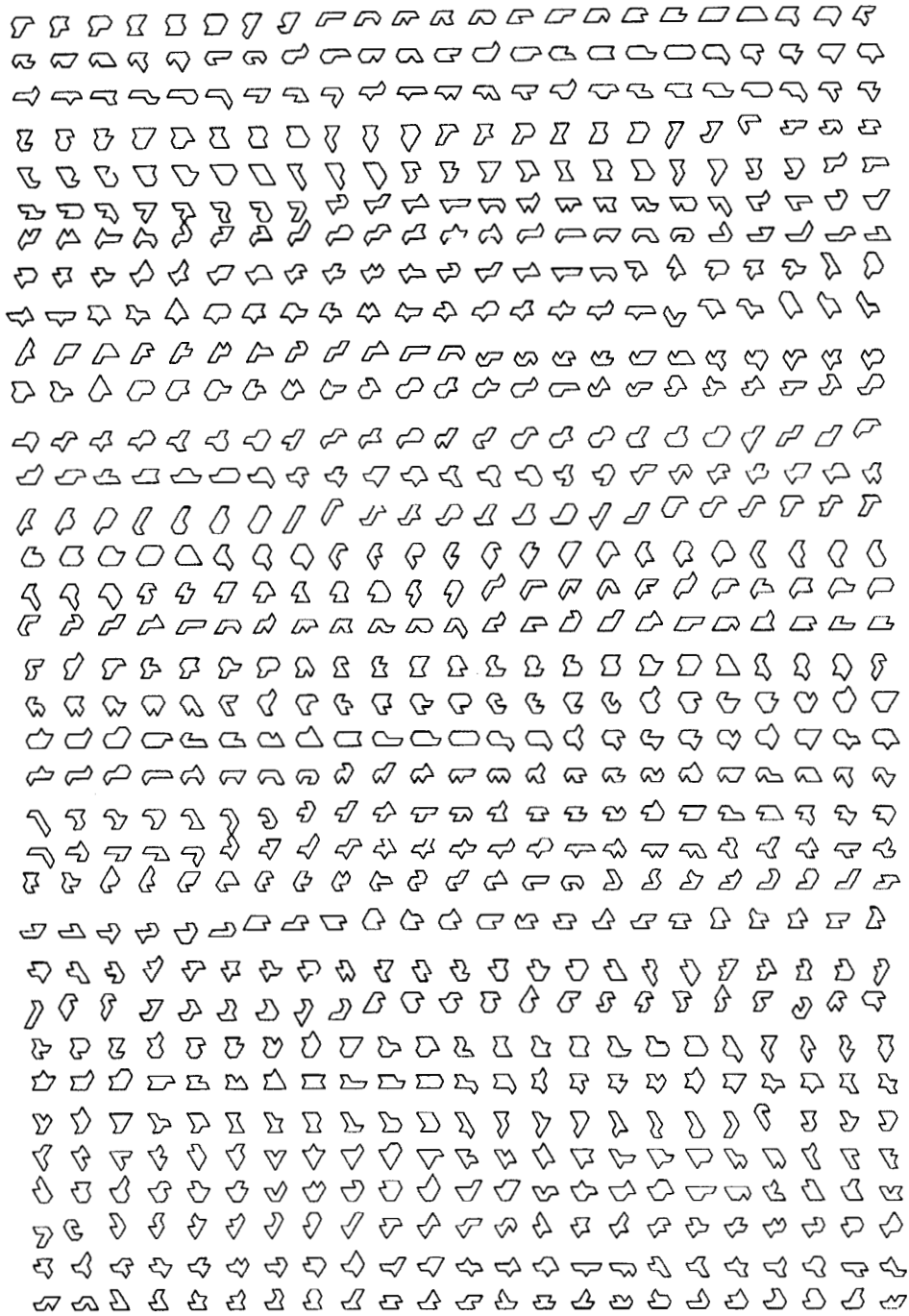


Figura 12. Parte del universo discreto de formas de 10 elementos, con cambios de dirección cada 60°.

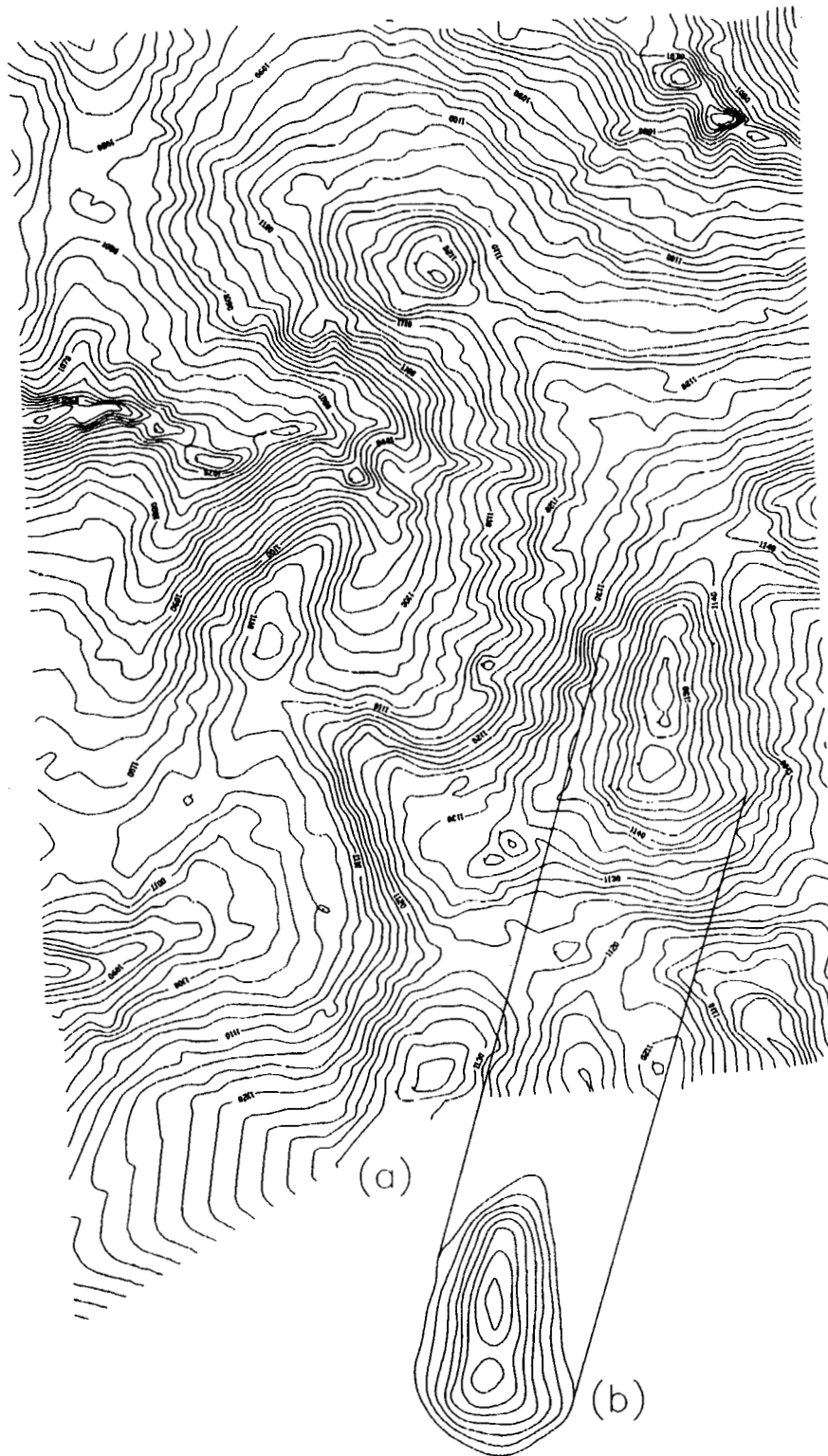


Figura 13. Ejemplos de suavización de curvas de nivel representadas por cadenas: (a) parte del mapa topográfico escala 1 : 2,000 de Tulimán, Estado de Guerrero; (b) curvas de nivel suavizadas utilizando técnicas de la NCD.

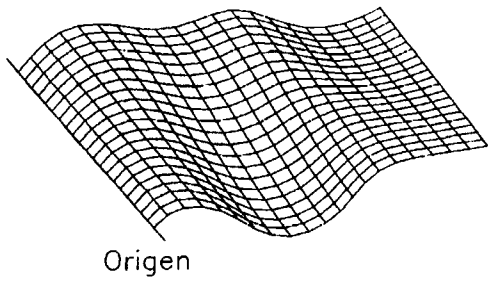
2.5 Representación de mallas 3D usando la Notación de Curvatura Discreta

Usando la NCD una superficie 3D puede ser representada por medio de una malla 3D colocandola sobre dicha superficie. Esta malla es normalizada a un número deseado de elementos (renglones \times columnas); un eje de la malla es identificado como *origen*, el cual será el inicio de todas las cadenas de la malla. Cada cadena comienza en un punto de coordenadas definidos por incrementos constantes sobre el eje del origen (la distancia entre los incrementos es igual al tamaño del segmento l). Así, una malla discreta 3D es una secuencia ordenada de cadenas, donde cada cadena tiene unas coordenadas de inicio como origen (ver la Figura 14 y también la Figura 2).

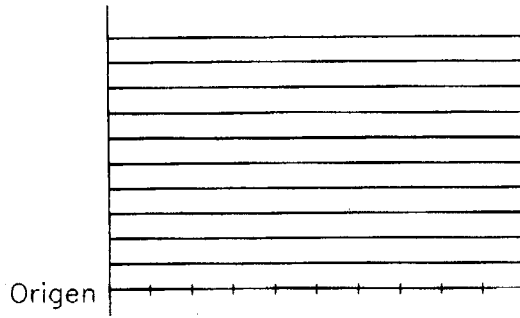
Los conceptos, definiciones y operaciones para formas 2D también se cumplen para mallas 3D. La Figura 15 muestra parte del universo discreto de las posibles superficies de cuatro cadenas usando mallas 3D, con sólo dos elementos de cambios de dirección con los siguientes valores posibles: -0.75 , -0.5 , -0.25 , 0 , 0.25 , 0.5 y 0.75 .

La Figura 16 ilustra el Modelo Digital de Terreno (MDT) correspondiente a la Ciudad de México, se utilizó una malla de 90×165 elementos. La parte izquierda coresponde al Norte de la ciudad, el lado derecho al Sur, la parte superior al Este y la parte baja al Oeste, respectivamente. En la parte norte se puede identificar con cierta facilidad la “Sierra de Guadalupe” y al Sur “El Ajusco”. Este tipo de mallas pueden ser representadas usando la NCD, como se verá a continuación.

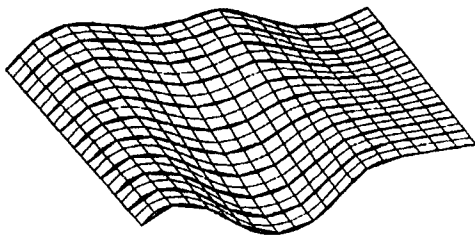
La Figura 17 muestra el MDT de la Ciudad de México usando una malla discreta 3D por medio del uso de cadenas de la NCD. Note que la longitud de cada perfil es constante (un perfil de terreno corresponde a una cadena de la NCD), esto debido a que cada cadena tiene el mismo número de elementos. La Figura 18 muestra una simple operación aritmética realizada a la malla: se le sumó un pequeño cambio de dirección a cada uno de sus elementos. Lo anterior puede ser de utilidad en el cálculo de proyecciones en fotogrametría.



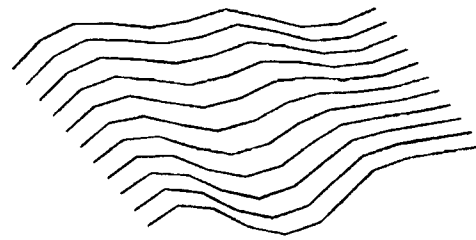
(a)



(b)



(c)



(d)

Figura 14. Conversión de una malla 3D continua en una malla discreta en la NCD: (a) un ejemplo de malla 3D; (b) definición del origen y del tamaño de los cambios de dirección para la malla discreta; (c) sobreposición de las mallas continua y discreta; (d) representación de la malla 3D discreta.

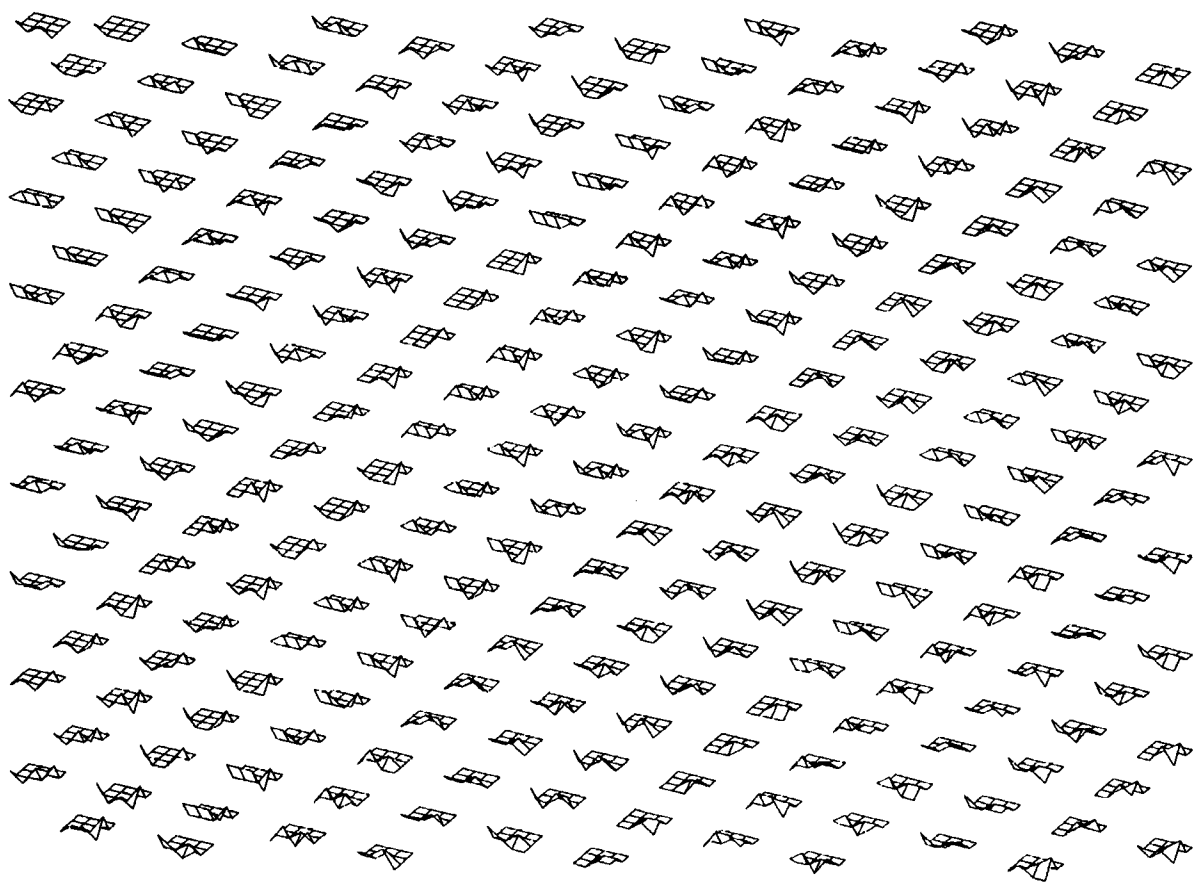


Figura 15. Parte del universo discreto de posibles superficies generadas por mallas 3D discretas de cuatro cadenas cada una, con solo dos cambios de dirección de valores posibles de: -0.75 , -0.5 , -0.25 , 0 , 0.25 , 0.5 y 0.75 .

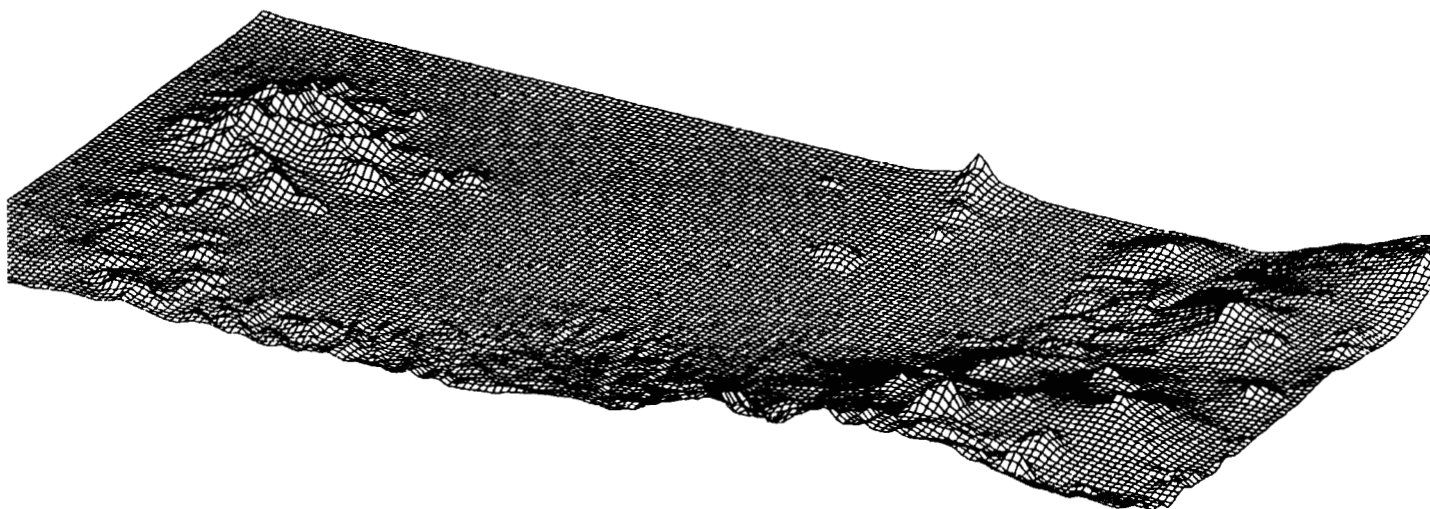


Figura 16. Modelo Digital de Terreno (MDT) de la Ciudad de México, representado en una malla 3D de 90×165 elementos. La parte izquierda corresponde al Norte de la ciudad, el lado derecho al Sur, la parte superior al Este y la parte baja al Oeste, respectivamente.

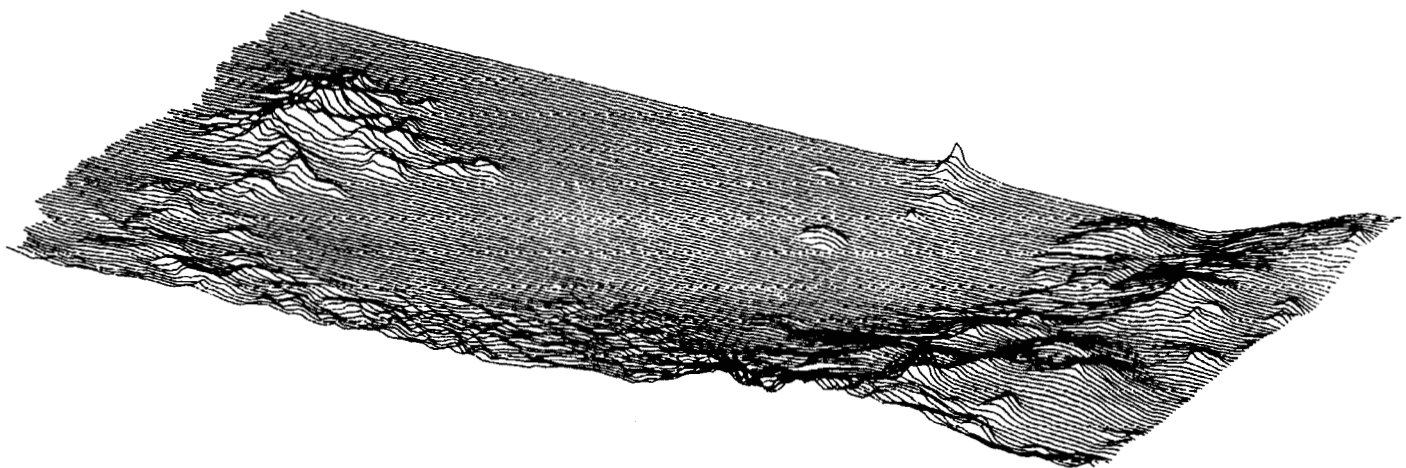


Figura 17. El MDT de la Ciudad de México representado por cadenas en NCD.

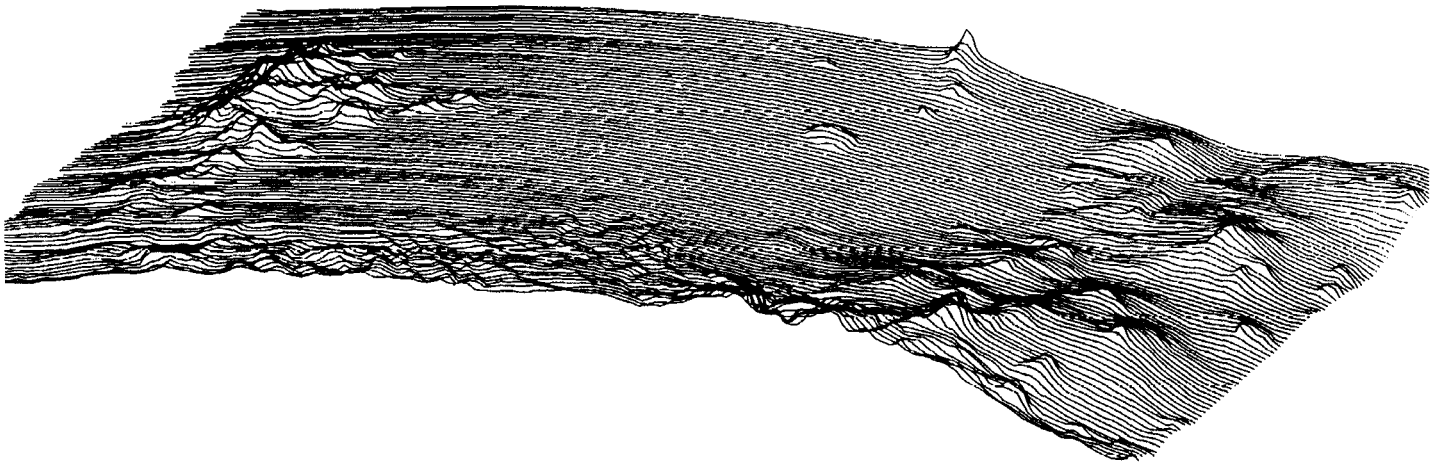


Figura 18. El MDT de la Ciudad de México representado por cadenas de la NCD. Esta malla discreta 3D representa el resultado de haber sumado un pequeño cambio de dirección, a la malla representada en la Figura 17.

2.6 Tamaño variable de segmento en función del cambio de dirección

En esta parte se introduce un nuevo concepto, se hace variar el tamaño del segmento como una función del cambio de dirección y se define como:

$$l = 1 - |a_i|$$

si la forma contiene alta curvatura los tamaños de los segmentos disminuirán; cuando el cambio de dirección está al máximo valor el tamaño del segmento será cero. El principal objetivo de esta teoría es el de mejorar la representación de la curva, reconfigurándose a las necesidades de la misma. En operaciones aritméticas entre formas, en algunas operaciones se puede exceder el valor del cambio de dirección, es decir, que con el acarreo se excede el valor máximo permitido de 1 o -1 , con el uso del segmento variable se elimina este problema [21]. La Figura 19 muestra la NCD con segmento variable. La Figura 19(a) despliega la curva correspondiente a $l = 1 - |a_i|$; la 19(b) una curva de ejemplo; la 19(c) la determinación de los cambios de dirección y los tamaños de los segmentos; y finalmente, la 19(d) la curva discreta de 19(b). Se nota como los tamaños de los segmentos se reconfiguran a la curva aumentando la precisión donde más se necesita.

2.7 La Notación de Curvatura Discreta como una gramática

El análisis de forma presentado en esta sección se basa en el uso de la notación de *curvatura discreta* (Bribiesca [1]) presentada en el trabajo de: Bribiesca, Rosenblueth y Garza [2]. Este análisis de forma se basa en el uso de técnicas gramaticales. El mapeo que se realiza permite hacer invariante en traslación, rotación y escala a cualquier forma. El objetivo de estas técnicas es poder hacer clasificación de formas 2D. La representación y reconocimiento de la forma de los objetos es un área de investigación importante en visión por computadora. Una de las ventajas más relevantes en la descripción de forma por medio de su contorno es la reducción de información, sin

perder precisión. Algunas referencias clásicas en esta área son las de Pavlidis [18], Ballard [3] y Otterloo [23], otra clásica en relación a un enfoque sintáctico es la de Fu [24].

En esta sección se propone un método para analizar propiedades locales y globales de forma. Así, cada curva es mapeada a una representación invariante en traslación, rotación y escala. Posteriormente este polígono es traducido a una cuerda de elementos, donde cada elemento indica el cambio de dirección sobre la curva. Finalmente, se usa una gramática para poder clasificar esos elementos. Originalmente la notación de curvatura discreta genera valores continuos, lo cual hace imposible el uso de técnicas gramaticales. Por lo tanto, en esta parte primero se discretizan esos elementos y posteriormente se aplica la gramática.

Para hacer invariantes las formas en rotación y origen de construcción, existen diferentes métodos entre los principales destacan el uso de los ejes (mayor, menor y principales) y el de centroides y momentos. El uso de los ejes principales [25] de la forma permite tener un origen único y un solo eje, lo cual facilita la invarianza en rotación. Así, la discretización se hace a cinco valores discretos de cambio de dirección: -120° , -60° , 0° , 60° y 120° , representados respectivamente, por: -2 , -1 , 0 , 1 y 2 . Lo anterior genera un patrón regular sobre el plano, es decir, un mosaico y permite el uso de técnicas gramaticales. Para propósitos prácticos se elimina el cambio de dirección de 180° ya que esto generaría polígonos no simples [26].

2.7.1 Detección de características por medio de una gramática

Considerando formas ya invariantes en traslación, rotación, escala y representadas por medio de cuerdas, se ilustra como los problemas de detección de forma se convierten en problemas sintácticos. Los problemas que se presentan a continuación corresponden a ejemplos generados en forma sintética, su adaptación al mundo real debe ser enriquecida con más parámetros.

1. **Diamantes.** Este es un problema simple de detección de propiedades locales de forma. Por ejemplo, reconocimiento de diamantes con dos ángulos internos de 60° y también con dos ángulos de 120° . El método asocia tales formas con un lenguaje formal: $L = (0^j 1 0^n 2 0^m 1 0^n 2 0^{m-j} : j, n, m \geq 0)$. Así, reconociendo cuerdas en L , es posible detectar tales diamantes. Este lenguaje es estrictamente sensible al contexto [27], por lo tanto no se pueden usar analizadores para lenguajes libres de contexto. Sin embargo, el lenguaje de programación Prolog [28] permite un diseño fácil de cualquier analizador usando gramáticas de cláusulas definidas.

2. **Formas con dos o más ángulos de 120° .** Una variante del ejemplo mencionado anteriormente consiste en la detección de formas con dos o más ángulos de 120° . En este caso, son las cuerdas con dos o más ocurrencias del símbolo 2. El lenguaje es: $C_{i=1}^n \alpha_i$ donde $\alpha_i = 2$ por lo menos para dos valores distintos de i .

3. **Ocurrencia de una forma dentro de otra.** Un problema que puede existir en la práctica es saber si parte de una forma existe en otra. El método traslada este problema a uno de comparación de cuerdas, en la referencia [29] se muestra el algoritmo en forma eficiente.

4. **Múltiples ocurrencias de cualquier forma dentro de otra.** Otro problema interesante en reconocimiento de patrones es el de determinar si una forma está formada por múltiples ocurrencias de otra. Así, α^n para $n \geq 2$, lo cual corresponde a encontrar esas formas que tienen n simetría rotacional. Como en el problema 1, este lenguaje no es de contexto libre, pero si es posible escribir un programa en Prolog para poner en marcha dicho analizador.

5. **Formas simétricas en espejo.** El analizador del lenguaje $\alpha\alpha^R$ detecta todas las formas simétricas en espejo, donde α^R es α escrita al contrario y con signo opuesto.

A continuación se presenta el código en Prolog para algunos de los ejemplos mostrados con anterioridad.

1. Diamantes.

```
diam(S) :- rot(S,T), diam1(T, []).  
diam1 --> [2], zero(N), [2], zero(M), [2], zero(N), [2], zero(M).  
zero(0,X,X).  
zero(s(N)) --> [0], zero(N).  
rot(X,Y) :- append(U,V,X), append(V,U,Y).  
append([],Y,Y).  
append([W|X],Y,[W|Z]) :- append(X,Y,Z).
```

2. Formas con dos o más ángulos de 120°.

```
two(S) :- rot(S,T), two1(T, []).  
two1 --> [2], ..., [2], ... .  
... --> [] .  
... --> [], ... .
```

3. Ocurrencia de una forma dentro de otra.

```
substr(S) :- rot(S,T), substr1(T, []).  
substr1 --> [2], [-1], [0], [-1], ... .
```

4. Formas simétricas en espejo.

```
twofold(S) :- append(X,X,S).
```

2.7.2 Resultados

Para ilustrar las capacidades del método propuesto, se presentan algunos ejemplos de detección de forma en un universo discreto de formas. La Figura 20 muestra un subconjunto de formas discretas cerradas usando solamente 14 cambios de dirección.

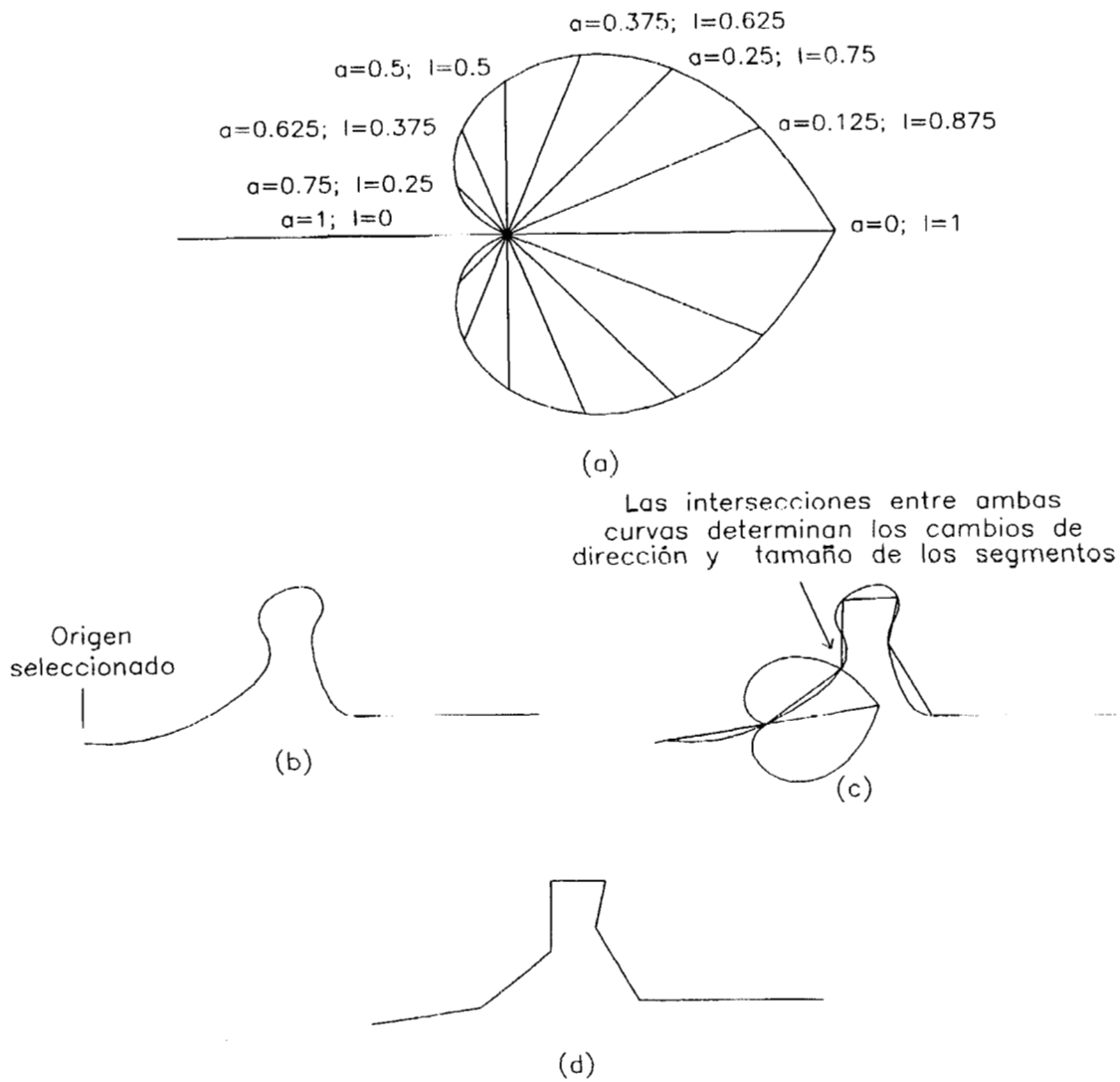


Figura 19. La notación de curvatura discreta usando segmentos de longitud variable: (a) la curva de $l = 1 - |a_i|$; (b) ejemplo; (c) determinación de los tamaños de los segmentos y de los cambios de dirección; (d) la curva discreta del ejemplo de (b).

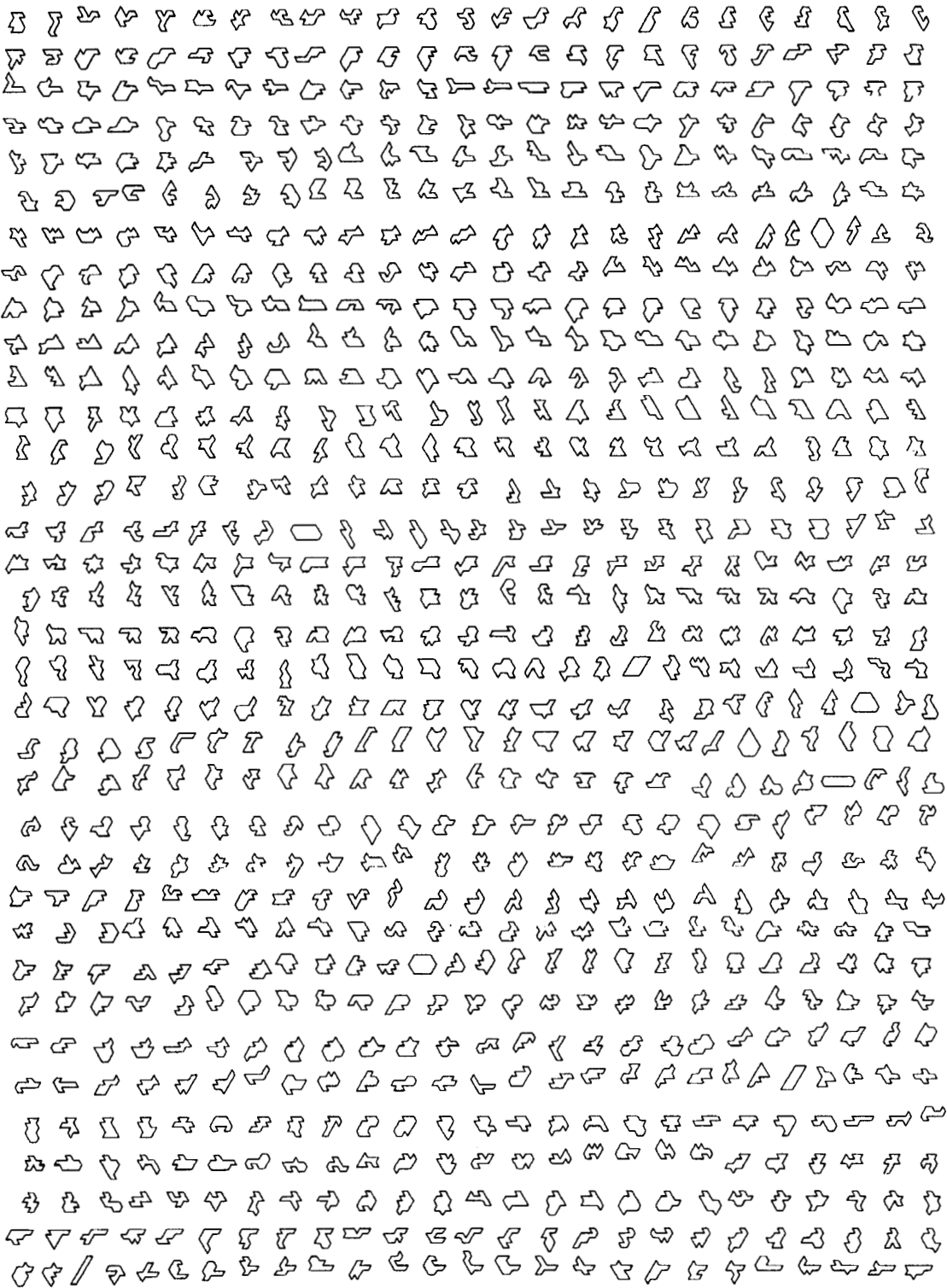


Figura 20. Parte del universo discreto de formas cerradas, sin cruces internos, usando solamente 14 cambios de dirección.

La Figura 21(a) muestra todas las formas de la Figura 20 que contienen las formas representadas por la siguiente subcuerda: $-1\ 2\ 1\ 1\ 0$. Este ejemplo fue seleccionado como un intento para identificar todas las formas que tienen cabeza de pájaro.

La Figura 21(b) presenta todas las formas de la Figura 20 que tiene las siguientes subcuerdas: $-1\ 2\ 1\ 1\ 0$ o $-2\ 2\ 1\ 1\ 1$. Este ejemplo ilustra las formas que tienen la característica de la Figura 21(a) o una pequeña variante que también se asemeja a una cabeza de pájaro.

La Figura 21(c) muestra las formas que consisten de dos ocurrencias de la misma subcuerda.

La Figura 21(d) muestra todas las formas de cinco picos.

La Figura 21(e) muestra las formas en nuestro universo que tienen el mayor número de símbolos iguales a la cuerda siguiente: $0\ 1\ -2\ 1\ 2\ 0\ 0\ 1\ 2\ -2\ 2\ -2\ 1\ 2$. Este es un intento de hacer *shape matching*. En la Figura 21(e) de lado izquierdo se muestra la forma como se presenta en la Figura 20, y del lado derecho se rotó para mayor facilidad en la identificación.

La Figura 21(f) muestra todas las formas que sólo contienen símbolos positivos. Lo anterior genera todas las formas convexas de la Figura 20.

La Figura 21(g) presenta todas las formas que contienen la siguiente cuerda: 0^n donde $n \geq 4$. Estas formas corresponden a las que tienen los lados rectos más largos.

La Figura 21(h) muestra todas las formas de la Figura 20 que contienen cuerdas con cinco elementos contiguos cuya suma es mayor o igual a seis. Esto corresponde a las formas parecidas a la “U”.

La Figura 21(i) muestra todas las formas con cuerdas que corresponden a las mostradas en la Figura 21(c) y que además tienen seis picos. Este tipo de formas pueden ser de interés en algunas aplicaciones reales, donde una parte de la forma se repite periódicamente.

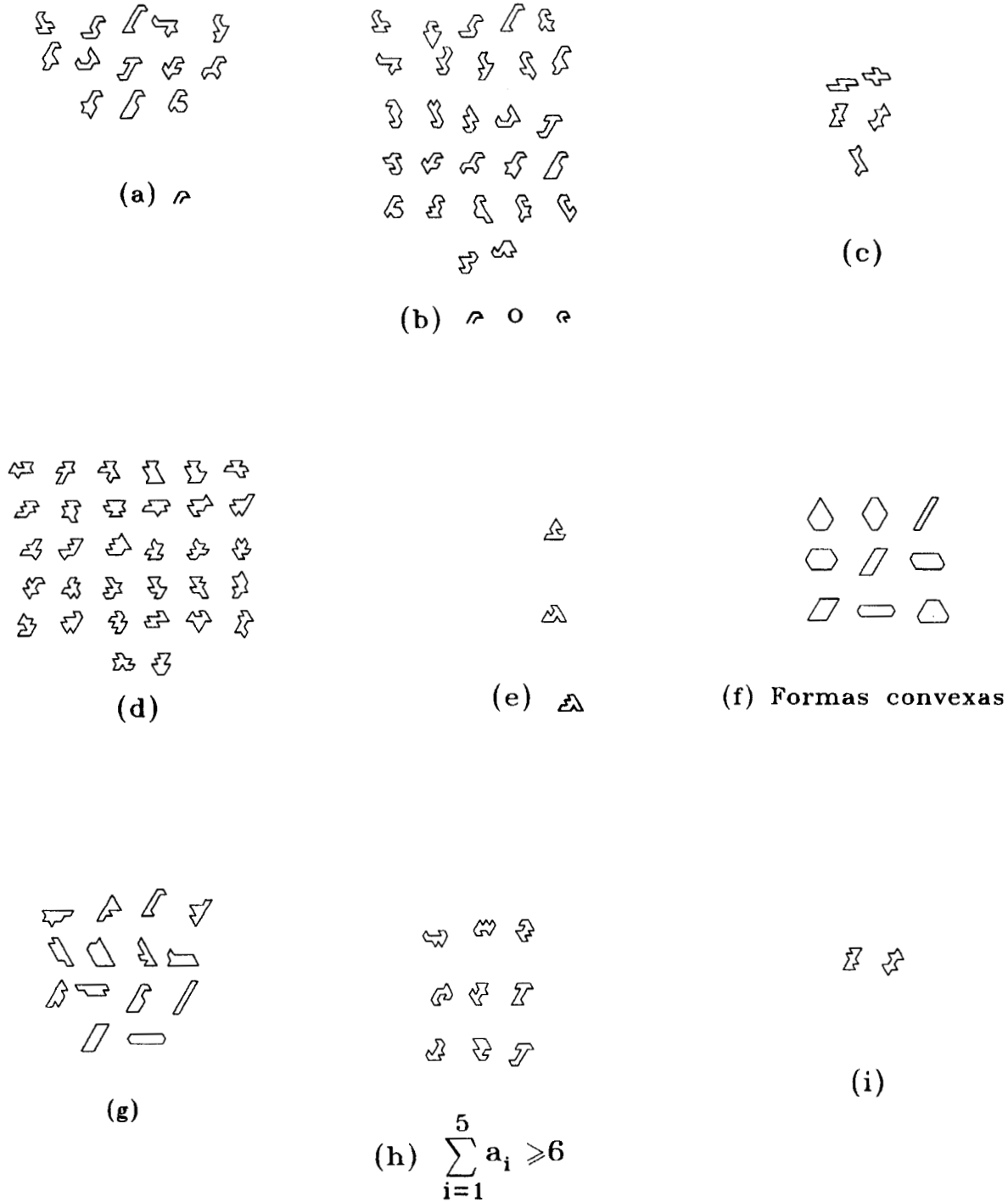


Figura 21. Resultados: (a) cabeza de pájaro; (b) dos diferentes tipos de cabeza de pájaro; (c) formas de simetría doble rotacional; (d) formas con cinco picos; (e) un ejemplo de *shape matching*; (f) formas convexas; (g) formas con lados rectos grandes; (h) formas concavas; (i) formas iguales a 21(c) con seis picos.

3 Descripción de Objetos 3D

La estructura geométrica utilizada es por medio de *voxels*. Un *voxel* $v(r,c,s)$ es una unidad discreta de volumen, representado geoméricamente por un cubo, localizado por medio de sus coordenadas (renglón, columna, capa), el cual puede ser marcado con materia o sin materia. Así, un objeto es representado por un arreglo tridimensional de unos y ceros, donde cada elemento del arreglo representa un *voxel*. El arreglo constituye una *imagen binaria 3D* denotada por $I(r, c, s)$. Los objetos son considerados sólidos de densidad constante y están representados por coordenadas Cartesianas de su centro.

Una parte importante en este trabajo es la consideración de que una *entidad* ha sido aislada del mundo real. Esta es llamada el *objeto* y es el resultado de un proceso previo: los objetos están formados por *voxels*. La Figura 22 muestra una esfera representada por *voxels* a diferentes resoluciones. Estos objetos fueron obtenidos como el resultado de un proceso previo y están representados como arreglos tridimensionales de celdas (*voxels*). Una ventaja importante de esta notación es la obtención de cortes de los objetos con gran facilidad.

3.1 Método de graficación de objetos representados por *voxels*

La esfera de mayor resolución de la Figura 22 representada por *voxels* está compuesta por más de 12,000 de ellos. Lo anterior representa un problema para la representación gráfica de los objetos: considerando que cada *voxel* está acotado por seis planos en 3D, se tienen más de 72,000 planos para representarlos en forma gráfica. El cálculo para realizar el ocultamiento de las líneas es bastante grande y como consecuencia es un proceso lento. Un camino para reducir el tiempo de procesamiento al graficar estos objetos es eliminar los *voxels* ocultos, es decir, todos los *voxels* que tienen vecinos alrededor de ellos deben ser eliminados: Se elimina un *voxel* $v(x, y, z)$ si todos los

seis *voxels* $v(x \pm 1, y \pm 1, z \pm 1)$ pertenecen al objeto. Es importante hacer notar que este método es usado sólo como una técnica para la representación gráfica del objeto en cuestión. El método propuesto de reconocimiento considera toda la cantidad de información que compone al objeto.

La Figura 23(a) muestra un objeto irregular representado por *voxels*. La Figura 23(b) despliega ese mismo objeto con un corte en la parte superior, mostrando una pared de un grosor de un *voxel*, aquí todos los *voxels* ocultos fueron eliminados. Por supuesto dependiendo de la forma del objeto, el método propuesto será más o menos rápido. Una mejora adicional a este método consiste en la eliminación de las caras ocultas, así, la pared del objeto tendrá un grosor de un plano.

Ambas formas de graficación son válidas para objetos conteniendo hoyos internos. La Figura 23(c) muestra una pirámide y la 23(d) la misma, con un corte en la parte superior, mostrando una pared de un grosor de un plano.

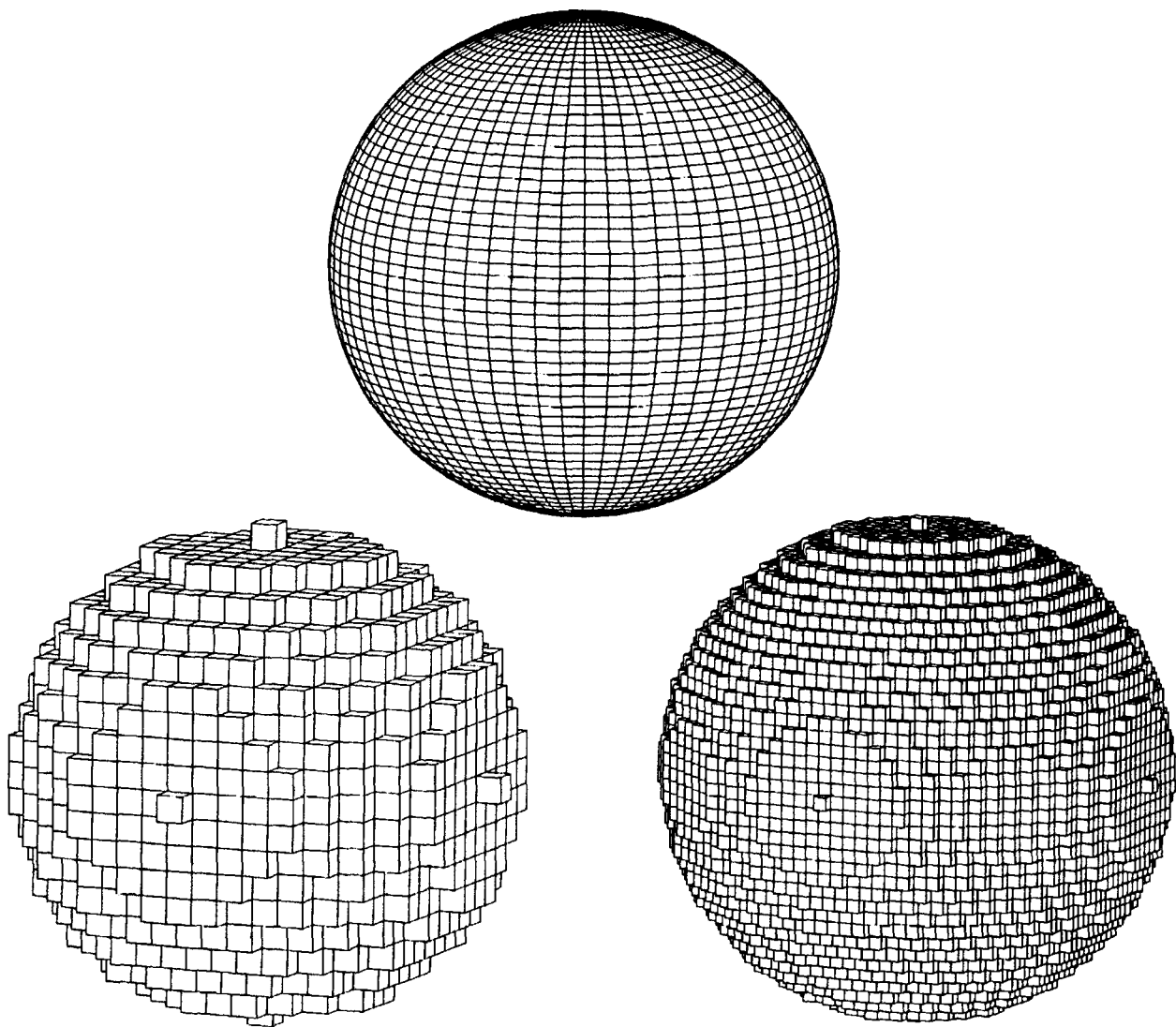


Figura 22. Esferas representadas por medio de *voxels* a diferentes resoluciones. Cada objeto está representado por un arreglo tridimensional de ceros y unos, donde los unos indican presencia de materia.

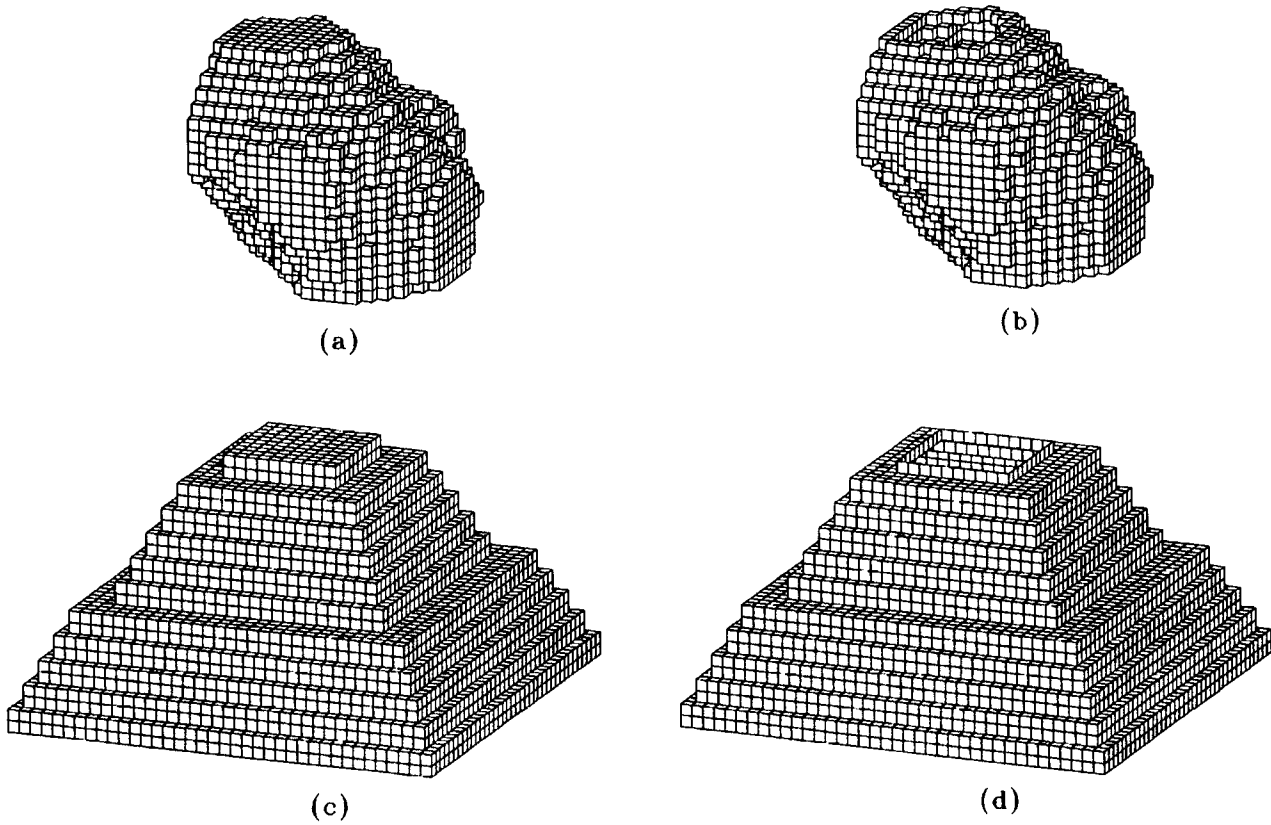


Figura 23. Técnicas para graficar objetos 3D representados por *voxels*: (a) un objeto sólido irregular; (b) el mismo objeto de (a) con un corte en la parte superior, mostrando una pared de un grosor de un *voxel*; (c) una pirámide sólida; (d) la pirámide con un corte en la parte superior mostrando una pared de un grosor de un plano. Note que estas representaciones son sólo para graficar, el método propuesto de reconocimiento considera toda la información de la que están compuestos los objetos.

4 Invariantes en Traslación y en Rotación

El problema de invariantes en *Reconocimiento de Patrones* se distingue por su alto grado de complejidad, por lo tanto no es de sorprender la gran variedad de técnicas y métodos que han sido desarrollados. Varios autores han usado diferentes técnicas para realizar los invariantes de reconocimiento de patrones, entre los clásicos destacan los trabajos de Bracewell [30] y Brigham [31] en relación a la transformada de Fourier, la cual produce invarianza en rotación y escala (dilataciones). Un resumen interesante de métodos de transformada de Fourier aplicada a los invariantes en reconocimiento de patrones aparece en Wechsler [32]. Hu [33] describe el reconocimiento de patrones visuales con base en invariantes por momentos, los cuales son obtenidos al tomar los cocientes y exponentes de los *momentos*.

Antes de hacer cualquier tipo de análisis de semejanza entre objetos es necesario hacerlos invariantes bajo transformaciones geométricas. En este capítulo se presentan algunos de estos métodos de uso actual basados en *ejes* y en *momentos*. Finalmente, se propone un método discreto que resuelve el problema de invarianza para formas complicadas, simétricas o aberrantes. Para probar los métodos y técnicas propuestas se utilizaron tres diferentes objetos de prueba. Estos fueron obtenidos por procedimientos previos y están representados por arreglos tridimensionales, donde cada elemento del arreglo representa un *voxel*. La Figura 24 nos muestra los objetos de prueba, la Figura 24(a) despliega el objeto *A*, la 24(b) el *B* y la 24(c) el *C*. Es importante notar que estos objetos difieren en: orientación, escala y cantidad de información (número de *voxels*) para ser descritos.

4.1 Invarianza en traslación

La *traslación de un objeto* es una transformación rígida lineal de \mathbb{R}^3 . Una traslación mueve cada *voxel* del objeto en la misma dirección y distancia [34]; esto es, si los *voxels* v_1 y v_2 son llevados a los *voxels* v'_1 y v'_2 respectivamente, entonces las distancias

v_1v_2 y $v'_1v'_2$ son iguales. En algunos casos, el origen trasladado se hace corresponder al *centroide* del objeto. Donde el *centroide* $(\bar{r}, \bar{c}, \bar{s})$ [35] de un volumen V (de densidad constante) es el centro de masa del volumen. La posición (renglón, columna, capa) del centroide, considerando todos los *voxels* del volumen, está dada por:

$$\bar{r} = \frac{1}{\#V} \sum_{(r,c,s) \in V} r, \quad \bar{c} = \frac{1}{\#V} \sum_{(r,c,s) \in V} c, \quad \bar{s} = \frac{1}{\#V} \sum_{(r,c,s) \in V} s.$$

En las secciones siguientes se verá que el origen es trasladado a la intersección entre el eje mayor y menor del objeto, los cuales serán definidos posteriormente. Así, se obtiene la invarianza. O sea, se normaliza con respecto a traslación en x, y, z .

4.2 Invarianza en rotación

La *rotación de un objeto* es una transformación rígida de \mathfrak{R}^3 para la cual un eje de rotación L es definido. Así, cada *voxel* del objeto es rotado alrededor de L en un mismo sentido y una misma magnitud [34]. Hay muchos métodos de rotación de objetos 3D los cuales son encontrados en textos de fotogrametría, especialmente en problemas de orientación relativa. Varios autores han usado la representación por medio de cuaternios para simplificar problemas de orientación absoluta, entre ellos: [36], [37], [38], [39] y [40]. Aunque los objetos presentados en el contenido de esta tesis tienen una representación discreta (están representados por *voxels*), las rotaciones efectuadas a ellos no son en un dominio discreto, se giran considerando valores reales y la voxelización (la cual se presenta más adelante) se orienta en el sistema discreto de coordenadas ya rotado de los *voxels*.

4.2.1 Solución estándar

Sea ω el ángulo de rotación alrededor del eje x , ϕ alrededor del y y κ alrededor de z , respectivamente. Entonces, si $R = R(\omega, \phi, \kappa)$ es la matriz de rotación de tamaño

3×3 , se tiene que $R(\omega, \phi, \kappa) = R(\kappa)R(\phi)R(\omega)$, donde:

$$R(\omega) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \omega & \sin \omega \\ 0 & -\sin \omega & \cos \omega \end{pmatrix}$$

$$R(\phi) = \begin{pmatrix} \cos \phi & 0 & -\sin \phi \\ 0 & 1 & 0 \\ \sin \phi & 0 & \cos \phi \end{pmatrix}$$

$$R(\kappa) = \begin{pmatrix} \cos \kappa & \sin \kappa & 0 \\ -\sin \kappa & \cos \kappa & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

por lo tanto

$$R(\omega, \phi, \kappa) =$$

$$\begin{pmatrix} \cos \phi \cos \kappa & \sin \omega \sin \phi \cos \kappa + \cos \omega \sin \kappa & -\cos \omega \sin \phi \cos \kappa + \sin \omega \sin \kappa \\ -\cos \phi \sin \kappa & -\sin \omega \sin \phi \sin \kappa + \cos \omega \cos \kappa & \cos \omega \sin \phi \sin \kappa + \sin \omega \cos \kappa \\ \sin \phi & -\sin \omega \cos \phi & \cos \omega \cos \phi \end{pmatrix}. \quad (1)$$

Para representar las relaciones entre la matriz de rotación y para evitar grandes expresiones trigonométricas, se establece la siguiente convención:

$$R(\omega, \phi, \kappa) = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}. \quad (2)$$

Los ángulos ω , ϕ y κ se pueden obtener directamente de los valores de r_{ij} :

$$\begin{aligned} \sin \phi &= r_{31}; \\ \tan \omega &= \frac{(-r_{32})}{r_{33}}; \\ \tan \kappa &= \frac{(-r_{21})}{r_{11}}. \end{aligned}$$

La ecuación (1) presenta la matriz de rotación $R(\omega, \phi, \kappa)$ cuando se conocen ω , ϕ y κ . Otra forma de obtener la matriz de rotación es por medio del uso de cuaternios, los cuales permiten una fácil programación.

4.2.2 Representación por cuaternios

Schut [36] presenta varias notaciones para obtener la matriz de rotación, entre ellas está la notación por cuaternios y muestra que para cualquier matriz antisimétrica

$$S = \begin{pmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{pmatrix}$$

una matriz de rotación R puede ser construida. Así, para cualquier escalar d , se puede observar que R depende de d y se define como:

$$R = (dI + S)(dI - S)^{-1} \quad (3)$$

La definición anterior garantiza que $R'R = I$:

$$\begin{aligned} R'R &= [(dI + S)(dI - S)^{-1}]'[(dI + S)(dI - S)^{-1}] \\ &= [(dI - S)^{-1}]'(dI + S)'(dI + S)(dI - S)^{-1} \\ &= (dI + S)^{-1}(dI - S)(dI + S)(dI - S)^{-1} \\ &= [(dI + S)^{-1}(dI + S)][(dI - S)(dI - S)^{-1}] \\ &= I. \end{aligned}$$

Si

$$S = \begin{pmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{pmatrix}$$

y usando la ecuación (3), se tiene

$$R = \begin{pmatrix} d^2 + a^2 - b^2 - c^2 & 2(ab - cd) & 2(ac + bd) \\ 2(ab + cd) & d^2 - a^2 + b^2 - c^2 & 2(bc - ad) \\ 2(ac - bd) & 2(bc + ad) & d^2 - a^2 - b^2 + c^2 \end{pmatrix} \frac{1}{a^2 + b^2 + c^2 + d^2}$$

Schut [36] demuestra que usando la representación por cuaternios: la matriz de rotación en 3D puede ser determinada a partir del conjunto de puntos 3D correspondiente, por medio de la solución de un sistema de ecuaciones lineales. Entonces Pope [37] usa la representación de cuaternios en la determinación de la matriz de rotación

en problemas de orientación absoluta. Hinsken [38] usa los cuaternios para simplificar los cálculos computacionales en la determinación de los parámetros de traslación y rotación también en problemas de orientación absoluta. En el trabajo aquí presentado para rotar objetos se usa la matriz de rotación de Hinsken [38] especificada por

$$R = \begin{pmatrix} d^2 + a^2 - b^2 - c^2 & 2(ab - cd) & 2(ac + bd) \\ 2(ab + cd) & d^2 - a^2 + b^2 - c^2 & 2(bc - ad) \\ 2(ac - bd) & 2(bc + ad) & d^2 - a^2 - b^2 + c^2 \end{pmatrix} \quad (4)$$

donde los parámetros a , b , c y d satisfacen la siguiente condición:

$$a^2 + b^2 + c^2 + d^2 = 1. \quad (5)$$

Otro trabajo importante para obtener la matriz de rotación derivado de los anteriores es propuesto por Horn [39], donde todavía se simplifica más la obtención de la matriz de rotación. En el trabajo aquí presentado se probaron diferentes métodos para hacer invariantes los objetos en rotación, entre los más simples fue la matriz de rotación presentada en el trabajo de Hinsken, aunque sí es importante mencionar que en la actualidad la mayoría de los paquetes gráficos de programación en 3D incluyen un comando de rotación de objetos.

4.2.3 Los ejes mayor y menor

Cuando se hace un objeto invariante en traslación y en rotación es necesario tener ejes o puntos de referencia robustos, que permitan un invariante fuerte al ruido. Existe un considerable número de métodos para seleccionar ejes únicos que sirvan como ejes de rotación. La gran mayoría han sido desarrollados para 2D, entre los principales destacan los ejes mayores, menores, los que pasan por el centroide del objeto, los ejes universales propuestos por Lin [25] y otros. El método propuesto en este trabajo, es del eje mayor y menor debido a las ventajas que presenta en objetos irregulares.

El *eje mayor* a de un objeto en 3D es la línea que une los dos *voxels* más alejados que pertenecen al objeto, medida a partir de los centroides de los *voxels* [22]. Para

objetos simétricos o aberrantes pueden existir varios ejes mayores: en este caso se deberán agregar más criterios en la selección del eje. La Figura 25(b) muestra el eje mayor del objeto representado en la Figura 25(a), es importante notar que en la Figura 25(b) el objeto parece hueco, esto se hace para poder observar el eje mayor, pero en realidad éste es un objeto sólido.

El *eje menor* b es la línea perpendicular al eje mayor que une éste con el *voxel* más lejano (la medida se toma a partir del centroide del *voxel*). La Figura 25(b) muestra el eje menor perpendicular al eje mayor. El eje menor tampoco es necesariamente único.

Finalmente, con base en la matriz de rotación de la ecuación (4) presentada por Hinsken [38], y usando como punto de referencia la intersección entre los ejes mayor y menor y éstos como ejes de rotación: la Figura 25 muestra la invarianza en rotación. La Figura 25(a) despliega un objeto con cualquier orientación, éste es un objeto sólido compuesto por *voxels*. La Figura 25(b) muestra la intersección de los ejes mayor y menor. La Figura 25(c) muestra las dos rotaciones: la primera, mostrada en la Figura 25(d); y la segunda, en la Figura 25(e). Por último, la Figura 25(f) muestra el objeto ya invariante en rotación.

4.3 Invarianza en traslación y rotación usando máxima correlación

En esta sección se presenta el método de máxima correlación, el cual resuelve el problema de invariantes en rotación cuando los objetos a rotar son simétricos o tienen formas aberrantes. El método propuesto se presenta para las formas de los objetos en 2D, es decir, trataremos con formas homeomórficas al disco D^2 . Se usaron tres diferentes formas de prueba representadas en la Figura 26: la 26(a) despliega la forma s_1 , la 26(b) la s_2 y la 26(c) la s_3 , respectivamente. Las formas mencionadas anteriormente difieren en orientación y tamaño.

4.3.1 Invarianza bajo área

Una parte importante en el método de máxima correlación es hacer las diferentes formas invariantes en área. Las figuras 26(d), 26(e) y 26(f) representan las figuras 26(a), 26(b) y 26(c) respectivamente, invariantes en área. Lo anterior se obtuvo haciendo cambios de escala.

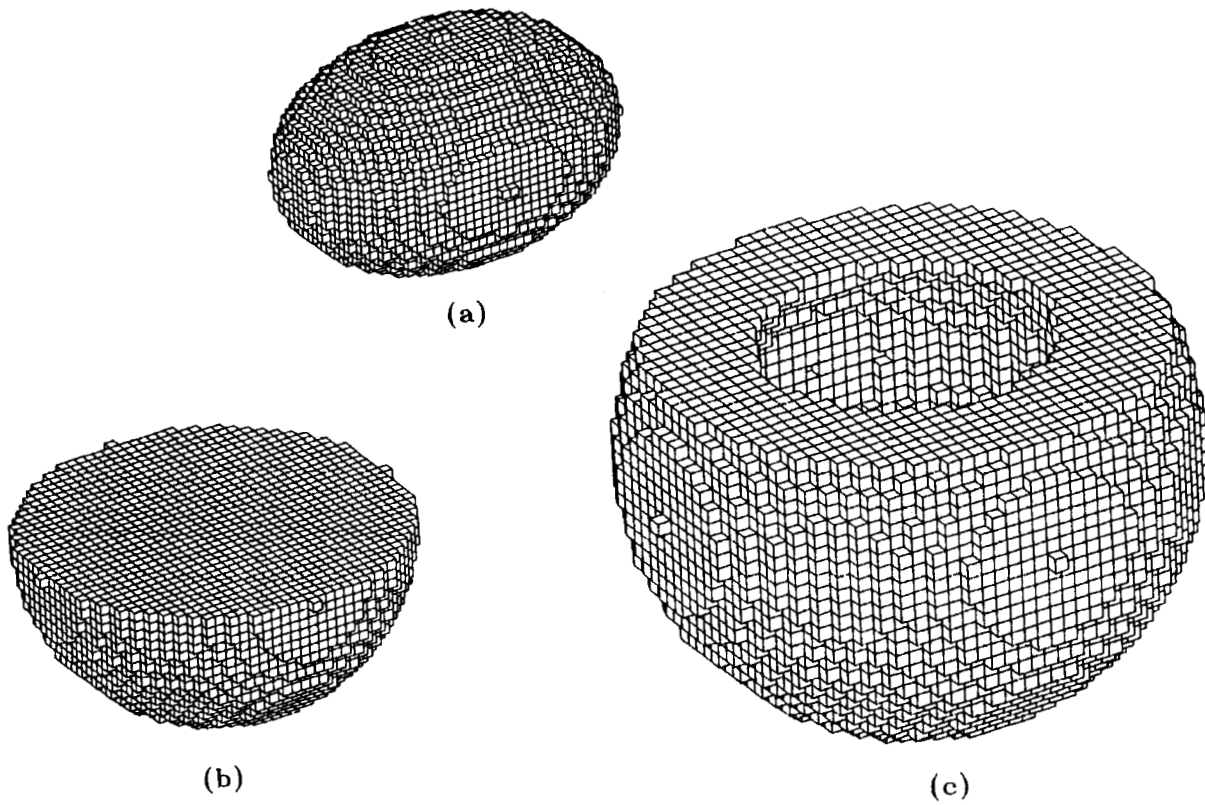


Figura 24. Objetos de prueba: (a) objeto *A*; (b) objeto *B*; (c) objeto *C*. Estos objetos difieren en: orientación, número de *voxels* y escala.

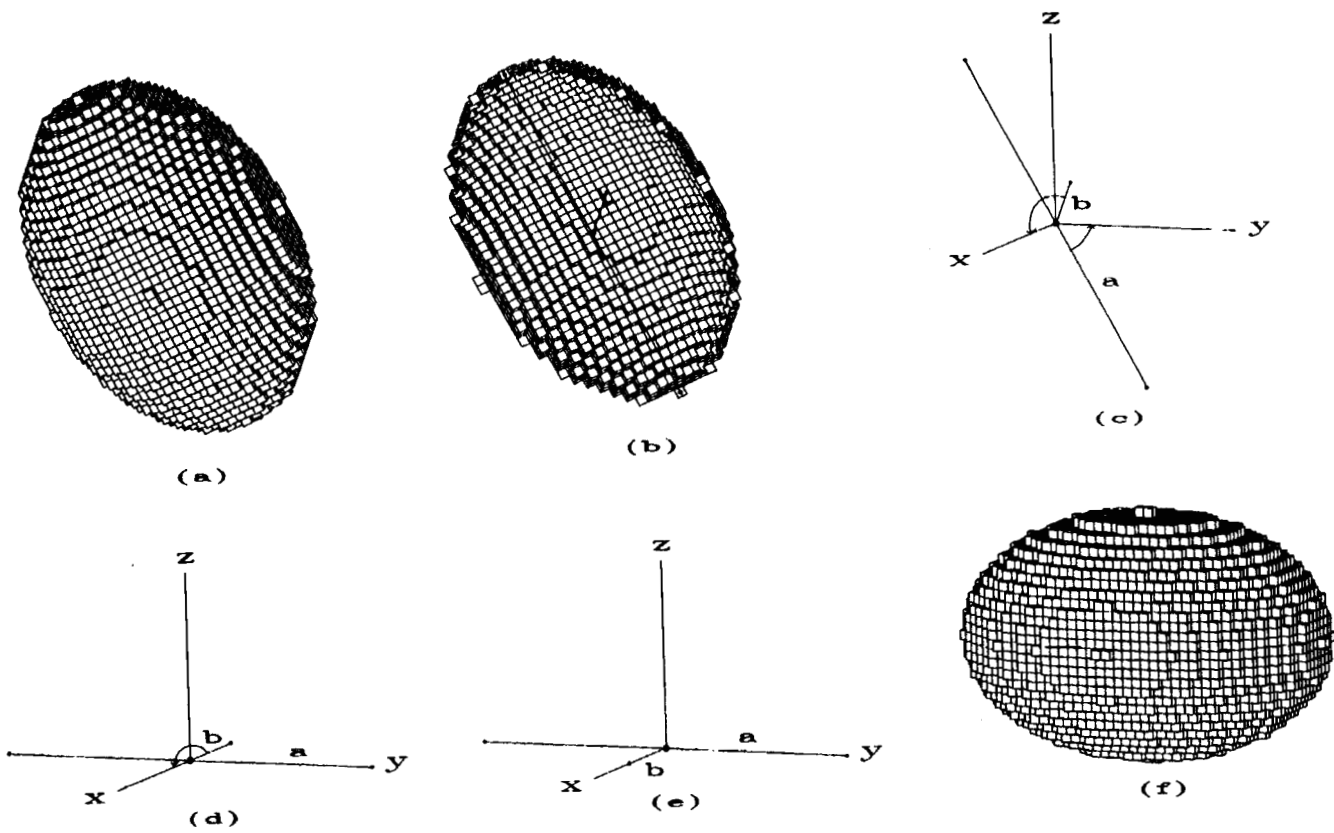


Figura 25. Invarianza en rotación: (a) el objeto a ser rotado; (b) ejes mayor y menor e intersección entre ambos; (c) las dos rotaciones a realizar; (d) primera rotación; (e) segunda rotación; (f) el objeto de (a), ya invariante en rotación.

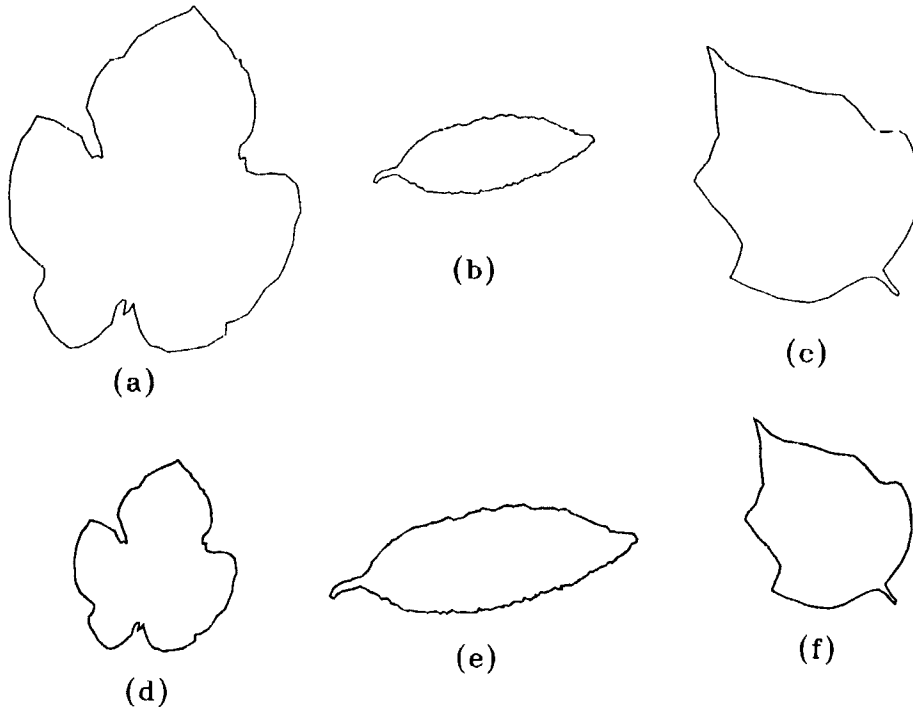


Figura 26. Formas 2D: (a) forma s_1 ; (b) forma s_2 ; (c) forma s_3 . Estas formas difieren en tamaño y posición. Las formas en (d), (e) y (f) corresponden a las formas de (a), (b) y (c), respectivamente, con la diferencia de que ya son invariantes en área.

4.3.2 Máxima correlación

El método de máxima correlación encuentra el máximo traslape entre dos formas a ser comparadas. El método consiste en calcular todas las traslaciones y rotaciones discretas entre las formas, es decir, una forma se mantiene fija, mientras la otra se rota y sobrepone a diferentes posiciones sobre la que está fija. En cada sobreposición se calcula el área de la superficie intersectada, el área máxima será el indicador del máximo traslape. El número de operaciones N esta dado por la siguiente expresión:

$$N = \Delta x \Delta y \frac{360^\circ}{\Delta \theta},$$

donde

Δx es el incremento en la dirección del eje x , Δy es el incremento en la dirección del eje y , y $\Delta \theta$ es el incremento en el ángulo. N es un valor entero, ya que el factor $\frac{360^\circ}{\Delta \theta}$ se hace entero poniendo a $\Delta \theta$ como submúltiplo de 360° . Δx y Δy , también se consideran como enteros.

El calcular todas las traslaciones y rotaciones discretas entre dos formas no siempre es lo más adecuado. En vez de usar este método se podrían usar otros métodos de invariantes en rotación, tales como por medio de los ejes o momentos de las formas. El eje mayor y menor [22], los ejes principales universales [25], entre otros los trabajos de invariantes de Bracewell [30], Brigham [31], Wechsler [32] y Hu [33]. Los métodos mencionados anteriormente garantizan la invarianza en rotación, pero no el mayor *matching* entre las formas, el cual si puede ser obtenido usando el método de máxima correlación.

El método de máxima correlación parece ser muy lento, pero con algunos trucos de programación es mejorado. En general, las rotaciones emplean más cálculo que las traslaciones, por esta razón: primero, se calcula una rotación; segundo, se calculan las traslaciones correspondientes a esa rotación. Las rotaciones usan funciones trigonométricas, las cuales son calculadas una sola vez y asignadas a unas variables,

posteriormente sólo se usa el valor de esas variables. Dependiendo de las características de las formas a ser analizadas, las variables Δx , Δy y $\Delta \theta$ pueden ser reducidas y, como consecuencia, el tiempo de proceso. Las ecuaciones para rotar una forma están dadas por:

$$x = x' \cos \theta - y' \sin \theta$$

$$y = x' \sin \theta + y' \cos \theta,$$

donde x y y corresponden a las coordenadas rotadas de la forma. El método de máxima correlación tiene importantes ventajas: garantiza la máxima correlación sin calcular ejes ni momentos; funciona para todas las formas independientemente de su tipo, ya sean aberrantes o simétricas y aproxima al óptimo los *pixels* comunes entre las formas comparadas, esto último es importante para la transformación de un objeto a otro como se verá más adelante. La Figura 27 despliega parte de las traslaciones y rotaciones discretas, en la parte inferior izquierda se muestran la formas s_1 y s_2 ; la forma s_2 es la que será girada y trasladada sobre s_1 , en cada paso se calcula el área de la superficie de traslape. La máxima área encontrada corresponde a las formas encerradas en un círculo, las cuales se localizan en la parte superior derecha de la gráfica. En algunos casos pueden existir varias áreas máximas iguales: la selección de cualquiera de ellas corresponde a la máxima correlación. Los valores asignados en este ejemplo para las variables son: $\Delta x = 17$; $\Delta y = 21$ y $\Delta \theta = 5^\circ$. La Figura 28 muestra los máximos traslapes entre las formas estudiadas, por el método de máxima correlación.

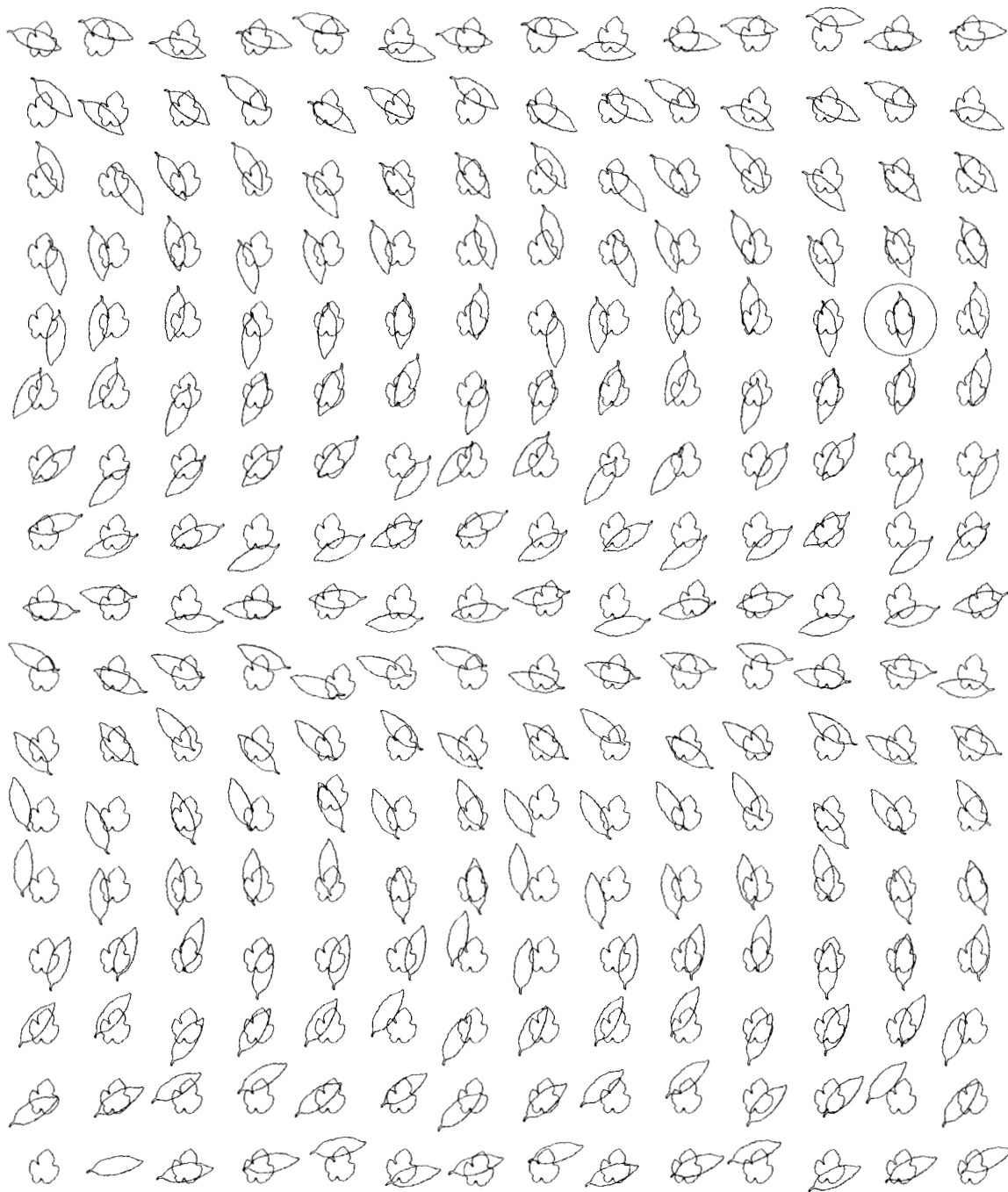


Figura 27. Parte de las traslaciones y rotaciones discretas entre las formas s_1 y s_2 . las formas marcadas con el círculo tienen la maxima correlacion.

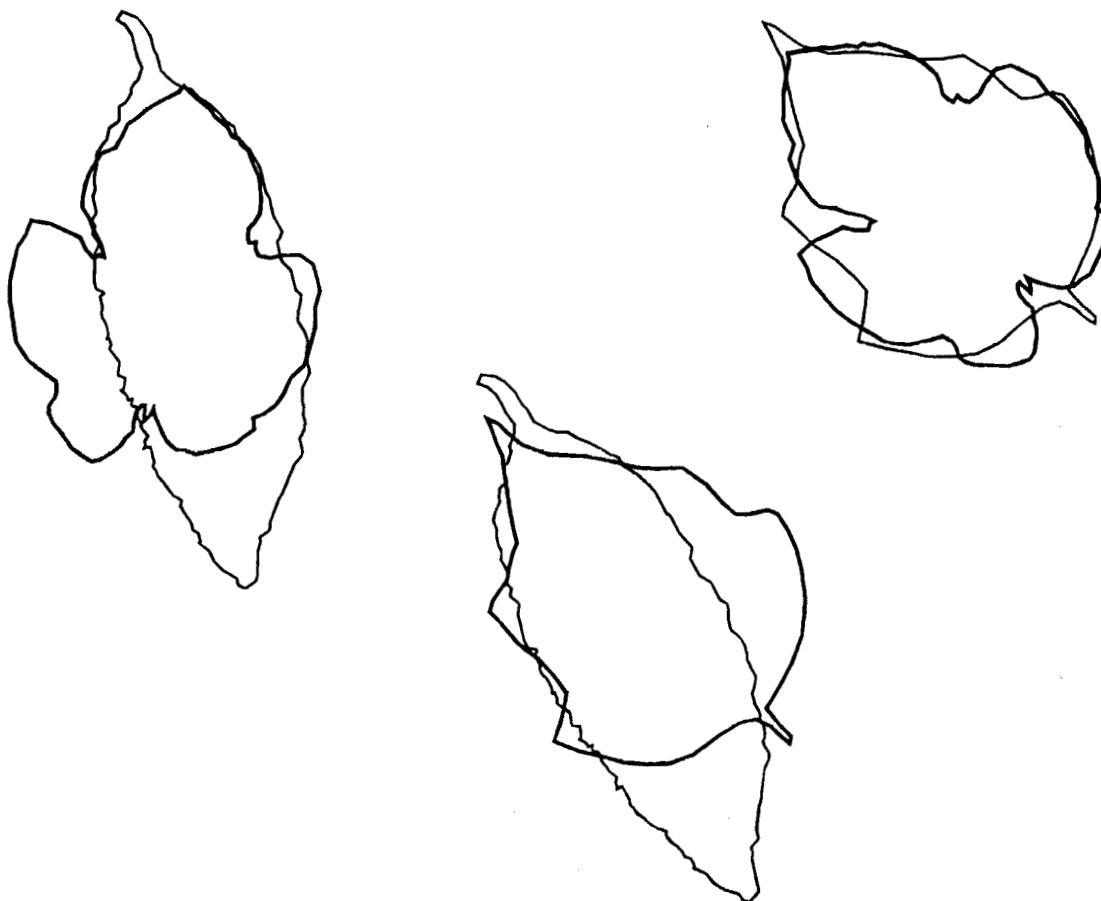


Figura 28. Máximos traslapes entre las formas estudiadas: s_1 , s_2 y s_3 .

5 Voxelización

La *voxelización* de un objeto corresponde a la normalización de ese objeto en relación a su volumen. Generalmente, los objetos están representados con diferentes cantidades de información. En el caso de la representación por *voxels*, el número de éstos que los representan es variable. Cuando se desea transformar de un objeto a otro se hace necesario que ambos tengan la misma cantidad de información. Por definición un objeto A representado por un número finito n de *voxels* podrá ser transformado en cualquier otro con n número de *voxels*. Un *universo discreto de objetos* de n *voxels* cada uno corresponde a un objeto “seis-conexo”, en terminología de la topología digital. La Figura 29 representa parte del universo de objetos compuestos por diez *voxels* cada uno, es importante hacer notar que estos objetos pueden ser ordenados por medio de su transformación, es decir, dado el objeto más compacto (cercano al cubo) se ordenan moviendo un voxel, posteriormente dos y así sucesivamente. Por medio del uso de la voxelización (la transformación de cualquier objeto a un número fijo de *voxels*) y de los universos discretos es posible analizar los objetos a un nivel de precisión constante.

La *voxelización* de un objeto se realiza por medio de los siguientes pasos:

1. Se *define* a n como *constante de normalización*, la cual representa el número de *voxels* del objeto. En los objetos que se estudian en este trabajo: $n = 6,000$.
2. Se *calcula* el volumen V del objeto, esto es el número de *voxels* por el volumen de cada uno. La Figura 30(a) muestra un objeto ya invariante bajo traslación y rotación compuesto por 13,768 *voxels*, si se considera que cada *voxel* tiene un volumen igual a 1, entonces el volumen del objeto es 13,768.
3. Se *selecciona* la intersección entre los ejes mayor y menor del objeto como el *origen*. La Figura 30(b) muestra la intersección entre los ejes mayor y menor.
4. Se *genera* una malla 3D a partir del origen, siguiendo la orientación de los ejes.

Donde la longitud l de cada lado de los cubos de la cuadrícula está dada por $\sqrt[3]{V_n}$. V_n es el volumen de cada cubo y se obtiene de la siguiente expresión: $V_n = \frac{V}{n}$. Los cubos de la cuadrícula corresponden a los nuevos *voxels* que definirán el objeto, los cuales serán llamados *voxels normalizados*. El tamaño de la cuadrícula debe ser calculado, considerando que el objeto será colocado dentro de ella.

5. Se *traslapa* la cuadrícula 3D y el objeto. Se calculan los *voxels* normalizados que pertenecen al objeto. Lo anterior se hace por medio del cálculo de las distancias entre los *voxels* del objeto y los normalizados, si la distancia de un *voxel* normalizado entre cualquier *voxel* del objeto es menor o igual que l , entonces ese *voxel* normalizado pertenece al objeto. La Figura 30(c) muestra la sobreposición del objeto y la cuadrícula.
6. Se *calcula el número de voxels normalizados* que pertenecen al objeto. Este número debe ser igual a la constante n . Pero, debido a varios factores entre los que destacan: el ruido del método (principalmente en lo que se refiere a la discretización), la forma del objeto, de los mismos *voxels* y en algunos casos la formación de huecos internos: el número de *voxels* normalizados puede ser diferente de n . En dado caso que el número de *voxels* normalizados sea menor que n , el programa incrementa el tamaño del objeto en una pequeña cantidad y recalcula otra vez los *voxels* normalizados; en caso contrario decrementa el tamaño del objeto. En algunos casos, este proceso es lento y tedioso. Finalmente, un proceso adicional “fuerza” a los *voxels* normalizados a ser iguales a n , esto es por medio de borrar o añadir *voxels* sin distorsionar mucho la forma original del objeto. La Figura 30(d) muestra los *voxels* normalizados que pertenecen al objeto. Finalmente, la Figura 30(e) despliega el objeto normalizado. Esta normalización fue realizada usando 6,000 *voxels*, el volumen de cada *voxel* V_n fue normalizado a la unidad para facilitar los procesos siguientes.

La Figura 31 muestra los objetos: A , B y C ya normalizados. Estos objetos corresponden a los de la Figura 24 en forma respectiva. Cada uno de los objetos de la Figura 31 están representados por 6,000 *voxels* cada uno y tienen la misma orientación con base en los invariantes utilizados o sea sus ejes. La conclusión de esta etapa permitirá la transformación de un objeto a otro, que es lo que se verá en el siguiente capítulo.

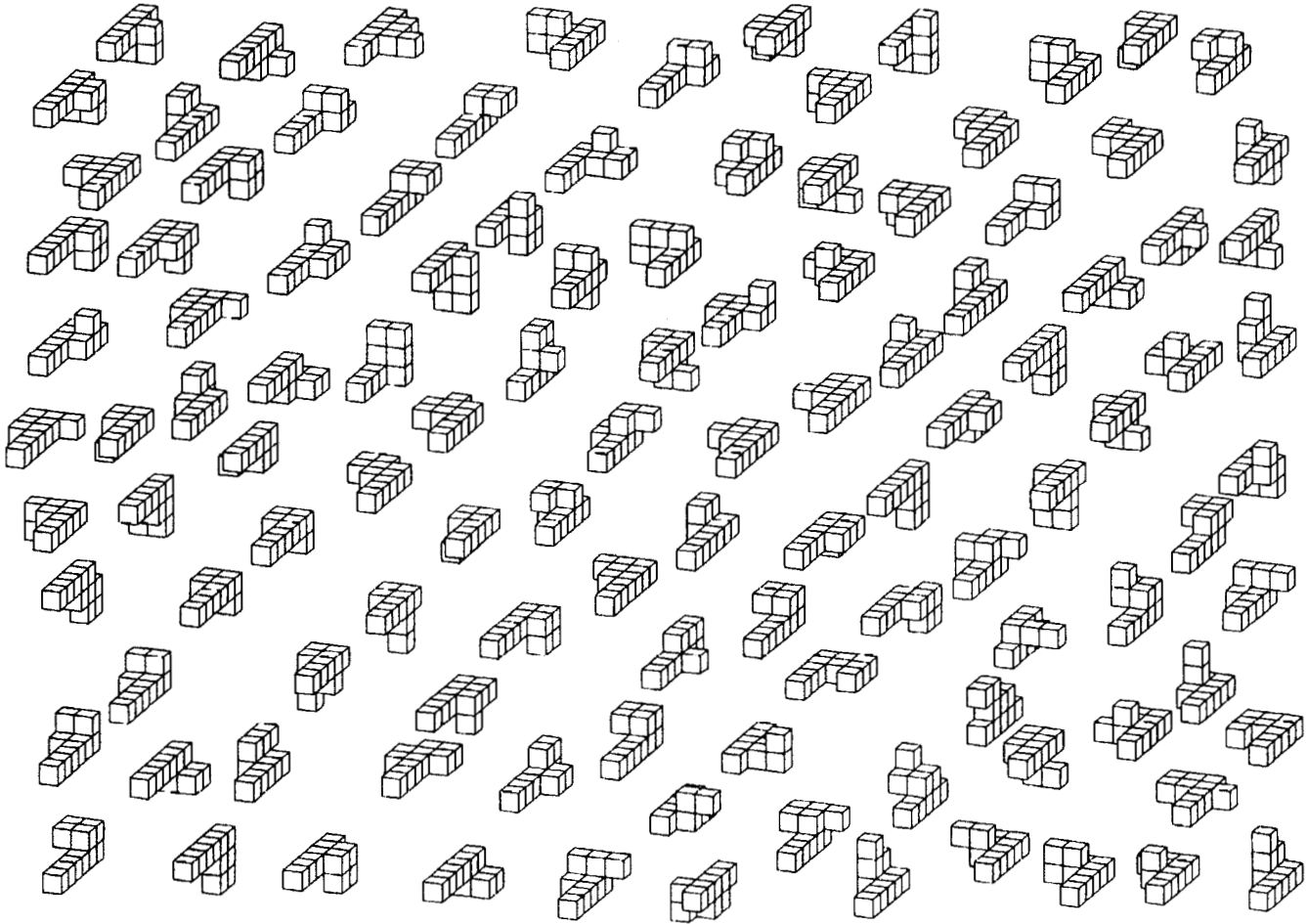


Figura 29. Parte del universo discreto de objetos compuestos por 10 *voxels* cada uno.

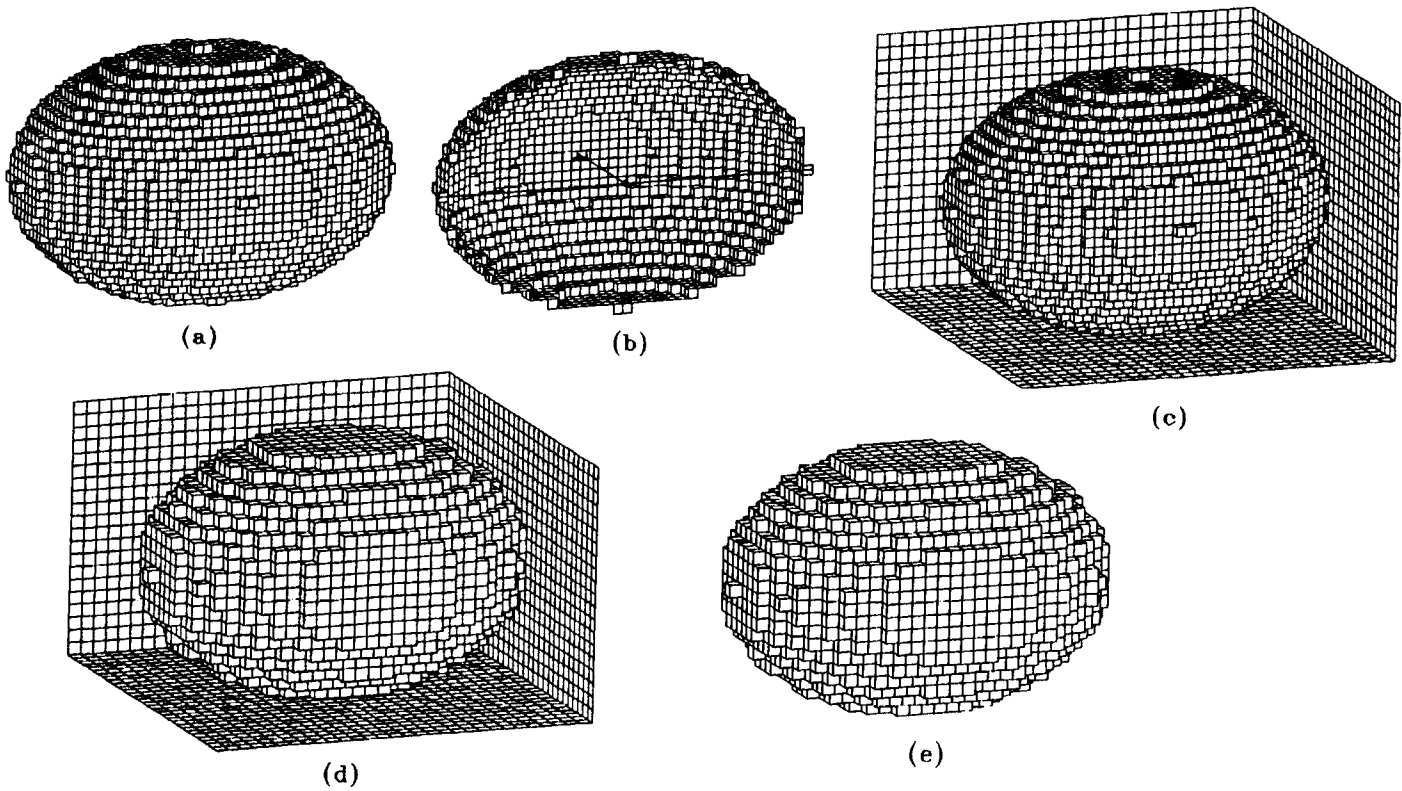


Figura 30. *Voxelización*: (a) Un objeto ya invariante bajo traslación y rotación; (b) Determinación del origen como la intersección entre el eje mayor y menor; (c) La sobreposición entre el objeto y la cuadrícula 3D; (d) *voxels* normalizados que pertenecen al objeto; (e) objeto normalizado representado por 6,000 *voxels*.

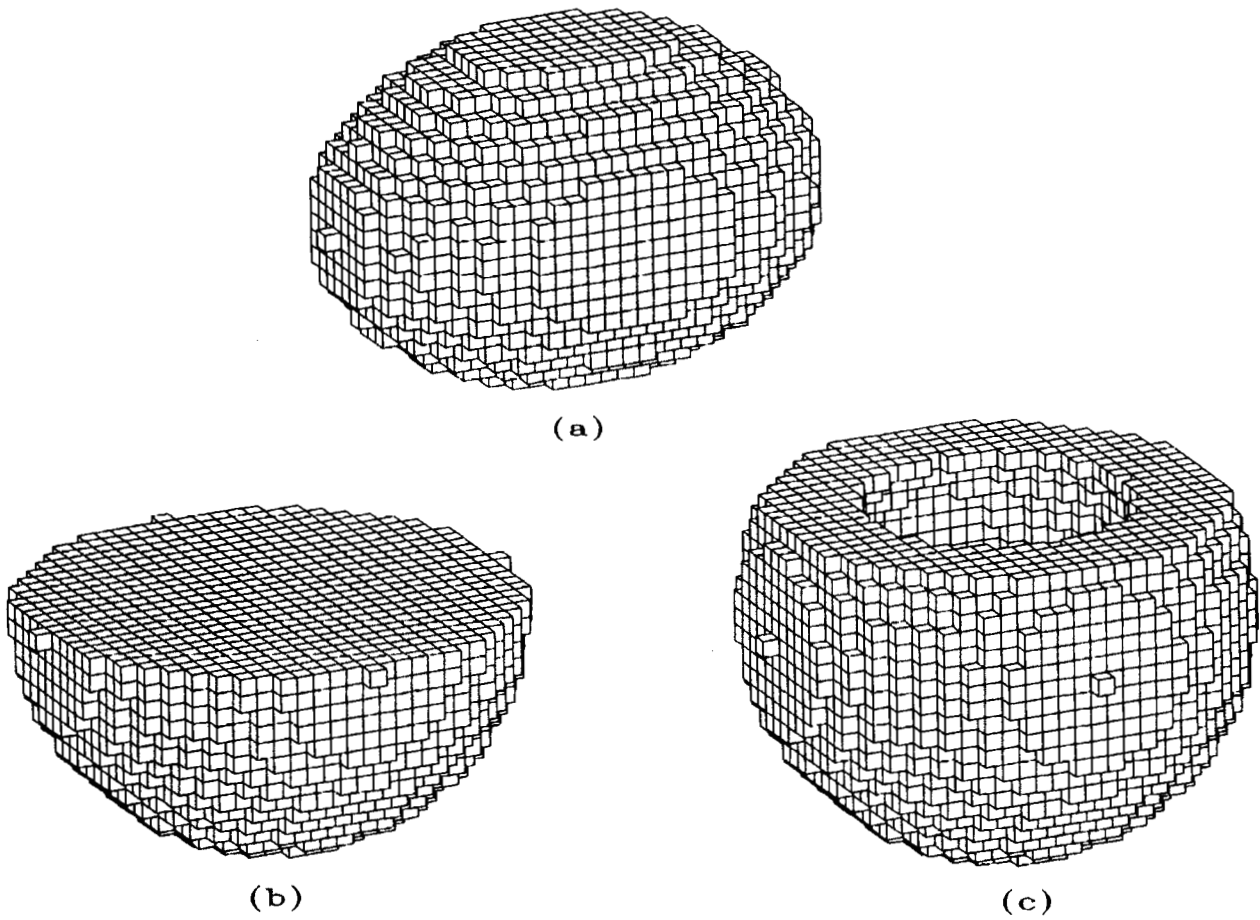


Figura 31. Objetos normalizados: (a) *A*; (b) *B*; (c) *C*. Estos objetos corresponden a los de la Figura 24 en forma respectiva, ya son invariantes bajo traslación, rotación y volumen (número de *voxels*).

6 Transformación de un Objeto a Otro

Considerando un *voxel* como una partícula. El trabajo dW hecho por la fuerza F , mientras la partícula recorre una distancia ds , está definido como el producto del desplazamiento por la fuerza en esa misma dirección.

$$dW = F ds. \quad (6)$$

En el caso que aquí concierne, la fuerza es considerada constante. Por lo tanto, el trabajo hecho para mover un *voxel* es función de la distancia Euclidean entre la posición $P(p_1, p_2, p_3)$ y $Q(q_1, q_2, q_3)$.

6.1 El trabajo realizado para transformar un objeto a otro

Con base en lo mencionado anteriormente, el trabajo para transformar un objeto A en otro B ($W(A \rightarrow B)$) define una métrica, donde:

$$\begin{aligned} W(A \rightarrow B) &= W(B \rightarrow A), \\ W(A \rightarrow B) &\geq 0, \\ W(A \rightarrow B) &= 0 \text{ si y solo si } A = B, \text{ y} \\ W(A \rightarrow C) &\leq W(A \rightarrow B) + W(B \rightarrow C). \end{aligned}$$

Considerando a A y B ya invariantes en traslación, rotación y volumen. El trabajo realizado para transformar A a B se calcula como sigue:

1. Se *sobreponen los objetos A y B* . La sobreposición está dada por la coincidencia de los ejes mayor y menor de ambos objetos. La Figura 32(a) muestra el objeto A , la 32(b) el B , respectivamente. La Figura 32(c) muestra la sobreposición de ambos objetos, los *voxels* del objeto B están marcados por puntos para diferenciarlos de A . En este caso de sobreposición se usó la información del origen (la intersección entre el eje mayor y menor). Otro origen bien puede ser la coincidencia de los centroides de los objetos a considerar.

2. Se encuentran los *voxels* comunes a ambos objetos y se dejan en su posición original. Sea A la imagen binaria 3D representada por $I_a(r, c, s)$ y B por $I_b(r, c, s)$, respectivamente. Entonces $I_d(r, c, s)$ se define como:

$$I_d(r, c, s) = I_a(r, c, s) \cap I_b(r, c, s).$$

La Figura 32(d) muestra la imagen $I_d(r, c, s)$, la cual corresponde a los *voxels* comunes entre $I_a(r, c, s)$ e $I_b(r, c, s)$. Este método de selección de *voxels* comunes no garantiza el máximo número de *voxels* comunes entre ambos objetos, es un método consistente que funciona para los objetivos planteados. Para garantizar la selección cercana a lo óptimo de *voxels* comunes se tendrá que aplicar el método visto en la subsección de máxima correlación 4.3.2 expandido a 3D.

3. *¿Qué voxels mover?* Los *voxels* a mover corresponden a la imagen binaria 3D definida por $I_e(r, c, s)$, esto es:

$$I_e(r, c, s) = I_a(r, c, s) \setminus I_b(r, c, s).$$

Los *voxels* a mover son llamados *voxels positivos*. La Figura 32(e) presenta los *voxels* positivos

4. *¿Dónde serán colocados los voxels positivos?* La imagen binaria 3D $I_f(r, c, s)$ representa los *voxels* (lugares) donde serán colocados los *voxels* positivos. Los *voxels* de $I_f(r, c, s)$ son llamados los *voxels negativos* y están definidos por:

$$I_f(r, c, s) = I_b(r, c, s) \setminus I_a(r, c, s)$$

La Figura 32(f) representa $I_f(r, c, s)$

5. *¿Qué voxel mover primero?* Hay muchas maneras de mover los *voxels*, si n es el número de *voxels* a mover entonces $n!$ es el número de diferentes maneras de colocarlos de I_e a I_f . Existen diferentes métodos para la solución óptima de este

problema, entre los principales destacan los mencionados en teoría de gráficas [41]. Así, se puede considerar una gráfica G bipartita con bipartición (I_e, I_f) , donde $I_e = \{i_{e_1}, i_{e_2}, \dots, i_{e_n}\}$ e $I_f = \{i_{f_1}, i_{f_2}, \dots, i_{f_n}\}$. La gráfica G corresponde a una gráfica bipartita completa con pesos asignados, los cuales están representados por las distancias entre los *voxels* positivos y negativos. Lo anterior describe un problema de asignación, con el uso del algoritmo de Kuhn-Munkres es posible encontrar el mínimo de trabajo empleado.

El uso del algoritmo de Kuhn-Munkres [41] para una gráfica bipartita completa con pesos asignados implica la utilización de una matriz $W = [w_{ij}]$, donde w_{ij} es el peso del segmento $i_{e_i}i_{f_j}$ en G . Una vez obtenida la solución óptima, el trabajo es minimizado. Las transformaciones progresivas de objetos se pueden realizar por la ordenación creciente de las distancias. Considerando a n lo suficientemente grande y las limitaciones en el equipo de cómputo utilizado se propuso utilizar un método heurístico de distancias mínimas, el cual resultó muy consistente. Así, el método utilizado para mover *voxels* es por medio del uso de la distancia mínima Euclideana, es decir, primero se *encuentran los dos voxels más cercanos* entre I_e e I_f .

$$d(I_{e_i}, I_{f_j}) = \min \sqrt{(r_{I_{f_j}} - r_{I_{e_i}})^2 + (c_{I_{f_j}} - c_{I_{e_i}})^2 + (s_{I_{f_j}} - s_{I_{e_i}})^2} \quad (7)$$

donde:

$d(I_{e_i}, I_{f_j})$ es la distancia mínima Euclideana entre los *voxels* de I_{e_i} e I_{f_j} .

I_{e_i} corresponde al i -ésimo *voxel* de I_e (*voxels* positivos).

I_{f_j} corresponde al j -ésimo *voxel* de I_f (*voxels* negativos).

Y segundo, se *encuentran los dos siguientes voxels más cercanos* entre I_e e I_f y así sucesivamente.

Para reducir el número de operaciones aritméticas, todas las distancias son calculadas por única vez y almacenadas en un arreglo, representadas por medio de

las siguientes tripletas:

$$(I_{e_1}, I_{f_1}, d(I_{e_1}, I_{f_1}))$$

$$(I_{e_1}, I_{f_2}, d(I_{e_1}, I_{f_2}))$$

.

.

$$(I_{e_1}, I_{f_n}, d(I_{e_1}, I_{f_n}))$$

$$(I_{e_2}, I_{f_1}, d(I_{e_2}, I_{f_1}))$$

$$(I_{e_2}, I_{f_2}, d(I_{e_2}, I_{f_2}))$$

.

.

$$(I_{e_n}, I_{f_n}, d(I_{e_n}, I_{f_n}))$$

Así, el número de distancias calculadas es n^2 . Posteriormente las distancias son ordenadas en forma creciente. La primer distancia es seleccionada y sus correspondientes *voxels* positivos y negativos, esto significa el primer movimiento de *voxels*; los *voxels* correspondientes a esta distancia son eliminados de la lista de tripletas y así sucesivamente.

No se ha probado que el método propuesto para mover *voxels* minimice el trabajo desarrollado para transformar de un objeto a otro, pero realmente es un método consistente en la transformación progresiva de objetos.

6.2 Ejemplos de transformaciones de objetos

La Figura 33 muestra las diferentes etapas de la transformación del objeto A en el objeto B en pasos de 400 *voxels*. Los objetos mostrados en las figuras 33(a) y 33(i) ya fueron normalizados y cada uno está representado por 6,000 *voxels* de información.

Los *voxels* en común entre ambos objetos son 2,787 y los *voxels* a ser movidos son 3,213. La Figura 33(b) muestra los primeros 400 *voxels* movidos, la Figura 33(c) los 800 *voxels* y así sucesivamente hasta la Figura 33(i). Se nota que usando el método de distancias mínimas primero se “mueven” los *voxels* más cercanos correspondientes a las primeras etapas de la transformación: figuras 33(b), 33(c), 33(d)... En las últimas figuras 33(g) y 33(h) principalmente, se nota que se están desplazando los *voxels* más lejanos.

Otro ejemplo de transformación de objetos se muestra en la Figura 34, donde se transforma el objeto *B* en el *C*, correspondientes a las figuras 34(a) y 34(i), respectivamente. Lo anterior se muestra en etapas de cada 270 *voxels* “movidos”. El número de *voxels* en común entre ambos objetos es de 3,820 y el número de *voxels* a ser colocados es de 2,180. La Figura 34(b) muestra los primeros 270 *voxels* que fueron colocados, la 34(c) los 540 y así sucesivamente.

La Tabla 2 resume los *voxels* en común entre los objetos normalizados: *A*, *B* y *C*. Se observa que los objetos *B* y *C* son los que tienen el mayor número de *voxels* en común. En forma contraria, entre los objetos *A* y *C* el número de *voxels* en común es el mínimo.

La Tabla 3 muestra los números de *voxels* a mover entre los objetos normalizados. Se nota que el mayor número de *voxels* a mover es entre los objetos *A* y *C*; y el menor entre *B* y *C*.

La figura 35 muestra otro ejemplo de transformaciones progresivas entre objetos, en este caso la variable de normalización fue menor que para los casos anteriores: resultando un tiempo menor en el proceso de transformación. La secuencia animada de estas etapas de transformación muestra las particularidades del algoritmo utilizado.

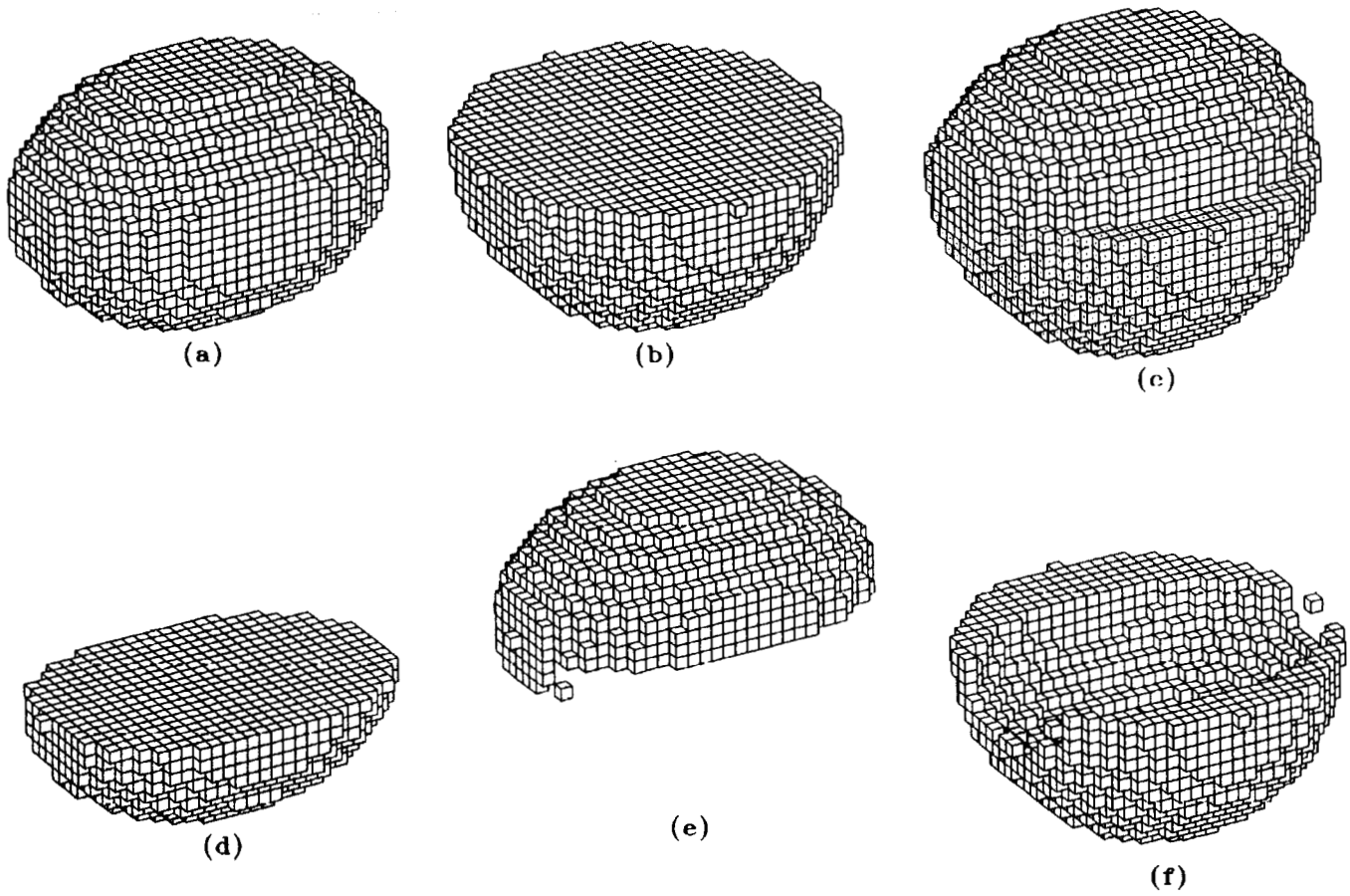


Figura 32. Pasos para la transformación de un objeto a otro: (a) el objeto *A*; (b) el objeto *B*; (c) la sobreposición de los objetos *A* y *B*. Los *voxels* correspondientes al objeto *B* son marcados con un punto central en cada uno de los planos de sus *voxels*, para diferenciarlos de *A*; (d) *voxels* comunes entre *A* y *B*; (e) *voxels* a ser movidos (*voxels* positivos); (f) *voxels* negativos, donde serán colocados los *voxels* positivos.

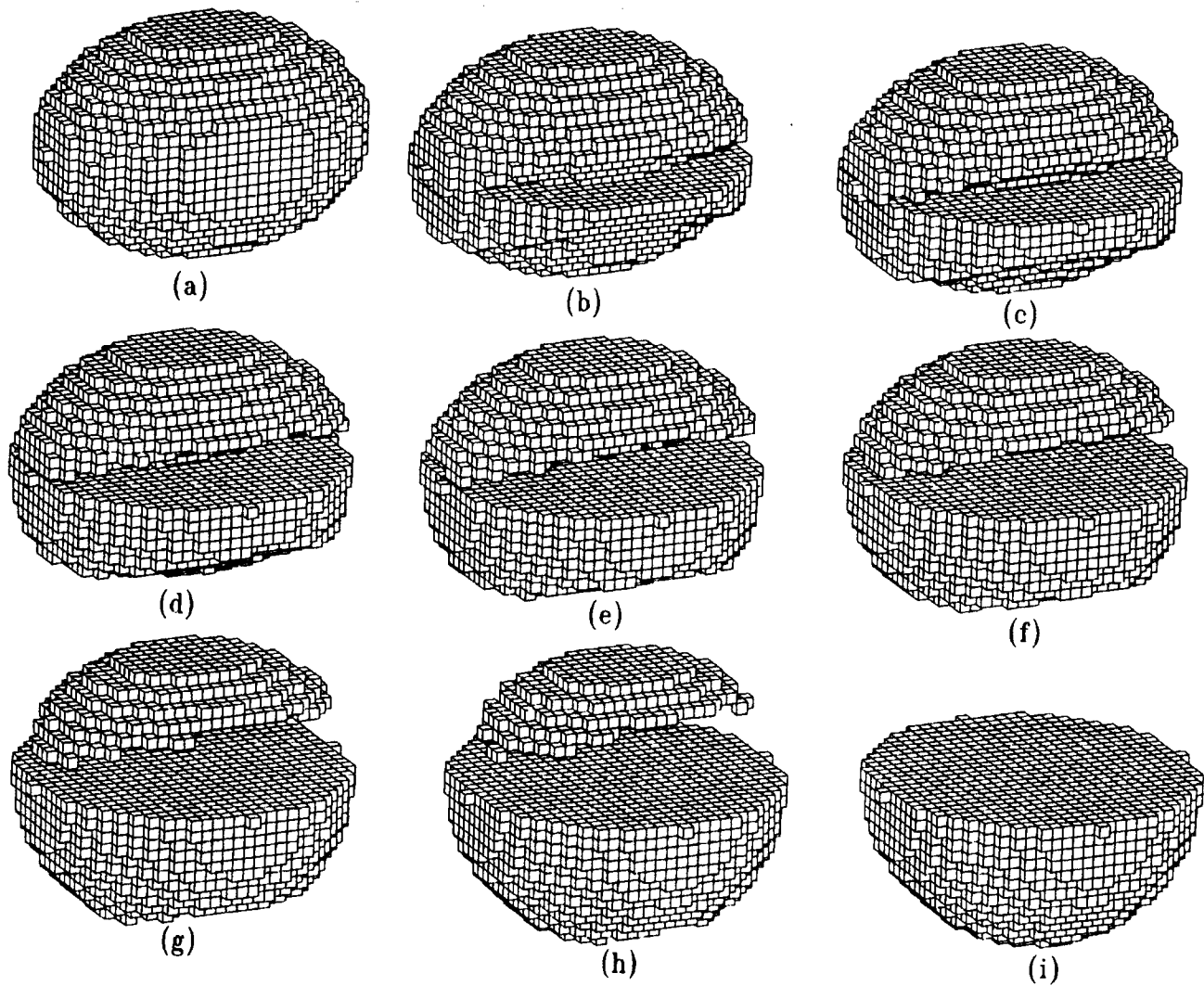


Figura 33. Trabajo realizado para transformar del objeto *A* al *B*. Las diferentes etapas de la transformación mostradas cada 400 *voxels*: (a) el objeto *A*; (b) los primeros 400 *voxels* colocados; (c) los 800; (d) los 1,200; (e) los 1,600; (f) los 2,000; (g) los 2,400; (h) los 2,800; y finalmente (i) los 3,213 *voxels* colocados.

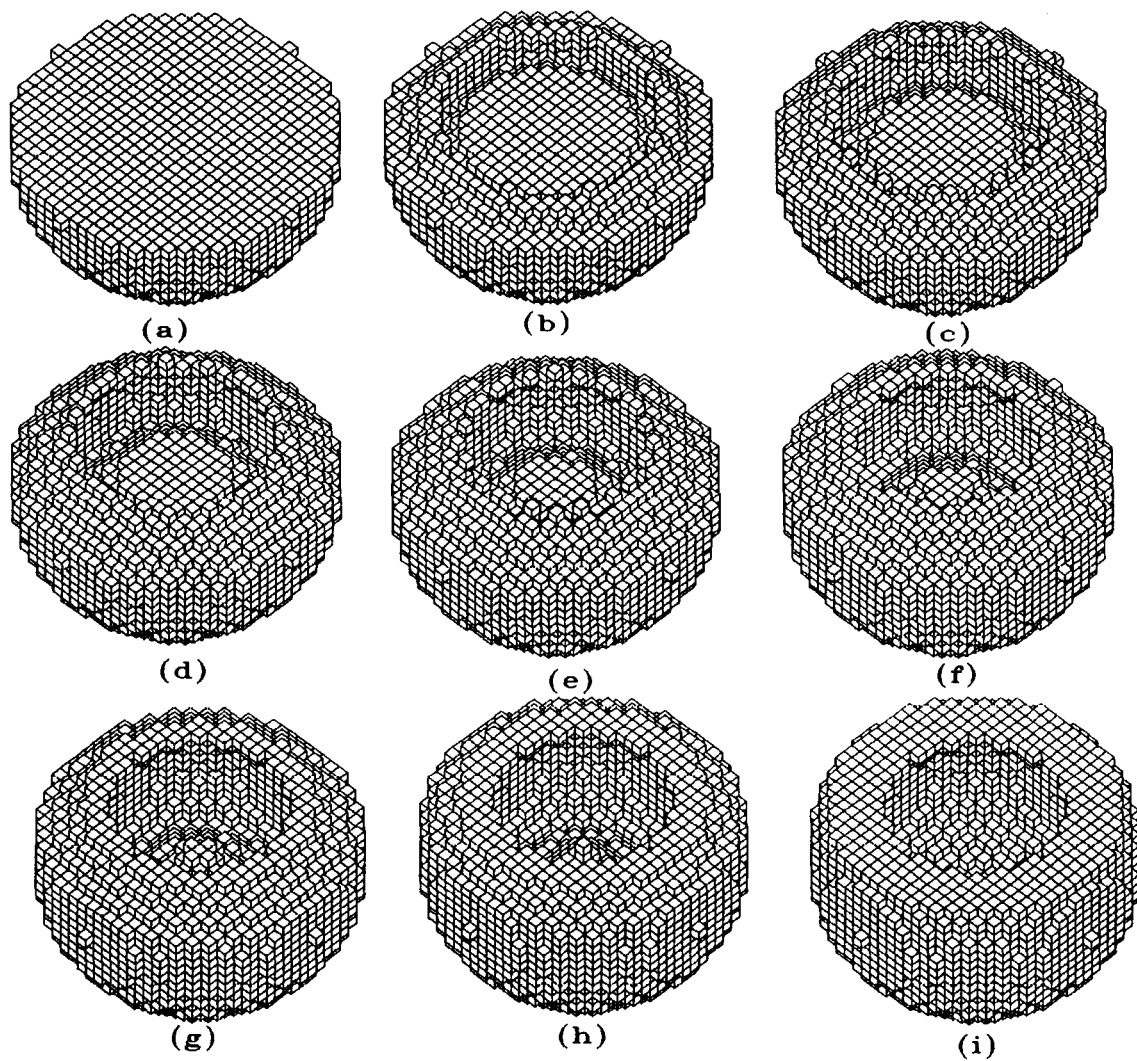


Figura 34. Trabajo realizado para transformar del objeto B al C . Las diferentes etapas de la transformación en pasos de 270 *voxels*: (a) el objeto B a ser transformado; (b) los primeros 270 *voxels* colocados; (c) los 540; (d) los 810; (e) los 1,080; (f) los 1,350; (g) los 1,620; (h) los 1,890; y finalmente (i) con los 2,180 *voxels* colocados.

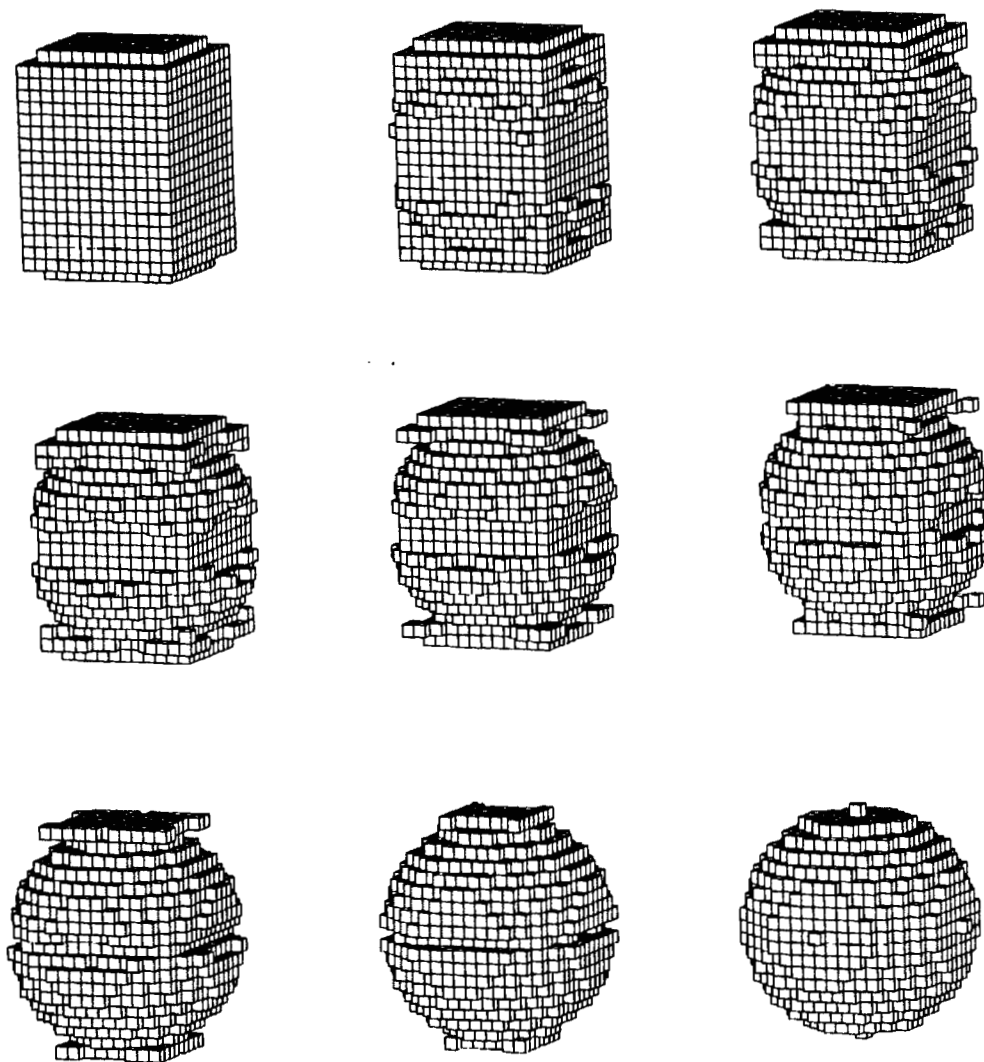


Figura 35. Transformaciones progresivas entre objetos usando menor cantidad de información que la de los objetos normalizados.

7 El Trabajo Desarrollado como Medida de Desemejanza entre Objetos

Objetos diferentes consumirán más trabajo en ser transformados de uno a otro, mientras que objetos similares emplearán una menor cantidad de trabajo. Cuando dos objetos son idénticos la cantidad de trabajo empleado en transformar de uno a otro es cero.

Así, la distancia D o desemejanza de forma entre dos objetos es obtenida contando cuántos *voxels* hay que mover y que distancia, para transformar uno a otro. La desemejanza entre los objetos A y B está definida por:

$$D(A, B) = \sum_{i,j}^n d(A_i, B_j). \quad (8)$$

La Tabla 4 despliega el trabajo realizado para transformar los objetos normalizados, esto es equivalente a una medida de desemejanza entre objetos. En conclusión: *los objetos más similares de los tres estudiados son el B y el C*, el trabajo realizado para transformar de uno a otro fue el menor que para los otros pares; en forma contraria, *los objetos más diferentes son el A y el C*, fueron los que requirieron la mayor cantidad de trabajo para ser transformados de uno al otro.

Voxels comunes	A	B	C
A	6000	2787	1624
B	2787	6000	3820
C	1624	3820	6000

Tabla 2. *voxels* en común entre los objetos normalizados.

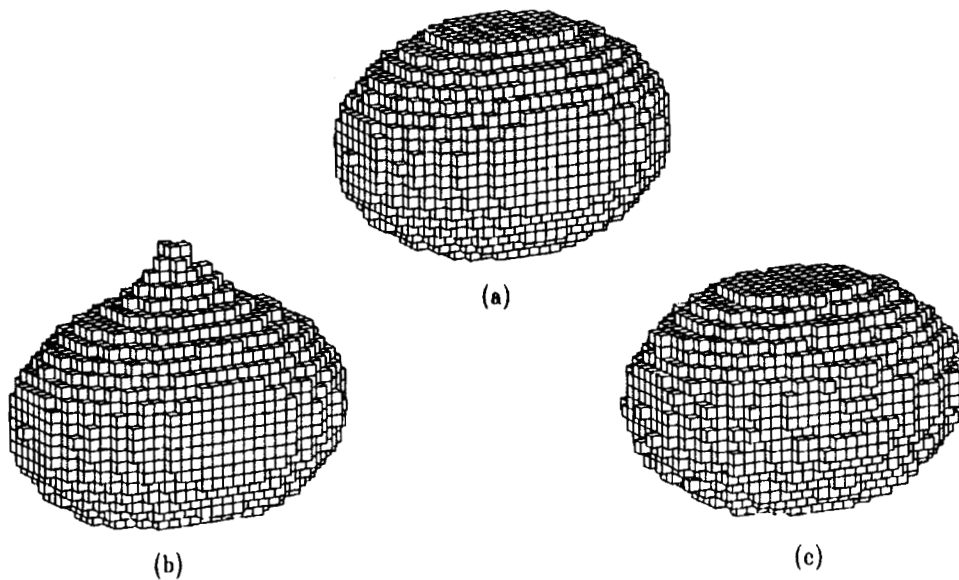
Voxels para colocar	A	B	C
A	0	3213	4376
B	3213	0	2180
C	4376	2180	0

Tabla 3. Número de *voxels* a ser colocados entre los diferentes objetos normalizados.

Medida de desemejanza	A	B	C
A	0	39,358.99	81,624.45
B	39,358.99	0	19,379.15
C	81,624.45	19,379.15	0

Tabla 4. Trabajo realizado para transformar un objeto a otro o medida de desemejanza entre objetos 3D.

Se puede suponer que la conclusión mencionada anteriormente pudo ser obtenida con sólo hacer una comparación entre objetos, es decir, calcular la diferencia entre las imágenes binarias que representan los objetos y que pueden ser vistas en la Tabla 2. Pero esto no es posible debido a lo siguiente: Los métodos de *image matching* sólo cuantifican la diferencia entre imágenes u objetos a comparar; este método cuantifica la diferencia y la distribución de esa diferencia o sea el lugar donde existe. La Figura 36 muestra lo que se ha mencionado, la Figura 36(a) muestra un objeto, la 36(b) y la 36(c) muestran los objetos en los que se va a transformar el objeto mostrado en 36(a). Se observa que el número de *voxels* comunes entre (a) y (b) y entre (a) y (c) es el mismo o sea 5,908 y el número de *voxels* a mover también es el mismo 92: se puede pensar que (a) se parece lo mismo con (b) y con (c). Pero no es así, a simple vista se nota una gran diferencia entre los objetos. Cuando se calcula la cantidad de trabajo para hacer las transformaciones de unos a otros, se observa que el trabajo realizado para transformar de (a) a (b) es de 1,836.03 (Figura 36(d)), mientras en el caso de transformar de (a) a (c) es de sólo 540.28 (Figura 36(e)). Cuando los objetos a comparar tienen la diferencia concentrada parecen más diferentes que cuando la tienen distribuida, y esto es detectado por este método que cuantifica cuántos *voxels* hay que mover y qué distancias tienen que recorrer.



Voxels comunes entre los objetos de (b) y (a) = 5908

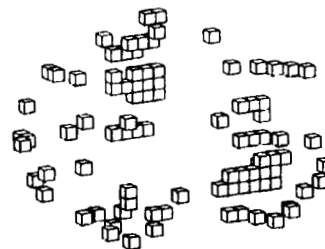
Voxels comunes entre (c) y (a) = 5908



Voxels a ser movidos = 92

Trabajo realizado para transformar de (b) a (a) = 1836.03

(d)



Voxels a ser movidos = 92

Trabajo para transformar de (c) a (a) = 540.28

(e)

Figura 36. Cuantificación de la diferencia de forma y distribución de la misma: (a) un objeto; (b) el objeto que será transformado al de (a); (c) el otro objeto que será transformado al de (a); (d) *voxels* a ser movidos y trabajo realizado para transformar de (b) a (a); (e) *voxels* a ser colocados y trabajo realizado para transformar de (c) a (a).

8 Conclusiones

La calidad del método de medida de semejanza de forma aquí presentado depende de dos partes: la primera, un método apropiado para una aproximación óptima de los *voxels* comunes, el método de máxima correlación extendido a 3D podría ser una opción; y segunda, un método que minimice el trabajo realizado en la transformación de un objeto a otro. La combinación adecuada de ambos métodos redundará en una mejor transformación de objetos. Los métodos presentados aquí son consistentes y arrojan resultados aceptables. El tiempo de procesamiento en la transformación de un objeto a otro fue de aproximadamente 190 segundos, en una computadora personal 486 a 66 Mhz. El tiempo cambia en forma directa al número de *voxels* a ser colocados. La parte lenta del programa es la representación gráfica de los objetos, debido al gran número de planos que se deben de ocultar.

Una parte importante del trabajo realizado es su aplicación en el reconocimiento de objetos irregulares, cuando la aplicación del uso de primitivas se dificulta. Unas partes complementarias a los métodos propuestos son:

1. *¿Cómo mover y colocar voxels?*. El método propuesto de colocación de *voxels* rompe el objeto a transformar. Otra forma de colocación de *voxels* puede ser sin romper el objeto a transformar, es decir, empujando los *voxels*.
2. *¿Cómo reducir el tiempo de procesamiento?*. decrementando el valor de la constante de normalización, sin afectar demasiado la precisión de los objetos.
3. *Normalización de la medida de semejanza entre objetos*. Normalizar la medida de semejanza a un estándar, como puede ser en porcentaje. Dos objetos idénticos tendrán una semejanza del 100%, mientras que objetos diferentes tendrán menor porcentaje.

8.1 Artículos derivados de esta tesis

1. E. Bribiesca, D. A. Rosenblueth, and M. Garza-Jinich, A definite-clause grammars for 2D shape analysis, *Computers & Mathematics with Applications*, Vol. 30, No. 8, pp. 95-103, (1995).

Este artículo presenta la parte inicial de la tesis enfocada al problema de formas en 2D. Usa el concepto de *curvatura discreta* presentado en la referencia [1] con técnicas gramaticales, para la clasificación de formas.

2. E. Bribiesca, Measuring 3D shape similarity using progressive transformations, *Pattern Recognition*, en prensa, aceptado para publicación el 28 de septiembre de 1995.

Este artículo describe la parte substantiva del trabajo de tesis, se enfoca a la transformación de objetos tridimensionales por medio del uso de transformaciones progresivas, describe dos métodos: primero, el uso de invariantes para maximizar el número de *voxels* en común; y segundo, el método para minimizar el trabajo realizado para la transformación de objetos.

3. E. Bribiesca and R. G. Wilson, A measure of 2D shape-of-object similarity, sometido en agosto a la revista *Applied Mathematics Letters*.

Este artículo propone un método para una aproximación óptima de los *voxels* en común de los objetos a transformar, este método es llamado de máxima correlación.

Apéndice A: Información 3D Utilizada

La información 3D utilizada para el inicio de esta tesis es la de los modelos digitales de terreno [42], provenientes de la carta topográfica escala 1:250,000, producidos por la Dirección General de Geografía del Instituto Nacional de Estadística, Geografía e Informática (INEGI). Esta información no es muy accesible a la mayoría de usuarios debido a su formato y proyección. El método presentado transforma este formato a uno estándar llamado DXF (Data eXchange File), compatible con la mayoría de los paquetes gráficos y geográficos actuales tales como: AutoCAD, ArcInfo, MGE (Microstation GIS Environment). El sistema de coordenadas también es modificado de coordenadas geográficas (ϕ, λ) , a coordenadas UTM (x, y) , estas últimas indicadas en metros, lo que simplifica el cálculo de áreas, distancias y ángulos.

La información cartográfica presentada sólo es una mínima parte de la existente, es la correspondiente a una parte del valle de México. En lo que sigue, a esta área, indistintamente se le llamará *sitio de prueba* o valle de México [43].

A.1 Formato de la información 3D utilizada

Las gráficas que se presentan en el contenido de esta tesis provienen de escala 1:250,000 y fueron digitalizadas considerando solamente las curvas de nivel, o sea la planimetría no fue digitalizada, aunque se puede incorporar posteriormente.

Toda la información está digitalizada y presentada en un grado de latitud por un grado de longitud, en la gran mayoría de los casos está cada tres segundos, es decir, la dimensión de la cuadrícula del MDT es de 1201 por 1201, con espaciamiento de cada tres segundos tanto en latitud como en longitud.

A la latitud del valle de México, un segundo es de aproximadamente 29.1 metros, es decir, el intervalo entre puntos del MDT es de 87.5 metros, por lo tanto la superficie que cubre el MDT es de aproximadamente 11,000 km². La orientación en que se

publican las cartas es la usual: el Norte hacia arriba y el Este a la derecha. Las elevaciones de cada uno de los puntos de la retícula se representa internamente en 1201 registros, cada uno con 1201 datos. Las elevaciones almacenadas en un registro dado corresponden a puntos sobre el mismo meridiano, ordenados de Sur a Norte; así, cada registro representa un *perfil* del terreno, orientado de Sur a Norte. Los registros se encuentran ordenados de Oeste a Este. Todas las elevaciones se representan con enteros binarios de 16 bits, justificados a la derecha con el bit de orden más alto indicando el signo. Los valores permitidos están en el rango de ± 32767 m. snm (sobre el nivel del mar). Los valores desconocidos se indican con todos los bits prendidos. Los valores negativos no se encuentran complementados.

A.2 Transformación de coordenadas

La información proporcionada por el INEGI es ya un MDT, como se mencionó en la sección anterior.

El *cambio de coordenadas* se propuso debido a que es más simple trabajar en metros, ya que pueden medir distancias, calcular áreas, diferenciar pendientes, etcétera.

Las fórmulas para la transformación de coordenadas geográficas a coordenadas de la cuadrícula UTM son:

Datos:

ϕ =Latitud en grados del punto a transformar

λ =Longitud en grados del punto a transformar

λ_0 =Longitud en grados del Meridiano Central correspondiente

$$\Delta\lambda = \lambda_0 - \lambda \quad (9)$$

XE =Valor de la coordenada x cuando el punto a transformar se encuentra al Este del Meridiano Central correspondiente.

XW =Valor de la coordenada x cuando el punto a transformar se encuentre al

Oeste del Meridiano Central correspondiente.

500 000=Falsa abscisa del Meridiano Central

$$XE = 500\ 000 + x \quad (10)$$

$$XW = 500\ 000 - x \quad (11)$$

$$x = \frac{111276.1806 \cos \phi \Delta \lambda}{(1 - 0.0067686579 \sin^2 \phi)^{1/2}} + \frac{5.649 \cos^3 \phi (\Delta \lambda)^3 (1 - \tan^2 \phi + 0.006815 \cos^2 \phi)}{(1 - 0.0067686579 \sin^2 \phi)^{1/2}} \quad (12)$$

$$y = 6332500.489(0.0175424666A - 0.0025601 \sin 2A + 0.0000027 \sin 4A) \quad (13)$$

Donde:

$$A = \phi + (0.0087861165 \tan \phi \cos^2 \phi (\Delta \lambda)^2 (1 - 0.0067686579 \sin^2 \phi))$$

Las ecuaciones (10) y (11) son empleadas cuando es necesario hacer una transformación en la traslación. Los coeficientes que están en las ecuaciones (12) y (13) corresponden a constantes establecidas por el cambio de coordenadas, si se deseara saber más sobre ellos es necesario consultar un libro sobre proyecciones. Utilizando las ecuaciones (12) y (13) cualquier punto definido por sus coordenadas geográficas es transformado a la cuadrícula UTM. Estas ecuaciones fueron utilizadas para generar el MDT ya en coordenadas UTM

A.3 Generación del archivo DXF

En esta parte del algoritmo se selecciona un elemento 3D para representar el MDT. El formato DXF es un formato estándar de *transferencia de información*. En el contenido

de esta tesis se selecciona AutoCAD como paquete de despliegue de información gráfica por ser de más amplio uso y de bajo costo, y también con menores requerimientos de *hardware*.

El elemento 3D llamado 3Dmesh es uno de los más apropiados para la representación del MDT, ya que almacena una matriz de hasta 256x256 elementos, es decir, queda un elemento superficial compuesto por planos. Para algunas aplicaciones este elemento, que en realidad es una superficie, puede ser transformado en planos independientes definidos por elementos 3Dfaces usando una de las instrucciones de AutoCAD.

La incorporación del MDT al archivo DXF se hace por medio de un superlenguaje, como puede ser: PASCAL, BASIC, C, FORTRAN, etcétera. En este caso se utilizó C, a continuación se genera un archivo DXF vacío, es decir, sin ninguna información gráfica. Posteriormente se procede a grabar el elemento gráfico 3Dmesh con un formato ya definido para ese elemento. El programa recupera la información del archivo de cotas del MDT, lo transforma a coordenadas UTM a cada uno de los valores y lo graba como vértice del 3Dmesh. La parte de información del MDT leída y transformada corresponde a la ventana de información seleccionada. Esta ventana se lee en un mapa topográfico escala 1:250,000 o 1:50,000 indicando, la x_{min}, y_{min} y x_{max}, y_{max} en coordenadas UTM. Existe una variable que permite seleccionar los incrementos entre puntos uniformemente en x y y a múltiplos del equivalente a 3 segundos.

Una vez grabado el MDT en el archivo DXF, puede ser utilizado e invocado por varios paquetes, en el caso de AutoCAD forma un dibujo DWG, al cual se le pueden incorporar otro tipo de elementos, o se pueden hacer lecturas y mediciones, ya que en esta parte todo el MDT está en coordenadas UTM

A.4 Resultados

La Figura 37 muestra una vista del valle de México, orientado con el Norte a la izquierda y el Este arriba. Las alturas fueron realizadas tres veces para mostrar más

detalle. Hacia el Sur se observa toda la parte que corresponde al Ajusco. Es notorio cómo el valle es encerrado por geformas realzadas en la parte Suroeste. En cambio en la dirección contraria hacia el Noreste, el valle se abre hacia Otumba. Hacia el Norte en forma aislada se puede distinguir lo que algunos llaman la Sierra de Guadalupe; la parte más alta corresponde al Pico Tres Padres, cerca el Cerro del Chiquihuite, el Cerro Zacatenco, etcétera. Observando a detalle en la zona central hacia el Sur se puede distinguir el Cerro de la Estrella en el área de Iztapalapa.

Es importante hacer notar que la Figura 37 no está al máximo detalle. Se leyeron los puntos a incrementos de 7 veces a partir de los datos originales. Esto es casi cada 630 metros sobre el terreno, siendo que la fuente original de información es de un punto cada 90 metros aproximadamente.

El MDT se puede representar como una matriz 3D compuesta de ceros y unos, se uso AutoLisp para poder representar cualquier imagen 3D por medio de *voxels*. Aunque la información de los modelos digitales de terreno fue importante para el inicio del trabajo de tesis, al final se usaron imágenes sintéticas, con las cuales fue más fácil corroborar los resultados.

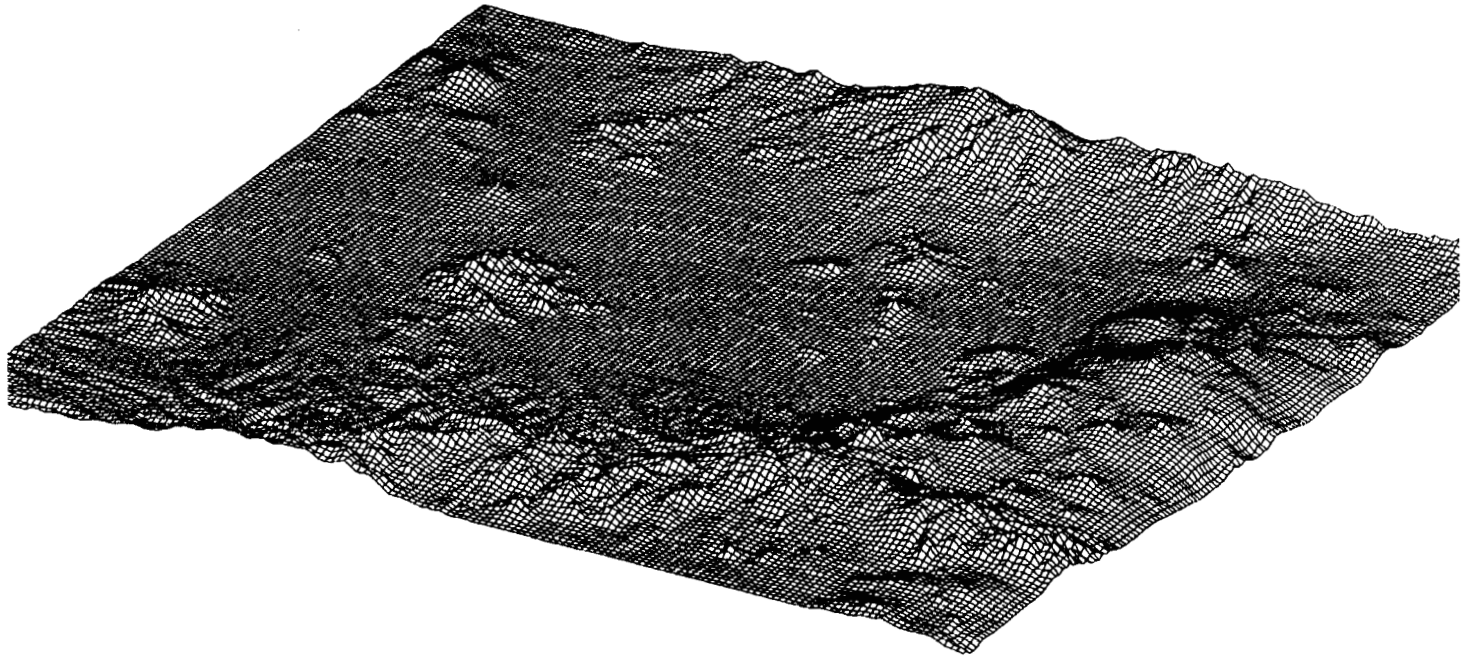


Figura 37. Vista del valle de México.

Bibliografía

1. E. Bribiesca, A geometric structure for two-dimensional shapes and three-dimensional surfaces, *Pattern Recognition*. **25**, 483-496 (1992).
2. E. Bribiesca, D.A. Rosenblueth, and M. Garza-Jinich, A definite-clause grammars for 2D shape analysis, *Computers & Mathematics with Applications*, Vol. 30, No. 8, pp. 95-103, 1995.
3. D. H. Ballard and C. M. Brown, *Computer Vision*, Prentice-Hall, Englewood Cliffs, New Jersey, 1982.
4. L. G. Roberts, Machine perception of three-dimensional solids, In *Optical and Electro-optical Information Processing*, (J. P. Tippett et. al. Eds.) MIT Press, Cambridge MA, 1965.
5. A. Guzmán, Decomposition of a visual scene into three-dimensional bodies, (Ph. D. Dissertation), in *Automatic Interpretation and Classification of Images*. (A. Grasseli Ed.) Academic Press, New York, 1969.
6. S. J. Dickinson, A. P. Pentland, and A. Rosenfeld, From volumes to views: an approach to 3-D object recognition, *CVGIP: Image Understanding*. **55**, 1992, 130-154.
7. A. A. G. Requicha, Representations of rigid solid objects, *Computer Surveys* **12**, 4, 1980.
8. P. Besl and R. Jain, Three-dimensional object recognition, *ACM Comput. Surv.* **17** (1), 1985, 75-145.
9. H. B. Voelcker and A. A. G. Requicha, Geometric modeling of mechanical parts and processes, *Computer* **10**, 1977, 48-57.

10. J. W. Boyse, Data structure for a solid modeller, *NSF Workshop on the Representation of Three-Dimensional Objects*, Univ. Pennsylvania, 1979.
11. B. I. Soroka, Generalised cylinders from parallel slices, *Proc., PRIP*, 1979, 421-426.
12. B. I. Soroka and R. K. Bajcsy, Generalized cylinders from serial sections, *Proc., 3rd IJ CPR*, 1976, 734-735.
13. R. Brooks, Model-based 3-D interpretations of 2-D images, *IEEE Trans. Pattern Anal. Mach. Intelligence* **5** (2), 1983, 140-150.
14. A. Pentland, Perceptual organization and the representation of natural form, *Artificial Intelligence* **28**, 1986, 293-331.
15. H. Freeman, Techniques for the digital computer analysis of chain encoded arbitrary plane curves. *Proc. Natn. Electron. Conf.* 18,312-324 (1961).
16. W. Perkins, A model-based vision system for industrial parts. *IEEE Trans. Computers*, C-27, 126-143 (1978).
17. H. Wechler, A structural approach to shape analysis using mirroring axes. *Computer Graphics and Image Processing* **9**, (3) 246-266 (1979).
18. T. Pavlidis, Algorithms for shape analysis of contours and waveforms. *Proc. 4th Int. Jnt. Conf. on Pattern Recognition*, Kyoto, 70-85 (1978).
19. R. F. A. Collard and H. F. J. M. Buffart, Minimization of structural information: a set-theoretical approach. *Pattern Recognition*, Vol. 16, No. 2, 231-242 (1983).
20. E. Persoon and K. S. Fu, Shape discrimination using Fourier descriptors. *Proc., 2nd IJ CPR*, 126-130 (1974).

21. E. Bribiesca, Arithmetic operations among shapes using shape numbers. *Pattern Recognition*, Vol. 13, No. 2, 123-137 (1981).
22. E. Bribiesca and A. Guzmán, How to describe pure form and how to measure differences in shapes using shape numbers, *Pattern Recognition*. **12**, 1980, 101-112.
23. P. J. van Otterloo, *A Contour-Oriented Approach to Shape Analysis*. Prentice Hall International (UK) Ltd. Great Britain (1991).
24. K. S. Fu, *Syntactic Methods in Pattern Recognition*. Academic Press, New York and London (1974).
25. J. Lin, Universal principal axes: an easy-to-construct tool useful in defining shape orientations for almost every kind of shape, *Pattern Recognition*, **26**, pp. 485-493, (1993).
26. G. James and R. C. James, *Mathematics Dictionary*, Fourth edition, Van Nostrand Reinhold Company (1976), p 295.
27. J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley (1979).
28. L. Sterling and E. Shapiro, *The Art of Prolog*. MIT Press (1986).
29. R. Baeza-Yates G. H. Gonnet, A new approach to text searching. *Communications of the ACM* 35, 10, 74-82 (1992).
30. R. N. Bracewell. *The Fourier Transform and Its Applications*, Electrical and Electronics Engineering Series, McGraw-Hill Book Company, second edition, (1978).
31. E. Brigham, *The Fast Fourier Transform and Its Applications*, Prentice-Hall, (1988).

32. H. Wechsler, Invariance in pattern recognition, In P. W. Hawkes, editor, *Advances in Electronics and Electron Physics*, **69**, pp. 262-322. Academic Press, (1987).
33. M. K. Hu, Visual pattern recognition by moment invariants, *IEEE Transactions in Information Theory*, **8** pp. 179-187, (1962).
34. W. Karush, *Webster's New World Dictionary of Mathematics*, Simon & Schuster, Inc., New York, 1989.
35. R. M. Haralick and L. G. Shapiro, Glossary of computer vision terms, *Pattern Recognition*. **24**, 1991, 69-93.
36. G. H. Schut, On exact equations for the computation of the rotational elements of absolute orientation, *Photogrammetria*. **17**, 1992-93, 34-37.
37. A. Pope, An Advantageous alternative parametrization of rotation for analytic photogrammetry, *Symposium on Computational Photogrammetry of the American Society of Photogrammetry*, Alexandria, VA. 1970.
38. L. Hinsken, A singularity-free algorithm for spatial orientation of bundles, *International Archives of Photogrammetry and Remote Sensing*, Vol. 27, Kyoto, Japan, 1988.
39. B. K. P. Horn, Closed-form solution of absolute orientation using unit quaternions, *Journal of the Optical Society of America A*, **4**, 1987, 629-642.
40. R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*, Volume II, Addison-Wesley Publishing Company, 1993.
41. J. A. Bondy and U. S. R. Murty, *Graph Theory with Applications*. The Macmillan Press Ltd (1976).

42. Y. Doytsher and B. Shmutter, Grids of elevations and topographic maps, Proceedings AUTO CARTO 5, Environmental Assessment and Resource Management, Crystal City, Virginia, U.S.A. (1982).
43. E. Bribiesca, Manual de utilización de banco de datos CETENAL, publicación de la Secretaría de la Presidencia, México (1977).