ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA

LÓGICA DIFUSA Y EXTRACCIÓN DE RASGOS TOPOLÓGICOS PARA EL RECONOCIMIENTO DE CARACTERES MANUSCRITOS

Tesis previa a la obtención del Título de

INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

Tito Polo Tituaña Pillajo

QUITO - ECUADOR

Noviembre de 1997

Certifico que la presente Tesis ha
sido elaborada en su totalidad por
el Sr. Tito Polo Tituaña

Dr. Gualberto Hidalgo H.
Director de Tesis

ÍNDICE GENERAL

ÍNDICE DE FIGURAS viii

RESUMEN xi

INTRODUCCIÓN xiii

CAPÍTULO I: LÓGICA DIFUSA

1.1. INTRODUCCIÓN A LOS SISTEMAS DIFUSOS 2
1.1.1. Aplicaciones de los Sistemas Difusos 2
1.2. Conjuntos Difusos y Funciones de Pertenencia 4
1.2.1. Teoria de Conjuntos Tradicionales 4
1.2.2. Conjuntos Difusos 6
1.2.3. Variables Linguísticas 9
1.2.4. Funciones de Pertenencia 11
1.2.5. Operaciones de Conjuntos 11
1.2.5.1. Operaciones con Conjuntos Tradicionales 11
1.2.5.2. Operaciones con Conjuntos Difusos 13
1.2.5.2.1. Conjuntos Especiales 14
1.2.5.2.2. Propiedades de la Unión, Intersección y Complemento 15
1.2.6. Lógica Difusa 16
1.2.6.1. Lógica Tradicional 16
1.2.6.2. Reglas de Inferencia Difusas 18
1.3.UNA APROXIMACIÓN ALGORÍTMICA DIFUSA PARA LA DEFINICIÓN DE CONCEPTOS COMPLEJOS E IMPRECISOS 21
1.3.1. Preguntas Atómicas 22
1.3.1.1. Preguntas Clasificacionales y atribucionales 24
1.3.1.2. Relacion entre Preguntas Clasificacionales y Atribucionales 24
1.3.2. Preguntas Compuestas y su Representación 25
1.3.2.1. Significado de B 27
1.3.2.2. Cuestionarios Ramificados 28
Referencias 30

CAPÍTULO II: RASGOS TOPOLÓGICOS

2.1. INTRODUCCIÓN 33
2.2. CARACTERÍSTICAS DE LOS RASGOS TOPOLÓGICOS 33
2.2.1. Algunos Métodos de Extracción de Rasgos 36
2.3. CATEGORIZACIÓN MULTINIVEL 40
Referencias 41

CAPÍTULO III: RECONOCIMIENTO DE CARACTERES MANUSCRITOS

3.1. EL PROBLEMA DE RECONOCIMIENTO DE PATRONES 44
3.1.1. Nociones de Reconocimiento de patrones y campo de Aplicación..... 44
3.1.2. Clasificación 46
3.1.3. Problemas fundamentales en el diseño de sistemas de reconocimiento de patrones 47
3.2. VISIÓN ARTIFICIAL 48
3.2.1. Introducción 48
3.3. PROCEDIMIENTO GENERAL PARA EL RECONOCIMIENTO DE PATRONES 50
3.3.1. Indicadores en los sistemas de reconocimiento de patrones 52
3.4. RECONOCIMIENTO DE CARACTERES MANUSCRITOS 53
3.4.1. Introducción 53
3.4.2. Perspectiva Histórica 55
3.4.3. Obtención de datos para la implementación de un sistema de reconocimiento de caracteres (SCANNER) 56
3.4.4. Sistema de reconocimiento de caracteres manuscritos 57
3.4.4.1. Consideraciones para el diseño de un sistema de Reconocimiento de caracteres 59
Referencias 59

CAPÍTULO IV: MÉTODOS DE RECONOCIMIENTO

4.1. LÓGICA DIFUSA (CHEN Y SIY) 62
4.2. EXTRACCIÓN DE RASGOS TOPOLÓGICOS 63
4.3. ANÁLISIS ESTRUCTURAL Y MATEMÁTICO DE LOS MÉTODOS MENCIONADOS 63
4.3.1. Representación de la imagen 63
4.3.1.1. Relaciones Básicas entre pixels 64
4.3.2. Esqueleto de una imagen binaria 68
4.3.2.1. Requerimientos que debe poseer el esqueleto de una imagen 68
4.3.3. Lógica Difusa 69
4.3.3.1. El Sistema de Reconocimiento 72
4.3.3.2. Operación del sistema 80
4.3.4. Extracción de Rasgos Topológicos 81
4.3.4.1. Estructura del sistema de reconocimiento 86
Referencias 89

CAPÍTULO V: PROGRAMA (SOFTWARE) PARA EL RECONOCIMIENTO DE CARACTERES MANUSCRITOS

5.1. DESCRIPCIÓN DEL PROGRAMA 92
5.2. ESTRUCTURA Y DIAGRAMAS DE FLUJO DEL PROGRAMA 93

5.2.1. Funcionamiento del algoritmo de la fase de entrenamiento 94
5.2.2. Funcionamiento de la fase de Reconocimiento 95
5.3. ORGANIZACIÓN DE LOS DATOS DEL PROGRAMA 96
5.4. MANUAL DEL USUARIO..... 97
5.4.1. Descripción del programa 97
5.4.2. Instalación del software..... 98
5.4.3. Funcionamiento del programa..... 98
5.4.4. Procedimiento para la simulación del programa *RECANUM* 103

CAPÍTULO VI: RESULTADOS OBTENIDOS Y CONCLUSIONES

6.1. OBTENCIÓN DE LAS MUESTRAS DE LOS CARACTERES MANUSCRITOS 107
6.2. DATOS Y RESULTADOS DE LA SIMULACIÓN DE LOS MÉTODOS DE RECONOCIMIENTO UTILIZADOS 108
6.2.1. Resultados obtenidos empleando el método de la Lógica Difusa..... 108
6.2.2. Resultados obtenidos empleando el método de la Extracción de Rasgos Topológicos 109
6.3. EVALUACIÓN DE LOS RESULTADOS OBTENIDOS DE LA SIMULACIÓN 110
6.3.1. Evaluación de los resultados obtenidos utilizando el método de la Lógica Difusa 110
6.3.2. Evaluación de los resultados obtenidos utilizando el método de la Extracción de Rasgos Topológicos..... 111
6.4. CONCLUSIONES 112

BIBLIOGRAFÍA 116

APÉNDICE A 119

APÉNDICE B 144

ÍNDICE DE FIGURAS

CAPÍTULO I

Fig. 1.1 Partición del conjunto U en subconjuntos: a) Por su Pertenencia, b) por su Color 5
Fig. 1.2. Ejemplos de Funciones de Pertenencia Difusa 7
Fig. 1.3. Función de pertenencia bivaluada para un conjunto no difuso 8
Fig. 1.4. Funciones de pertenencia para T(presión) 10
Fig. 1.5. Ilustración de la Unión e Intersección de conjuntos Difusos 15
Fig. 1.6. Tabla de verdad para operaciones aplicadas a las proposiciones 17
Fig. 1.7. Representación gráfica de una pregunta atómica 23
Fig. 1.8. Representación relacional de Q 25
Fig. 1.9. Ejemplo de un cuestionario ramificado 29

CAPÍTULO II

Fig. 2.1. Rasgos Topológicos fundamentales 35
Fig. 2.2. Puntos singulares 36
Fig. 2.3. Histogramas de proyección vertical y horizontal 37
Fig. 2.4. Contorno del dígito 5 38
Fig. 2.5. a) Código de cadena 4-direccional, b) Código de cadena 8-direccional 38
Fig. 2.6. Pasos para obtener el código de cadena 39
Fig. 2.7. Estrategia Ascendente 41

CAPÍTULO III

Fig. 3.1. Numerales cero: a) impresos por máquina, b) manuscritos 54
Fig. 3.2. Procedimientos en un sistema de reconocimiento de caracteres 58

CAPÍTULO IV

Fig. 4.1. Representación de la imagen 64
Fig. 4.2. Vecinos de un pixel p 65
Fig. 4.3. a) Vecindad 3×3 alrededor de un punto(x,y) en una imagen, b) Máscara 3×3 que muestra los coeficientes 67
Fig. 4.4. a) Imagen digitalizada sin preprocesamiento, b) Esqueleto de la imagen 68
Fig. 4.5. Conjunto de trazos característicos 71
Fig. 4.6. Ejemplo típico de un numeral 5 71
Fig. 4.7. Diagramas de bloques del sistema de reconocimiento. a) Fase de aprendizaje, b) Fase de reconocimiento 72

Fig. 4.8. Vecindad-8 de p 74
Fig. 4.9. Construcción de una línea para la clasificación de una curva en un tipo de rama 77
Fig. 4.10. Caracter 1 con sus respectivos nodos numerados a ser codificado 79
Fig. 4.11. Grilla Octogonal 82
Fig. 4.12. Grilla Octogonal para el caracter 2 85
Fig. 4.13. Estructura básica del sistema de reconocimiento: a) Aprendizaje, b) Reconocimiento 86
Fig. 4.14. Grilla de medición 87

CAPÍTULO V

Fig. 5.1. Diagrama de flujo general del programa RECANUM 94
Fig. 5.2. Diagrama de flujo del proceso de entrenamiento 95
Fig. 5.3. Diagrama de flujo del procesamiento de reconocimiento 96
Fig. 5.4. Arbol de archivo de muestras *. TIF 96
Fig. 5.5. Arbol de archivo de datos provenientes del proceso de entrenamiento 97
Fig. 5.6. Pantalla de presentación del programa RECANUM 99
Fig. 5.7. Pantalla de menú de opciones 100
Fig. 5.8. Pantalla de opciones seleccionadas 101
Fig. 5.9. Pantalla de pedido de archivo *.TIF a procesarse 101
Fig. 5.10. Pantalla de salida de datos para carácter reconocido 102
Fig. 5.11. Pantalla de salida de datos para carácter no reconocido 102
Fig. 5.12. Pantalla de salida para entrenamiento de un archivo *.TIF 103
Fig 5.13 Región a ser rastreada 104

CAPÍTULO VI

Fig. 6.1. Cuadrícula digitalizada con 100 muestras de caracteres numéricos... 107

RESUMEN

En este trabajo se presentan dos sistemas de reconocimiento de caracteres manuscritos basados en el concepto de la Lógica Difusa y en la Extracción de rasgos Topológicos. Un algoritmo de esqueletonización (adelgazamiento) se aplica a los caracteres manuscritos en la etapa de preprocesamiento, luego del cual se extraen las características esenciales del carácter, y mediante una adecuada representación de estas características, el carácter puede ser reconocido

Cada uno de los métodos han sido implementados en un software que permite simularlos.

El programa que permite la simulación se denomina RECANUM. En cada uno de los métodos se llevan a cabo dos procesos: el de entrenamiento, y reconocimiento.

Los resultados obtenidos de la simulación alcanzan porcentajes de reconocimiento del 78.6% para el caso de la Lógica Difusa y del 54% para el caso de la extracción de rasgos topológicos. Estos resultados fueron obtenidos con un conjunto de entrenamiento de 70 y 200 muestras de cada numeral respectivamente.

El tiempo de procesamiento en la fase de entrenamiento para los dos métodos fue de 75 minutos por cada 100 caracteres, tiempo empleado por un computador Cx586-S de 100 Mhz.

INTRODUCCIÓN

El reconocer es un atributo que poseen los seres humanos, lo que le ha permitido al hombre desarrollarse de una manera muy eficiente dentro del medio ambiente que le rodea.

La importancia de este atributo le ha impulsado al hombre a desarrollar sistemas dedicados especialmente al análisis de los mecanismos mentales que intervienen en la adquisición y reconocimiento de la información proporcionada por el mundo exterior, así como también de sistemas informáticos capaces de realizar estas mismas funciones dentro del marco de la inteligencia artificial.

Muchos esfuerzos para el desarrollo e investigación de los sistemas de reconocimiento de caracteres ópticos se han realizado desde el inicio de los años 50, enmarcados en el campo del reconocimiento de patrones.

El reconocimiento de caracteres manuscritos tiene muchos campos de aplicación, por lo que su estudio ha desarrollado cientos de sistemas de reconocimiento, los cuales pueden desarrollar tareas específicas.

La presente Tesis abarca campos que permiten un estudio general del reconocimiento de patrones y específicamente el de reconocimiento de caracteres manuscritos. De esta manera los capítulos I y II permiten obtener una visión adecuada de los medios utilizados para desarrollo de sistemas de reconocimiento de caracteres manuscritos, basados en la Lógica Difusa y la Extracción de Rasgos Topológicos.

El capítulo III contiene información relevante para el entendimiento de lo que realmente constituyen los sistemas de reconocimiento de patrones.

En el capítulo IV se analizan los procedimientos a ser utilizados para la implementación de dos sistemas de reconocimiento de caracteres manuscritos, basados en la Lógica Difusa y en la extracción de rasgos Topológicos.

La parte final de la presente tesis que comprende los capítulos V y VI se enmarcan en el desarrollo del software que permitirá la simulación de los métodos propuestos.

1.1. INTRODUCCIÓN A LOS SISTEMAS DIFUSOS [1] [2]

Vivimos en un mundo de complejidad y variedad maravillosa, un mundo donde los eventos nunca se repiten exactamente. Aunque los eventos nunca son exactamente los mismos, ellos no son tampoco completamente diferentes. Hay un hilo de continuidad, similaridad, y predictibilidad que nos permite generalizar, a menudo correctamente, de experiencias pasadas o anteriores a eventos futuros.

La mayoría de los fenómenos que encontramos a diario son imprecisos. Esta imprecisión puede estar asociada con sus formas, posición, momento, color, o aún la semántica que describe lo que son ellas, de tal forma que por ejemplo, cuando un instructor de conducción da instrucciones al piloto acerca de cuando frenar al entrar en una curva. Diría el instructor : "frena 15 metros antes de tomar la curva"?, o algo como : "frena cuando estés bastante cerca de la curva" ?. Por supuesto que la última de las instrucciones, ya que aunque es imprecisa , la expresión "cerca" puede ejecutarse de mejor manera pero de forma aproximada solamente.

El lenguaje cotidiano es un ejemplo de las formas de imprecisión que se usa y se propaga. La imprecisión en datos e información recogida en nuestro medio ambiente puede ser ESTADISTICA (Por ejemplo el lanzamiento de una moneda es asunto de probabilidad) o NO ESTADISTICA (Por ejemplo, " frena cuando estés bastante cerca de la curva "). Este último tipo de incertidumbre se conoce como "DIFUSIDAD" ("FUZZINESS").

Desde niños aprendemos a interpretar y ejecutar instrucciones difusas . Todos asimilamos y usamos datos difusos, información y reglas imprecisas, de la misma manera que somos capaces de formular planes, tomar decisiones, y reconocer conceptos compatibles con altos niveles de indecisión y ambigüedad.

1.1.2 Aplicaciones de los Sistemas Difusos

Los Sistemas Difusos tienen muchas aplicaciones en el mundo de hoy, estos almacenan bancos de asociaciones difusas o "reglas" de sentido común . Estas

asociaciones relacionan un conjunto difuso de entrada con un conjunto difuso de salida los cuales tienen la estructura de reglas condicionales difusas (*If-then rules*) (*Si... entonces*). Así por ejemplo un controlador de temperatura difuso puede contener la asociación difusa "*Si* la temperatura del motor esta alta (*caliente*), *entonces incremente la velocidad* del ventilador" . Los fenómenos difusos admiten grados. Algunas configuraciones de temperatura son más altas que otras. Algunas velocidades son más rápidas que otras. La simple asociación difusa (ALTA, MAYOR), codifica todas estas combinaciones [3].

Los ingenieros han aplicado exitosamente los sistemas difusos en distintas áreas.

Aplicaciones de Control.- control de naves espaciales (Rockwell Corp.), operación del metro (Hitachi), transmisión automática (Nissan, Subaru), acoplamiento de enlace espacial (NASA), control de aire acondicionado, Control de semaforización urbana para tráfico vehicular [4].

Itinerario y optimización.- itinerario de un elevador (Hitachi, Mitsubishi) y análisis de stock en un supermercado (Yamaichi Securities) [4].

Análisis de señales para Sintonización e Interpretación .- ajuste de la imagen para Televisión (Sony), Reconocimiento manuscrito (Sony Palm Top), autoenfoque en cámara de video (Sanyo, Fisher, Canon), estabilizador de video imagen (Matshushita/Panasonic) [4].

Como se puede observar, la mayoría de empresas que han aplicado los sistemas difusos son Japonesas, pero actualmente estos productos son vendidos en todo el mundo.

Los sistemas difusos razonan mediante deducciones asociativas paralelas. Cuando se hace una pregunta o se ingresa un dato de entrada, un sistema difuso, emite cada regla difusa en paralelo, pero a diferente grado, para deducir una conclusión o salida. Así los sistemas difusos razonan con conjuntos, conjuntos difusos o multivaluados, en vez de proposiciones bivaluadas.

1.2 CONJUNTOS DIFUSOS Y FUNCIONES DE PERTENENCIA

Introducción

Muy a menudo las clases de objetos que encontramos en el mundo real fisico no tienen un criterio precisamente definido de pertenencia. Así por ejemplo, la clase de animales, incluye de una manera clara, perros, gatos, caballos, tigres, etc. como miembros que pertenecen a este grupo, y claramente también excluye objetos tales como rocas, líquidos, plantas, etc. Sin embargo objetos tales como estrellas de mar, bacterias, etc. tienen una posición ambigua respecto a la clase de animales. La misma clase de ambigüedad se da en el caso de un número tal como el 20 respecto a la "*clase*" de todos los números reales que sean mucho mayores que 1.

Evidentemente, la "clase de todos los números reales que sean mucho más grandes que 1,"o "la clase de mujeres bonitas," o "la clase de hombres altos" no constituyen clases o conjuntos en el sentido matemático tradicional de estos términos.
Sin embargo del hecho de que queden definidas de una manera imprecisa las "*clases*", estas juegan o desempeñan un papel importante en el pensamiento humano, particularmente en los dominios del reconocimiento de patrones, comunicación de información, y abstracción.

Los conjuntos difusos fueron introducidos en 1965 por Lotfi Zadeh como una manera nueva de representar la imprecisión en la vida cotidiana. Son una generalización de la teoría de conjuntos tradicional, una de las estructuras básicas esenciales en los modelos y matemáticas computacionales [1].

1.2.1 Teoría de Conjuntos Tradicionales [4]

Recordemos que un conjunto tradicional A en un universo U (el cual provee el conjunto de valores permitidos para una variable) puede definirse enlistando todos sus elementos o identificando los elementos $x \in A$. *Una forma de hacer esto es especificando la condición por la que* $x \in A$. *De tal forma que A puede ser definido como:*

A={x / x satisface alguna condición}

Alternativamente podemos introducir una función denominada ***función de pertenencia cero-uno*** (también llamada función característica, función de discriminación, o función indicadora) la cual nos sirve para definir un conjunto A, denotada por:

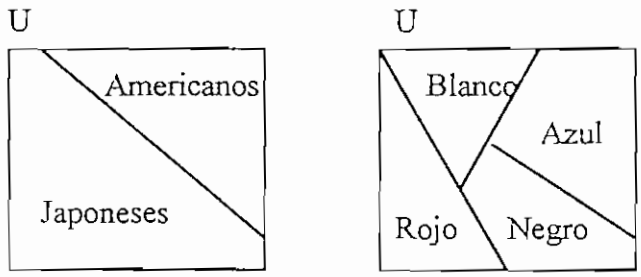
$$\mu_A(x)$$

De tal manera que :

$$\mu_A(x) \Rightarrow \begin{cases} 1 & si\ x \in A \\ 0 & si\ x \notin A \end{cases} \qquad (1\text{-}1)$$

A es matemáticamente equivalente a su función de pertenencia $\mu_A(x)$ en el sentido de que, conocer $\mu_A(x)$ es lo mismo que conocer A.

Por ejemplo: Consideremos un conjunto de todos los automóviles en Quito; es decir el conjunto U (Universo). De este conjunto U puede obtenerse muchos subconjuntos diferentes como se puede ver a continuación:



(a) (b)

Fig.1.1 Partición del conjunto U en subconjuntos por : (a) Por su procedencia, y (b) por su color.

Si tomamos en consideración el conjunto de automóviles americanos, de tal forma que;

B={x / x sea americano}

Si el automóvil es americano entonces el valor de la función de pertenencia $\mu_B(x)= 1$, caso contrario $\mu_B(x) = 0$.

Los conjuntos tradicionales corresponden a la ***lógica bivaluada***, es decir es o no es, encendido o apagado, blanco o negro, 1 o 0. En Lógica los valores de μ_B se denominan ***valores de verdad*** en relación a la pregunta, "está x en B?". Si la respuesta es si, $\mu_B(x) = 1$; y si la respuesta es no, $\mu_B(x) = 0$.

1.2.2. Conjuntos Difusos

Un conjunto Difuso es una generalización de un conjunto tradicional cuya función de pertenencia toma únicamente uno de los dos valores o cero (0) o uno (1).

Un conjunto Difuso ***F definido en un universo en consideración U, se caracteriza por su función de pertenencia*** $\mu_F(x)$***, la cual toma valores en el intervalo [0.0, 1.0].*** **La función de pertenencia** ***provee una medida del grado de similaridad de un elemento de U al subconjunto difuso [1] .***

Un Conjunto Difuso entonces es aquel que contiene objetos, tales que estos satisfacen propiedades imprecisas que varían en diferentes grados. Por ejemplo

Sea:

$$F=\{x \in R / x \text{ sea cercano a } 8\}$$

En 1965 Zadeh propuso representar a F por una ***función de pertenencia,*** denominada μ_F, la cual puede tomar cualquier valor en el intervalo [0.0 ,1.0]. El valor de $\mu_F(x)$ se denomina grado de pertenencia de x en F. Este convenio definido por los conjuntos difusos corresponde a la ***lógica continuamente valuada*** o ***multivaluada***; por lo que se puede decir que la escala de gris entre negro (=1) y blanco (=0) se puede describir utilizando estas definiciones.

Debido a que la propiedad "cercano a 8" es difusa, no existe una única función de pertenencia para F. Por lo tanto se deja al diseñador que decida, basándose en la aplicabilidad y propiedades que se deban dar a F.

Para este conjunto difuso en particular se pueden dar propiedades que parecen razonables:

Normalidad: $\mu_F(8) = 1$

Monotonía: Mientras más cercano este x a 8, mas cercano está $\mu_F(x)$ de 1, y a la inversa.

Simetría: Números igualmente lejos tanto a izquierda como a derecha de 8 deberían tener igual grado de pertenencia, ya que no hay ninguna razón para creer que números a la izquierda de 8 estén cercanos a 8 en distinta forma que los números a la derecha de 8. Pero hay que ***enfatizar*** en el hecho de que el diseñador es libre de hacer otro tipo de interpretaciones.

Para este caso particular se pueden dar algunos ejemplos de funciones de pertenencia de forma gráfica:

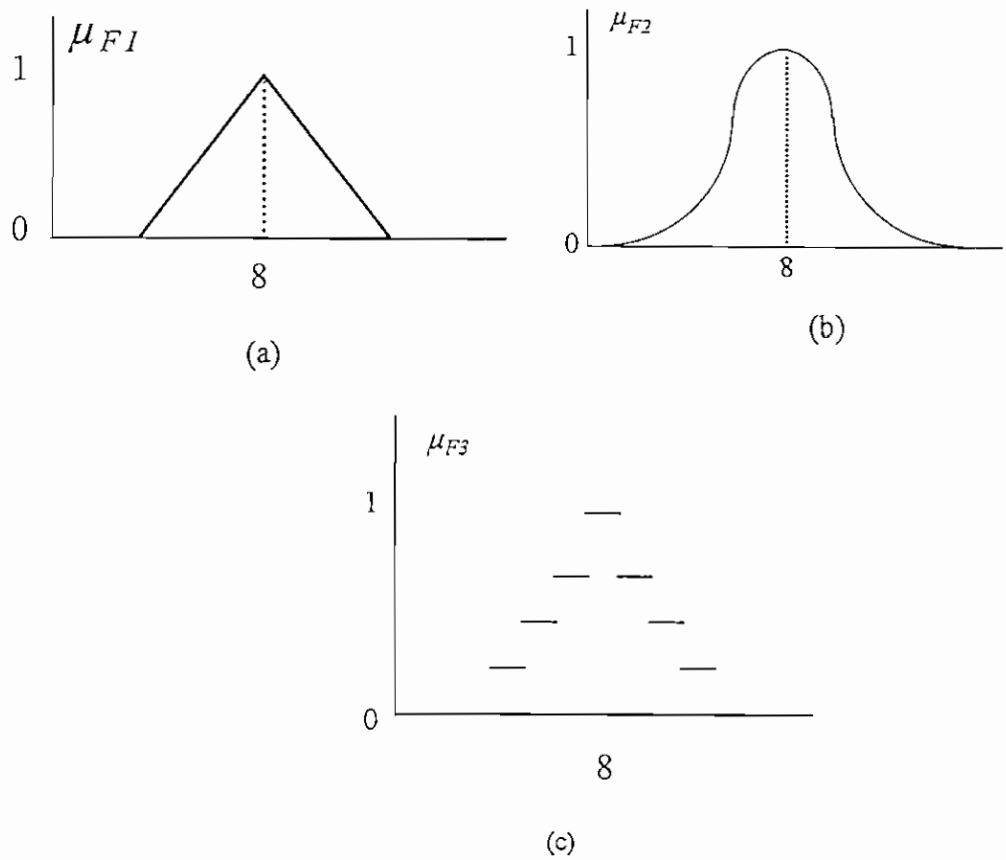


Fig.1.2 (a), (b), y (c): Ejemplos de Funciones de pertenencia difusa

Como se puede observar μ_{F1} y μ_{F2} son continuas pero μ_{F3} es discreta. μ_{F1} es continua pero no es suave, mientras μ_{F2} es continua y suave.

Entonces se puede apreciar que mediante estas funciones de pertenencia descritas arriba se puede describir los números cercanos a 8. Se puede además encontrar la diferencia del ***tipo de información*** que entrega un modelo difuso y un modelo tradicional.

Supongamos que la función de pertenencia para el conjunto tradicional es:

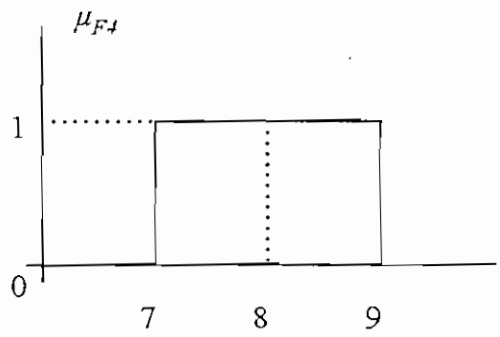


Fig.1.3 Función de pertenencia bivaluada para un conjunto no difuso

Supongamos que los números en *Fi* son horas de llegada al trabajo en la mañana. Si alguien dice que $\mu_{F4}(x)=1$ entonces sabemos que la hora de entrada al trabajo es entre las 7 y 9 de la mañana. Por otro lado conociendo que $\mu_{F1}(x)=0.98$, entonces esto nos dice que las horas de llegada al trabajo son bastante cercanas a las 8 de la mañana.

Por lo tanto esto nos ilustra el punto importante de que el ***tipo de información*** entregado por el modelo difuso es diferente tanto en ***cantidad*** como en ***calidad***.

Un conjunto difuso ***F*** en **U** puede representarse como un conjunto de pares ordenados de un elemento genérico ***x*** con su grado de función de pertenencia:

$$F = \{ (x, \mu_F(x)) / x \in U \}$$

Cuando U es continuo (por ejemplo números reales), F es comúnmente escrito como:

$$F = \int_U \mu_F(x) / x$$

En esta ecuación el signo de integral no denota integración; éste denota la colección de todos los puntos $x \in U$ con su función de pertenencia asociada $\mu_F(x)$.

Cuando U es discreta, F se escribe comúnmente como:

$$F = \sum_U \mu_F(x) / x$$

En esta ecuación el signo de sumatorio no denota una suma aritmética; sino la colección de todos los puntos $x \in U$ con su función de pertenencia asociada $\mu_F(x)$. El slash (raya de quebrado) en estas últimas expresiones asocia los elementos de U con sus grados de pertenencia, donde $\mu_F(x) \geq 0$.

1.2.3 Variables Lingüísticas

Zadeh nos dice " alejándose de la precisión ante la complejidad intensa, es natural explorar el uso de lo que se pueden llamar *variables* lingüísticas, es decir, variables cuyos valores no son números sino palabras u oraciones en un lenguaje natural . ... La motivación del uso de palabras u oraciones en vez de números es que las caracterizaciones lingüísticas son, en general, menos específicas que las caracterizaciones numéricas [5].

Consideremos a u representando el nombre de una variable lingüística (por ejemplo temperatura). Los valores numéricos de una variable lingüística u son representados o denotados por x, donde $x \in U$.

Una variable lingüística es generalmente descompuesta en un conjunto de términos, $T(u)$, el cual cubre su universo en consideración.

Por ejemplo , consideremos la *presión (u)* como una variable lingüística. Esta puede ser descompuesta en los siguientes conjuntos de términos:

T(presión)={ tenue, bajo, bien, fuerte, alto}

donde cada término en *T(presión)* está caracterizado por un conjunto difuso en el universo en consideración :

U= [100 psi, 2300psi]

Así podemos interpretar *tenue* a la presión debajo de 200 psi, *bajo* a la presión cercana a 700 psi, *bien* (okay) a una presión cercana a 1050 psi, fuerte como una presión cercana a 1500 psi, y alta como una presión sobre los 2200 psi. Estos términos pueden ser caracterizados como conjuntos difusos cuyas funciones de pertenencia se muestran en la siguiente figura.

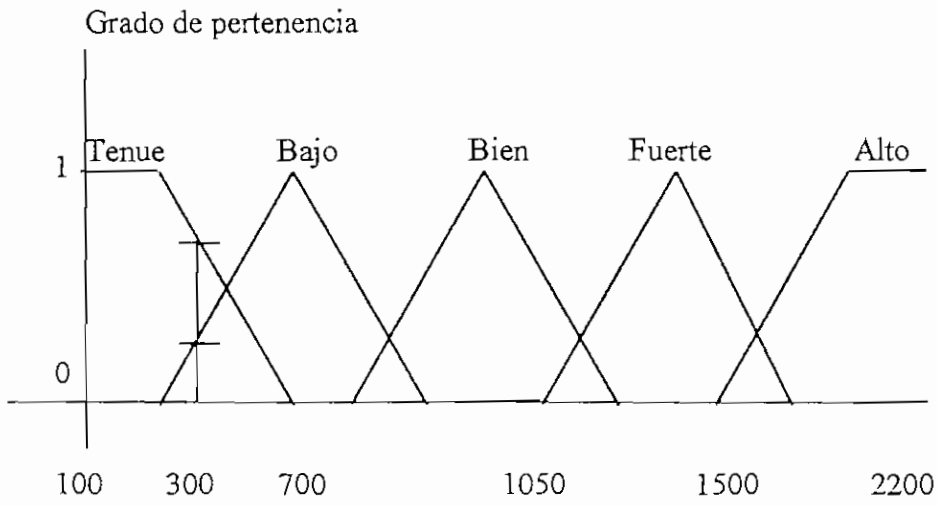


Fig.1.4 Funciones de pertenencia para *T(presión)* ={tenue, bajo, bien, fuerte, alto}. El tipo de curva y la peudiente de las curvas de las funciones de pertenencia son completamente arbitrarias

Los valores de las presiones obtenidas (x) se encuentras sobre el eje de la presión. En este ejemplo se puede observar que si se traza una vertical para un valor cualquiera de x se pueden obtener dos valores de intersección para las funciones de pertenencia. De esta manera para la línea vertical trazada para el valor de x=300 psi por ejemplo, esta reside en los conjuntos difusos *presión tenue* y *presión baja*, pero a diferentes grados de pertenencia.

1.2.4 Funciones de Pertenencia

En la mayoría de aplicaciones de los conjuntos difusos y de la lógica difusa, las funciones de pertenencia, $\mu_F(x)$, están en la generalidad de los casos, asociados con términos que aparecen como antecedentes o consecuentes de reglas, como las que se indican a continuación:

- *Si* (IF) estamos rastreando un objetivo en un instante de tiempo, *entonces* (THEN) el objetivo no estará *tan lejos* en el próximo instante de tiempo [$\mu_{\text{TAN-LEJOS}}(x)$];

- *Si* (IF) *y(t)* es *cercano a* 0.5, *entonces* (THEN) *f(y)* es *cercano a cero* [$\mu_{\text{FCERCANO-A-0.5}}(y)$, $\mu_{\text{FCERCANO-A CERO}}((f(y))$].

Las formas de curvas más utilizadas para las funciones de pertenencia son triangulares, trapezoidales, líneas rectas , curvas Gausianas. Pero las funciones de pertenencia son como ya se mencionó anteriormente escogidas de manera arbitraria, basándose en la experiencia del usuario o diseñador. Por lo tanto las funciones de pertenencia para dos diseñadores pueden ser completamente diferentes dependiendo de sus experiencias, perspectivas, culturas etc [4].

1.2.5 Operaciones de Conjuntos

Ahora que se ha definido lo que son los conjuntos difusos, entonces podemos hacer muchas cosas de la misma manera que podemos hacer con los conjuntos tradicionales, haciendo una analogía con estos.

1.2.5.1 Operaciones con Conjuntos Tradicionales.

Sea U el espacio de puntos (objetos), con un elemento genérico de U denotado por x. Consideremos A y B ser dos Subconjuntos de U.

UNION.- la unión de A y B, denotado por $A \cup B$ contiene todos los elementos en A o en B, así;

$$\mu_{(A\cup B)}(x)= \begin{cases} 1 & \text{Si } x \in A \text{ o } x \in B \\ 0 & \text{Si } x \notin A \text{ y } x \notin B \end{cases} \qquad (1\text{-}2)$$

INTERSECCION.- la intersección de A y B , denotado por A∩B, contiene todos los elementos que están simultáneamente en A y en B, así;

$$\mu_{(A\cap B)}(x) = \begin{cases} 1 & \text{Si } x \in A \wedge x \in B \\ 0 & \text{Si } x \notin A \vee x \notin B \end{cases} \qquad (1\text{-}3)$$

COMPLEMENTO.- el complemento de A denotado A', contiene todos los elementos que no están o no pertenecen a A, es decir

$$\mu_{A'}(x) = \begin{cases} 1 & \text{Si } x \notin A \\ 0 & \text{Si } x \in A \end{cases} \qquad (1\text{-}4)$$

De estas definiciones se puede mostrar que:

$$A \cup B \Rightarrow \mu_{A\cup B}(x) = max\,[\mu_A(x), \mu_B(x)] \qquad (1\text{-}5)$$

$$A \cap B \Rightarrow \mu_{A\cap B}(x) = min\,[\mu_A(x), \mu_B(x)] \qquad (1\text{-}6)$$

$$\mu_{A'}(x) = 1 - \mu_A(x) \qquad (1\text{-}7)$$

podemos demostrar que:

$$\mu_{A\cup B}(x) = max\,[\mu_A(x), \mu_B(x)]$$

$x \in A$ o $x \in B$ significa que:

$$(\mu_A(x) = 1,\ \mu_B(x) = 1)$$

$$(\mu_A(x) = 1,\ \mu_B(x) = 0)$$
$$(\mu_A(x) = 0,\ \mu_B(x) = 1)$$

Por lo tanto:

$$max\ [\mu_A(x),\ \mu_B(x)] = 1$$

$x \notin A$ y $x \notin B$ significa que:

$$(\mu_A(x) = 0,\ \mu_B(x) = 0)$$

Por lo tanto:

$$max\ [\mu_A(x),\ \mu_B(x)] = 0$$

Consecuentemente:

$$max\ [\mu_A(x),\ \mu_B(x)]$$

nos provee la correcta función de pertenencia para la Unión $(A \cup B)$.

1.2.5.2 Operaciones con Conjuntos Difusos [6]

Sea U el espacio de puntos (objetos), con un elemento genérico de U denotado por x.

Los *conjuntos Difusos* (*clases*) A y B en U se caracterizan por una Función de pertenencia (característica) $\mu_A(x)$ y $\mu_B(x)$ representando los grados de pertenencia de x en A y B respectivamente.

Las siguientes definiciones son obvias extensiones de las correspondientes definiciones de los conjuntos tradicionales.

1.2.5.2.1 Conjuntos especiales

Un conjunto difuso es *vacío si y solamente si su función de pertenencia es idénticamente cero en U.*

IGUALDAD.- dos conjuntos difusos A y B son iguales, denotados como A=B, si y solamente si:

$$\mu_A(x) = \mu_B(x) \qquad \forall x \in U \tag{1-8}$$

INCLUSIÓN .- A está contenido en B (o, equivalentemente, A es un subconjunto de B, o A es más pequeño o igual que B) si y solamente si :

$$A \subset B \Leftrightarrow \mu_A(x) \leq \mu_B(x) \tag{1-9}$$

Operaciones:

UNIÓN.- La unión de dos conjuntos difusos A y B con sus respectivas funciones de pertenencia $\mu_A(x)$ y $\mu_B(x)$ es otro conjunto difuso $C = A \cup B$ cuya función de pertenencia relaciona aquellas funciones de pertenencia de A y de B.

$$A \cup B \Rightarrow \mu_{A \cup B}(x) = max\,[\mu_A(x), \mu_B(x)] \tag{1-10}$$

Notar que $\cup$ tiene la propiedad Asociativa, es decir:

$$A \cup (B \cup C) = (A \cup B) \cup C$$

INTERSECCIÓN.- La intersección de dos conjuntos A y B con sus respectivas funciones de pertenencia $\mu_A(x)$ y $\mu_B(x)$ es otro conjunto difuso $C = A \cap B$ cuya función de pertenencia relaciona aquellas funciones de pertenencia de A y de B.

$$A \cap B \Rightarrow \mu_{A \cap B}(x) = min\,[\mu_A(x), \mu_B(x) \tag{1-11}$$

COMPLEMENTO.- el complemento de un conjunto difuso A denotado por A', está definido por:

$$\mu_{A'}(x) = 1 - \mu_A(x) \qquad (1\text{-}12)$$

Tanto la Unión como la Intersección de dos conjuntos difusos se ilustran en la siguiente figura.

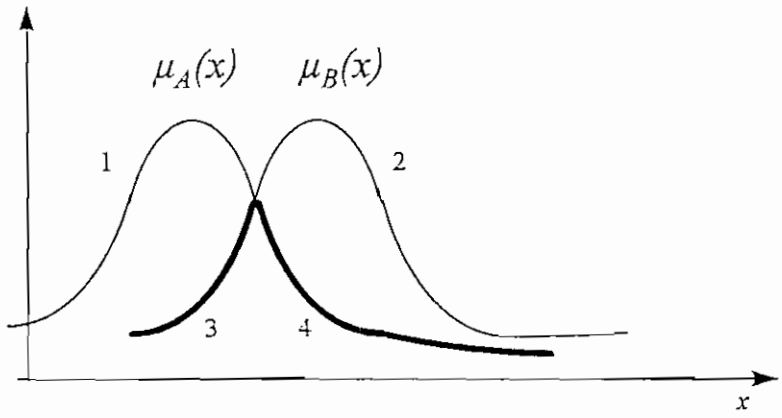


Fig. 1.5 Ilustración de la unión e intersección de conjuntos difusos.

La función de pertenencia de la unión está compuesta de los segmentos de curva 1 y 2; la función de pertenencia de la intersección está compuesta de los segmentos de curva 3 y 4 (líneas más gruesas).

La unión de A y B es el conjunto Difuso más pequeño que contiene tanto a A como a B.

La intersección de A y B es el conjunto difuso más grande que está contenido tanto en A como en B.

1.2.5.2.2 Propiedades de la Unión, Intersección, y complemento

Se puede extender muchas de las identidades básicas las cuales se relacionan con los conjuntos tradicionales.

Asociativa:

$$A \cup (B \cup C) = (A \cup B) \cup C \qquad (1\text{-}13)$$
$$A \cap (B \cap C) = (A \cap B) \cap C$$

Distributiva: Sean A, B, y C tres conjuntos difusos.

$$A \cap (B \cup C) = (A \cap B) \cup (A \cap C) \qquad (1\text{-}14)$$
$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$$

Leyes de Morgan:

$$(A \cup B)' = A' \cap B' \qquad (1\text{-}15)$$
$$(A \cap B)' = A' \cup B'$$

1.2.6 Lógica Difusa

La noción central en los sistemas difusos son los ***valores de verdad*** (en Lógica Difusa) o los valores miembros (en Conjuntos difusos), los que se encuentran como ya se estudió anteriormente en el rango de [0.0, 1.0], de tal forma que 0.0 representa Falsedad Absoluta y 1.0 representa verdad absoluta.

La lógica Difusa al igual que la teoría de los Conjuntos Difusos se basa en las nociones de la Lógica proposicional tradicional . Uno de los mayores componentes que intervienen en la lógica Difusa son *Reglas*. Estas reglas serán expresadas como implicaciones lógicas, por ejemplo de formas de sentencias *Si-Entonces (IF-THEN)*.

1.2.6.1 Lógica Tradicional

En este campo las reglas están en forma de proposiciones. Una *proposición* es una expresión que involucra términos, que han sido definidos, por ejemplo "El valor de el ángulo θ es cercano a 90° ." Consecuentemente , podemos obtener la siguiente regla: "Si (IF) el ángulo θ es cercano a 90°, Entonces (THEN) la función senθ tiende a un valor

cercano a 1. En la Lógica proposicional tradicional, una proposición debe ser clara, comprensible para poder llamarla "verdadera" o "falsa," si o no.

El razonamiento lógico es el proceso de combinar proposiciones dadas para llegar a otras proposiciones. Las proposiciones se pueden combinar de distintas formas, utilizando las tres operaciones fundamentales: La *Conjunción* denotada por $(p \wedge q)$, *Disyunción* denotada por $(p \vee q)$, e *Implicación* denotado por $(p \rightarrow q)$ la cual toma la forma de una regla Si-Entonces (IF-THEN). La parte correspondiente a Si (IF) de una implicación se denomina *antecedente*, mientras la parte correspondiente al Entonces (THEN) se denomina *consecuente*. Además se tiene la operación de *Negación* denotada por $(\sim p)$.

Una tabla de verdad es conveniente para mostrar las relaciones entre las proposiciones.

P	q	p∧p	p∨q	p→q	p↔q	~p
V	V	V	V	V	V	F
V	F	F	V	F	F	F
F	V	F	V	V	F	V
F	F	F	F	V	V	V

Fig.1.6 Tabla de verdad para operaciones aplicadas a las proposiciones

Se ha añadido la relación de equivalencia ($\leftrightarrow$) que significa que tanto p y q deben ser verdaderas o falsas.

Una tautología es una proposición que resulta de la combinación de proposiciones (p, q, r,...). Una tautología importante para nuestro trabajo es:

$$(p \rightarrow q) \leftrightarrow \sim (p \wedge (\sim q)) \qquad (1\text{-}16)$$
$$(p \rightarrow q) \leftrightarrow \sim p \vee q$$

La importancia de éstas tautologías es que ellas nos permiten expresar la función de pertenencia para $(p \rightarrow q)$ en términos de cada una de las proposiciones p y $(\sim q)$ o $\sim(p)$ y q, lo cual es el principal objetivo de esta sección.

Algunas de las importantes equivalencias matemáticas.entre la lógica y la teoría de conjuntos son:

Lógica	Teoría de conjuntos
$\wedge$	$\cap$
$\vee$	$\cup$
$\sim$	$(')$

En la lógica proposicional tradicional existen dos reglas de inferencia que son importantes. *Modus Ponens* y el *Modus Tollens*.

Modus Ponens.- En términos de proposiciones p y q . El Modus Ponens se expresa de la siguiente manera:

$$(p \wedge (p \rightarrow q)) \rightarrow q \qquad (1\text{-}17)$$

Modus Tollens.- En términos de proposiciones p y q, Modus Tollens se expresa de la siguiente manera:

$$(\sim q \wedge (p \rightarrow q)) \rightarrow \sim p \qquad (1\text{-}18)$$

En aplicaciones para la ingeniería Modus Ponens juega un papel muy importante.

1.2.6.2 Reglas de Inferencia difusas

La Lógica Difusa [4] [8],al igual que la teoría de conjuntos difusos, pide prestado las nociones de la Lógica tradicional para su desarrollo, sin embargo para aplicaciones en ingeniería de la lógica difusa es inadecuado hacer esto, ya que en ingeniería, *la causa y efecto es la piedra angular de la modelación, mientras que en la lógica proposicional*

tradicional no lo es [4]. Esto permite definir los operadores de implicación para ingeniería como se verá más adelante.

Como en la extensión que se hizo de la teoría de conjuntos tradicionales a la teoría de los conjuntos difusos, se puede extender de la misma manera la lógica tradicional a la lógica difusa mediante el reemplazo de las funciones de pertenencia bivalentes con las funciones de pertenencia difusas.

Uno de los mayores componentes en los sistemas de lógica difusa son las reglas, las mismas que serán expresadas como implicaciones lógicas, es decir de la forma de sentencias Si-Entonces (IF-THEN).

"Si (IF) u es A, entonces (THEN) v es B"

Donde : $u \in U$ y $v \in V$

la cual tiene una función de pertenencia

$$\mu_{A\rightarrow B}(x,y)$$

donde $\mu_{A\rightarrow B}(x,y)$ mide el grado de verdad de la relación de implicación entre x y y.

Utilizando las nociones de la lógica tradicional, las equivalencias entre la lógica y la teoría de conjuntos y las tautologías definidas anteriormente se pueden obtener las siguientes funciones de pertenencia para $\mu_{A\rightarrow B}(x,y)$.

Sabiendo que: $(p\rightarrow q) \leftrightarrow \sim(p \wedge (\sim q))$

Se obtiene:

$$\mu_{A\rightarrow B}(x,y) = 1 - min[\, \mu_A(x),\ 1 - \mu_B(y)] \qquad (1\text{-}19)$$

Sabiendo que: $(p\rightarrow q) \leftrightarrow \sim p \vee q$

Se obtiene:

$$\mu_{A\rightarrow B}(x,y) = max\,[\,1 - \mu_A(x),\ \mu_B(y)\,] \qquad (1\text{-}20)$$

En Lógica Difusa, el Modus Ponens se extiende, al *Modus Ponens Generalizado*, es decir por ejemplo:

Premisa 1 : "u es A^* "
Premisa 2 : "Si u es A entonces v es B "
Consecuencia: "v es B^* "

Para explicar lo anterior consideremos el siguiente ejemplo:

Consideremos la *regla* "SI un hombre es pequeño, ENTONCES no será un muy buen portero de fútbol profesional."
El conjunto difuso A es *hombre pequeño.*
El conjunto difuso B es *no será un muy buen portero de fútbol profesional.*

Ahora podemos dar la *premisa 1*:
"Este hombre tiene menos de 1.65 cm. de estatura".
Aquí A^* es el conjunto difuso *hombre de menos de 1.65 cm. de estatura.* Por lo tanto se puede observar que :
$A^* \neq A$
Pero se puede notar que A^* es similar a A.
Ahora se puede deducir la siguiente consecuencia:
"El será un portero de fútbol profesional deficiente."
Aquí B^* es el conjunto difuso *será un portero de fútbol profesional deficiente*, y
$B^* \neq B$
Pero B^* es realmente similar a B.

En la lógica tradicional una regla se disparará solamente si la primera premisa es exactamente la misma que el antecedente de la regla, y, el resultado de tal disparo de la regla es el consecuente actual de la regla.

En la lógica difusa , una regla se dispara siempre que haya un grado de similaridad no-cero entre la primera premisa y el antecedente de la regla, y, el resultado de tal disparo de la regla es un consecuente que tiene un grado de similaridad no-cero con el consecuente de la regla.

1.3. UNA APROXIMACIÓN ALGORÍTMICA DIFUSA PARA LA DEFINICIÓN DE CONCEPTOS COMPLEJOS E IMPRECISOS.

La mayoría de conceptos encontrados en varios dominios del conocimiento humano son en realidad muy complejos de aceptar por definición simple o por definición precisa. Por ejemplo ; los conceptos de recesión y utilidad en economía, la artritis en medicina, verdad y casualidad en filosofia; inteligencia y creatividad en psicología.

La aproximación provee un sistema para la definición de estos conceptos a través del uso de algoritmos difusos los cuales tienen una estructura de un cuestionario ramificado.

Los altos niveles de precisión prevalecen en las ciencias exactas como las matemáticas, la física, la química, la ingeniería entre otras, en cambio en las ciencias como la sociología, psicología, historia, filosofia, etc. se extiende un nivel de imprecisión. Esta marcada diferencia en los niveles de precisión se debe a que las ciencias exactas tienen un comportamiento que puede ser descrito en términos cuantitativos, mientras que en las otras ciencias el juicio humano, la percepción, y las emociones juegan un papel muy importante [16], [17].

En la aproximación lingüística (Zadeh), la cual representa un alejamiento de los métodos convencionales, se utilizan palabras o expresiones en lugar de números para describir fenómenos que son tan complejos de ser caracterizados en términos cuantitativos.

Una característica clave de la aproximación lingüística tiene que ver con la noción de los conjuntos difusos, lo cual nos sirve como unidad de medida para analizar los sistemas humanísticos, los cuales no pueden ser descritos en términos cuantitativos.

En la aproximación lingüística donde los conjuntos difusos pueden aplicarse, se pueden generar otros conjuntos a través del uso de modificadores lingüísticos tales como *muy*, *completamente*, *más o menos*, *extremadamente*, etc.

El propósito principal de la aproximación lingüística es la definición de conceptos tan complejos o imprecisos que son susceptibles de una definición exacta. Tales conceptos son difusos en el sentido de que ellos corresponden a clases de objetos o ideas que no tienen límites definidos, por ejemplo el concepto de *óvalo*, *amor*, *joven*, *masculino*, etc. mientras que aquellos conceptos como el de la *línea recta*, *hermano*, *casado*, etc. no son difusos. Se puede notar claramente que *óvalo* es un concepto más complejo que *línea recta*. Verdaderamente los conceptos complejos tienden a ser difusos.

Como se verá más adelante, una definición algorítmica difusa tiene la forma de un cuestionario ramificado, Q, en la cual tanto a las preguntas como las respuestas se les permiten ser difusas. Por ejemplo, una pregunta como: *"Es Adriana alta?"*, la respuesta puede ser *"bastante alta"*, esto puede ser visualizado como ser equivalente a la asignación del valor lingüístico "bastante alta" en el grado de pertenencia de Valentina en la clase de gente alta.

1.3.1. Preguntas Atómicas

Nuestra discusión de la definición algorítmica difusa comenzará con la noción de pregunta atómica. Esta noción servirá como una base para la definición de una pregunta compuesta, la que en su turno conducirá al concepto de un cuestionario ramificado algorítmico difuso.

Una pregunta *atómica* no tiene otros constituyentes que si mismo, por el contrario, una pregunta *compuesta* como su nombre lo indica, está compuesta de una colección de preguntas constituyentes.

Por ejemplo la pregunta Q = Es Rosa alta? es una pregunta atómica si ningunas otras preguntas tienen que ser hechas para poder contestar Q.

NOTACION.- Una pregunta atómica, Q, se caracteriza por una tripleta

$$Q = (X, B, A) \qquad (1\text{-}21)$$

donde:

X= *conjunto-objeto*

esto es un conjunto de objetos al que se le aplica Q.

B= *cuerpo de* Q

esto es una etiqueta de una clase o un atributo.

A= *conjunto-respuesta*

es decir un conjunto de respuestas admisibles a la pregunta.

Cuando sea necesario especificar ejemplos de Q, X, y A estos serán denotados por *q*, *x*, y *a* respectivamente. Cuando X y A están implícitos , Q será escrito como:

$Q \cong B?$

Una pregunta específica junto con una respuesta admisible a esta será expresada como

$Q/A \cong B?\ a$

El par Q/A se referirá a un *par pregunta/respuesta.*

Gráficamente se puede representar a una pregunta atómica como un abanico.

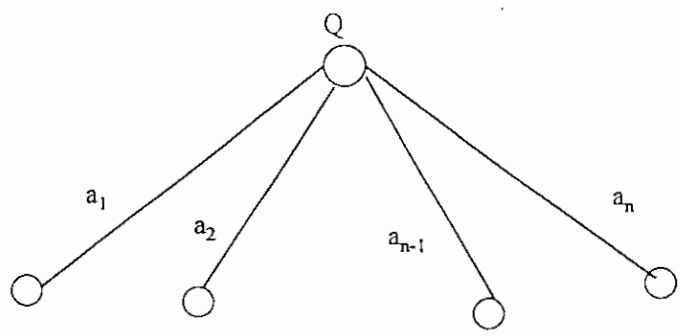


Fig 1.7 Representación gráfica de una pregunta atómica.

1.3.1.1 Preguntas clasificacionales y Atribucionales

Una pregunta Qi, en Q puede ser *clasificacional* o *atribucional* [15]. En el caso de una pregunta clasificacional, Qi se interesa en el grado de pertenencia del tema o asunto en un conjunto difuso Fi. Aquí su cuerpo B, es la etiqueta de un conjunto difuso o no difuso.

En el caso de una pregunta atribucional, Qi se refiere al valor de un atributo de un tema o asunto. La respuesta a esta pregunta puede ser numérica la cual representa el valor del atributo, B, o lingüística. En este caso B , es la etiqueta de un atributo.

En el caso de una pregunta clasificacional, una respuesta, ***a***, representa el grado de pertenencia de x en el conjunto difuso B. La respuesta puede ser numérica o lingüística.

Por ejemplo cuando la respuesta es numérica, entonces a≅ 0.8, o linguística, por ejemplo, a≅alto.

En el caso de una pregunta atribucional, Q ≅B?, una respuesta, a, representa el valor del atributo, B, de un objeto x, por ejemplo B ≅edad y x ≅Lourdes, de nuevo, a, puede ser, a ≅35, o linguístico, a ≅muy joven.

1.3.1.2 Relación entre preguntas Clasificacionales y Atribucionales.

En el caso de una pregunta clasificacional no-difusa, el conjunto-respuesta, **A**, tiene solamente dos elementos los cuales son generalmente designados como {si, no},{verdad, falso} o {0, 1}. Por el contrario, el conjunto-respuesta de una pregunta atribucional es generalmente continuo en U o un conjunto contable de valores lingüísticos definidos en U. De esta manera, en general, una respuesta a una pregunta atribucional lleva considerablemente más información que una respuesta a una pregunta clasificacional no-difusa.

En el caso de preguntas clasificacionales difusas, sin embargo, el conjunto-respuesta puede ser el intervalo unidad [0, 1] o un conjunto contable de valores

lingüísticos definidos sobre [0, 1]. En tales casos, la distinción entre preguntas clasificacionales y atribucionales es menos definida y, de hecho, existirá equivalencia entre ellas.

1.3.2 Preguntas Compuestas y su representación

El concepto de una pregunta atómica ya vista anteriormente nos provee una base para la definición de un concepto más general de una pregunta compuesta.

Se establece que una ***pregunta compuesta n-adica Q, con cuerpo B,*** *es una pregunta compuesta de n preguntas constituyentes* Q_1, ... Q_n *con cuerpos* B_1, ...B_n, *respectivamente, tal que la respuesta a Q es dependiente de las respuestas a* Q_1, ... Q_n, así una pregunta monádica tiene una sola constituyente, una pregunta diádica tiene dos constituyentes, una pregunta triádica tiene tres constituyentes, etc. Una pregunta constituyente puede ser atómica o compuesta.

Una pregunta compuesta n-adica o, simplemente una pregunta n-adica, Q, se caracteriza por su representación relacional, $B(B_1, ..., B_n)$ (o simplemente B, cuando no exista confusión con el cuerpo B de Q), la cual puede representarse mediante una tabla como la siguiente;

Q_1	Q_2	Q_j	Q_n	Q
r_{11}	r_{12}	r_{1j}	r_{1n}	r_1
r_{21}	r_{22}	r_{2j}	r_{2n}	r_2
.	.	.	.	.
.	.	.	.	.
r_{i1}	r_{i2}	r_{ij}	r_{in}	ir
.	.	.	.	.
.	.	.	.	.
r_{m1}	r_{m2}	r_{mj}	r_{mn}	r_m

Fig. 1.8 Representación Relacional de Q

En esta tabla r_{ij} y r_i recogen todas las respuestas admisibles a Q_j y Q, respectivamente, con A_j y A representando los conjuntos respuesta asociados con Q_j y Q, y, a_j y a denotan sus elementos genéricos.

Si Q es una pregunta n-adica, entonces B es una relación no difusa (n+1) aria del producto cartesiano $A_1 \times \ldots \times A_n$ en A. En particular, si Q es una pregunta monádica, entonces B es una relación Binaria.

Generalmente las entradas en B son lingüísticas por naturaleza, por ejemplo son valores de atributo lingüístico y/o valores de verdad lingüístico y/o grados de pertenencia lingüística.

Una representación algebráica de la i-ésima fila de la tabla de B puede expresarse como una *secuencia* Q/A de la forma siguiente

$$Q1r_{i1}Q2r_{i2}\ldots Qnr_{in}//Qr_i \qquad i=1,\ldots m \qquad (1\text{-}22)$$

de una manera más simple como una *fila* de Q/A

$$r_{i1}\ r_{i2}\ldots r_{in}//\ r_i \qquad (1\text{-}23)$$

donde:

$$r_{ij} \qquad j = 1, \ldots, n$$

es una respuesta admisible a la pregunta constituyente Qj, y

$$r_i$$

es una respuesta admisible a la pregunta compuesta Q.

B como un todo puede expresarse como la suma o Unión de las filas Q/A de la siguiente manera:

$$B= r_{11}r_{12}\ldots r_{1n}//r_1 + r_{21}r_{22}\ldots r_{2n}//r_2 + \ldots\ r_{m1}r_{m2}\ldots r_{mn}//r_m \qquad (1\text{-}24)$$

o, de la manera más compacta

$$B= \Sigma_i (r_{i1}r_{i2}\ldots r_{in}//r_i) \qquad (1\text{-}25)$$
$$i= 1\ldots m$$

1.3.2.1. Significado de B

Hemos estado hablando en la sección 1.3.2 acerca de la representación relacional de B, pero qué es lo que realmente constituye B. Para dilucidar esta pregunta analicemos conjuntos. Supongamos subconjuntos $W_1W_2\ldots W_n$ de un universo en consideración U.

Consideremos dos conjuntos no difusos G y H

$$G= \{ g_1,g_2, \ldots,g_m \}=g_1+g_2+\ldots+g_m$$
$$H= \{ h_1,h_2, \ldots, h_n\}= h_1+h_2+\ldots+h_n$$

Ahora definimos el Producto Carteciano G×H

$$G\times H=GH$$
$$G\times H = (g_1+g_2+\ldots+g_m)\times(h_1+h_2+\ldots+h_n) \qquad (1\text{-}26)$$
$$= (g_1h_1+g_2h_2+\ldots+g_m h_n) \qquad (1\text{-}27)$$

donde (+) expresa unión y g_ih_j deben ser interpretados como pares ordenados (g_i,h_j).

Ahora supongamos que G y H son conjuntos difusos finitos definidos por

$$G= \mu_1/g_1+\ldots+\mu_m/g_m$$
$$H= v_1/h_1+\ldots+v_n/h_n$$

donde:

$$\mu_i/g_i$$

indica que el grado de pertenencia de g_i en G, es μ_i. De la misma manera para H. Entonces, el producto cartesiano de G y H se obtiene

$$G\times H=(\mu_1/g_1+\ldots+\mu_m/g_m) \times (v_1/h_1+\ldots+v_n/h_n) \qquad (1\text{-}28)$$
$$= \mu_1\wedge v_1/g_1h_1+\ldots+\mu_m\wedge v_n/g_mh_n \qquad (1\text{-}29)$$

donde de acuerdo con las operaciones de conjuntos difusos

$$\mu_i \wedge v_j = min\ (\mu_i, v_j)$$

De una manera general, considerando G_1. ..., G_n subconjuntos difusos de U_1, ..., U_n definidos por

$$Gj = \sum_{i=1}^{mj} uij/vij \qquad (1\text{-}30)$$

entonces se obtiene

$$G_1 \times \ldots \times G_n = \sum_i \mu^1_{i1} \wedge \ldots \wedge \mu^n_{in} / v^1_{i1} \ldots v^n_{in} \qquad (1\text{-}31)$$

Regresando a la pregunta de qué constituye B, enfoquemos la expresión algebráica de B expresada en (1-24), en la cual ya podemos de una manera más objetiva emplear las definiciones del producto cartesiano y definir diciendo que *cada término en la ecuación que define B es un producto cartesiano de conjuntos difusos, matemáticamente definida por (1-31), y B como un todo es la unión de tales productos cartesianos.*

1.3.2.2 Cuestionarios Ramificados

El concepto de cuestionario ramificado juega un papel importante en el campo del Reconocimiento de Patrones. El término cuestionario ramificado será utilizado para referirnos a una representación de una pregunta compuesta, Q = B ?, en la cual las preguntas constituyentes Q_1, ..., Q_n son presentadas (preguntadas) en un orden que es determinado por las respuestas de las preguntas previas.

Una representación de un cuestionario ramificado de Q = B ? será denotado por Q^*. Un cuestionario ramificado Q*, puede ser representado de una forma de un árbol como se muestra en la siguiente figura;

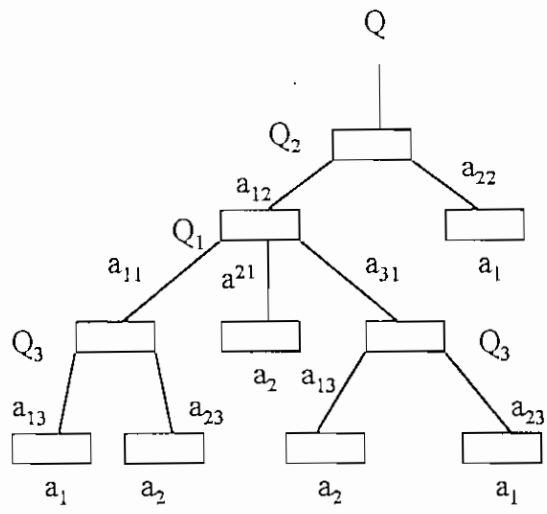


Fig. 1.9 Ejemplo de un cuestionario ramificado

La raíz de este árbol es etiquetada con el nombre de la pregunta compuesta, Q, o con el nombre del cuerpo de Q; las hojas son etiquetadas con las respuestas admisibles para Q; y los nodos internos son etiquetados con los nombres de las preguntas constituyentes. Así, de esta manera cada abanico del árbol representa una pregunta constituyente, con cada rama del abanico correspondiendo a una respuesta admisible a la pregunta. Si una rama como a_{12} de la pregunta Q_2 termina en Q_1, significa que la respuesta a la pregunta Q_2 es a_{12}, entonces la próxima pregunta a ser presentada o cuestionadaes Q_1. Luego si la respuesta a Q_1 es a_{31}, entonces la próxima pregunta a ser presentada o cuestionada es Q_3, y si la respuesta a Q_3 es a_{13}, entonces la respuesta a Q es a_2. Esta secuencia se puede observar en el cuestionario ramificado del ejemplo.

Cada camino desde la raíz del árbol a una hoja representa una secuencia particular Q/A, por ejemplo

$$\mathbf{Q_2a_{12}Q_1a_{11}Q_3a_{13}//Qa_1} \qquad 1\text{-}32$$

lo cual puede escribirse de una manera más simple como:

$$\mathbf{a_{12}a_{11}a_{13}//a_1} \qquad (1\text{-}33)$$

Es importante notar que la única condición en la estructura de un cuestionario ramificado es que, en cualquier camino desde la raíz a una hoja, cada pregunta constituyente se encuentre a lo sumo una sola vez.

La unión (Suma) de todas las secuencias Q/A de la forma (1-33) constituye una representación algebraica de Q*. Así para el cuestionario ramificado de la figura mostrada anteriormente se tiene la siguiente representación

$$Q^* = a_{12}a_{11}a_{13}//a_1 + a_{12}a_{11}a_{23}// a_2 + a_{12}a_{21}//a_2 + a_{12}a_{31}a_{13}// a_2 \quad + \quad a_{12}a_{31}a_{23}//a_1 + a_{22}//a_1 \qquad \textbf{(1-34)}$$

Referencias:

[1] Bezdek, J. C. y Pal, S. K. *Fuzzy Model for Pattern Recognition*, New York: *IEEE*, Inc, 1992

[2] Cox, Earl D. The Fuzzy Systems Handbook: A Practitioner's Guide to Building, Using and Maintaining Fuzzy Sistems, New York, 1994.

[3] B. Kosko, *Neural Networks and Fuzzy Systems: A Dinamical Systems Approach to Machine Inteligence*, Prentice-Hall, Inc, 1992.

[4] Jerry M. Mendel., *Fuzzy Logic Systems for Engineering*: A Tutorial, IEEE,1995.

[5] L. A. Zadeh, "The concept of a Linguistic variable and its Application to approximate reasoning," *Inf Sciences*, vol. 8, pp. 199-249, and vol. 9, pp. 43-80, 1975.

[6] L. A. Zadeh, "Fuzzy Sets," *Information and Control*, vol. 8, pp 338-353, 1965.

[7] H. J. Zimmermann, Fuzzy Set Theory and its Applications, 2nd de. Boston, MA: Kluwer, 1991.

[8] R. R. Yager, " On the implication operator in Fuzzy Logic," *Inf Sciences*, vol. 31, pp. 141- 164, 1983.

[9] J. Maiers and Y. S. Sherif, "Applications of Fuzzy Set Theory," *IEEE Trans. Syst.., Man, and Cyber.*,vol SMC-15, pp 175-189, Jan./Feb . 1995.

[10] Klir, George J., and Folger, Tina A. (1988). *Fuzzy Sets, Uncertainty, and Information*, Prentice Hall, Englewood Cliffs, NJ.

[11] Kosko, Bart and Isaka, Satoru (1993). "Fuzzy Logic," *Scientific American* (July), pp. 76-81.

[12] D. Dubois and H. Prade, "Fuzzy sets in approximate reasoning, part 1: Inference with possibilty distributions," *Fuzzy sets Syst.*, vol. 40, 1991, pp. 143-202.

[13] Jamshidi, M,. Vadiee, N., and Ross, Timothy J., Eds. (1993*). Fuzzy Logic and Control.* Prentice Hall, Englewood Cliffs, NJ.

[14] Cox, Earl D. (1992). "Fuzzy Fundamentals." *IEEE Spectrum* (October), pp.58-61.

[15] Cox, Earl D. (1992). "Applications of Fuzzy System Models." *AI Expert* (October), pp. 34-39.

[16] L. A. Zadeh. Fuzzy Algorithmic-Aproach, *Int. J. Man Machine Studies* (1976) 8,249-291.

[17] L. A. Zadeh. (1972). A Fuzzy-set-theoretic interpretation of linguistic hedges. *Journal of Cybernetics*,2 ,4.

2.1 INTRODUCCIÓN

La mayoría de sistemas de escritura han sido desarrollados históricamente en base a sistemas de gráficos que fueron trazados usando barras en materiales tales como la arcilla. Por lo tanto estos cuentan con rasgos que pueden ser ligeramente curvados o rectos. Esto es particularmente verdad con sistemas que tienen un origen antiguo, tales como los alfabetos griego, latín, y los ideogramas chinos. Sistemas que se desarrollaron más tarde (cuando fue posible utilizar un pincel) pusieron menos énfasis en los rasgos bien definidos pero se puso énfasis en la continuidad [1].

En base a estos rasgos de antigua utilización para la definición de un sistema de escritura, se puede definir a los Rasgos Topológicos como aquellos que definen propiedades o particularidades que presenta un objeto en su configuración superficial independientemente de su tamaño y orientación. Por lo que, aplicando esta definición se pueden obtener características que definen de una u otra manera un carácter manuscrito.

2.2 CARACTERÍSTICAS DE LOS RASGOS TOPOLÓGICOS

La definición abstracta de símbolos del alfabeto puede ser muy simple, si se expresa éste en términos de sus rasgos. Por ejemplo, la letra **T** puede se definida como "un rasgo vertical recto con un rasgo horizontal recto que se balancea en la parte superior del rasgo vertical."

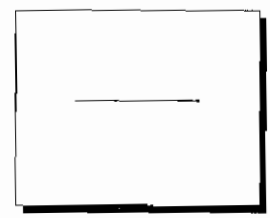
Cuando un símbolo tiene que ser escrito, debe tomarse en cuenta la definición abstracta de éste, pero sin rigurosidad matemática. Por ejemplo, el rasgo "vertical" no necesita ser una recta a 90°, se puede aceptar una ligera pendiente. Un "rasgo recto" no necesita ser exactamente recto; ya que una ligera curvatura puede ser aceptable. Si la continuidad del proceso de escritura es importante, la silueta puede ser más distorsionada para acomodar la conexión con otros símbolos.

La principal dificultad en la investigación y desarrollo de sistemas de reconocimiento de caracteres manuscritos está en la variedad de deformaciones de las siluetas o formas. En particular, a través de más de un cuarto de siglo de investigación,

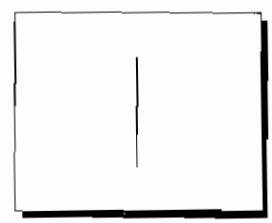
se encontró que algunos rasgos cualitativos tales como rasgos topológicos, o rasgos direccionales, son efectivos para representar varios tipos de las siluetas o formas [1].

RASGOS TOPOLÓGICOS FUNDAMENTALES

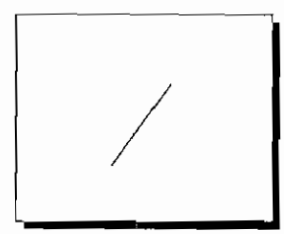
En las siguientes figuras se presentan los rasgos cualitativos fundamentales de los rasgos topológicos:



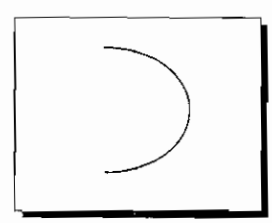
(a)



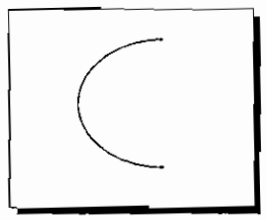
(b)



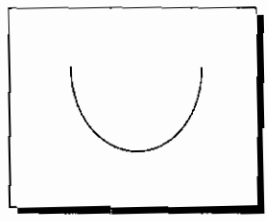
(c)



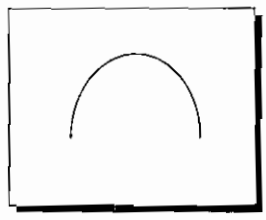
(d)



(e)



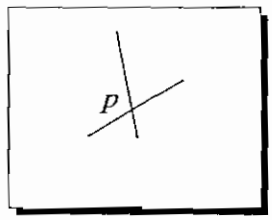
(f)



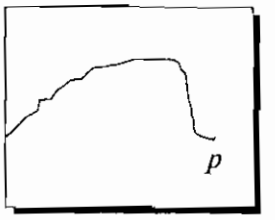
(g)

Fig. 2.1 Rasgos Topológicos fundamentales: a) Línea recta; b) Línea vertical; c) Línea inclinada; d) Curva con abertura hacia la izquierda; e) Curva con abertura a la derecha; f) Curva con abertura hacia arriba; g) Curva con abertura hacia abajo.

Además de estas características, dentro de los rasgos topológicos se deben incluir los Puntos Singulares. Es decir puntos de cruce y puntos finales, que se representan gráficamente a continuación:



(a)



(b)

Fig. 2.2 Puntos Singulares: a) Punto de cruce *p*; b) Punto final *p*.

En base a estas descripciones cualitativas, que serán las componentes de un carácter se describirán las formas de los caracteres.

2.2.1. Algunos Métodos de extracción de Rasgos [2]

En la búsqueda de alcanzar un alto rendimiento en un sistema de reconocimiento de caracteres, diferentes métodos de extracción de características o rasgos han sido desarrollados para representar a los caracteres manuscritos. Por lo tanto la selección de un método de extracción de rasgos es un factor muy importante. Pero qué método es el mejor para una aplicación dada?. Esta pregunta nos conduce a caracterizar unos pocos métodos a seleccionarsede tal forma que puedan ser escogidos, aunque para seleccionar uno u otro, la evaluación experimental nos daría la respuesta. Nuestro interés es el mostrar que existen varios métodos de extracción de rasgos que son aplicables al reconocimiento de caracteres numéricos manuscritos.

Cabe notar que muchos métodos de extracción de rasgos trabajan con imágenes en niveles de gris, pero para nuestro estudio mencionaremos métodos que trabajan con imágenes binarias.

La representación digital de una imagen en escala binaria es una simplificación de la representación de la imagen en escala de gris. Es decir, sea una función Z(x,y) la función de la imagen, la cual toma únicamente valores de 0 o 1 en vez de los 256 valores de niveles de gris. Por lo tanto, los métodos que se han desarrollado para imágenes con niveles de gris pueden ser aplicables en imágenes binarias.

■ Proyección de Histogramas [3]

La proyección de histogramas fue introducida en 1956 en un hardware de un sistema de reconocimiento de caracteres por Glauberman [3] .

La imagen es proyectada en el eje y, así como también en el eje x. Para una proyección horizontal $y(x_i)$ es el número de pixels[1] presentes cuando $x=x_i$. De la misma manera se obtiene en la proyección vertical el valor de $x(y_i)$ representando el número de pixels presentes cuando $y=y_i$. Esto lo podemos observar en la siguiente figura

Fig.2.3 Histogramas de proyección vertical y horizontal

Para el reconocimiento se mide la similaridad que existe entre dos histogramas que representan las características de dos caracteres.

■ Extracción de rasgos de un contorno [2]

La curva de un contorno[2] exterior cerrado de un carácter es una curva lineal cerrada que pasa por los centros de todos los pixels que están conectados al borde de la figura.

---

[1] Elemento más pequeño de una imagen digitalizada al que se le puede dar intensidad y color. Equivale al significado de un punto.

[2] Conjunto de puntos que limitan una figura

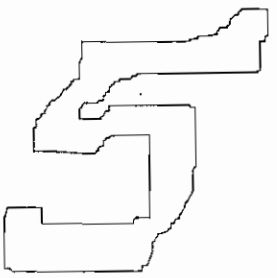


Fig. 2.4 Contorno del dígito 5

El contorno de la curva puede ser aproximado por una expresión paramétrica , cuyos coeficientes de la aproximación pueden ser utilizados como características o rasgos.

La motivación para usar los contornos es que el contorno puede ser aproximado por una función, la cual puede ser obtenida utilizando diferentes métodos. [4], [5], [6], [7].

■ **Rasgos Direccionales (Códigos de cadena) [8]**

Los Rasgos direccionales se usan para representar una frontera o contorno como un conjunto de segmentos con dirección y longitud especificadas. Es posible establecer rasgos direccionales con 4 , 8 o más direcciones.

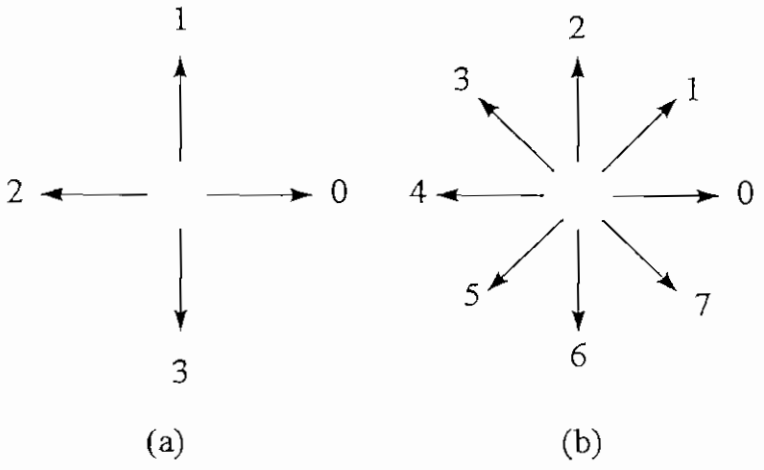
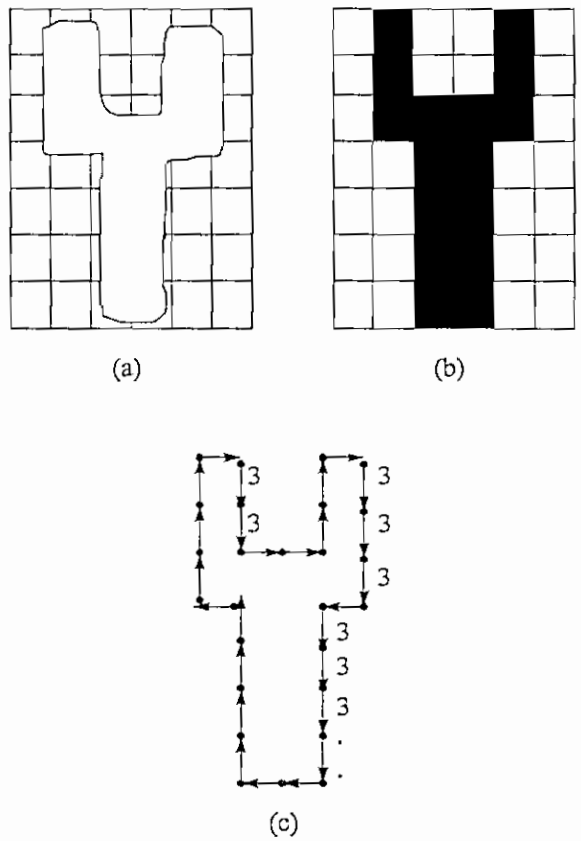


Fig.2.5 a) Código de cadena 4-direccional. b) Código de cadena 8-direccional

Para obtener los rasgos direccionales de una frontera dada en primer lugar se selecciona un tamaño de cuadrícula, en cuyo interior se encuentra el contorno del carácter, si una célula incluye más de una cantidad determinada de área interior a la frontera (por ejemplo 50%) se le asigna un 1 (tono oscuro) a dicha célula, caso contrario se le asignará el valor de 0. Finalmente se procede a codificar la frontera en color oscuro usando los códigos de dirección (códigos de cadena). Las siguientes figuras resumen este procedimiento.



Código de cadena:0330011033323 ...

Fig.2.6 (a), (b), y (c): Pasos para obtener el código de cadena

La figura 2.6c) se obtiene aplicando a la figura 2.6b) los códigos de cadena de la figura 2.5a).

Existen muchos otros métodos de extracción de características pero en las páginas que preceden solo se mencionan a un nivel general, ya que no está dentro del objeto de la presente tesis un estudio detallado de los mismos.

2.3 CATEGORIZACIÓN MULTINIVEL

Para poder explorar la esencia de la ingeniería de software, es importante conocer una técnica que permita a los ingenieros controlar la complejidad de un sistema, esta técnica se conoce como categorización multinivel o división de un proceso. Esto implica dividir el desarrollo de un sistema en niveles.

La categorización multinivel tiene que ver con las etapas que un programa de software de un sistema de reconocimiento de patrones tiene que llevar a cabo durante su ejecución, dependiendo del grado de dificultad que se tenga para dar una respuesta lógica. Si el procedimiento escogido no tiene los fundamentos necesarios para dar una salida confiable, se utiliza un algoritmo más sofisticado con nuevos datos o características que le permitan obtener la información necesaria para desplegar una salida correcta y confiable. Pero si tiene fundamentos suficientes en la primera etapa, puede sin más desplegar una salida confiable [9].

Esta técnica es considerada como un proceso de estrategia ascendente, la cual permite añadir nuevas unidades de programa (algoritmos) a partir de la base o primer nivel. De esta manera, podemos representar los niveles de construcción en función de los algoritmos que se agregan.

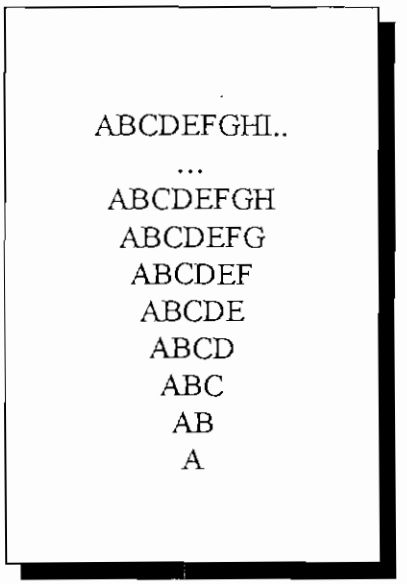


Fig. 2.7 Estrategia ascendente

Este proceso se lo puede explicar con un ejemplo relacionado con la medicina. Cuando un paciente necesita de un diagnóstico clínico para poder evaluar su enfermedad, los médicos empiezan por obtener información acerca de sus síntomas y signos. Si con esta información no se puede determinar la enfermedad que sufre el paciente, entonces se recurrirá a otro tipo de análisis como es la obtención de datos de laboratorio, luego serán los análisis de datos radiológicos, y en secuencia se irán obteniendo nuevos datos hasta que se pueda determinar de una manera exacta el tipo de enfermedad que sufre el paciente. Entonces lo que el diseñador (médico) hace es atacar el problema de diseño (diagnóstico) en varios niveles.

Cuando el objeto que se diseña es un programa de computación, la respuesta a la pregunta cuándo detener la ejecución, es aquella que permite dar una salida confiable al usuario.

Referencias:

[1] Rocha and T. Pavlidis, "A Shape Analysis Model with Applications to a Character Recognition System," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 16, no. 4, pp. 393-404, Apr. 1993.

[2] O. D. Trier, A. K. Jain, and T. Taxt, "Feature extraction methods for character recognition-a survey," *Pattern Recognition*, Vol 29, No 4, pp 641-662, 1996.

[3] M. H. Glauberman, Character Recognition for business machine, *Electronics*, Febraury 1956.

[4] F. Kimura and M. Shridhar, Handwritten numerical recognition based on multiple algorithms, *Pattern Recognition*, 1991.

[5] I. Sekita, K. Toraichi, R. Mori, K. Yamamoto and H. Yamada, Feature Extraction of handwritten Japanese characters by spline functions for relaxation matching, *Pattern Recognition*, **21** (1), 1998.

[6] C. S. Lin and C. L. Hwang, New forms of shape invariants from elliptic Fourier Descriptors, *Pattern Recognition* **20**(5), 1987.

[7] F. P. Kuhl and C. R. Giardina, Elliptic Fourier features of a closed contour, Comput. Vis. Graphics Image Process. **18**, 1982.

[8] K. S. Fu, R. C. Gonzalez and C.S.G. Lee, *Robótica, Control, Detección, Visión e Inteligencia*, Mc Graw-Hill, 1990.

[9] J. T. Tou and R. C. Gonzalez, "Recognition of handwritten characters by Topological feature extraction and multilevel characterization,"*IEEE Trans. Comput.*, vol C-21, July 1972.

[10] D. C. Ince. *Ingenieria de Software*, Adison-Wesley Iberoamericana, Wilmington, Delaware, E.U.A., 1993.

3.1 EL PROBLEMA DE RECONOCIMIENTO DE PATRONES

3.1.1. Nociones sobre el Reconocimiento de patrones y campo de aplicación

Se puede considerar que patrones *son los medios a través de los cuales se puede interpretar el mundo* [1]. Un niño aprende a distinguir los patrones visuales de sus padres, patrones auditivos de los sonidos de la naturaleza, de la música y de la palabra, así como también los patrones sensitivos de frío, calor, etc. A medida que el niño va creciendo es capaz de definir de una manera más detallada su reconocimiento de patrones, de tal forma que podrá distinguir una canción de Julio Jaramillo de una de José Luis Rodríguez, o una manzana de una pera. Algunos patrones tienen una manifestación física (tipos de caracteres, tipos de cosas o animales). Otros en cambio, solo se perciben de forma abstracta (patrones sociales o económicos).

Se puede definir al ***reconocimiento de patrones de una manera sencilla como la clasificación o separación de objetos (generalmente vectores en R^n) en grupos que contengan características similares*** [2].

Es importante notar que el reconocimiento de patrones sería simple, siempre y cuando los objetos de cada clase a clasificarse tengan características exactamente idénticas, pero esto en la realidad no ocurre, ya que en el mundo real más bien estas características entre objetos de la misma clase coinciden solo parcialmente, lo cual hace difícil la clasificación de los objetos de la misma clase, por tanto más difícil el reconocimiento de patrones.

Mayores problemas puede causar el hecho de que muchos objetos (vectores) tienen características similares a otros objetos de distinta clase, lo cual puede provocar que el sistema de reconocimiento cometa mayor número de errores en la clasificación. Por lo que el diseñador deberá trabajar más en el diseño de un *clasificador*[3] más preciso y consecuentemente más complejo.

[3] Se denomina clasificador a cualquier sistema capaz de asignar una clase (etiquetar) a todo vector de entradas en R^n, en base al establecimiento de regiones de decisión obtenidas de muestras de datos.

Debido a los extraordinarios resultados del reconocimiento sensorial de patrones efectuado por el hombre, existe cierta tendencia a esperar resultados automáticos de reconocimiento de patrones basado en algoritmos matemáticos diseñados y utilizados con la ayuda de un computador. Como en cualquier problema de tipo práctico, es necesario tener un gran cuidado en la preparación y análisis de los datos, así como en la implementación de los métodos apropiados..

En la última década se han incrementado los estudios e investigaciones referentes al reconocimiento de patrones, lo cual ha permitido que se disponga de información para obtener un avance considerable en este campo, y además que se desarrollen sistemas que lleven a cabo este tipo de tareas. El reconocimiento de patrones es por su naturaleza, una ciencia inexacta, por lo que admite muchas aproximaciones en la solución de un problema dado.

Se puede visualizar al Reconocimiento de patrones como un área de la investigación y desarrollo provocado por la necesidad de procesar datos e información obtenidas de la intercomunicación entre científicos, tecnólogos, y la sociedad.

Una motivación para el desarrollo de la actividad en este campo con gran ímpetu es la necesidad que tiene la gente de comunicarse con máquinas computadoras mediante un idioma natural. Otra y más importante motivación para el estudio de esta área, es que los científicos e ingenieros están preocupados de la idea de diseñar y construir máquinas inteligentes capaces de llevar a cabo tareas con habilidades comparables a la habilidad humana.

Es posible destacar, los campos más comunes de aplicación de las técnicas de reconocimiento de patrones en el siguiente cuadro [1], [2], [3].

APLICACIÓN	DATOS DE ENTRADA	RESPUESTA OBTENIDA
Reconocimiento de caracteres	Señales ópticas	Nombre del caracter
Reconocimiento auditivo	Señales auditivas o acústicas	Nombre de la palabra, identificación del orador, etc.

Reconocimiento de fotografías	Fotografías	Rechazo de fotografías que no sean de interés.
Biomedicina	Cardiogramas Encefalogramas,síntomas	Enfermedad
Análisis Meteorológicos	Humedad, turbulencia, presión, dirección y velocidad del viento, radiación solar de la superficie	Clima
Reconocimiento de siluetas	Señales emitidas por radar	Identificación de tipos de naves o aviones
Predicción económica	Series económicas de tiempo	Grado de inflación de un país para un siguiente periodo.
Identificación de Recursos naturales.	Fotografías tomadas por satélites o aviones convencionales.	Tipologías tales como oceanográficas, geológicas, mineras, agrícolas, forestales, etc.

3.1.2. Clasificación

La clasificación de patrones consiste en la separación de todo el espacio de entradas (R^n) en regiones de decisión de clase, de tal forma que una vez definidas dichas

regiones , el clasificador es capaz de asignar una clase (etiquetar) a cualquier vector de entrada en R^n.

3.1.3.Problemas fundamentales en el diseño de sistemas de reconocimiento de patrones

El diseño de un sistema de reconocimiento automático de patrones involucra ciertos problemas de consideración.

- Tiene que ver con la representación de los datos de entrada, es decir datos obtenidos de los objetos a ser reconocidos que serán luego entregados al clasificador.

  Existen varias formas de representar los objetos a ser clasificados: *representación sintáctica o simbólica*, la cual utiliza adjetivos, oraciones, etc.; *representación contextual, conceptual y basada en reglas*, que utiliza conceptos, conocimiento experto, etc.; *representación numérica* que utiliza vectores de datos del objeto (vectores de características).

  De los tipos de representación antes mencionados, el más utilizado es la representación numérica.

  Las características representativas de los patrones son descritos por cantidades medibles o cuantificables.

- Este problema tiene que ver con la extracción de características semejantes de los datos de entrada recibidos y la reducción de la dimensión del vector de patrones, lo que se logra mediante el análisis de las características. El análisis de las características utiliza métodos que nos permite explorar y mejorar los datos obtenidos en el proceso de obtención de datos. Los métodos utilizados pueden ser preprocesamiento, extracción de características.

  El preprocesamiento incluye operaciones como escalamiento, suavizamiento, adelgazamiento, y otras técnicas de limpieza.

El preprocesamiento facilita el trabajo del clasificador permitiendo que exista un mayor acercamiento entre vectores pertenecientes a una misma clase y a su vez que los vectores que pertenecen a clases distintas se separen más entre sí.

- Este problema involucra la determinación de funciones de decisión, que son fundamentales en la etapa de la identificación de los patrones.

El sistema diseñado tendrá como finalidad el reconocimiento de *n* diferentes clases de patrones (w_1, w_2, ..., w_n), de esta manera se puede generar un espacio con *n* regiones, que agruparán patrones de su misma clase.
Los Patrones a ser reconocidos por un sistema de reconocimiento automático deben poseer un conjunto de características cuantificables o medibles. Por lo tanto cuando esas medidas son idénticas o similares dentro de un grupo de patrones entonces es posible hablar de los miembros de una misma clase, es decir sus miembros son isomorficos.

La confiabilidad del sistema de reconocimiento depende de la cantidad de información relevante contenida en las mediciones obtenidas y de la utilización eficiente de esta información.

3.2. VISIÓN ARTIFICIAL

3.2.1. Introducción

La visión artificial puede ser definida como el proceso de obtención, caracterización e interpretación de información de imágenes tomadas de un entorno bidimensional o tridimensional. Este proceso, también se llama habitualmente *visión por computadora* [4].

La visión puede ser considerada como una de las más potentes capacidades sensoriales de una máquina.

Los procesos anteriormente mencionados pueden ser subdivididos en seis áreas principales:

- Captación
- Preprocesamiento
- Segmentación
- Descripción
- Reconocimiento
- Interpretación.
-

A continuación se da una breve explicación de estos procesos.

La *captación* es el proceso a través del cual se obtiene una imagen visual. El *Preprocesamiento* incluye técnicas tales como la reducción de ruido, esqueletonización, realce de detalles. La *Segmentación* es el proceso que divide una imagen en objetos que sean de nuestro interés. A través de los procesos de *Descripción* se obtienen características (por ejemplo, tamaño, forma) convenientes de tal forma que se pueda diferenciar un tipo de objeto de otro. El *Reconocimiento* es el proceso que identifica estos objetos (por ejemplo un tornillo, una letra, un número) . Finalmente , la *Interpretación* asocia un significado a un conjunto de objetos reconocidos.

Es conveniente agrupar estas diversas áreas de acuerdo a la complejidad que lleva consigo su implementación. Consideremos tres niveles de procesamiento: *visión de bajo, de medio y de alto nivel*. Debido a que no existe unas fronteras claras entre estos tres niveles nos ceñiremos a un marco de trabajo útil para categorizar los diversos procesamientos que componen un sistema de visión artificial.

Visión de Bajo Nivel.- Se asociarán a la visión de Bajo Nivel aquellos procesos que son primarios en el sentido que pueden ser considerados "reacciones automáticas "sin requerir ningún tipo de inteligencia. Se puede considerar para nuestro caso a la captación y al preprocesamiento como funciones de Visión de Bajo nivel. Este procesamiento puede ser comparado con los procesos de percepción y adaptación que un hombre realiza cuando intenta encontrar un asiento en una sala de teatro el cual está obscuro, inmediatamente después de que ha estado caminando bajo un día soleado. Por lo tanto el

proceso inteligente de conseguir un asiento desocupado no se podrá llevar a cabo hasta que no tenga el hombre una imagen adecuada.

Visión de Nivel Intermedio.- Se asociarán a la visión de nivel intermedio aquellos procesos que extraen, caracterizan y etiquetan componentes de la imagen que ha sido obtenida por la visión de bajo nivel. Consideremos a los procesos de segmentación, descripción y reconocimiento de objetos individuales como funciones de la visión de nivel intermedio.

Visión de alto nivel.- Se refiere a procesos que tratan de emular la cognición.

Las categorías y subdivisiones anteriormente presentadas están basadas en la forma en que la mayoría de sistemas de visión artificial han sido desarrollados o implementados.

Estas relaciones no pueden ser modeladas analíticamente debido a que no se las puede entender bien. De tal forma que la subdivisión de funciones anteriormente presentadas puede servirnos como un esquema práctico para llevar a cabo nuevas realizaciones de sistemas de visión artificial, usando nuestro nivel de conocimiento y las herramientas analíticas que se disponen para emplearse en este campo.

3.3. PROCEDIMIENTO GENERAL PARA EL RECONOCIMIENTO DE PATRONES

Qué es el proceso de Reconocimiento de patrones? Cuando se analiza, por ejemplo, una página de periódico y se reconocen sin vacilación sus caracteres, se están aplicando reglas fijas que han sido obtenidas en base a la experiencia. Quizá no se puede definir en qué consisten exactamente estas reglas, pero al diferenciar en general todas las letras, generalmente las reglas existen. Puede ser que con el tiempo se hayan observado por ejemplo muestras de las letras "p" y "q" aprendiendo a distinguir sus características o rasgos, de tal forma que el examen de esta muestra "controlada" (de la población de entrenamiento) haya producido una regla de decisión. Esta regla de decisión sirve para poder decidir, sin ninguna otra tutoría, si un nuevo caracter pertenece a la "clase p" o a la "clase q".

Dos aspectos fundamentales existen en todo reconocimiento de patrones, el desarrollo o creación de una regla de decisión (*Clasificador*) y la utilización de la misma.

El problema de diseño de un clasificador comienza, en líneas generales, con la definición de las clases objeto de estudio (números, letras, cosas, o animales, etc), y la representación adecuada de las muestras de la población de entrenamiento. El problema finaliza cuando se ha obtenido una regla de decisión en base a la cual se pueda clasificar una muestra que no pertenece a la población de entrenamiento.

El Reconocimiento se efectúa cuando se utiliza la regla; el *Patrón* se define a base de un proceso de aprendizaje (*learning processs*) . Para definir un patrón se utiliza la población de entrenamiento [1]. Los patrones de una muestra son ejemplos de cada una de las clases de patrones.

Aunque las reglas de decisión en reconocimiento de patrones deben ser sencillas y rápidas en su resolución, la obtención de las mismas necesitan de un considerable esfuerzo.

De lo anteriormente expuesto se puede decir que un sistema de reconocimiento o de una manera más formal un clasificador determinado debe ejecutar tres trabajos para cumplir su función, estos son:

1) Ejecutar un algoritmo de aprendizaje en el que los datos de entrada son algunas de las características de los elementos de un conjunto de muestras de entrenamiento. Una vez cumplida con esta fase de aprendizaje, el clasificador habrá almacenado de una manera codificada las muestras de los patrones.

2) Tomar cada muestra de prueba a ser clasificada para luego procesar sus características mediante algún método definido por el tipo de clasificador.

3) Una vez que se han definido las características de la muestra de prueba, debe tomar la decisión, es decir, le asigna una clase o lo rechaza.

3.3.1. Indicadores en los sistemas de reconocimiento de patrones

Para el análisis de efectividad de uno u otro método de reconocimiento de patrones, es importante tomar en cuenta ciertos indicadores como los siguientes.

a) *Tiempo de entrenamiento.*- tiempo que requiere el sistema para completar su fase de entrenamiento, con un determinado número de muestras de entrenamiento.

b) *Tiempo de Reconocimiento.*- tiempo que requiere el sistema para clasificar un vector de entrada. Los tiempos de entrenamiento y reconocimiento pueden medirse en tiempo real (segundos, minutos, horas, etc.), pero es importante notar las características del computador empleado.

c) *Tasa de reconocimiento.*- es la relación o porcentaje de elementos de una población de prueba que son correctamente clasificados. Matemáticamente se tiene;

$$t_r\% = \frac{Número\ de\ objetos\ correctamente\ clasificados}{Número\ de\ objetos\ de\ la\ población\ de\ prueba} * 100 \qquad (3\text{-}1)$$

Los elementos del conjunto de prueba pueden ser los mismos del conjunto de entrenamiento, y/o elementos de entrada nuevos. Pero la eficiencia de operación del sistema se verá con elementos de entrada nuevos.

d) *Tasa de error.*- es la relación o porcentaje de elementos de un conjunto de prueba que son clasificados erróneamente. Matemáticamente se tiene;

$$t_e\% = \frac{Número\ de\ objetos\ erróneamente\ clasificados}{Número\ de\ objetos\ de\ la\ población\ de\ prueba} * 100 \qquad (3\text{-}2)$$

e) Tasa de rechazo.- en ciertos sistemas de reconocimiento, algunos elementos del

conjunto de prueba son rechazados, debido a que no han sido clasificados por el sistema. Estos elementos rechazados forman un grupo denominado Desconocidos. Por lo tanto la tasa de rechazo se define como la relación o porcentaje elementos del conjunto de prueba que han sido rechazados por el sistema. Matemáticamente se tiene;

$$t_{rech}\% = \frac{Número\ de\ objetos\ rechazados}{Número\ de\ objetos\ de\ la\ población\ de\ prueba} * 100 \qquad (3\text{-}3)$$

Estas relaciones, matemáticamente cumplen lo siguiente;

$$(t_r + t_e + t_{rech})\% = 100\% \qquad (3\text{-}4)$$

3.4. RECONOCIMIENTO DE CARACTERES MANUSCRITOS.

3.4.1. Introducción

El área de acción de reconocimiento de patrones es amplia, por lo que cualquier problema de clasificación de objetos puede ser resuelto mediante el diseño de un sistema de reconocimiento de patrones. Un ejemplo de aplicación muy práctica del reconocimiento de patrones es el reconocimiento de caracteres.

El reconocimiento de caracteres es un área del reconocimiento de patrones que ha sido materia de investigación durante las últimas tres décadas. A diferencia del avance, tanto en la teoría como en la práctica del reconocimiento de caracteres aislados, relativamente pocos estudios han sido dedicados al reconocimiento de caracteres cursivos. Una de las razones para la escasez de estudios de reconocimiento de caracteres cursivos, es que, las características del texto cursivo son diferentes y más complejas que otros tipos de texto.

Para el estudio de este campo de reconocimiento de caracteres podemos definir a dos tipos de caracteres: impresos y manuscritos. Por lo que el campo de reconocimiento

de caracteres se divide en, reconocimiento de caracteres impresos y reconocimiento de caracteres manuscritos.

En el Reconocimiento de caracteres impresos , los caracteres a ser reconocidos provienen de máquinas de impresión.

Para el reconocimiento de caracteres manuscritos, las muestras de los caracteres son obtenidos de personas, estas muestras de caracteres presentan mucha variabilidad en su forma, lo que hace que la etapa de reconocimiento sea más difícil . Las causas pueden ser debido al hábito de la escritura, estilo, educación, salud, y otras condiciones del escritor. Además, otros factores como el instrumento de escritura, la superficie de la escritura, métodos de rastreo, contribuyen en la variabilidad del manuscrito.

De este campo, el reconocimiento de caracteres manuscritos es el más estudiado e implementado debido a su mayor aplicabilidad.

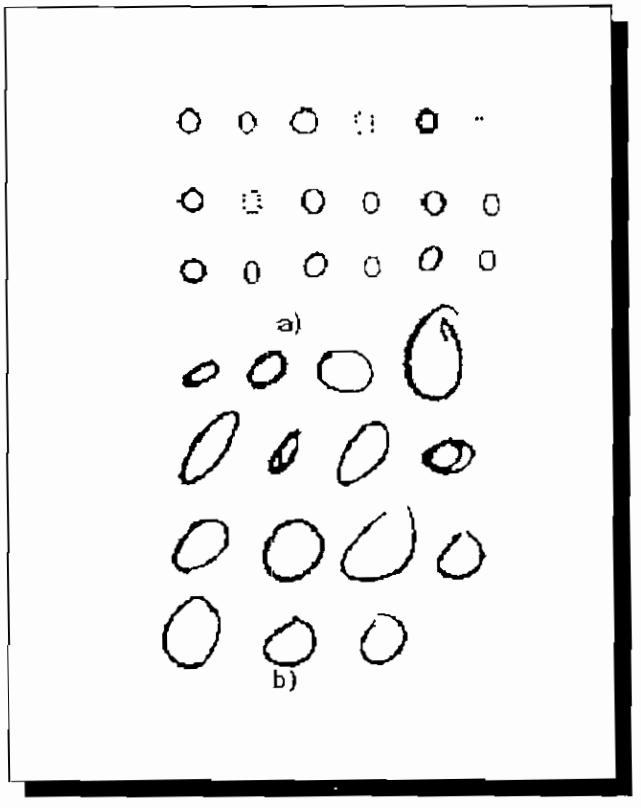


Fig.3.1 Numerales cero: a) impresos por máquina, b) manuscritos.

3.4.2. Perspectiva Histórica

La tecnología moderna de los sistemas de reconocimiento de caracteres se dice que nació en 1951 con el invento de M. Sheppard, GISMO, un robot que escribía y leía. En 1954, J. Rainbow desarrolló una máquina prototipo que era capaz de leer letras mayúsculas escritas en máquina a una velocidad de un carácter por minuto [3].

Varias empresas de Equipos de reconocimiento incluyendo IBM mercadearon este tipo de sistemas por el año de 1967. A finales de la década de 1960, esta tecnología experimentó muchos desarrollos, pero estos sistemas de reconocimiento de caracteres fueron considerados exóticos y futuristas, siendo usados solamente por agencias del gobierno o grandes corporaciones, ya que los sistemas tenían un altísimo costo.

Hoy los sistemas de reconocimiento de caracteres son menos costosos, más rápidos y más confiables. Existen sistemas de reconocimiento de caracteres capaces de reconocer varios cientos de caracteres por minuto.

La productividad de estos sistemas se va incrementando debido a que permite la reducción de la intervención humana y la habilidad de almacenar textos de una manera eficiente.

Investigaciones actuales en el área de sistemas de reconocimiento de caracteres van más allá del reconocimiento de caracteres manuscritos., esto es, están orientadas a sistemas de lectura, utilizando toda la información disponible para formular una interpretación del documento. Muchas de las aplicaciones de los sistemas de lectura *(lectores)* llevan a cabo tareas como: leer cheques de banco, direcciones de correo, tarjetas de crédito, boletos de aviones , pasaportes [3].

Para la obtención de datos en un sistema de reconocimiento de caracteres se utiliza generalmente un dispositivo electrónico denominado SCANNER, periférico del computador que permite transformar una imagen dibujada sobre papel en un archivo binario, con lo que se obtiene una imagen digital. Una imagen digital está dispuesta en forma de puntos donde cada uno de los puntos toma el nombre de pixel, al cual se le

puede dar color. El color del pixel está determinado por un valor discreto que puede codificarse con un número específico de bits.

3.4.3. Obtención de datos para la implementación de un sistema de reconocimiento de caracteres (SCANNER)

El scanner contiene un sensor óptico que mide los niveles de iluminación sobre cada pixel de la imagen discretizada. Estos niveles de iluminación son normalizados, discretizados (con un número fijo de bits/pixel) y transmitidos desde el dispositivo scanner hacia el computador, donde son almacenados en forma de un archivo binario.

El scanner discretiza una imagen plana (plasmada en papel) mediante un número finito de elementos de imagen (pixel) con una densidad que es predefinida (por ejemplo para el presente trabajo se utiliza 300 pixels/pulgada)

Una imagen monocromática puede mostrar solo dos colores, de tal manera que cada pixel de la imagen es definido por un bit, es decir 1 bit/pixel. Por lo tanto se tiene 1 (blanco) cuando su iluminación supera un valor umbral, y 0 (negro) cuanto su iluminación es menor que dicho umbral.

El número de bits/pixel requeridos para la codificación de la imagen está en relación con la paleta de colores, en la que se especifica los colores necesarios para la reproducción de la imagen. Existen paletas de 2, 4, 16, 256, 65536 y 16'777.216, en los que cada color será representado con 1, 2, 4, 8, 16, y 24 bits, respectivamente.

Un scanner de 4 bits/pixel transformará un punto blanco sobre el papel en un pixel de máxima iluminación (1111) en un archivo binario. Un punto negro en un pixel de mínima iluminación (0000); y un punto gris en un pixel de iluminación media (entre 0001 y 1110).

3.4.4. Sistemas de reconocimiento de caracteres manuscritos

Un sistema de reconocimiento de caracteres manuscritos es el proceso de convertir imágenes rastreadas de texto manuscrito (numerales, letras y símbolos) en un formato que sea procesable por computadora.

Dos componentes esenciales existen en un algoritmo de reconocimiento de caracteres, el extractor de características y el clasificador. El análisis de las características define el conjunto de características, que se usarán para describir todos los caracteres. Dada la imagen del carácter, el extractor de características o rasgos obtiene las características que el carácter posee. Las características obtenidas son entonces usadas como entradas para el clasificador de caracteres.

Para una imagen de entrada de un caracter, se extraen las características del caracter y luego se aplica un sistema basado en reglas al reconocimiento del mismo.

Un sistema de reconocimiento de caracteres consiste típicamente de los siguientes procedimientos [5]:

1. *Rastreo (Scanning)* de la imagen a ser procesada ya sea en escala de gris o en binario para el presente caso, con una apropiada resolución, típicamente 300 - 1000 dpi (dot per inch) puntos por pulgada.

2. *Preprocesamiento*
   que incluye operaciones tales como:
   - escalamiento, realce, suavizado, segmentación para aislar caracteres individuales.
   - Conversión a otro tipo de representación del carácter o patrón, por ejemplo, esqueleto, curva de contorno, etc.

3. *Extracción de las características o rasgos*

4. *Reconocimiento* usando uno o más clasificadores.

Los pasos anteriores pueden resumirse en la siguiente gráfica

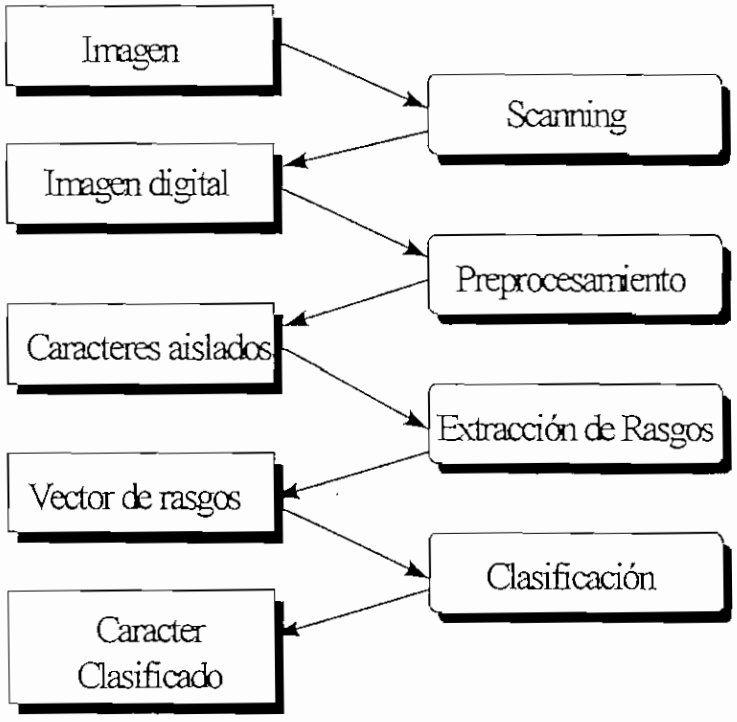


Fig.3.2 Procedimientos en un sistema de reconocimiento de caracteres.

Como se mencionó anteriormente en la sección 3.4.1. el reconocimiento de caracteres es una de las aplicaciones más exitosas del reconocimiento automático de patrones. Hoy en día su desarrollo e investigación se ha incrementado, de tal manera que se pueden comprar paquetes sobre reconocimiento de caracteres, sin embargo, estos únicamente son capaces de reconocer caracteres de textos impresos de alta calidad o caracteres de texto manuscrito escritos con gran precisión.

Por este gran desarrollo e investigación en este campo se han presentado una gran variedad de sistemas de reconocimiento de caracteres de varios idiomas o lenguas, tales como Chino, Inglés, Arábigo, etc. Por ejemplo Jairo Rocha y Theo Pavlidis [6] proponen un Modelo para el Análisis de Formas para un sistema de reconocimiento de caracteres, I. S. Abuhaiba, S. A. Mahmoud, y R. J. Green [7] proponen un sistema de reconocimiento de caracteres manuscritos Arábigos. Estos métodos son una muestra de como se pueden extraer las características fundamentales de un carácter, tales como sus rasgos geográficos o topológicos.

3.4.4.1. Consideraciones para el diseño de un Sistema de Reconocimiento de caracteres

El paso primordial antes de seleccionar el método de extracción de rasgos o características, es tomar en cuenta ciertas consideraciones mediante las cuales operará el sistema de reconocimiento de caracteres. Entre estas podemos mencionar las siguientes:

- Tipo de caracteres de entrada a ser reconocidos
  - Impresos (máquina)
  - manuscritos
- Formato de la imagen
  - BMP (Mapa de bits para Windows).
  - TIFF (Formato de imágenes etiquetado).
  - PCX (Paintbrush para PC).
  - etc.
- Nivel de la imagen
  - gris
  - binaria
- Resolución del scanner
- Tipo de clasificador
- Software disponible

Referencias:

[1] Escudero L. F., Reconocimiento de Patrones, Centro de Investigacion UAM-IBM, Madrid (España), 1997.

[2] Bezdek, J. C. y Pal, S. K. *Fuzzy Model for Pattern Recognition*, New York: *IEEE*, Inc, 1992.

[3] Sargur N. Srihari and Stephen W. Lam, Character Recognition. *Center of excellence for Documente Analysis And Recognition*, State University of New York.

[4] K. S. Fu, R. C. Gonzalez and C.S.G. Lee, *Robótica, Control, Detección, Visión e Inteligencia*, Mc Graw-Hill, 1990.

[5] Oivind Due Trier and Anil K. Jain, "Goal-Directed Evaluation of Binarization Methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 12, pp. 1191-1201, December 1995.

[6] Jairo Rocha and Theo Pavlidis, "A Shape Analysis Model with applications to a Character Recognition System," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 4, pp. 393-404, April 1994.

[7] I. S. I. Abuhaiba, S. A. Mahmoud, and R. J. Green, "Recognition of Handwritten Cursive Arabic Characters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 6, pp. 664-672. June 1994.

[8] S. N. Srihari, High-Perfomance reading machines, Proceedings of the IEEE, 80(70, July 1992, pp. 1120-1132.

[9] J. R. Ullman, "Pattern Recognition Techniques," Butterworth, London (1973).

[10] V. K. Givindan and a. P. Shivaprasad, "Character Recognition - A Review," *Pattern Recognition*,**23**(7), 671-683, (1990).

[11] T. Wakahara, Toward Robust Hanwritten character recognition, *Pattern Recognition*, **14**, 345-354 (April 1993).

4.1 LÓGICA DIFUSA (CHEN Y SIY)

El procedimiento que utiliza como fundamento los sistemas difusos y entre estos el de la lógica difusa (*FUZZY LOGIC*) para el reconocimiento de los caracteres numéricos manuscritos [1] fue desarrollado por Pepe Siy, y C. S. Chen miembros del departamento de Ingeniería Eléctrica de la Universidad de Akron, Ohio.

El presente método considera a caracter manuscrito como un gráfico abstracto, del cual se obtiene un conjunto de nodos, definidos como puntas, esquinas y uniones, y el conjunto de ramas que consiste de segmentos de líneas que conectan un par de nodos adyacentes. Los segmentos de línea (ramas) son clasificadas de una manera difusa en tipos de ramas (rasgos) tales como líneas rectas, círculos o porciones de círculos.

Debido a que los rasgos bajo consideración son difusos por naturaleza, se utiliza el concepto de los conjuntos difusos, y los rasgos son tratados como variables difusas. Una variable difusa se caracteriza , como se estudió en el capítulo (I) por el valor del grado de la función de pertenencia, el cual puede tomar valores en el intervalo cerrado unidad [0, 1]. Los grados de pertenencia extremos, es decir 0 o 1 definen un estado de no pertenencia y pertenencia completa de la rama en consideración.

Los caracteres manuscritos son considerados objetos indefinidos. Con la ayuda de las funciones difusas tales objetos pueden ser definidos y estudiados de una manera objetiva. Un caracter manuscrito es representado mediante la función difusa, que relaciona sus variables difusas, y mediante el par de nodos involucrados en cada variable difusa.

El proceso de reconocimiento se lleva a cabo en dos etapas o pasos. En el **primer paso**, el caracter desconocido es preprocesado mediante la esqueletonización, de tal forma que de éste se pueda extraer los rasgos constituyentes o ramas que constituirán las variables difusas. Cada rasgo será codificado para formar un vector cuyas componentes serán los códigos de los rasgos constituyentes del caracter para luego ser almacenado en una memoria prototipo. Como **segundo paso**, la clasificación del caracter desconocido se reduce a encontrar un caracter (previamente aprendido) cuya representación sea isomórfica a la representación del caracter desconocido.

4.2 EXTRACCIÓN DE RASGOS TOPOLÓGICOS (TOU Y GONZALEZ)

Este sistema de Reconocimiento de caracteres manuscritos mediante la Extracción de rasgos Topológicos [2] fue diseñado por J. T. Tou miembro del Centro de Investigaciones Informáticas de la Universidad de Florida, y por R. C. Gonzalez miembro del Departamento de Ingeniería Eléctrica de la Universidad de Tennessee.

El sistema de reconocimiento se ha diseñado haciendo uso de la extracción de rasgos topológicos.

Mediante la especificación de un conjunto de Rasgos topológicos fácilmente detectables del caracter es posible convertir automáticamente los caracteres manuscritos en formas convencionales o estilizadas y luego clasificarlas en clases con configuraciones topológicas similares. Las configuraciones topológicas del caracter son codificadas y luego agrupadas en un vector que contendrá todas las características del caracter bajo consideración.

El reconocimiento final se lleva a cabo en una etapa que realiza el análisis del caracter de entrada y los caracteres previamente aprendidos.

4.3 .ANÁLISIS ESTRUCTURAL Y MATEMÁTICO DE LOS MÉTODOS MENCIONADOS

Antes de iniciar el análisis de los métodos mencionados, realizaremos a continuación una revisión de conceptos fundamentales relacionados con el tratamiento de una imagen.

4.3.1. Representación de la imagen [3]

La imagen bidimensional digitalizada que se obtiene a partir de un mecanismo denominado Scanner es luego presentada en el dispositivo de imagen (monitor), donde para su análisis será representada por una función $f(x,y)$ donde x e y indican las coordenadas rectangulares en el plano de la imagen y el valor de $f(x,y)$ nos indica el nivel

de gris. Para el caso del presente trabajo este valor será de 0 (cero) o uno (1) debido a que la imagen es una imagen binaria. A este elemento *f(x,y)* se le llama elemento de imagen o pixel. La siguiente figura ilustra este concepto, así como el convenio de coordenadas en el cual estará basado nuestro estudio.

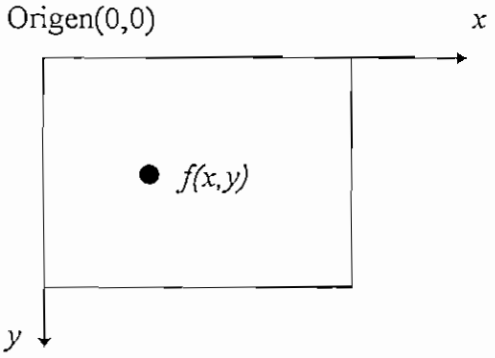


Fig 4.1 Representación de la imagen

Tomando como referencia el origen se observa que *f*(0,0) representa el pixel del origen de la imagen, *f*(1,0) será el pixel que se encuentra a su derecha, y así sucesivamente. Cabe notar que para imágenes en niveles de gris cada pixel son potencias enteras de dos.

4.3.1.1. Relaciones Básicas entre pixels

En esta sección se verán algunas relaciones básicas, pero importantes, entre los pixels de una imagen digital [3]. Como en la sección anterior, una imagen vendrá dada por *f*(x,y). Al referirnos a un pixel particular usaremos letras minúsculas, tales como *p* y *q*.

Vecinos de un pixel

$(x-1,y-1)$	$(x, y-1)$	$(x+1,y-1)$
$(x-1, y)$	$P(x,y)$	$(x+1,y)$
$(x-1,y+1)$	$(x, y+1)$	$(x+1,y+1)$

Fig. 4.2 Vecinos de un pixel p

Un pixel p de coordenadas (x,y) tiene cuatro vecinos horizontales y verticales cuyas coordenadas vienen dadas por

$$(x-1,y) \quad (x+1,y) \quad (x, y-1) \quad (x,y+1)$$

Este conjunto de pixels, son llamados los *4-vecinos de p*. Cada uno de estos cuatro pixels se encuentran a una distancia unitaria de (x,y). Algunos vecinos de p se encontrarán fuera de la imagen digital si (x,y) se encuentra en el borde de la imagen.

Los cuatro vecinos diagonales de p tienen coordenadas

$$(x-1,y-1) \quad (x+1,y-1) \quad (x-1,y+1) \quad (x+1,y+1)$$

Estos puntos, junto con los *4-vecinos de p* definidos anteriormente, se llaman los *8-vecinos de p*. De la misma manera algunos de estos puntos se encontrarán fuera de la imagen si (x,y) está en el borde de la imagen.

Conectividad

Es importante definir lo que significa la conectividad para el caso de una imagen digitalizada.

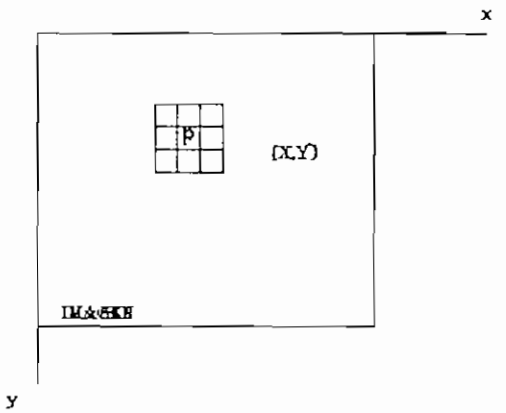
Sea un subconjunto de pixel *f(x,y)* notado por S. Por lo tanto un subconjunto de una imagen digitalizada está conectada si para cualquiera de dos puntos p y q del subconjunto existe una secuencia de puntos

$$p = p_0, p_1, p_2, p_3, \ldots, p_{n-1}, p_n = q$$

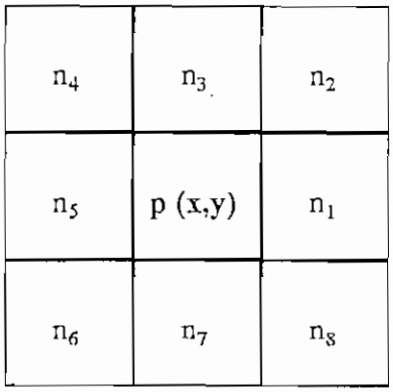
de tal forma que p_i es un vecino de p_{i-1}, $1 \leq i \leq n$.

Máscara de convolución

Una máscara de convolución o denominada también ventana, es básicamente una matriz bidimensional (3×3 por ejemplo que será utilizada en el presente trabajo), como el de la siguiente figura:



(a)



(b)

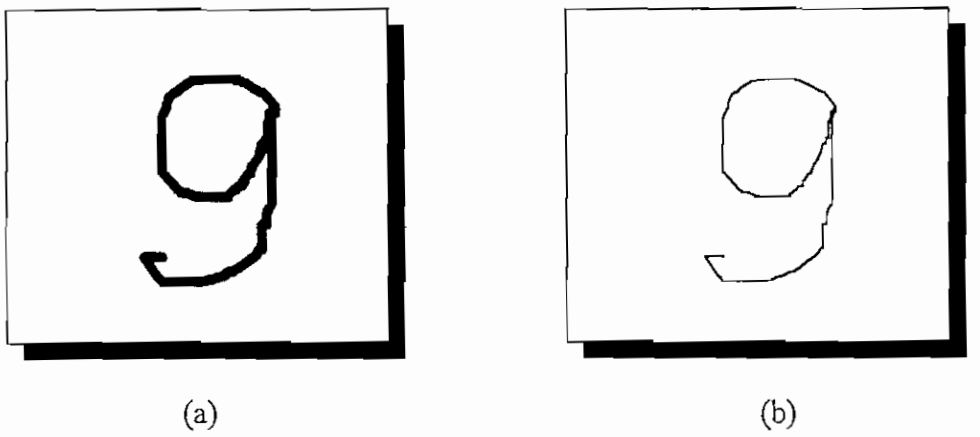
Fig. 4.3 a) Vecindad 3×3 alrededor de un punto (x,y) en una imagen. b) Máscara 3×3 que muestra los coeficientes.

Esta máscara de convolución es la más usada para definir un entorno de vecindad sobre (x,y). El centro de la subimagen se irá desplazando de pixel en pixel comenzando, por ejemplo en la esquina superior izquierda hasta la esquina superior derecha, y al mismo tiempo se irá examinando su vecindad para aplicar una operación sobre dichos pixels. A los vecinos n_1, n_2, n_3, n_4, n_5, n_6, n_7, n_8 también se les denomina coeficientes, que para nuestra aplicación tomarán valores de 0 (cero), o 1 (uno) por ser una imagen binaria. Las posiciones de los coeficientes en una máscara 3×3 con respecto al pixel central (x,y) será:

$n_1 \Rightarrow (x+1,y)$
$n_2 \Rightarrow (x+1,y-1)$
$n_3 \Rightarrow (x,y-1)$
$n_4 \Rightarrow (x-1,y-1)$
$n_5 \Rightarrow (x-1,y)$
$n_6 \Rightarrow (x-1,y+1)$
$n_7 \Rightarrow (x,y+1)$
$n_8 \Rightarrow (x+1,y+1)$

4.3.2. Esqueleto de una imagen binaria (Adelgazamiento) [3]

El problema de reducir una imagen digitalizada a líneas delgadas es de gran interés en el campo de reconocimiento de patrones [3], [6], más aún en el campo de reconocimiento de caracteres ya que la información esencial de su forma está contenida o almacenada en su esqueleto.



(a) (b)

Fig. 4.4 a) Imágen digitalizada sin preprocesamiento, b) Esqueleto de Ia imagen.

Muchos algoritmos para la obtención de esqueletos (llamados también procesos de ***esqueletonización***) han sido propuestos para extraer esqueletos[4], [5], [6], [7] cada uno de ellos intentan obtener una representación del esqueleto más precisa para un patrón. Para lo cual estos algoritmos formulan reglas complejas para ser aplicadas en el patrón.

4.3.2.1. Requerimientos que debe poseer el esqueleto de una imagen [6]

1.- ***Delgada***

Primeramente, el esqueleto, se requiere que consista de líneas delgadas. Lo cual se consigue erosionando el subconjunto de imagen mediante la eliminación de puntos que

yacen en los bordes hasta que queden líneas de un punto o de un pixel de ancho. Es importante señalar que los puntos deben ser procesados o tratados secuencialmente.

2.- ***Conectividad***

Este requerimiento es aquel en virtud del cual el proceso no debe alterar la conectividad del subconjunto. Esto se consigue probando si cada punto a ser removido altera o no la conectividad. Si altera, entonces el pixel o punto se retiene aún cuando esté en el borde.

3.- ***Posición***

Un punto es removido solamente si yace en el borde de un subconjunto inicial, a pesar de que otros puntos han sido removidos. Esto significa que se requieren varias pasadas a través de los puntos del subconjunto. En cada pasada la capa más externa de los puntos son removidos para dar un subconjunto "*más delgado*" para la próxima pasada.

4.- ***Estabilidad***

Es necesario que tan pronto como se obtenga un esqueleto satisfactorio, o parte de un esqueleto, este debería ser estable, y no erosionarse más por subsecuentes pasadas, ya que el esqueleto se va obteniendo mediante varias pasadas. Sin embargo, un punto que yace en la punta de una línea delgada puede ser removido sin alterar la conectividad. Por lo que se requiere una condición adicional para el efecto, ya que tal punto no puede ser removido, porque de otra manera las líneas del esqueleto llegarán a ser gradualmente más y más cortas. Esta condición confiere estabilidad al esqueleto.

4.3.3 Lógica Difusa [1], [10]

El sistema de reconocimiento de caracteres numéricos manuscritos consiste de dos unidades básicas A) Extracción de rasgos y la representación del carácter numérico manuscrito, y B) el clasificador de patrones.

A. Representación del Carácter Manuscrito

Los caracteres manuscritos son una variación distorsionada de los caracteres impresos; por lo tanto, cualquier estudio de caracteres manuscritos debe comenzar con caracteres impresos. Existen esencialmente tres elementos básicos de los caracteres alfa numéricos:

1) La línea recta (vertical, horizontal, e inclinada)
2) El círculo, y
3) Una porción de círculo con varias orientaciones.

Mediante el uso de los "conjuntos difusos" introducidos en 1965 por L. A. Zadeh [9], [10], [11] y estudiados en el capítulo I, las clases de líneas dibujadas a mano, círculos, o porciones de círculos son considerados como conjuntos difusos en este trabajo.

Los siguientes rasgos son suficientes para representar numerales manuscritos.

1) Tipos de línea recta

- Horizontal (LH)
- vertical (LV)
- pendiente positiva (LP)
- pendiente negativa (LN)

2) Tipos de porciones de círculo

- Curva con abertura hacia arriba (CU)
- curva con abertura hacia abajo (CA)
- curva con abertura hacia la izquierda (CI)
- curva con abertura hacia la derecha (CD)

3) Círculos

- Círculo (O)

Considerando el conjunto

$$B = \{ LH, LV, LP, LN, CU, CA, CD, CI, O \} \qquad (4\text{-}1)$$

que denota el conjunto de trazos característicos que se muestran en la siguiente figura

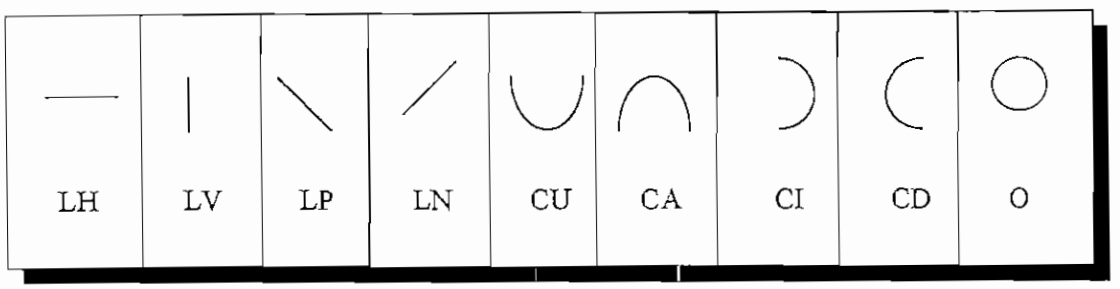


Fig. 4.5 Conjunto de trazos característicos

Los modelos de los caracteres son descritos mediante sus descripciones funcionales, es decir mediante los trazos característicos componentes, por ejemplo:

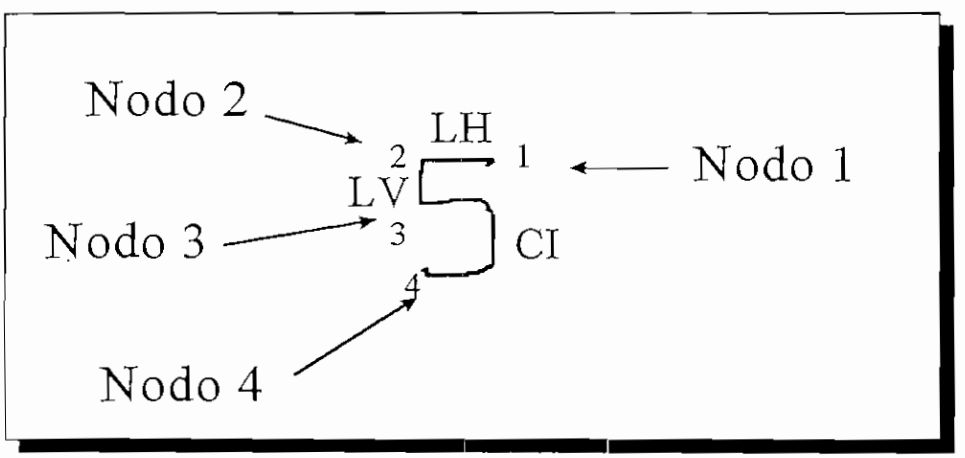


Fig. 4.6 Ejemplo típico de un numeral 5

La descripción funcional de este numeral corresponde a

$$F(5)= LH(1,2) * LV(2,3) * CI(3,4) \qquad (4\text{-}2)$$

Este ejemplo nos da la idea de generalización, lo que significa que uno tiene que aprender un ejemplo típico del carácter 5 para reconocer cualquier alternativa equivalente. Sin embargo existen varias descripciones de un mismo caracter.

B. Clasificador de Patrones

El criterio de decisión usado en la clasificación de patrones se ejecuta de la siguiente manera:

Las característica de las ramas del patrón a ser clasificado y las características de las ramas de los prototipos de cada clase son comparados. Aquellos prototipos que coinciden perfectamente con los trazos característicos son tomados, es decir cuando son isomórficos son aceptados.

4.3.3.1 El Sistema de Reconocimiento

El diagrama de bloques del sistema de reconocimiento para la fase de aprendizaje y la fase de reconocimiento se muestran a continuación

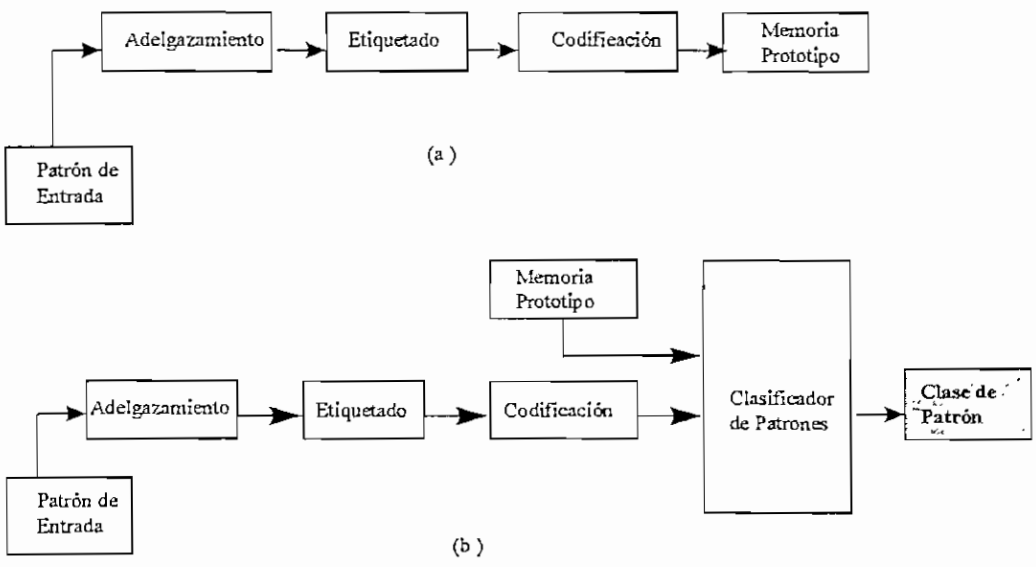


Fig. 4.7 Diagrama de bloques del sistema de reconocimiento. (a) Fase de Aprendizaje. (b) Fase de reconocimiento.

Las principales características de estos diagramas consisten de los siguientes componentes: patrón de entrada, adelgazamiento, etiquetado, codificación, memoria prototipo, y el clasificador de patrones. Cada uno de estos componentes se describen a continuación.

A. Patrón de entrada

El patrón de entrada o carácter de entrada del sistema debe ser digitalizado en un arreglo de una estructura de imagen $P= \{ p= (i,j)/ 1 \leq i \leq n,\ 1 \leq j \leq m;\ n,m \in N\}$. El patrón es una imagen binaria; es decir, los puntos en el patrón asumen el valor de uno mientras los otros puntos tomarán el valor de cero.

B. Adelgazamiento

El esqueleto de la imagen se obtiene usando el algoritmo de adelgazamiento de Hilditch [8]. Básicamente, el algoritmo prueba los puntos en el borde del patrón de entrada. Estos puntos son removidos de tal forma que no se altere la conectividad.

Algoritmo de Hilditch [8]

Para cumplir con los requerimientos antes mencionados, el algoritmo a ser implementado requiere varias pasadas. Al inicio de una pasada se da un subconjunto Q de alguna subimagen f_0 de P ; al final de cada pasada se define un subconjunto Q' de Q, el cual es una capa más "delgada". Inicialmente Q es el subconjunto a ser reducido a líneas delgadas; el resultante Q' entonces llega a ser Q para la segunda pasada y así sucesivamente; finalmente se obtiene un Q', el cual es el esqueleto de la forma requerida.

Es importante considerar que el subconjunto Q se define como aquellos puntos de P para los cuales f_0 toma uno de los valores de los siguientes conjuntos

$$I= \{1\} \qquad (4\text{-}3)$$

$$N= \{0\} \qquad (4\text{-}4)$$

$$R= \{a\} \text{ ; conjunto a eliminarse} \qquad (4\text{-}5)$$

Es decir los puntos que pertenecen al subconjunto Q tienen valor de uno, lo que quiere decir que pertenecen a *I (puntos de la imagen)*, todos los otros puntos tienen valor de cero (*elementos de N*), y cuando un punto va a eliminarse su valor es de *a*.

Condiciones para la eliminación de un punto

Considerar a *p* un punto de la imagen, *f(p)* el valor que toma este punto ya sea en *I*, *N* o en *R*, y $f(n_i)$ el valor que toma cada uno de los 8 vecinos de p.

n4	n3	n2
n5	***p***	n1
n6	n7	n8

Fig. 4.8 Vecindad-8 de *p*

$f(n_i)$ = valor que toma cada uno de los 8 vecinos de p.

$f(n_i) \in N$, $f(n_i) \in Y$, o $f(n_i) \in R$.

Es decir:

Si $f(n_i) \in N$, entonces $f(n_i) = 0$

Si $f(n_i) \in I$, entonces $f(n_i) = 1$

Si $f(n_i) \in R$, entonces $f(n_i) = a$

Un punto p será eliminado, es decir pasará al conjunto en R, si y solamente si satisface todas las siguientes condiciones:

1. Pertenece a *Q*

$$f(p) \in I \qquad (4\text{-}6)$$

2. Yace en el borde de *Q*, es decir, al menos uno de sus vecinos axialmente adyacentes n_1, n_3, n_5, n_7 no pertenecen a *Q*; por ejemplo, Si

$$\mu(p) = a_1 + a_3 + a_5 + a_7 \qquad (4\text{-}7)$$

donde $a_i = 1$ si $f(n_i) \in N$, $a_i = 0$ de otra manera

Entonces se tiene la condición de borde

$$\mu(p) \geq 1 \qquad (4\text{-}8)$$

3. No es la punta de una línea delgada, es decir, tiene más de un vecino que pertenece a Q, por ejemplo, Si

$$v(p) = \sum_{i=1}^{8} (1 - ai) \qquad (4\text{-}9)$$

(a_i como se definió anteriormente)

Entonces se tiene la condición de que "no es una punta"

$$v(p) \geq 2 \qquad (4\text{-}10)$$

4. No es el último punto restante de un subconjunto circular pequeño , es decir, tiene al menos un vecino en Q el cual no ha sido eliminado o removido; por ejemplo, Si

$$w(p) = \sum_{i=1}^{8} ci \qquad (4\text{-}11)$$

donde $c_i = 1$ Si $f(ni) \in 1$, $c_i = 0$ de otra manera

Entonces se tiene la condición

$$w(p) \geq 1 \qquad (4\text{-}12)$$

5. Su eliminación no altera la conectividad, es decir, el área del plano definido por el subconjunto Q puede ser continuamente deformada para dar el área definida por el subconjunto Q - {p}.

El número de cruces $X(p)$ se calcula

$$X(p) = \sum_{i=1}^{4} bi \qquad (4\text{-}13)$$

donde:

- $bi = 1$ *Si* $f(n_{2i-1}) \in N$ *y* $(f(n_{2i}) \in I \cup R$ *o* $f(n_{2i-1}) \in I \cup R$
- $bi = 0$ *de otra manera*

La condición de conectividad está dada por

$$X(p) = 1 \qquad (4\text{-}14)$$

C. Etiquetado

1) Etiquetado y detección del nodo: El conjunto de nodos en el esqueleto de un patrón se define como la colección de puntas (puntos que tienen un vecino), esquinas (puntos que tienen dos vecinos y ocurre un cambio abrupto en la dirección de la línea), uniones (puntos que tienen tres o más vecinos).
2) Detección de ramas y etiquetado: Una rama es un segmento de línea que conecta un par de nodos adyacentes. Para el caso degenerado de un patrón de línea sin nodos, el patrón se considera como una rama circular.

En la clasificación de ramas, dos fuentes de difusidad se atribuyen para la medida de:

- Rectitud, y
- Orientación

Medida de Rectitud

La medida de rectitud se determina ajustando una línea recta que une sus nodos adyacentes, con la curva o rama analizada, y a partir de su comparación se obtiene un valor de error LSM, (ver figura 4.9) definido por:

$$LSM = Coordenada\ en\ x\ de\ la\ recta - coordenada\ en\ x\ de\ la\ rama \quad (4\text{-}15)$$

Si $|LSM| \leq$ Umbral, entonces es una rama recta

Si $|LSM| >$ Umbral, entonces es una rama curva

Medida de orientación

Para la medida de la orientación definimos la siguiente expresión

$$\theta = arctan\ m \quad (4\text{-}16)$$

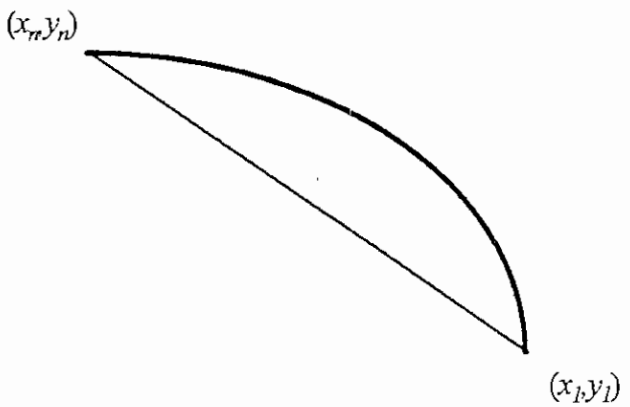


Fig. 4.9 Construcción de una línea para la clasificación de una curva en un tipo de rama

donde:

$$m = (y_n - y_1)/(x_n - x_1) \quad (4\text{-}17)$$

Clases de línea recta:

Esta clase contiene los conjuntos H(Línea horizontal), V (Línea vertical), P (Línea con pendiente Positiva), N (Línea con pendiente Negativa). Para poder clasificar una rama dada en uno de estos conjuntos se requiere, una medida de la orientación basada en el valor de θ . Con esta pendiente se definen las siguientes funciones de pertenencia para los conjuntos H, V, P, N

$$FH(\theta) = 1 - \min\{ |\theta/45|, |(180 - \theta)/45|, |(360 - \theta)/45|, 1\} \quad (4\text{-}18)$$

$$FV(\theta) = 1 - \min \{|(90 - \theta)/45|,|(270 - \theta)/45|, 1\} \quad (4\text{-}19)$$

$$FN(\theta) = 1 - \min \{|(135 - \theta)/45|, |(315 - \theta)/45|, 1\} \quad (4\text{-}20)$$

$$FP(\theta) = 1 - \min \{|(45 - \theta)/45|, |(225 - \theta)/45|, 1\} \quad (4\text{-}21)$$

La rama es clasificada dentro del conjunto con asignación de grado de pertenencia más alta.

Clases de porciones de círculo

Esta clase contiene los conjuntos CU, CA, CD, CI. Estas clases pueden ser agrupadas como curvas verticales (CD, CI), como horizontales (CU, CA). Para asignar una rama dada a uno de estos grupos, se toma en consideración la medida de orientación, es decir la pendiente θ = arctan $(y_n - y_1)/(x_n - x_1)$ de la línea que conecta los puntos terminales de la rama en consideración.

Para llevar a cabo esto, se consideran dos conjuntos difusos denominados, el de la curva horizontal (HC) y el de la curva vertical (CV). Basándose en la pendiente, las funciones de grado de pertenencia para HC, VC se definen de la siguiente manera

$$FHC(\theta) = 1 - \min \{| \theta/90|, |(180 - \theta)/90|, |(360 - \theta)/90|, 1\} \quad (4\text{-}22)$$

$$FVC(\theta) = 1 - \min \{|(90 - \theta)/90|, |(270 - \theta)/90|, 1\} \quad (4\text{-}23)$$

Por lo tanto la rama se clasifica dentro del conjunto con el grado de pertenencia más alto. La rama se caracteriza de una manera más definida dentro de uno de los grupos, considerando detalles obtenidos de una manera no difusa.

D. Codificación de los caracteres

Los elementos LH, LV, LN, LP, CD, CI, CA, CU, O, del conjunto de características de una rama descritos en la figura 4.4 son codificados por 10,11,12, 13,15,16,17, 18, y 20 respectivamente. Para la identificación de las características de cada carácter, se recurre a un vector, (1×m) con m= 6, el cual contiene las características propias de cada carácter.

La codificación consiste de 2 componentes, la primera que consta de los códigos de las características del carácter c_{11} a c_{15}, y la segunda de la etiqueta del carácter, la cual consta de un solo dígito c_{16}. Así pues el vector de características queda en la forma:

$$(c_{11}, c_{12}, c_{13}, c_{14}, c_{15}, \mathbf{c_{16}})$$

Por ejemplo:

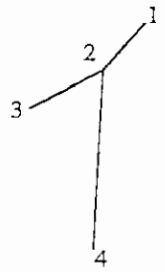


Fig. 4.10 Carácter 1 con sus respectivos nodos numerados a ser codificado

La codificación del carácter 1 será (*12,12,11,00,00*, **1**), donde podemos apreciar las dos componentes del vector, es decir la primera (con letra cursiva) correspondiendo a los códigos de las características del carácter, y el último componente (en negrilla) correspondiendo a la etiqueta del numeral.

A este vector lo denominaremos también vector de características (rasgos).

E. Memoria Prototipo

Cada uno de los vectores de la codificación representa un prototipo de un carácter. La memoria prototipo estará constituida por n de estos vectores, donde n será variable.

F. Clasificador de Patrones

El criterio de decisión, se basa en el vector de los datos codificados, ya que se compara los componentes del vector de las características de las ramas del patrón a ser clasificado con los componentes del vector de las características de las ramas del prototipo de cada clase. Si los prototipos encajan perfectamente con las características del patrón, entonces se lee la etiqueta del vector, para determinar cual es el carácter analizado, de otra manera, es rechazado.

4.3.3.2 Operación del sistema

La operación del sistema de reconocimiento consiste de dos fases:A) fase de aprendizaje, y B) la fase de reconocimiento.

A. *FASE DE APRENDIZAJE*

Los prototipos de cada clase de patrón son aprendidos y almacenados en la memoria prototipo. Cada patrón es adelgazado, etiquetado, y codificado. El código del primer patrón es almacenado en la memoria prototipo, luego viene el siguiente, pero se compara si el código anterior es el mismo. Si es el mismo código no lo almacena caso contrario lo almacena. Este proceso se repite para los códigos de todos los patrones de entrada. El proceso de comparación se lo realiza con el fin de que los datos a almacenarse no sean redundantes. Al final del proceso de aprendizaje, la memoria (archivo de datos) contiene todos los códigos de los prototipos

B. FASE DE RECONOCIMIENTO

Los códigos de los prototipos aprendidos son usados para clasificar los patrones (caracteres) de entrada.

Cada carácter de entrada (carácter a ser reconocido), es adelgazado, etiquetado, y codificado. Los códigos del patrón de entrada en la fase de reconocimiento no incluyen la etiqueta de identificación.
La codificación del carácter definida por su vector es comparada con cada una de las primeras cinco componentes del vector de la memoria prototipo, hasta encontrar el código que coincida con el del carácter de entrada. Una vez encontrada esta matriz, se procede a leer la segunda componente de la matriz prototipo que es la etiqueta la cual indicará cual fue el carácter de entrada.

Se debe tener en cuenta que se pueden obtener tres resultados luego de la ejecución del sistema , clasificación, error en la clasificación, y rechazo.

- Clasificación.- Ocurre cuando el carácter ha sido reconocido, es decir la etiqueta de identificación de la matriz prototipo coincide con el carácter de entrada (a ser reconocido).
- Error en la clasificación.- Ocurre cuando se ha encontrado que la matriz del caracter a ser reconocido es igual a la primera componente de la matriz prototipo, pero la etiqueta de identificación de ésta es diferente del carácter de entrada.
- Rechazo.- Ocurre cuando en la memoria prototipo no se ha encontrado una matriz isomórfica a la del carácter de entrada.

4.3.4 Extracción de Rasgos Topológicos [2]

La estructura básica del sistema reconocimiento se divide en las siguientes etapas:
1) Extracción de las características (rasgos topológicos) y codificación de vector de las características, y
2) Clasificación.

1) Extracción de características y Codificación

Para llevar a cabo este proceso se utiliza una herramienta básica que es la grilla octogonal, en la que sus octantes se encuentran numerados, como se muestra en la siguiente figura.

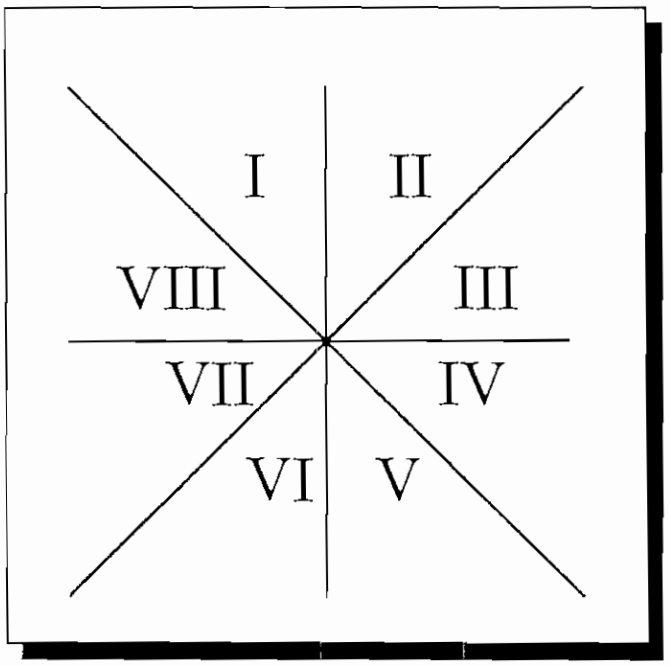


Fig. 4.11 Grilla octogonal

El conjunto de rasgos topológicos básicos mostrados a continuación y estudiados en el capítulo II, nos permiten representar numerales manuscritos.

FORMAS STANDARD DE LAS MEDIDAS TOPOLÓGICAS.

Tipo de característica	Símbolo	Explicación
Ninguna característica	0	No existe ninguna señal en un octante

1 Una característica o rasgo que atraviesa los octantes 1, 2, 5, o 6 es aproximada a una línea horizontal.

2 Un rasgo que atraviesa los octantes 3, 4, 7 o 8 es aproximada a un línea vertical.

3 Un rasgo que termina en un octante en el sentido de las manecillas del reloj.

4 Un rasgo que termina en un octante en el sentido contrario al de las manecillas del reloj.

5 Un lazo en un octante con abertura hacia la derecha se considera únicamente en los octantes 1, 6, 7 y 8.

6 Un lazo en un octante, con abertura hacia la izquierda se considera únicamente en los octantes2, 3, 4, 5.

	7	Un lazo en un octante, con abertura hacia arriba. Se considera únicamente en los octantes 4, 5, 6, y 7.
	8	Un lazo en un octante, con abertura hacia abajo. Se considera únicamente en los octantes 1, 2, 3, y 8.
	9	Un rasgo que se encuentra contenido en un octante.

En este cuadro podemos observar que cada uno de los rasgos topológicos han sido representados por un numeral del 0 al 9.

Mediante el uso de estas características topológicas y del uso de una grilla octogonal centrada en el carácter bajo consideración se van obteniendo las características en cada uno de los octantes de la grilla comenzando en el octante (I) hasta el octante (VIII) como se muestra en el siguiente ejemplo:

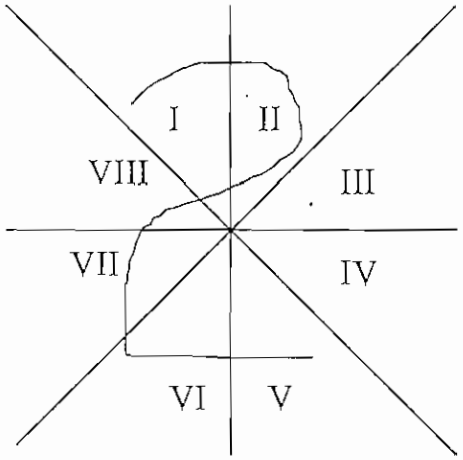


Fig. 4.12 Grilla Octogonal para el carácter 2

Características obtenidas del numeral 2

Octante	Característica1 (Símbolo)	Característica2 (Símbolo)	Característica3 (Símbolo)
I	4	1	0
II	6	0	0
III	0	0	0
IV	0	0	0
V	4	0	0
VI	0	0	1
VII	0	0	2
VII	0	0	2

2) Clasificación

El criterio para la clasificación de los patrones de entrada se lleva a cabo de la siguiente forma;

Se comparan las características del patrón de entrada y las características de los patrones de cada clase almacenados en una memoria prototipo, y si el patrón de entrada coincide perfectamente con el prototipo, entonces es aceptado, caso contrario es rechazado.

4.3.4.1 Estructura del sistema de reconocimiento

El sistema de reconocimiento se muestra mediante un diagrama en la siguiente figura.

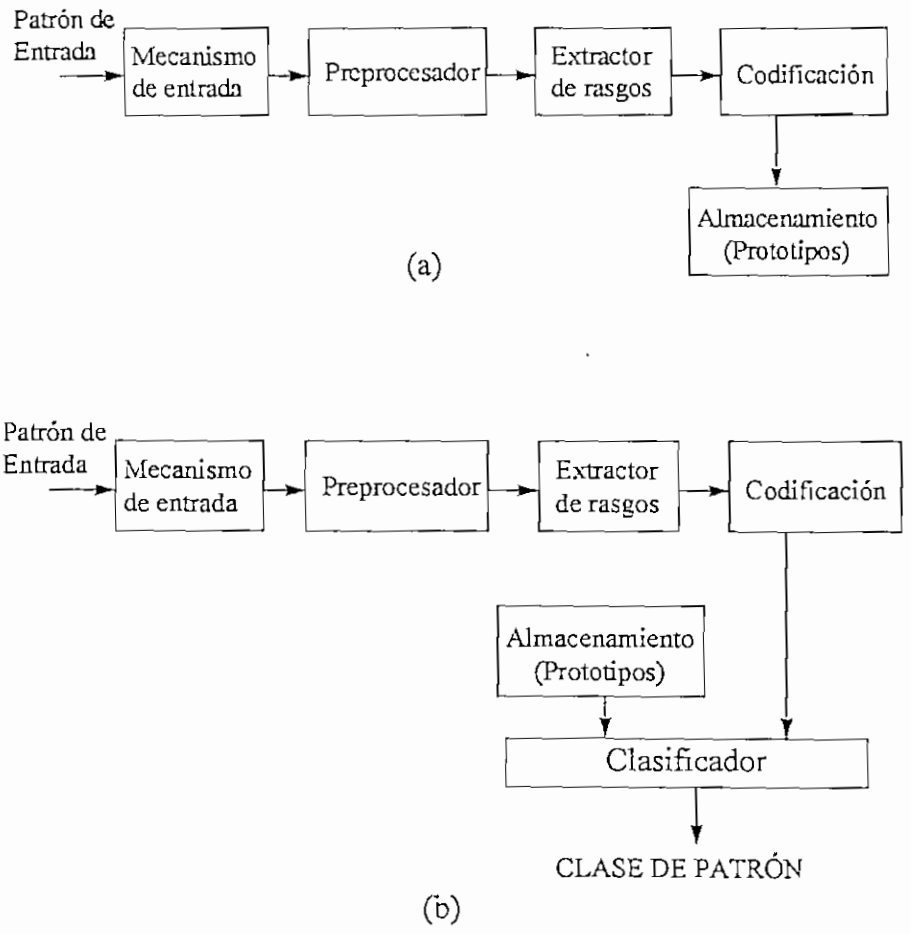


Fig. 4.13 Estructura básica del sistema de Reconocimiento: a) Aprendizaje, b) Reconocimiento.

Mecanismo de entrada

Convierte una imagen física en una imagen digital mediante un scanner.

El Preprocesador

Centra la grilla y adelgaza el patrón mediante el Algoritmo de Hilditch [8] estudiado en la sección 4.3.3.1, además busca los puntos finales y nodos de los rasgos o

características del carácter. Los octantes son arbitrariamente enumerados con números romanos como se muestra en la figura 4.11. La constante **c** determina la medida del tamaño de la grilla. La naturaleza simétrica de la grilla facilitará la detección de las características del carácter además de que tiene la ventaja de que proporciona un software de detección de características independiente del tamaño del carácter.

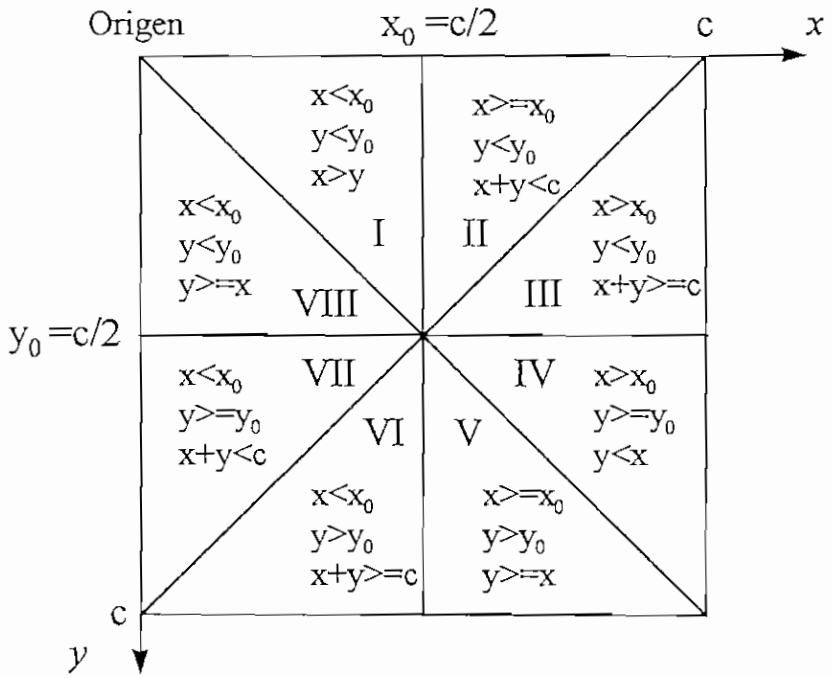


Fig. 4.14 Grilla de Medición

Extractor de rasgos

Determina las características básicas componentes del carácter bajo consideración de octante en octante, en base a las características mostradas en el cuadro de las formas standard de las medidas topológicas.

Codificación

Una vez que los rasgos topológicos básicos han sido detectados en un carácter o patrón, estos son codificados mediante un vector de símbolos al cual se le denomina vector de rasgos y se lo expresa de la siguiente manera

$$X=\{xij\} \qquad (4\text{-}24)$$

$$i = 1,\ldots 8$$
$$j = 1,\ \ldots 5$$

$$X= (x_{11},\ x_{12},x_{13},x_{14},x_{15};\ x_{21},x_{22},x_{23},x_{24},x_{25};\ \ldots\ ;\ x_{81},x_{82},x_{83},x_{84},x_{85}) \qquad (4\text{-}25)$$

donde:

$x_{i1},x_{i2},x_{i3},x_{i4}$:son los símbolos de cada característica contenida en el octante i.

x_{i5} :representa la etiqueta del caracter.

Es importante notar que solamente 4 rasgos son permitidos por cada octante, ya que en la práctica este número nunca se excede.

La longitud del vector se mantiene constante insertando ceros en aquellos componentes con menos de 4 rasgos.

Por ejemplo el siguiente vector de rasgos corresponde al número 2 mostrado en la fig. 4.12.

X = { 4,0,1,0,2; 0,6,0,0,2;0,0,0,0,2; 0,0,0,0,2; 4,0,0,0,2; 0,0,2,0,2; 0,0,2,0,2}

Almacenamiento

Cada uno de los vectores de rasgos del carácter son almacenados como prototipos y la cantidad de estos vectores dependerá del número de patrones de entrada para el aprendizaje.

Clasificador

El vector de rasgos a ser clasificado, se compara con los vectores de rasgos de los prototipos almacenados en la fase de aprendizaje, de tal forma que cuando se encuentra un vector de rasgos que sea idéntico al del patrón de entrada, entonces se

determina a que carácter representa el patrón de entrada, leyendo la etiqueta del vector de rasgos del prototipo, caso contrario el patrón de entrada es rechazado.

Referencias:

[1] Pepe Siy and C. S. Chen, " Fuzzy Logic for Handwritten Numeral Character Recognition," *IEEE Transactions on systems, Man and Cybernetics*, November 1994.

[2] J. T. Tou and R. C. Gonzalez, " Recognition of Handwritten characters by topological feature extraction and multilevel categorization" *IEEE Trans. Comput.*, vol C-21, pp 776-785, July 1972.

[3] K. S. Fu, R. C. Gonzalez, y C. S. Lee, "Robótica: Control, Detección, Visión e Inteligencia", McGraw-Hill, 1994.

[4] T. Pavlidis, Algorithms for Graphics and Image Processing. Rocville, MD: Computer Science Press, 1982, pp. 195-214.

[5] S. H. Hung and T. Kasvand, "Critical points on a perfectly 8- or perfectly 6-connected thin binary line," *Pattern Recognition*, vol 16, no. 3,pp. 297-306, 1983.

[6] N. J. Naccache and R. Shinghal, "SPTA: A Proposed algorithm for thinning binary patterns," IEEE Trans. Syst. Man Cybernet., vol SMC-14, no 3, pp. 409-418, May/June 1984.

[7] E. R. Davies and A. P. N. Plummer, "Thinning Algorithms: A critique and a new methodology," *Pattern Recognition*, vol. 14, nos. 1-6, pp. 53-63, 1981.

[8] J. C. Hilditch, "Linear Skeletons from square cupboards," in Machine Intelligence 4, B. Meltzer and D. Michie, Eds. New York: American Elsevier, 1969, pp. 403-420.

[9] L. A. Zadeh, "Fuzzy Sets," Inform. Contr., vol. 8, pp. 338-353, 1965.

[10] C. Earl, The Fuzzy Systems Handbook: A Practitioner's Guide to Building, Using and Maintaining Fuzzy System, New York, 1994.

[11] B. Kosko, Neural Networks and Fuzzy Systems: A Dinamical Systems Approach to Machine Intelligence, Prentice-Hall, Inc, 1992.

CAPÍTULO V

PROGRAMA (SOFTWARE) PARA EL RECONOCIMIENTO DE CARACTERES MANUSCRITOS

5.1. DESCRIPCIÓN DEL PROGRAMA

El programa *RECANUM* de reconocimiento de caracteres numéricos manuscritos nos permite evaluar el funcionamiento de los algoritmos estudiados anteriormente en el capítulo IV.

RECANUM permite simular en un computador digital el funcionamiento de dos sistemas de reconocimiento de caracteres manuscritos. Las muestras de los caracteres manuscritos son recogidas sobre hojas blancas con cuadrículas especialmente diseñadas para el efecto.

El conjunto de datos es utilizado para el proceso de entrenamiento y reconocimiento (población de entrenamiento), y además otro conjunto de datos para el reconocimiento sin entrenamiento previo (población de prueba). La codificación de cada uno de los datos es distinto en cada método de reconocimiento.

Para la ejecución del programa *RECANUM* se efectúan los siguientes pasos:

- De los datos obtenidos de un archivo binario proveniente de un scanner se obtiene la imagen de los caracteres manuscritos a través de un subprograma, el cual se encuentra dentro del programa principal. Este proceso nos permite obtener el conjunto de muestras para la fase de aprendizaje y reconocimiento.

- Se selecciona el método a ser utilizado es decir Lógica Difusa o Rasgos topológicos para luego iniciar el proceso de aprendizaje o entrenamiento.

- Se realiza la fase de reconocimiento, donde se evalúa la efectividad del algoritmo (método) implementado.

La efectividad del método seleccionado será evaluada mediante el uso de los indicadores de los sistemas de reconocimiento.

Los procedimientos antes señalados describen en forma general el funcionamiento del programa *RECANUM.*

La estructura del programa permite implementar nuevos algoritmos de reconocimiento de caracteres, para lo cual el diseñador deberá añadir las sentencias necesarias que hagan posible su funcionamiento.

RECANUM fue editado, depurado y convertido en un archivo ejecutable (*RECANUM.EXE*) mediante la utilización de un software computacional QuickBasic, y la parte correspondiente al interfaz con el usuario fue editado en Visual Basic para luego enlazarlos en un solo programa ejecutable con el mismo nombre de *RECANUM.EXE.* El listado del programa se adjunta en el apéndice ***B***.

5.2. ESTRUCTURA Y DIAGRAMAS DE FLUJO DEL PROGRAMA

El programa *RECANUM* comprende 3 etapas que pueden ser invocadas una a la vez y repetidas indefinidamente. Estas son:

1) Selección del método de Reconocimiento
2) Fase de entrenamiento
3) Fase de Reconocimiento.

Se debe considerar que la fase de reconocimiento se efectuará únicamente cuando se haya realizado previamente la fase de entrenamiento. Pero una vez que se tengan datos en el archivo de entrenamiento, puede efectuarse cualquiera de las opciones.

Un diagrama de flujo general del programa *RECANUM* se presenta a continuación;

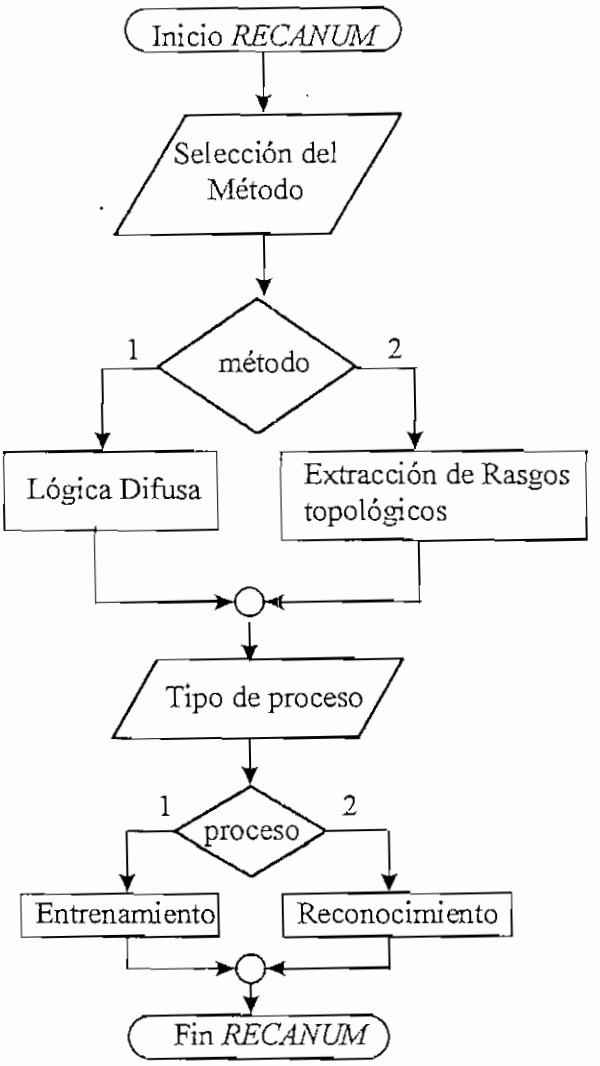


Fig 5.1 Diagrama de flujo general del Programa *RECANUM*.

5.2.1 Funcionamiento del Algoritmo de la fase Entrenamiento

En la figura 5.2 se muestra el diagrama de flujo mediante el cual se ejecuta el algoritmo de entrenamiento del programa de reconocimiento. Procedimiento que se lleva a cabo una vez que se ha seleccionado el método, luego del cual el programa se encarga de llevar a cabo el procedimiento de entrenamiento.

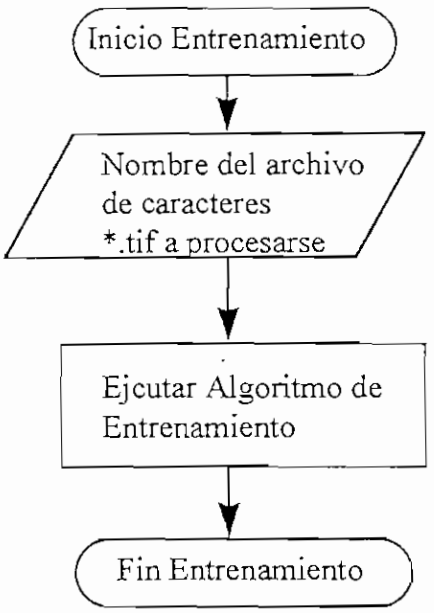


Fig. 5.2. Diagrama de flujo del proceso de entrenamiento.

5.2.2. Funcionamiento de la fase de Reconocimiento

Una vez que la fase de aprendizaje ha concluido, el programa esta listo para que la fase de reconocimiento pueda llevarse a cabo. En esta fase, es posible efectuar el reconocimiento tanto de los patrones de la población de entrenamiento, así como también de patrones de un conjunto particular de entrada (que no pertenecen a la población de entrenamiento), esto dependerá del archivo *.TIF que se ingrese para el reconocimiento. Es decir, el reconocimiento puede llevarse a cabo en dos modalidades, con:

1) Caracteres de un archivo de entrenamiento
2) Caracteres de un archivo sin previo entrenamiento.

Se debe tener en cuenta que el reconocimiento de caracteres sin previo entrenamiento puede permitirnos comprobar la eficiencia del método empleado con respecto a otro método en particular.

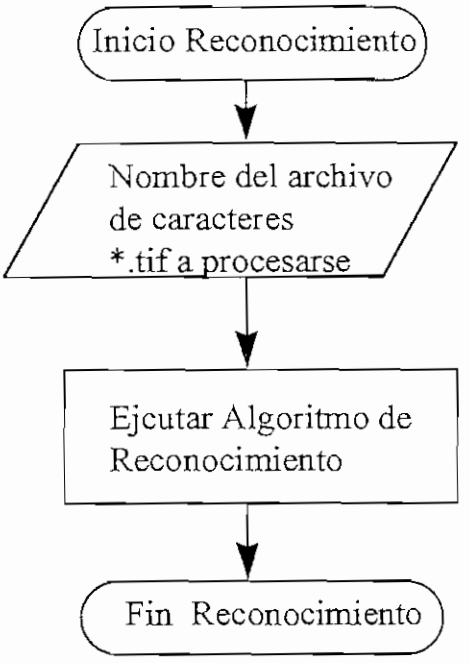


Fig. 5.3. Diagrama de flujo del procesamiento de Reconocimiento

5.3 ORGANIZACIÓN DE LOS DATOS DEL PROGRAMA

Los datos utilizados para la ejecución de los métodos de reconocimiento implementados en el programa *RECANUM* se almacenan en archivos que se encuentran en subdirectorios contenidos en el directorio raíz (C:\).

Los datos de las muestras de los caracteres, es decir archivos de tipo *.TIF son almacenados en un directorio denominado TOMAS, el cual se encuentra dentro de directorio raíz, de la siguiente manera:

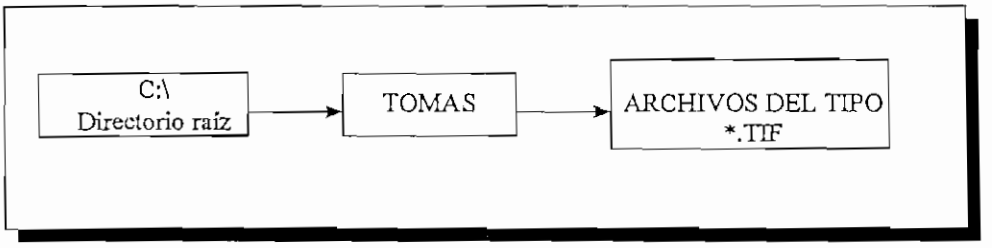


Fig. 5.4. Arbol de archivo de muestras *.TIF

Los datos provenientes de la ejecución de los procesos de entrenamiento son almacenados de la siguiente manera:

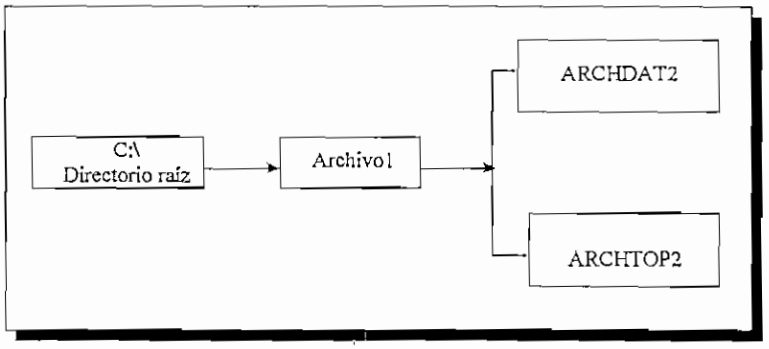


Fig. 5.5. Arbol de Archivo de datos provenientes del proceso de entrenamiento

Los archivos de datos ARCHDAT2 y ARCHTOP2, corresponden a los archivos que contienen los datos de los vectores de características provenientes del proceso de entrenamiento de los métodos de la Lógica Difusa y Extracción de Rasgos Topológicos respectivamente.

5.4. MANUAL DEL USUARIO

Para la persona que requiera operar el presente programa necesitará únicamente familiarizarse con las nociones básicas de reconocimiento de patrones, debido a que necesitará entender lo que es entrenamiento y reconocimiento, estas nociones se las presentan en el capítulo III.

5.4.1 Descripción del programa

El presente programa (*RECANUM*) nos permite simular dos tipos de sistemas de reconocimiento:

1) Utilizando La Lógica Difusa, y
2) Utilizando la Extracción de Rasgos Topológicos.

Estos métodos se describen en el capítulo IV.

Las muestras para el procedimiento de entrenamiento y reconocimiento se encuentran en imágenes obtenidas a través de un scanner mediante un formato *.TIF,

imágenes que deben cumplir con las especificaciones de rastreo definidas en la sección 5.4.4 (Scanner).

5.4.2 Instalación del Software

RECANUM es un programa que requiere ciertas condiciones mínimas para su funcionamiento:

- Microprocesador 386 o superior
- 640 Kbytes de memoria RAM convencional
- Monitor gráfico VGA o superior
- Scanner que permita la salida de archivos *.tif de dos niveles (1= blanco,0= negro) y resolución de 300 pixels por pulgada.
- Sistema operativo DOS, o Windows 95

El programa fue simulado en un computador DTK compatible con procesador 586 de 100 MHz con 8 Mbytes de memoria RAM, Monitor SVGA y bajo el sistema operativo Windows 95.

Es importante tomar en cuenta que, los archivos de las imágenes (*.tif) provenientes del Scanner necesitan de una buena cantidad de memoria disponible, por lo que se aconseja almacenarlos en el disco duro.

5.4.3 Funcionamiento del programa

Para que el programa de reconocimiento se ejecute se deberá activar el ícono correspondiente a *RECANUM* en la pantalla principal del WINDOWS 95.

Una vez ingresado en el programa, aparece una pantalla de presentación como se muestra a continuación;

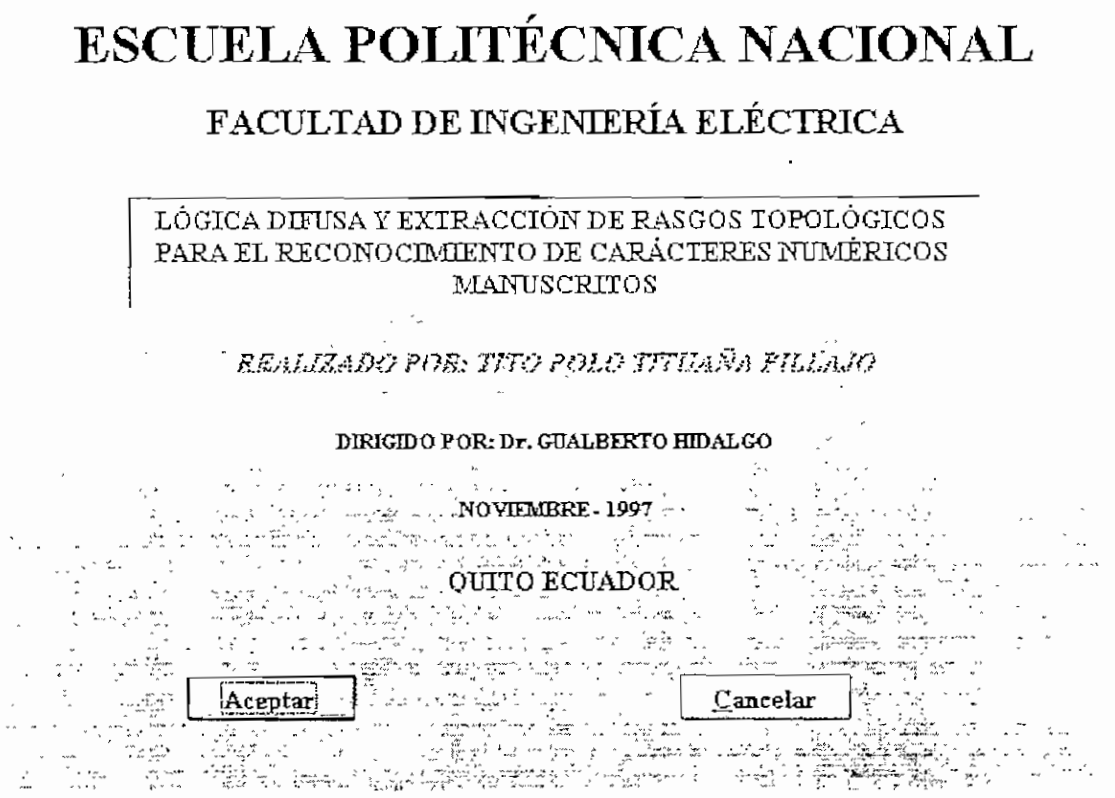


Fig. 5.6. Pantalla de presentación del programa *RECANUM*

La pantalla presenta dos opciones, ACEPTAR y CANCELAR. Con la activación de CANCELAR, el programa retorna a la pantalla de inicio del Windows 95. Mediante la activación de ACEPTAR, el programa permite continuar con la ejecución del programa. Luego del cual se ingresa a otra pantalla correspondiente al menú de opciones como se muestra a continuación;

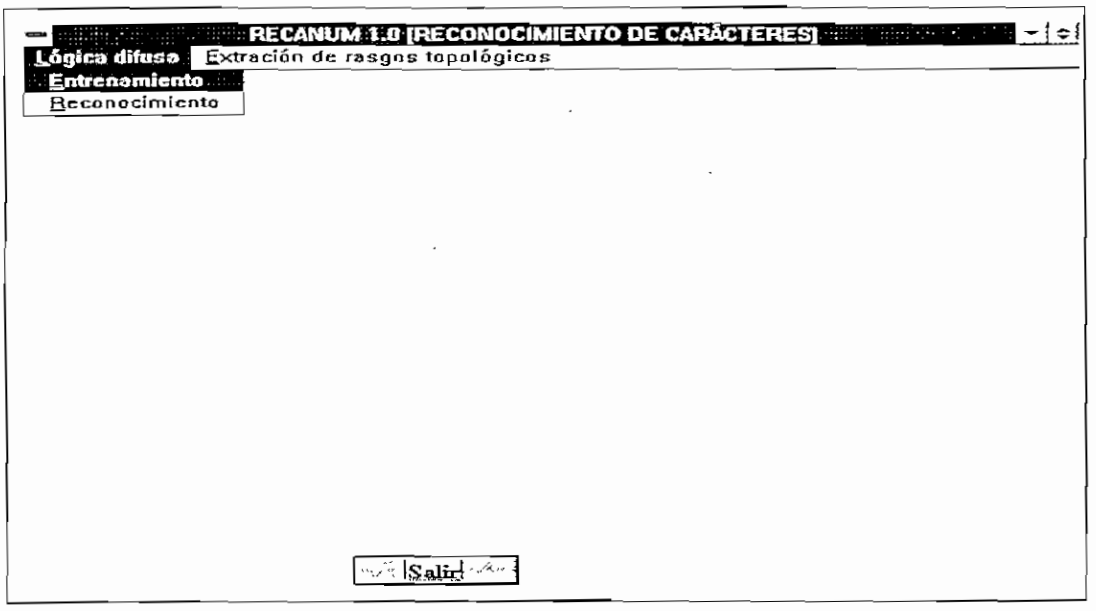


Fig. 5.7. Pantalla Menú de opciones

Esta pantalla de menú permite escoger los métodos, procedimientos y la opción de SALIR del programa.

Si se desea salir del programa, bastará pulsar el cuadro de control SALIR. En el caso de continuar con el programa, se deberá escoger uno de los dos métodos que se encuentran desplegados en el menú (Lógica Difusa, Extracción de Rasgos Topológicos), pulsando con el botón izquierdo del ratón la región especificada, luego de la cual se despliega un submenú que permite escoger entre ENTRENAMIENTO / RECONOCIMIENTO. De la misma manera se tiene que activar con el ratón una de las dos opciones, luego de lo cual se muestran en la pantalla las opciones escogidas por el usuario dentro de marcos (Método, Procedimiento) y además aparecerá un cuadro de control con la opción ACEPTAR como puede observarse en la pantalla del menú de opciones, que se muestra a continuación.

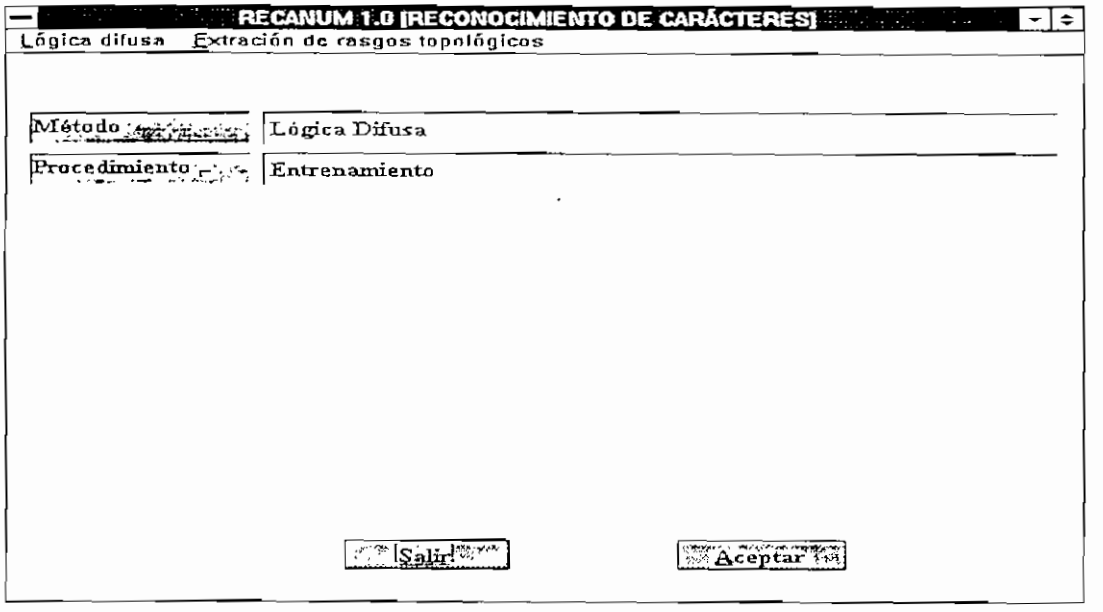


Fig. 5.8 Pantalla de opciones seleccionadas

Pulsando con el ratón la opción de Aceptar permitirá continuar con el programa. Cualquiera sea el método elegido y cualquiera que sea el tipo de procesamiento elegido, se pide ingresar por teclado el nombre del archivo a procesarse del tipo *.TIF.

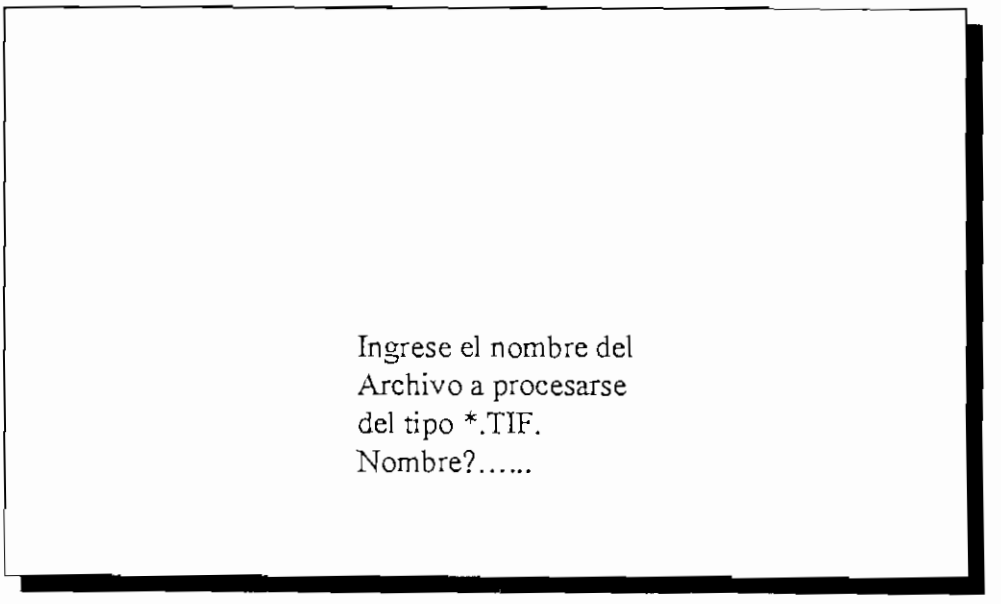


Fig. 5.9 Pantalla de pedido de archivo *. TIF a procesarse

Con este dato el programa empieza el procesamiento elegido, luego de lo cual el programa emite datos. Así, si el proceso es de Reconocimiento, el programa entregará como resultado el dato del numeral reconocido o rechazado de la siguiente manera;

RECONOCIMIENTO:

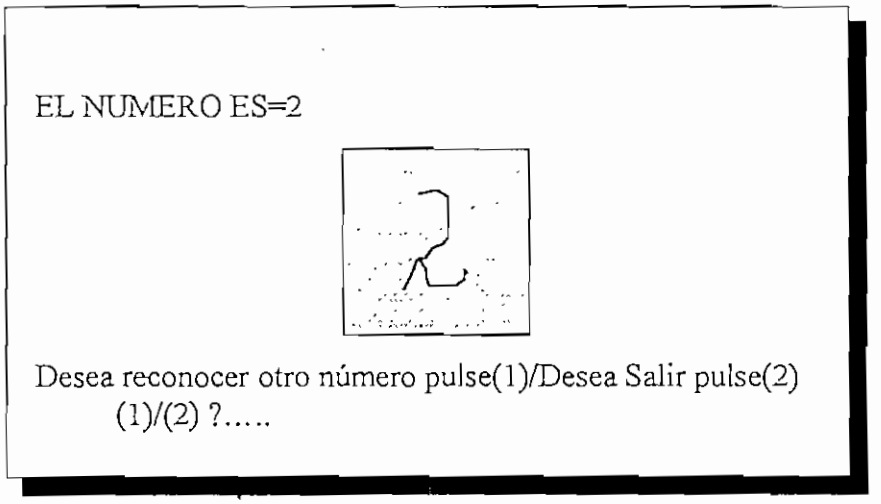


Fig. 5.10 Pantalla de salida de datos para carácter reconocido

Si el numeral procesado no encontró un vector de características isomórfico en la memoria prototipo o banco de datos del entrenamiento, es decir no reconoció el caracter entonces se despliega en la pantalla lo siguiente.

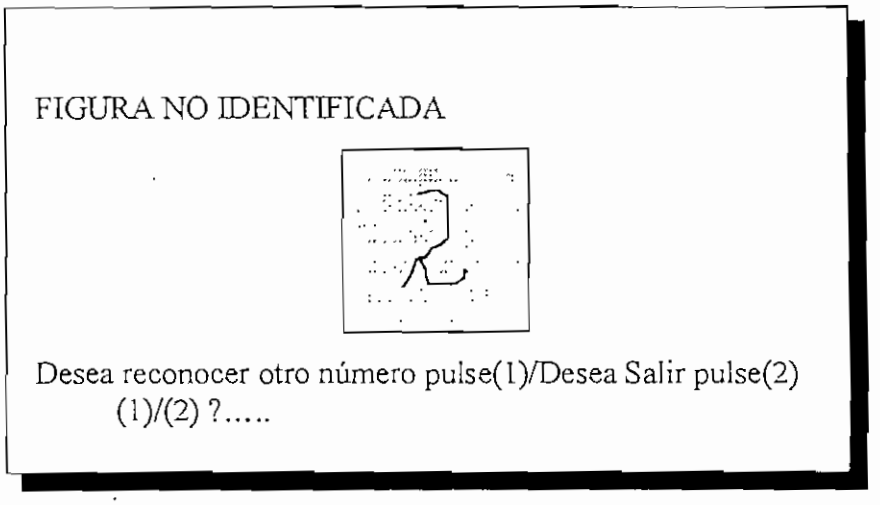


Fig. 5.11 Pantalla de salida de datos para carácter no reconocido

Si el proceso elegido fue de entrenamiento entonces la pantalla de salida luego de procesar 100 caracteres contenidos en un archivo *.tif será;

ENTRENAMIENTO:

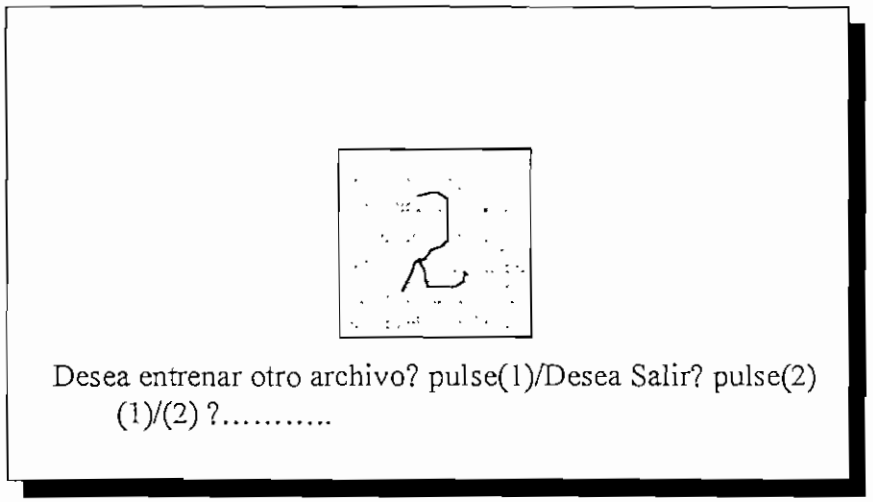


Fig. 5.12 Pantalla de salida para entrenamiento de un archivo *.tif

5.4.4 Procedimiento para la simulación del programa *RECANUM*

El proceso de simulación consta de los siguientes pasos:

1) ***Obtención de los datos.***- En unas hojas blancas con una cuadrícula previamente diseñada se escriben los caracteres empleando un marcador negro (fino). Se debe tener en cuenta que el bolígrafo a utilizarse debe ser tipo marcador, ya que este permite que se tenga una buena continuidad en los rasgos de los caracteres obtenidos. Estos caracteres son obtenidos de diferentes personas, de tal manera que se tenga una muestra poblacional muy variada, es decir caracteres con diferentes estilos de escritura. Para el presente trabajo se han utilizado más de 70 ejemplos de caracteres de la misma clase para el caso de la Lógica difusa y más de 200 ejemplos de la misma clase para el caso del método de la extracción de Rasgos Topológicos, pero es fundamental tener en cuenta que mientras se tenga una muestra de entrenamiento más variada se tendrá mejores resultados.

2) ***Scanner.***- Las muestras han sido obtenidas sobre papel bond tamaño A4, la cual contiene una sola cuadrícula disponible para 100 caracteres. Esta cuadrícula debe ser

rastreada (scanning) de tal manera que solo se detecte la cuadrícula. Por lo que, *para el funcionamiento del programa RECANUM se debe rastrear la imagen exclusivamente* con los siguientes parámetros:

- Archivo del tipo .tif, de dos niveles (blanco y negro)
- Resolución 300 pixel/pulgada
- Ancho de la toma 5.3 Pulgadas (inch)
- Altura de la toma 4.79 Pulgadas (inch)
- Escala 100%

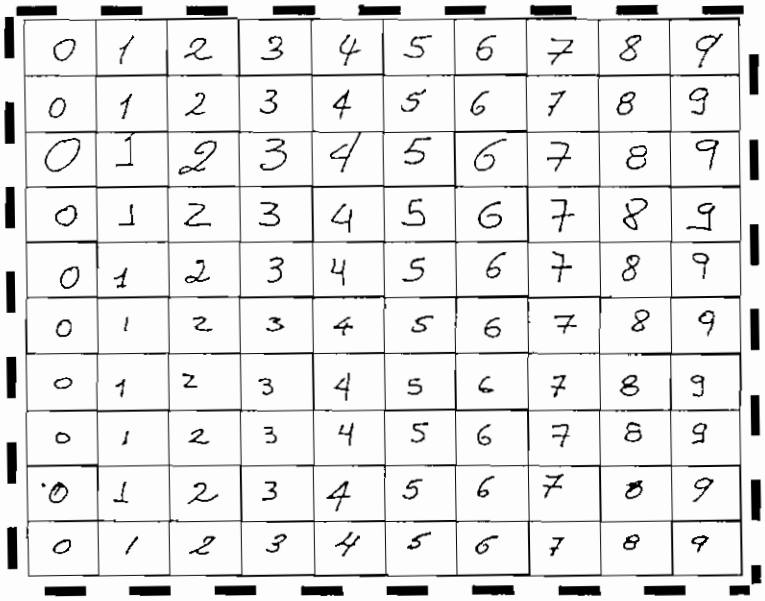


Fig. 5.13 Región a ser rastreada

El recuadro con línea segmentada indica la región a ser rastreada, y este debe ser de las dimensiones antes especificadas (5,3×4.79) pulgadas, recuadro que permite obtener únicamente la cuadrícula de caracteres.

NOTA: SE DEBE RECORDAR QUE ESTAS CONDICIONES DE RASTREO DEBEN SER CUMPLIDAS PARA EL CORRECTO FUNCIONAMIENTO DEL PROGRAMA.

3) *Método.*- En el menú principal se escoge el método que se desea simular LÓGICA DIFUSA/EXTRACCIÓN DE RASGOS TOPOLÓGICOS.
4) *Entrenamiento.*- Una vez definido el método se selecciona en el menú la opción ENTRENAMIENTO, luego de lo cual se llevará a efecto este proceso.

5) *Reconocimiento.*- Una vez definido el método se selecciona en el menú la opción RECONOCIMIENTO, esta opción permite reconocer caracteres pertenecientes tanto al conjunto de la población de entrenamiento, como de caracteres no pertenecientes al conjunto de entrenamiento es decir caracteres no entrenados.

6.1 OBTENCIÓN DE LAS MUESTRAS DE LOS CARACTERES MANUSCRITOS

Las muestras de los caracteres fueron obtenidas en hojas de papel bond tamaño A4 en la cual se tiene una cuadrícula, la cual contendrá 100 caracteres. Las muestras pertenecen a diferentes tipos de personas de diversas condiciones tales como edad sexo, cultura, procedencia, etc.

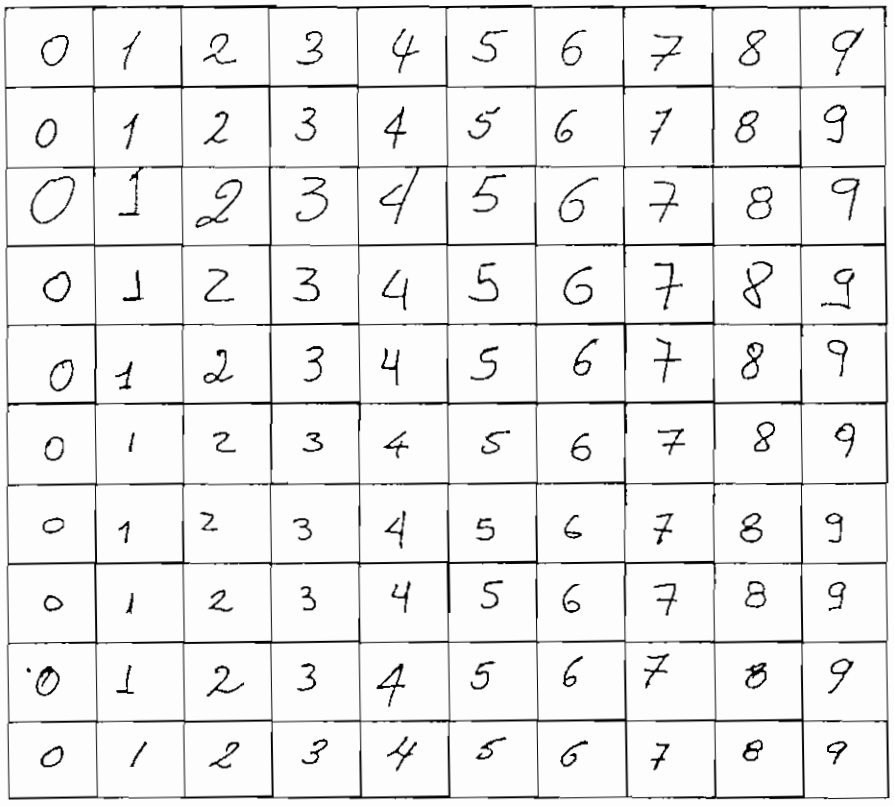


Fig. 6.1 Cuadrícula digitalizada con 100 muestras de caracteres numéricos.

Cada hoja de datos contiene 100 caracteres dispuestos en 10 filas y 10 columnas, y su dimensión es de 13,7 × 12,4 cm. En cada recuadro se escribe un número y cada fila contiene los 10 caracteres (0, 1, 2, 3, 4, 5, 6, 7, 8, 9) que corresponden a una persona como se muestra en la figura 6.1. Esta cuadrícula con los 100 caracteres fue digitalizada y almacenada en un archivo del tipo *.tif mediante la utilización de un scanner.

Se recopilaron 2000 muestras de caracteres para el entrenamiento utilizando el método de La Extracción de Rasgos Topológicos. De este mismo conjunto de muestras,

se utilizaron 700 muestras para el entrenamiento por el método de laLógica Difusa . Para probar el desempeño de los dos métodos propuestos se obtuvieron por separado otro conjunto de muestras.

6.2 DATOS Y RESULTADOS DE LA SIMULACIÓN DE LOS MÉTODOS DE RECONOCIMIENTO UTILIZADOS

Para poner a prueba el funcionamiento de los dos métodos de reconocimiento se emplearon las mismas muestras de caracteres de entrenamiento tanto para el caso del método de la Lógica difusa como para el caso del reconocimiento utilizando Rasgos Topológicos. Y además otro conjunto de muestras para el caso de reconocimiento con muestras sin entrenamiento previo. En las siguientes secciones se presentan los resultados de la simulación.

6.2.1 Resultados obtenidos empleando el Método de la Lógica Difusa

■ **Entrenamiento**

El entrenamiento se llevó a cabo con 700 muestras de caracteres, es decir 7 hojas de muestras, empleándose aproximadamente 75 minutos en el procesamiento de cada hoja. El tiempo utilizado en el procesamiento es el empleado por un computador 586 de 100 MHz.

■ **Reconocimiento**

TABLA I

Tasas de reconocimiento, rechazo y error para el Método de la Lógica Difusa.

Tipo de caracteres de Reconocimiento	Tasa de Reconocimiento (%)	Tasa de Rechazo (%)	Tasa de Error (%)
No entrenados*	78.6	8.6	12.8
Entrenados**	93.75	0	6.25

- ** Reconocimiento efectuado utilizando caracteres de entrada que pertenecen al conjunto de caracteres de entrenamiento.
- * Reconocimiento efectuado utilizando caracteres de entrada que no pertenecen al conjunto de entrenamiento.

6.2.2 Resultados obtenidos empleando el método de la Extracción de Rasgos Topológicos

■ **Entrenamiento**

El entrenamiento se llevó a cabo con 2000 muestras de caracteres empleándose como en el caso anterior, 75 minutos en el procesamiento de cada hoja.

■ **Reconocimiento**

TABLA II

Tasas de reconocimiento, rechazo y error para el Método de la Extracción de Rasgos Topológicos.

Tipo de caracteres de Reconocimiento	Tasa de Reconocimiento (%)	Tasa de Rechazo (%)	Tasa de Error (%)
No entrenados*	56.5	36	7.5
Entrenados**	99.5	0	0.5

- ** Reconocimiento efectuado utilizando caracteres de entrada que pertenecen al conjunto de caracteres de entrenamiento.
- * Reconocimiento efectuado utilizando caracteres de entrada que no pertenecen al conjunto de entrenamiento.

6.3 EVALUACION DE LOS RESULTADOS OBTENIDOS DE LA SIMULACIÓN

Una tasa de reconocimiento ideal del 100% podría ser obtenida si los rasgos que caracterizan a cada clase de carácter fueran distintos. Sin embargo, en los caracteres manuscritos esta situación ideal generalmente no ocurre, ya que debido a la forma en que una persona escribe, diferentes caracteres pueden parecerse. Esta situación crea caracteres ambiguos, y por lo tanto se produce también una confusión en los rasgos que representan a cada carácter.

6.3.1 Evaluación de los resultados obtenidos utilizando el método de la Lógica Difusa

Se pudo apreciar que a medida que se eleva el número de caracteres de entrenamiento la tasa de reconocimiento aumenta, pero es importante señalar que también se incrementa el tiempo de entrenamiento, ya que como se mencionó anteriormente se necesita aproximadamente 47 segundos para el procesamiento de cada carácter.

Se puede apreciar en los dos casos que la tasa de reconocimiento es superior para el caso de la simulación con caracteres previamente entrenados.

Debido a la situación explicada en la sección 6.3 relacionada con la ambigüedad, se producen ciertos errores en el reconocimiento, especialmente con los siguiente caracteres:

TABLA III

Cuadro de caracteres de posible clasificación errónea

Caracter Real	Clasificado como
0	8
0	6
4	9
5	6
7	3
9	4

La columna correspondiente a "caracter real" indica que es el caracter de entrada a ser clasificado o reconocido, y la columna de Carácter "clasificado como" indica el carácter que le fue asignado.

El presente cuadro nos permite observar los posibles caracteres que pueden ser clasificados erróneamente, debido a que sus vectores de características coinciden. Pero como se puede observar de la tabla I, la tasa de clasificación errónea es relativamente baja.

6.3.2 Evaluación de los resultados obtenidos utilizando el método de la Extracción de Rasgos Topológicos

El reconocimiento mediante este proceso necesita de una buena cantidad de caracteres de entrenamiento, de tal manera que para la simulación de reconocimiento se necesitarán al menos 2000 caracteres de entrenamiento. Para la simulación se utilizó un banco de datos de entrenamiento de 2000 caracteres, compuesto de 200 muestras de cada numeral.

En este procedimiento se puede apreciar claramente que mientras se incrementa el número de caracteres de entrenamiento, la tasa de reconocimiento se

incrementa considerablemente, de tal manera que se puede concluir que mientras se tenga un gran banco de datos de entrenamiento, el presente método entregará una aceptable tasa de reconocimiento. El gran banco de datos disminuirá el valor de la tasa de rechazo, incrementándose la tasa de reconocimiento.

En la simulación de reconocimiento también se produjeron errores por efecto de la ambigüedad entre caracteres, los cuales pueden resumirse en el siguiente cuadro

TABLA IV

Cuadro de caracteres de posible clasificación errónea

Posible Carácter	Clasificado como
0	8
5	9
4	9
9	4

6.4 CONCLUSIONES

Mediante la realización de la presente tesis se ha tratado de contribuir en alguna manera a la enseñanza de nuevos conceptos que puedan ser aplicados en nuestra rama de la Ingeniería Electrónica tanto en el presente como en el futuro, para de esta manera retribuir en algo los beneficios obtenidos en la Escuela Politécnica Nacional.

El presente trabajo permitirá también contribuir en la investigación de temas que puedan relacionarse con el Reconocimiento de Patrones, y de esta manera servir de referencia para trabajos futuros de investigación.

La importancia del tema desarrollado y el avance tecnológico basado en los Sistemas difusos, permite recomendar que estos conceptos sean abordados en alguna materia específica en nuestra Facultad de Ingeniería Eléctrica, ya que esto contribuye a la formación de un ingeniero.

El presente trabajo ha tratado de detallar la teoría sobre el Reconocimiento de patrones, y dentro de este campo, el reconocimiento de caracteres manuscritos. A partir de este estudio se han implementado dos métodos de reconocimiento de caracteres.

El reconocimiento de caracteres manuscritos ofrece muchos métodos, pero se debe tener en cuenta que para escoger uno u otro método, no se puede anticipar los resultados o lo que es lo mismo cual es el mejor, por lo que la evaluación experimental de ellos nos dará la respuesta de su eficiencia en relación a los demás.

De los métodos implementados se puede apreciar que cuando se han simulado con caracteres que no pertenecen al conjunto de entrenamiento, el método de la Lógica Difusa ha logrado una mejor tasa de reconocimiento en relación al método de la Extracción de Rasgos Topológicos. Además se necesita un número mucho menor de caracteres de entrenamiento para el caso de la Lógica Difusa lo que permite ahorrar una buena cantidad de memoria en el almacenamiento de los datos de entrenamiento. Por lo que se puede concluir que, el método de la lógica difusa posee la característica especial de *generalización.*

En la simulación con caracteres previamente entrenados, la tasa de reconocimiento para el método de La Extracción de Rasgos Topológicos, supera considerablemente al de la Lógica Difusa, y por ende su tasa de error es mínima Esto permite concluir que el método de la extracción de rasgos topológicos es un sistema *memorista.*

El método de reconocimiento basado en la extracción de rasgos topológicos, debido a que es un sistema muy memorista, puede ser útil para el reconocimiento o identificación de la persona quien escribió el numeral.

Los métodos de reconocimiento implementados pueden extendenderse al reconocimiento de los 27 caracteres alfabéticos, pero como se mencionó anteriormente, necesitaría una gran cantidad de memoria para el almacenamiento de los datos provenientes del proceso de entrenamiento.

La implementación de un sistema basado en los métodos utilizados en esta tesis, permitirá el desarrollo de un sistema más avanzado que sea capaz de reconocer: texto manuscrito, firmas en documentos, tarjetas de crédito, huellas digitales, etc.

Es importante recalcar la gran importancia del estudio de los Sistemas Difusos, ya que las últimas tecnologías que tienen que ver con nuestro campo de estudio que es de la electrónica, están basadas en este tema. Por lo tanto se debería realizar un esfuerzo para la obtención de una buena bibliografía para nuestras bibliotecas, ya que un aspecto fundamental para la investigación es el de tener al alcance la necesaria información.

El proceso de entrenamiento es de gran importancia, debido a que permite relacionar al reconocimiento de patrones con la inteligencia artificial, y más precisamente al reconocimiento de caracteres con la visión artificial y robótica.

Existen ciertos factores como el tipo de papel, tipo de tinta, estado de animo de la persona, edad, y muchos otros, que afectan el reconocimiento de caracteres manuscritos, los cuales aunque parezcan sin importancia, si influyen en esta tarea.

* Albuja D. Jorge. *Reconocimiento de caracteres manuscritos por medio de redes neurales y neuro-difusas artifiales.*Escuela Politécnica Nacional, Quito-Ecuador, 1995.

* Bezdek, J. C. y Pal, S. K. *Fuzzy Model for Pattern Recognition*, New York: *IEEE*, Inc, 1992

* Cox, Earl D. The Fuzzy Systems Handbook: A Practitioner's Guide to Building, Using and Maintaining Fuzzy Sistems, New York, 1994.

* Cox, Earl D. (1992). "Fuzzy Fundamentals." *IEEE Spectrum* (October), pp.58-61.

* Cox, Earl D. (1992). "Applications of Fuzzy System Models." *AI Expert* (October), pp. 34-39.

* Dubois and H. Prade, "Fuzzy sets in approximate reasoning, part 1: Inference with possibilty distributions," *Fuzzy sets Syst.*, vol. 40, 1991, pp. 143-202

* Fu K. S., Gonzalez R. C., and Lee C. S. G., *Robótica, Control, Detección, Visión e Inteligencia*, Mc Graw-Hill, 1990.

* Glauberman M. H., Character Recognition for business machine, *Electronics*, Febraury 1956.

* Ince D. C., *Ingeniería de Software,* Adison-Wesley Iberoamericana, Wilmington, Delaware, E.U.A., 1993.

* Jamshidi, M,. Vadiee, N., and Ross, Timothy J., Eds. (1993*). Fuzzy Logic and Control.* Prentice Hall, Englewood Cliffs, NJ.

* Jerry M. Mendel., *Fuzzy Logic Systems for Engineering*: A Tutorial, IEEE,1995.

* Kimura F. and Shridhar M., Handwritten numerical recognition based on multiple algorithms, *Pattern Recognition*, 1991.

* Klir, George J., and Folger, Tina A. (1988). *Fuzzy Sets, Uncertainty, and Information*, Prentice Hall, Englewood Cliffs, NJ.

* Kosko, Bart. *Neural Networks and Fuzzy Systems: A Dinamical Systems Approach to Machine Inteligence*, Prentice-Hall, Inc, 1992.

* Kuhl P., and Giardina C. R., Elliptic Fourier features of a closed contour, Comput. Vis. Graphics Image Process. **18**, 1982.

* Kosko, Bart and Isaka, Satoru (1993). "Fuzzy Logic," *Scientific American* (July), pp. 76-81.

* Lin C. S., and Hwang C. L., New forms of shape invariants from elliptic Fourier Descriptors, *Pattern Recognition* **20**(5), 1987.

* Maiers J. and Sherif Y. S., "Applications of Fuzzy Set Theory," *IEEE Trans. Syst.., Man, and Cyber.*,vol SMC-15, pp 175-189, Jan./Feb . 1995.

* Oivind Due Trier and Anil K. Jain, "Goal-Directed Evaluation of Binarization Methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 12, pp. 1191-1201, December 1995

* Rocha J. and Pavlidis T., "A Shape Analysis Model with Applications to a Character Recognition System," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 16, no. 4, pp. 393-404, Apr. 1993.

* Sargur N. Srihari and Stephen W. Lam, Character Recognition. *Center of excellence for Documente Analysis And Recognition*, State University of New York

* Sekita I., Toraichi K., Mori R., Yamamoto K., and Yamada H., Feature Extraction of handwritten Japanese characters by spline functions for relaxation matching, *Pattern Recognition*, **21** (1), 1998.

* Tou J. T., and Gonzalez R. C., "Recognition of handwritten characters by Topological feature extraction and multilevel characterization,"*IEEE Trans. Comput.*, vol C-21, July 1972

* Trier O. D., Jain A. K., and Taxt T., "Feature extraction methods for character recognition-a survey," *Pattern Recognition*, Vol 29, No 4, pp 641-662, 1996.

* Yager R. R. " On the implication operator in Fuzzy Logic," *Inf Sciences*, vol. 31, pp. 141- 164, 1983.

* Zadeh L. A., "The concept of a Linguistic variable and its Application to approximate reasoning," *Inf Sciences*, vol. 8, pp. 199-249, and vol. 9, pp. 43-80, 1975.

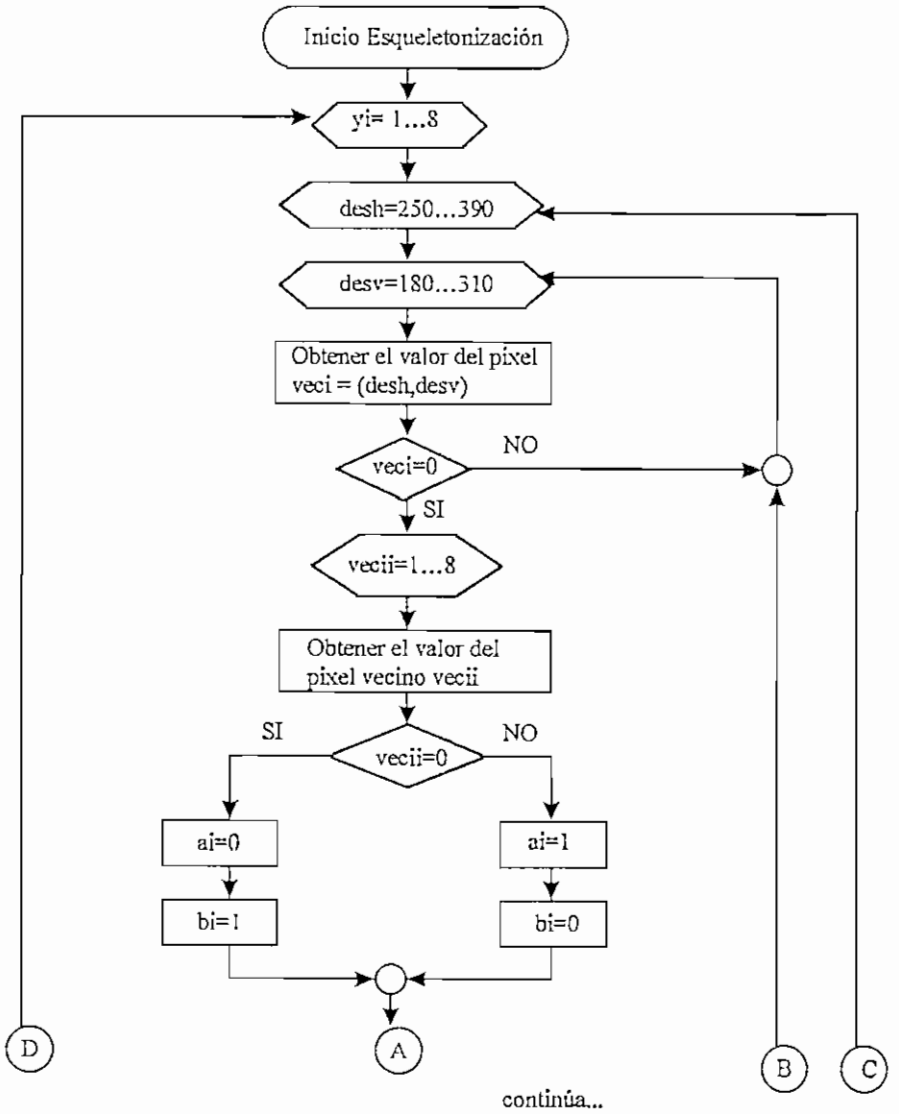
* Zadeh L. A. "Fuzzy Sets," *Information and Control*, vol. 8, pp 338-353, 1965.

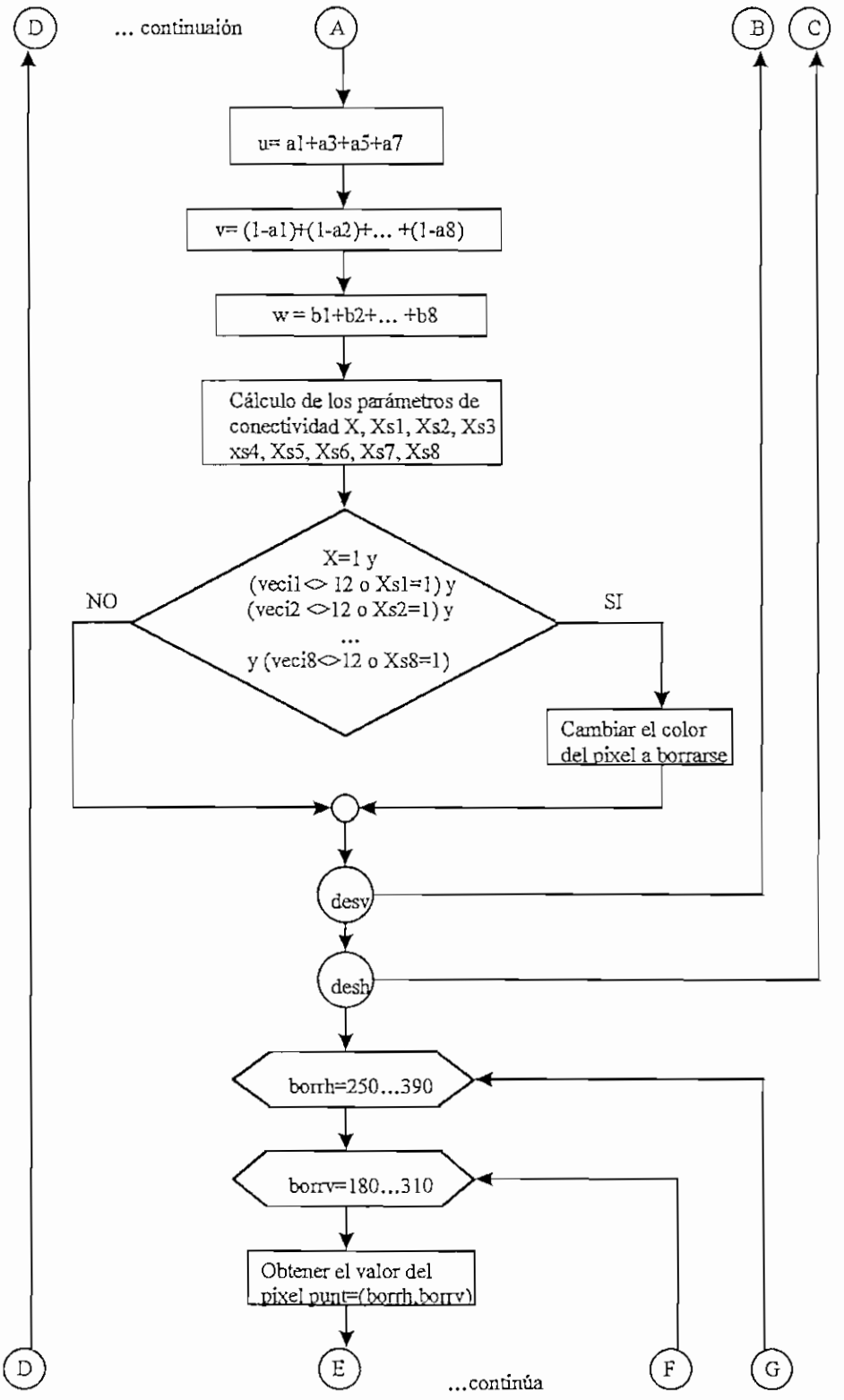
* Zadeh L. A. Fuzzy Algorithmic-Aproach, *Int. J. Man Machine Studies* (1976) 8,249-291.

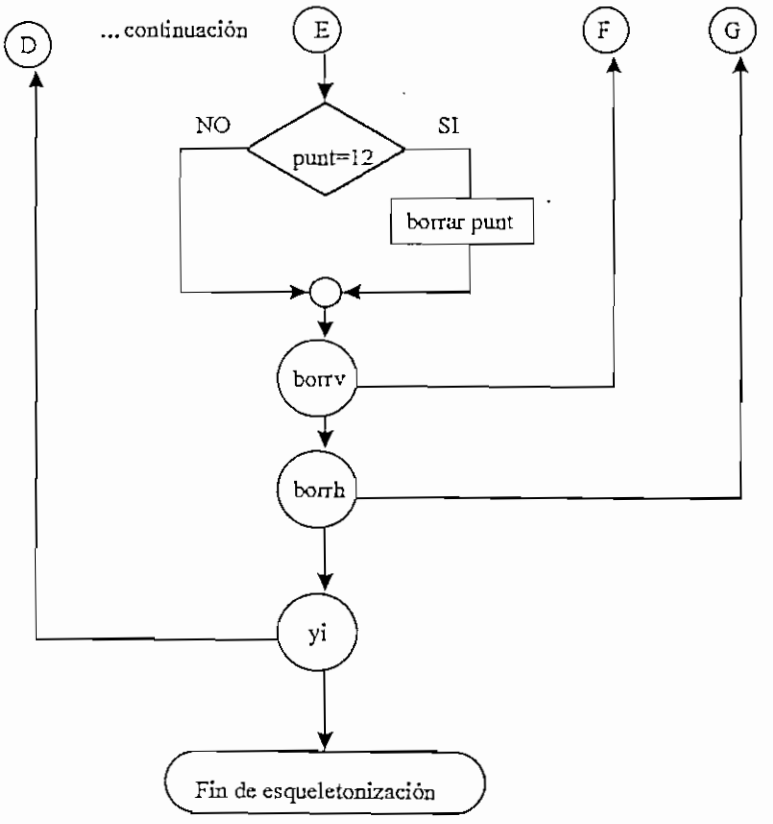
* Zadeh L. A. (1972). A Fuzzy-set-theoretic interpretation of linguistic hedges. *Journal of Cybernetics*,2 ,4.

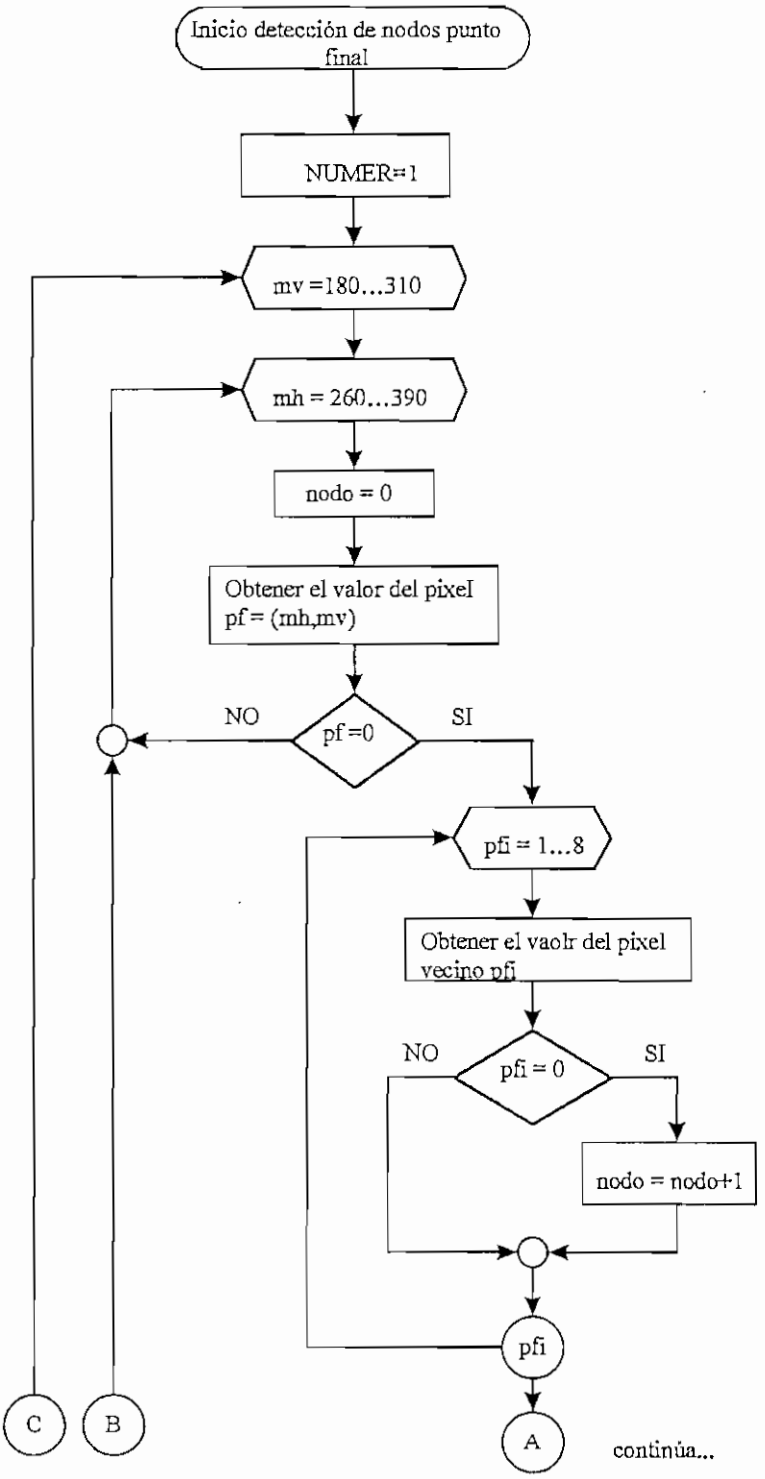
* Zimmermann H. J. Fuzzy Set Theory and its Applications, 2nd de. Boston, MA: Kluwer, 1991.

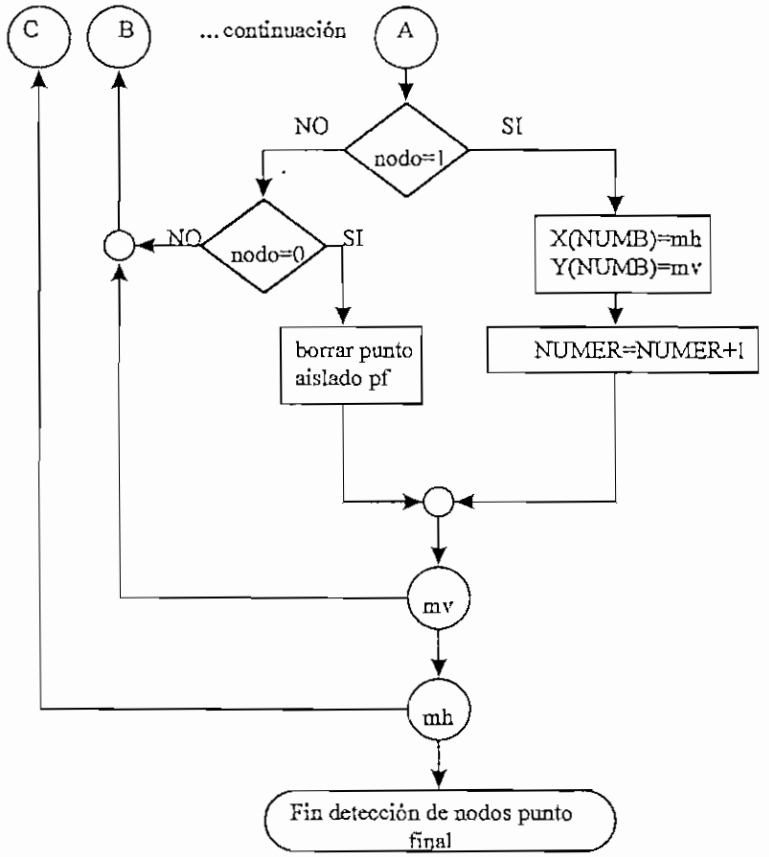
DIAGRAMAS DE FLUJO PARA EL MÉTODO DE LA LÓGICA DIFUSA

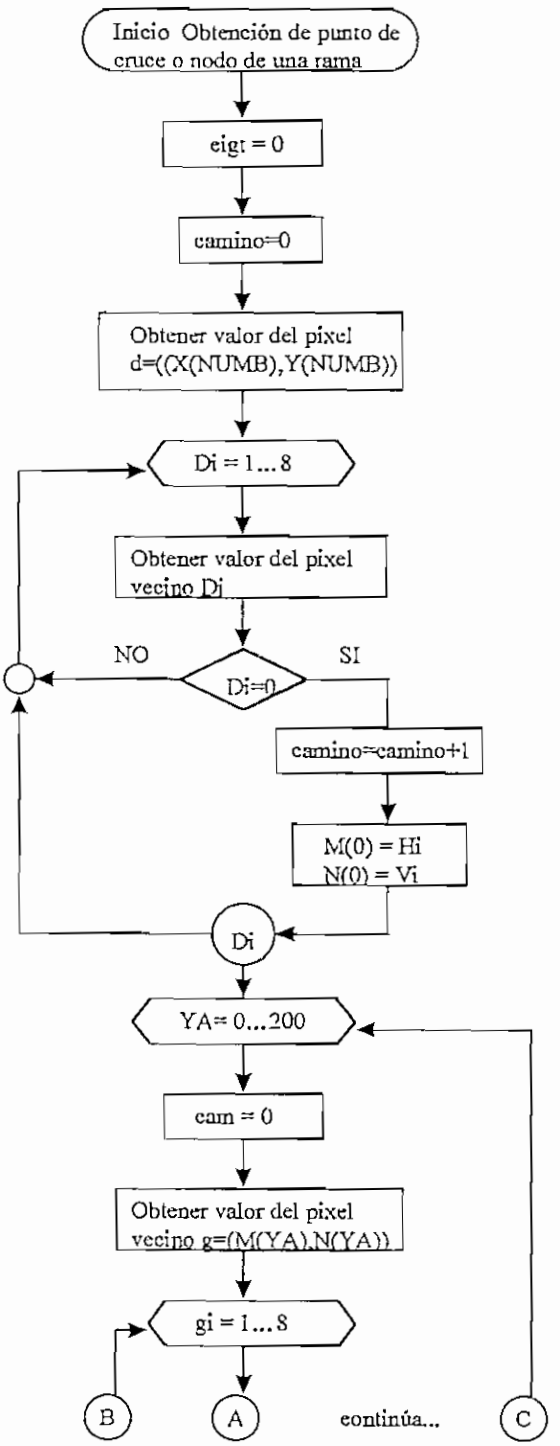


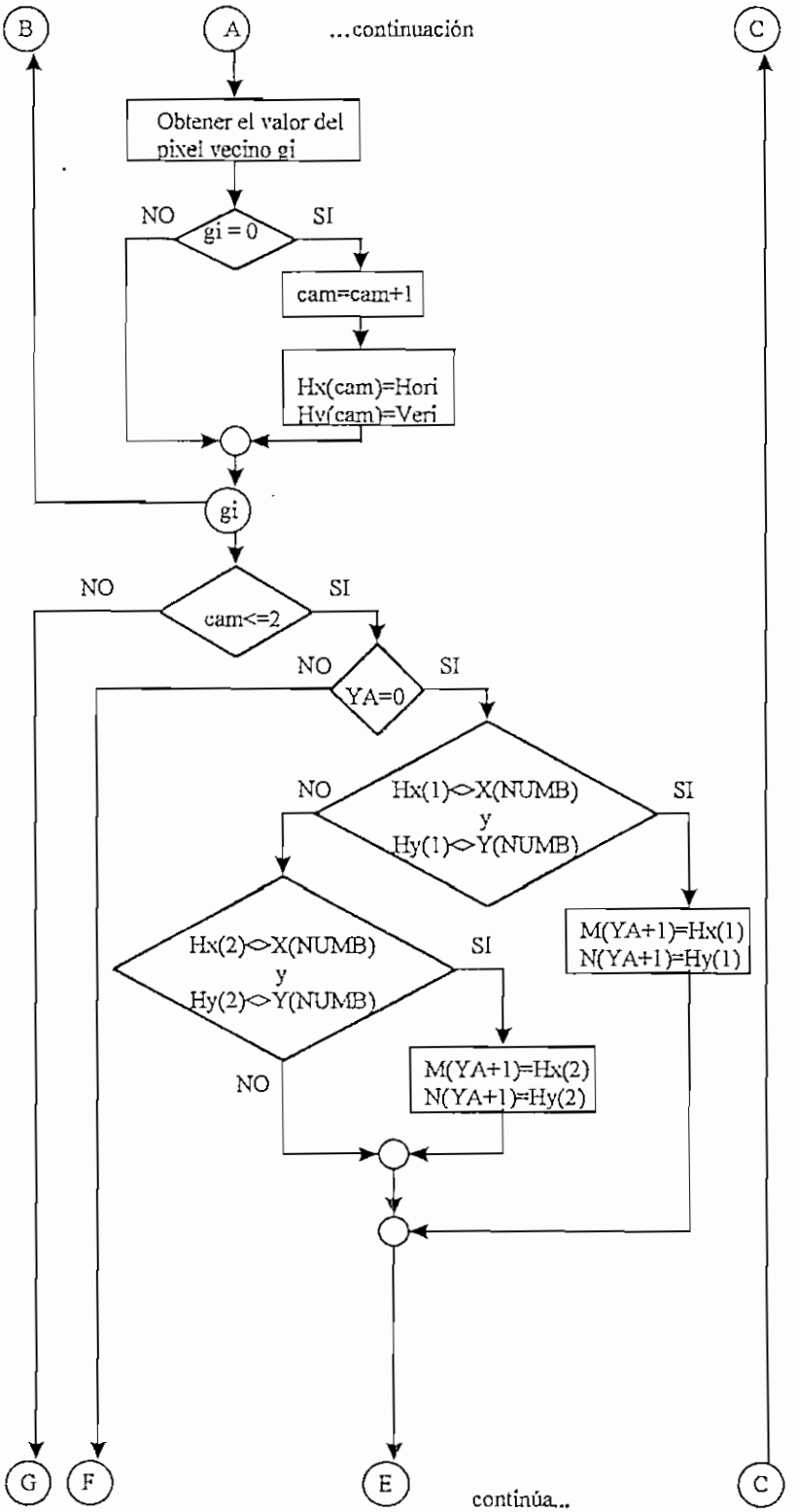

D
... continuaión
A
B
C
u= a1+a3+a5+a7
v= (1-a1)+(1-a2)+... +(1-a8)
w = b1+b2+... +b8
Cálculo de los parámetros de conectividad X, Xs1, Xs2, Xs3 xs4, Xs5, Xs6, Xs7, Xs8
X=1 y (veci1<> 12 o Xs1=1) y (veci2 <>12 o Xs2=1) y ... y (veci8<>12 o Xs8=1)
NO
SI
Cambiar el color del pixel a borrarse
desv
desh
borrh=250...390
borrv=180...310
Obtener el valor del pixel punt=(borrh,borrv)
D
E
...continúa
F
G

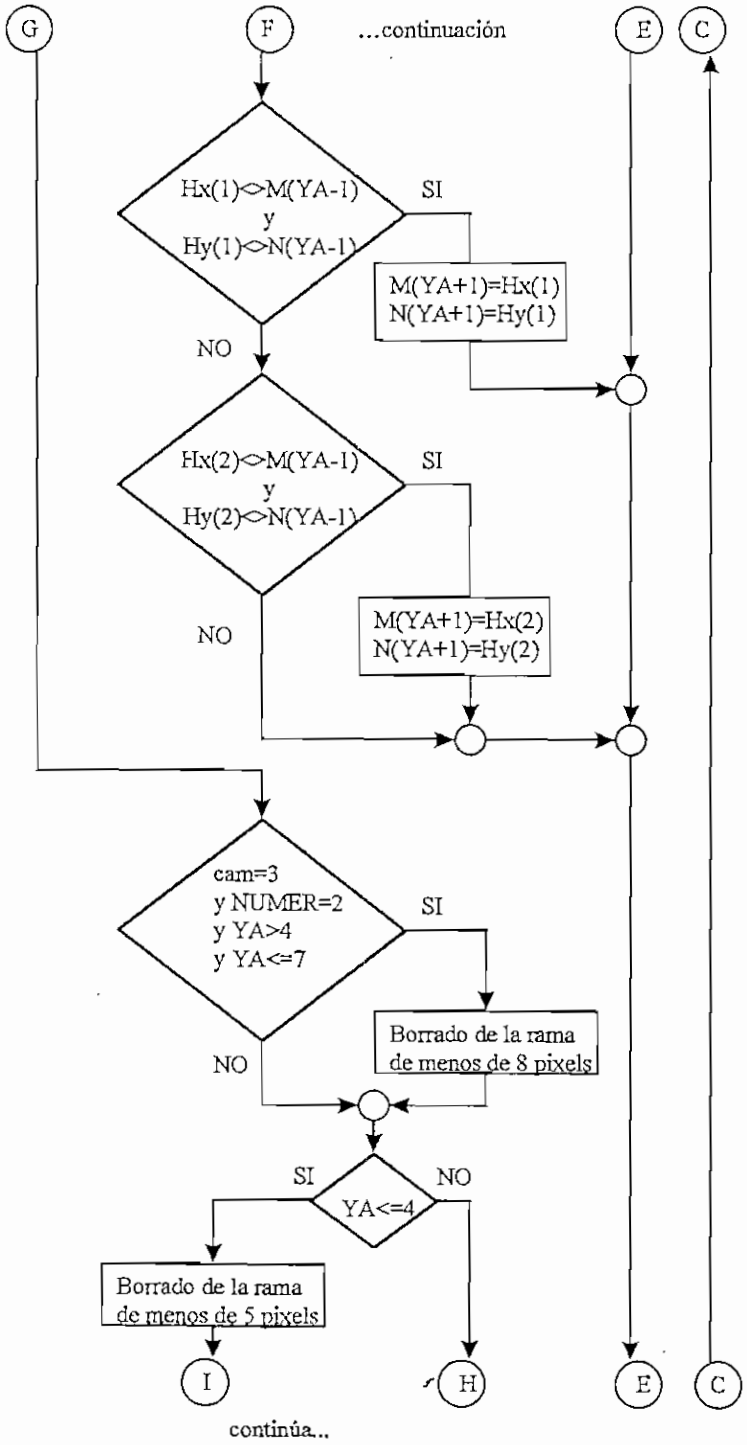

D
... continuación
E
F
G
NO
punt=12
SI
borrar punt
borrv
borrh
yi
Fin de esqueletonización

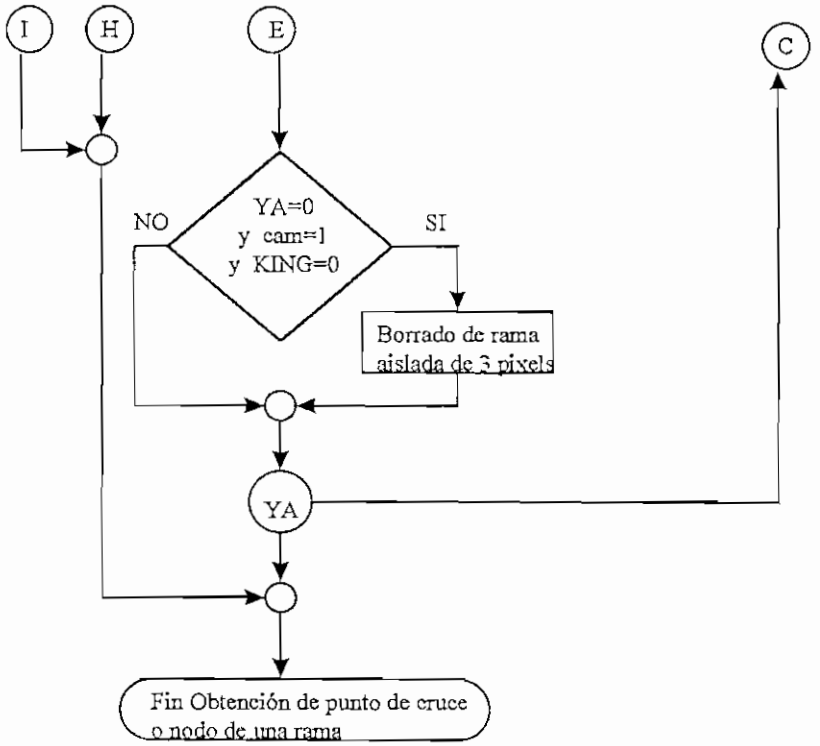

Inicio detección de nodos punto final
NUMER=1
mv =180...310
mh = 260...390
nodo = 0
Obtener el valor del pixel pf = (mh,mv)
NO
pf =0
SI
pfi = 1...8
Obtener el vaolr del pixel vecino pfi
NO
pfi = 0
SI
nodo = nodo+1
pfi
C
B
A
continúa...

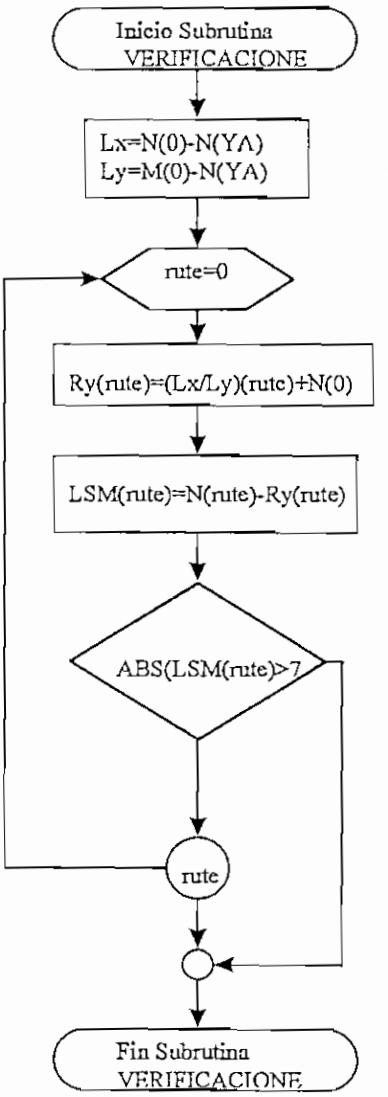

C
B
... continuación
A
NO
nodo=1
SI
NO
nodo=0
SI
X(NUMB)=mh
Y(NUMB)=mv
borrar punto aislado pf
NUMER=NUMER+1
mv
mh
Fin detección de nodos punto final

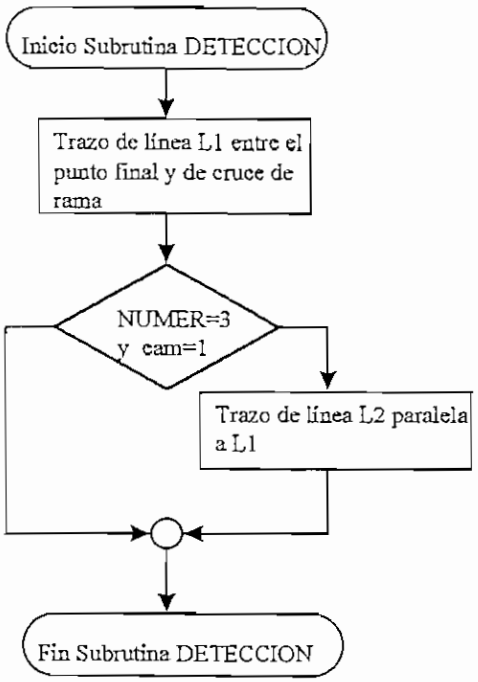

Inicio Obtención de punto de cruce o nodo de una rama
eigt = 0
camino=0
Obtener valor del pixel d=((X(NUMB),Y(NUMB))
Di = 1...8
Obtener valor del pixel vecino Di
NO
Di=0
SI
camino=camino+1
M(0) = Hi
N(0) = Vi
Di
YA= 0...200
cam = 0
Obtener valor del pixel vecino g=(M(YA),N(YA))
gi = 1...8
B
A
continúa...
C

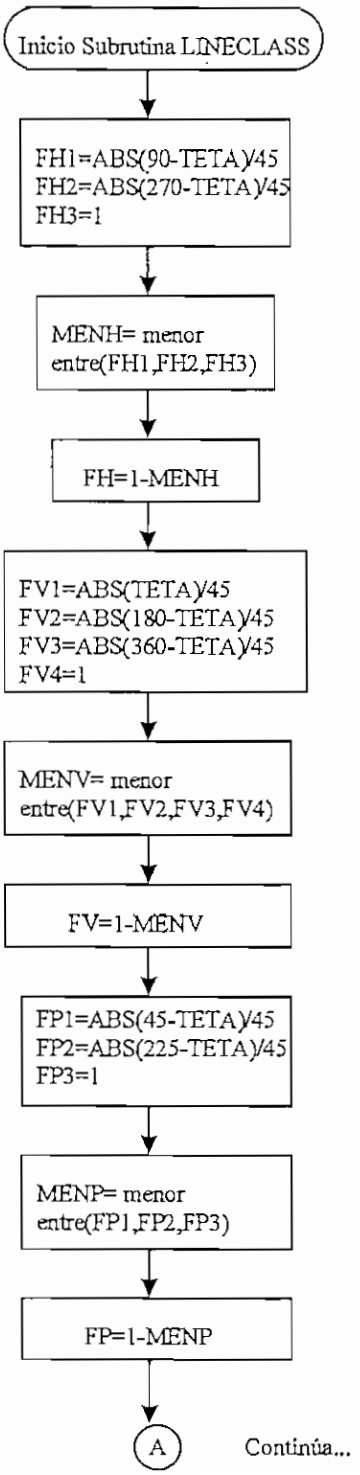

B
A
...continuación
C
Obtener el valor del pixel vecino gi
NO
gi = 0
SI
cam=cam+1
Hx(cam)=Hori
Hy(cam)=Veri
gi
NO
cam<=2
SI
NO
YA=0
SI
NO
Hx(1)<>X(NUMB)
y
Hy(1)<>Y(NUMB)
SI
M(YA+1)=Hx(1)
N(YA+1)=Hy(1)
Hx(2)<>X(NUMB)
y
Hy(2)<>Y(NUMB)
SI
M(YA+1)=Hx(2)
N(YA+1)=Hy(2)
NO
G
F
E
continúa...
C

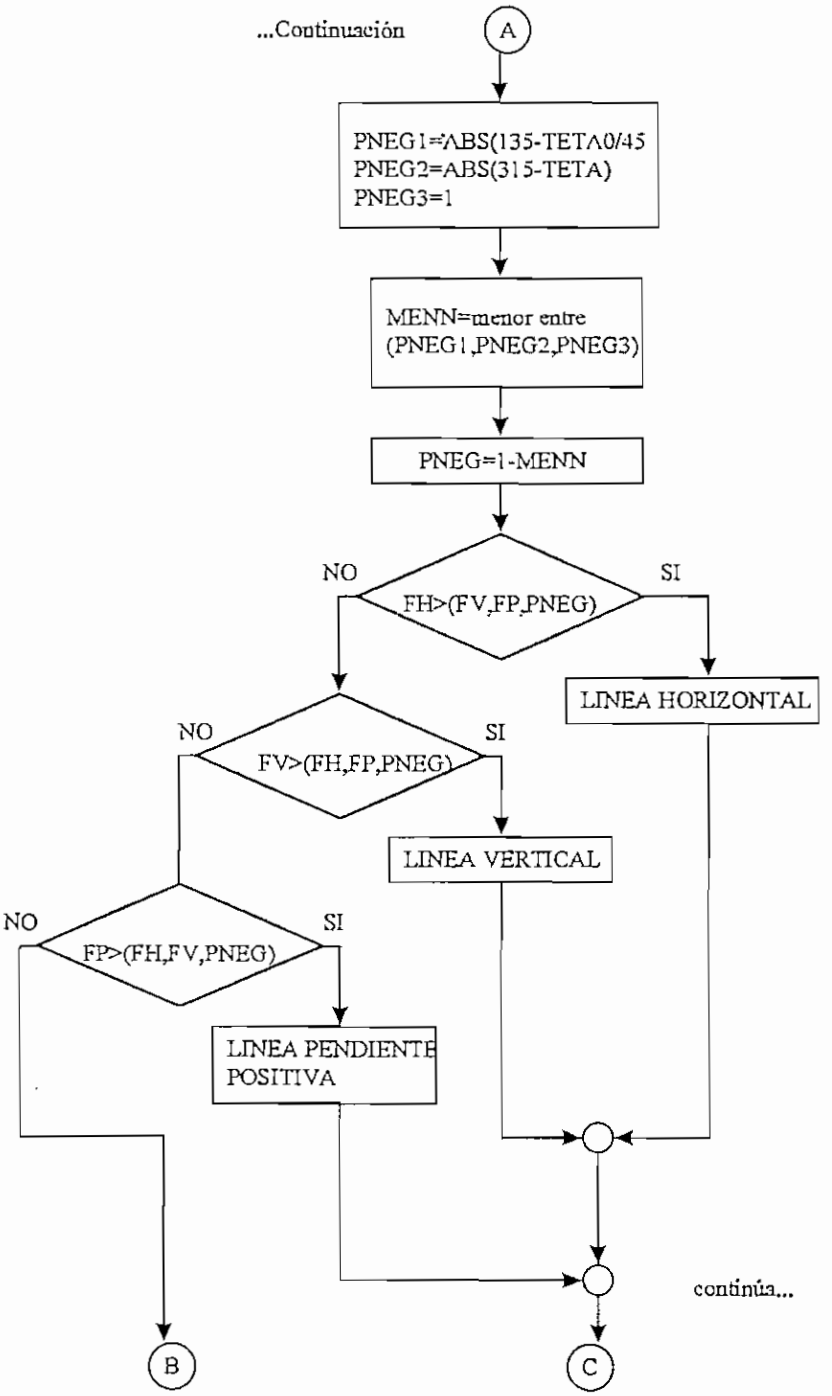

G
F
...continuación
E
C
Hx(1)<>M(YA-1)
y
Hy(1)<>N(YA-1)
SI
M(YA+1)=Hx(1)
N(YA+1)=Hy(1)
NO
Hx(2)<>M(YA-1)
y
Hy(2)<>N(YA-1)
SI
M(YA+1)=Hx(2)
N(YA+1)=Hy(2)
NO
cam=3
y NUMER=2
y YA>4
y YA<=7
SI
Borrado de la rama
de menos de 8 pixels
NO
SI
YA<=4
NO
Borrado de la rama
de menos de 5 pixels
I
H
E
C
continúa...

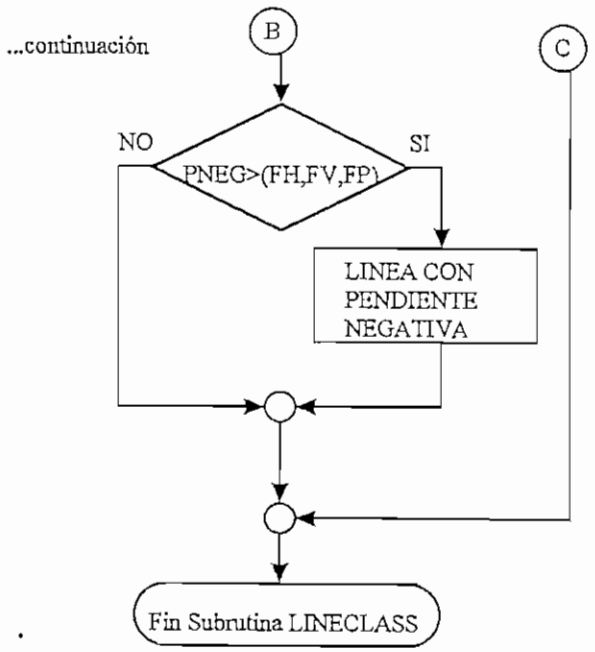

I
H
E
C
YA=0
y cam=1
y KING=0
NO
SI
Borrado de rama
aislada de 3 pixels
YA
Fin Obtención de punto de cruce
o nodo de una rama

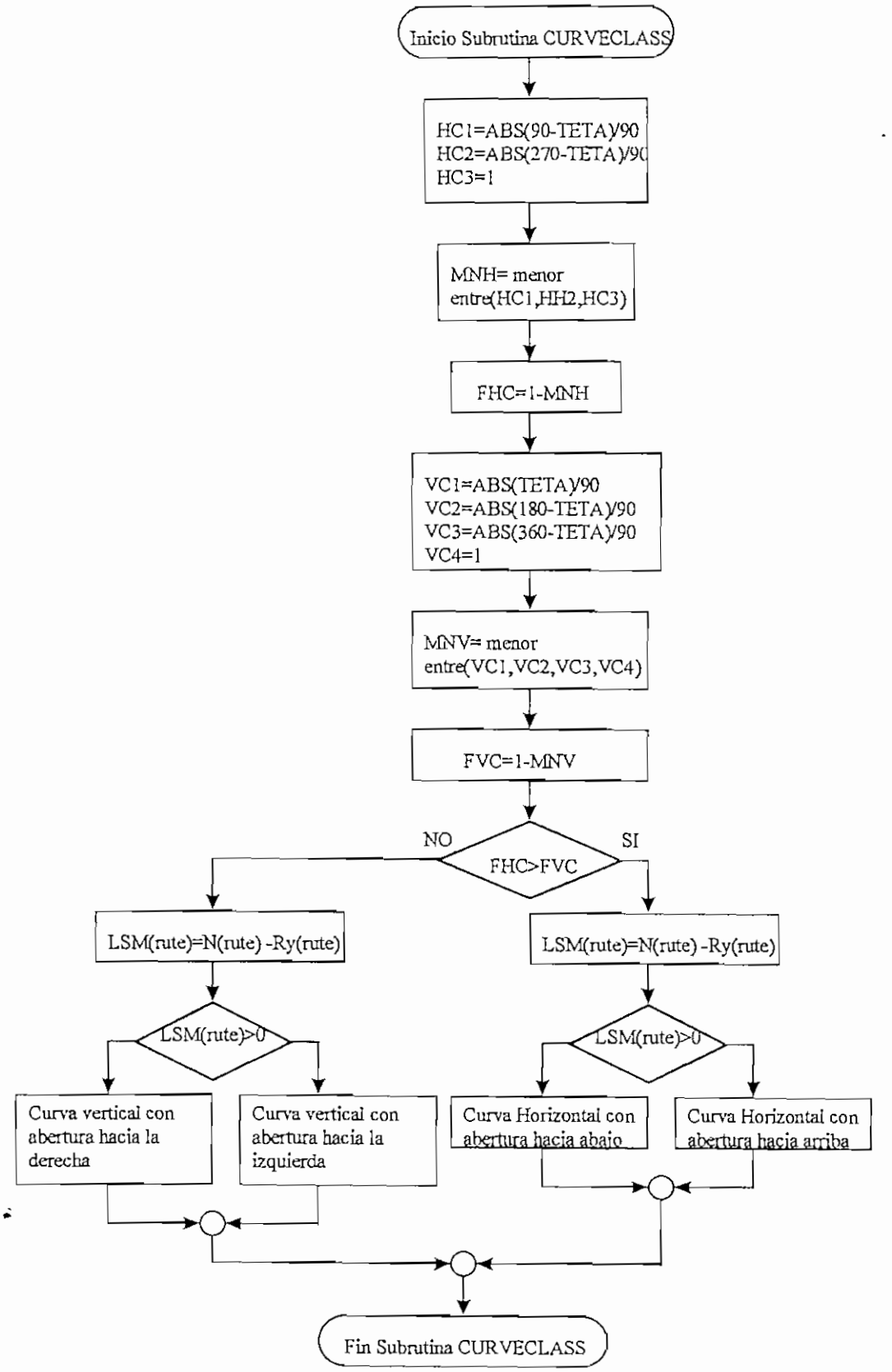

Inicio Subrutina
VERIFICACIONE
Lx=N(0)-N(YA)
Ly=M(0)-N(YA)
rute=0
Ry(rute)=(Lx/Ly)(rute)+N(0)
LSM(rute)=N(rute)-Ry(rute)
ABS(LSM(rute)>7
rute
Fin Subrutina
VERIFICACIONE

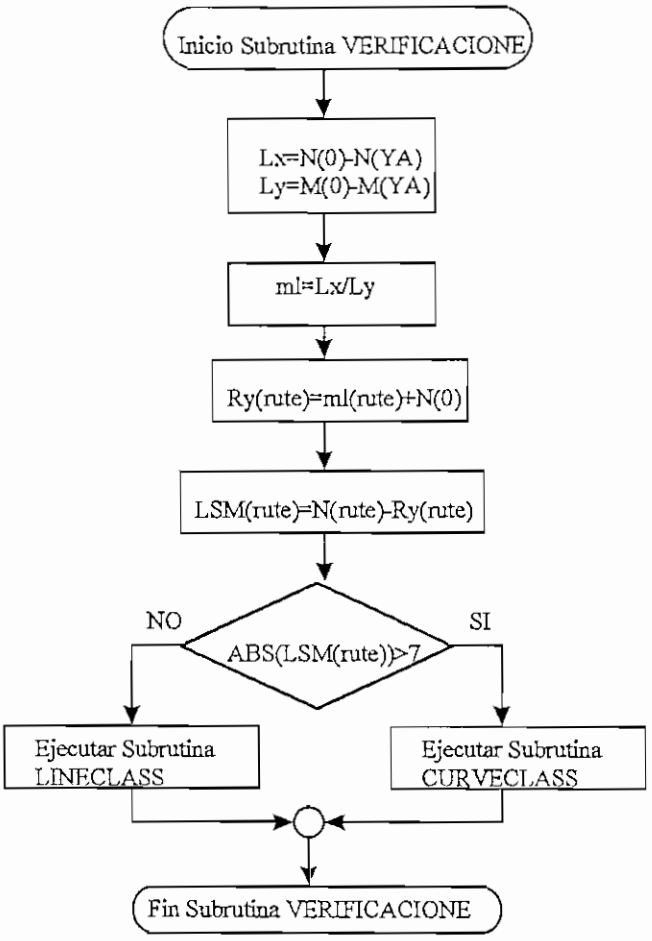

Inicio Subrutina DETECCION
Trazo de línea L1 entre el punto final y de cruce de rama
NUMER=3 y cam=1
Trazo de línea L2 paralela a L1
Fin Subrutina DETECCION

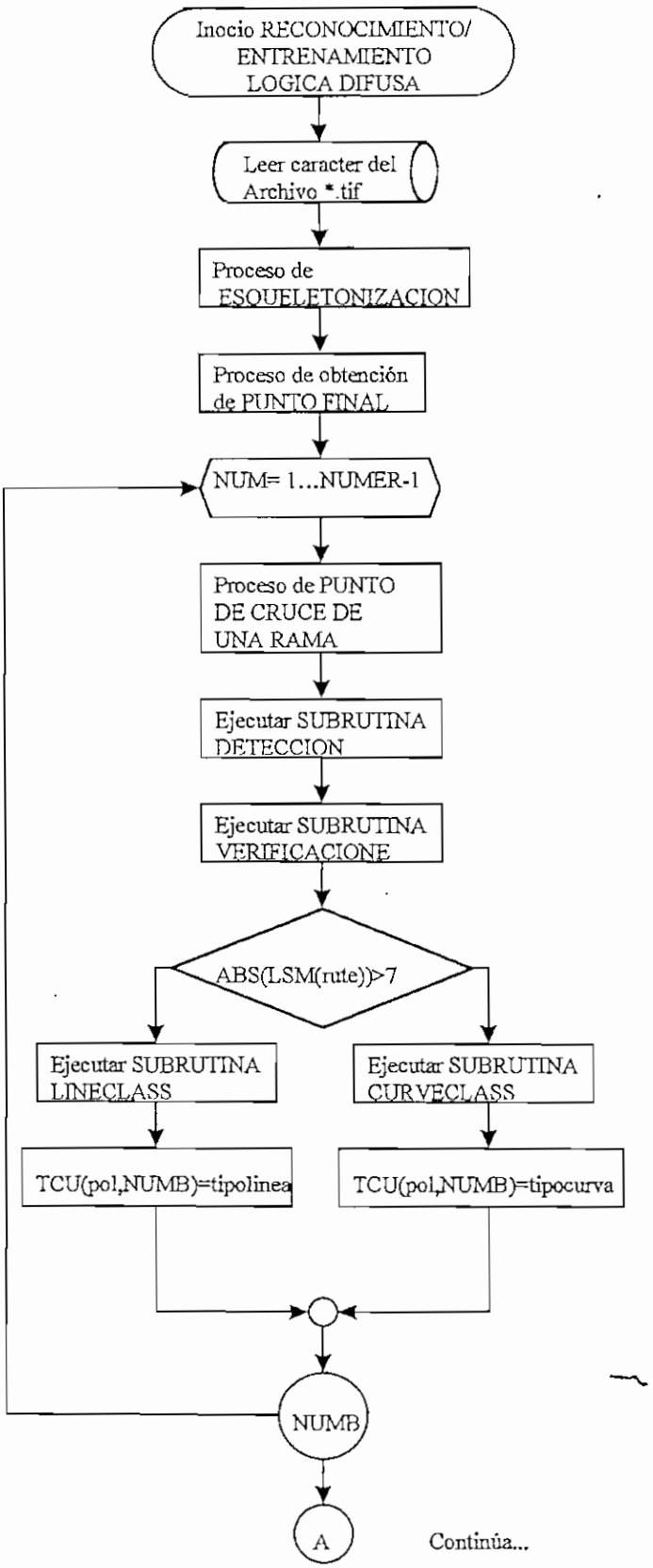

Inicio Subrutina LINECLASS
FH1=ABS(90-TETA)/45
FH2=ABS(270-TETA)/45
FH3=1
MENH= menor entre(FH1,FH2,FH3)
FH=1-MENH
FV1=ABS(TETA)/45
FV2=ABS(180-TETA)/45
FV3=ABS(360-TETA)/45
FV4=1
MENV= menor entre(FV1,FV2,FV3,FV4)
FV=1-MENV
FP1=ABS(45-TETA)/45
FP2=ABS(225-TETA)/45
FP3=1
MENP= menor entre(FP1,FP2,FP3)
FP=1-MENP
A
Continúa...

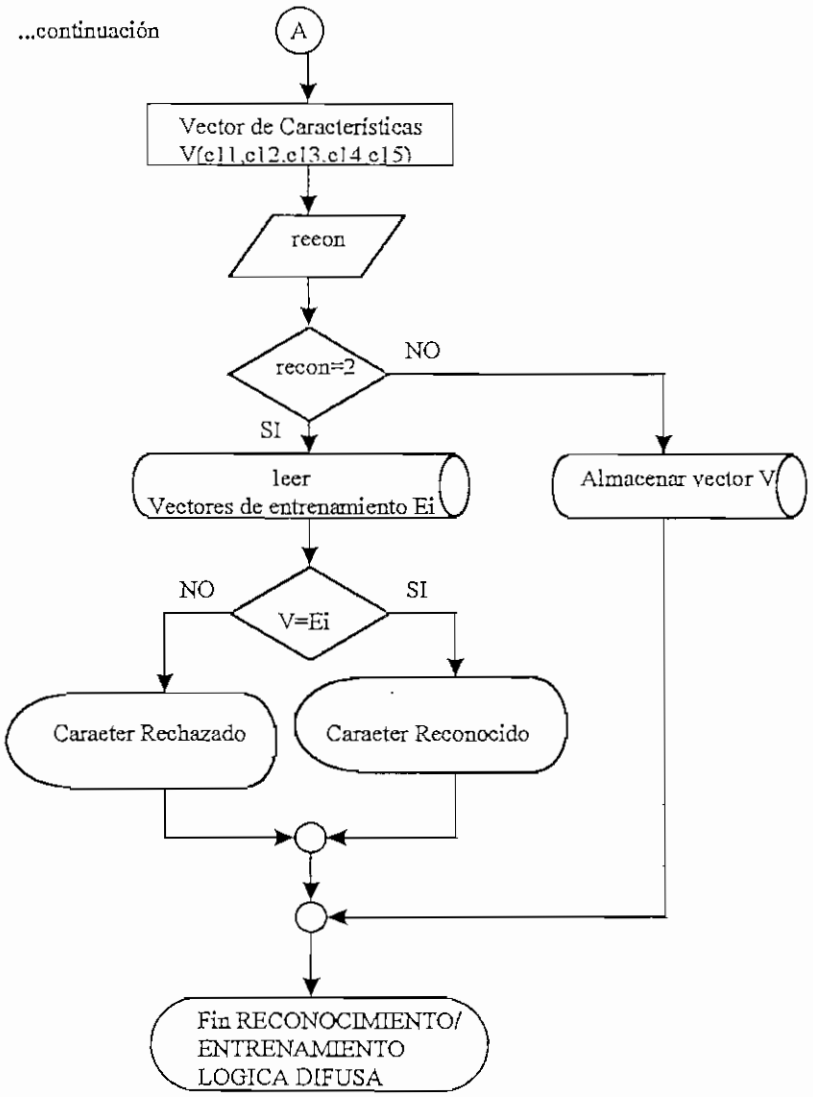

...Continuación
A
PNEG1=ABS(135-TETA0/45
PNEG2=ABS(315-TETA)
PNEG3=1
MENN=menor entre
(PNEG1,PNEG2,PNEG3)
PNEG=1-MENN
NO
FH>(FV,FP,PNEG)
SI
LINEA HORIZONTAL
NO
FV>(FH,FP,PNEG)
SI
LINEA VERTICAL
NO
FP>(FH,FV,PNEG)
SI
LINEA PENDIENTE
POSITIVA
continúa...
B
C


...continuación
B
C
NO
SI
PNEG>(FH,FV,FP)
LINEA CON
PENDIENTE
NEGATIVA
Fin Subrutina LINECLASS

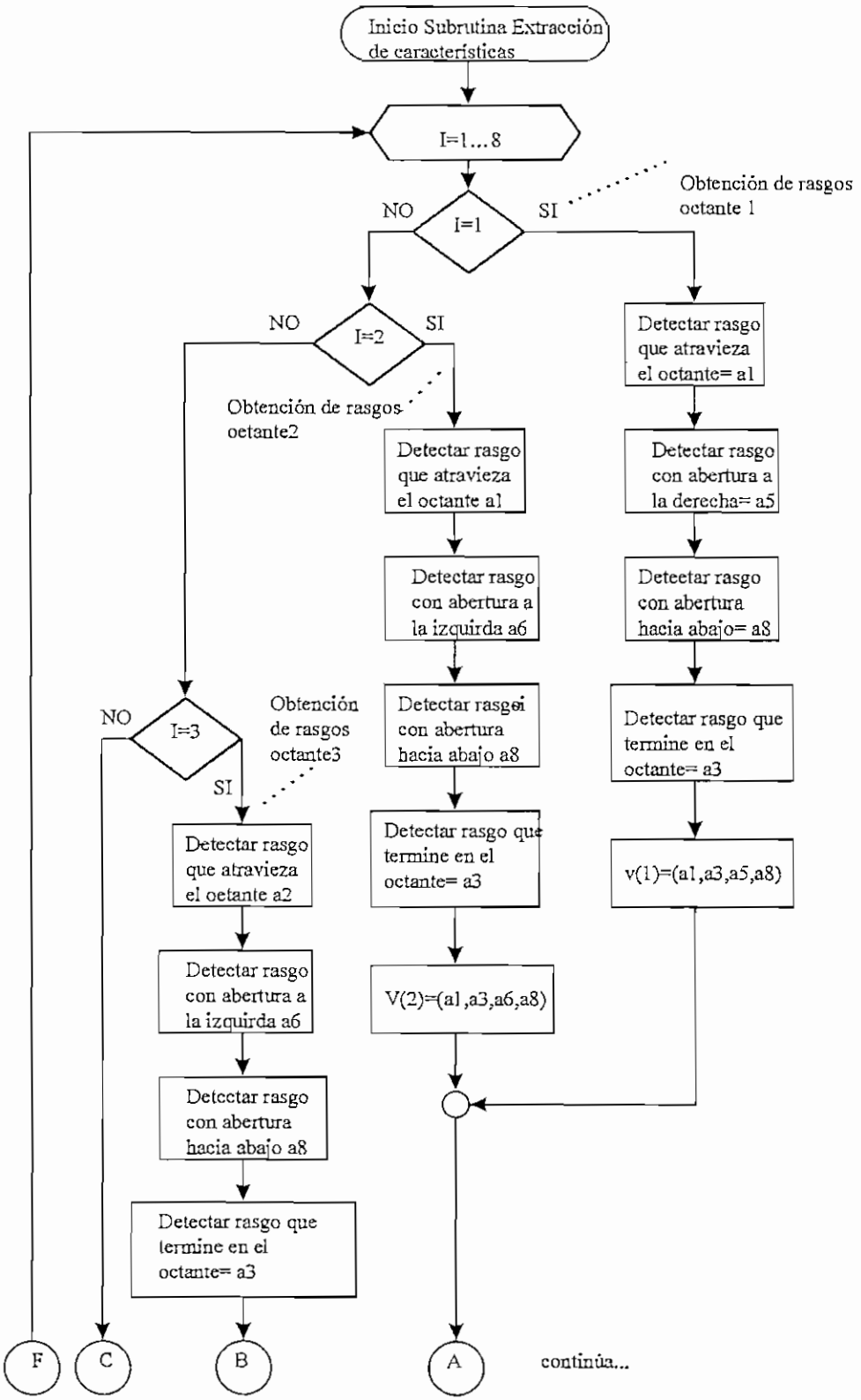

Inicio Subrutina CURVECLASS
HC1=ABS(90-TETA)/90
HC2=ABS(270-TETA)/90
HC3=1
MNH= menor entre(HC1,HH2,HC3)
FHC=1-MNH
VC1=ABS(TETA)/90
VC2=ABS(180-TETA)/90
VC3=ABS(360-TETA)/90
VC4=1
MNV= menor entre(VC1,VC2,VC3,VC4)
FVC=1-MNV
NO
FHC>FVC
SI
LSM(rute)=N(rute) -Ry(rute)
LSM(rute)=N(rute) -Ry(rute)
LSM(rute)>0
LSM(rute)>0
Curva vertical con abertura hacia la derecha
Curva vertical con abertura hacia la izquierda
Curva Horizontal con abertura hacia abajo
Curva Horizontal con abertura hacia arriba
Fin Subrutina CURVECLASS

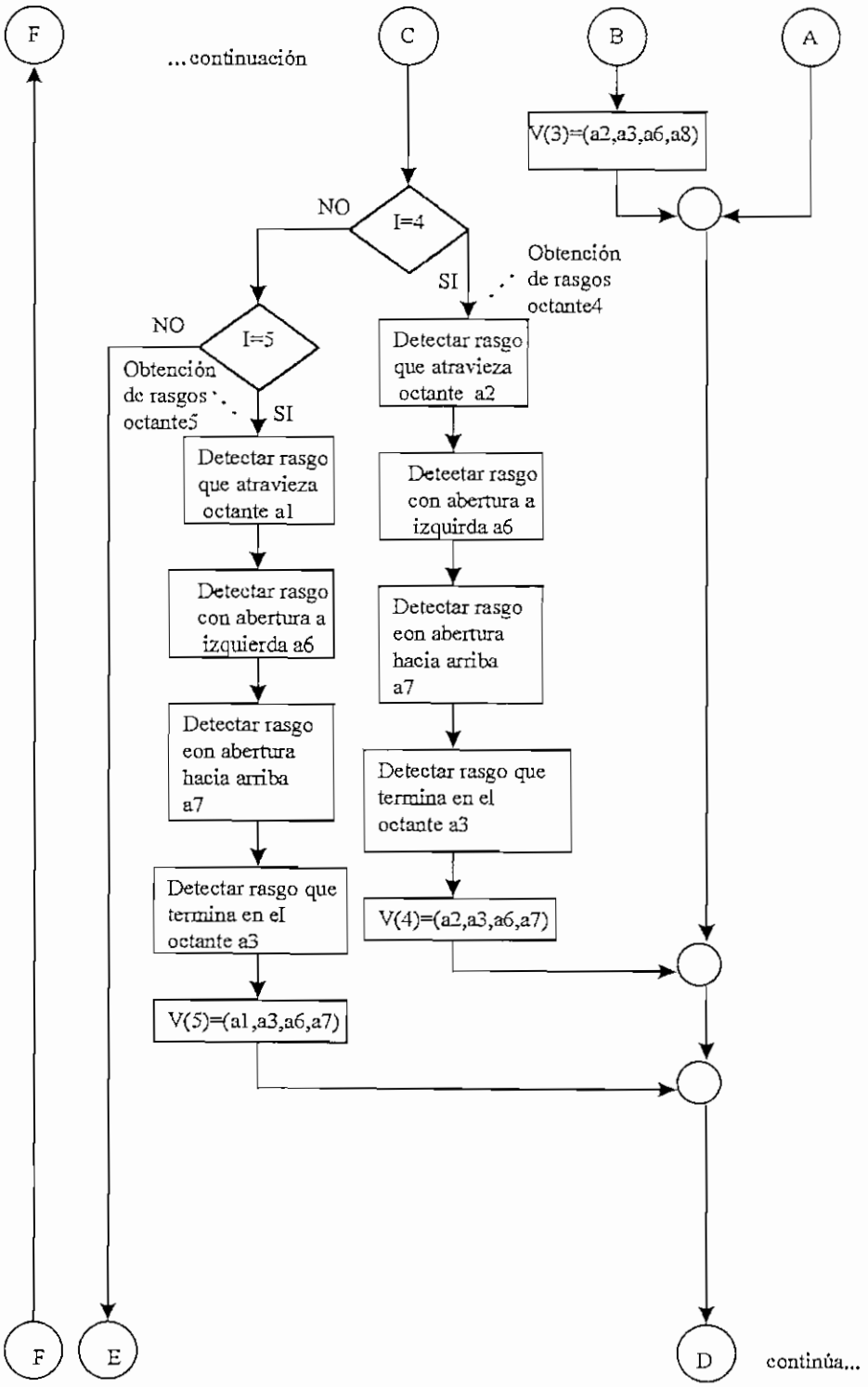

Inicio Subrutina VERIFICACIONE
Lx=N(0)-N(YA)
Ly=M(0)-M(YA)
ml=Lx/Ly
Ry(rute)=ml(rute)+N(0)
LSM(rute)=N(rute)-Ry(rute)
NO
ABS(LSM(rute))>7
SI
Ejecutar Subrutina LINECLASS
Ejecutar Subrutina CURVECLASS
Fin Subrutina VERIFICACIONE

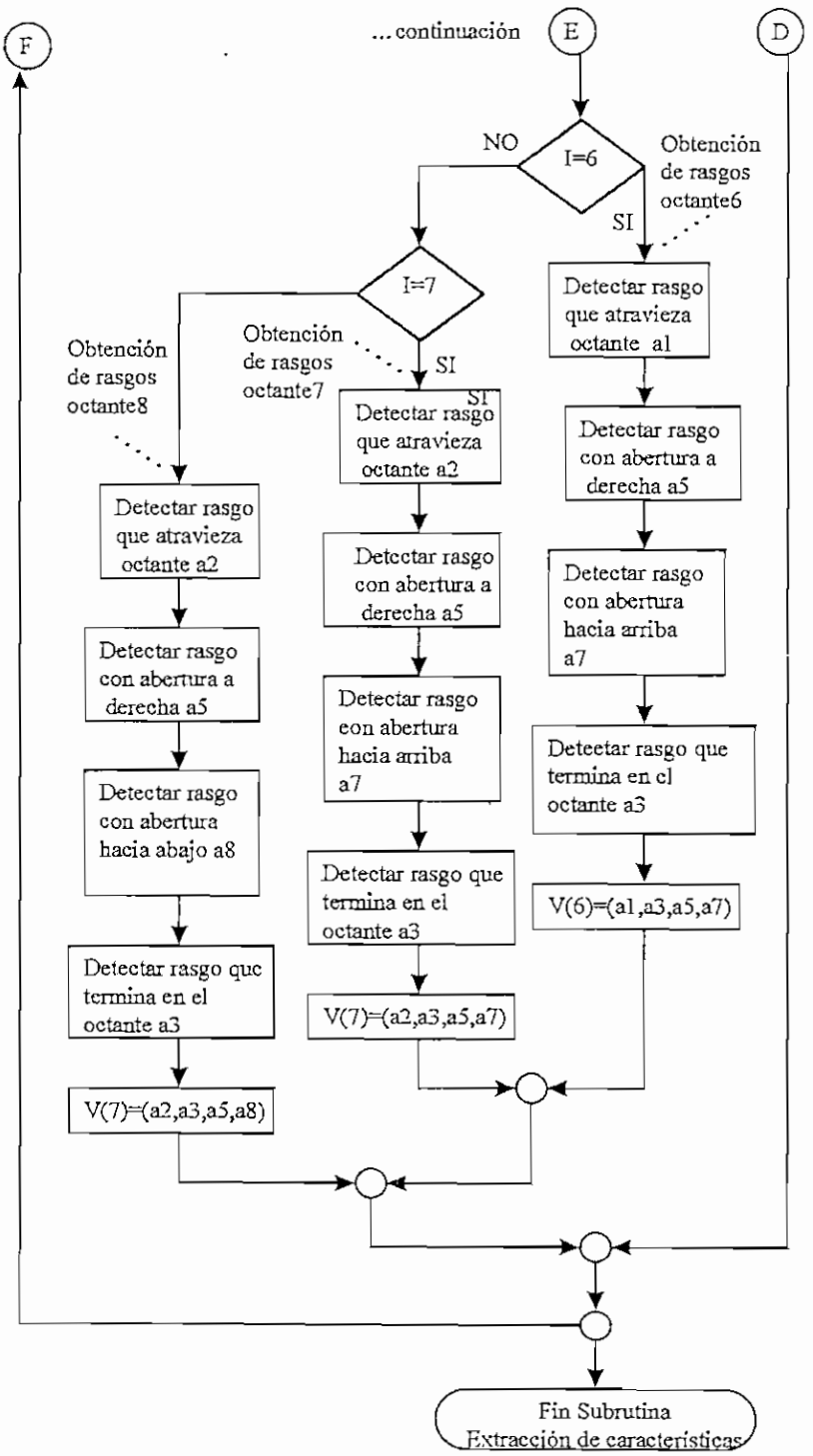

Inocio RECONOCIMIENTO/ ENTRENAMIENTO LOGICA DIFUSA
Leer caracter del Archivo *.tif
Proceso de ESQUELETONIZACION
Proceso de obtención de PUNTO FINAL
NUM= 1...NUMER-1
Proceso de PUNTO DE CRUCE DE UNA RAMA
Ejecutar SUBRUTINA DETECCION
Ejecutar SUBRUTINA VERIFICACIONE
ABS(LSM(rute))>7
Ejecutar SUBRUTINA LINECLASS
Ejecutar SUBRUTINA CURVECLASS
TCU(pol,NUMB)=tipolinea
TCU(pol,NUMB)=tipocurva
NUMB
A
Continúa...

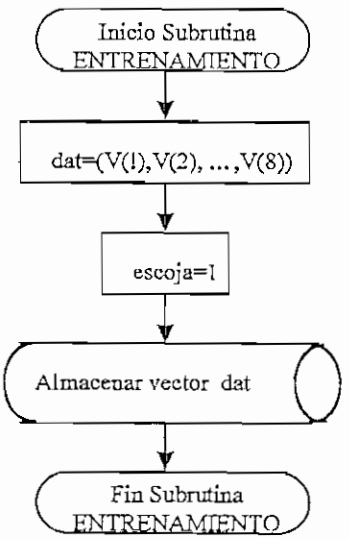

...continuación
A
Vector de Características
V[c11,c12,c13,c14,c15)
recon
recon=2
NO
SI
leer
Vectores de entrenamiento Ei
Almacenar vector V
NO
V=Ei
SI
Caraeter Rechazado
Caraeter Reconocido
Fin RECONOCIMIENTO/
ENTRENAMIENTO
LOGICA DIFUSA

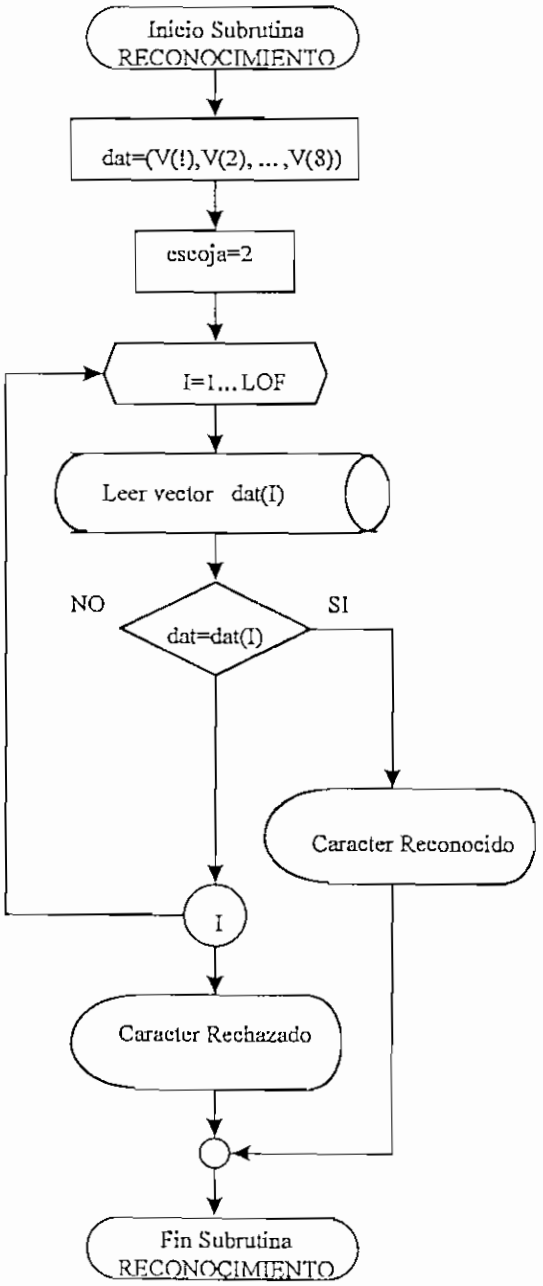
DIAGRAMAS DE FLUJO PARA EL MÉTODO DE EXTRACCIÓN DE RASGOS TOPOLÓGICOS

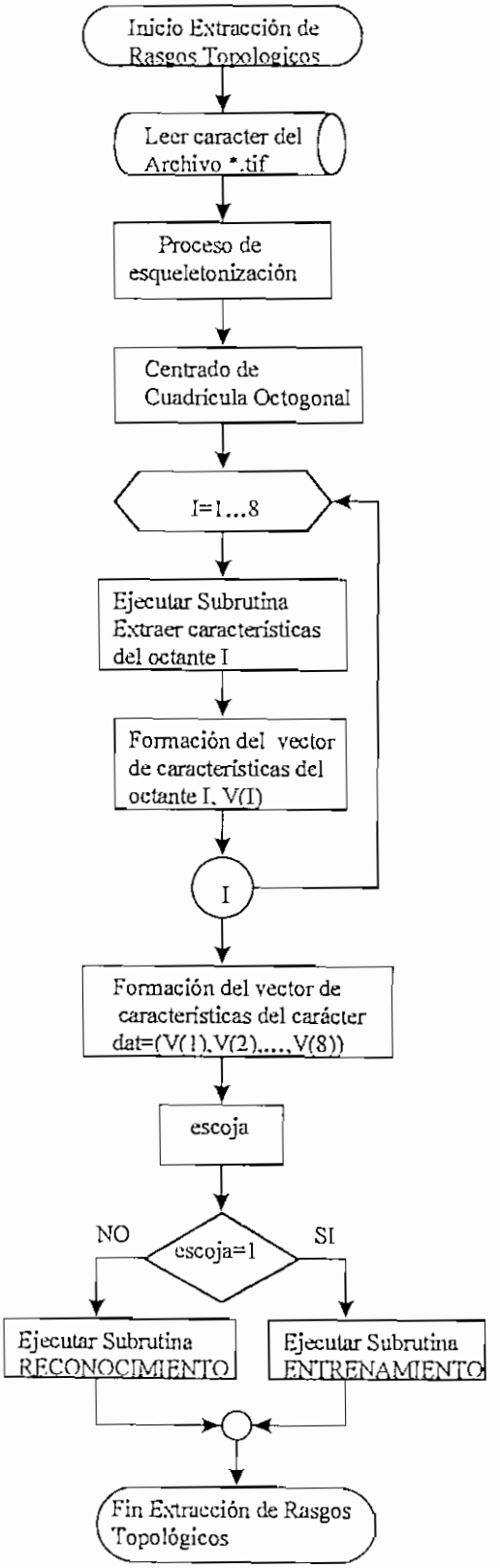



F
...continuación
C
B
A
V(3)=(a2,a3,a6,a8)
NO
I=4
SI
Obtención de rasgos octante4
NO
I=5
Obtención de rasgos octante5
SI
Detectar rasgo que atravieza octante a2
Detectar rasgo que atravieza octante a1
Deteetar rasgo con abertura a izquirda a6
Detectar rasgo con abertura a izquierda a6
Detectar rasgo eon abertura hacia arriba a7
Detectar rasgo eon abertura hacia arriba a7
Detectar rasgo que termina en el octante a3
Detectar rasgo que termina en eI octante a3
V(4)=(a2,a3,a6,a7)
V(5)=(a1,a3,a6,a7)
F
E
D
continúa...


F
... continuación
E
D
I=6
NO
SI
Obtención de rasgos octante6
Detectar rasgo que atravieza octante a1
Detectar rasgo con abertura a derecha a5
Detectar rasgo con abertura hacia arriba a7
Deteetar rasgo que termina en el octante a3
V(6)=(a1,a3,a5,a7)
I=7
Obtención de rasgos octante7
SI
Detectar rasgo que atravieza octante a2
Detectar rasgo con abertura a derecha a5
Detectar rasgo con abertura hacia arriba a7
Detectar rasgo que termina en el octante a3
V(7)=(a2,a3,a5,a7)
Obtención de rasgos octante8
Detectar rasgo que atravieza octante a2
Detectar rasgo con abertura a derecha a5
Detectar rasgo con abertura hacia abajo a8
Detectar rasgo que termina en el octante a3
V(7)=(a2,a3,a5,a8)
Fin Subrutina Extracción de características


Inicio Subrutina
ENTRENAMIENTO
dat=(V(1),V(2), ... ,V(8))
escoja=1
Almacenar vector dat
Fin Subrutina
ENTRENAMIENTO


Inicio Subrutina
RECONOCIMIENTO
dat=(V(!),V(2), ... ,V(8))
escoja=2
I=1... LOF
Leer vector dat(I)
NO
dat=dat(I)
SI
Caracter Reconocido
I
Caracter Rechazado
Fin Subrutina
RECONOCIMIENTO


Inicio Extracción de Rasgos Topologicos
Leer caracter del Archivo *.tif
Proceso de esqueletonización
Centrado de Cuadrícula Octogonal
I=1...8
Ejecutar Subrutina Extraer características del octante I
Formación del vector de características del octante I, V(I)
I
Formación del vector de características del carácter dat=(V(1),V(2),....,V(8))
escoja
NO
escoja=1
SI
Ejecutar Subrutina RECONOCIMIENTO
Ejecutar Subrutina ENTRENAMIENTO
Fin Extracción de Rasgos Topológicos

LOGICA DIFUSA Y EXTRACIóN DE RASGOS TOPOLóGICOS PARA EL RECONOCIMIENTO DE CARACTERES MANUSCRITOS

INTERFAZ CON EL USUARIO EN VISUAL BASIC

```
Form=PRESENTA.FRM
Form=TESIS-01.FRM
Object={F9043C88-F6F2-101A-A3C9-08002B2F49FB}#1.0#0; COMDLG16.OCX
Object={BDC217C8-ED16-11CD-956C-0000C04E4C0A}#1.0#0; TABCTL16.OCX
Object={FAEEE763-117E-101B-8933-08002B2F4F5A}#1.0#0; DBLIST16.OCX
Object={00028C01-0000-0000-0000-000000000046}#1.0#0; DBGRID16.OCX
Reference=*\G{BEF6E001-A874-101A-8BBA-00AA00300CAB}#1.0#0#C:\WINDOWS\SYSTEM\OC25.DLL#Standard OLE
Types
Reference=*\G{00025E01-0000-0000-C000-000000000046}#2.5#0#C:\WINDOWS\SYSTEM\DAO2516.DLL#Microsoft DAO 2.5
Object Library
ProjWinSize=120,120,320,128
ProjWinShow=2
IconForm="Form1"
Name="Proyect1"
HelpContextID="0"
StartMode=0
VersionCompatible="0"
MajorVer=1
MinorVer=0
RevisionVer=0
AutoIncrementVer=0
ServerSupportFiles=0
VersionCompanyName="TITOSA"

VERSION 4.00
Begin VB.Form Form1
  BackColor     = &H00FFFF00&
  BorderStyle   = 3 'Fixed Dialog
  ClientHeight  = 4530
  ClientLeft    = 945
  ClientTop     = 1395
  ClientWidth   = 7275
  ClipControls  = 0 'False
  ControlBox    = 0 'False
  FillColor     = &H00FF00FF&
  BeginProperty Font
    name        = "Times New Roman"
    charset     = 1
    weight      = 700
    size        = 8.25
    underline   = 0 'False
    italic      = 0 'False
    strikethrough = 0 'False
  EndProperty
  ForeColor     = &H000000FF&
  Height        = 4935
  Left          = 885
  LinkTopic     = "Form1"
  MaxButton     = 0 'False
  MinButton     = 0 'False
  ScaleHeight   = 4530
  ScaleWidth    = 7275
  ShowInTaskbar = 0 'False
  Top           = 1050
  Width         = 7395
  WindowState   = 2 'Maximized
  Begin VB.TextBox Text1
    Alignment     = 2 'Center
    BackColor     = &H00FFFFFF&
    BeginProperty Font
      name        = "Times New Roman"
      charset     = 1
      weight      = 700
      size        = 12
      underline   = 0 'False
      italic      = 0 'False
      strikethrough = 0 'False
    EndProperty
    Height      = 975
```

```
  Left        = 1080
  MultiLine   = -1  'True
  TabIndex    = 6
  Text        = "PRESENTA.frx":0000
  Top         = 1920
  Width       = 7455
End
Begin VB.CommandButton Command2
  Caption     = "&Cancelar"
  BeginProperty Font
    name      = "Times New Roman"
    charset   = 1
    weight    = 700
    size      = 12
    underline = 0  'False
    italic    = 0  'False
    strikethrough = 0  'False
  EndProperty
  Height      = 400
  Left        = 5880
  TabIndex    = 1
  Top         = 6120
  Width       = 1500
End
Begin VB.CommandButton Command1
  Caption     = "Aceptar"
  BeginProperty Font
    name      = "Times New Roman"
    charset   = 1
    weight    = 700
    size      = 12
    underline = 0  'False
    italic    = 0  'False
    strikethrough = 0  'False
  EndProperty
  Height      = 400
  Left        = 1560
  TabIndex    = 0
  Top         = 6120
  Width       = 1500
End
Begin VB.Label Label6
  Alignment   = 2  'Center
  BackColor   = &H00FFFF00&
  BackStyle   = 0  'Transparent
  Caption     = "QUITO ECUADOR"
  BeginProperty Font
    name      = "Times New Roman"
    charset   = 1
    weight    = 700
    size      = 12
    underline = 0  'False
    italic    = 0  'False
    strikethrough = 0  'False
  EndProperty
  Height      = 255
  Left        = 1560
  TabIndex    = 8
  Top         = 5160
  Width       = 6615
End
Begin VB.Label Label5
  Alignment   = 2  'Center
  BackColor   = &H00FFFF00&
  BackStyle   = 0  'Transparent
  Caption     = "NOVIEMBRE - 1997"
  BeginProperty Font
    name      = "Times New Roman"
    charset   = 1
    weight    = 700
    size      = 9.75
    underline = 0  'False
    italic    = 0  'False
    strikethrough = 0  'False
  EndProperty
```

```
  Height      = 255
  Left        = 1080
  TabIndex    = 7
  Top         = 4560
  Width       = 7455
End
Begin VB.Label Label4
  Alignment   = 2 'Center
  BackColor   = &H00FFFF00&
  BackStyle   = 0 'Transparent
  Caption     = "DIRIGIDO POR: Dr. GUALBERTO HIDALGO"
  BeginProperty Font
    name      = "Times New Roman"
    charset   = 1
    weight    = 700
    size      = 9.75
    underline = 0 'False
    italic    = 0 'False
    strikethrough = 0 'False
  EndProperty
  Height      = 255
  Left        = 1080
  TabIndex    = 5
  Top         = 3960
  Width       = 7455
End
Begin VB.Label Label3
  Alignment   = 2 'Center
  BackColor   = &H00FFFF00&
  BackStyle   = 0 'Transparent
  Caption     = "REALIZADO POR: TITO POLO TITUAÑA PILLAJO"
  BeginProperty Font
    name      = "Times New Roman"
    charset   = 1
    weight    = 700
    size      = 12
    underline = 0 'False
    italic    = -1 'True
    strikethrough = 0 'False
  EndProperty
  ForeColor   = &H00000000&
  Height      = 375
  Left        = 1080
  TabIndex    = 4
  Top         = 3240
  Width       = 7455
End
Begin VB.Label Label2
  Alignment   = 2 'Center
  BackColor   = &H00FFFF00&
  BackStyle   = 0 'Transparent
  Caption     = "FACULTAD DE INGENIERÍA ELÉCTRICA"
  BeginProperty Font
    name      = "Times New Roman"
    charset   = 1
    weight    = 700
    size      = 15.75
    underline = 0 'False
    italic    = 0 'False
    strikethrough = 0 'False
  EndProperty
  Height      = 495
  Left        = 480
  TabIndex    = 3
  Top         = 1080
  Width       = 8655
End
Begin VB.Label Label1
  Alignment   = 2 'Center
  BackColor   = &H00FFFF00&
  BackStyle   = 0 'Transparent
  Caption     = "ESCUELA POLITÉCNICA NACIONAL"
  BeginProperty Font
    name      = "Times New Roman"
    charset   = 1
```

```
    weight        = 700
    size          = 24
    underline     = 0  'False
    italic        = 0  'False
    strikethrough = 0  'False
  EndProperty
  Height      = 615
  Left        = 360
  TabIndex    = 2
  Top         = 300
  Width       = 8895
 End
End
Attribute VB_Name = "Form1"
Attribute VB_Creatable = False
Attribute VB_Exposed = False
Private Sub Command1_Click()

Unload Form1
Form2.Show

End Sub

Private Sub Command2_Click()
End

End Sub


VERSION 4.00
Begin VB.Form Form2
 BackColor     = &H00FFFFFF&
 Caption       = "RECANUM 1.0 [RECONOCIMIENTO DE CARÁCTERES]"
 ClientHeight  = 2970
 ClientLeft    = 1830
 ClientTop     = 2400
 ClientWidth   = 7380
 BeginProperty Font
   name        = "Times New Roman"
   charset     = 1
   weight      = 700
   size        = 12
   underline   = 0  'False
   italic      = 0  'False
   strikethrough = 0  'False
 EndProperty
 Height        = 3660
 Left          = 1770
 LinkTopic     = "Form2"
 ScaleHeight   = 2970
 ScaleWidth    = 7380
 Top           = 1770
 Width         = 7500
 WindowState   = 2  'Maximized
 Begin VB.TextBox Text2
   Height      = 400
   Left        = 2300
   TabIndex    = 3
   Top         = 1200
   Visible     = 0  'False
   Width       = 4215
 End
 Begin VB.CommandButton Command2
   Caption     = "&Aceptar"
   Height      = 375
   Left        = 6000
   TabIndex    = 2
   Top         = 5880
   Visible     = 0  'False
   Width       = 1500
 End
 Begin VB.CommandButton Command1
   Caption     = "&Salir"
   Height      = 375
   Left        = 3000
```

```
    TabIndex      = 1
    Top           = 5880
    Width         = 1500
  End
  Begin VB.TextBox Text1
    Height        = 400
    Left          = 2300
    TabIndex      = 0
    Top           = 700
    Visible       = 0  'False
    Width         = 4215
  End
  Begin VB.Label Label2
    BackColor     = &H00FFFF00&
    BorderStyle   = 1  'Fixed Single
    Caption       = "Procedimiento"
    Height        = 400
    Left          = 200
    TabIndex      = 5
    Top           = 1200
    Visible       = 0  'False
    Width         = 2000
  End
  Begin VB.Label Label1
    BackColor     = &H00FFFF00&
    BorderStyle   = 1  'Fixed Single
    Caption       = "Método"
    Height        = 400
    Left          = 200
    TabIndex      = 4
    Top           = 700
    Visible       = 0  'False
    Width         = 2000
  End
  Begin VB.Menu Logica
    Caption       = "&Lógica difusa"
    Begin VB.Menu EntreL
      Caption     = "&Entrenamiento"
    End
    Begin VB.Menu ReconL
      Caption     = "&Reconocimiento"
    End
  End
  Begin VB.Menu Rasgos
    Caption       = "&Extración de rasgos topológicos"
    Begin VB.Menu EntreR
      Caption     = "&Entrenamiento"
    End
    Begin VB.Menu ReconR
      Caption     = "&Reconocimiento"
    End
  End
End
Attribute VB_Name = "Form2"
Attribute VB_Creatable = False
Attribute VB_Exposed = False
Dim Archivo, Recon, Control, Paso

Private Sub Command1_Click()

End

End Sub

Private Sub Command2_Click()

Select Case Control
  Case 0
    Select Case Recon
      Case 0
        Shell "C:\RECANUM\LDENTREN.EXE"
      Case 1
        Shell "C:\RECANUM\LDRECONO.EXE"
    End Select
  Case 1
```

```
    Select Case Recon
      Case 0
        Shell "C:\RECANUM\ERTENTRE.EXE"
      Case 1
        Shell "C:\RECANUM\ERTRECON.EXE"
    End Select
End Select

End Sub

Private Sub Entrel_Click()

On Error GoTo Mensaje
Text1.Text = "Lógica Difusa"
Text2.Text = "Entrenamiento"
Text1.Width = Form2.ScaleWidth - 2500
Text2.Width = Form2.ScaleWidth - 2500
Command2.Visible = -1
Text1.Visible = -1
Text2.Visible = -1
Label1.Visible = -1
Label2.Visible = -1
Recon = 0
Control = 0
Exit Sub

Mensaje:

msg = "No se ha especificado ningún archivo"
MsgBox msg, 16

End Sub

Private Sub EntreR_Click()

On Error GoTo Mensaje

Text1.Text = "Extracción de rasgos topológicos"
Text2.Text = "Entrenamiento"
Text1.Width = Form2.ScaleWidth - 2500
Text2.Width = Form2.ScaleWidth - 2500
Command2.Visible = -1
Text1.Visible = -1
Text2.Visible = -1
Label1.Visible = -1
Label2.Visible = -1
Recon = 0
Control = 1

Exit Sub

Mensaje:

msg = "No se ha especificado ningún archivo"
MsgBox msg, 16

End Sub

Private Sub Form_Load()

Paso = 0

End Sub

Private Sub ReconL_Click()

On Error GoTo Mensaje

Text1.Text = "Lógica Difusa"
Text2.Text = "Reconocimiento"
Text1.Width = Form2.ScaleWidth - 2500
Text2.Width = Form2.ScaleWidth - 2500
Command2.Visible = -1
Text1.Visible = -1
Text2.Visible = -1
```

```
Label1.Visible = -1
Label2.Visible = -1
Recon = 1
Control = 0

Exit Sub


Mensaje:

msg = "No se ha especificado ningún archivo"
MsgBox msg, 16

End Sub



Private Sub ReconR_Click()

Text1.Text = "Extracción de rasgos topológicos"
Text2.Text = "Reconocimiento"
Text1.Width = Form2.ScaleWidth - 2500
Text2.Width = Form2.ScaleWidth - 2500
Command2.Visible = -1
Text1.Visible = -1
Text2.Visible = -1
Label1.Visible = -1
Label2.Visible = -1
Recon = 1
Control = 1

End Sub

PROCESOS DE ENTRENAMIENTO Y RECONOCIMIENTO
'METODO DE LA LOGICA DIFUSA PARA EL RECONOCIMIENTO DE CARACTERES MMANUSCRITOS

DIM X(300), Y(300), M(500), N(500), Hx(20), Hy(20), LSM(110), Ry(110)
DIM I(199), dib(199), fila(30), bit(10, 199), count(199), GRX(80), GRY(80)
DIM NUTX(20), NUTY(20), TCU(600, 7)
count = 0: NO = 0: Dif = 0: tip = 0: pol = 0: utr = 0: nota = 0
fu = 0: paso = 0
CLS :
numarc = 0
 DO UNTIL numarc = 7
 IF numarc >= 1 THEN
    INPUT "otro archivo(SI/NO)"; r$
    g$ = "NO"
    s$ = "no"
    IF r$ = g$ OR r$ = s$ THEN
      END
    END IF
 END IF
 numarc = numarc + 1

LOCATE 9, 8: PRINT " DESEA ENTRENAMIENTO PULSE (1)/DESEA RECONOCIMIENTO PULSE (2)"
LOCATE 12, 30: INPUT "PULSE (1)/(2)"; recon
CLS :
   IF recon = 1 AND paso = 0 THEN
     LOCATE 7, 30: PRINT "ENTRENAMIENTO"
     OPEN "C:\archivo1\archdat2" FOR RANDOM AS #3
     tip = LOF(3) / 17
     paso = 1
     CLOSE #3

   ELSE
      IF recon = 1 AND paso = 1 THEN
        LOCATE 7, 30: PRINT "ENTRENAMIENTO"
      END IF
      LOCATE 7, 30: PRINT "RECONOCIMIENTO"
   END IF


LOCATE 15, 25: PRINT "INGRESE EL NOMBRE DEL ARCHIVO A"
LOCATE 18, 28: PRINT "PROCESARSE DEL TIPO.TIF"
LOCATE 20, 25: INPUT "NOMBRE"; a$
```

```
CLS : OPEN "C:\tomas\" + aS + ".tif" FOR BINARY AS #1
repit:
    IF k > 4 AND k < 7 THEN
     fu = 1
    END IF
    IF k >= 7 THEN
     fu = 2
    END IF
    IF Dif >= 7 THEN
     dism = 4
    ELSE
     dism = 0
    END IF
    b = 199 - fu + (count * 199) + NO * 20 + Dif * 28456 - dism * 398
    DIM F AS STRING * 199
    GET #1, b, F: LN = LOF(1)
    FOR I = 1 TO 20
     MS = MIDS(FS, I, 1)
     r = ASC(MS)
     R1 = r / 2
     IF R1 < .5 THEN
      bit(1, I) = 0
      GOTO otro2
     END IF
     bit(1, I) = r - 2 * (FIX(R1))
otro2:
     R2 = FIX(R1) / 2
     IF R2 < .5 THEN
      bit(2, I) = 0
      GOTO otro3
     END IF
     bit(2, I) = FIX(R1) - 2 * (FIX(R2))
otro3:
     R3 = FIX(R2) / 2
     IF R3 < .5 THEN
       bit(3, I) = 0
       GOTO otro4
     END IF
     bit(3, I) = FIX(R2) - 2 * (FIX(R3))
otro4:
     R4 = FIX(R3) / 2
     IF R4 < .5 THEN
      bit(4, I) = 0
      GOTO otro5
     END IF
     bit(4, I) = FIX(R3) - 2 * (FIX(R4))
otro5:
     R5 = FIX(R4) / 2
     IF R5 < .5 THEN
       bit(5, I) = 0
       GOTO otro6
     END IF
     bit(5, I) = FIX(R4) - 2 * (FIX(R5))
otro6:
     R6 = FIX(R5) / 2
     IF R6 < .5 THEN
       bit(6, I) = 0
       GOTO otro7
     END IF
     bit(6, I) = FIX(R5) - 2 * (FIX(R6))
otro7:
     R7 = FIX(R6) / 2
     IF R7 < .5 THEN
      bit(7, I) = 0
      GOTO otro8
     END IF
     bit(7, I) = FIX(R6) - 2 * (FIX(R7))
otro8:
     R8 = FIX(R7) / 2
     IF R8 < .5 THEN
      bit(8, I) = 0
      GOTO binar
     END IF
     bit(8, I) = FIX(R7) - 2 * (FIX(R8))
binar:
```

```
    NEXT I
    SCREEN 12: COLOR 10
    FOR dib = 1 TO 20
    FOR fila = 1 TO 8
     IF count >= 481 THEN
      IF bit(fila, dib) = 0 THEN PSET (8 + (dib - 1) * 8 - fila, count - 480) ELSE PSET (8 + (dib - 1) * 8 - fila, count - 480), 0
      GOTO proximo
     END IF
     IF bit(fila, dib) = 0 THEN PSET (248 + (dib - 1) * 8 - fila, 170 + count) ELSE PSET (248 - (dib - 1) * 8 - fila, 170 + count), 0
proximo:
    NEXT fila
    NEXT dib
    p = LOC(1)
    count = count + 1
    DO UNTIL count = 143
      p = LOF(1)
      GOTO repit
    LOOP

hildich:
'     INPUT "ALGORITMO DE HILDICH"; IS
      COLOR 10
      FOR yi = 1 TO 8
       FOR desh = 250 TO 390
       FOR desv = 180 TO 310
         veci = POINT(desh, desv)
         IF veci = 10 THEN GOTO PROXp ELSE GOTO verifica
verifica:
  veci1 = POINT(desh + 1, desv)
  veci2 = POINT(desh + 1, desv - 1)
  veci3 = POINT(desh, desv - 1)
  veci4 = POINT(desh - 1, desv - 1)
  veci5 = POINT(desh - 1, desv)
  veci6 = POINT(desh - 1, desv + 1)
  veci7 = POINT(desh, desv + 1)
  veci8 = POINT(desh + 1, desv + 1)
condi1:
condicion2:
  IF veci1 = 10 THEN a1 = 1 ELSE a1 = 0
  IF veci2 = 10 THEN a2 = 1 ELSE a2 = 0
  IF veci3 = 10 THEN a3 = 1 ELSE a3 = 0
  IF veci4 = 10 THEN a4 = 1 ELSE a4 = 0
  IF veci5 = 10 THEN a5 = 1 ELSE a5 = 0
  IF veci6 = 10 THEN a6 = 1 ELSE a6 = 0
  IF veci7 = 10 THEN a7 = 1 ELSE a7 = 0
  IF veci8 = 10 THEN a8 = 1 ELSE a8 = 0
  u = a1 + a2 + a3 + a4
  v = (1 - a1) + (1 - a2) + (1 - a3) + (1 - a4) + (1 - a5) + (1 - a6) + (1 - a7) + (1 - a8)
condicion3:
  IF veci1 = 0 THEN b1 = 1 ELSE b1 = 0
  IF veci2 = 0 THEN b2 = 1 ELSE b2 = 0
  IF veci3 = 0 THEN b3 = 1 ELSE b3 = 0
  IF veci4 = 0 THEN b4 = 1 ELSE b4 = 0
  IF veci5 = 0 THEN b5 = 1 ELSE b5 = 0
  IF veci6 = 0 THEN b6 = 1 ELSE b6 = 0
  IF veci7 = 0 THEN b7 = 1 ELSE b7 = 0
  IF veci8 = 0 THEN b8 = 1 ELSE b8 = 0

  W = b1 + b2 + b3 + b4 + b5 + b6 + b7 + b8
condicion4:
 IF veci1 = 10 AND ((veci2 <> 10) OR (veci3 <> 10)) THEN s1 = 1 ELSE s1 = 0
 IF veci3 = 10 AND ((veci4 <> 10) OR (veci5 <> 10)) THEN s2 = 1 ELSE s2 = 0
 IF veci5 = 10 AND ((veci6 <> 10) OR (veci7 <> 10)) THEN s3 = 1 ELSE s3 = 0
 IF veci7 = 10 AND ((veci8 <> 10) OR (veci1 <> 10)) THEN s4 = 1 ELSE s4 = 0
 X = s1 + s2 + s3 + s4
condicion5:
   tu5 = 0
   IF veci5 = 12 THEN
     veci5 = 10
     tu5 = tu5 + 1
   END IF

   IF veci1 = 10 AND ((veci2 <> 10) OR (veci3 <> 10)) THEN si1 = 1 ELSE si1 = 0
   IF veci3 = 10 AND ((veci4 <> 10) OR (veci5 <> 10)) THEN si2 = 1 ELSE si2 = 0
   IF veci5 = 10 AND ((veci6 <> 10) OR (veci7 <> 10)) THEN si3 = 1 ELSE si3 = 0
```

```
IF veci7 = 10 AND ((veci8 <> 10) OR (veci1 <> 10)) THEN si4 = 1 ELSE si4 = 0
xs5 = si1 + si2 + si3 + si4

IF tu5 = 1 THEN
  veci5 = 12
END IF

tu7 = 0
IF veci7 = 12 THEN
  veci7 = 10
  tu7 = tu7 + 1
END IF

IF veci1 = 10 AND ((veci2 = 0 OR veci2 = 12) OR (veci3 = 0 OR veci3 = 12)) THEN se1 = 1 ELSE se1 = 0
IF veci3 = 10 AND ((veci4 = 0 OR veci4 = 12) OR (veci5 = 0 OR veci5 = 12)) THEN se2 = 1 ELSE se2 = 0
IF veci5 = 10 AND ((veci6 = 0 OR veci6 = 12) OR (veci7 = 0 OR veci7 = 12)) THEN se3 = 1 ELSE se3 = 0
IF veci7 = 10 AND ((veci8 = 0 OR veci8 = 12) OR (veci1 = 0 OR veci2 = 12)) THEN se4 = 1 ELSE se4 = 0
xs7 = se1 + se2 + se3 + se4

IF tu7 = 1 THEN
  veci7 = 12
END IF

tu1 = 0
IF veci1 = 12 THEN
  veci1 = 10
  tu1 = tu1 + 1
END IF

IF veci1 = 10 AND ((veci2 <> 10) OR (veci3 <> 10)) THEN sif1 = 1 ELSE sif1 = 0
IF veci3 = 10 AND ((veci4 <> 10) OR (veci5 <> 10)) THEN sif2 = 1 ELSE sif2 = 0
IF veci5 = 10 AND ((veci6 <> 10) OR (veci7 <> 10)) THEN sif3 = 1 ELSE sif3 = 0
IF veci7 = 10 AND ((veci8 <> 10) OR (veci1 <> 10)) THEN sif4 = 1 ELSE sif4 = 0
xs1 = sif1 + sif2 + sif3 + sif4

IF tu1 = 1 THEN
  veci1 = 12
END IF

tu2 = 0
IF veci2 = 12 THEN
  veci2 = 10
  tu2 = tu2 + 1
END IF

IF veci1 = 10 AND ((veci2 = 0 OR veci2 = 12) OR (veci3 = 0 OR veci3 = 12)) THEN ser1 = 1 ELSE ser1 = 0
IF veci3 = 10 AND ((veci4 = 0 OR veci4 = 12) OR (veci5 = 0 OR veci5 = 12)) THEN ser2 = 1 ELSE ser2 = 0
IF veci5 = 10 AND ((veci6 = 0 OR veci6 = 12) OR (veci7 = 0 OR veci7 = 12)) THEN ser3 = 1 ELSE ser3 = 0
IF veci7 = 10 AND ((veci8 = 0 OR veci8 = 12) OR (veci1 = 0 OR veci2 = 12)) THEN ser4 = 1 ELSE ser4 = 0
xs2 = ser1 + ser2 + ser3 + ser4

IF tu2 = 1 THEN
  veci2 = 12
END IF

tu3 = 0
IF veci3 = 12 THEN
  veci3 = 10
  tu3 = tu3 + 1
END IF

IF veci1 = 10 AND ((veci2 <> 10) OR (veci3 <> 10)) THEN sih1 = 1 ELSE sih1 = 0
IF veci3 = 10 AND ((veci4 <> 10) OR (veci5 <> 10)) THEN sih2 = 1 ELSE sih2 = 0
IF veci5 = 10 AND ((veci6 <> 10) OR (veci7 <> 10)) THEN sih3 = 1 ELSE sih3 = 0
IF veci7 = 10 AND ((veci8 <> 10) OR (veci1 <> 10)) THEN sih4 = 1 ELSE sih4 = 0
xs3 = sih1 + sih2 + sih3 + sih4

IF tu3 = 1 THEN
  veci3 = 12
END IF
tu4 = 0
IF veci4 = 12 THEN
  veci4 = 10
  tu4 = tu4 + 1
END IF
```

```
    IF veci1 = 10 AND ((veci2 = 0 OR veci2 = 12) OR (veci3 = 0 OR veci3 = 12)) THEN sch1 = 1 ELSE sch1 = 0
    IF veci3 = 10 AND ((veci4 = 0 OR veci4 = 12) OR (veci5 = 0 OR veci5 = 12)) THEN sch2 = 1 ELSE sch2 = 0
    IF veci5 = 10 AND ((veci6 = 0 OR veci6 = 12) OR (veci7 = 0 OR veci7 = 12)) THEN sch3 = 1 ELSE sch3 = 0
    IF veci7 = 10 AND ((veci8 = 0 OR veci8 = 12) OR (veci1 = 0 OR veci2 = 12)) THEN sch4 = 1 ELSE sch4 = 0
    xs4 = sch1 + sch2 + sch3 + sch4

    IF tu4 = 1 THEN
      veci4 = 12
    END IF
    tu6 = 0
    IF veci6 = 12 THEN
      veci6 = 10
      tu6 = tu6 + 1
    END IF

    IF veci1 = 10 AND ((veci2 <> 10) OR (veci3 <> 10)) THEN sip1 = 1 ELSE sip1 = 0
    IF veci3 = 10 AND ((veci4 <> 10) OR (veci5 <> 10)) THEN sip2 = 1 ELSE sip2 = 0
    IF veci5 = 10 AND ((veci6 <> 10) OR (veci7 <> 10)) THEN sip3 = 1 ELSE sip3 = 0
    IF veci7 = 10 AND ((veci8 <> 10) OR (veci1 <> 10)) THEN sip4 = 1 ELSE sip4 = 0
    xs6 = sip1 + sip2 + sip3 + sip4
    IF tu6 = 1 THEN
      veci6 = 12
    END IF
    tu8 = 0
    IF veci8 = 12 THEN
      veci8 = 10
      tu8 = tu8 + 1
    END IF
    IF veci1 = 10 AND ((veci2 = 0 OR veci2 = 12) OR (veci3 = 0 OR veci3 = 12)) THEN sep1 = 1 ELSE sep1 = 0
    IF veci3 = 10 AND ((veci4 = 0 OR veci4 = 12) OR (veci5 = 0 OR veci5 = 12)) THEN sep2 = 1 ELSE sep2 = 0
    IF veci5 = 10 AND ((veci6 = 0 OR veci6 = 12) OR (veci7 = 0 OR veci7 = 12)) THEN sep3 = 1 ELSE sep3 = 0
    IF veci7 = 10 AND ((veci8 = 0 OR veci8 = 12) OR (veci1 = 0 OR veci2 = 12)) THEN sep4 = 1 ELSE sep4 = 0
    xs8 = sep1 + sep2 + sep3 + sep4
    IF tu8 = 1 THEN
      veci8 = 12
    END IF

PRUEB0:
    IF X = 1 AND (veci8 <> 12 OR xs8 = 1) AND (veci1 <> 12 OR xs1 = 1) AND (veci2 <> 12 OR xs2 = 1) AND (veci3 <> 12 OR
xs3 = 1) AND (veci4 <> 12 OR xs4 = 1) AND (veci6 <> 12 OR xs6 = 1) AND (veci5 <> 12 OR xs5 = 1) AND u >= 1 AND v >= 2
AND W >= 1
 AND (veci7 <> 12 OR xs7 = 1) THEN GOTO cero4 ELSE GOTO PROXp

cero4:
    COLOR 12
    PSET (desh, desv)
PROXp:
    NEXT desv
  NEXT desh


    FOR borrh = 250 TO 390
    FOR borrv = 180 TO 310
      punt = POINT(borrh, borrv)
      IF punt = 12 THEN PSET (borrh, borrv), 10 ELSE GOTO proxi
proxi:
    NEXT borrv
    NEXT borrh

  NEXT yi

POINTF:

    FOR UK = 1 TO 6
      TCU(pol, UK) = 0
    NEXT UK


'   ---------------------------
'   "localizacion de nodos puntos final"
'   ---------------------------

    apunte = 0
    FOR KING = 0 TO 1
```

```
    NUMER = 1
      FOR mv = 180 TO 310
       FOR mh = 260 TO 390
        nodo = 0
        pf = POINT(mh, mv)
        IF pf = 10 THEN GOTO REGRESE ELSE GOTO BUSCAR
BUSCAR:
         PF0 = POINT(mh + 1, mv)
         IF PF0 = 0 THEN
          nodo = nodo + 1
         END IF
        PF1 = POINT(mh + 1, mv - 1)
        IF PF1 = 0 THEN
          nodo = nodo + 1
        END IF
        PF2 = POINT(mh, mv - 1)
        IF PF2 = 0 THEN
          nodo = nodo + 1
        END IF
        PF3 = POINT(mh - 1, mv - 1)
        IF PF3 = 0 THEN
          nodo = nodo + 1
        END IF
        PF4 = POINT(mh - 1, mv)
        IF PF4 = 0 THEN
          nodo = nodo + 1
        END IF
        PF5 = POINT(mh - 1, mv + 1)
        IF PF5 = 0 THEN
          nodo = nodo + 1
        END IF
        PF6 = POINT(mh, mv + 1)
        IF PF6 = 0 THEN
          nodo = nodo + 1
        END IF
        PF7 = POINT(mh + 1, mv + 1)
        IF PF7 = 0 THEN
          nodo = nodo + 1
        END IF
        IF nodo = 1 THEN
          X(NUMER) = mh
          Y(NUMER) = mv
          NUMER = NUMER + 1
        END IF
    ' Borrado de puntos aislados
        IF nodo = 0 THEN
          PSET (mh, mv), 15
        END IF

REGRESE:
       NEXT mh
      NEXT mv

     ' "OBTENCION DE NODOS VECINOS 3 Y 4"
      cigt = 0
      NUMB = 0
  DO UNTIL NUMB = NUMER - 1
  NUMB = NUMB + 1
  camino = 0
  d = POINT(X(NUMB), Y(NUMB))
  D0 = POINT(X(NUMB) + 1, Y(NUMB))
  D1 = POINT(X(NUMB) + 1, Y(NUMB) - 1)
  D2 = POINT(X(NUMB), Y(NUMB) - 1)
  D3 = POINT(X(NUMB) - 1, Y(NUMB) - 1)
  D4 = POINT(X(NUMB) - 1, Y(NUMB))
  D5 = POINT(X(NUMB) - 1, Y(NUMB) + 1)
  D6 = POINT(X(NUMB), Y(NUMB) + 1)
  D7 = POINT(X(NUMB) + 1, Y(NUMB) + 1)

  geo0 = d XOR D0
  IF geo0 = 0 THEN
   camino = camino + 1
   M(0) = X(NUMB) + 1
   N(0) = Y(NUMB)
  END IF
```

```
 geo1 = d XOR D1
 IF geo1 = 0 THEN
  camino = camino + 1
  M(0) = X(NUMB) + 1
  N(0) = Y(NUMB) - 1
 END IF
 geo2 = d XOR D2
 IF geo2 = 0 THEN
 camino = camino + 1
  M(0) = X(NUMB)
  N(0) = Y(NUMB) - 1
 END IF
 geo3 = d XOR D3
 IF geo3 = 0 THEN
  camino = camino + 1
  M(0) = X(NUMB) - 1
  N(0) = Y(NUMB) - 1
 END IF
 geo4 = d XOR D4
 IF geo4 = 0 THEN
  camino = camino + 1
  M(0) = X(NUMB) - 1
  N(0) = Y(NUMB)
 END IF
 geo5 = d XOR D5
 IF geo5 = 0 THEN
  camino = camino + 1
  M(0) = X(NUMB) - 1
  N(0) = Y(NUMB) + 1
 END IF
 geo6 = d XOR D6
 IF geo6 = 0 THEN
  camino = camino + 1
  M(0) = X(NUMB)
  N(0) = Y(NUMB) + 1
 END IF
 geo7 = d XOR D7
 IF geo7 = 0 THEN
  camino = camino + 1
  M(0) = X(NUMB) + 1
  N(0) = Y(NUMB) + 1
 END IF

FOR YA = 0 TO 200
 cam = 0
 g = POINT(M(YA), N(YA))
 g0 = POINT(M(YA) + 1, N(YA))
 g1 = POINT(M(YA) + 1, N(YA) - 1)
 g2 = POINT(M(YA), N(YA) - 1)
 g3 = POINT(M(YA) - 1, N(YA) - 1)
 g4 = POINT(M(YA) - 1, N(YA))
 g5 = POINT(M(YA) - 1, N(YA) + 1)
 g6 = POINT(M(YA), N(YA) + 1)
 g7 = POINT(M(YA) + 1, N(YA) + 1)
 e0 = g XOR g0
  IF e0 = 0 THEN
    cam = cam + 1
    Hx(cam) = M(YA) + 1
    Hy(cam) = N(YA)
  END IF
  e1 = g XOR g1
  IF e1 = 0 THEN
    cam = cam + 1
    Hx(cam) = M(YA) + 1
    Hy(cam) = N(YA) - 1
  END IF
  e2 = g XOR g2
  IF e2 = 0 THEN
    cam = cam + 1
    Hx(cam) = M(YA)
    Hy(cam) = N(YA) - 1
  END IF
  e3 = g XOR g3
  IF e3 = 0 THEN
```

```
    cam = cam + 1
    Hx(cam) = M(YA) - 1
    Hy(cam) = N(YA) - 1
   END IF
   e4 = g XOR g4
   IF e4 = 0 THEN
    cam = cam + 1
    Hx(cam) = M(YA) - 1
    Hy(cam) = N(YA)
   END IF
   e5 = g XOR g5
   IF e5 = 0 THEN
    cam = cam + 1
    Hx(cam) = M(YA) - 1
    Hy(cam) = N(YA) + 1
   END IF
   e6 = g XOR g6
   IF e6 = 0 THEN
    cam = cam + 1
    Hx(cam) = M(YA)
    Hy(cam) = N(YA) + 1
   END IF
   e7 = g XOR g7
   IF e7 = 0 THEN
    cam = cam + 1
    Hx(cam) = M(YA) + 1
    Hy(cam) = N(YA) + 1
   END IF
     IF cam <= 2 THEN GOTO NOMBRAR ELSE GOTO ENCONTRAR
NOMBRAR:
       IF YA = 0 THEN
         IF Hx(1) <> X(NUMB) OR Hy(1) <> Y(NUMB) THEN
          M(YA + 1) = Hx(1)
          N(YA + 1) = Hy(1)
         END IF
       IF Hx(2) <> X(NUMB) OR Hy(2) <> Y(NUMB) THEN
         M(YA + 1) = Hx(2)
         N(YA + 1) = Hy(2)
       END IF
       GOTO REGRESO
     END IF

DIST2:
    IF Hx(1) <> M(YA - 1) OR Hy(1) <> N(YA - 1) THEN
     M(YA + 1) = Hx(1)
     N(YA + 1) = Hy(1)
    END IF
    IF Hx(2) <> M(YA - 1) OR Hy(2) <> N(YA - 1) THEN
     M(YA + 1) = Hx(2)
     N(YA + 1) = Hy(2)
    END IF
    GOTO REGRESO
ENCONTRAR:
'     "BORRADO DE LINEAS MENORES DE d=7"
       IF cam = 3 AND NUMER = 2 AND YA > 4 AND YA <= 7 THEN
          IF (g0 = 0 AND g7 = 0) XOR (g2 = 0 AND g4 = 0 AND g7 = 0) THEN
            PSET (M(YA), N(YA)), 10
          END IF
          PSET (X(NUMB), Y(NUMB)), 10
          RESU = 0
          WT = 18
          DO UNTIL WT = 0
           RESU = RESU + 1
           WT = YA - RESU
           PSET (M(WT), N(WT)), 10
          LOOP
       END IF

       IF YA <= 4 THEN GOTO borrar ELSE GOTO PROCESAR
borrar:
         IF cam = 4 THEN
           IF YA = 0 THEN GOTO BORR ELSE GOTO BORR2
BORR:
         PSET (M(YA), N(YA)), 10
         PSET (X(NUMB), Y(NUMB)), 10
```

```
        EXIT FOR

BORR2:
        PSET (X(NUMB), Y(NUMB)), 10
        FIGU = 0
        WE = 15
        DO UNTIL WE = 0
         FIGU = FIGU + 1
         WE = YA - FIGU
         PSET (M(WE), N(WE)), 10
        LOOP
        EXIT FOR
       END IF
       IF cam = 3 THEN
        IF YA = 0 THEN GOTO BOR ELSE GOTO BOR2
BOR:
        IF (g0 = 0 AND g7 = 0) XOR (g5 = 0 AND g6 = 0) THEN
          PSET (M(YA), N(YA)), 10
        END IF
        PSET (X(NUMB), Y(NUMB)), 10
        EXIT FOR

BOR2:
        IF (g0 = 0 AND g7 = 0) XOR (g5 = 0 AND g6 = 0) XOR (g2 = 0 AND g3 = 0) XOR (g3 = 0 AND g4 = 0) THEN
          PSET (M(YA), N(YA)), 10
        END IF
        PSET (X(NUMB), Y(NUMB)), 10
        RESU = 0
        WT = 18
        DO UNTIL WT = 0
          RESU = RESU + 1
          WT = YA - RESU
          PSET (M(WT), N(WT)), 10
        LOOP
       END IF
       EXIT FOR
PROCESAR:
PARAR:
'      "ESTIMACION DIRECTA DEL NUMERAL 6 y 9"

       ROSIY = N(YA) - N(0)

       IF KING = 1 AND NUMER = 2 THEN
         IF ROSIY > 0 THEN
           TCU(pol, NUMB) = 60
           GOTO PRONUM
         ELSE
           TCU(pol, NUMB) = 90
           GOTO PRONUM
         END IF
       END IF

'      "PROCESO DE DEFINICION DE CARACTERISTICAS"
       IF KING = 1 THEN
         GOSUB VERIFICACIONE
       END IF

       EXIT FOR
REGRESO:

'  borrado de lineas menores de 3 pixels que no tenga nodo

       IF YA = 0 AND cam = 1 AND KING = 0 THEN
         PSET (X(NUMB), Y(NUMB)), 9
         PSET (M(YA), N(YA)), 9
       END IF
       IF cam = 1 AND YA >= 1 AND YA <= 3 AND KING = 0 THEN
         PSET (X(NUMB), Y(NUMB)), 9
         FIGU = 0
         WE = 15
         DO UNTIL WE = 0
          FIGU = FIGU + 1
          WE = YA - FIGU
          PSET (M(WE), N(WE)), 9
         LOOP
```

```
      END IF

'     Definicion de pendiente

      IF cam = 1 AND NUMER <= 5 AND KING = 1 THEN
       IF NUMER = 3 AND apunte = 0 THEN
        GOSUB DETECCION
        GOSUB DIF65
        IF apunte >= 2 XOR apunte = 0 THEN
         IF NUMB = 1 AND M(YA) <> M(0) THEN
          pend = (N(YA) - N(0)) / (M(YA) - M(0))
          IF pend > 0 THEN
           TCU(pol, 3) = 40
          ELSE
           TCU(pol, 3) = 80
          END IF
         END IF
         GOTO CONVERIFICA
        ELSE
         corte = 1
         GOTO LAZONOD
        END IF
       END IF
CONVERIFICA:
       IF corte = 1 THEN
        NUMB = NUMB - 1
       END IF
       GOSUB VERIFICACIONE
       IF corte = 1 THEN
        NUMB = NUMB + 1
        corte = 0
       END IF
       EXIT FOR
      END IF
    NEXT YA
SIGPUNT:
LAZONOD:
      LOOP
   NEXT KING

'      "NUMERO DE NODOS "
     duo = 0
     NUT = 0
     FOR desih = 260 TO 380
     FOR desiv = 190 TO 300
       vecin = POINT(desih, desiv)
       IF vecin = 10 THEN GOTO PEROp ELSE GOTO verify
verify:
       vecin1 = POINT(desih + 1, desiv)
       vecin2 = POINT(desih + 1, desiv - 1)
       vecin3 = POINT(desih, desiv - 1)
       vecin4 = POINT(desih - 1, desiv - 1)
       vecin5 = POINT(desih - 1, desiv)
       vecin6 = POINT(desih - 1, desiv + 1)
       vecin7 = POINT(desih, desiv + 1)
       vecin8 = POINT(desih + 1, desiv + 1)

       IF vecin1 = 10 THEN vecin1 = 0 ELSE vecin1 = 1
       IF vecin2 = 10 THEN vecin2 = 0 ELSE vecin2 = 1
       IF vecin3 = 10 THEN vecin3 = 0 ELSE vecin3 = 1
       IF vecin4 = 10 THEN vecin4 = 0 ELSE vecin4 = 1
       IF vecin5 = 10 THEN vecin5 = 0 ELSE vecin5 = 1
       IF vecin6 = 10 THEN vecin6 = 0 ELSE vecin6 = 1
       IF vecin7 = 10 THEN vecin7 = 0 ELSE vecin7 = 1
       IF vecin8 = 10 THEN vecin8 = 0 ELSE vecin8 = 1
       conteo = vecin1 + vecin2 + vecin3 + vecin4 + vecin5 + vecin6 + vecin7 + vecin8
       IF conteo >= 3 THEN
        IF (vecin3 = 1 AND vecin7 = 1 AND vecin8 = 1) XOR (vecin1 = 1 AND vecin4 = 1 AND vecin5 = 1) THEN
         GOTO PEROp
        END IF
        IF (vecin3 = 1 AND vecin6 = 1 AND vecin7 = 1) XOR (vecin1 = 1 AND vecin5 = 1 AND vecin8 = 1) THEN
         GOTO PEROp
        END IF
        NUT = NUT + 1
        IF NUT <= 5 THEN
```

```
          NUTX(NUT) = desih
          NUTY(NUT) = desiv
         ELSE
          duo = 1
          GOTO ceros
         END IF
        END IF
PEROp:
    NEXT desiv
    NEXT desih

    IF NUT = 0 THEN
     GOTO ceros
    ELSE
     NUDOS = 0
     DO UNTIL NUDOS = NUT
      NUDOS = NUDOS + 1
     LOOP
    END IF
ceros:
'     "DEFINICION DE CEROS Y OCHOS"

    IF NUMER = 1 AND NUT = 0 THEN
      GOTO PRONUM
    END IF
    IF NUMER = 1 AND NUT >= 2 THEN
      IF duo = 1 THEN
        TCU(pol, NUMB + 1) = 20
        GOTO PRONUM
      END IF
      eigt = 1
      GOSUB PROCESO8
      IF ABS(M(YA) - M(0)) <= 4 THEN
        TCU(pol, NUMB + 1) = 20
        GOTO PRONUM
      END IF
      GOSUB VERIFICACIONE
    END IF

PRONUM:

    IF recon = 1 THEN
      utr = utr + 1
      GOSUB INGDATOS
    ELSE
      GOSUB RECONOCIMIENTO
    END IF

PRONUM2:
    IF recon = 2 THEN
     LOCATE 24, 8: INPUT "DESEA OTRO NUMERO PULSE (1)/ DESEA TERMINAR PULSE(2)"; opt
     IF opt = 2 THEN
       END
     END IF
    END IF
INPUT "proximonumer"; HS
pol = pol + 1: count = 0: nota = 0: NO = NO + 1: CLS : k = k + 1: IF k = 10 GOTO fin ELSE GOTO repit
fin:
fu = 0
k = 0
pol = 0
utr = 0: DO UNTIL Dif = 9: Dif = Dif + 1: PRINT "p="; p: PRINT " d = "; Dif: GOTO repit
LOOP
count = 0
NO = 0
Dif = 0
REM CLOSE #3
CLOSE #1
LOOP
TERMINO:
    END

DETECCION:

      PINT = 0
```

```
      LINE (M(YA) + 165, N(YA))-(M(0) + 165, N(0)), 14
      LUC = 120
      DO UNTIL LUC = 0
       LUC = YA - PINT
        PSET (M(LUC) + 165, N(LUC)), 10
       PINT = PINT + 1
      LOOP

      IF NUMER = 3 AND cam = 1 THEN
       IF nota = 0 THEN
        LINE (M(YA) + 167, N(YA) + 5)-(M(0) + 176, N(0) - 9), 12
        nota = nota + 1
       END IF
      END IF

      RETURN

LINECLASS:

       FH1 = ABS(90 - TETA) / 45
       FH2 = ABS(270 - TETA) / 45
       FH3 = 1
       IF FH1 < FH2 THEN
        IF FH1 < FH3 THEN
         MENH = FH1
        ELSE
         MENH = FH3
        END IF
       ELSE
        IF FH2 < FH3 THEN
         MENH = FH2
        ELSE
         MENH = FH3
        END IF
       END IF
       FH = 1 - MENH
       FV1 = ABS(TETA) / 45
       FV2 = ABS(180 - TETA) / 45
       FV3 = ABS(360 - TETA) / 45
       FV4 = 1
       IF FV1 < FV2 THEN
          IF FV1 < FV3 THEN
            IF FV1 < FV4 THEN
              MENV = FV1
            ELSE
              MENV = FV4
            END IF
          ELSE
            IF FV3 < FV4 THEN
              MENV = FV3
            ELSE
              MENV = FV4
            END IF
          END IF
        ELSE
          IF FV2 < FV3 THEN
            IF FV2 < FV4 THEN
             MENV = FV2
            ELSE
             MENV = FV4
            END IF
          ELSE
            IF FV3 < FV4 THEN
              MENV = FV3
            ELSE
              MENV = FV4
            END IF
          END IF
       END IF

       FV = 1 - MENV
       FP1 = ABS(45 - TETA) / 45
       FP2 = ABS(225 - TETA) / 45
       FP3 = 1
       IF FP1 < FP2 THEN
```

```
      IF FP1 < FP3 THEN
        MENP = FP1
      ELSE
        MENP = FP3
      END IF
    ELSE
      IF FP2 < FP3 THEN
        MENP = FP2
      ELSE
        MENP = FP3
      END IF
    END IF
    FP = 1 - MENP

    PNEG1 = ABS(135 - TETA) / 45
    PNEG2 = ABS(315 - TETA) / 45
    PNEG3 = 1
    IF PNEG1 < PNEG2 THEN
      IF PNEG1 < PNEG3 THEN
        MENN = PNEG1
      ELSE
        MENN = PNEG3
      END IF
    ELSE
      IF PNEG2 < PNEG3 THEN
        MENN = PNEG2
      ELSE
        MENN = PNEG3
      END IF
    END IF
    PNEG = 1 - MENN
    IF FH > FV THEN
      IF FH > FP THEN
        IF FH > PNEG THEN
          MAYR = FH
          TCU(pol, NUMB) = 10
        ELSE
          MAYR = PNEG
          TCU(pol, NUMB) = 12
        END IF
      ELSE
        IF FP > PNEG THEN
          MAYR = FP
          TCU(pol, NUMB) = 13
        ELSE
          MAYR = PNEG
          TCU(pol, NUMB) = 12
        END IF
      END IF
    ELSE
     IF FV > FP THEN
       IF FV > PNEG THEN
         MAYR = FV
         TCU(pol, NUMB) = 11
       ELSE
         MAYR = PNEG
         TCU(pol, NUMB) = 12
       END IF
     ELSE
       IF FP > PNEG THEN
         MAYR = FP
         TCU(pol, NUMB) = 13
       ELSE
         MAYR = PNEG
         TCU(pol, NUMB) = 12

       END IF
     END IF
    END IF
   RETURN

  ' "TIPO DE CURVA VERTICAL U HORIZONTAL
  '  PARA ARRIBA O HACIA ABAJO"

CURVECLASS:
```

```
IF eigt = 1 THEN
  TCU(pol, NUMB + 1) = 20
  RETURN
END IF

HC1 = ABS(90 - TETA) / 90
HC2 = ABS(270 - TETA) / 90
HC3 = 1
IF HC1 < HC2 THEN
  IF HC1 < HC3 THEN
    MINH = HC1
  ELSE
    MINH = HC3
  END IF
ELSE
  IF HC2 < HC3 THEN
    MINH = HC2
  ELSE
    MINH = HC3
  END IF
END IF
FHC = 1 - MINH

VC1 = ABS(TETA) / 90
VC2 = ABS(180 - TETA) / 90
VC3 = ABS(360 - TETA) / 90
VC4 = 1
IF VC1 < VC2 THEN
  IF VC1 < VC3 THEN
    IF VC1 < VC4 THEN
      MINV = VC1
    ELSE
      MINV = VC4
    END IF
  ELSE
    IF VC3 < VC4 THEN
      MINV = VC3
    ELSE
      MINV = VC4
    END IF
  END IF
 ELSE
  IF VC2 < VC3 THEN
    IF VC2 < VC4 THEN
      MINV = VC2
    ELSE
      MINV = VC4
    END IF
  ELSE
    IF VC3 < VC4 THEN
      MINV = VC3
    ELSE
      MINV = VC4
    END IF
  END IF
 END IF
 FVC = 1 - MINV
 IF FHC > FVC THEN
   MAXI = FHC

   IF LSM(rute) > 0 THEN
     TCU(pol, NUMB) = 18

   ELSE
     TCU(pol, NUMB) = 17
   END IF
 ELSE
   MAXI = FVC
   IF LSM(rute) > 0 THEN
     TCU(pol, NUMB) = 16

   ELSE
     TCU(pol, NUMB) = 15
   END IF
 END IF
```

```
        RETURN

    ' "verificacion de LSM,TETA"

VERIFICACIONE:

Lx = N(0) - N(YA)
Ly = M(0) - M(YA)

REM ----------
REM "CURVAS HORIZONTALES SOBRE EL EJE Y CON Lx=0"
REM ----------
IF Lx = 0 THEN
  rute = 0
   pon = 0
    CARL = 0
 DO UNTIL rute = ABS(Ly)
  Ry(rute) = N(0)
   DO WHILE M(rute + pon) = M(rute + pon + 1)
    pon = pon + 1
   LOOP
    LSM(rute) = N(rute + pon) - Ry(rute)
  IF ABS(LSM(rute)) > 7 THEN
     CARL = CARL + 1
     'PRINT "es una curva"
     GOTO DETECT
  END IF
 rute = rute + 1
 LOOP
END IF
IF Lx = 0 THEN
GOTO DETECT
END IF

REM ----------
REM "CURVAS VERTICALES SOBRE EL EJE X CON Ly=0"
REM ----------
IF Ly = 0 THEN
 rute = 0
  pon = 0
   CARL = 0
    DO UNTIL rute = ABS(Lx)
     Ry(rute) = M(0)
      DO WHILE N(rute + pon) = N(rute + pon + 1)
       pon = pon + 1
    LOOP
  LSM(rute) = M(rute + pon) - Ry(rute)
 IF ABS(LSM(rute)) > 7 THEN
    CARL = CARL + 1
    'PRINT "es una curva"
     GOTO DETECT
 END IF
  rute = rute + 1
LOOP
END IF
IF Ly = 0 THEN
GOTO DETECT
END IF

REM ----------
REM "CURVAS  APROXIMACION A  LINEAS HORIZONTALES SOBRE EL EJE Y"
REM ----------

REM "ESTABA EN 16 CAMBIO A>=8 luego a A>=6"

IF ABS(Ly / Lx) < 57.3 AND ABS(Ly / Lx) > 1 THEN
IF Lx < 0 AND Ly > 0 XOR Lx > 0 AND Ly > 0 THEN
ml = -(Lx / Ly)
GOTO EVALUA
END IF
ml = Lx / Ly
EVALUA:
rute = 0
pon = 0
CARL = 0
```

```
REM DO UNTIL rute = ABS(Ly)
LSM(rute) = 0
FOR rute = 0 TO ABS(Ly)


Ry(rute) = ml * (rute) + N(0)
DO WHILE M(rute + pon) = M(rute + pon + 1)
pon = pon + 1
LOOP
LSM(rute) = N(rute + pon) - Ry(rute)
IF ABS(LSM(rute)) > 7 THEN
  CARL = CARL + 1
  'PRINT "es una curva"
  GOTO DETECT
END IF
NEXT rute

ELSE
REM "las otras tipos de curva"
IF N(YA) > N(0) AND M(YA) > M(0) XOR (M(YA) < M(0) AND N(YA) > N(0)) THEN
ml = -Ly / Lx
GOTO INITI
END IF
ml = Ly / Lx

INITI:
cis = 0
pon = 0
rute = 0
CARL = 0
LSM(rute) = 0

FOR rute = 0 TO ABS(Lx)
Ry(rute) = ml * (rute) + M(YA)
 IF (YA - rute - pon - 1) < 0 THEN
    LSM(rute) = LSM(rute - 1)
    EXIT FOR
 END IF

DO WHILE N(YA - rute - pon) = N(YA - rute - pon - 1)
   pon = pon + 1
   IF (YA - rute - pon) = 0 THEN
     LSM(rute) = LSM(rute - 1)
     cis = 1
     EXIT DO
   END IF
LOOP
      IF cis = 1 THEN
        EXIT FOR
      END IF
LSM(rute) = M(YA - rute - pon) - Ry(rute)
IF ABS(LSM(rute)) > 7 THEN
   CARL = CARL + 1
   'PRINT "ES UNA CURVA"
   GOTO DETECT
END IF
NEXT rute

END IF

DETECT: REM INPUT "DETECCIN"; X


REM "UTILIZACION DE LAS SERIES DIFUSAS"
REM "PARA VER EL ANGULO DE INCLINACION DE "
REM "LAS LINEAS"
IF Lx = 0 THEN
TETA = 90
END IF
IF Ly = 0 THEN
 TETA = 0
END IF
IF Lx > 0 AND Ly > 0 THEN
   TER = ATN(Ly / Lx)
   TETA = TER * 180 / 3.1416
```

```
   END IF
IF Lx > 0 AND Ly < 0 THEN
  TER = ATN(Ly / Lx)
  TETA = (TER * 180 / 3.1416) + 360
    END IF
IF Lx < 0 AND Ly < 0 THEN
   TER = ATN(Ly / Lx)
   TETA = (TER * 180 / 3.1416) + 180
    END IF
IF Lx < 0 AND Ly > 0 THEN
    TER = ATN(Ly / Lx)
    TETA = (TER * 180 / 3.1416) + 180
     END IF

REM IF ABS(LSM(rute)) > 7 THEN
IF CARL = 1 THEN
 GOSUB CURVECLASS
    ELSE
  GOSUB LINECLASS
END IF

REM GOSUB LINECLASS


GOSUB DETECCION

REM LINE (M(YA), N(YA))-(X(NUMB), Y(NUMB)), 0
RETURN

DIF65:
cruz = 0
FOR iv = 180 TO 310
FOR ih = 415 TO 555
neig = POINT(ih, iv)
nei = 0
IF neig = 12 THEN
 neig1 = POINT(ih + 1, iv)
 neig2 = POINT(ih + 1, iv - 1)
 neig3 = POINT(ih, iv - 1)
 neig4 = POINT(ih - 1, iv - 1)
 neig5 = POINT(ih - 1, iv)
 neig6 = POINT(ih - 1, iv + 1)
 neig7 = POINT(ih, iv + 1)
 neig8 = POINT(ih + 1, iv + 1)
IF neig1 = 10 THEN
  nei = nei + 1
END IF
IF neig2 = 10 THEN
  nei = nei + 1
END IF
IF neig3 = 10 THEN
  nei = nei + 1
END IF
IF neig4 = 10 THEN
  nei = nei + 1
END IF
IF neig5 = 10 THEN
  nei = nei + 1
END IF
IF neig6 = 10 THEN
  nei = nei + 1
END IF
IF neig7 = 10 THEN
  nei = nei + 1
END IF
IF neig8 = 10 THEN
  nei = nei + 1
END IF

 IF nei >= 1 THEN
   cruz = cruz + 1
     GRX(cruz) = ih
     GRY(cruz) = iv
       IF cruz = 1 THEN
         apunte = apunte + 1
```

```
        PSET (ih, iv), 9
       END IF
      IF cruz >= 2 THEN

       IF ABS(GRX(cruz) - GRX(cruz - 1)) > 2 AND ABS(GRY(cruz) - GRY(cruz - 1)) > 2 THEN
        REM INPUT "CRUZ"; LS
        PSET (ih, iv), 9
        apunte = apunte + 1
       END IF
      END IF
 END IF
 END IF
    NEXT ih
    NEXT iv

RETURN

PROCESO8:
colour3 = POINT(NUTX(1) - 1, NUTY(1) - 1)
IF colour3 = 0 THEN
M(0) = NUTX(1) - 1
N(0) = NUTY(1) - 1
GOTO ANALIZAR
END IF
colour2 = POINT(NUTX(1), NUTY(1) - 1)
IF colour2 = 0 THEN
M(0) = NUTX(1)
N(0) = NUTY(1) - 1
GOTO ANALIZAR
END IF
colour1 = POINT(NUTX(1) + 1, NUTY(1) - 1)
IF colour1 = 0 THEN
M(0) = NUTX(1) + 1
N(0) = NUTY(1) - 1
GOTO ANALIZAR
END IF
ANALIZAR:

FOR YA = 0 TO 200
cam = 0
g = POINT(M(YA), N(YA))
g0 = POINT(M(YA) + 1, N(YA))
g1 = POINT(M(YA) + 1, N(YA) - 1)
g2 = POINT(M(YA), N(YA) - 1)
g3 = POINT(M(YA) - 1, N(YA) - 1)
g4 = POINT(M(YA) - 1, N(YA))
g5 = POINT(M(YA) - 1, N(YA) + 1)
g6 = POINT(M(YA), N(YA) + 1)
g7 = POINT(M(YA) + 1, N(YA) + 1)
e0 = g XOR g0
IF e0 = 0 THEN
cam = cam + 1
Hx(cam) = M(YA) + 1
Hy(cam) = N(YA)
END IF
e1 = g XOR g1
IF e1 = 0 THEN
cam = cam + 1
Hx(cam) = M(YA) + 1
Hy(cam) = N(YA) - 1
END IF
e2 = g XOR g2
IF e2 = 0 THEN
cam = cam + 1
Hx(cam) = M(YA)
Hy(cam) = N(YA) - 1
END IF
e3 = g XOR g3
IF e3 = 0 THEN
cam = cam + 1
Hx(cam) = M(YA) - 1
Hy(cam) = N(YA) - 1
END IF
e4 = g XOR g4
IF e4 = 0 THEN
```

```
cam = cam + 1
Hx(cam) = M(YA) - 1
Hy(cam) = N(YA)
END IF
e5 = g XOR g5
IF e5 = 0 THEN
cam = cam + 1
Hx(cam) = M(YA) - 1
Hy(cam) = N(YA) + 1
END IF
e6 = g XOR g6
IF e6 = 0 THEN
cam = cam + 1
Hx(cam) = M(YA)
Hy(cam) = N(YA) + 1
END IF
e7 = g XOR g7
IF e7 = 0 THEN
cam = cam + 1
Hx(cam) = M(YA) + 1
Hy(cam) = N(YA) + 1
END IF
IF cam <= 2 THEN GOTO NOMBRAR2 ELSE GOTO ENCONTRAR2
NOMBRAR2:
IF YA = 0 THEN
IF Hx(1) <> NUTX(1) OR Hy(1) <> NUTY(1) THEN
M(YA + 1) = Hx(1)
N(YA + 1) = Hy(1)
END IF
IF Hx(2) <> NUTX(1) OR Hy(2) <> NUTY(1) THEN
M(YA + 1) = Hx(2)
N(YA + 1) = Hy(2)
END IF
GOTO REGRESO2
END IF

DIST22:
IF Hx(1) <> M(YA - 1) OR Hy(1) <> N(YA - 1) THEN
M(YA + 1) = Hx(1)
N(YA + 1) = Hy(1)
END IF
IF Hx(2) <> M(YA - 1) OR Hy(2) <> N(YA - 1) THEN
M(YA + 1) = Hx(2)
N(YA + 1) = Hy(2)
END IF
GOTO REGRESO2
ENCONTRAR2:
IF YA = 0 THEN
  IF Hy(1) < Hy(2) THEN
    IF Hy(1) < Hy(3) THEN
     M(1) = Hx(1)
     N(1) = Hy(1)
    ELSE
     M(1) = Hx(3)
     N(1) = Hy(3)
    END IF
   ELSE
     IF Hy(2) < Hy(3) THEN
     M(1) = Hx(2)
      N(1) = Hy(2)
     ELSE
     M(1) = Hx(3)
     N(1) = Hy(3)
     END IF
 END IF
GOTO REGRESO2
END IF
EXIT FOR
REGRESO2: NEXT YA
RETURN

' ---------------------
' almacenar los datos en el archivo
' ---------------------
```

```
INGDATOS:
'-------------------------------
' abrir archivo de ingreso de datos
'-------------------------------
 TYPE TIPOREG
 datu AS STRING * 5
 datu2 AS STRING * 5
 datu3 AS STRING * 5
 datu4 AS STRING * 2
 END TYPE
 DIM regist AS TIPOREG

OPEN "C:\archivo1\archdat2" FOR RANDOM AS #3 LEN = LEN(regist)

'  ------------ DATOS PARA EL ALMACENAMIENTO-------
regist.datu = STRS((TCU(pol, 1) * 100 + TCU(pol, 2)))
regist.datu2 = STRS((TCU(pol, 3) * 100 + TCU(pol, 4)))
regist.datu3 = STRS((TCU(pol, 5) * 100 + TCU(pol, 6)))
regist.datu4 = STRS(utr - 1)
PRINT "ALM="; regist.datu
PRINT "alm2="; regist.datu2
PRINT "alm3"; regist.datu3
PRINT "alm4"; regist.datu4
tip = tip + 1
PUT #3, tip, regist
  PRINT LOC(3)
   CLOSE #3
RETURN

'-------------------------------------
'      Extraccion de datos de la memoria prototipo
'------ para el reconocimiento del caracter ---------
RECONOCIMIENTO:
 encon = 0
 TYPE TipoRegistro
    det AS STRING * 5
    det1 AS STRING * 5
    det2 AS STRING * 5
    det3 AS STRING * 2
 END TYPE
 DIM registro AS TipoRegistro

 OPEN "C:\archivo1\archdat2" FOR RANDOM AS #3 LEN = LEN(registro)
 registros = LOF(3) / LEN(registro)
 FOR busc = 1 TO registros
  GET #3, busc, registro

    fa1$ = MIDS(registro.det, 2, 2)
    yol1 = VAL(fa1$)
  IF yol1 = TCU(pol, 1) THEN

    fa2$ = MIDS(registro.det, 4, 2)
    yol2 = VAL(fa2$)
   IF yol2 = TCU(pol, 2) THEN
     fa3$ = MIDS(registro.det1, 2, 2)
     yol3 = VAL(fa3$)
    IF yol3 = TCU(pol, 3) THEN
     fa4$ = MIDS(registro.det1, 4, 2)
     yol4 = VAL(fa4$)
      IF yol4 = TCU(pol, 4) THEN
        fa5$ = MIDS(registro.det2, 2, 2)
        yol5 = VAL(fa5$)
         IF yol5 = TCU(pol, 5) THEN
           fa6$ = MIDS(registro.det2, 4, 2)
           yol6 = VAL(fa6$)
            IF yol6 = TCU(pol, 6) THEN
              PRINT "EL NUMERO ES="; registro.det3
              encon = 1
              EXIT FOR
            END IF
          END IF
       END IF
    END IF
   END IF
  END IF
```

```
sele: NEXT busc
IF encon = 0 THEN
PRINT "NO ENCONTRO"
END IF
CLOSE #3
RETURN


   '_____________________________________
   ' "PROGRAMA DE RECONOCIMIENTO UTLIZANDO EXTRACCION DE
   '  RASGOS TOPOLOGICOS"
   '_____________________________________

DIM X(300), Y(300), M(500), N(500), Hx(20), Hy(20), LSM(60), Ry(60)
DIM I(199), dib(199), fila(30), bit(10, 199), count(199), ph(80), pv(80)
DIM dat(8, 4), pph(80), ppv(80), puntoy(80)

count = 0: No = 0: Dif = 0: oct = 0: nvect = 800: etiq = 0: pol = 0: utr = 0
fu = 0: paso = 0
CLS :
numarc = 0
 DO UNTIL numarc = 7
    LOCATE 9, 8: PRINT " DESEA ENTRENAMIENTO PULSE (1)/DESEA RECONOCIMIENTO PULSE (2)"
    LOCATE 12, 30: INPUT "PULSE (1)/(2)"; escoja
CLS :
  IF escoja = 1 AND paso = 0 THEN
    LOCATE 7, 30: PRINT "ENTRENAMIENTO"
    OPEN "C:\archivo1\archtop2" FOR RANDOM AS #4
    nvect = LOF(4) / 7
    paso = 1
    CLOSE #4

  ELSE
     IF escoja = 1 AND paso = 1 THEN
        LOCATE 7, 30: PRINT "ENTRENAMIENTO"
     END IF
    LOCATE 7, 30: PRINT "RECONOCIMIENTO"
  END IF

 IF numarc >= 1 THEN
   INPUT "otro archivo(SI/NO)"; r$
   g$ = "NO"
   s$ = "no"
   IF r$ = g$ OR r$ = s$ THEN
    END
   END IF
 END IF
  numarc = numarc + 1

     LOCATE 15, 25: PRINT "INGRESE EL NOMBRE DEL ARCHIVO A"
     LOCATE 18, 28: PRINT "PROCESARSE DEL TIPO.TIF"
     LOCATE 20, 25: INPUT "NOMBRE"; a$
CLS :  OPEN "C:\tomas\" + a$ + ".tif" FOR BINARY AS #1
repit:
    ' Procesos de lectura del caracter y esqueletonizaci¢n
    ' el mismo que el efectuado en
    ' el m,todo de l¢gica difusa

CENTRAR:

     FOR CH = 260 TO 370
     FOR CV = 190 TO 304
      TCOL = POINT(CH, CV)
      IF TCOL = 0 THEN
        Plx = CH
        Ply = CV
        EXIT FOR
      ELSE
        GOTO BUSC
      END IF
BUSC:
     NEXT CV
      IF TCOL = 0 THEN
        EXIT FOR
```

```
      END IF
     NEXT CH
HORIZONTAL:
     FOR CV = 190 TO 304
     FOR CH = 260 TO 390
      TCOL = POINT(CH, CV)
      IF TCOL = 0 THEN
       PAx = CH
       PAy = CV
       EXIT FOR
      ELSE
       GOTO BUSC1
      END IF
BUSC1: NEXT CH
      IF TCOL = 0 THEN
       EXIT FOR
       END IF
      NEXT CV
VERTABAJO:
      FOR CH = 390 TO 260 STEP -1
      FOR CV = 190 TO 304
       TCOL = POINT(CH, CV)
       IF TCOL = 0 THEN
        PDx = CH
        PDy = CV
        EXIT FOR
       ELSE
        GOTO BUSC2
       END IF
BUSC2:
       NEXT CV
       IF TCOL = 0 THEN
        EXIT FOR
       END IF
       NEXT CH
HORIZABAJ:

  FOR CV = 304 TO 190 STEP -1
  FOR CH = 260 TO 390
   TCOL = POINT(CH, CV)
  IF TCOL = 0 THEN
   PFx = CH
   PFy = CV
   REM PSET (CH, CV), 14
   EXIT FOR
   ELSE
   GOTO BUSC4
   END IF
BUSC4:
   NEXT CH
   IF TCOL = 0 THEN
    EXIT FOR
   END IF
   NEXT CV
   DHOR = PDx - PIx
   DVER = PFy - PAy
   CENTx = PIx + (FIX(DHOR / 2))
   CENTy = PAy + (FIX(DVER / 2))
   LINE (CENTx - 38, CENTy)-(CENTx + 38, CENTy), 12
   LINE (CENTx, CENTy - 38)-(CENTx, CENTy + 38), 12
   LINE (CENTx + 38, CENTy - 38)-(CENTx - 38, CENTy + 38), 12
   LINE (CENTx + 38, CENTy + 38)-(CENTx - 38, CENTy - 38), 12
' ---------------------------------
' proceso de deteccion de caracteristicas
' ---------------------------------
   ' cuadrante1
   '   "localizacion de nodos puntos final"
    cual = 0
    apunte = 0
    NUMER = 1
    FOR mv = CENTy - 38 TO CENTy - 1
    FOR mh = CENTx - 38 TO CENTx - 1
      IF (mh - CENTx + 38) > (mv - CENTy + 38) THEN
       nodo = 0
       pf = POINT(mh, mv)
```

```
      IF pf = 0 THEN
       GOSUB LPUNTOFINAL
      END IF
     END IF
REGRESE:
    NEXT mh
    NEXT mv

    eigt = 0
    NUMB = 0
    DO UNTIL NUMB = NUMER - 1
     NUMB = NUMB + 1
     GOSUB VECINDAD1
     FOR YA = 0 TO 100
      cam = 0
      GOSUB BUSCACAMINO
      IF cam <= 2 THEN GOTO NOMBRAR ELSE GOTO ENCONTRAR
NOMBRAR:
       IF YA = 0 THEN
        IF cam = 1 THEN
          IF M(YA) = CENTx - 1 THEN
            dat(1, 1) = 4
            PSET (M(YA), N(YA)), 14
            cual = cual + 1
            EXIT FOR
          ELSE
            dat(1, 1) = 3
            PSET (M(YA), N(YA)), 14
            cual = cual + 1
            EXIT FOR
          END IF
        END IF
        GOSUB CAMINO1
        GOTO REGRESO
       END IF

DIST2:
       GOSUB CAMINO2
       IF cam = 1 THEN
        IF M(YA) = CENTx - 1 THEN
          dat(1, 1) = 4
          PSET (M(YA), N(YA)), 14
          cual = cual + 1
          EXIT FOR
        ELSE
          dat(1, 1) = 3
          PSET (M(YA), N(YA)), 14
          cual = cual + 1
          EXIT FOR
        END IF
      END IF
      GOTO REGRESO
ENCONTRAR:
       IF cam = 3 THEN
         dat(1, 1) = 9
         cual = cual + 1
         EXIT FOR
       END IF
REGRESO:
      NEXT YA
     LOOP
      IF cual = 0 THEN
       dat(1, 1) = 0
      END IF
    ' deteccion de rasgo con abertura hacia la derecha
    ' y hacia abajo
    cual = 0
    primp = 0
    cruz = 0
    aber = 0
    aber1 = 0
    FOR hort = CENTx - 38 TO CENTx - 1
    FOR verti = CENTy - 38 TO CENTy - 1
     IF (hort - CENTx + 38) > (verti - CENTy + 38) THEN
       pinta = POINT(hort, verti)
```

```
        IF pinta = 0 THEN
           IF hort <> CENTx - 1 THEN
             GOSUB BUSCADATOS
             GOTO SIGUIENTE
           ELSE
             IF aber1 = 0 THEN
               primp = primp + 1
               ppv(primp) = verti
               aber1 = aber1 + 1
             ELSE
               GOSUB PUNTOEJEY
             END IF

           END IF
        END IF
      END IF
SIGUIENTE:
    NEXT verti
    NEXT hort
     IF aber = 0 AND aber1 = 2 THEN
      dat(1, 2) = 5
      cua1 = cua1 + 1
     ELSE
      IF aber = 2 AND aber1 = 0 THEN
        dat(1, 2) = 8
        cua1 = cua1 + 1
      END IF
     END IF
     IF cua1 = 0 THEN
       dat(1, 2) = 0
     END IF

'    deteccion de rasgo horizontal
     ante2 = 0
     ant1 = 0
     cant1 = 0
     can1 = 0
     can = 0
     FOR hort = CENTx - 38 TO CENTx - 1
       FOR verti = CENTy - 38 TO CENTy - 1
        IF (hort - CENTx + 38) > (verti - CENTy + 38) THEN
           col = POINT(hort, verti)
           IF col = 0 THEN
             IF can = 0 THEN
               GOSUB RASGOSVH
                 IF can = 1 THEN
                   can1 = can1 + 1
                   EXIT FOR
                 END IF
             END IF
             IF hort = CENTx - 1 THEN
               ante2 = ante2 + 1
               puntoy(ante2) = verti
               IF ante2 = 1 THEN
               ant1 = ant1 + 1
             ELSE
                IF ABS(puntoy(ante2) - puntoy(ante2 - 1)) >= 1 THEN
                  ant1 = ant1 + 1
                END IF
             END IF
           END IF
         END IF
        END IF
       NEXT verti
      NEXT hort
      IF can1 = 1 AND ante2 >= 1 AND ant1 = 1 THEN
       dat(1, 3) = 1
       dat(1, 4) = 0
      ELSE
        IF can1 = 1 AND ante2 >= 1 AND ant1 >= 2 THEN
          dat(1, 3) = 1
          dat(1, 4) = 2
        ELSE
          dat(1, 3) = 0
        END IF
```

```
      END IF

      IF escoja = 1 THEN
       ' definicion del vector del cuadrante I
       etiq = etiq + 1
       oct = oct + 1
       GOSUB ALMACENARDATO
      END IF
      ' cuadranteII
      '   "localizacion de nodos puntos final"
      cua2 = 0
      apunte = 0
      NUMER = 1
      FOR mh = CENTx + 38 TO CENTx + 1 STEP -1
      FOR mv = CENTy - 38 TO CENTy - 1
       IF (mh - CENTx + 38) + (mv - CENTy + 38) < 76 THEN
        nodo = 0
        pf = POINT(mh, mv)
         IF pf = 0 THEN
          GOSUB LPUNTOFINAL
         END IF
       END IF
REGRESE2:
      NEXT mv
      NEXT mh
      eigt = 0
      NUMB = 0
      DO UNTIL NUMB = NUMER - 1
       NUMB = NUMB + 1
       GOSUB VECINDAD1
       FOR YA = 0 TO 100
       cam = 0
       GOSUB BUSCACAMINO
       IF cam <= 2 THEN GOTO NOMBRAR2 ELSE GOTO ENCONTRAR2
NOMBRAR2:
       IF YA = 0 THEN
        IF cam = 1 THEN
         IF M(YA) = CENTx + 1 THEN
          dat(2, 1) = 3
          PSET (M(YA), N(YA)), 14
          cua2 = cua2 + 1
          EXIT FOR
         ELSE
          dat(2, 1) = 4
          PSET (M(YA), N(YA)), 14
          cua2 = cua2 + 1
          EXIT FOR
         END IF

        END IF
        GOSUB CAMINO1
        GOTO REGRESO2
       END IF

DIST22:
       GOSUB CAMINO2
       IF cam = 1 THEN
        IF M(YA) = CENTx + 1 THEN
         dat(2, 1) = 3
         PSET (M(YA), N(YA)), 14
         cua2 = cua2 + 1
         EXIT FOR
        ELSE
         dat(2, 1) = 4
         PSET (M(YA), N(YA)), 14
         cua2 = cua2 + 1
         EXIT FOR
        END IF
       END IF
       GOTO REGRESO2
ENCONTRAR2:
       IF cam = 3 THEN
        dat(2, 1) = 9
        cua2 = cua2 + 1
        EXIT FOR
```

```
      END IF
REGRESO2:
      NEXT YA
      LOOP
      IF cua2 = 0 THEN
       dat(2, 1) = 0
      END IF

      ' Deteccion de rasgo con abertura hacia la izquierda
      ' y hacia abajo

      cua2 = 0
      primp = 0
      cruz = 0
      aber = 0
      aber1 = 0
      FOR hort = CENTx + 38 TO CENTx + 1 STEP -1
      FOR verti = CENTy - 38 TO CENTy - 1
        IF (hort - CENTx + 38) + (verti - CENTy + 38) < 76 THEN
         pinta = POINT(hort, verti)
         IF pinta = 0 THEN
           IF hort <> CENTx + 1 THEN
             GOSUB BUSCADATOS
             GOTO SIGUIENTE2
           ELSE
             IF aber1 = 0 THEN
               primp = primp + 1
               ppv(primp) = verti
               aber1 = aber1 + 1
             ELSE
               GOSUB PUNTOEJEY
             END IF
           END IF
         END IF
        END IF
SIGUIENTE2: NEXT verti
        NEXT hort
        IF aber = 0 AND aber1 = 2 THEN
         dat(2, 2) = 6
         cua2 = cua2 + 1
        ELSE
         IF aber = 2 AND aber1 = 0 THEN
          dat(2, 2) = 8
          cua2 = cua2 + 1
         END IF
      END IF
        IF cua2 = 0 THEN
         dat(2, 2) = 0
        END IF

        ' localizacion de rasgo horizontal

        ante2 = 0
        ant1 = 0
        cua2 = 0
        can = 0
        can2 = 0
        FOR hort = CENTx + 38 TO CENTx + 1 STEP -1
        FOR verti = CENTy - 38 TO CENTy - 1
        IF (hort - CENTx + 38) + (verti - CENTy + 38) < 76 THEN
         cal = POINT(hort, verti)
         IF cal = 0 THEN
          IF can = 0 THEN
            GOSUB RASGOSVH
            IF can = 1 THEN
             can2 = can2 + 1
             EXIT FOR
            END IF
          END IF
          IF hort = CENTx + 1 THEN
            ante2 = ante2 + 1
            puntoy(ante2) = verti
               IF ante2 = 1 THEN
                ant1 = ant1 + 1
               ELSE
```

```
            IF ABS(puntoy(ante2) - puntoy(ante2 - 1)) >= 1 THEN
              ant1 = ant1 + 1
            END IF
           END IF
         END IF
        END IF
       END IF
       NEXT verti
       NEXT hort
       IF can2 = 1 AND ante2 >= 1 AND ant1 = 1 THEN
        dat(2, 3) = 1
        dat(2, 4) = 0
       ELSE
        IF can2 = 1 AND ante2 >= 1 AND ant1 >= 2 THEN
         dat(2, 3) = 1
         dat(2, 4) = 2
        ELSE
         dat(2, 3) = 0
        END IF
       END IF
       IF escoja = 1 THEN
        oct = oct + 1
        GOSUB ALMACENARDATO
       END IF

       ' cuadrante III

       '  "localizacion de nodos puntos final"
       cua3 = 0
       apunte = 0
       NUMER = 1
       FOR mv = CENTy - 38 TO CENTy - 1
       FOR mh = CENTx + 38 TO CENTx + 1 STEP -1
       IF (mv - CENTy + 38) + (mh - CENTx + 38) >= 76 THEN
         nodo = 0
         pf = POINT(mh, mv)
         IF pf = 0 THEN
            GOSUB LPUNTOFINAL
         END IF
       END IF
REGRESE3:  NEXT mh
       NEXT mv

       eigt = 0
       NUMB = 0
       DO UNTIL NUMB = NUMER - 1
        NUMB = NUMB + 1
        GOSUB VECINDAD1
        FOR YA = 0 TO 100
         cam = 0
         GOSUB BUSCACAMINO
         IF cam <= 2 THEN GOTO NOMBRAR3 ELSE GOTO ENCONTRAR3

NOMBRAR3:
          IF YA = 0 THEN
           IF cam = 1 THEN
            IF N(YA) = CENTy - 1 THEN
              dat(3, 1) = 4
              PSET (M(YA), N(YA)), 14
              cua3 = cua3 + 1
              EXIT FOR
            ELSE
              dat(3, 1) = 3
              PSET (M(YA), N(YA)), 14
              cua3 = cua3 + 1
              EXIT FOR
            END IF
           END IF
           GOSUB CAMINO1
           GOTO REGRESO3
          END IF

DIST23:
       GOSUB CAMINO2
       IF cam = 1 THEN
```

```
        IF N(YA) = CENTy - 1 THEN
          dat(3, 1) = 4
          PSET (M(YA), N(YA)), 14
          cua3 = cua3 + 1
          EXIT FOR
        ELSE
          dat(3, 1) = 3
          PSET (M(YA), N(YA)), 14
          cua3 = cua3 + 1
          EXIT FOR
        END IF
       END IF
       GOTO REGRESO3
ENCONTRAR3:
       IF cam = 3 THEN
        dat(3, 1) = 9
        cua3 = cua3 + 1
        EXIT FOR
       END IF
REGRESO3:
      NEXT YA
     LOOP
     IF cua3 = 0 THEN
      dat(3, 1) = 0
     END IF

     ' Deteccion de rasgo con abertura hacia la izquierda
     ' y hacia abajo

     cua3 = 0
     primp = 0
     cruz = 0
     aber = 0
     aber1 = 0
     FOR verti = CENTy - 38 TO CENTy - 1
     FOR hort = CENTx + 38 TO CENTx + 1 STEP -1
     IF (verti - CENTy + 38) + (hort - CENTx + 38) >= 76 THEN
       pinta = POINT(hort, verti)
       IF pinta = 0 THEN
         IF verti <> CENTy - 1 THEN
           GOSUB BUSCADATOS
           GOTO SIGUIENTE3
         ELSE
           IF aber1 = 0 THEN
             primp = primp + 1
             pph(primp) = hort
             aber1 = aber1 + 1
           ELSE
             GOSUB PUNTOEJEX
           END IF
         END IF
       END IF
     END IF
SIGUIENTE3:
     NEXT hort
     NEXT verti
     IF aber = 0 AND aber1 = 2 THEN
      dat(3, 2) = 8
      cua3 = cua3 + 1
     ELSE
      IF aber = 2 AND aber1 = 0 THEN
       dat(3, 2) = 6
       cua3 = cua3 + 1
      END IF
     END IF

     IF cua3 = 0 THEN
      dat(3, 2) = 0
     END IF

     '    Localizacion de rasgo vertical(III)

     cua3 = 0
     aber1 = 0
     primp = 0
```

```
      anb = 0
      can3 = 0
      can = 0
      FOR verti = CENTy - 38 TO CENTy - 1
      FOR hort = CENTx + 38 TO CENTx + 1 STEP -1
       IF (verti - CENTy + 38) + (hort - CENTx + 38) >= 76 THEN
        neibo = POINT(hort, verti)
        IF neibo = 0 THEN
         IF can = 0 THEN
          GOSUB RASGOSVH
          IF can = 1 THEN
           can3 = can3 + 1
           EXIT FOR
          END IF
         END IF
            IF verti = CENTy - 1 THEN
                IF aber1 = 0 THEN
                  primp = primp + 1
                  pph(primp) = hort
                  aber1 = aber1 + 1
                ELSE
                  GOSUB PUNTOEJEX
                END IF
             END IF
        END IF
       END IF
      NEXT hort
      NEXT verti
      IF can3 = 1 AND aber1 = 1 THEN
       dat(3, 3) = 2
      ELSE
       dat(3, 3) = 0
      END IF
      IF escoja = 1 THEN
       oct = oct + 1
       ' definicion del vector del cuadrante III
       GOSUB ALMACENARDATO
      END IF

      '              cuadrante IV

      '  "localizacion de nodos puntos final"

      cua4 = 0
      apunte = 0
      NUMER = 1
      FOR mv = CENTy + 38 TO CENTy + 1 STEP -1
      FOR mh = CENTx + 38 TO CENTx + 1 STEP -1
      IF (mv - CENTy + 38) < (mh - CENTx + 38) THEN
       nodo = 0
       pf = POINT(mh, mv)
        IF pf = 0 THEN
         GOSUB LPUNTOFINAL
        END IF
      END IF

REGRESE4:
      NEXT mh
      NEXT mv
      cig = 0
      NUMB = 0
      DO UNTIL NUMB = NUMER - 1
       NUMB = NUMB + 1
       GOSUB VECINDAD1
       FOR YA = 0 TO 100
       cam = 0
       GOSUB BUSCACAMINO
       IF cam <= 2 THEN GOTO NOMBRAR4 ELSE GOTO ENCONTRAR4
NOMBRAR4:
       IF YA = 0 THEN
        IF cam = 1 THEN
         IF N(YA) = CENTy + 1 THEN
          dat(4, 1) = 3
          PSET (M(YA), N(YA)), 14
          cua = cua + 1
```

```
            EXIT FOR
           ELSE
            dat(4, 1) = 4
            PSET (M(YA), N(YA)), 14
            cua4 = cua4 + 1
            EXIT FOR
           END IF
          END IF
          GOSUB CAMINO1
          GOTO REGRESO4
         END IF

DIST24:
         GOSUB CAMINO2
         IF cam = 1 THEN
          IF N(YA) = CENTy + 1 THEN
            dat(4, 1) = 3
            PSET (M(YA), N(YA)), 14
            cua4 = cua4 + 1
            EXIT FOR
          ELSE
            dat(4, 1) = 4
            PSET (M(YA), N(YA)), 14
            cua4 = cua4 + 1
            EXIT FOR
          END IF
         END IF
         GOTO REGRESO4
ENCONTRAR4:
         IF cam = 3 THEN
          dat(4, 1) = 9
          cua4 = cua4 + 1
          EXIT FOR
         END IF
REGRESO4:
         NEXT YA
        LOOP
        IF cua4 = 0 THEN
         dat(4, 1) = 0
        END IF

        ' Deteccion de rasgo con abertura hacia la izquierda
        ' y hacia arriba

       cua4 = 0
       primp = 0
       cruz = 0
       aber = 0
       aber1 = 0
       FOR verti = CENTy + 38 TO CENTy + 1 STEP -1
       FOR hort = CENTx + 38 TO CENTx + 1 STEP -1
       IF (verti - CENTy + 38) + (hort - CENTx + 38) >= 76 THEN
         pinta = POINT(hort, verti)
         IF pinta = 0 THEN
           IF verti <> CENTy + 1 THEN
              GOSUB BUSCADATOS
              GOTO SIGUIENTE4
            ELSE
              IF aber1 = 0 THEN
                primp = primp + 1
                pph(primp) = hort
                aber1 = aber1 + 1
              ELSE
                GOSUB PUNTOEJEX
              END IF
            END IF
           END IF
       END IF
SIGUIENTE4:
       NEXT hort
       NEXT verti
       IF aber = 0 AND aber1 = 2 THEN
         dat(4, 2) = 7
         cua4 = cua4 + 1
       ELSE
```

```
    IF (aber = 2 XOR aber = 3) AND aber1 = 0 THEN
     dat(4, 2) = 6
     cua4 = cua4 + 1
    END IF
  END IF
  IF cua4 = 0 THEN
   dat(4, 2) = 0
  END IF


' localizacion de rasgo vertical(IV)

primp = 0
aber1 = 0
alf1 = 0
can4 = 0
can = 0
FOR verti = CENTy + 38 TO CENTy + 1 STEP -1
FOR hort = CENTx + 38 TO CENTx + 1 STEP -1
  IF (verti - CENTy + 38) + (hort - CENTx + 38) >= 76 THEN
     neibor = POINT(hort, verti)
     IF neibor = 0 THEN
       IF can = 0 THEN
         GOSUB RASGOSVH
         IF can = 1 THEN
           can4 = can4 + 1
           EXIT FOR
         END IF
       END IF
           IF verti = CENTy + 1 THEN
             IF alfa1 = 0 THEN
               aber1 = aber1 + 1
               primp = primp + 1
               pph(primp) = hort
               alfa1 = alfa1 + 1
             ELSE
               GOSUB PUNTOEJEX
             END IF
           END IF
    END IF
  END IF
  NEXT hort
  NEXT verti
  IF can4 = 1 AND aber1 = 1 THEN
   dat(4, 3) = 2
  ELSE
   IF can4 = 1 AND aber1 = 2 THEN
    dat(4, 3) = 2
    dat(4, 4) = 9
   ELSE
    dat(4, 3) = 0
   END IF
  END IF
  IF escoja = 1 THEN
   oct = oct + 1
   ' definicion del vector del cuadrante IV
   GOSUB ALMACENARDATO
  END IF

' cuadrante V

'   "localizacion de nodos puntos final(V)"

cua5 = 0
apunte = 0
NUMER = 1
FOR mh = CENTx + 38 TO CENTx + 1 STEP -1
FOR mv = CENTy + 38 TO CENTy + 1 STEP -1
   IF (mv - CENTy + 38) >= (mh - CENTx + 38) THEN
     nodo = 0
     pf = POINT(mh, mv)
      IF pf = 0 THEN
          GOSUB LPUNTOFINAL
      END IF
```

```
      END IF
REGRESE5:
    NEXT mv
    NEXT mh
    cig = 0
    NUMB = 0
    DO UNTIL NUMB = NUMER - 1
     NUMB = NUMB + 1
     GOSUB VECINDAD1
     FOR YA = 0 TO 100
      cam = 0
      GOSUB BUSCACAMINO
      IF cam <= 2 THEN GOTO NOMBRAR5 ELSE GOTO ENCONTRAR5
NOMBRAR5:
        IF YA = 0 THEN
          IF cam = 1 THEN
            IF M(YA) = CENTx + 1 THEN
               dat(5, 1) = 4
               PSET (M(YA), N(YA)), 14
               cua5 = cua5 + 1
               EXIT FOR
             ELSE
               dat(5, 1) = 3
               PSET (M(YA), N(YA)), 14
               cua5 = cua5 + 1
               EXIT FOR
             END IF
          END IF
          GOSUB CAMINO1
          GOTO REGRESO5
        END IF

DIST25:
        GOSUB CAMINO2
        IF cam = 1 THEN
         IF M(YA) = CENTx + 1 THEN
           dat(5, 1) = 4
           PSET (M(YA), N(YA)), 14
           cua5 = cua5 + 1
           EXIT FOR
         ELSE
           dat(5, 1) = 3
           PSET (M(YA), N(YA)), 14
           cua5 = cua5 + 1
           EXIT FOR
         END IF
        END IF
        GOTO REGRESO5
ENCONTRAR5:
        IF cam = 3 THEN
          dat(5, 1) = 9
          cua5 = cua5 + 1
          EXIT FOR
        END IF
REGRESO5:
        NEXT YA
       LOOP
        IF cua5 = 0 THEN
          dat(5, 1) = 0
        END IF


       '-------------------------------------
       ' Deteccion de rasgo con abertura hacia la izquierda
       ' y hacia arriba
       '-------------------------------------
       cua5 = 0
       primp = 0
       cruz = 0
       aber = 0
       aber1 = 0
       FOR hort = CENTx + 38 TO CENTx + 1 STEP -1
       FOR verti = CENTy + 38 TO CENTy + 1 STEP -1
       IF (verti - CENTy + 38) >= (hort - CENTx + 38) THEN
         pinta = POINT(hort, verti)
```

```
      IF pinta = 0 THEN
         IF hort <> CENTx + 1 THEN
           GOSUB BUSCADATOS
           GOTO SIGUIENTE5
         ELSE
           IF aber1 = 0 THEN
             REM INPUT "abert"; rei
             primp = primp + 1
             ppv(primp) = verti
             aber1 = aber1 + 1
           ELSE
             GOSUB PUNTOEJEY
           END IF
         END IF
       END IF
     END IF
SIGUIENTE5:
     NEXT verti
     NEXT hort
     IF aber = 0 AND aber1 = 2 THEN
       dat(5, 2) = 6
       cua5 = cua5 + 1
     ELSE
       IF aber = 2 AND aber1 = 0 THEN
        dat(5, 2) = 7
        cua5 = cua5 + 1
       END IF
     END IF
     IF cua5 = 0 THEN
      dat(5, 2) = 0
     END IF


     ' localizacion de rasgo horizontal
     '-------------------------
     ant1 = 0
     ante2 = 0
     can5 = 0
     can = 0
     FOR hort = CENTx + 38 TO CENTx + 1 STEP -1
     FOR verti = CENTy + 38 TO CENTy + 1 STEP -1
     IF (verti - CENTy + 38) >= (hort - CENTx + 38) THEN
      neig = POINT(hort, verti)
      IF neig = 0 THEN
       IF can = 0 THEN
        GOSUB RASGOSVH
        IF can = 1 THEN
         can5 = 1
         EXIT FOR
        END IF
       END IF
          IF hort = CENTx + 1 THEN
            ante2 = ante2 + 1
            puntoy(ante2) = verti
            IF ante2 = 1 THEN
            ant1 = ant1 + 1
            ELSE
             IF ABS(puntoy(ante2) - puntoy(ante2 - 1)) >= 1 THEN
               ant1 = ant1 + 1
             END IF
            END IF
          END IF
      END IF
     END IF
     NEXT verti
     NEXT hort
     IF can5 = 1 AND ante2 >= 1 AND ant1 = 1 THEN
       dat(5, 3) = 1
       dat(5, 4) = 0
     ELSE
       IF can5 = 1 AND ante2 >= 1 AND ant1 >= 2 THEN
         dat(5, 3) = 1
         dat(5, 4) = 2
       ELSE
         dat(5, 3) = 0
```

```
        END IF
      END IF
      IF escoja = 1 THEN
       oct = oct + 1
       ' definicion del vector del cuadrante V
       GOSUB ALMACENARDATO
      END IF


      ' cuadrante VI
      '_____
      '   "localizacion de nodos puntos final"

      cua6 = 0
      apunte = 0
      NUMER = 1
      FOR mh = CENTx - 38 TO CENTx - 1
      FOR mv = CENTy + 38 TO CENTy + 1 STEP -1
      IF (mv - CENTy + 38) + (mh - CENTx + 38) >= 76 THEN
       nodo = 0
       pf = POINT(mh, mv)
       IF pf = 0 THEN
         GOSUB LPUNTOFINAL
       END IF
      END IF
REGRESE6:
      NEXT mv
      NEXT mh
      eigt = 0
      NUMB = 0
      DO UNTIL NUMB = NUMER - 1
      NUMB = NUMB + 1
      GOSUB VECINDAD1
      FOR YA = 0 TO 100
       cam = 0
       GOSUB BUSCACAMINO
       IF cam <= 2 THEN GOTO NOMBRAR6 ELSE GOTO ENCONTRAR6
NOMBRAR6:
       IF YA = 0 THEN
        IF cam = 1 THEN
         IF M(YA) = CENTx - 1 THEN
           dat(6, 1) = 3
           PSET (M(YA), N(YA)), 14
           cua6 = cua6 + 1
           EXIT FOR
         ELSE
           dat(6, 1) = 4
           PSET (M(YA), N(YA)), 14
           cua6 = cua6 + 1
           EXIT FOR
        END IF
       END IF
       GOSUB CAMINO1
       GOTO REGRESO6
      END IF

DIST26:
      GOSUB CAMINO2
      IF cam = 1 THEN
        IF M(YA) = CENTx - 1 THEN
           dat(6, 1) = 3
           PSET (M(YA), N(YA)), 14
           cua6 = cua6 + 1
           EXIT FOR
        ELSE
           dat(6, 1) = 4
           PSET (M(YA), N(YA)), 14
           cua6 = cua6 + 1
           EXIT FOR
        END IF
      END IF
      GOTO REGRESO6
ENCONTRAR6:
       IF cam = 3 THEN
         dat(6, 1) = 9
```

```
        cua6 = cua6 + 1
        EXIT FOR
      END IF
REGRESO6:
    NEXT YA
    LOOP
      IF cua6 = 0 THEN
       dat(6, 1) = 0
      END IF

' Deteccion de rasgo con abertura hacia la derecha
' y hacia arriba
'-------------------------------------
      cua6 = 0
      primp = 0
      cruz = 0
      aber = 0
      aber1 = 0
      FOR hort = CENTx - 38 TO CENTx - 1
      FOR verti = CENTy + 38 TO CENTy + 1 STEP -1
      IF (verti - CENTy + 38) + (hort - CENTx + 38) >= 76 THEN
        pinta = POINT(hort, verti)
        IF pinta = 0 THEN

          IF hort <> CENTx - 1 THEN
            GOSUB BUSCADATOS
            GOTO SIGUIENTE6
          ELSE
            IF aber1 = 0 THEN
              primp = primp + 1
              ppv(primp) = verti
              aber1 = aber1 + 1
            ELSE
              GOSUB PUNTOEJEY
            END IF
          END IF
        END IF
      END IF
SIGUIENTE6:
      NEXT verti
      NEXT hort
      IF aber = 0 AND (aber1 = 2 XOR aber1 = 3) THEN
        dat(6, 2) = 5
        cua6 = cua6 + 1
      ELSE
        IF aber = 2 AND aber1 = 0 THEN
          dat(6, 2) = 7
          cua6 = cua6 + 1
        END IF
      END IF
      IF cua6 = 0 THEN
       dat(6, 2) = 0
      END IF


    ' localizacion de rasgo horizontal
    '-------------------------------
      artu = 0
      artu1 = 0
      can6 = 0
      can = 0
      FOR hort = CENTx - 38 TO CENTx - 1
      FOR verti = CENTy + 38 TO CENTy + 1 STEP -1
      IF (verti - CENTy + 38) + (hort - CENTx + 38) >= 76 THEN
       neibr = POINT(hort, verti)
       IF neibr = 0 THEN
          IF can = 0 THEN
           GOSUB RASGOSVH
           IF can = 1 THEN
            can6 = can6 + 1
            EXIT FOR
           END IF
          END IF
            IF hort = CENTx - 1 THEN
               artu1 = artu1 + 1
```

```
            END IF
          END IF
        END IF
        NEXT verti
        NEXT hort
        IF can6 = 1 AND artu1 = 1 THEN
          dat(6, 3) = 1
        ELSE
          IF can6 = 1 AND artu1 = 2 THEN
           dat(6, 3) = 1
           dat(6, 4) = 2
          ELSE
           dat(6, 3) = 0
          END IF
        END IF
        IF escoja = 1 THEN
         oct = oct + 1
         ' definicion del vector del cuadrante VI
         GOSUB ALMACENARDATO
        END IF


        '_____________________
        ' cuadrante VII
        '_____________

        '  "localizacion de nodos puntos final"
        '  ------------------------------
        cua7 = 0
        apunte = 0
        NUMER = 1
        FOR mv = CENTy + 38 TO CENTy + 1 STEP -1
        FOR mh = CENTx - 38 TO CENTx - 1
        IF (mv - CENTy + 38) + (mh - CENTx + 38) < 76 THEN
         nodo = 0
         pf = POINT(mh, mv)
         IF pf = 0 THEN
          GOSUB LPUNTOFINAL
         END IF
        END IF
REGRESE7:
        NEXT mh
        NEXT mv
        eigt = 0
        NUMB = 0
        DO UNTIL NUMB = NUMER - 1
         NUMB = NUMB + 1
         GOSUB VECINDAD1
         FOR YA = 0 TO 100
          cam = 0
          GOSUB BUSCACAMINO
          IF cam <= 2 THEN GOTO NOMBRAR7 ELSE GOTO ENCONTRAR7
NOMBRAR7:
          IF YA = 0 THEN
           IF cam = 1 THEN
            IF N(YA) = CENTy + 1 THEN
             dat(7, 1) = 4
             PSET (M(YA), N(YA)), 14
             cua7 = cua7 + 1
             EXIT FOR
            ELSE
             dat(7, 1) = 3
             PSET (M(YA), N(YA)), 14
             cua7 = cua7 + 1
             EXIT FOR
            END IF
           END IF
           GOSUB CAMINO1
           GOTO REGRESO7
          END IF

DIST27:
         GOSUB CAMINO2
         IF cam = 1 THEN
          IF N(YA) = CENTy + 1 THEN
```

```
            dat(7, 1) = 4
            PSET (M(YA), N(YA)), 14
            cua7 = cua7 + 1
            EXIT FOR
           ELSE
            dat(7, 1) = 3
            PSET (M(YA), N(YA)). 14
            cua7 = cua7 + 1
            EXIT FOR
           END IF
         END IF
         GOTO REGRESO7
ENCONTRAR7:
         IF cam = 3 THEN
          dat(7, 1) = 9
          cua7 = cua7 + 1
          EXIT FOR
         END IF
REGRESO7:

         NEXT YA
        LOOP
      ' definicion de no existencia de rasgo
       IF cua7 = 0 THEN
        dat(7, 1) = 0
       END IF

      ' Deteccion de rasgo con abertura hacia la derecha
      ' y hacia arriba
      '-------------------------------------
      cua7 = 0
      primp = 0
      cruz = 0
      aber = 0
      aber1 = 0
      FOR verti = CENTy + 38 TO CENTy + 1 STEP -1
      FOR hort = CENTx - 38 TO CENTx - 1
      IF (verti - CENTy + 38) + (hort - CENTx + 38) < 76 THEN
       pinta = POINT(hort, verti)
        IF pinta = 0 THEN

            IF verti <> CENTy + 1 THEN
              GOSUB BUSCADATOS
              GOTO SIGUIENTE7
            ELSE
              IF aber1 = 0 THEN
                primp = primp + 1
                pph(primp) = hort
                aber1 = aber1 + 1
              ELSE
                GOSUB PUNTOEJEX

              END IF
            END IF
         END IF
        END IF
SIGUIENTE7:
         NEXT hort
         NEXT verti
         IF aber = 0 AND aber1 = 2 THEN
          dat(7, 2) = 7
          cua7 = cua7 + 1
         ELSE
          IF aber = 2 AND aber1 = 0 THEN
          dat(7, 2) = 5
          cua7 = cua7 + 1
         END IF
      END IF
       ' definicion de la no existencia de rasgo
       IF cua7 = 0 THEN
         dat(7, 2) = 0
       END IF

       ' localizacion de rasgo vertical
       '-------------------------
```

```
      primp = 0
      aber1 = 0
      alfa = 0
      alfa1 = 0
      can7 = 0
      can = 0
      FOR verti = CENTy + 38 TO CENTy + 1 STEP -1
      FOR hort = CENTx - 38 TO CENTx - 1
      IF (verti - CENTy + 38) + (hort - CENTx + 38) < 76 THEN
        becin = POINT(hort, verti)
        IF becin = 0 THEN
          IF can = 0 THEN
            GOSUB RASGOSVH
            IF can = 1 THEN
              can7 = can7 + 1
              EXIT FOR
            END IF
          END IF
            IF verti = CENTy + 1 THEN
                IF alfa1 = 0 THEN

                aber1 = aber1 + 1
                primp = primp + 1
                pph(primp) = hort
                alfa1 = alfa1 + 1
                ELSE
                GOSUB PUNTOEJEX
                END IF
              END IF
              END IF
          END IF
      NEXT hort
      NEXT verti
      IF can7 = 1 AND aber1 = 1 THEN
        dat(7, 3) = 2
      ELSE
        IF can7 = 1 AND aber1 = 2 THEN
          dat(7, 3) = 2
          dat(7, 4) = 9
        ELSE
          dat(7, 3) = 0
        END IF
      END IF
      IF escoja = 1 THEN
        oct = oct + 1
        ' definicion del vector del cuadrante VII
        GOSUB ALMACENARDATO
      END IF

      ' cuadrante VIII
      '----------
      '   "localizacion de nodos puntos final"
      cua8 = 0
      apunte = 0
      NUMER = 1
      FOR mv = CENTy - 38 TO CENTy - 1
      FOR mh = CENTx - 38 TO CENTx - 1
      IF (mv - CENTy + 38) >= (mh - CENTx + 38) THEN
        nodo = 0
        pf = POINT(mh, mv)
        IF pf = 0 THEN
          GOSUB LPUNTOFINAL
        END IF
      END IF
REGRESE8:
      NEXT mh
      NEXT mv
      eigt = 0
      NUMB = 0
      DO UNTIL NUMB = NUMER - 1
        NUMB = NUMB + 1
        GOSUB VECINDAD1
        FOR YA = 0 TO 100
          cam = 0
          GOSUB BUSCACAMINO
```

```
        IF cam <= 2 THEN GOTO NOMBRAR8 ELSE GOTO ENCONTRAR8
NOMBRAR8:
        IF YA = 0 THEN
         IF cam = 1 THEN
          IF N(YA) = CENTy + 1 THEN
           dat(8, 1) = 3
           PSET (M(YA), N(YA)), 14
           cua8 = cua8 + 1
           EXIT FOR
          ELSE
           dat(8, 1) = 3
           PSET (M(YA), N(YA)), 14
           cua = cua8 + 1
           EXIT FOR
          END IF

         END IF
         GOSUB CAMINO1
         GOTO REGRESO8
        END IF

DIST28:
        GOSUB CAMINO2
        IF cam = 1 THEN
         IF N(YA) = CENTy + 1 THEN
          dat(8, 1) = 3
          PSET (M(YA), N(YA)), 14
          cua8 = cua8 + 1
          EXIT FOR
         ELSE
          dat(8, 1) = 4
          PSET (M(YA), N(YA)), 14
          cua8 = cua8 + 1
          EXIT FOR
         END IF
        END IF
        GOTO REGRESO8

ENCONTRAR8:

        IF cam = 3 AND ((N(YA) - CENTy + 38) >= (M(YA) - CENTx + 38)) THEN
         PSET (M(YA), N(YA)), 15
         dat(8, 1) = 9
         cua8 = cua8 + 1
         EXIT FOR
        END IF

REGRESO8:

        NEXT YA
       LOOP
       IF cua8 = 0 THEN
         dat(8, 1) = 0
       END IF

    ' deteccion de rasgo con abertura hacia la derecha
    ' y hacia abajo
    '-------------------------
    cua8 = 0
    primp = 0
    cruz = 0
    aber = 0
    aber1 = 0
    FOR verti = CENTy - 38 TO CENTy - 1
    FOR hort = CENTx - 38 TO CENTx - 1
      IF (verti - CENTy + 38) >= (hort - CENTx + 38) THEN
        pinta = POINT(hort, verti)
        IF pinta = 0 THEN
          IF verti <> CENTy - 1 THEN
             GOSUB BUSCADATOS
             GOTO SIGUIENTE8
          ELSE
             IF aber1 = 0 THEN
               REM INPUT "abert"; rei
               primp = primp + 1
```

```
            pph(primp) = hort
            aber1 = aber1 + 1
          ELSE
            GOSUB PUNTOEJEX
          END IF
        END IF
      END IF
    END IF

SIGUIENTE8:

    NEXT hort
    NEXT verti
    IF aber = 0 AND aber1 = 2 THEN
      dat(8, 2) = 8
      cua8 = cua8 + 1
    ELSE
      IF aber = 2 AND aber1 = 0 THEN
        dat(8, 2) = 5
        cua8 = cua8 + 1
      END IF
    END IF
    IF cua8 = 0 THEN
     dat(8, 2) = 0
    END IF

    '_____________________
    '  Deteccion de rasgo vertical
    '_____________________
    siga = 0
    aber1 = 0
    primp = 0
    beta1 = 0
    can8 = 0
    can = 0
    FOR verti = CENTy - 38 TO CENTy - 1
    FOR hort = CENTx - 38 TO CENTx - 1
     IF (verti - CENTy + 38) >= (hort - CENTx + 38) THEN
       ventan = POINT(hort, verti)
       IF ventan = 0 THEN
        IF can = 0 THEN
          GOSUB RASGOSVH
          IF can = 1 THEN
            can8 = can8 + 1
            EXIT FOR
          END IF
        END IF
            IF verti = CENTy - 1 THEN
              IF aber1 = 0 THEN
                REM INPUT "abert"; rei
                primp = primp + 1
                pph(primp) = hort
                aber1 = aber1 + 1
              ELSE
                GOSUB PUNTOEJEX
              END IF

            END IF
            END IF
          END IF
    NEXT hort
    NEXT verti
    IF (can8 = 1 AND aber1 = 1) XOR (can8 = 1 AND siga <= 2 AND siga >= 1) THEN
     dat(8, 3) = 2
    ELSE
     IF can8 = 1 AND siga >= 4 THEN
      dat(8, 3) = 5
     ELSE
      dat(8, 3) = 0
     END IF
    END IF
    IF escoja = 1 THEN
     oct = oct + 1
     ' definicion del vector del cuadrante VIII
     GOSUB ALMACENARDATO
```

```
    END IF
    IF escoja = 2 THEN
     GOSUB RECOGNITION
    END IF

PRONUM:

    IF escoja = 2 THEN
     LOCATE 24, 8: INPUT "DESEA OTRO NUMERO PULSE (1)/ DESEA TERMINAR PULSE(2)"; opti
     IF opti = 2 THEN
      END
     END IF
    END IF

    INPUT "proximonumer"; h$
    count = 0: No = No + 1: CLS : k = k + 1: IF k = 10 GOTO fin ELSE GOTO repit
fin:
    fu = 0
    k = 0
    etiq = 0
    utr = 0: DO UNTIL Dif = 9: Dif = Dif + 1: PRINT "p="; p: PRINT " d = "; Dif: GOTO repit
    LOOP
    count = 0
    No = 0
    Dif = 0
    CLOSE #1
  LOOP
END


'-------------------
' SUBRUTINAS
'-------------------
LPUNTOFINAL:
      PF0 = POINT(mh + 1, mv)
      IF PF0 = 0 XOR PF0 = 12 THEN
       nodo = nodo + 1
      END IF
      PF1 = POINT(mh + 1, mv - 1)
       IF PF1 = 0 XOR PF1 = 12 THEN
        nodo = nodo + 1
       END IF
      PF2 = POINT(mh, mv - 1)
      IF PF2 = 0 XOR PF2 = 12 THEN
       nodo = nodo + 1
      END IF
      PF3 = POINT(mh - 1, mv - 1)
      IF PF3 = 0 XOR PF3 = 12 THEN
       nodo = nodo + 1
      END IF
      PF4 = POINT(mh - 1, mv)
      IF PF4 = 0 XOR PF4 = 12 THEN
       nodo = nodo + 1
      END IF
      PF5 = POINT(mh - 1, mv + 1)
      IF PF5 = 0 XOR PF5 = 12 THEN
       nodo = nodo + 1
      END IF
      PF6 = POINT(mh, mv + 1)
      IF PF6 = 0 XOR PF6 = 12 THEN
       nodo = nodo + 1
      END IF
      PF7 = POINT(mh + 1, mv + 1)
      IF PF7 = 0 XOR PF7 = 12 THEN
       nodo = nodo + 1
      END IF

      IF nodn = 1 THEN
       X(NUMER) = mh
       Y(NUMER) = mv
       NUMER = NUMER + 1
      END IF
    RETURN

VECINDAD1:
```

```
camino = 0
d = POINT(X(NUMB), Y(NUMB))
D0 = POINT(X(NUMB) + 1, Y(NUMB))
D1 = POINT(X(NUMB) + 1, Y(NUMB) - 1)
D2 = POINT(X(NUMB), Y(NUMB) - 1)
D3 = POINT(X(NUMB) - 1, Y(NUMB) - 1)
D4 = POINT(X(NUMB) - 1, Y(NUMB))
D5 = POINT(X(NUMB) - 1, Y(NUMB) + 1)
D6 = POINT(X(NUMB), Y(NUMB) + 1)
D7 = POINT(X(NUMB) + 1, Y(NUMB) + 1)
geo0 = d XOR D0
IF geo0 = 0 THEN
 camino = camino + 1
 M(0) = X(NUMB) + 1
 N(0) = Y(NUMB)
END IF
geo1 = d XOR D1
IF geo1 = 0 THEN
 camino = camino + 1
 M(0) = X(NUMB) + 1
 N(0) = Y(NUMB) - 1
END IF
geo2 = d XOR D2
IF geo2 = 0 THEN
  camino = camino + 1
  M(0) = X(NUMB)
  N(0) = Y(NUMB) - 1
END IF
geo3 = d XOR D3
IF geo3 = 0 THEN
 camino = camino + 1
 M(0) = X(NUMB) - 1
 N(0) = Y(NUMB) - 1
END IF
geo4 = d XOR D4
IF geo4 = 0 THEN
 camino = camino + 1
 M(0) = X(NUMB) - 1
 N(0) = Y(NUMB)
END IF
geo5 = d XOR D5
IF geo5 = 0 THEN
 camino = camino + 1
 M(0) = X(NUMB) - 1
 N(0) = Y(NUMB) + 1
END IF
geo6 = d XOR D6
IF geo6 = 0 THEN
 camino = camino + 1
 M(0) = X(NUMB)
 N(0) = Y(NUMB) + 1
END IF
geo7 = d XOR D7
IF geo7 = 0 THEN
 camino = camino + 1
 M(0) = X(NUMB) + 1
 N(0) = Y(NUMB) + 1
END IF
RETURN

BUSCACAMINO:

 g = POINT(M(YA), N(YA))
 IF g = 0 THEN
  g0 = POINT(M(YA) + 1, N(YA))
  g1 = POINT(M(YA) + 1, N(YA) - 1)
  g2 = POINT(M(YA), N(YA) - 1)
  g3 = POINT(M(YA) - 1, N(YA) - 1)
  g4 = POINT(M(YA) - 1, N(YA))
  g5 = POINT(M(YA) - 1, N(YA) + 1)
  g6 = POINT(M(YA), N(YA) + 1)
  g7 = POINT(M(YA) + 1, N(YA) + 1)
  IF g0 = 0 THEN
   cam = cam + 1
```

```
        Hx(cam) = M(YA) + 1
        Hy(cam) = N(YA)
       END IF
       IF g1 = 0 THEN
        cam = cam + 1
        Hx(cam) = M(YA) + 1
        Hy(cam) = N(YA) - 1
       END IF
       IF g2 = 0 THEN
        cam = cam + 1
        Hx(cam) = M(YA)
        Hy(cam) = N(YA) - 1
       END IF
       IF g3 = 0 THEN
        cam = cam + 1
        Hx(cam) = M(YA) - 1
        Hy(cam) = N(YA) - 1
       END IF
       IF g4 = 0 THEN
        cam = cam + 1
        Hx(cam) = M(YA) - 1
        Hy(cam) = N(YA)
       END IF
       IF g5 = 0 THEN
        cam = cam + 1
        Hx(cam) = M(YA) - 1
        Hy(cam) = N(YA) + 1
       END IF
       IF g6 = 0 THEN
        cam = cam + 1
        Hx(cam) = M(YA)
        Hy(cam) = N(YA) + 1
       END IF
       IF g7 = 0 THEN
        cam = cam + 1
        Hx(cam) = M(YA) + 1
        Hy(cam) = N(YA) + 1
       END IF
     END IF
   RETURN

CAMINO1:
     IF Hx(1) <> X(NUMB) OR Hy(1) <> Y(NUMB) THEN
       M(YA + 1) = Hx(1)
       N(YA + 1) = Hy(1)
     END IF
     IF Hx(2) <> X(NUMB) OR Hy(2) <> Y(NUMB) THEN
      M(YA + 1) = Hx(2)
      N(YA + 1) = Hy(2)
     END IF
     RETURN

CAMINO2:

       IF Hx(1) <> M(YA - 1) OR Hy(1) <> N(YA - 1) THEN
        M(YA + 1) = Hx(1)
        N(YA + 1) = Hy(1)
       END IF
       IF Hx(2) <> M(YA - 1) OR Hy(2) <> N(YA - 1) THEN
         M(YA + 1) = Hx(2)
         N(YA + 1) = Hy(2)
       END IF
      RETURN

BUSCADATOS:

      pinta1 = POINT(hort + 1, verti)
      IF pinta1 = 12 THEN
           cruz = cruz + 1
           ph(cruz) = hort
           pv(cruz) = verti
             IF cruz = 1 THEN
               aber = aber + 1
               GOTO REGRESA
             ELSE
```

```
            IF ABS(ph(cruz) - ph(cruz - 1)) >= 2 AND ABS(pv(cruz) - pv(cruz - 1)) >= 2 THEN
              aber = aber + 1

              GOTO REGRESA
            END IF
          END IF
        END IF
      pinta2 = POINT(hort + 1, verti - 1)
        IF pinta2 = 12 THEN
          cruz = cruz + 1

          ph(cruz) = hort
          pv(cruz) = verti
            IF cruz = 1 THEN
              aber = aber + 1
              GOTO REGRESA
            ELSE
              IF ABS(ph(cruz) - ph(cruz - 1)) >= 2 AND ABS(pv(cruz) - pv(cruz - 1)) >= 2 THEN
                aber = aber + 1

                GOTO REGRESA
              END IF
            END IF
        END IF


    pinta3 = POINT(hort, verti - 1)
        IF pinta3 = 12 THEN
          cruz = cruz + 1

          ph(cruz) = hort
          pv(cruz) = verti
            IF cruz = 1 THEN
              aber = aber + 1
              GOTO REGRESA
            ELSE
              IF ABS(ph(cruz) - ph(cruz - 1)) >= 2 AND ABS(pv(cruz) - pv(cruz - 1)) >= 2 THEN
                aber = aber + 1

                GOTO REGRESA
              END IF
            END IF
        END IF

    pinta4 = POINT(hort - 1, verti - 1)
        IF pinta4 = 12 THEN
          cruz = cruz + 1

          ph(cruz) = hort
          pv(cruz) = verti
            IF cruz = 1 THEN
              aber = aber + 1
              GOTO REGRESA
            ELSE
              IF ABS(ph(cruz) - ph(cruz - 1)) >= 2 AND ABS(pv(cruz) - pv(cruz - 1)) >= 2 THEN
                aber = aber + 1

                GOTO REGRESA
              END IF
            END IF
        END IF


    pinta5 = POINT(hort - 1, verti)
        IF pinta5 = 12 THEN
          cruz = cruz + 1

          ph(cruz) = hort
          pv(cruz) = verti
            IF cruz = 1 THEN
              aber = aber + 1
              GOTO REGRESA
            ELSE
              IF ABS(ph(cruz) - ph(cruz - 1)) >= 2 AND ABS(pv(cruz) - pv(cruz - 1)) >= 2 THEN
                aber = aber + 1
```

```
                GOTO REGRESA
              END IF
            END IF
        END IF


pinta6 = POINT(hort - 1, verti + 1)
      IF pinta6 = 12 THEN
          cruz = cruz + 1

          ph(cruz) = hort
          pv(cruz) = verti
            IF cruz = 1 THEN
              aber = aber + 1
              GOTO REGRESA
            ELSE
              IF ABS(ph(cruz) - ph(cruz - 1)) >= 2 AND ABS(pv(cruz) - pv(cruz - 1)) >= 2 THEN
                aber = aber + 1

                GOTO REGRESA
              END IF
            END IF
        END IF

pinta7 = POINT(hort, verti + 1)
    IF pinta7 = 12 THEN
          cruz = cruz + 1

          ph(cruz) = hort
          pv(cruz) = verti
            IF cruz = 1 THEN
              aber = aber + 1
              GOTO REGRESA
            ELSE
              IF ABS(ph(cruz) - ph(cruz - 1)) >= 2 AND ABS(pv(cruz) - pv(cruz - 1)) >= 2 THEN
                aber = aber + 1

                GOTO REGRESA
              END IF
            END IF
        END IF

pinta8 = POINT(hort + 1, verti + 1)
       IF pinta8 = 12 THEN
          cruz = cruz + 1

          ph(cruz) = hort
          pv(cruz) = verti
           IF cruz = 1 THEN
              aber = aber + 1
              GOTO REGRESA
           ELSE
             IF ABS(ph(cruz) - ph(cruz - 1)) >= 2 AND ABS(pv(cruz) - pv(cruz - 1)) >= 2 THEN
                aber = aber + 1

                GOTO REGRESA
              END IF
            END IF
        END IF

REGRESA: RETURN

PUNTOEJEY:

            primp = primp + 1
            ppv(primp) = verti
            IF ABS(ppv(primp) - ppv(primp - 1)) >= 2 THEN
             aber1 = aber1 + 1
            END IF
            RETURN
PUNTOEJEX:

            primp = primp + 1
            pph(primp) = hort
```

```
            IF ABS(pph(primp) - pph(primp - 1)) >= 2 THEN
              aber1 = aber1 + 1
            ELSE
              siga = siga + 1
            END IF
            RETURN

RASGOSVH:
        ' "Dátos para rasgos vertical y horizontal"

       cal1 = POINT(hort + 1, verti)
       IF cal1 = 12 THEN
        can = can + 1
        GOTO REGRESAS
       END IF
      cal2 = POINT(hort + 1, verti - 1)
      IF cal2 = 12 THEN
        can = can + 1
        GOTO REGRESAS
       END IF
      cal3 = POINT(hort, verti - 1)
      IF cal3 = 12 THEN
        ' PSET (hort, verti - 1), 9
        can = can + 1
        GOTO REGRESAS
       END IF
      cal4 = POINT(hort - 1, verti - 1)
      IF cal4 = 12 THEN
        can = can + 1
        GOTO REGRESAS
       END IF
      cal5 = POINT(hort - 1, verti)
      IF cal5 = 12 THEN
        can = can + 1
        GOTO REGRESAS
       END IF
      cal6 = POINT(hort - 1, verti + 1)
      IF cal6 = 12 THEN
        can = can + 1
        GOTO REGRESAS
      END IF
     cal7 = POINT(hort, verti + 1)
     IF cal7 = 12 THEN
        can = can + 1
        GOTO REGRESAS
     END IF
      cal8 = POINT(hort + 1, verti + 1)
        IF cal8 = 12 THEN
         can = can + 1
         GOTO REGRESAS
        END IF
REGRESAS: RETURN

ALMACENARDATO:

        TYPE tipodat
        comp1 AS STRING * 5
        comp2 AS STRING * 2
        END TYPE
        DIM vector AS tipodat
        OPEN "C:\archivo1\archtop2" FOR RANDOM AS #4 LEN = LEN(vector)
        vector.comp1 = STRS(dat(oct, 1) * 1000 + dat(oct, 2) * 100 + dat(oct, 3) * 10 + dat(oct, 4))
        vector.comp2 = STRS(etiq - 1)
        IF oct = 8 THEN
          oct = 0
        END IF
        nvect = nvect + 1
        PUT #4, nvect, vector
        PRINT LOC(4)
        CLOSE #4
      RETURN

RECOGNITION:

        TYPE TipoComparar
```

```
    lee1 AS STRING * 56
    END TYPE
    DIM sacadato AS TipoComparar
    OPEN "C:\archivo1\architop2" FOR RANDOM AS #4 LEN = LEN(sacadato)
    numdatos = LOF(4) / LEN(sacadato)
    encont = 0
    FOR contar = 1 TO numdatos
      GET #4, contar, sacadato
      FOR I = 1 TO 8
       pin = 2 + (7 * (I - 1))
       tiS = MIDS(sacadato.lee1, pin, 4)
       compa = VAL(tiS)
       total1 = dat(I, 1) * 1000 + dat(I, 2) * 100 + dat(I, 3) * 10 + dat(I, 4)
       subtotal = compa - total1
       IF subtotal = 0 THEN
        IF I = 8 THEN
         ti2S = RIGHTS(sacadato.lee1, 1)
         solu = VAL(ti2S)
         PRINT "el numero es="; solu
         ti3S = MIDS(sacadato.lee1, 7, 1)
         sl = VAL(ti3S)
         encont = 1
        END IF
       ELSE
        EXIT FOR
       END IF
      NEXT I
      IF encont = 1 THEN
        EXIT FOR
      END IF
    NEXT contar
    CLOSE #4
    IF encont = 0 THEN
     PRINT "Figura no reconocida"
    END IF
  RETURN
```