



UNIVERSIDAD AUTÓNOMA METROPOLITANA

Maestría en Ciencias de la Computación

Reporte de actividades del proyecto de
Investigación en computación I

Título de la tesis

“Detección y clasificación de objetos dentro
de un salón de clases empleando técnicas
de procesamiento digital de imágenes”

Presenta:

➤ *Elías García Santillán*

Asesor:

➤ *Dr. Carlos Avilés Cruz*

México Distrito Federal a 27 de noviembre de 2006

TABLA DE CONTENIDOS

CAPÍTULO 1. INTRODUCCIÓN

1.1 Visión por computadora	8
1.2 Aplicaciones de visión artificial	9
1.3 Aplicaciones con tratamiento digital de imágenes	10
1.4 Etapas del reconocimiento automático de objetivos	11
1.5 Descripción del proyecto	12

CAPÍTULO 2. ESTADO DEL ARTE

2.1 Probando el desempeño de varias combinaciones de clasificadores y vectores de rasgos en la determinación de la identidad y el numero de objetos circulando en una banda transportadora.....	16
2.2 Localización de placas en vehículos automotores.....	21
2.3 Nueva técnica para contar objetos en imágenes.....	24

CAPÍTULO 3. ADQUISICIÓN DE LA IMAGEN

3.1 Elementos de percepción visual	30
3.2 Captación de la imagen por la cámara	34
3.3 Espacios de color	36
3.4 Iluminación	42
3.5 Resolución de la imagen.....	43
3.6 Formatos de imágenes	44

CAPÍTULO 4. PREPROCESAMIENTO

4.1 Conversión de una imagen en color a niveles de gris....	47
4.2 Conversión de una imagen en niveles de gris a blanco y negro	48
4.3 Mejoramiento de histograma	49
4.4 Uso de filtros para el mejoramiento de la imagen	53
4.5 Suavizado binario.....	57
4.6 Operaciones lógicas sobre imágenes binarias	58
4.7 Operaciones aritméticas sobre imágenes en tonos de gris	60
4.8 Operaciones morfológicas básicas	62
4.9 Extracción de contornos	66

CAPÍTULO 5. SEGMENTACIÓN

5.1 Detección de regiones basada en umbrales.....	74
5.2 Método de segmentación de Otsu	75

CAPÍTULO 6. REPRESENTACIÓN Y DESCRIPCIÓN

(Proyecto terminal II)

CAPÍTULO 7. RECONOCIMIENTO E INTERPRETACIÓN

(Proyecto terminal II)

CAPÍTULO 8. GRABACIÓN DE ARCHIVOS DE AUDIO

(Proyecto terminal II)

CAPÍTULO 9. IMPLEMENTACIÓN DEL PROYECTO Y RESULTADOS PARCIALES

9.1 Adquisición de la imagen	84
9.2 Preprocesamiento de la imagen	87
9.3 Segmentación	91

.
. .
.

CONCLUSIONES

(Proyecto terminal II)

BIBLIOGRAFÍA

Bibliografía general	98
Bibliografía electrónica	99

ANEXOS

(Proyecto terminal II)

Índice de figuras

Capítulo 1

Figura 1.2.1	Aplicaciones de Visión Artificial	9
Figura 1.3.1	Cefalograma	10
Figura 1.3.2	Fallas en estructuras	10
Figura 1.4.1	Etapas fundamentales en el reconocimiento automático de objetivos	11
Figura 1.5.1	Diagrama de bloques del sistema	12
Figura 1.5.2	Método de reconocimiento piramidal	13
Figura 1.5.3	Objetos a reconocer por nuestro sistema	14

Capítulo 2

Figura 2.1.1	Objetos convertidos en tuplas de características	17
Figura 2.1.2	Objetos a reconocer	18
Figura 2.2.1	Imagen original	22
Figura 2.2.2	Operador de textura	22
Figura 2.2.3	Patrón de la placa	22
Figura 2.2.4	Regiones seleccionadas	23
Figura 2.2.5	Función de correlación	23
Figura 2.2.6	Operador and	23
Figura 2.2.7	Valores de correlación.....	23
Figura 2.2.8	Localización de la placa	23
Figura 2.3.1	Ejemplod de blobs	25

Capítulo 3

Figura 3.1.1:	Anatomía del ojo humano	30
Figura 3.1.2:	Formación de imágenes en el ojo humano	31
Figura 3.1.3:	Distribución de conos y bastones en la retina	32
Figura 3.1.4:	Respuesta del ojo humano a diferentes longitudes de onda	33
Figura 3.1.5:	Mezclas de luz	33
Figura 3.1.6:	Mezclas de pigmentos	33
Figura 3.2.1:	dispositivo de acoplamiento de carga	34
Figura 3.3.1:	Distribución de colores en el cubo RGB	36
Figura 3.3.2:	Distribución de color YCbCr	37
Figura 3.3.3:	Distribución del color en el modelo XYZ	38
Figura 3.3.4:	Espacio de color CIELAB	39
Figura 3.3.5:	Espacio de color HSI	41
Figura 3.4.1:	Reflexión Especular	42
Figura 3.4.2:	Reflexión difusa	42
Figura 3.5.1:	Diferentes tipos de resolución	43
Figura 3.6.1:	Tamaño de una imagen con diferentes tipos de formatos	44

Capítulo 4

Figura 4.1.1: Representación de una imagen en color	47
Figura 4.1.2: Imagen es escala de gris con su matriz de intensidades.....	48
Figura 4.2.1: imagen binarizada y su matriz de intensidades.....	48
Figura 4.3.1: Histograma de una imagen en escala de gris	49
Figura 4.3.2: Diferentes tipos de histogramas	49
Figura 4.3.3: Imagen oscura	50
Figura 4.3.4: Histograma de la imagen oscura	50
Figura 4.3.5: Imagen con desplazamiento	50
Figura 4.3.6: Histograma con desplazamiento	50
Figura 4.3.7: Histograma correspondiente a la tabla 4.3.1	52
Figura 4.3.8: Histograma obtenido después de la ampliación	52
Figura 4.3.9: Imagen obtenida por el método de ampliación de histograma.....	52
Figura 4.3.10: Histograma correspondiente a la ampliación de histograma.....	52
Figura 4.4.1: Proceso de filtrado	53
Figura 4.4.2: Ejemplo de máscara de 3x3	54
Figura 4.4.3: conjunto de coordenadas de una mascara de 3x3	54
Figura 4.4.4: imagen en tonos de gris.	54
Figura 4.4.5: imagen resultado al aplicar un filtro promedio.....	54
Figura 4.4.6: filtro “punto máximo”	56
Figura 4.4.7: filtro “punto mínimo”	56
Figura 4.4.8: filtro “mediana”	56
Figura 4.5.1: Coordenadas del punto p(x,y) y sus vecinos	57
Figura 4.5.2: Asignación de letras a cada vecino del punto “p”	57
Figura 4.5.3: Eliminación de pequeños huecos.....	57
Figura 4.5.4: Eliminación de pequeñas protuberancias.....	58
Figura 4.6.1: Aplicación de operaciones binarias básicas.....	59
Figura 4.7.1. Efecto de aplicar la operación de suma para combinar dos imágenes	61
Figura 4.7.2: Efecto de aplicar la operación de resta para detectar movimiento .	61
Figura 4.7.3: Efecto de aplicar la multiplicación para aislar regiones de interés ..	61
Figura 4.8.1: Elemento estructural N4.....	62
Figura 4.8.2: Elemento estructural N8	62
Figura 4.8.3: Dilatación	63
Figura 4.8.4: Erosión con un elemento estructurante de 1x2	63
Figura 4.8.5: Erosión con un elemento estructurante de 3x3.....	63
Figura 4.8.6. Apertura	64
Figura 4.8.7. Cerradura	64
Figura 4.8.9: Extracción de frontera	65
Figura 4.9.1: Gráfica de una imagen discontinua.	66
Figura 4.9.2: Mascaras de Sobel	68
Figura 4.9.3: Imagen muestra	68

Figura 4.9.4: Aplicación de mascararas de Sobel en figuras	68
Figura 4.9.5: Máscaras de Prewitt	69
Figura 4.9.6: Operadores de Prewitt.	69
Figura 4.9.7: Operador de Roberts.....	70
Figura 4.9.8: Mascaras de Kirsch	71
Figura 4.9.9: Operador de Kirsch	71
Figura 4.9.10: Máscaras de operadores Laplaciana	72
Figura 4.9.11 Aplicación de los operadores Laplaciana.....	72

Capítulo 5

Figura 5.1.1: Imagen en tonos de gris	74
Figura 5.1.2: Histograma	74
Figura 5.1.3: umbral óptimo 97	74
Figura 5.1.4: umbral = 45	74
Figura 5.1.5: umbral =140	74

Capítulo 6

(Proyecto terminal II)

Capítulo 7

(Proyecto terminal II)

Capítulo 8

(Proyecto terminal II)

Capítulo 9

(Proyecto terminal II)

Capítulo 10

Figura 10.1.1: VideoCam Genius NB.....	84
Figura 10.1.2: Captación de la imagen a través de la VideoCam.....	84
Figura 10.1.3: Diferentes tipos de resolución	85
Figura 10.1.4: Objeto a reconocer con una mala iluminación.....	85
Figura 10.1.5: Consideraciones importantes en la captación de imágenes.....	86
Figura 10.2.1. Conversión de una imagen en color a blanco y negro.....	87
Figura 10.2.2: Uso de filtros para una imagen con ruido aleatorio.....	88
Figura 10.2.3. Ampliación de histograma	90
Figura 10.3.1. Binarización de la imagen.....	91
Figura 10.3.2 Operaciones morfológicas.....	93
Figura 10.3.3 Eliminación de huecos y pequeños objetos.....	93
Figura 10.3.4 Multiplicación de matrices punto a punto.....	94
Figura 10.3.5 Aplicación algoritmos para la detección de contornos.....	95

Índice de Tablas

Capítulo 2

Tabla 2.1 Desempeño de la combinación Bayesiano-Flusse-Suk.....	19
Tabla 2.2 Desempeño de la combinación Bayesiano-Flusse-Suk ante cambios de iluminación.....	20

Capítulo 4

Tabla 4..3.1: Distribución de los niveles de gris antes y después de la igualación.....	51
Tabla 4.6.1: Operaciones binarias AND y OR.....	58
Tabla 4.6.2: Operación NOT.	59

Índice de programas en Matlab

Capítulo 10

Programa #1 Captación de la imagen.....	86
Programa #2 Conversión a tonos de gris.....	87
Programa #3: Filtros para imágenes en tonos de gris.....	89
Programa #4: Ampliación de histograma.....	90
Programa #5: Segmentación con diferentes umbrales.....	92
Programa #6: Segmentación con el método de Otsu.....	92
Programa #7: Detección de contornos.....	96

Capítulo 1

INTRODUCCIÓN

1.1 Visión por computadora

La visión es uno de los mecanismos sensoriales de percepción más importantes que tiene el ser humano y la mayoría de los organismos biológicos. La utilizamos para desenvolvemos eficientemente dentro del ambiente que nos rodea y detectar los objetos que son de nuestro interés por medio de su forma, color, relieve, dimensiones, distancia a la que se encuentra, etcétera.

La visión por computadora es la capacidad de la máquina para ver el mundo que le rodea, para deducir la estructura y las propiedades del mundo tridimensional a partir de una o más imágenes bidimensionales. La implantación en una máquina de habilidades como la de detectar y determinar la identidad de los objetos no sólo liberan al hombre de tareas tediosas y peligrosas sino también permite la realización de algunas otras tareas imposibles de realizar para el ser humano.

El problema de visión por computadora es un problema abierto para el cuál no existe algún algoritmo eficaz que reconozca cualquier tipo de objeto en cualquier tipo de ambiente y en el tiempo en que nuestro sentido de la vista lo realiza.

1.2. Aplicaciones de visión artificial

Existen cientos de algoritmos para el reconocimiento automático de objetivos (*Automatic Target Recognition*) para algún caso en particular, por ejemplo: la industria automotriz utiliza para el reconocimiento y ensamblado de piezas [11]; en otras empresas se usa para, la detección de placas de autos [2], el reconocimiento de personas [14], seguimiento automático de objetos [4], recolección de frutos [5], conteo de bacterias [10], usos militares [13], juego de fútbol en competencias de robots [17], el reconocimiento automático de señales de tránsito [12], entre muchas otras aplicaciones (ver figura 1.2.1).

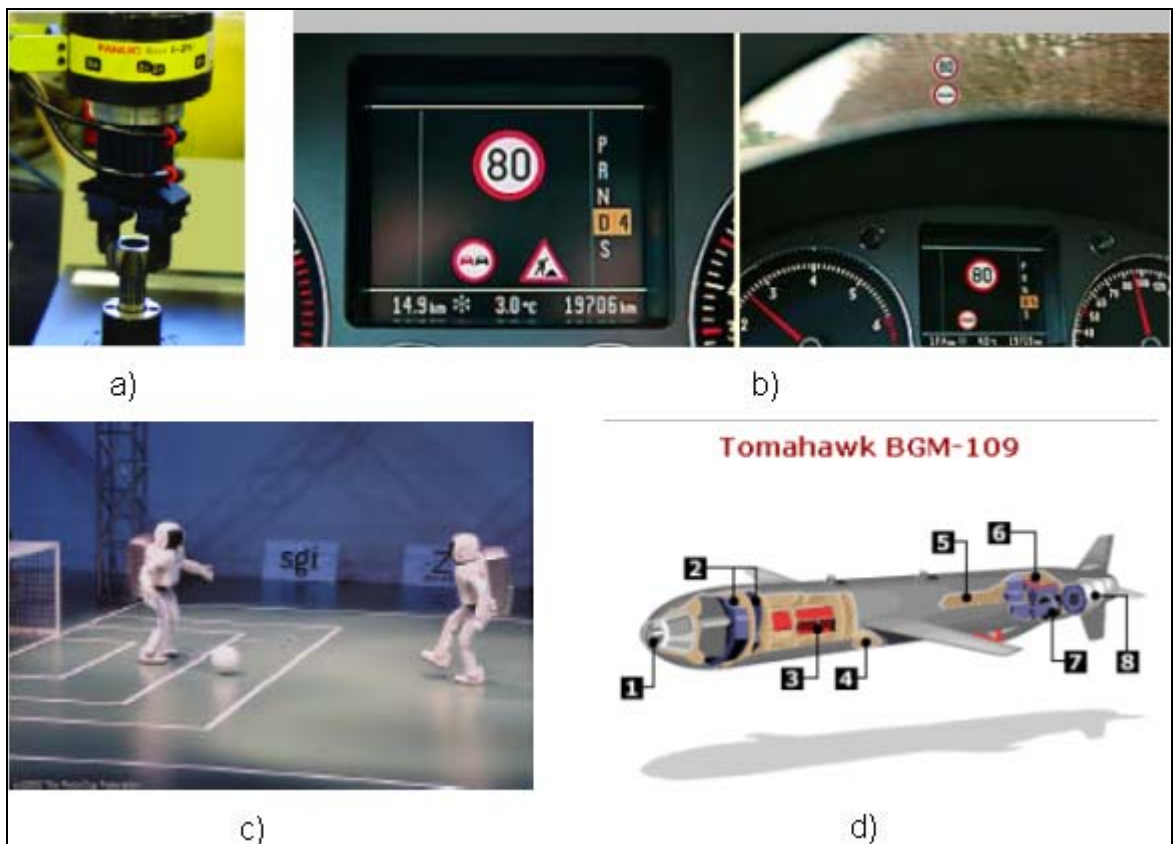


figura 1.2.1 Aplicaciones de Visión Artificial. a) Ensamble de piezas por un robot, b) Reconocimiento automático de señales de tránsito. c) Juego de fútbol con humanoides. d) Usos militares.

1.3 Aplicaciones con tratamiento digital de imágenes

El procesamiento de imágenes también tiene muchas aplicaciones; por ejemplo, en la medicina se utiliza para el mejoramiento de imágenes obtenidas con fines de diagnóstico médico. Alterando los valores de la luminosidad de los píxeles mediante transformaciones matemáticas en imágenes se logra detectar fallas de estructuras metálicas. Otro ejemplo es el mejoramiento de imágenes aéreas para realizar exámenes de diagnóstico del terreno, localizar plantíos de droga, analizar recursos naturales, las fallas geológicas, etcétera. En las Figuras 1.3.1 y 1.3.2 se muestran dos ejemplos sobre el tratamiento digital de imágenes.

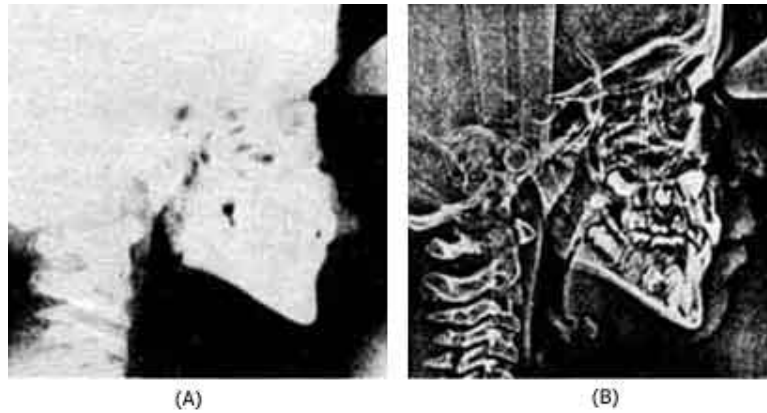


Figura 1.3.1. Cefalograma: A) Imagen original y B) imagen procesada.

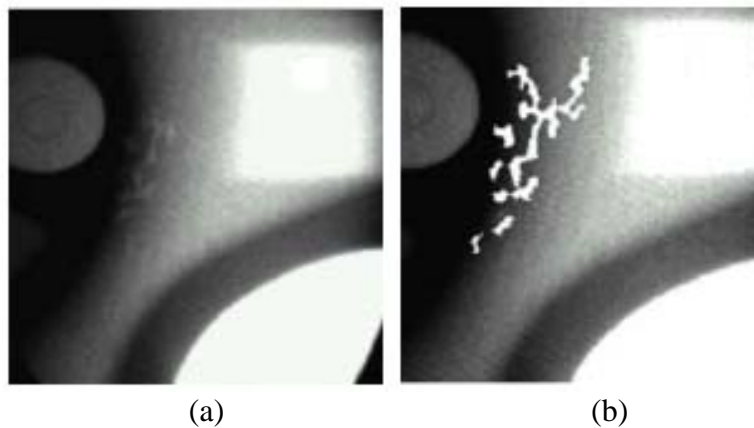


Figura 1.3.2. Fallas en estructuras: a) Imagen original, b) imagen procesada.

1.4 Etapas del reconocimiento automático de objetivos

El proceso para el reconocimiento automático de objetivos se inicia con la Adquisición de la imagen de una escena en tres dimensiones, a continuación esta imagen se procesa para mejorar la calidad y eliminar posibles imperfecciones, el siguiente paso es separar el objeto de interés del fondo de la imagen, seguida de la extracción de sus características que describen al objeto (color, textura y geometría), para finalmente, comparar estas características con las de otros objetos que se tienen en la base de conocimiento y así determinar el tipo de objeto (figura 1.4.1).

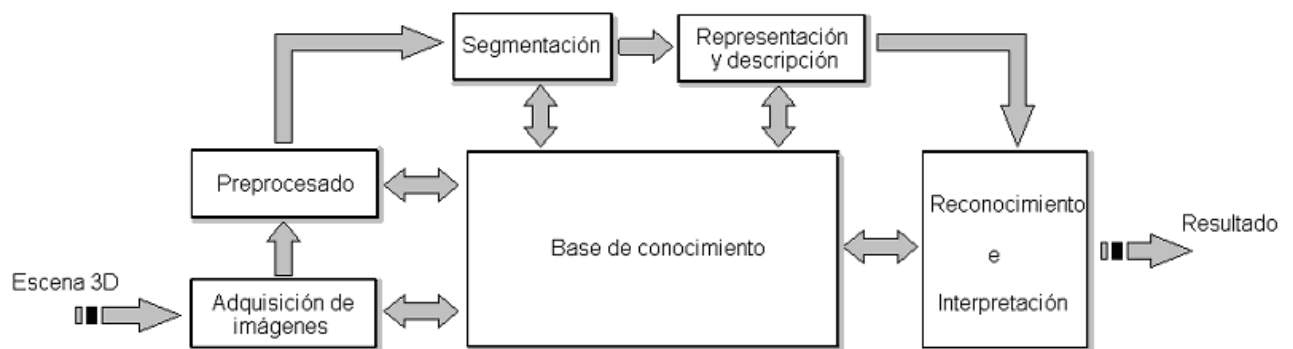


Figura 1.4.1: Etapas fundamentales en el reconocimiento automático de objetivos

En cada una de las etapas anteriores se requiere de un conocimiento previo, como puede ser: el tamaño del objeto, la distancia a la que se encuentra, las condiciones ambientales, el número de objetos, la inclinación, entre otros. Toda esta información es necesaria para la utilización de determinadas técnicas y el desarrollo de algoritmos adicionales para el reconocimiento del objeto.

Cada una de estas etapas se describirán con mas detalle en los siguientes capítulos de esta tesis.

1.5 Descripción del proyecto

El sistema de visión de reconocimiento de objetos que se presenta en esta tesis inicialmente se planteó para personas invidentes; pero debido a la problemática existente en el diseño de algoritmos generales eficaces, y diversos factores que intervienen en su desarrollo (que son de gran importancia para la obtención de buenos resultados) como son: iluminación, condiciones atmosféricas, tamaño de los objetos, distancia de los objetos, fondos contrastantes con los objetos a reconocer, objetos traslapados, entre otros. Debido a lo anterior, nuestro sistema se implementa dentro de un salón de clases y se limita a reconocer 10 objetos pequeños.

Se utilizó una videocámara para la captación de la imagen y un láser para el cálculo de la distancia del objeto, a continuación se procesa esta imagen por medio de nuestro sistema de reconocimiento para su clasificación y finalmente se muestra en forma audible el nombre del objeto identificado y su distancia con respecto de la videocámara (figura 1.5.1).



Figura 1.5.1 Diagrama de bloques del sistema

El método de reconocimiento que se implementa en forma piramidal (figura 1.5.2) teniendo como primer criterio el color del objeto, como segundo criterio la textura y por último el reconocimiento por su geometría. Con esta técnica se pretende mejorar el porcentaje de certeza para el reconocimiento del objeto utilizando tres técnicas de reconocimiento. Si al implementar el reconocimiento por color tenemos una certeza mayor al 95%, no será necesario utilizar las otras dos técnicas; de lo contrario, se utiliza el reconocimiento por textura y si no fueron suficientes los primeros dos métodos para reconocer con certeza el objeto en cuestión se utiliza el reconocimiento por su geometría.

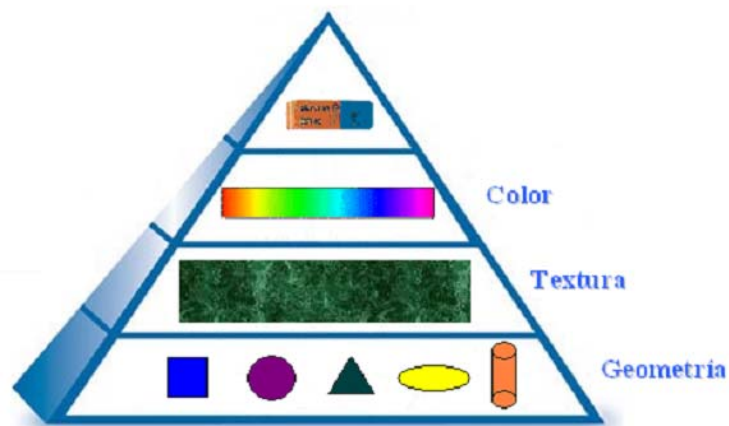


Figura 1.5.2: Método de reconocimiento piramidal

Los objetos a reconocer (figura 1.5.3) son pequeños con características muy variadas en cuanto al color, textura y forma.

Por ejemplo: el color del lápiz puede variar (amarillo, blanco, morado, etc), puede tener goma, punta, estar un poco usado, entre otras características. Cada tipo de lápiz debe estar registrado en la base de conocimientos de nuestro sistema con sus características propias, y talvez no se pueda reconocer directamente por el color debido a que el color del lápiz puede ser el mismo de otro objeto a reconocer, pero la textura puede variar o la forma alargada de el lápiz marcará la diferencia. En síntesis, se requiere que un objeto contenga las mismas características de color, textura y forma para que pueda confundirse con otro objeto distinto.

El reconocimiento se realiza en un ambiente controlado como el salón de clases y con una buena iluminación. El objeto a reconocer debe de estar en una posición canónica para extraer mejor sus características y sin tener otros objetos transpuestos.

				
Lápiz	Pluma	Resistol	Borrador	Goma
				
Sacapuntas	calculadora	Diurex	Corrector	Marcador

Figura 1.5.3: Objetos a reconocer por nuestro sistema.

Capítulo 2

ESTADO DEL ARTE

En esta etapa se recopiló y analizó información que sirve como base y nos ideas para el desarrollo de nuestro proyecto a elaborar. Nos Ayuda a conocer la problemática a la que nos enfrentamos al desarrollar nuestro sistema y ver las soluciones que se plantean.

Los artículos presentados en este capítulo, tesis de maestría y doctorales son muy importantes ya que muestran un panorama muy amplio sobre las técnicas empleadas en el reconocimiento automático de objetivos (RAO).

También se analizaron diversos artículos que no fueron de gran importancia para el desarrollo del proyecto o en algunos casos solo se menciona la técnica que emplearon para el reconocimiento, pero no se explica como se realizó ni que algoritmos o fórmulas emplearon; por lo tanto, no se mencionan en esta tesis.

Cabe mencionar que aún falta por agregar en este capítulo dos artículos sobre el "reconocimiento por textura" ya que pertenecen al proyecto terminal II y por el momento se están analizando las fórmulas empleadas para aplicar estas técnicas a nuestro sistema de reconocimiento.

2.1. Probando el desempeño de varias combinaciones de clasificadores y vectores de rasgos en la determinación de la identidad y el número de objetos circulando en una banda transportadora.

Karla Penélope Xicotencatl Aguilar. Tesis de maestría, 2001, Ed. Instituto Politécnico Nacional.

Resumen

El conocer el tipo de objeto, el número circundando en una banda transportadora es muy importante. Esto permitiría a un brazo manipulador, localizar y planear la trayectoria para tomar dicho objeto y realizar con él una tarea especificada.

Un objeto puede aparecer en la banda en posiciones variadas. Cuando dicho objeto es tomado por el brazo manipulador puede aparecer incluso más chico o grande, dependiendo de la distancia y posición de dicho sensor al objeto. Otro punto importante es considerar las sombras y los cambios de iluminación.

Los tres clasificadores probados son: el basado en el cálculo de la distancia mínima, el basado en el cálculo de la distancia de Mahalanobis y el Bayesiano. Los descriptores usados son los invariantes a traslaciones, rotaciones y cambios de escala de Hu y los invariantes a afinidades propuestos por Flusser y Suk.

Un sistema de reconocimiento de patrones opera con todos los posibles objetos individuales que se van a reconocer. Estos objetos suelen denominarse patrones.

Existen varias maneras de reconocer un patrón. Una de ellas consiste en transformar dicho patrón en una tupla "X" cuyas componentes se denominan rasgos o características (ver figura 2.1.1). Cada tupla de rasgos se compara con un

diccionario de tuplas preestablecidas, compuesto por las tuplas de rasgos de todos los objetos del universo de trabajo.

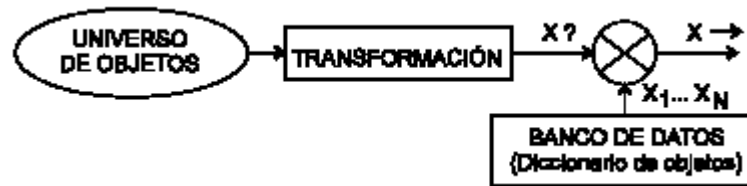


Figura 2.1.1 Objetos convertidos en tuplas de características

Se puede observar que los objetos individuales se convierten en tuplas “X” de rasgos antes de ser reconocidos que corresponden a las características del objeto.. La notación usada para las tuplas de rasgos es de tipo columna:

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{bmatrix} \quad (2.1.1)$$

en donde, x_1, x_2, \dots, x_p , pueden ser cualquier tipo de variable.

Una vez calculada la tupla “X” asociada a un objeto individual, su reconocimiento se basa en determinar su grado de semejanza con las tuplas previamente definidas. La selección del número y tipo de rasgos descriptores será tal que patrones individuales puedan ser discriminados o diferenciados. Es deseable que el valor de cada rasgo sea también invariante ante transformaciones.

El método para probar el desempeño de los clasificadores y descriptores seleccionados consta de dos etapas:

1) Etapas de obtención de funciones discriminantes: Este procedimiento describe los pasos de la etapa de entrenamiento para cada combinación clasificador-vector de rasgos. Dado una combinación clasificador-vector de rasgos, C_i y un conjunto de objetos a reconocer, $O_j, j = 1, \dots, N$ realizar los siguientes pasos:

- ☆ Obtener P imágenes del objeto O_j
- ☆ Para cada imagen del objeto O_j . Aislar el objeto y calcular sus invariantes de Hu y de $Flusser$.
- ☆ Obtener los vectores medios y matrices de covarianza respectivos, según sea el caso.

2) Etapa de prueba: Dada una imagen f , conteniendo una o más instancias de uno o más objetos del universo de objetos usados durante la etapa de entrenamiento realizar los siguientes pasos:

1. Obtener la versión binaria b de la imagen f .
2. Aplicar cualquier algoritmo de etiquetado sobre b para obtener las regiones R_k $k=1, \dots, M$ de cada objeto.
3. Para cada R_k , obtener los rasgos característicos.
4. Aplicar la combinación C_i a cada R_k para obtener su clase correspondiente.
5. Obtener las estadísticas de desempeño para cada combinación clasificador-vector de rasgos.

El procedimiento anterior da como resultado los porcentajes de aciertos para cada combinación de clasificador-vector de rasgos.

Pruebas de desempeño

Los seis objetos usados para la experimentación se muestran en la figura 2.1.2. Se nota la presencia de sombras y brillos, que como es sabido influyen fuertemente en los procesos de entrenamiento.



Figura 2.1.2. Objetos a reconocer.

En todos los casos, se probó el desempeño de las siguientes combinaciones clasificador-vector de rasgos:

1. Clasificadores de distancia mínima–vector completo de Hu: [$\phi_1, \phi_2, \phi_3, \phi_4, \phi_5, \phi_6, \phi_7$].
2. Clasificador de distancia mínima–vector reducido de Hu: [$\phi_4, \phi_5, \phi_6, \phi_7$].
3. Clasificador de distancia mínima–vector completo de Flusser-Suk: [$I_1, I_2, I_3, I_4, I_5, I_6$].
4. Clasificador de distancia mínima–vector reducido de Flusser-Suk: [I_2, I_4, I_5].
5. Clasificadores de distancia de Mahalanobis–vector completo de Hu: [$\phi_1, \phi_2, \phi_3, \phi_4, \phi_5, \phi_6, \phi_7$].
6. Clasificador de distancia de Mahalanobis–vector reducido de Hu: [$\phi_4, \phi_5, \phi_6, \phi_7$].
7. Clasificador de distancia de Mahalanobis–vector completo de Flusser-Suk: [$I_1, I_2, I_3, I_4, I_5, I_6$].
8. Clasificador de distancia de Mahalanobis–vector reducido de Flusser-Suk: [I_2, I_4, I_5].
9. Clasificador Bayesiano–vector completo de Hu: [$\phi_1, \phi_2, \phi_3, \phi_4, \phi_5, \phi_6, \phi_7$].
10. Clasificador Bayesiano–vector reducido de Hu: [$\phi_4, \phi_5, \phi_6, \phi_7$].
11. Clasificador Bayesiano–vector completo de Flusser-Suk: [$I_1, I_2, I_3, I_4, I_5, I_6$].
12. Clasificador Bayesiano–vector reducido de Flusser-Suk: [I_2, I_4, I_5].

Se usaron vectores reducidos ya que se quería verificar si el desempeño de estos vectores con los respectivos clasificadores daba un mejor desempeño. Los vectores se obtuvieron al escoger aquellos invariantes con medias más separadas.

Desempeño de las combinaciones sin cambios de iluminación

Se tomaron 25 imágenes para cada una de las inclinaciones de 0, 22.5, 25 y 67.5 grados del sensor con respecto a la vertical a diversas distancias para obtener las deformaciones requeridas. El total de objetos reconocidos y porcentaje de reconocimiento de cada combinación se muestra en la tabla 2.1.1. Podemos notar que el clasificador Bayesiano–Flusser-Suk da el mejor resultado. El segundo lugar lo ocupó la combinación Mahalanobis-vector reducido de Flusser-Suk.

Combinación No.	Total de objetos reconocidos	Porcentaje de aciertos
1	316	52.6
2	323	53.8
3	359	59.8
4	342	57.0
5	455	75.8
6	418	69.6
7	462	77.0
8	469	78.1
9	463	77.1
10	451	75.1
11	477	79.5
12	449	74.8

Tabla 2.1.1 Desempeño de la combinación Bayesiano-Flusse-Suk.

Se probó el desempeño de la combinación Bayesiano-Flusser-Zuk ante cambios de iluminación dado que en el experimento anterior se obtuvieron mejores resultados. Se usaron 4 cambios de iluminación. Los resultados obtenidos se muestran en la tabla 2.1.2, obteniendo que el desempeño de la combinación sube y después baja al final del cuarto nivel de iluminación.

Cambio de iluminación No.	Total de objetos reconocidos	Porcentaje de aciertos
1	405	67.5
2	469	78.1
3	477	79.5
4	470	78.3

Tabla 2.1.2 Desempeño de la combinación Bayesiano-Flusse-Suk ante cambios de iluminación.

Se probó también el desempeño de la combinación Bayesiano-Flusser-Suk con objetos pintados de negro para eliminar los brillos, y como era de esperarse, el desempeño del clasificador fué del 91.16%.

Otra prueba que se realizó es el de determinar la identidad y número de objetos circulando por una banda transportadora. Para ello se debe calcular la velocidad de la banda, el sistema toma una imagen del objeto dentro del campo visual del sensor y calcula su centroide en el tiempo t_1 . Toma una siguiente imagen calculando de nuevo el centroide del mismo objeto en el tiempo t_2 . Con esta información se estima la velocidad v como:

$$v = \frac{x_2 - x_1}{t_2 - t_1} \quad (2.1.2)$$

donde, x_1 y x_2 son las coordenadas de los centros del objeto.

Se probó únicamente el desempeño de la combinación Bayesiano-Flusser-Suk, obteniéndose un desempeño del 96.66%. Los errores en la determinación del número de objetos circulando sobre la banda se debieron a los tiempos de procesamiento, los cambios de velocidad de la banda y, las sombras y brillos en los objetos.

2.2. Localización de placas en vehículos automotores.

Arturo Legarda Sáenz, .Tesis de maestría, 2001, Ed. Instituto Tecnológico de Chihuahua.

Resumen

El proceso implementado consta de varias etapas: adquisición de imágenes de vehículos, reprocesamiento a las imágenes capturadas, extracción y selección de regiones candidatas a contener la placa del vehículo y la localización de la placa.

Adquisición de imágenes

Fueron adquiridas a través de una videocámara Soni en formato BMP a (640x480x16) de resolución.

Preprocesamiento

A las imágenes capturadas con baja resolución se les aplica un operador de textura con el fin de localizar regiones. Este operador se basa en la medición de la energía de la textura presente en las superficies:

$$Tif : I(m,n) \rightarrow I_t(m,n) \quad (2.2.1)$$

donde: $I(m,n)$ es una imagen y $I_t(m,n)$ es la imagen generada por el operador de textura. Tif se define como $\text{avg}(\text{var}(\text{vecindad } i))$, representando avg la función promedio y var es la varianza alrededor del pixel i de la imagen considerando cuatro direcciones, dentro de un vecindario vertical, horizontal, diagonal derecha y diagonal izquierda en una ventana de 3x3. El operador de textura genera regiones marcadas por bordes fuertes. A estos bordes se les aplica un suavizado con un filtro Gaussiano con desviación estándar de 0.8. para ampliar las líneas de píxeles que conectan regiones.

Extracción De Regiones

Se elimina información no necesaria mediante binarización con valor de umbral de 120. Se eliminan bordes que no generan una región cerrada y se aplica un operador morfológico de apertura, el cual está dado por:

$$Ro = (Ra \ominus Rb) \oplus Rb \quad (2.2.2)$$

donde, Ra es la región a la cual se le aplica la apertura, Rb el elemento estructural en que se basa la apertura, en nuestro caso es un rectángulo de 3x3 píxeles, y Ro corresponde a la región resultante después de la apertura que es el resultado de una erosión (operador \ominus) de un área Ra por un elemento estructural Rb seguido por una dilatación (operador \oplus) usando el mismo elemento estructural. A continuación se seleccionan las regiones que cumplen con área mínima=2000 píxeles y máxima de 300000 que es casi la totalidad de la imagen y se crea una nueva imagen con los píxeles de la imagen original que cumplen las siguientes condiciones:

$$r(s,t) = \frac{\sum_m \sum_n [I(m,n) - \hat{I}(m,n)] [W(m-s,n-t) - w]}{\{ \sum_m \sum_n [I(m,n) - \hat{I}(m,n)]^2 + \sum_m \sum_n [W(m-s,n-t) - w]^2 \}^{1/2}} \quad (2.2.3)$$

donde: $I(m,n)$ es la imagen, $\hat{I}(m,n)$ es el promedio de la imagen, $W(m,n)$ es el patrón y w el promedio de $W(m,n)$. El patrón representa el objeto a identificar, en este caso la placa, El patrón (template), se generó con las imágenes del promedio de varias placas de automóviles. Para la obtención de las imágenes de placas se procedió a seleccionar y recortar de la imagen correspondiente a la placa. Como se muestra en las figuras 2.2.1, 2.2.2 y 2.2.3.



Fig.2.2.1 Imagen original **Fig. 2.2.2** operador de textura. **Fig. 2.2.3** Patrón de la placa

Al realizar la búsqueda de la placa en la imagen se obtuvo un porcentaje de localización menor al 40% el cual no es aceptable.

Para mejorar el desempeño del sistema se modificó el operador de textura que mide la energía de la textura con lo que las áreas uniformes se minimizan, quedando los bordes resaltados. Esta imagen de textura se combinó con la imagen original usando un operador “and” lógico. Obteniéndose una imagen en la que los bordes aparecen resaltados al tener valores de gris altos, mientras que las regiones uniformes sin bordes desaparecen (ver figuras 2.2.4-2.2.8).



Fig. 2.2.4 Regiones seleccionadas

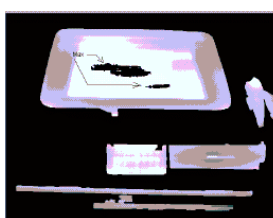


Fig.2.2.5 Función de correlación



Fig. 2.2.6 Operador and



Fig. 2.2.7 valores de correlación



Fig. 2.2.8 localización de la placa

Resultados

Los resultados obtenidos haciendo la modificación anterior tiene un 83% de localización correcta de la placa. Este desempeño aunque no es muy alto para un localizador es relativamente bueno comparado con sistemas que incluyen un conjunto de restricciones los cuales reportan rangos que van del 89% al 100% en la localización de la placa.

2.3. Nueva técnica para contar objetos en imágenes.

Rafael Sotelo Rangel, Tesis de maestría, 2001, Ed. Instituto Politécnico Nacional.

Resumen

El método del que habla el autor , se aplica a imágenes que contengan objetos semi-circulares, de contornos suaves, sin hoyos pero si puede contener traslapes entre objetos. El autor nos da varias definiciones, colorarios y teoremas para la mejor comprensión del tema (ver figura 2.3.1):

Un blob.- Es una región conectada por píxeles tal que entre cualesquiera de dos píxeles p1 y p2 de dicha región puede encontrar un camino conectando dichos píxeles.

Un q-blob.- Es un blob compacto semi-circular cuyo factor de circularidad asociado se encuentra muy cerca de la unidad.

Factor de circularidad, FC de una forma binaria es:

$$FC = 4\pi \frac{A}{P^2} \quad (2.3.1)$$

donde, P es el perímetro del objeto y A su área.

Punto de concavidad singular.- es un punto de contorno resultado del traslape de dos q-blobs

Un conglomerado.- Es un conjunto de q-blobs traslapándose unos con otros.

Un hoyo.- Es una región tipo fondo resultado del traslape de varios q-blobs.

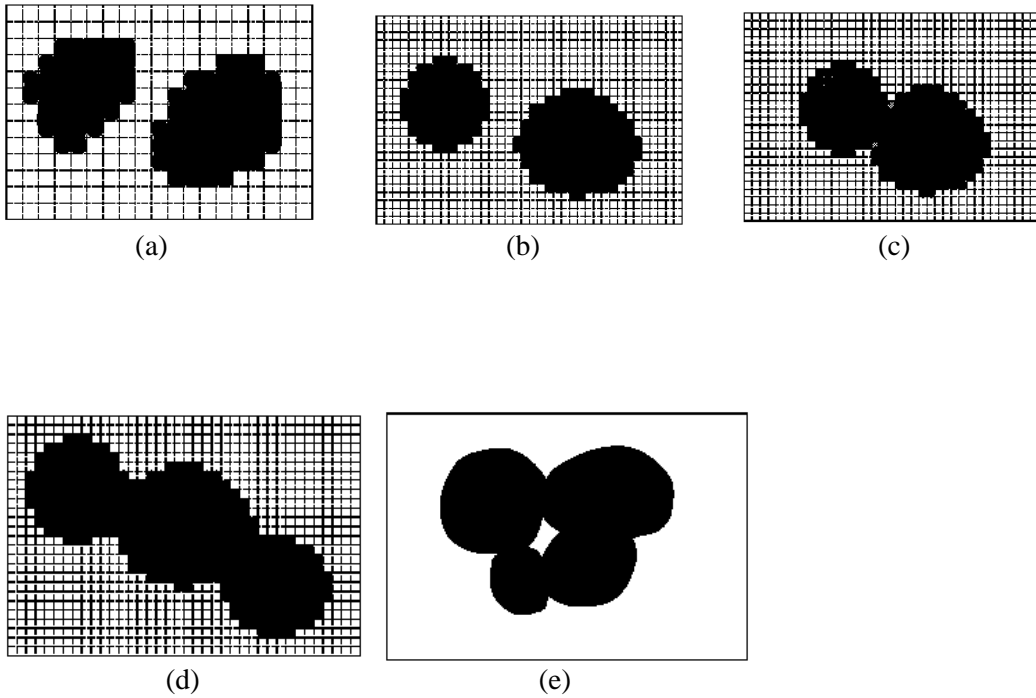


Figura 2.3.1 (a) Ejemplo de 2 blobs, (b) Ejemplo de 2 q-blob, (c) Puntos de concavidad, (d) Conglomerado, (e) Hoyo .

Lema 1. La manera de adicionar dos puntos de concavidad singular es a un conglomerado consiste en traslapar un q-blob con dicho conglomerado sin generar un hoyo.

Lema 2. La manera en que un hoyo puede ser generado consiste en traslapar un q-blob con un conglomerado de al menos dos q-blobs. Dicho q-blob debe traslapar a ambos q-blobs formando el conglomerado.



Teorema 1: sea ΔNB el número de q-blobs y ΔNP el número de puntos de concavidad singulares adicionados a un conglomerado. El número de hoyos adicionales ΔH vale siempre:

$$\Delta H = \frac{\Delta NP}{2} - \Delta NB \quad (6)$$

Teorema 2: El adicionar dos puntos de concavidad singulares a un conglomerado no altera el número de hoyos.

Teorema 3(a). El número de hoyos para un conglomerado viene siempre dado por:

$$NH = \frac{NP}{2} - NB + 1 \quad (7)$$

Teorema 3(b). El número de hoyos para NC conglomerados viene siempre dado como:

$$NH = \frac{NP}{2} - NB + NC \quad (8)$$

Corolario 1. Si NP es el número de puntos de concavidad singulares, NC el número de conglomerados, y NH el número de hoyos en una imagen, entonces el número de q-blobs, NB en una imagen viene dado como:

$$NB = \frac{NP}{2} - NC + NH \quad (9)$$

La Técnica

La técnica consta de cinco etapas. Cada una de estas se menciona a continuación:

Preprocesamiento de la imagen

En esta etapa una imagen monocromática proveniente de un sensor CCD es primeramente binarizada manualmente.

Determinación del número de conglomerados.

Se obtiene a través del etiquetado de todas y cada una de las componentes conectadas en la imagen binarizada. La imagen es barrida de izquierda a derecha y de arriba a abajo hasta localizar el primer píxel, negro, asignándole la primera etiqueta. Una vez encontrado este píxel, sus ocho vecinos son analizados, si cualquiera de éstos es negro es etiquetado con la misma etiqueta del píxel central.

Determinación del número de hoyos en una imagen.

Consiste en invertir la imagen original, contar el número de regiones usando el procedimiento para contar conglomerados, para obtener el valor deseado.

Determinación del número de puntos de concavidad singulares.

La imagen original es primeramente dilatada usando un elemento estructural circular de diámetro siete. La imagen resultante es enseguida erosionada con el mismo elemento estructural. Esto da como resultado una imagen cerrada, morfológicamente hablando. La imagen original es restada de esta imagen cerrada para obtener la imagen final, la cual contendrá tantas regiones conexas como puntos de concavidad singulares haya en la imagen original

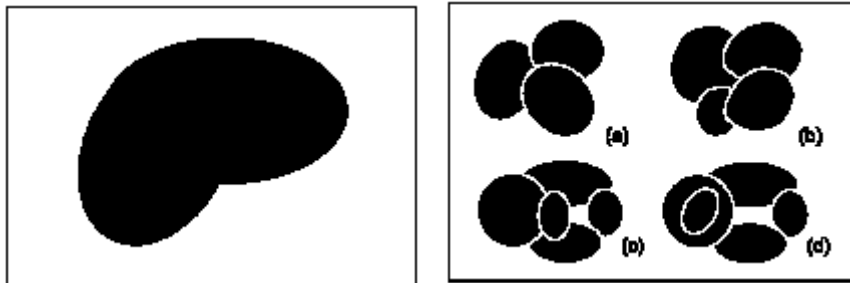
Determinación del número de q-blobs en la imagen

El cálculo del número de q-blobs se reduce a aplicar el corolario número 1.

Casos patológicos.

Caso 1. Este caso se presenta cuando al traslaparse dos o más q-blobs, algunos de los puntos de concavidad singulares, no puede ser detectados, debido a las curvaturas de dos q-blobs coinciden.

Caso 2. Este caso se presenta cuando tres o más q-blobs al traslaparse no permiten que el hoyo u hoyos correspondientes se formen.



Caso 1

Caso 2

Ejemplo:

En este caso, la imagen de prueba de la figura 2.3.2.(a) contiene 11, conglomerados, 3 hoyos y 64 puntos de concavidad singulares. Al ser procesada a través del sistema, nos indica que en la imagen hay 40 q-blobs, resultado acertado.

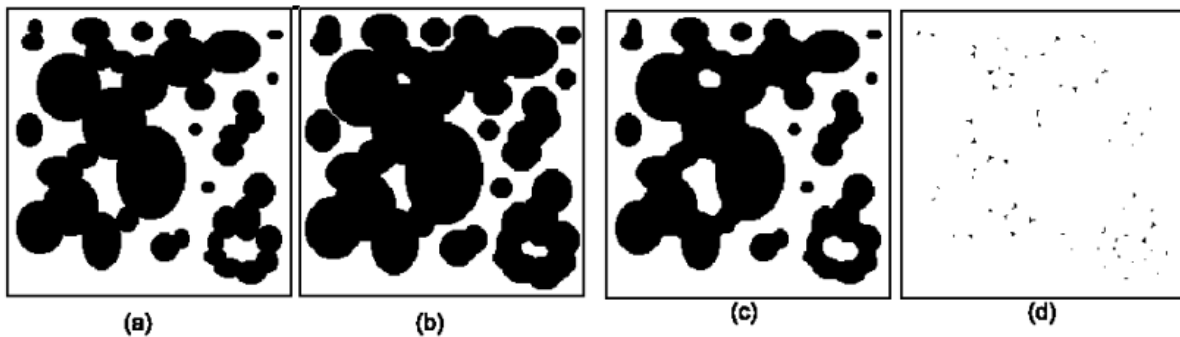


Figura 2.3.2 (a) imagen binaria de prueba, (b) imagen dilatada, (c) imagen erosionada, (d) imagen de concavidades.

Comentarios

Los resultados obtenidos son muy buenos, solamente falla cuando ocurren casos patológicos o no se determina bien el elemento estructural a utilizar.

Capítulo 3

ADQUISICIÓN DE IMÁGENES

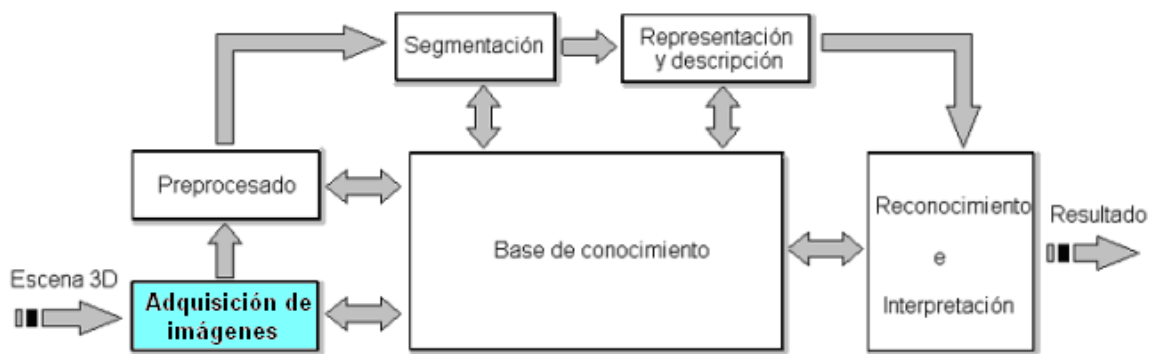


Figura 3.1.0: Etapas fundamentales en el reconocimiento automático de objetivos

En la Figura 3.1.0 se resalta el proceso de Adquisición de imágenes. Este proceso es de gran importancia y determinante para el éxito o fracaso de nuestro sistema de reconocimiento. Se dice que la captación de la imagen es el 50% del sistema y sería un error muy serio pensar que se puede compensar una imagen inadecuada con algún algoritmo; es mejor tener una buena calidad de imagen y utilizar menos algoritmos para su mejoramiento.

El ojo humano tiene casi el mismo funcionamiento que una cámara de vídeo pero no se compara en cuanto a su definición de imágenes y al tiempo de procesamiento, ya que el ojo humano es uno de los órganos más especializados. Por todo esto, en este capítulo se menciona la anatomía y el funcionamiento del ojo humano, así como el funcionamiento de la cámara de vídeo, aspectos de iluminación y algunos conceptos importantes.

3.1. Elementos de percepción visual

Estructura del ojo humano

El ojo humano tiene la capacidad para captar las imágenes con gran nitidez, el movimiento, la profundidad, el color, enfoca a cualquier objeto en diferentes distancias, entre otros. Es casi esférico con un diámetro de 20 mm. Es capaz de adaptarse a distintos niveles de iluminación gracias a que el diafragma formado por el iris que puede cambiar de diámetro, proporcionando un agujero central (la pupila) que varía entre 2 mm para iluminación intensa y 8 mm para situaciones de poca iluminación.

La luz entra en el ojo a través de la córnea (tejido resistente y transparente que cubre la superficie anterior del ojo) y el iris, atravesando la lente del cristalino hasta llegar a la parte trasera de la esfera ocular llamada retina que está recubierta por una capa de células fotosensibles la cual recibe una pequeña imagen invertida de ese mundo exterior. La lente del cristalino altera su forma para enfocar la imagen. En la figura 3.1.1 se muestran los principales elementos que conforman el ojo humano.

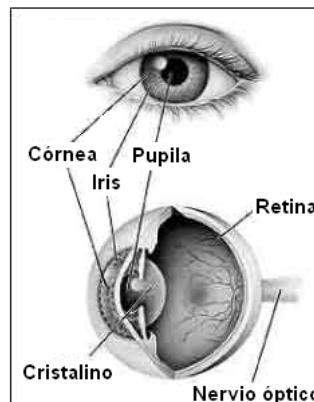


Figura 3.1.1: Anatomía del ojo humano

Para enfocar objetos distantes, los músculos de control hacen que el cristalino se aplane relativamente. De forma similar, estos músculos hacen el cristalino más grueso cuando se enfocan objetos cercanos al ojo.

La distancia entre el punto focal del cristalino y la retina varía entre 17 m y 14m cuando el poder refractivo del cristalino aumenta desde su mínimo hasta su máximo. Cuando el ojo enfoca a un objeto distante a más de 3 m, el cristalino se coloca en su posición de menor poder refractivo, y cuando el ojo enfoca a un objeto más cercano el cristalino es más fuertemente refractivo. En la figura 3.1.2, el observador esta mirando una palmera de 15m de altura que se encuentra a 100m de distancia, si x es el tamaño en milímetros de la imagen en la retina, para determinar el valor de x. Por geometría implica que $15/100 = x/17$, lo que da $x = 2.25\text{m}$

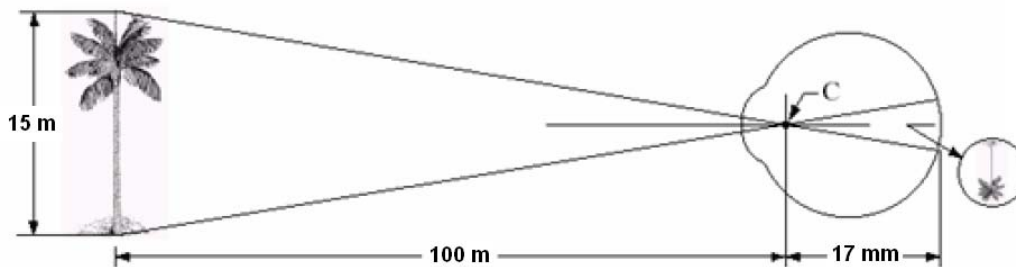


Figura 3.1.2: Formación de imágenes en el ojo humano

En la retina existen dos tipos de receptores: los conos y los bastones (ver figura 3.1.3). En cada ojo existen entre 6 y 7 millones de conos localizados principalmente en la región central de la retina, denominada fovea, y que son muy sensibles al color. La visión mediante los conos se denomina fotópica o visión de luz brillante. El número de bastones es de 75 a 150 millones que se encuentran distribuidos sobre la superficie de la retina. Los bastones no están implicados en la visión en color y son sensibles a niveles de iluminación bajos. Por ejemplo, los objetos que aparecen con colores brillantes bajo la luz del día, cuando se ven en la luz de la luna aparecen como formas sin color puesto que sólo se estimulan los bastones. Este fenómeno se conoce como visión tenue o visión escotópica.

La figura 3.1.3 también nos muestra la densidad de bastones y conos para una sección transversal del ojo derecho cruzando la región por donde emerge el nervio óptico. La ausencia de receptores en esta zona produce lo que se denomina punto ciego. Normalmente no percibimos el punto ciego ya que al ver un objeto con ambos ojos la parte del mismo que incide sobre el punto ciego de uno de ellos, incide sobre una zona sensible del otro. Si cerramos un ojo tampoco seremos conscientes de la existencia del punto ciego debido a que el cerebro normalmente nos engaña y completa la parte que falta de la imagen.

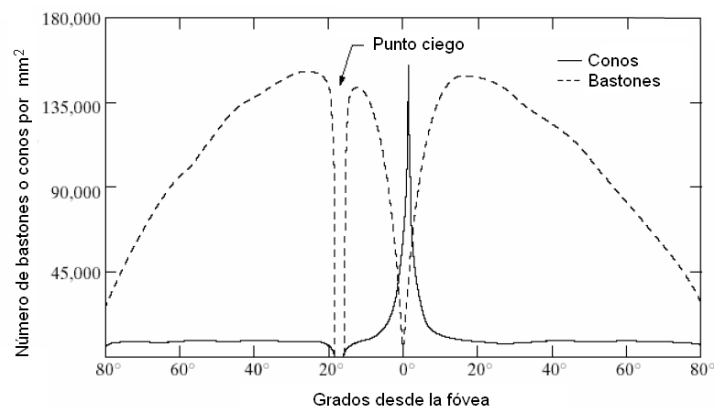


Figura 3.1.3: Distribución de conos y bastones en la retina

Percepción de color

Como se ha mencionado el ser humano percibe el color mediante los sensores (conos) que traducen la energía lumínica en señales nerviosas. Estos se pueden dividir en tres clases, dependiendo de la banda de longitudes de onda a la cual son más sensibles. Los sensores tipo alfa tienen una mayor sensibilidad a 480 nanómetros (azul), los tipo beta a 540 nm (verde) y los tipo gama a 570 nm (rojo) como se aprecia en la figura 3.1.4.

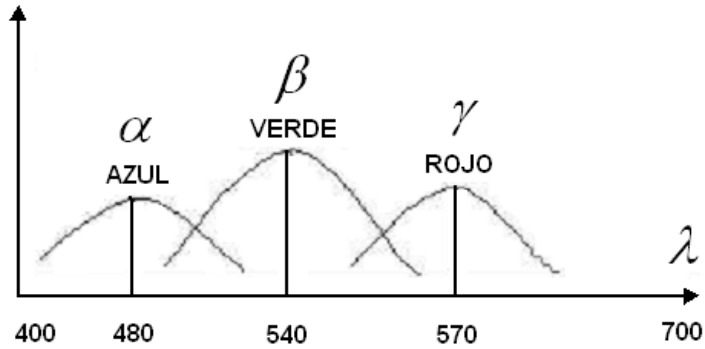


Figura 3.1.4: Respuesta del ojo humano a diferentes longitudes de onda.

La identificación del color en la imagen se hace mediante la combinación de estas tres señales, de donde se perciben la gran variedad de colores que podemos distinguir. A estos se les denomina colores primarios (rojo, verde y azul). De la combinación aditiva en pares iguales de éstos, obtenemos los colores secundarios (amarillo, magenta y cian); y de la combinación de los tres obtenemos el blanco. Al combinar los secundarios sustractivamente obtenemos los colores primarios y el negro (figuras 3.1.5 y 3.1.6).



Figura 3.1.5: Mezclas de luz (adición de primarios)



Figura 3.1.6: Mezclas de pigmentos (sustracción de secundarios).

3.2 Captación de la imagen por la cámara

La captación de la imagen se realiza por medio de sensores que se encuentran dentro de la cámara. Estos sensores son componentes sensibles a la luz que modifican su señal eléctrica en función de la intensidad luminosa que perciben.

La tecnología más habitual en este tipo de sensores es el CCD (dispositivos de acoplamiento de carga) donde se integra en un mismo chip los elementos fotosensibles y el conjunto de puertas lógicas y circuitería de control asociada (ver figura 3.2.1). En éstos dispositivos, la señal eléctrica que transmiten los fotodiodos es función de la intensidad luminosa que reciben, su espectro, y el tiempo de integración.

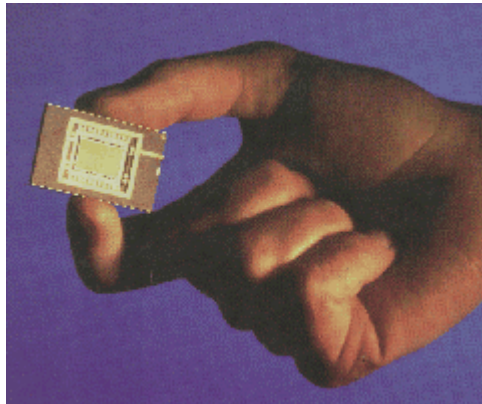


Figura 3.2.1: Dispositivo de acoplamiento de carga

Otra tecnología que está empezando a extenderse son los sensores CMOS (complementary metal oxide semiconductor) dada las ventajas de éstos sobre los CCD, y la reducción de precios de estos dispositivos. En cuanto al rango dinámico se pasa de los 70 decibeles de los sensores CCD a los 120dB de los sensores CMOS, valor más cercano a los 200dB del ojo humano, lo que facilita la autoadaptación en el propio chip al brillo existente en el entorno.

Existen diferentes arquitecturas de sensores. En primer lugar están los sensores lineales. En éstos, el sensor es una línea de fotodiodos. Esta arquitectura permite la utilización de sensores de 1×1024 , 1×2048 , 1×4096 , e incluso 1×6000 píxeles, lo que la hace muy adecuada para trabajar con altas resoluciones sobre superficies en movimiento. En segundo lugar están los sensores de área. Estos alcanzan resoluciones habituales de 1024×1024 . En este caso existen tecnologías de adquisición de imágenes, entrelazada y no entrelazada. El método entrelazado captura las líneas pares e impares que forman una imagen en instantes de tiempo diferentes. La tecnología de no entrelazado captura todas las líneas en el mismo instante de tiempo. Es más costoso económicamente, pero indispensable para trabajar con objetos en movimiento.

3.3 Espacios de color

Modelo RGB

El modelo RGB (Red, Green, Blue), maneja 3 canales, uno para el rojo, uno para el verde y uno para el color azul y la imagen resultante tiene tonalidades que surgen de las combinaciones entre los colores primarios. En esta codificación por cada píxel se necesitan 24 bits, es decir 3 bytes, codificando cada color primario con 1 byte (8 bits) que pueden tomar 256 valores con un rango de 0 a 255. Los píxeles RGB son tridimensionales, $\text{pixel}_{\text{RGB}} = f(R,G,B)$, donde el valor de f es la intensidad del color rojo, verde y azul en las coordenadas (x,y) del $\text{pixel}_{\text{RGB}}$. De esta forma los colores se representan en coordenadas cartesianas dentro de un cubo unitario (figura 3.3.1).

Cada color se representa como un vector del origen y la diagonal principal corresponde a la escala de grises. En este modelo se basan las cámaras y receptores de televisión. Sin embargo, no es recomendable aplicarlo al procesamiento de imágenes y visión.

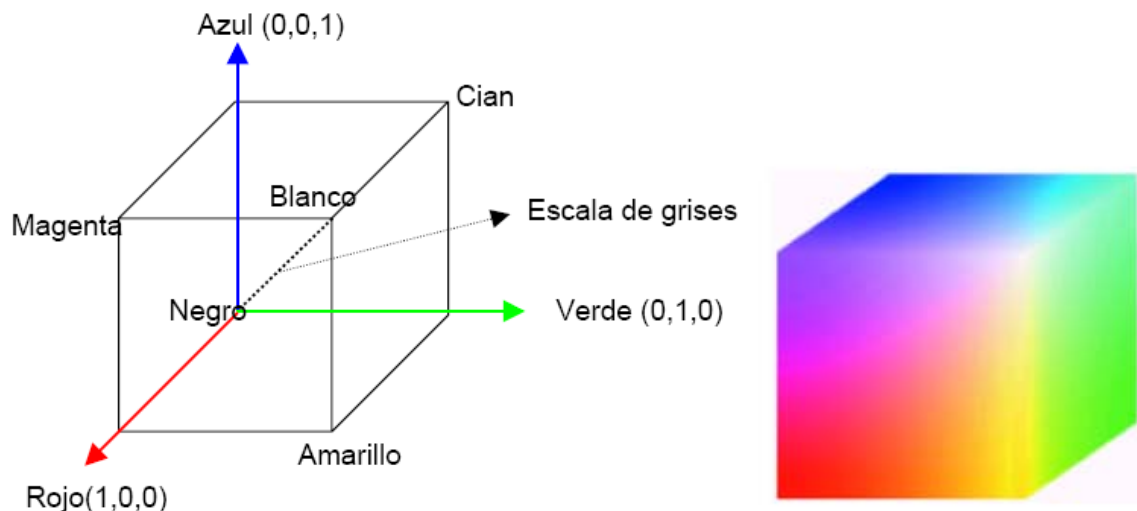


Figura 3.3.1: Distribución de colores en el cubo RGB

Modelo YCbCr

En el modelo YCbCr el color es representado por la luminancia (Y) y por dos valores diferentes de color (Cb y Cr) que son características colorimétricas del color. La luminancia es la cantidad lineal de luz que puede ser calculada como la suma ponderada de los componentes lineales del espacio de color RGB (Figura 3.3.2).

La obtención de este espacio de color a partir del RGB es la siguiente:

$$\begin{aligned} Y &= 0.2989 * R + 0.5866 * G + 0.1145 * B \\ C_b &= -0.1688 * R - 0.3312 * G + 0.5 * B \\ C_r &= 0.5 * R - 0.4184 * G - 0.08116 * B \end{aligned} \quad (3.3.1)$$

siendo R, G y B son los valores del canal rojo, verde y azul respectivamente.

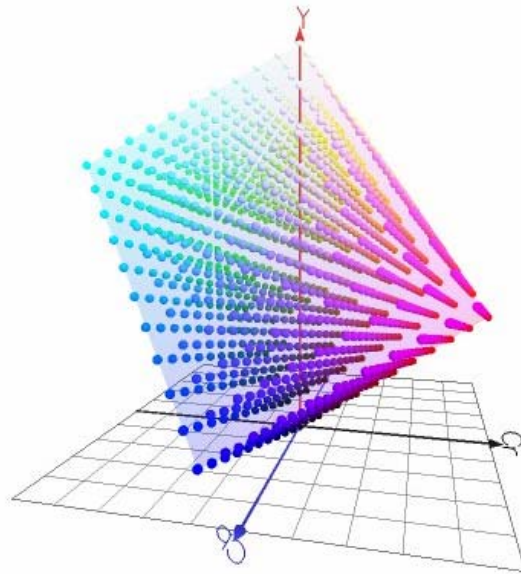


Figura 3.3.2: Distribución de color YCbCr

Modelo XYZ

Es el primer espacio de color estandarizado en 1931. Estas componentes no son reales, sino imaginarias, pero cualquier color se puede definir como combinación de ellas. De esta forma, toda la información de intensidad esta comprendida únicamente en la componente Y.

Estos valores se pueden obtener de las componentes RGB como:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0.607 & 0.174 & 0.200 \\ 0.299 & 0.587 & 0.114 \\ 0.000 & 0.066 & 1.116 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} \quad (3.3.2)$$

En la figura 3.3.3. se muestra como los colores se distribuyen en la curva en función de su longitud de onda. Se puede observar que con este espacio de color no es fácil la separación de colores.

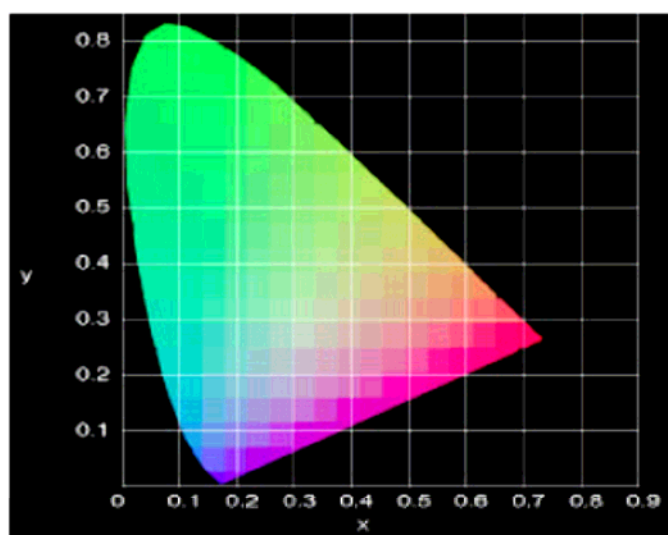


Figura 3.3.3: Distribución del color en el modelo XYZ

Modelo CIE(Lab)

Una vez que se dispone de las coordenadas XYZ, puede optarse por trabajar en distintos espacios CIE. Entre ellos, el CIE(Lab) es muy popular por tratarse de un espacio uniforme y adecuado para la segmentación. La mayor ventaja de estos sistemas es que el parecido entre dos colores se puede medir por distancia euclídeana.(Figura 3.3.4).

Para obtener los valores Lab de una imagen color, pueden usarse las ecuaciones:

$$L = 25 \left(100 \left(\frac{Y}{Y_0} \right)^{1/3} \right) - 16 \quad (3.3.3)$$

$$a = 500 \left[\left(\frac{X}{X_0} \right)^{1/3} - \left(\frac{Y}{Y_0} \right)^{1/3} \right] \quad (3.3.4)$$

$$b = 200 \left[\left(\frac{Y}{Y_0} \right)^{1/3} - \left(\frac{Z}{Z_0} \right)^{1/3} \right] \quad (3.3.5)$$

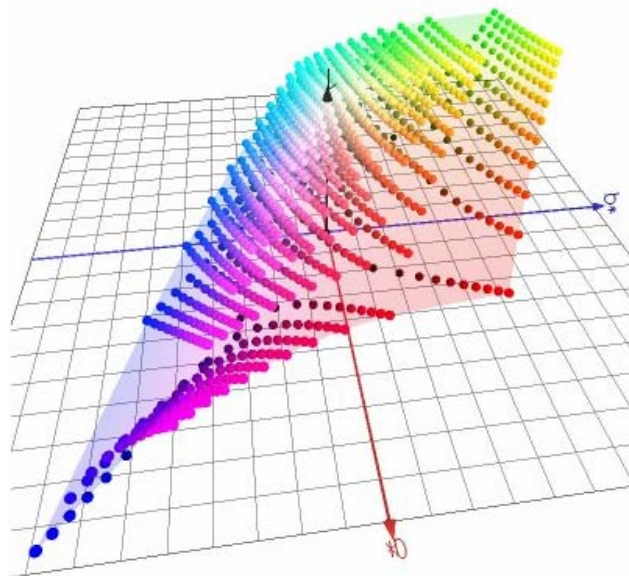


Figura 3.3.4: Espacio de color CIELAB

Modelo HSI

En este modelo los colores se representan mediante sus propiedades: Tono, Saturación e Intensidad (*HSI: Hue, Saturation, Intensity*), que describen el brillo, color y pureza del color. El tono equivale al color que nosotros apreciamos. La saturación se refiere a la pureza de dicho color (su grado de mezcla con los otros colores primarios). Por último, la intensidad puede identificarse como el brillo de la imagen.

Un valor de color es representado como un punto en el cilindro. El tono es el ángulo de rotación sobre la superficie circular del cilindro. El rojo es definido a cero grados. La saturación es representada por la longitud del vector desde el centro del círculo al valor del color en cuestión. Para los colores puros, este vector se extiende al borde de la superficie cilíndrica. La intensidad es representada por la longitud del vector desde la base del cilindro al valor del color (ver figura 3.3.5).

La conversión de la representación RGB a HSI puede ser llevada a cabo considerando las siguientes ecuaciones:

$$\theta = \cos^{-1} \left(\frac{1/2((R-G) + (R-B))}{(R-G)^2 + (R-B)(G-B)^{1/2}} \right) \quad (3.3.4)$$

$$H = \begin{cases} \theta, & \text{si } G \geq B \\ 2\pi - \theta & \text{en caso contrario} \end{cases} \quad (3.3.5)$$

$$S = 1 - \frac{\min(R, G, B)}{I} \quad (3.3.6)$$

$$I = \frac{R + G + B}{3} \quad (3.3.7)$$

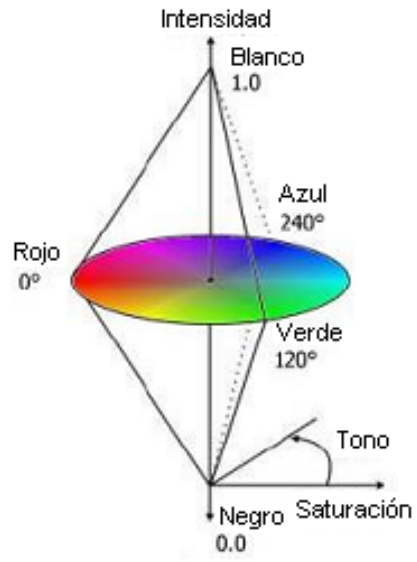


Figura 3.3.5 Espacio de color HSI

3.4. Iluminación

La iluminación es el aspecto más decisivo de cualquier aplicación de visión artificial, si el objeto tiene una buena iluminación se pueden lograr muy buenos resultados en nuestro sistema.. Muchas aplicaciones buenas han fallado por la falta de una iluminación apropiada. En un sistema de visión artificial, la mejor imagen es aquella que tiene mayor contraste, donde las áreas de interés se destacan del fondo (*background*) que no tiene mayor importancia.

La luz es reflejada de dos maneras llamadas reflexión especular y reflexión difusa. En la reflexión especular, cada rayo incidente se refleja en una única dirección como en el caso de algunos metales o el espejo que pueden saturar el sensor de la cámara, lo mejor es utilizar alguna técnica de iluminación especial. Las reflexiones difusas son tenues pero estables, la intensidad de reflexión puede ser de 10 a 1000 veces menor que la fuente de luz (ver figuras 3.4.1 y 3.4.2).

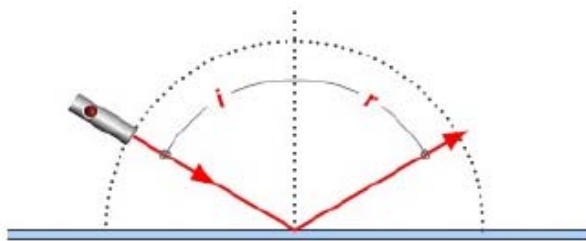


Figura 3.4.1: Reflexión Especular

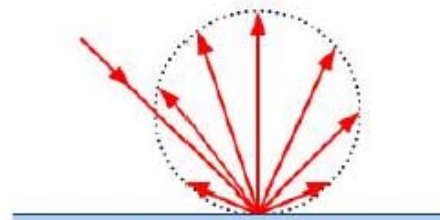


Figura 3.4.2: Reflexión difusa

3.5. Resolución de la imagen

Para las imágenes digitales almacenadas como mapa de bits, la resolución de la imagen describe cuán nítida es una imagen de fotografía por medio de dos números enteros, donde el primero es la cantidad de columnas de píxeles (cuántos píxeles tiene la imagen a lo ancho) y el segundo es la cantidad de filas de píxeles (cuántos píxeles tiene la imagen a lo alto). En la figura 3.5.1 se presenta una ilustración sobre cómo se vería una imagen con diferente resolución.

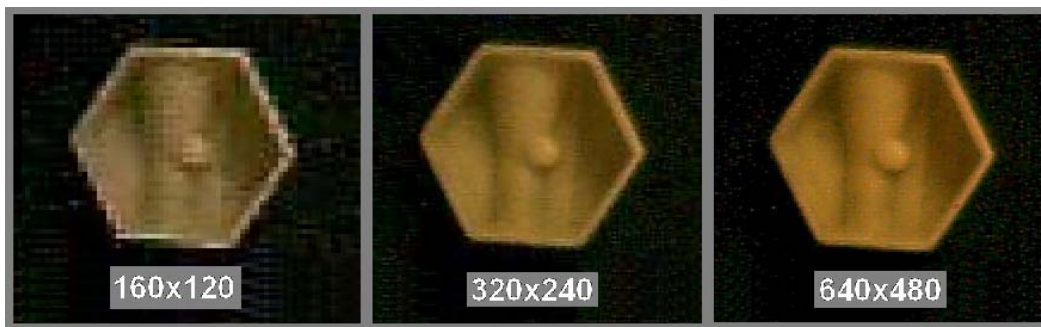


Figura 3.5.1: Diferentes tipos de resolución .

La imagen capturada a 160x200 píxeles pierde detalles del objeto a reconocer; por el contrario, la imagen capturada a 640x480 píxeles es muy nítida, pero con el inconveniente de que contiene mucho más información y su procesamiento es más lento.

3.6. Formatos de imágenes

Para almacenar imágenes en forma digital las compañías de diseño lo hacen de diferente manera (formato) de acuerdo a sus necesidades: calidad de la imagen, compresión, número de imágenes en un solo archivo, entre otras; es por esto que al momento de intentar abrir a un archivo de imagen, nos damos cuenta que no se puede leer o que la imagen es de mala resolución. Para almacenar fotografías con una buena calidad se utilizan los formatos .bmp y .tiff, pero tienen la desventaja de que ocupan 10 veces o más que un jpg. Ver figura 3.6.1

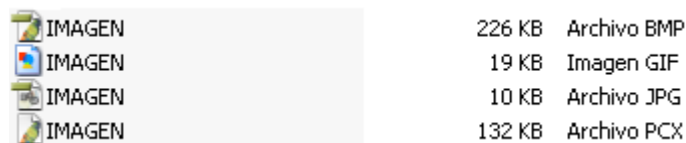


IMAGEN	226 KB	Archivo BMP
IMAGEN	19 KB	Imagen GIF
IMAGEN	10 KB	Archivo JPG
IMAGEN	132 KB	Archivo PCX

Figura 3.6.1: Tamaño de una imagen con diferentes tipos de formatos.

A continuación se muestran algunos de los formatos más comunes y las posibilidades que brinda cada uno:

BMP.- El formato BMP (Bit Map) es el formato de las imágenes en bitmap de Windows. Aunque muy extendido, tiene la dificultad de la escasa compresión que realiza en los archivos por lo que ocupan mucho espacio Pero el formato de Mapa de Bits tiene una importante característica a su favor, es que casi todos los usuarios tienen una PC que puede soportarlo mediante el programa Paint.

TIFF.- El formato TIFF, que corresponde a las siglas Tagged-Image File Format, se utiliza cuando se van a realizar impresiones en papel de la imagen. Es un formato que admite una compresión muy baja, por lo que la pérdida en la calidad de imagen es prácticamente nula. Se conoce como formato de compresión sin pérdida. Una desventaja es que los archivos .tiff son de gran tamaño.

GIF.- Estas tres letras, representan las siglas de "Graphics Interchange Format". La tecnología del GIF fue desarrollada por la empresa CompuServe, destacando en sus características la posibilidad de trabajar, con un máximo de 256 colores y la posibilidad de trabajar con mas de una imagen, el GIF es un formato ideal para utilizar en la web en imágenes pequeñas o de pocos colores y no es recomendable para utilizar en impresión, ya que la calidad es limitada.

JPEG (o JPG).- El formato JPEG(Joint Photographic Experts Group) está diseñado para realizar compresión de imágenes, permitiendo reducir la cantidad de información de las mismas, con una consecuente reducción de peso del archivo final. Es por ello que dicho formato también es de uso común en la web. La resolución de este formato es baja y no es recomendable para la impresión.

PCX.- Este es el formato desarrollado por Zsoft para su programa Paintbrush para PC. En un principio sólo guardaba 16 colores, pero las últimas actualizaciones acomodan el color de 8 y de 16 bits.

Capítulo 4

PREPROCESAMIENTO DE LA IMAGEN

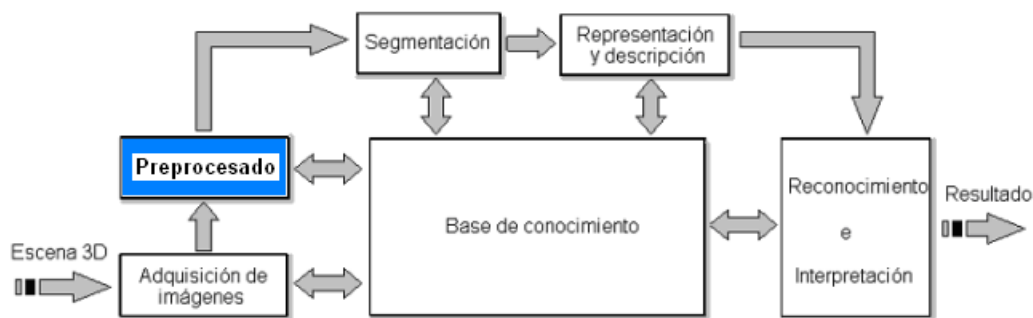


Figura 4.1.0: Etapas fundamentales en el reconocimiento automático de objetivos

En la Figura 4.1.0 se resalta el segundo paso para el reconocimiento automático de objetivos: el Preprocesado o procesamiento de la imagen.

Cuando se adquiere una imagen mediante cualquier sistema de captura, por lo general esta no se utiliza directamente por el sistema de visión. La aparición de variaciones en intensidad debidas al ruido, por deficiencias en la iluminación, o la obtención de imágenes de bajo contraste, hace necesario un preprocesamiento de la imagen con el objetivo fundamental de corregir estos problemas, además de aplicar aquellas transformaciones a la imagen que acentúen las características que se deseen extraer de las mismas, de manera que se facilite las operaciones de las etapas posteriores.

En el presente capítulo se mencionan las diferentes técnicas empleadas para el mejoramiento de la imagen y las transformaciones necesarias para la aplicación de estas técnicas.

4.1. Conversión de una imagen en color RGB a niveles de gris

En el proceso de captación de imagen, obtuvimos una imagen digital en el espacio de color RGB con sus tres matrices de intensidades para los colores rojo, verde y azul respectivamente. La figura 4.1.1 nos muestra las tres matrices correspondientes únicamente al recuadro de la imagen con intensidades para cada color que van del 0 al 255.

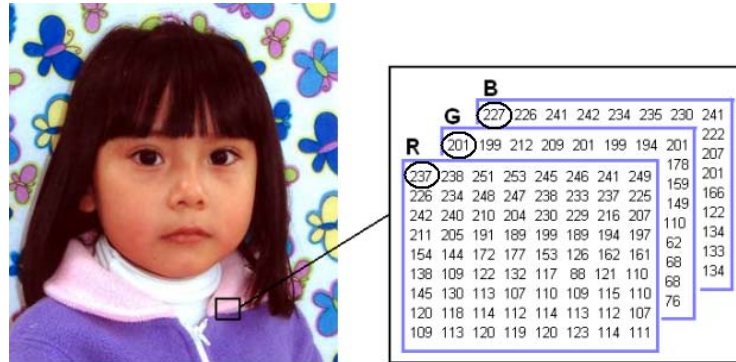


Figura 4.1.1: Representación de una imagen en color con sus tres matrices de intensidades.

El proceso de conversión a niveles de gris consiste en calcular el promedio de cada intensidad para las matrices de los colores rojo, verde y azul. Cada valor de la conversión se redondea para formar una matriz de intensidades donde el valor correspondiente a cada punto, indica el valor de intensidad que puede ir de [0 a 255]; el cero representa el negro absoluto y el 255 el blanco absoluto. La figura 4.1.2 se obtuvo como resultado de aplicar a la figura 4.1.1. la siguiente fórmula:

$$I = \text{Round} \left\{ \frac{1}{3} (R + G + B) \right\} \quad (4.1.1)$$

Por ejemplo, para el cálculo de los dos primeros elementos de la matriz en tonos de gris tenemos:

$$I_{11} = \text{Round} \left\{ \frac{1}{3} (237 + 201 + 227) \right\} = \text{Round} \{221.6\} = 222$$

$$I_{12} = \text{Round} \left\{ \frac{1}{3} (238 + 199 + 226) \right\} = \text{Round} \{221.0\} = 221$$

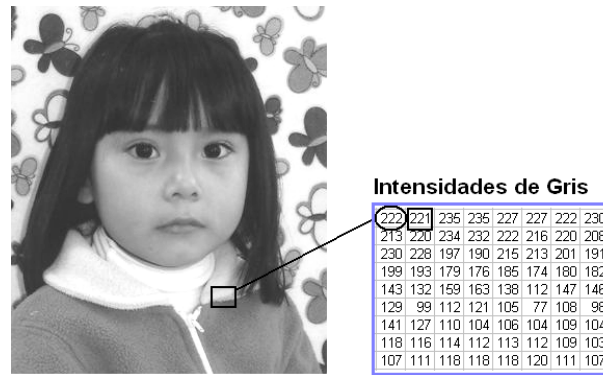


Figura 4.1.2: Imagen es escala de gris con su matriz de intensidades.

4.2. Conversión de una imagen en niveles de gris a blanco y negro (Binarización)

Con esta técnica, los valores de píxel en la imagen de entrada menores a un cierto umbral (entre 0 y 255) son convertidos a negro mientras que los píxeles con valores mayores al umbral son convertidos a blanco.

En la figura 4.2.1 se muestra la transformación de la imagen en escala de gris a blanco y negro. El valor de la intensidad mayor a un umbral (127) se les asigna el máximo valor de gris (255=blanco). Por su parte, los píxeles comprendidos por debajo de este valor toman el valor mínimo (0=negro).



Figura 4.2.1: imagen binarizada y su matriz de intensidades.

4.3. Mejoramiento de histograma

El histograma de una imagen es una función que representa el número de píxeles existentes para cada valor de nivel de gris. Para una imagen con 256 niveles de gris posibles, la abscisa del histograma varía entre 0 y 255 (figura 4.3.1). La probabilidad “ P_i ” de ocurrencia de un determinado nivel “ i ” se define como:

$$P_i = \frac{f_i}{N * M} \quad (4.3.1)$$

Donde “ f_i ” es el número de píxeles en el nivel de intensidad “ i ” y “ $N * M$ ” es la cuenta total de píxeles correspondientes a las filas y columnas de la imagen. Todos los valores de “ P_i ” son menores o iguales que “1” y la suma de todos los valores de “ P_i ” es 1.

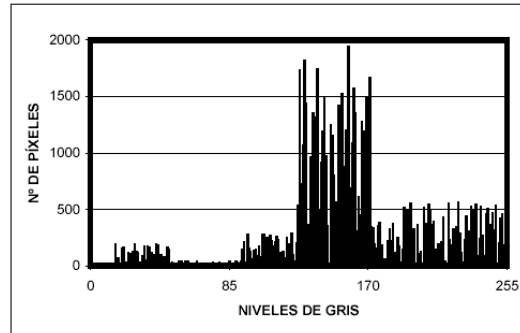


Figura 4.3.1: Histograma de una imagen en escala de gris

Un histograma con la gráfica hacia la izquierda resulta muy oscura, si está hacia la derecha contiene mucho brillo y si se encuentra al centro tiene poco contraste. La representación de un histograma ideal sería la de una recta horizontal, ya que eso nos indicaría que todos los posibles valores de grises están distribuidos de manera uniforme en nuestra imagen como se muestra en la figura 4.3.2.

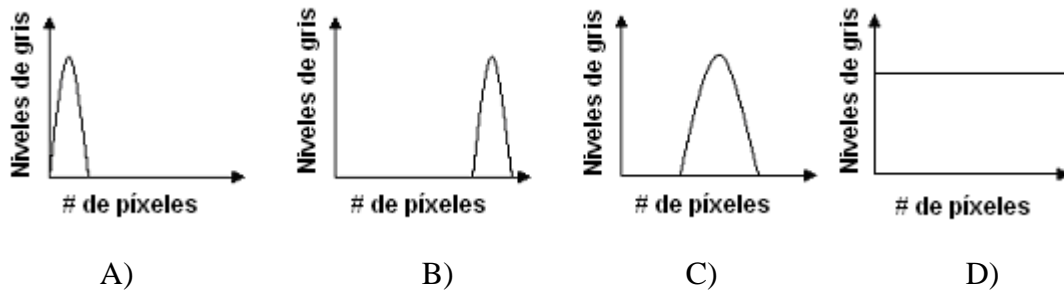


Figura 4.3.2: Diferentes tipos de histogramas: A) imagen muy oscura, B) imagen muy clara, C) imagen con poco contraste, D) histograma ideal.

Desplazamiento de histograma

El desplazamiento de histograma se usa para aclarar u oscurecer una imagen pero manteniendo la relación entre los valores de los niveles de gris. Esta operación puede llevarse a cabo por simple adición o sustracción de un número fijo a todos los niveles de gris: $g(x, y) = f(x, y) \pm \text{desplazamiento}$

Las figuras 4.3.5 y 4.3.6 muestran la imagen resultante y su respectivo histograma tras aplicar un desplazamiento del histograma 100 unidades a la derecha para aclarar la imagen de la figura 4.3.3.



Figura 4.3.3: Imagen oscura

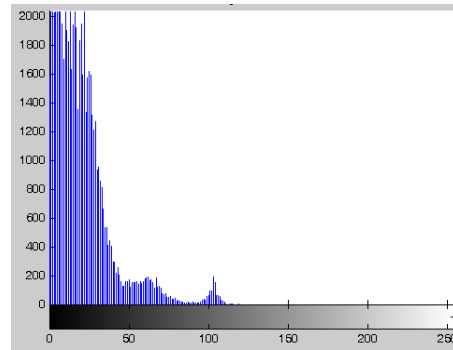


Figura 4.3.4: Histograma de la imagen oscura.

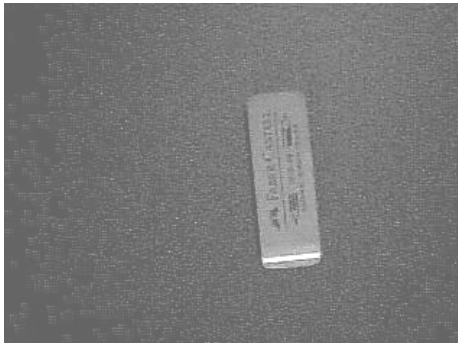


Figura 4.3.5: Imagen con desplazamiento

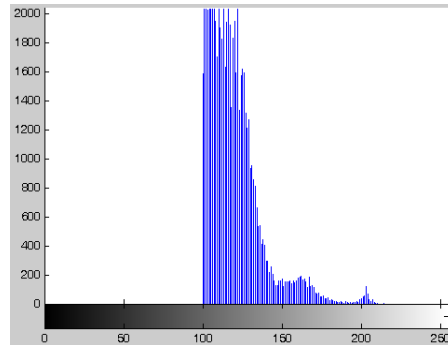


Figura 4.3.6: Histograma con desplazamiento.

Ampliación de histograma

Es una de las técnicas más utilizadas para la mejora del contraste de la imagen mediante una determinada transformación o modificación del histograma. Se trata de encontrar una función $F(g)$ que realce el contraste de la imagen original expandiendo la distribución de los niveles de gris. Dicha expansión debe ser lo más suave posible en el sentido de que idealmente debería haber el mismo número de píxeles por niveles de gris.

Como ejemplo utilizaremos la siguiente función analítica,

$$F(g) = \max \left\{ 0, \text{round} \left(\frac{N_g \times F_a}{N \times M} \right) - 1 \right\} \quad (4.3.2)$$

donde “ F_a ” es el número total de píxeles en el nivel g más los inferiores, es decir es la frecuencia acumulativa para los niveles de g e inferiores. Utilizando esta técnica en la tabla 4.3.1. Si consideramos una imagen de 70 píxeles ($N \times M=70$) y 16 niveles de intensidad ($N_g=16$) con valores en el rango $[0...15]$, “ g ” son los niveles de gris, “ F ” es la frecuencia de aparición de dichos niveles de gris, F_a es la frecuencia acumulativa de los niveles de gris y $F(g)$ es el resultado de la transformación final. En las figuras 4.3.7 y 4.3.8 se muestra el histograma correspondiente a la tabla 4.1 antes y después de la transformación.

G	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
F	3	5	10	12	12	9	7	2	6	4	0	0	0	0	0	0
F_a	3	8	18	30	42	51	58	60	66	70	70	70	70	70	70	70
$F(g)$	0	1	3	6	9	11	12	13	14	15	15	15	15	15	15	15

Tabla 4.3.1: Distribución de los niveles de gris antes y después de la igualación

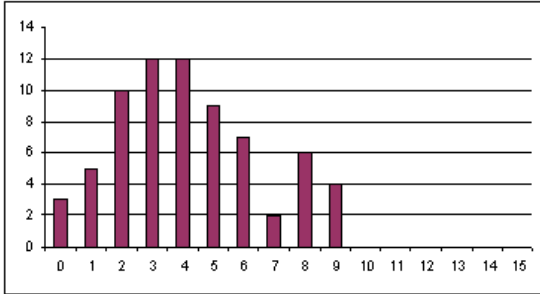


Figura 4.3.7: Histograma correspondiente a la tabla 4.3.1 antes de la ampliación.

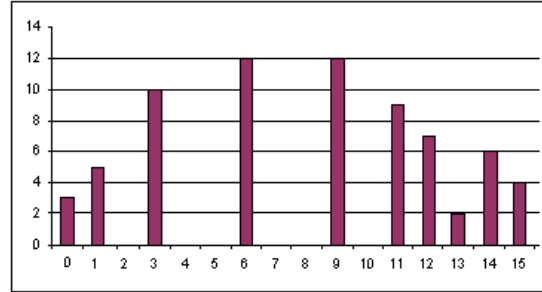


Figura 4.3.8: Histograma obtenido después de la ampliación

En las figuras 4.3.9 y 4.3.10 se implementó este método de ampliación de histograma para mejorar el contraste de la imagen de la figura 4.3.3. Como nos damos cuenta, se resalta el objeto a reconocer, pero también resalta el ruido producido por reflejos de luz y por el sistema de captación de la imagen. Es muy importante que posteriormente se realice una buen filtrado y una adecuada segmentación para que los resultados en el sistema de reconocimiento sean los adecuados.

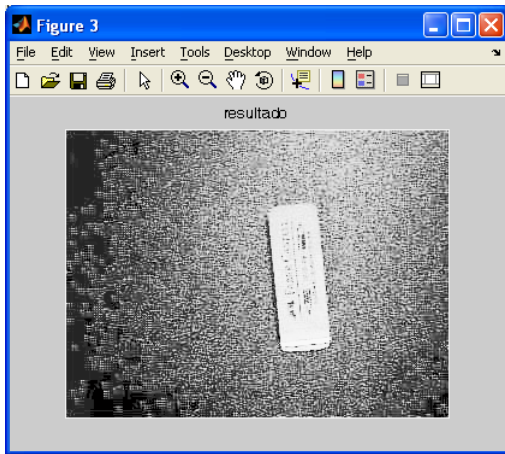


Figura 4.3.9: Imagen obtenida por el método de ampliación de histograma.

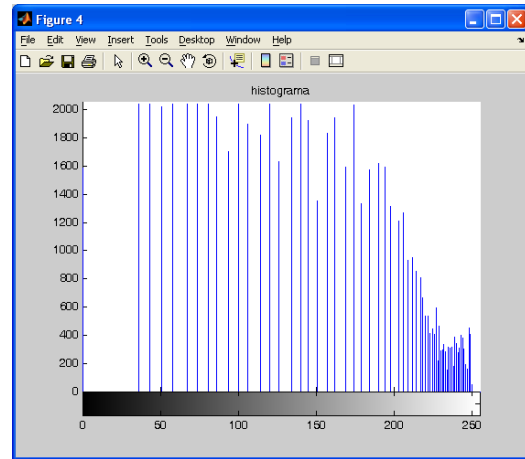


Figura 4.3.10: Histograma correspondiente a la ampliación de histograma

4.4. Uso de filtros para el mejoramiento de la imagen

En este proceso se trata de reducir al máximo el ruido contenido en la imagen producido por el tipo de iluminación, las sombras entre los objetos, pequeños puntos no deseados dentro de la imagen y otros efectos que pueden estar presentes en una imagen digital como resultado del muestreo, cuantización, transmisión o por perturbaciones en el sistema como pueden ser las partículas de polvo en el sistema óptico. Para la eliminación de todo este tipo de ruido se emplean diferentes tipos de filtros.

El filtrar una imagen (f) consiste en aplicar una transformación (T) para obtener una nueva imagen (g) de forma que ciertas características son acentuadas o disminuidas:

$$g(x, y) = T[f(x, y)] \quad (4.4.1)$$

Se puede considerar que la señal (imagen) pasa a través de una caja (filtro) cuya salida es la imagen filtrada (figura 4.4.1).

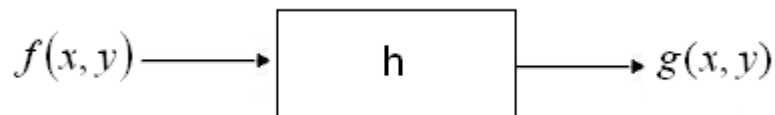


Figura 4.4.1: Proceso de filtrado

De acuerdo a la teoría de sistemas, al pasar una señal por un sistema lineal, la salida es la convolución de la transformación del sistema (función de transferencia) con la señal de entrada:

$$g(x, y) = h(x, y) * f(x, y) \quad (4.4.2)$$

Filtrado en el dominio espacial

Las técnicas de filtrado en el dominio espacial operan directamente sobre los píxeles de la imagen mediante una máscara cuadrada o rectangular. Una máscara es una pequeña imagen que consiste en una serie de valores predeterminados para cada posición (ver figura 4.4.2). La máscara se centra sobre el píxel de interés de forma que el nuevo valor del píxel depende de los píxeles que cubren la máscara.

$w_{1,1}$	$w_{2,1}$	$w_{3,1}$
$w_{2,1}$	$w_{2,2}$	$w_{3,2}$
$w_{3,1}$	$w_{2,3}$	$w_{3,3}$

Figura 4.4.2: Ejemplo de máscara de 3x3

A cada celda de la máscara le corresponde un peso o coeficiente (w), de forma que el nuevo valor del píxel es la sumatoria de el producto de los píxeles vecinos con el peso correspondiente:

$$g(x, y) = \sum_x \sum_y f(x, y)w(x, y) \quad (4.4.3)$$

Filtro promedio aritmético

Sea w_{xy} un conjunto de coordenadas en una subimagen cuadrada (*ventana*) de tamaño n^2 centrada en el punto $p(x, y)$, como se muestra en la figura 4.4.3. El filtro promedio aritmético calcula el valor promedio de la imagen con ruido en el área w_{xy} . El valor de la imagen restaurada \hat{g} en el punto $p(x, y)$ es el promedio aritmético calculado en esa vecindad.

$$\hat{g}(x, y) = \frac{1}{n^2} \sum_{(x, y) \in w_{xy}} f(x, y) \quad (4.4.4)$$

$x-1,y-1$	$x,y-1$	$x+1,y-1$
$x-1,y$	x,y	$x+1,y$
$x-1,y+1$	$x,y+1$	$x+1,y+1$

Figura 4.4.3: conjunto de coordenadas de una mascara de 3x3 (w_{xy}) para el punto $p(x,y)$

El valor de la intensidad en el punto (x,y) será el resultado de la ecuación anterior haciendo un barrido sobre toda la imagen original iniciando en el segundo renglón y segunda columna y terminando en el renglón $n-1$ y columna $n-1$ aplicando la siguiente mascara:

$$p = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad (4.4.5)$$

Por ejemplo, si tenemos la siguiente mascara:

$$p = \frac{1}{9} \begin{bmatrix} 39 & 46 & 41 \\ 10 & 33 & 49 \\ 72 & 77 & 36 \end{bmatrix}$$

el valor del punto (x,y) de la imagen restaurada (\hat{g}) es igual a la suma de los 9 valores de la matriz divididos por 9, el resultado correspondiente haciendo el redondeo al entero más próximo para el punto "p" será de 45. Como se puede observar en las figuras 4.4.4 y 4.4.5, el filtro promedio tiene el inconveniente de desdibujar bordes y formas; pero los puntos oscuros o con mucho brillo se atenúan.

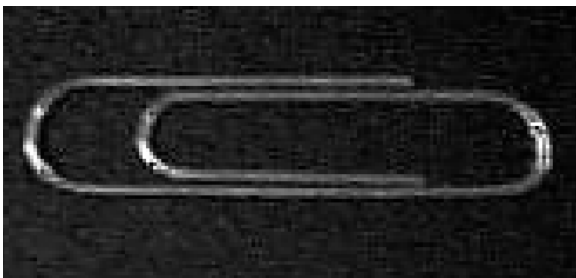


Figura 4.4.4: imagen en tonos de gris.

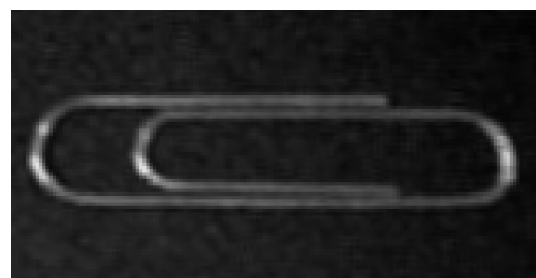


Figura 4.4.5: imagen resultado al aplicar un filtro promedio

Filtro máximo, mínimo y mediana

El filtro Máximo consiste en asignar al píxel de referencia el valor más alto de los encontrados en la máscara, por lo que su efecto es ensanchar las zonas claras y adelgazar las oscuras. El filtro Mínimo hace justo lo contrario, seleccionando el valor más bajo. En el Filtro Mediana se escoge el píxel de en medio del vector (ordenado previamente de menor a mayor). En las figuras 4.4.6, 4.4.7. y 4.4.8 se muestran los resultados obtenidos al aplicar estos tipos de filtros a la imagen de la figura 4.5.4. Sus ecuaciones son las siguientes:

$$p_{\text{max}} = \max\{f_1, f_2, \dots, f_n\} \quad (4.4.6)$$

$$p_{\text{min}} = \min\{f_1, f_2, \dots, f_n\} \quad (4.4.7)$$

$$p_{\text{med}} = \text{med}\{f_1 \leq f_2 \leq f_3 \leq \dots f_n\} \quad (4.4.8)$$

En la ecuación 1 y ecuación 2 se colocan en un vector los nueve valores y el valor del punto(x,y) de la imagen restaurada \hat{f} será el máximo o el mínimo de estos valores, en la ecuación (3) la mediana se escoge del píxel de en medio. Ejemplos:

$$p_{\text{max}} = \max\{39,46,41,10,33,49,72,77,36\} = 77$$

$$p_{\text{min}} = \min\{39,46,41,10,33,49,72,77,36\} = 10$$

$$p_{\text{med}} = \text{med}\{10,33,36,39,41,46,49,72,77\} = 41$$

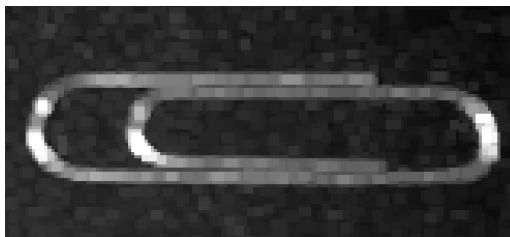


Figura 4.4.6: filtro "punto máximo"

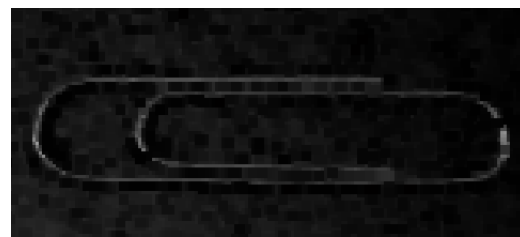


Figura 4.4.7: filtro "punto mínimo"

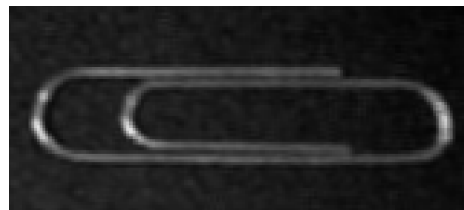


Figura 4.4.8: filtro "mediana"

4.5. Suavizado binario

Al binarizar una imagen se producen pequeños huecos, esquinas perdidas, puntos aislados y contornos irregulares (ruido). Para reducir este tipo de ruido se utilizó una expresión booleana específica sobre un entorno de vecindad centrado sobre un píxel “p” y dependiendo de la configuración espacial y los valores binarios de sus vecinos se le asigna al píxel “p” el valor de “0” o un “1”, en este caso se utiliza la siguiente la mascara de 3x3 de la figura 4.5.1 y para simplificación se utiliza la mascara de la figura 4.5.2, donde se asigna una letra a cada píxel de la mascara.

x-1,y-1	x,y-1	x+1,y-1
x-1,y	x,y	x+1,y
x-1,y+1	x,y+1	x+1,y+1

Figura 4.5.1: Coordenadas del punto p(x,y) y sus vecinos

<i>a</i>	<i>b</i>	<i>c</i>
<i>d</i>	<i>p</i>	<i>e</i>
<i>f</i>	<i>g</i>	<i>h</i>

Figura 4.5.2: Asignación de letras a cada vecino del punto “p”

Para rellenar los pequeños huecos de un píxel en zonas oscuras y eliminar los pequeños cortes y muescas en segmentos de lados rectos se utiliza la siguiente expresión booleana:

$$x1 = p + b * g * (d + e) + d * e * (b + g) \quad (4.5.1)$$

donde el signo “+” y el “*” representan las funciones lógicas AND y OR respectivamente. Si x1 toma el valor de 1, asignamos un “1” a p y en caso contrario este píxel tomará el valor de “0”. El resultado de la ecuación anterior se almacena en una nueva imagen para que el resultado de las operaciones no afecte a la imagen original (figura 4.5.3).

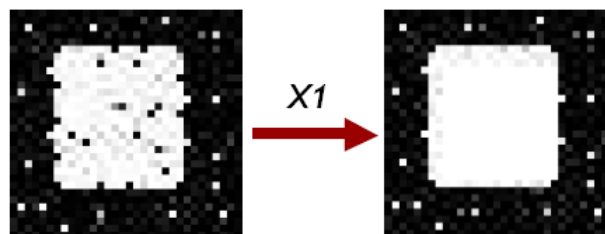


Figura 4.5.3: Eliminación de pequeños huecos.

Si queremos eliminar los unos aislados y las pequeñas protuberancias a lo largo de los segmentos de lados rectos (ver figura 4.5.4), se utiliza la siguiente expresión booleana:

$$x2 = p * [(a + b + d) * (e + g + h) + (b + c + e) * (d + f + g)] \quad (4.5.2)$$

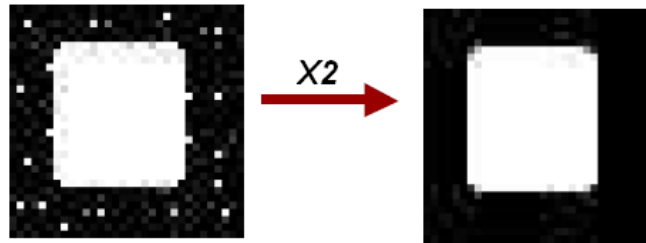


Figura 4.5.4: Eliminación de pequeñas protuberancias.

4.6. Operaciones lógicas sobre imágenes en blanco y negro

Las operaciones lógicas han sido ampliamente usadas en visión por computadora para la detección de rasgos y análisis de imágenes. Las principales operaciones lógicas entre imágenes son AND, OR y NOT. Las operaciones se realizan entre píxeles de imágenes binarias. De esta forma, si p1 y p2 son respectivamente los píxeles de dos imágenes en cuestión, entonces:

P1	P2	P1 AND P2	P1 OR P2
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Tabla 4.6.1: Operaciones binarias AND y OR.

Como se aprecia en la tabla anterior, el resultado de la operación AND es “1” cuando los dos píxeles de las imágenes en cuestión son “1” y en la operación OR basta con que alguno de los dos píxeles sea “1” para que el resultado sea “1”. La operación NOT se lleva a cabo con los píxeles de una sola imagen y el resultado es el complemento o la negación del píxel en cuestión:

P1	NOT P1
0	1
1	0

Tabla 4.6.2: Operación NOT.

Existen mas operaciones que se derivan de estas operaciones básicas, por ejemplo: NAND, NOR y XOR. La figuras 4.6.1 ejemplifica las operaciones básicas aplicadas a imágenes binarias.

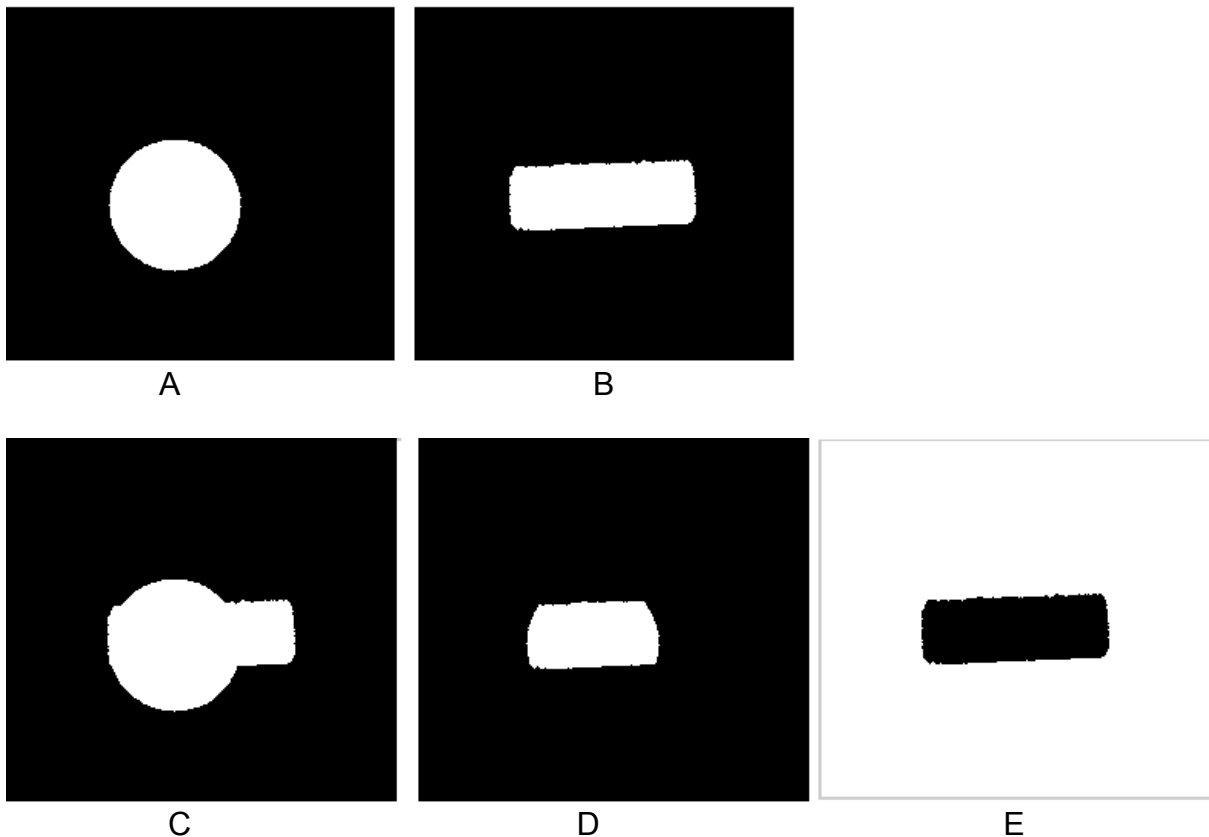


Figura 4.6.1 Aplicación de operaciones binarias básicas: A) imagen “A”, B) imagen “B”, C) imagen C = A or B, D) D = A and B, E) E = not B.

4.7. Operaciones aritméticas sobre imágenes en tonos de gris

Las operaciones aritméticas han sido ampliamente utilizadas en el procesamiento digital de imágenes. Las más comunes son:

Operación	Fórmula	
Suma	$(p1 + p2) / 2$	(4.7.1)
Resta	$abs(p1 - p2)$	(4.7.2)
Multiplicación	$p1 * p2$	(4.7.3)
división	$p1 / p2.$	(4.7.4)

Estas operaciones se realizan entre píxeles de imágenes distintas. De esta manera, $p1$ pertenece a una de las dos imágenes y $p2$ a la otra.

La operación de suma se utiliza para reducir el ruido en una imagen, también se utiliza para combinar dos o más imágenes. La resta de imágenes a sido usada extensamente para la detección de movimiento entre imágenes captadas en instantes de tiempo sucesivos, y para remover información estática del fondo. La multiplicación es utilizada para aislar regiones de interés en la imagen, multiplicando una imagen en tonos de gris por una máscara binaria. La División al igual que la multiplicación se aplican principalmente para ajustar el brillo en una imagen o para corregir sombras causadas por iluminación no uniforme durante el proceso de adquisición. En la figuras 4.7.1, 4.7.2 y 4.7.3 se muestran algunos ejemplos de operaciones aritméticas con imágenes.



Figura 4.7.1. Efecto de aplicación de suma para combinar dos imágenes.

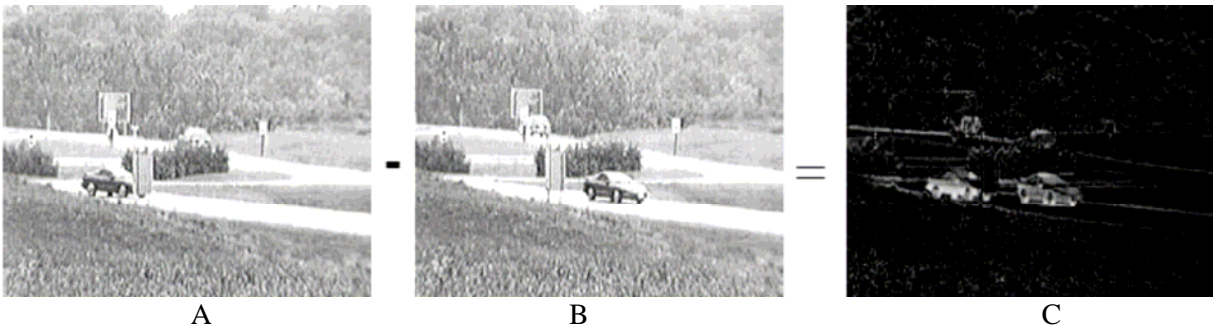


Figura 4.7.2: Efecto de aplicar la operación de resta para detectar movimiento $C = A - B$.

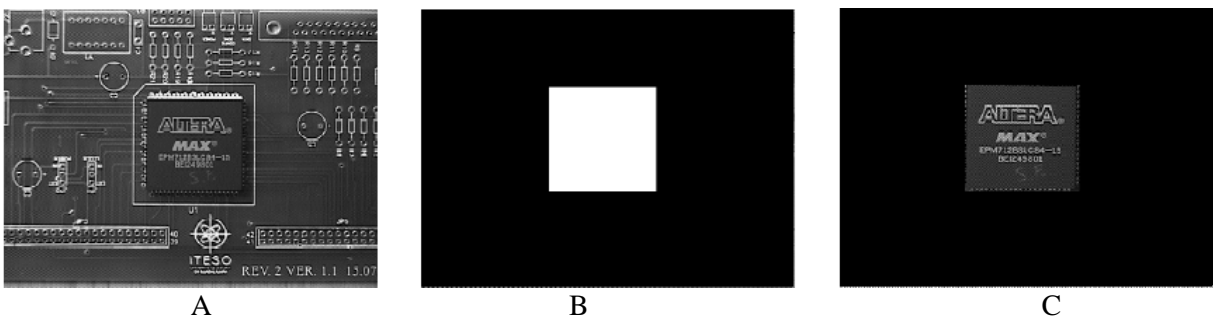


Figura 4.7.3: Efecto de aplicar la multiplicación para aislar regiones de interés $C = A * B$.

4.8. Operaciones morfológicas básicas

Se utilizan para eliminar pequeños puntos aislados que no pertenecen a nuestro objeto en cuestión mediante ciertas técnicas de morfología matemática como son: *erosión, dilatación, apertura y cerradura*.

Dilatación.- La dilatación $A \oplus B$ es el conjunto de puntos de todas las posibles sumas de pares de elementos uno de cada conjunto A y B:

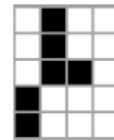
$$A \oplus B = \bigcup_{\beta \in B} (A + \beta) \quad (4.8.1)$$

Por ejemplo, dados A y B en forma de vectores para los valores correspondientes a los unos cuyas coordenadas se definen con respecto a $[0,0]$.

$$A = \{(0,1), (1,1), (2,1), (2,2), (3,0), (4,0)\}$$

$$B = \{(0,0), (0,1)\}$$

$$A \oplus B = \left\{ \begin{array}{l} (0,1), (1,1), (2,1), (2,2), (3,0), (4,0), \\ (0,2), (1,2), (2,2), (2,3), (3,1), (4,1) \end{array} \right\}$$

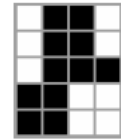


A



B

elemento
estructurante



$A \oplus B$

En los conjuntos A y B, se considera que A es la imagen y B es el elemento estructural. El elemento estructural es en morfología matemática lo que la máscara (o núcleo) de convolución es en los filtros lineales. Los elementos estructurales más comunes son los conjuntos que están 4-conectados (N_4), y 8-conectados (N_8), que se ilustran en las figuras 4.8.1 y 4.8.2:

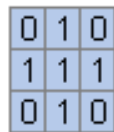


Figura 4.8.1: Elemento estructural N_4



Figura 4.8.2: Elemento estructural N_8

La operación de dilatación hace que los objetos se expandan y la cantidad o la forma en que crezcan depende del elemento estructurante que se elija (figura 4.8.3).

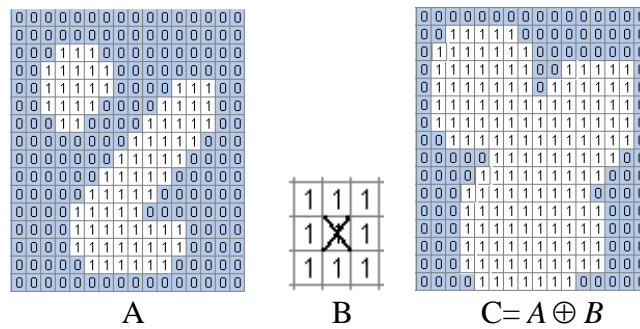


Figura 4.8.3. Dilatación: A) imagen original, B) elemento estructurante N8, C) imagen dilatada.

Erosión.- Combina dos conjuntos usando la resta vectorial y es el dual de la dilatación:

$$A \ominus B = \bigcup_{\beta \in B} (A - \beta) \quad (4.8.2)$$

En otras palabras, son los puntos A para los cuales todas las posibles traslaciones definidas por B también están en A (figuras 4.8.4 y 4.8.5).



Figura 4.8.4. Erosión con un elemento estructurante de 1x2

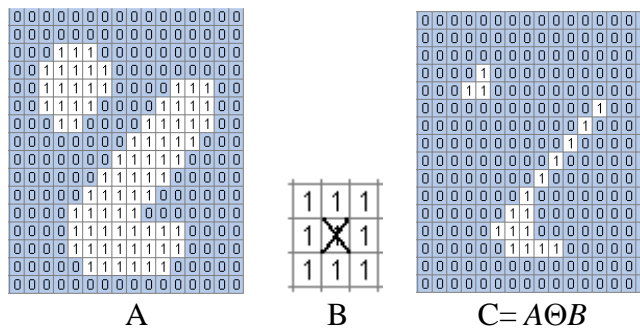


Figura 4.8.5. Erosión: A) imagen original, B) elemento estructurante de 3x3, C) imagen erosionada.

Apertura.- La apertura de A por B es una erosión de A por B seguida de la dilatación utilizando en ambas operaciones el mismo elemento estructurante.

$$A \circ B = (A \ominus B) \oplus B \quad (4.8.3)$$

Se utiliza para alisar el contorno de un objeto, rompe uniones pequeñas uniones entre objetos y elimina componentes ruidosas (puntos blancos). En la figura 4.8.6 se utiliza la apertura como un filtro de tamaño, de esta manera se eliminan los objetos pequeños menores a un elemento estructurante y también elimina el ruido contenido en la imagen.

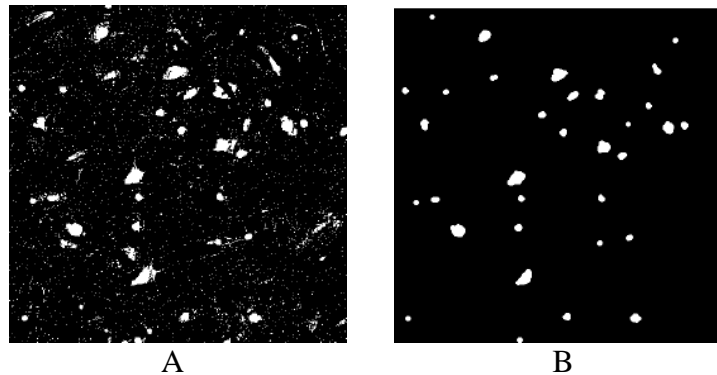


Figura 4.8.6. Apertura: A) imagen original, B) imagen resultante después de aplicar la apertura.

Cerradura.- La cerradura de A por B es la dilatación de A por B seguida de la erosión utilizando en ambas operaciones el mismo elemento estructurante.

$$A \bullet B = (A \oplus B) \ominus B \quad (4.8.4)$$

La Cerradura tiende a eliminar pequeños agujeros (puntos negros) del objeto, fusiona pequeños rompimientos y rellena pequeñas entradas al objeto como se muestra en la figura 4.8.7.

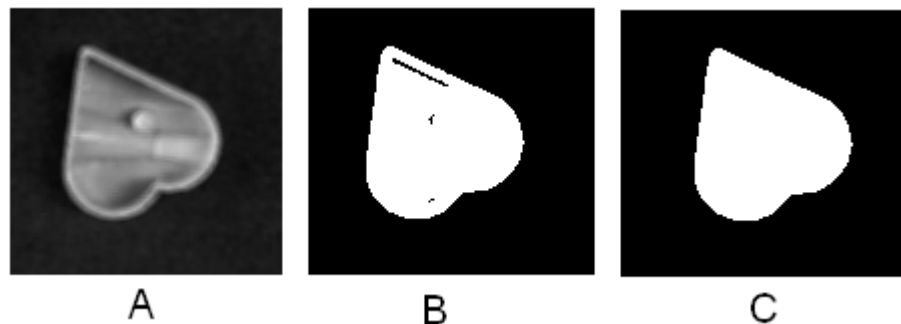


Figura 4.8.7. Cerradura: A) imagen en tonos de gris, B) imagen en blanco y negro C) imagen resultante después de aplicar cerradura.

Extracción de Frontera.- La frontera de un conjunto A, escrita como $\beta(A)$, se puede obtener erosionando A por B y luego calcular la diferencia entre A y su erosión. La figura 4.8.9 muestra el resultado de aplicar la extracción de frontera a una imagen binaria usando un elemento estructural de 3x3 estándar N_8 .



Figura 4.8.9: Extracción de frontera: A) imagen binaria,
B) el resultado de aplicar extracción de frontera

4.9. Extracción de contornos

La manera mas común de detectar contornos es aplicar algún tipo de derivada o diferencial, aplicado en un vecindario pequeño (mascara). La derivada nos permite calcular variaciones entre un punto y su vecindario. Viendo la imagen como una función, un contorno implica una discontinuidad en dicha función, es decir donde la función tiene un valor de gradiente o derivada alta (ver figura 4.9.1).

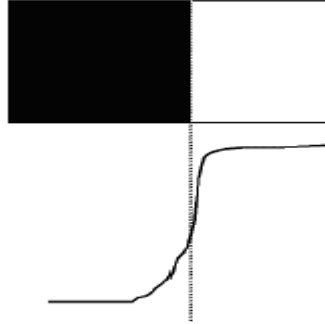


Figura 4.9.1: grafica de una imagen discontinua.

Operadores de gradiente

Las técnicas clásicas de detección de contornos se basan en encontrar la derivada respecto a los ejes x, y, o gradiente. El gradiente de una función se define como un vector bidimensional perpendicular al borde:

$$G[f(x, y)] = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{d}{dx} f(x, y) \\ \frac{d}{dy} f(x, y) \end{bmatrix} \quad (4.9.1)$$

donde el vector G apunta en la dirección de variación máxima de f en el punto(x,y) por unidad de distancia con la magnitud dada por:

$$|G| = \sqrt{G_x^2 + G_y^2} \quad (4.9.2)$$

En la práctica se aproxima la magnitud del gradiente con valores absolutos:

$$|G| \approx |G_x| + |G_y| \quad (4.9.3)$$

Para calcular la derivada en la ecuación (1) se pueden utilizar las diferencias de primer orden entre dos píxeles adyacentes:

$$G_x = \frac{f(x + \nabla x) - f(x - \nabla x)}{2\nabla x} \quad (4.9.4)$$

$$G_y = \frac{f(y + \nabla y) - f(y - \nabla y)}{2\nabla y} \quad (4.9.5)$$

$$g(x, y) = \begin{cases} 1 & \text{si } G[f(x, y)] > T \\ 0 & \text{si } G[f(x, y)] \leq T \end{cases} \quad (4.9.6)$$

donde T es un valor de umbral no negativo (en este caso T=80). Sólo los píxeles de borde cuyo gradiente excedan el valor de T se consideran importantes.

Operadores de Sobel

Los operadores de gradiente tienen la desventaja de aumentar el ruido en la imagen, tanto los operadores de Sobel como el resto de operadores de vecindad tienen la propiedad de suavizar la imagen, eliminando parte del ruido y minimiza la aparición de falsos contornos.

A partir de las ecuaciones (4) y (5), las derivadas basadas en los operadores de Sobel son:

$$G_x = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7) \quad (4.9.7)$$

$$G_y = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3) \quad (4.9.8)$$

donde los valores de z en la región de la figura 4.9.2 (a), son los niveles de gris de los píxeles solapados por las máscaras en cualquier localización de la imagen. A partir de las ecuaciones (7) y (8) se determinan las máscaras para obtener G_x y G_y para el punto central (figuras 4.9.2 a y b). Una vez que se ha obtenido la magnitud del gradiente, se puede decidir si un determinado punto es de borde o no aplicando la ecuación (6) obteniendo como resultado una imagen binaria.

$$\begin{array}{ccc}
 \begin{bmatrix} z_1 & z_2 & z_3 \\ z_4 & z_5 & z_6 \\ z_7 & z_8 & z_9 \end{bmatrix} & \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} & \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \\
 \text{(a)} & \text{(b)} & \text{(c)}
 \end{array}$$

Figura 4.9.2: (a) Región de la imagen de dimensión 3x3, (b) Mascara usada para obtener G_x en el punto central, (c) Mascara usada para obtener G_y en el mismo punto central.

Por ejemplo, supongamos que la matriz de la figura 4.9.3 representa una imagen y se desea calcular el gradiente en el píxel marcado con el punto negro. La región de la máscara en donde se desea calcular el gradiente está marcada con los puntos en blanco.

$$\begin{bmatrix} \circ 0 & \circ 8 & \circ 8 & 8 & 8 & 8 \\ \circ 1 & \bullet 8 & \circ 9 & 7 & 6 & 8 \\ \circ 2 & \circ 3 & \circ 4 & 8 & 8 & 8 \\ 2 & 2 & 2 & 8 & 8 & 8 \\ 2 & 2 & 2 & 2 & 8 & 8 \end{bmatrix}$$

Figura 4.9.3: Imagen muestra

Para calcular el gradiente aplicamos las ecuaciones (4.6.7) y (4.6.8), obteniendo $G_x = 30 - 4 = 26$ y $G_y = 12 - 24 = 12$, utilizando la ecuación (4.6.3) se obtiene $|G| = 48$, si fijamos un valor de umbral $T=30$, el píxel marcado con el punto negro sería un punto de borde y si por el contrario, el umbral es mayor que 48, dicho punto no sería punto de borde. La figura 4.9.4 muestra un ejemplo de detección de bordes utilizando los operadores de Sobel.

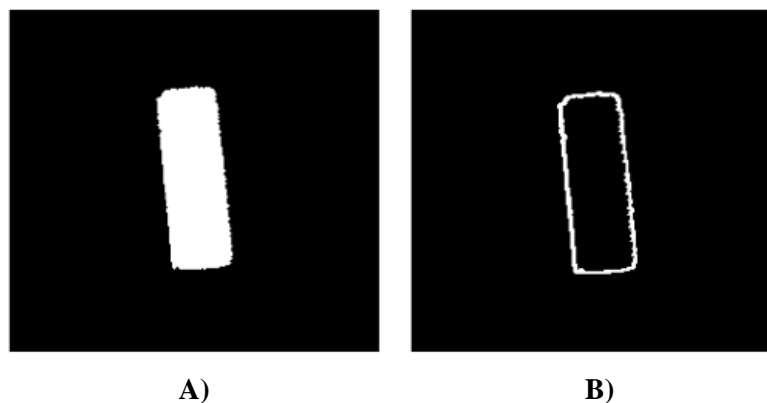


Figura 4.9.4: A) imagen original, B) imagen obtenida al aplicar la máscara de la figura 4.9.2

Operador Prewitt

El operador Prewitt es similar al de Sobel, lo único que cambia son los coeficientes de las máscaras, como se muestra en la figura 4.9.5.

$$\begin{array}{cc} \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} & \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \\ \text{(a)} & \text{(b)} \end{array}$$

Figura 4.9.5: Máscaras de Prewitt. A) máscara usada para obtener G_x en el punto central, b) máscara usada para obtener G_y en el mismo punto.

La figura 4.9.6 es el resultado de aplicar los operadores de Prewitt a la imagen 4.9.4 (A)

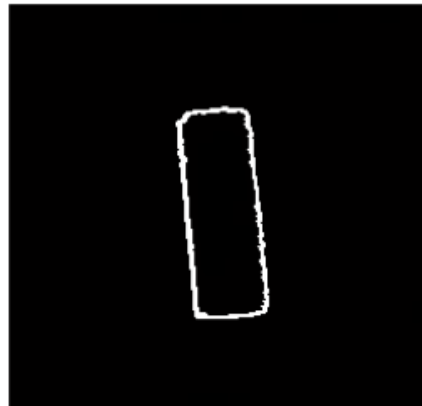


Figura 4.9.6: Operador de Prewitt.

Operador de Roberts

El operador de Roberts es muy simple y trabaja muy bien en imágenes binarias. Existen dos formulas para calcular el operador de Roberts:

$$\sqrt{[f(x, y) - f(x-1, y-1)]^2 + [f(x, y-1) - f(x-1, y)]^2} \quad (4.9.9)$$

$$|f(x, y) - f(x-1, y-1)| + |f(x, y-1) - f(x-1, y)| \quad (4.9.10)$$

La más usada es la ecuación (2) por ser más simple y utilizar menos coste de cómputo. En las figura 4.9.7 se muestran la imagen obtenida por aplicación de la ecuación (6) a la imagen 4.9.4.(a) utilizando los operadores de Roberts.

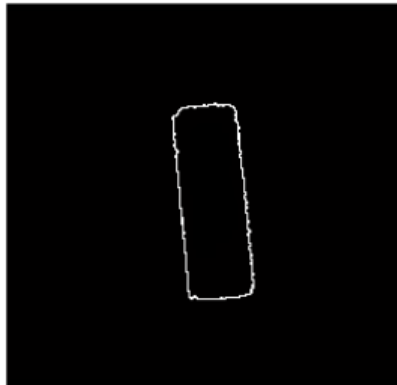


Figura 4.9.7: Operador de Roberts

Mascaras de Kirsch

Las máscaras de Kirch se denominan también de brújula porque se definen considerando una máscara simple y rotándola en las ocho direcciones de la brújula. Las máscaras se definen como sigue en la figura 4.9.8.

$$\begin{aligned}
 k_0 &\equiv \begin{bmatrix} -3 & -3 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & 5 \end{bmatrix} & k_1 &\equiv \begin{bmatrix} -3 & 5 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & -3 \end{bmatrix} & k_2 &\equiv \begin{bmatrix} 5 & 5 & 5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix} & k_3 &\equiv \begin{bmatrix} 5 & 5 & -3 \\ 5 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix} \\
 k_4 &\equiv \begin{bmatrix} 5 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & -3 & -3 \end{bmatrix} & k_5 &\equiv \begin{bmatrix} -3 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & 5 & -3 \end{bmatrix} & k_6 &\equiv \begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & -3 \\ 5 & 5 & 5 \end{bmatrix} & k_7 &\equiv \begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & 5 \\ -3 & 5 & 5 \end{bmatrix}
 \end{aligned}$$

Figura 4.9.8: Mascaras de Kirsch en las ocho direcciones.

Para cada punto de la imagen se obtienen 8 valores, resultantes de la convolución con cada una de las máscaras, el valor del módulo del gradiente es el máximo de esos ocho valores. En la figura 4.9.9 se muestra la imagen del módulo del gradiente obtenida con el operador Kirch a partir de la figura 4.6.4. (a).

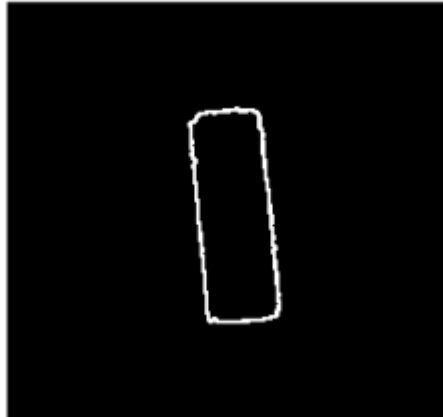


Figura 4.9.9: operador de Kirsch

Operador Laplaciana

La Laplaciana de una función 2D $f(x,y)$ es un operador de segunda derivada definido como:

$$\nabla^2 f = \frac{d^2 f}{dx^2} + \frac{d^2 f}{dy^2} \quad (4.9.11)$$

La ecuación anterior se puede implementar en forma digital como:

$$\nabla^2 f = 4z_5 - (z_2 + z_4 + z_6 + z_8) \quad (4.9.12)$$

donde los coeficientes z se han definido en la figura 4.6.2 (a) y el requisito básico es que los coeficientes asociados con el píxel central y los coeficientes asociados con el resto de píxeles sean negativos. Puesto que la Laplaciana es una derivada, la suma de los coeficientes debe ser cero. Por lo tanto, la respuesta es cero siempre que el punto en cuestión y sus vecinos tienen el mismo valor.

Las tres máscaras Laplacianas de la figura 4.9.10 representan diferentes aproximaciones del operador Laplaciano y son capaces de detectar bordes en todas las direcciones espaciales. En la figura 4.9.11 se muestran los resultados de aplicar los operadores de la figura 4.6.10.

$$\begin{matrix} \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} & \begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix} & \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \\ \text{(a)} & \text{(b)} & \text{(c)} \end{matrix}$$

Figura 4.9.10: Máscaras de operadores Laplaciana

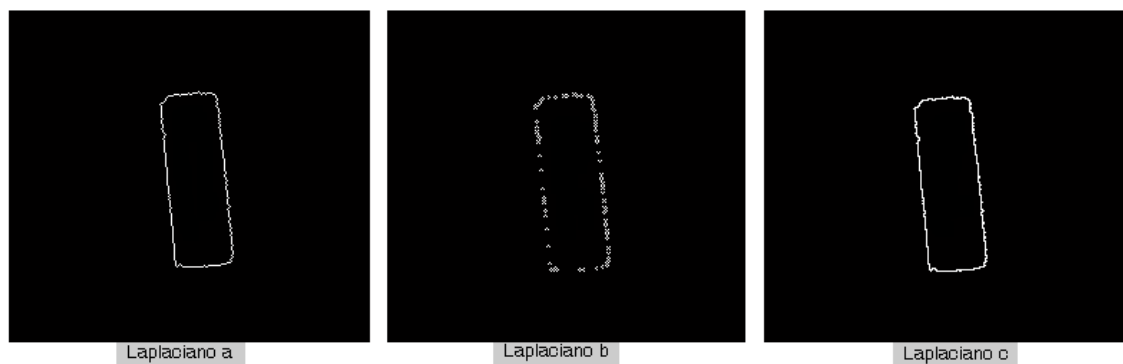


Figura 4.9.11: Moperadores Laplaciana

Capítulo 5

SEGMENTACIÓN

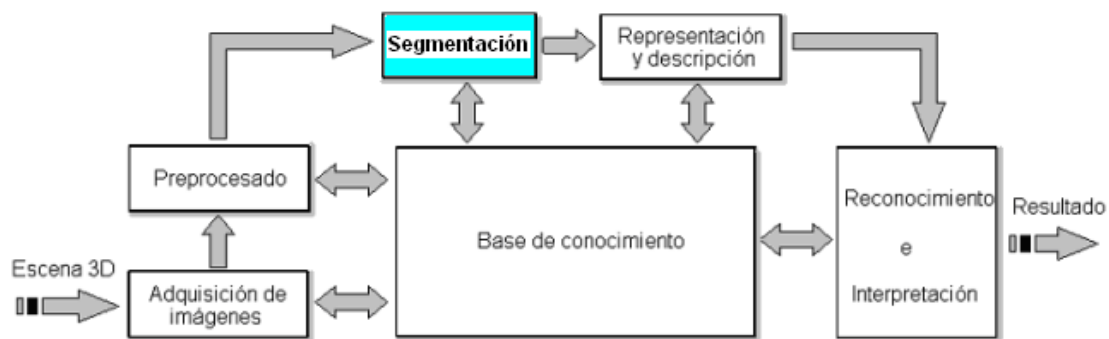


Figura 5.1.0: Etapas fundamentales en el reconocimiento automático de objetivos

En la figura 5.1.0 se resalta la etapa de segmentación que es un proceso por el cual se particiona o divide la imagen en sus componentes (*objetos de interés*) para su uso posterior. La segmentación autónoma es una de las tareas más difíciles del procesamiento de imágenes ya que esta etapa determina el éxito o el fracaso del proyecto.

La segmentación puede ser orientada en regiones (área de la imagen en la que sus píxeles poseen propiedades similares de intensidad o de color) o a bordes (líneas que separan dos regiones). Tanto la detección de regiones como la de bordes implican una manipulación de la imagen original, donde los valores de los píxeles originales son modificados mediante ciertas operaciones de transformación u operadores.

5.1 Detección de regiones basada en umbrales

Si analizamos el histograma de una imagen en tonos de gris que corresponde a una imagen con un objeto claro sobre fondo oscuro (ver figura 5.1.1), de tal forma que los píxeles del objeto y del fondo tienen los niveles de gris agrupados en dos modos dominantes, como podemos ver en el histograma de la figura 5.1.2, una forma de extraer los objetos del fondo es elegir un umbral “ T ” que separe dichos grupos. Cualquier punto (x, y) para el que $f(x, y) > T$ se asigna al objeto, en caso contrario al fondo; en otras palabras, se pueden separar los píxeles en dos regiones si asignamos con un valor de intensidad cero a todos los píxeles cercanos o iguales a cero (fondo negro) y con 255 al resto de los valores (objeto).

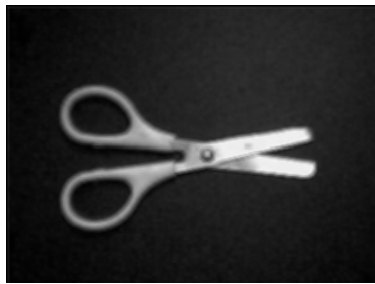


Figura 5.1.1: Imagen en tonos de gris

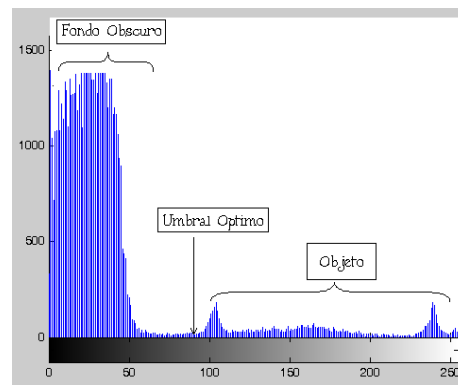


Figura 5.1.2: Histograma

Analizando el histograma, se deduce que un umbral óptimo que segmenta a la imagen en dos regiones es $= 97$. La primera región correspondiente al fondo y la otra al objeto. Al utilizar un umbral más pequeño, los puntos claros correspondientes al fondo se unen con el objeto a segmentar: por el contrario, si tomamos un umbral más grande corremos el riesgo de tomar zonas de el objeto que son un poco oscuras y unirse con el fondo (ver figuras 5.1.3, 5.1.4 y 5.1.5).



Figura 5.1.3: umbral óptimo 97



Figura 5.1.4: umbral = 45



Figura 5.1.5: umbral = 140

5.2 Método de segmentación de Otsu

La importancia del método de Otsu radica en que es automático, es decir, no necesita supervisión humana ni información previa de la imagen antes de su procesamiento. Este método se emplea cuando hay una clara diferencia entre los objetos a extraer respecto del fondo de la escena. En este método, se elige el umbral óptimo maximizando la varianza entre clases mediante una búsqueda exhaustiva. A medida que el número de clases de una imagen aumenta, el método de Otsu necesita mucho más tiempo para seleccionar un umbral multinivel adecuado.

Al aplicar un umbral, T , la imagen en escala de grises, $f(x,y)$, quedará binarizada; etiquetando con "1" los píxeles correspondientes al objeto y con "0" aquellos que son del fondo. Si los objetos son claros respecto del fondo, se aplica la siguiente fórmula:

$$g(x, y) = \begin{cases} 1 \Leftrightarrow f(x, y) > T \\ 0 \Leftrightarrow f(x, y) \leq T \end{cases} \quad (5.2.1)$$

Descripción del Método de Otsu para un umbral óptimo

El número de píxeles con nivel de gris " i " se denota como " f_i ", y la probabilidad de ocurrencia del nivel de gris " i " en la imagen está dada por:

$$P_i = \frac{f_i}{N * M} \quad (5.2.2)$$

Para la umbralización en dos niveles de una imagen, los píxeles son divididos en dos clases: **C1**, con niveles de gris [1,..., t]; y **C2**, con niveles de gris [t+1,..., L]. Entonces, la distribución de probabilidad de los niveles de gris para las dos clases son:

$$C_1 = \frac{p_1}{\omega_1(t)}, \dots, \frac{p_t}{\omega_1(t)} \quad (5.2.3)$$

$$C_2 = \frac{p_{t+1}}{\omega_2(t)}, \frac{p_{t+2}}{\omega_2(t)}, \dots, \frac{p_L}{\omega_2(t)} \quad (5.2.4)$$

donde:

$$\omega_1(t) = \sum_{i=1}^t p_i \quad (5.2.5)$$

$$\omega_2(t) = \sum_{i=t+1}^L p_i \quad (5.2.6)$$

y la media para la clase C1 y para la clase C2 son:

$$\mu_1 = \sum_{i=1}^t \frac{i * p_i}{\omega_1(t)} \quad (5.2.7)$$

$$\mu_2 = \sum_{i=t+1}^L \frac{i * p_i}{\omega_2(t)} \quad (5.2.8)$$

Otsu definió la variancia entre clases de una imagen umbralizada como:

$$\sigma_B^2 = \varpi_1(\mu_1 - \mu_T)^2 + \varpi_2(\mu_2 - \mu_T)^2 \quad (5.2.9)$$

también verificó que el umbral óptimo t^* se elige de manera que σ_B^2 sea máxima:

$$t^* = \underset{t}{Max} \{ \sigma_B^2(t) \} \quad 1 \leq t \leq L \quad (5.2.10)$$

Ejemplo:

Supongamos un imagen de $N*M=100$ píxeles con cuatro niveles de gris comprendidos en $[1,4]$ (1 el negro, 4 el blanco) y supongamos también que el número de píxeles con nivel de gris es el siguiente:

Nivel	# de píxeles f_i
1	10
2	20
3	30
4	40
Total	100

La Probabilidad de ocurrencia $P_i = \frac{f_i}{N * M}$ para cada nivel sería:

Nivel	# de píxeles f_i	Probabilidad de ocurrencia p_i
1	10	$P_1=10/100=0.1$
2	20	$P_2=20/100=0.2$
3	30	$P_3=30/100=0.3$
4	40	$P_4=40/100=0.4$

para una umbralización en dos niveles de esta imagen tomemos $t=2$ de manera que la clase C_1 consista en los tonos de gris 1 y 2, y la clase C_2 contenga los tonos 3 y 4.

por lo tanto, la Distribución de Probabilidad para

$$\omega_1(t) = \sum_{i=1}^t p_i \quad \text{y} \quad \omega_2(t) = \sum_{i=t+1}^L p_i \quad \text{es:}$$

Nivel	# de píxeles f_i	Probabilidad de ocurrencia p_i	Distribución de Probabilidad
1	10	$P_1=10/100=0.1$	$\omega_1(t)=0.1+0.2=0.3$
2	20	$P_2=20/100=0.2$	
3	30	$P_3=30/100=0.3$	$\omega_2(t)=0.3+0.4=0.7$
4	40	$P_4=40/100=0.4$	

Se comprueba que $\omega_1(t)+\omega_2(t)=1$.

Por último, la media para la clase C_1 y para la clase C_2 estará dada por:

$$\mu_1 = \sum_{i=1}^2 \frac{i * p_i}{\omega_1(t)} = \frac{1 * 0.1 + 2 * 0.2}{0.3} \approx 1.667$$
$$\mu_2 = \sum_{i=3}^4 \frac{i * p_i}{\omega_2(t)} = \frac{3 * 0.3 + 4 * 0.4}{0.7} \approx 3.57$$

y

$$\mu_t = \varpi_1 \mu_1 + \varpi_2 \mu_2 = 0.3 * 1.667 + 3.57 \approx 3$$

El valor óptimo puede encontrarse al buscar en el rango $1 \leq t \leq L$ el valor de variancia entre clases que de máxima.

Capítulo 6

REPRESENTACIÓN Y DESCRIPCIÓN

Proyecto terminal II

Capítulo 7

RECONOCIMIENTO E INTERPRETACIÓN

Proyecto terminal II

Capítulo 8

CÁLCULO DE DISTANCIA CON UN PUNTERO DE LÁSER

Proyecto terminal II

Capítulo 9

GRABACIÓN DE ARCHIVOS DE AUDIO

Proyecto terminal II

Capítulo 10

IMPLEMENTACIÓN DEL PROYECTO

En este capítulo se analizan algunos aspectos para la implementación de nuestro proyecto, desde la captación de la imagen hasta la etapa de segmentación.

Las etapas de representación y descripción, reconocimiento e interpretación, así como la grabación de archivos de audio analizarán en el proyecto terminal II.

Se muestran también los programas realizados en Matlab 7.0 mediante algoritmos que se presentaron en capítulos anteriores. También se presenta la forma de utilizar algunas funciones o pequeñas rutinas ya establecidas Matlab 7.0 y que se utilizan en el procesamiento de imágenes.

Se realizaron pruebas sobre el formato de almacenamiento de imágenes, el tipo de resolución a emplear, las consideraciones mas importantes en la captación de imágenes, así como el operador mas sencillo y eficiente para la detección de contornos.

10.1. Adquisición de la imagen

En la etapa de Adquisición de la imagen se utilizó una VideoCam Genius NB y algunas herramientas de adquisición de imágenes “*Image Acquisition Toolbox*” y de procesamiento de imágenes “*Image Processing Toolbox*”) del lenguaje MATLAB versión 7.0 (ver apéndice A).

Las imágenes se capturaron en formato BMP por ser un formato estándar y no distorsiona los colores. En la figura 10.1.1. se muestra la VideoCam utilizada y en la figura 10.1.2 los resultados la imagen captada.



Figura 10.1.1: VideoCam Genius NB

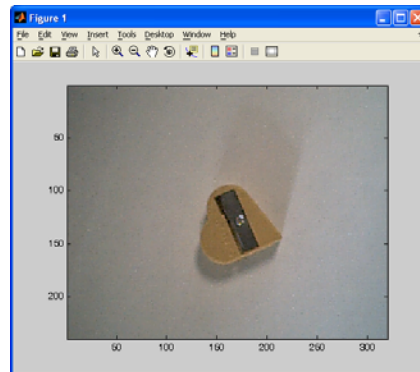


Figura 10.1.2: Captación de la imagen a través de la VideoCam

La resolución que se utilizó para este sistema de reconocimiento es de 320x200 píxeles, ya que si se utilizaba una resolución más baja se pierde detalle del objeto a reconocer y si se aumenta la resolución mejora la calidad de la imagen pero también aumenta el tiempo de proceso (ver figura 10.1.3).

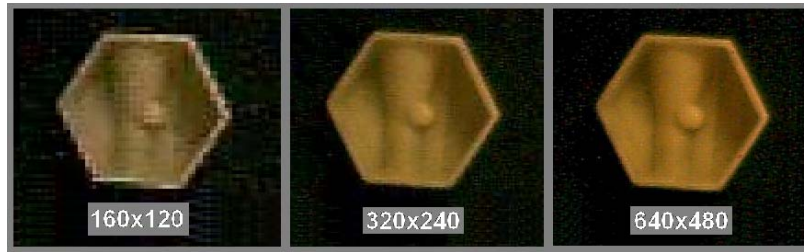


Figura 10.1.3: Diferentes tipos de resolución .

Uno de los problemas frecuentes en la etapa de captación de la imagen es la inadecuada iluminación del objeto a reconocer ver figura 10.1.4, ya que es posible que las sombras se tomen como parte del objeto o los cambios de tono del color hacen que el sistema de reconocimiento no reconozca el color real del objeto. En nuestro sistema de reconocimiento se utilizó un fondo negro para evitar las sombras producidas por la iluminación, se realizó dentro de un salón de clases con buena iluminación



Figura 10.1.4: Objeto a reconocer con una mala iluminación.

Otros puntos que se deben tomar en cuenta en la etapa de captación de la imagen son: el tamaño del objeto, el ángulo de captura, el enfoque de la VideoCam, entre otros. Si nuestro objeto es muy grande, la VideoCam no se alcanza a captar todo el objeto; si el ángulo de captación de la VideoCam no es el adecuado, las dimensiones del objeto cambian y si no está bien enfocado el objeto aparece muy borroso y no se puede determinar la identidad del objeto (ver figura 10.1.5). Para evitar este tipo de detalles, se colocó la VideoCam a 20 cm del objeto, se realizó un buen enfoque del objeto y se utilizaron objetos pequeños que no rebasaran el ángulo de visión de la VideoCam.

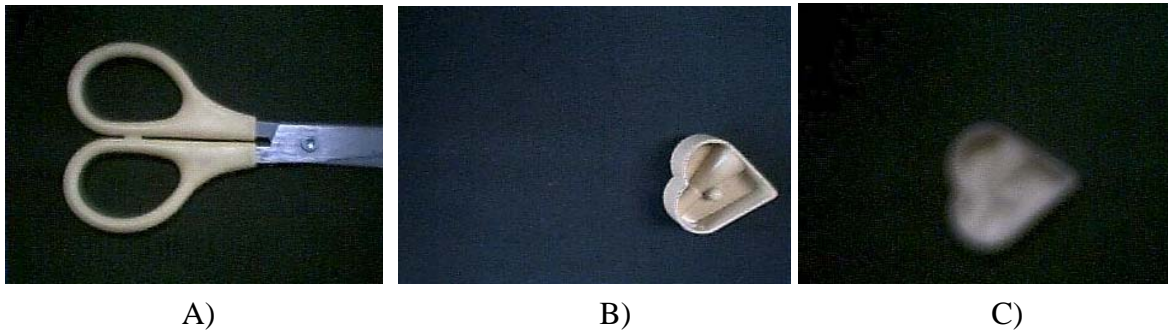


Figura 10.1.5: Consideraciones importantes en la captación de imágenes:
 A) Tamaño del objeto, B) ángulo de captación y C) desenfoque.

El código para la captura de imágenes a través de la VideoCam se muestra en el programa #1, donde se utiliza el dispositivo número 2 “Genius VideoCAM NB” con un formato de captura en color “RGB24_320x240” (para mayor detalle de cada una de las instrucciones se sugiere revisar el apéndice “A”).

```
clear all
video = videoinput('winvideo', 2, 'RGB24_320x240');
preview(video)
x=input('ENTER PARA CAPTURAR IMAGEN');
imagen = getsnapshot(video);
imagesc(imagen)
imwrite(imagen, 'sacapuntas1.bmp', 'bmp');
delete(video)
```

Programa #1 Captación de la imagen

9.2. Preprocesamiento de la imagen

Antes de realizar mejoras a la imagen es necesario hacer una conversión a tonos de gris y así poder eliminar o suavizar un poco el ruido en la imagen obtenida por la VideoCam. Para realizar la conversión a tonos de gris se utilizó el promedio las tres matrices de cada píxel (ver figura 10.2.1).

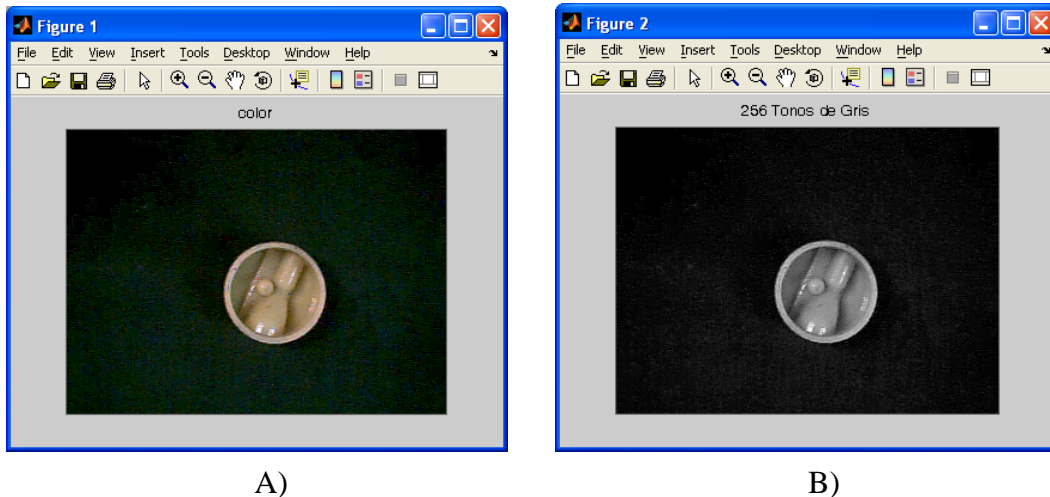


Figura 10.2.1. Conversión de una imagen en color a blanco y negro:
A) imagen en color, B) imagen en tonos de gris.

La codificación del programa de conversión a tonos de gris se muestra en el programa #2. Donde se separan cada uno de los componentes en RGB de la imagen en color y se realiza el promedio de los tres colores por cada píxel.

```
clear; close all
im_color = imread('sacapuntas1.bmp');
rojo=double(im_color(:,:,1));
verde=double(im_color(:,:,2));
azul=double(im_color(:,:,3));
im_gris=round((rojo(:,:,)+verde(:,:,)+azul(:,:,))/3);
im_gris=uint8(im_gris);
figure, imshow(im_color); title('color');
figure, imshow(im_gris); title('256 Tonos de Gris');
```

Programa #2 Conversión a tonos de gris

Se utilizaron diferentes tipos de filtros descritos en el capítulo 4 como son: *el filtro promedio, mediana, mínimo, máximo y punto medio*; pero el filtro con el que se obtuvieron mejores resultados para nuestro proyecto fue el filtro promedio ya que además de eliminar un poco el ruido, realiza un suavizado de la imagen, como se muestra en la figura 10.2.2.

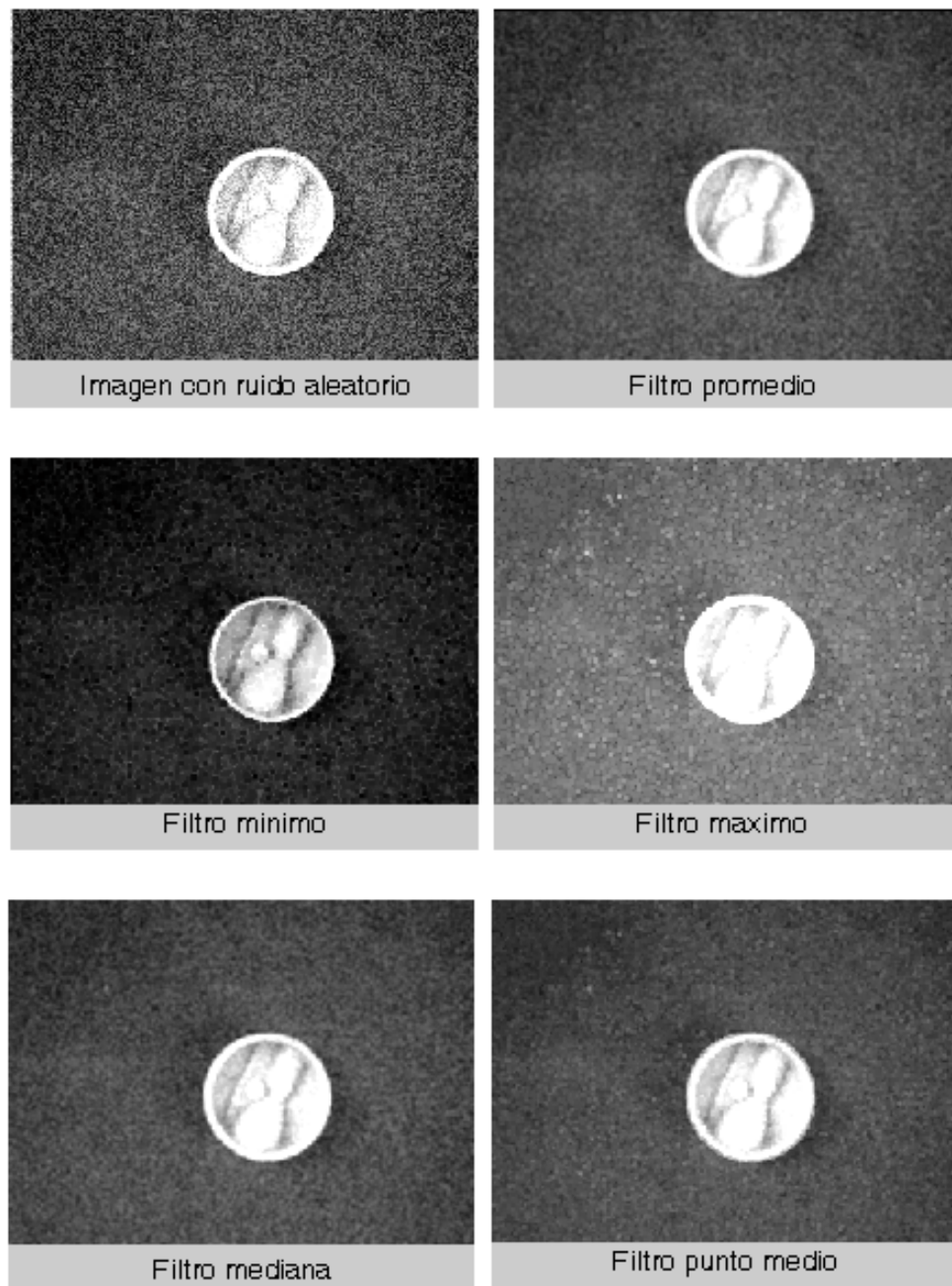


Figura 10.2.2: Uso de filtros para una imagen con ruido aleatorio.

La realización del programa #3 se implementó con máscaras de 3x3 (ver capítulo 4). Se comenzó con la segunda fila y segunda columna de la imagen para terminar con la penúltima fila y penúltima columna de la misma imagen; pero la imagen resultante para cada filtro tiene dos columnas y dos filas menos que la original.

```

clear all; close all;
im_color=imread('sacapuntas1.bmp');
im_color = imadjust(im_color,stretchlim(im_color));
rojo=double(im_color(:,:,1));
verde=double(im_color(:,:,2));
azul=double(im_color(:,:,3));
im_gris=round(rojo(:,:,:)*0.33)+round(verde(:,:,:)*0.59)+round(azul(:,:,:)*0.11);
im=double(im_gris);

ruido_aleatorio=floor(rand(fil,col)*100);
im_ruido=im+ruido_aleatorio;
im=im_ruido;
[fil col]=size(im);
for x=2:1:fil-1
    for y=2:1:col-1
        a=im(x-1,y-1);b=im(x,y-1);c=im(x+1,y-1);d=im(x-1,y);e=im(x,y);f=im(x+1,y);
        g=im(x-1,y+1);h=im(x,y+1);i=im(x+1,y+1);
        im_fil_prom(x,y)=round((a+b+c+d+e+f+g+h+i)/9);
        vector(1:9)=sort([a b c d e f g h i]);
        im_fil_mediana(x,y)=round(sum(vector)/9);
        im_fil_minimo(x,y)=vector(1);
        im_fil_maximo(x,y)=vector(9);
        im_fil_punto_m(x,y)=round((vector(1)+vector(9))/2);
    end
end

figure
hold on
subplot(2,3,1); imshow(uint8(im_ruido));title('Imagen con ruido aleatorio');
subplot(2,3,2); imshow(uint8(im_fil_prom));title('Filtro promedio');
subplot(2,3,3); imshow(uint8(im_fil_mediana));title('Filtro mediana');
subplot(2,3,4); imshow(uint8(im_fil_minimo));title('Filtro minimo');
subplot(2,3,5); imshow(uint8(im_fil_maximo));title('Filtro maximo');
subplot(2,3,6); imshow(uint8(im_fil_punto_m));title('Filtro punto medio');

```

Programa #3: Filtros para imágenes en tonos de gris.

La ecualización de histograma resalta el contraste de la imagen, el problema es que también resalta el ruido y sombras (ver figura 10.2.3); por lo tanto, no se utilizó esta técnica en la implementación de nuestro sistema de reconocimiento. El código para realizar esta técnica se presenta en el programa # 4.

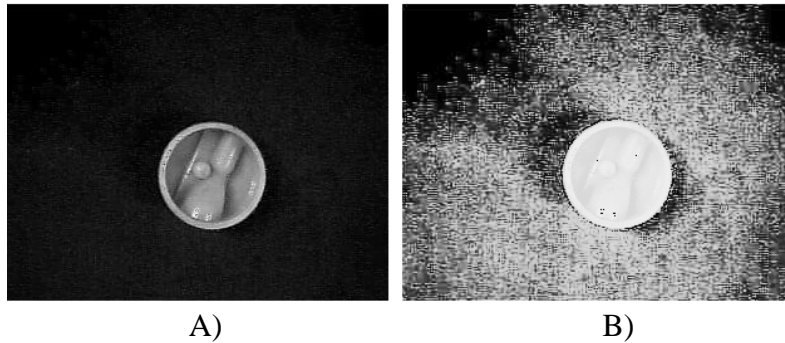


Figura 10.2.3. Ampliación de histograma A) imagen original, B) imagen resultado de la ampliación de histograma.

```

im=double(im_gris);
[fil col]=size(im);
vectorG=zeros(1,256);
for i=1:1:256; vectorG(i)=i-1; end;
vectorF=double(zeros(1,256));
vectorFa=double(zeros(1,256));
vectorFg=double(zeros(1,256));
im=double(im_gris);
for i=1:1:256
    for x=1:1:fil
        for y=1:1:col
            in=double(im(x,y));
            if i==in vectorF(i)=vectorF(i)+1; end;
        end
    end
end
vectorFa(1)=vectorF(1);
for i=2:1:256 vectorFa(i)=vectorFa(i-1)+vectorF(i); end;
for i=1:1:256
    vectorFg(i)=round((256*vectorFa(i))/(fil*col))-1;
    if vectorFg(i)<0 vectorFg(i)=0; end
end

```

```

im_res=zeros(fil,col);
im=double(im_gris);
for i=1:1:256
for x=1:1:fil
for y=1:1:col
in=double(im(x,y));
if in==vectorG(i)&& im_res(x,y)<in im_res(x,y)=vectorFg(i); end
end
end
end
figure; imshow(uint8(im_gris));title('Imagen gris');
figure; imhist(uint8(im_gris));title('histograma');
figure; imshow(uint8(im_res));title('resultado');
figure; imhist(uint8(im_res));title('histograma');

```

Programa #4: Ampliación de histograma.

9.3. Segmentación

En la etapa de segmentación, se utilizó la binarización con diferentes umbrales para aislar la imagen del fondo y eliminar el ruido. Las pruebas que se realizaron con diferentes umbrales se muestran en la figura 10.3.1. En algunos casos se forman manchas de color negro dentro de la imagen y en otras se agregan pequeños puntos blancos fuera de la imagen; pero para la implementación de nuestro sistema se utilizó el método de otsu que realiza la binarización en forma automática.

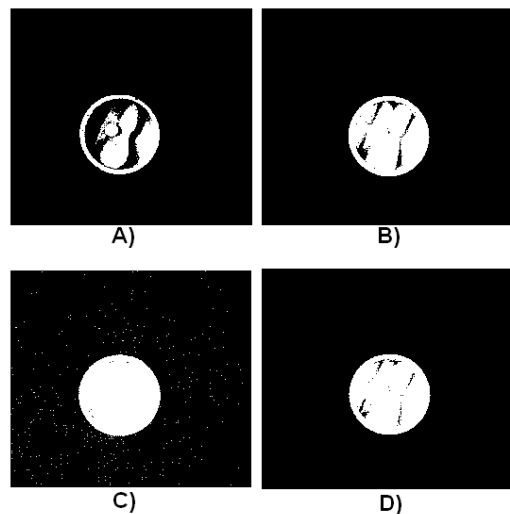


Figura 10.3.1. Binarización de la imagen; A) umbral = 192, B) umbral = 128, C) umbral = 64, D) umbral = 73 (umbral de Otsu para este caso).

El problema de la binarización es encontrar un umbral óptimo que divida la imagen a blanco y negro sin que se agregue ruido a la imagen a segmentar. El programa #5 muestra como binarizar una imagen en tonos de gris a blanco y negro con diferentes umbrales y en el programa #6 se muestra la función “graythresh” que utiliza el método de Otsu para calcular el umbral óptimo de forma automática directamente de la imagen en color.

```
[ancho largo]=size(im_gris)
for x=1:1:ancho
    for y=1:1:largo
        intensidad=im_gris(x,y);
        im_negativo(x,y)=uint8(255-double(intensidad));
        if intensidad>192 im_byn1(x,y)=255; else im_byn1(x,y)=0; end
        if intensidad>128 im_byn2(x,y)=255; else im_byn2(x,y)=0; end
        if intensidad>64 im_byn3(x,y)=255; else im_byn3(x,y)=0; end
    end
end
figure, imshow(im_color); title('color');
figure, imshow(im_gris); title('256 Tonos de Gris');
figure, imshow(im_negativo);title('negativo');
figure, imshow(im_byn1);title('blanco y negro 1');
figure, imshow(im_byn2);title('blanco y negro 2');
figure, imshow(im_byn3);title('blanco y negro 3');
```

Programa #5: Segmentación con diferentes umbrales

```
clear all; close all
im_color=imread('sacapuntas1.bmp');
nivel = graythresh(im_color);
im_byn = im2bw(im_color,nivel);
figure, imshow(im_byn);title('imagen en blanco y negro Otsu');
```

Programa #6: Segmentación con el método de Otsu

En la figura 10.3.2 se muestran dos técnicas para eliminación de ruido mediante operaciones morfológicas sobre imágenes binarias. La cerradura seguida de una apertura une los puntos blancos que están casi unidos, eliminando pequeños puntos negros; La apertura seguida de la cerradura disuelve casi por completo los puntos blancos y se obtienen mejores resultados.

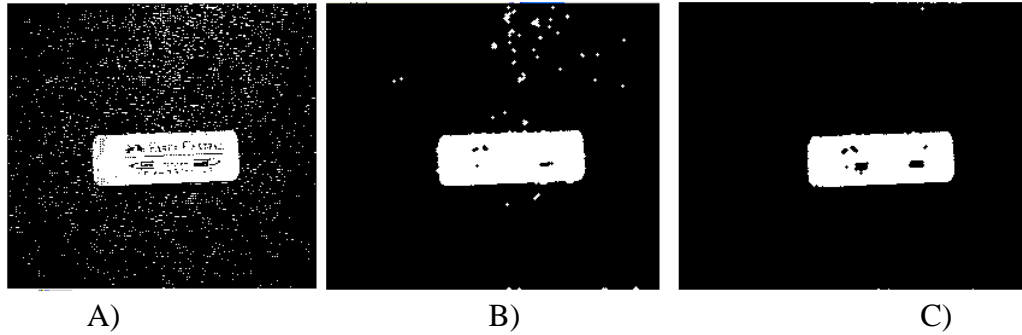


Figura 10.3.2 Operaciones morfológicas: A) imagen binarizada con ruido, B) aplicación de cerradura seguida de una apertura, C) aplicación de una apertura seguida de una cerradura.

Después de aplicar la cerradura seguida de una apertura, es necesario eliminar los huecos que se encuentran en el interior del objeto a describir utilizando la instrucción de “imfill”. También será necesario eliminar pequeños objetos que por algún motivo se filtraron; para ello se utilizó la instrucción “bwareaopen”.

A continuación se presenta su implementación:

```
im = im_open_close;
im = imfill(im,'holes');
im = bwareaopen(im,500);
```

donde, en la primera línea se almacena la imagen resultante de aplicar la apertura y cerradura en la imagen “im”, la segunda línea elimina los huecos de la imagen “im” y la tercera línea elimina elementos menores a 500 píxeles de la imagen “im”. En la figura 10.3.3 se presentan el resultado obtenido al utilizar las funciones anteriores.

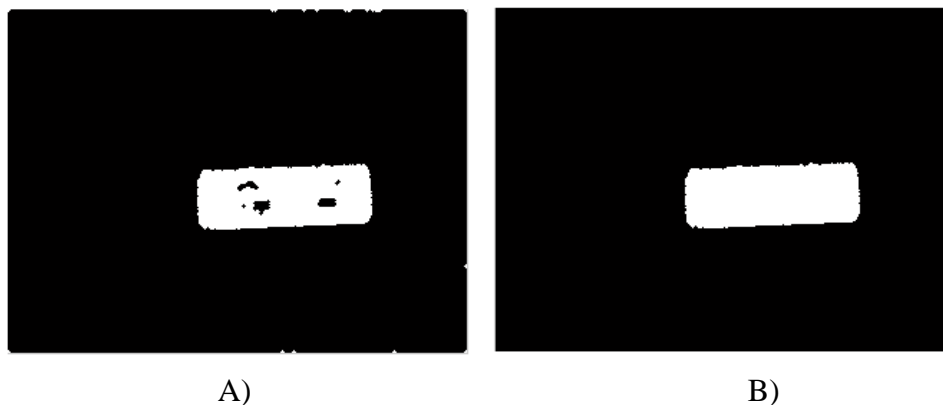


Figura 10.3.3 Eliminación de huecos y pequeños objetos: A) imagen con huecos y pequeños puntos, B) Resultado de aplicar las funciones “imfill” y “bwareaopen”.

El siguiente paso fue aislar por completo al objeto para determinar las coordenadas de cada punto perteneciente al objeto y así calcular su área, perímetro, color, etc. y realizar el reconocimiento. La forma en que se realizó el proceso fue multiplicando cada valor de la matriz correspondiente a cada color de la imagen por cada valor de la imagen binaria, como se muestra a continuación:

```
im=uint8(im);  
im_color2(:,:,1)=im_color(:,:,1).*im;  
im_color2(:,:,2)=im_color(:,:,2).*im;  
im_color2(:,:,3)=im_color(:,:,3).*im;
```

donde, "im_color" es la imagen original en color, "im_color2" es la imagen destino y "im" es la imagen binaria. Antes de realizar el proceso es necesario convertir la imagen binaria "im" a formato entero de 8 bits. El resultado de este proceso se muestra en la figura 10.3.4.



A)

B)

C)

Figura 10.3.4 Multiplicación de matrices punto a punto: A) imagen en color, imagen en blanco y negro, C) imagen resultado de la multiplicación.

También se hicieron pruebas con diferentes tipos de algoritmos para la detección de contornos como los que se muestran en la figura 10.3.5. Uno de los algoritmos mas simples y que dio muy buenos resultados fue el de Roberts en su versión simplificada (ver capítulo 4). El código para detección de contornos se muestra en el programa # 7.

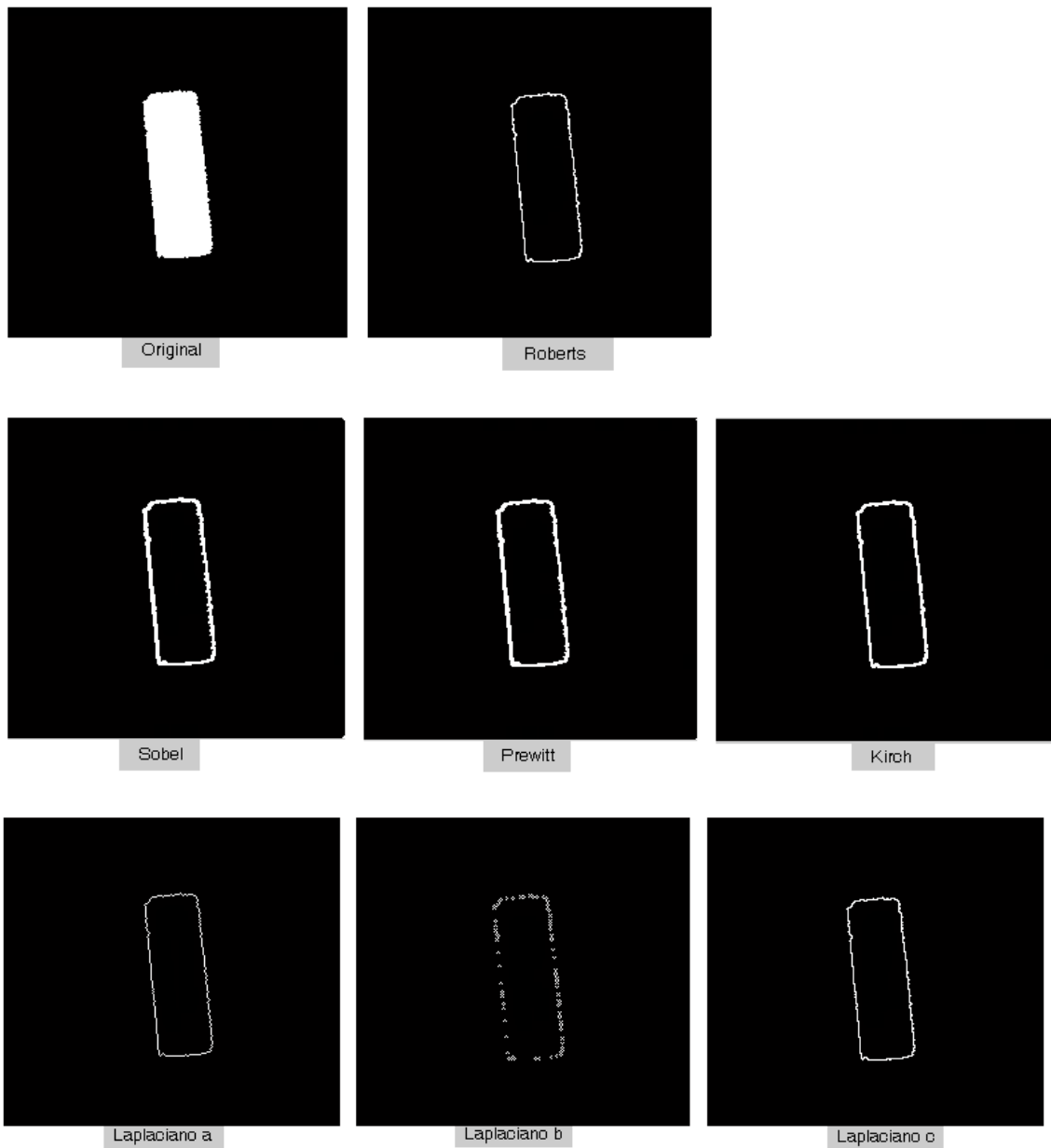


Figura 10.3.5 Aplicación de diferentes algoritmos para la detección de contornos

```

im=double(im);
[ancho largo]=size(im)
for x=2:1:ancho-1
    for y=2:1:largo-1
        a=im(x-1,y-1);b=im(x,y-1);c=im(x+1,y-1);d=im(x-1,y);e=im(x,y);
        f=im(x+1,y); g=im(x-1,y+1);h=im(x,y+1);i=im(x+1,y+1);
        a=double(a);b=double(b);c=double(c);d=double(d);e=double(e);
        f=double(f);g=double(g);h=double(h);i=double(i);
        Roberts(x,y)=abs(e-a)+abs(b-d);
        SobelM1=a*(-1)+d*(-2)+g*(-1)+c+f*2+i;
        SobelM2=a*(-1)+b*(-2)+c*(-1)+g+h*2+i;
        valores(1:2)=[SobelM1 SobelM2];
        Sobel(x,y)=uint8(max(valores));
        PrewittM1=a*(-1)+b*(-1)+c*(-1)+g+h+i;
        PrewittM2=c*(-1)+f*(-1)+i*(-1)+a+d+g;
        valores(1:2)=[PrewittM1 PrewittM2];
        Prewitt(x,y)=uint8(max(valores));
        KirchM1=a*(+5)+b*(+5)+c*(+5)+d*(-3)+f*(-3)+g*(-3)+h*(-3)+i*(-3);
        KirchM2=a*(-3)+b*(+5)+c*(+5)+d*(-3)+f*(+5)+g*(-3)+h*(-3)+i*(-3);
        KirchM3=a*(-3)+b*(-3)+c*(+5)+d*(-3)+f*(+5)+g*(-3)+h*(-3)+i*(+5);
        KirchM4=a*(-3)+b*(-3)+c*(-3)+d*(-3)+f*(+5)+g*(-3)+h*(+5)+i*(+5);
        KirchM5=a*(-3)+b*(-3)+c*(-3)+d*(-3)+f*(-3)+g*(+5)+h*(+5)+i*(+5);
        KirchM6=a*(-3)+b*(-3)+c*(-3)+d*(+5)+f*(-3)+g*(+5)+h*(+5)+i*(-3);
        KirchM7=a*(+5)+b*(-3)+c*(-3)+d*(+5)+f*(-3)+g*(+5)+h*(-3)+i*(-3);
        KirchM8=a*(+5)+b*(+5)+c*(-3)+d*(+5)+f*(-3)+g*(-3)+h*(-3)+i*(-3);
        valores(1:8)=[KirchM1 KirchM2 KirchM3 KirchM4 KirchM5 KirchM6 KirchM7
            KirchM8];
        Kirch(x,y)=uint8(max(valores));
        LaplacianoM1(x,y)=b*(-1)+d*(-1)+e*(4)+f*(-1)+h*(-1);
        LaplacianoM2(x,y)=a+b*(-2)+c+d*(-2)+e*(4)+f*(-2)+g+h*(-2)+i;
        LaplacianoM3(x,y)=a*(-1)+b*(-1)+c*(-1)+d*(-1)+e*(8)+f*(-1)+g*(-1)+h*(-1)+i*(-1);
    end
end
figure, imshow(im) ; title('Original');
figure, imshow(Roberts);title('Roberts');
figure, imshow(Sobel);title('Sobel');
figure, imshow(Prewitt);title('Prewitt');
figure, imshow(Kirch);title('Kirch');
figure, imshow(LaplacianoM1);title('Laplaciano a');
figure, imshow(LaplacianoM2);title('Laplaciano b');
figure, imshow(LaplacianoM3);title('Laplaciano c');

```

Programa #7: Detección de contornos

Conclusiones

Proyecto terminal II

Bibliografía

Bibliografía general

- [1] Aguilar, Karla P, “*Probando el desempeño de varias combinaciones de clasificadores y vectores de rasgos en la determinación de la identidad y el numero de objetos circulando en una banda transportadora*”, Tesis de maestría, 2001, Ed. Instituto Politécnico Nacional.
- [2] Legarda, Arturo S, “*Localización de placas en vehículos automotores*”, Tesis de maestría, 2001, Ed. Instituto Tecnológico de Chihuahua.
- [3] Rangel, Rafael S, “*Nueva técnica para contar objetos en imágenes*”, Tesis de maestría, 2001, Ed. Instituto Politécnico Nacional.
- [4] Meléndez, Alba R, “*Estimación de fondo y primer plano en secuencias de imágenes para la detección de objetos en movimiento*”, Tesis de maestría, 2003, Ed. Instituto nacional de astrofísica óptica y electrónica de puebla.
- [5] Jiménez, Antonio, “*Sistema de reconocimiento y localización de objetos cuasi-esféricos por telemetría láser. Aplicación a la detección automática de frutos para el robot AgriBot*”, Tesis de doctoral, 2000, Universidad Complutense de Madrid.
- [6] Pajares, Gonzalo, “*Visión por computador*”, Alfaomega, México, 2002.
- [7] González, Rafael C, “*Tratamiento digital de imágenes*”, Addison-Weslwy, New York, 1996.
- [8] Zagal, Juan, “*Adaptación evolutiva de un sistema visual de reconocimiento de objetos para el campeonato de fútbol robótico robocup*”, Universidad de chile, 2002.
- [9] Mery, Domingo, “*Detección de fallas en piezas fundidas usando una metodología de reconocimiento de patrones*”, Universidad de Santiago de Chile, 2003.
- [10] Tadeo, Fernando, “*Clasificación de microorganismos mediante procesado de imagen*”, Ingeniería de Sistemas y Automática, 2001
- [11] López, Manuel, “*manufactura inteligente utilizando visión para robots*”, Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas, 2004.

Bibliografía electrónica

[12] http://www.cochenet.com/sabelotodo/articulos/sistemas%20ayuda/coche_inteligente_vw/coche_inteligente_vw.htm

[13] <http://www.elmundo.es/elmundo/2003/03/20/enespecial/1048122765.html>

[14] <http://neurologia.rediris.es/congreso-1/conferencias/neuropsicologia-2-2.html>

[15] <http://vision.arc.nasa.gov/>

[17] <http://www.robocup.org.mx/>