



UNIVERSIDAD NACIONAL
"PEDRO RUIZ GALLO"



FACULTAD DE CIENCIAS FISICAS Y MATEMATICAS
ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA

T E S I S

**Implementación de un Algoritmo para la
Detección de Cuerpos Extraños de las Leguminosas
de Grano Seco Utilizando la Visión Artificial**

Tesis para optar el título profesional de:

INGENIERO ELECTRONICO

PRESENTADO POR:

Bach: JUAN EDUARDO JUÁREZ ELERA
Bach: LUIS GUSTAVO RODRIGUEZ GONZALES

ING. HUGO JAVIER CHICLAYO PADILLA
PRESIDENTE

Lambayeque, Enero del 2017

ING. CARLOS LEONARDO GUTIERREZ VERA
SECRETARIO

ING. OSCAR LUCIELY ROMERO CORTES
VOCAL

**UNIVERSIDAD NACIONAL
"PEDRO RUIZ GALLO"**

**FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA**

TESIS

**Implementación de un Algoritmo para la Detección de
Cuerpos Extraños de las Leguminosas de Grano Seco
Utilizando la Visión Artificial.**

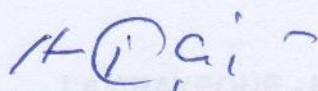
**PARA OPTAR EL TÍTULO PROFESIONAL DE
INGENIERO ELECTRÓNICO**

PRESENTADO POR:


Bach: JUAN EDUARDO JUÁREZ ELERA

Bach: LUIS GUSTAVO RODRIGUEZ GONZALES


SUSTENTADO Y APROBADO ANTE EL SIGUIENTE JURADO



**ING. HUGO JAVIER CHICLAYO PADILLA
PRESIDENTE**

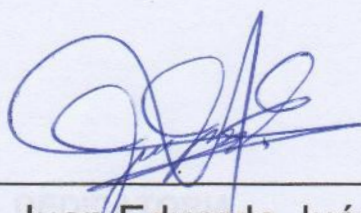


**ING. CARLOS LEONARDO OBLITAS VERA
SECRETARIO**



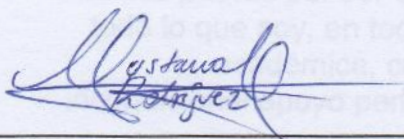
**ING. OSCAR UCHELLY ROMERO CORTEZ
VOCAL**

Implementación de un Algoritmo para la Detección de Cuerpos Extraños de las Leguminosas de Grano Seco Utilizando la Visión Artificial



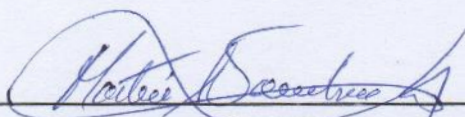
Bach. Juan Eduardo Juárez Elera

AUTOR



Bach. Luis Gustavo Rodríguez Gonzales

AUTOR



Ing. Martín Nombera Lossio

ASESOR

LAMBAYEQUE - PERÚ

ENERO 2017

Implementación de un Algoritmo para la Detección de Cuerpos Extraños de las Leguminosas de Grano Seco Utilizando la Visión Artificial

Bach. Juan Eduardo Juárez Elera

Bach. Gustavo Rodríguez Gonzales

Tesis para optar el título de
Ingeniero Electrónico

Asesor:

Martín Nombera Lossio
Ingeniero Electrónico

DEDICATORIA

A mis padres por ser el pilar fundamental en todo lo que soy, en toda mi educación, tanto académica, como de la vida, por su incondicional apoyo perfectamente mantenido a través del tiempo.
Todo este trabajo ha sido posible gracias a ellos.

AGRADECIMIENTOS

Primeramente agradezco a la Universidad Nacional Pedro Ruiz Gallo por haberme aceptado ser parte de ella y abierto las puertas de su seno científico para poder estudiar mi carrera, así como también a los diferentes docentes que brindaron sus conocimientos y su apoyo para seguir adelante día día.

RESUMEN

El presente trabajo propone la implementación de un algoritmo para la detección de cuerpos extraños de las leguminosas de grano seco, para el posterior procesamiento del mismo. La implementación del algoritmo se basa en técnicas de procesamiento digital de imágenes, las cuales se han implementado con la ayuda de una computadora personal con su respectiva cámara (webcam). El sistema detectará periódicamente las leguminosas de grano seco aptos para el consumo y proceso, y los diferenciará de cuerpos extraños por medio de su forma y de color.

El proceso de selección se realizará utilizando algoritmos de pre procesamiento de imágenes (filtros), algoritmos de detección de bordes y analizando el modelo de color HSI de la imagen. Después de la selección, se enviará la información necesaria al sistema mecánico encargado de la extracción de los cuerpos indeseados (fréjoles no aptos y cuerpos extraños) que será objeto para una posterior investigación.

ABSTRACT

This paper proposes the implementation of an algorithm for detecting foreign bodies of dry grain legumes for further processing thereof. The implementation of the algorithm is based on techniques of digital image processing, which have been implemented with the help of a personal computer with its respective camera (webcam). The system detected periodically dry grain legumes fit for consumption and processing, and foreign bodies differentiated by their shape and color.

The selection process will be performed using image pre-processing algorithms (filters), edge detection algorithms and analyzing the HSI color model image. After selection, the information necessary to the mechanical system responsible for removing unwanted body (frijoles unfit and foreign bodies) which will be subject to further investigation will be sent.

ÍNDICE GENERAL

	Pág.
RESUMEN	IV
ABSTRACT	V
INDICE GENERAL	VI
INDICE DE FIGURAS	VII
INTRODUCCION	11
CAPITULO 1	
1. DEFINICIONES PRELIMINARES	12
1.1 Visión Artificial	12
1.2 Aplicaciones de la Visión Artificial	12
1.3 Visión Artificial en la Robótica	17
1.4 Visión Artificial en la Industria	18
1.5 Usos de la Visión Artificial	24
1.6 Componentes de un Sistema de Visión Artificial	25
1.7 Como Funciona la Visión por Computador	26
1.8 Justificación de la Inversión en Visión Artificial	27
CAPITULO 2	
2. ETAPAS DE UN SISTEMA DE VISION ARTIFICIAL	29
2.1 Adquisición de la Imagen	29
2.1.1 Cámara	29
2.1.2 Digitalizador	31
2.1.3 Sistema de Iluminación	32
2.1.4 Fuentes de Iluminación	48
2.2 Pre-procesamiento de la Imagen	48
2.2.1 Conversión a Escala de Grises	49
2.2.2 Filtrado Espacial	50
2.3 Segmentación de la Imagen	52

2.3.1 Segmentación Basado en Píxeles	52
2.3.2 Operaciones Morfológicas	54
2.3.3 Etiquetado	57
2.3.4 Filtro de Tamaño	59
2.4 Descripción	60
2.5 Reconocimiento	60
CAPITULO 3	61
3. DESCRIPCION Y EXTRACCION DE CARACTERISTICAS	61
3.1 Propiedades en la Extracción de Características	61
3.2 Extracción de Características	62
3.2.1 Reduciendo la Dimensionalidad de los Datos	63
3.2.2 Cambiando el espacio de representación	64
3.3 Métodos de Descripción	65
3.3.1 Descriptores de forma	65
3.3.2 Descriptores de textura	72
3.4 Momentos Invariantes	72
3.4.1 Momentos Invariantes a Traslaciones	73
3.4.2 Momentos Invariantes a Traslaciones, Rotacionales	75
3.5 Cálculo de los Momentos Generales a partir del código	76
CAPITULO 4	82
4. RECONOCIMIENTO DE PATRONES	82
4.1 Introducción	82
4.2 Enfoque de un Sistema de Reconocimiento	82
4.2.1 Heurísticas	83
4.2.2 Matemáticas	83
4.2.3 Lingüísticas	84
4.3 Reconocimiento Estadístico de Patrones	85
4.3.1 Representación de los Patrones	86
4.3.2 Similitud entre Patrones	86
4.3.3 Variabilidad entre Patrones	87

4.4 Diseño de un Reconocedor de Formas	89
4.5 Modelo de Clasificadores	92
4.5.1 Aprendizaje Supervisado	92
4.5.2 Aprendizaje no Supervisado	93
CAPITULO 5	93
5. ALGORITMO DE PROCESAMIENTO DIGITAL DE IMAGENES	94
5.1 Modelos de Color	94
5.2 Filtrado de Imágenes	96
5.3 Fase de Construcción	100
5.3.1 Captura de la Imagen Digital	100
5.3.2 Procesamiento de la Imagen	102
5.3.3 Resultados	120
CAPITULO 6	122
6. CONCLUSIONES Y RECOMENDACIONES	122
6.1 Conclusiones	127
6.2 Recomendaciones	127
ANEXOS	123
BIBLIOGRAFIA	148

INDICE DE FIGURAS

Figura 1.1	Defecto de falso cierre	20
Figura 1.2	Defecto de pico y rebabas	20
Figura 1.3	Defecto de pestañas golpeadas	20
Figura 2.1	CCD tipo IT	30
Figura 2.2	CCD tipo ITF	31
Figura 2.3	Espectro Electromagnético	33
Figura 2.4	División del rayo de luz	34
Figura 2.5	Sensibilidad Espectral de un sensor CCD	40
Figura 2.6	Distribución Espectral de varias fuentes de luz	40
Figura 2.7	Patrón Anular de en una inspección de válvulas para farmacia.	41
Figura 2.8	Patrón de luz hemisférico	42
Figura 2.9	Retroalimentación difusa	45
Figura 2.10	Iluminación frontal	46
Figura 2.11	Iluminación direccional	47
Figura 2.12	Iluminación estructurada	47
Figura 2.13	Componentes RGB de una imagen	50
Figura 2.14	Mascara resultante	51
Figura 2.15	Entorno 3x3	53
Figura 2.16	Histograma de la imagen de una botella	54
Figura 2.17	Segmentación de la imagen de una botella	55
Figura 2.18	Ejemplo de dilatación	57
Figura 2.19	Ejemplo de erosión	58
Figura 2.20	Etiquetado	59
Figura 2.21	Imagen en forma de U	60
Figura 2.22	Filtrado de área	61
Figura 3.1	Selección de las 3 variables más significativas	63
Figura 3.2	Transformación de las 100 variables	64
Figura 3.3	Código de cadena – vecindad de 4	66
Figura 3.4	Código de cadena – vecindad de 8	66
Figura 3.5	Codificación de una frontera	67
Figura 3.6	Conectividad 4 y Conectividad 8	67
Figura 3.7	Región Continua y acotada	76
Figura 3.8	Tramo de un pixel	78
Figura 3.9	Curva del Área	79
Figura 4.1	Variabilidad entre patrones	88
Figura 4.2	Agrupamientos solapados	89
Figura 4.3	Diagrama de las etapas en el diseño del sistema de reconocimiento automático	91
Figura 5.1	Modelo RGB	94
Figura 5.2	Modelo HSI	95
Figura 5.3	Vector gradiente	97
Figura 5.4	Aproximaciones del vector gradiente	97

Figura 5.5	Discretización del Vector gradiente	98
Figura 5.6	Convolución de la imagen con las máscaras HF y HC.	98
Figura 5.7	Magnitud del Vector gradiente.	98
Figura 5.8	Mascaras del gradiente fila y columna.	98
Figura 5.9	Máscaras de Prewitt, Sobel y Frei-Chen	99
Figura 5.10	Vecindad de Moore de un pixel P	100
Figura 5.11	Imagen capturada de una cámara web	101
Figura 5.12	Secuencia de procesos para procesar la imagen	102
Figura 5.13	Imagen transformada a escala de grises	103
Figura 5.14	Imagen con filtro Pasa Bajo	104
Figura 5.15	Análisis del histograma de la imagen	104
Figura 5.16	Resultados de la Binarización	105
Figura 5.17	Elemento estructural (punto base en el centro)	105
Figura 5.18	Proceso para dilatar una imagen	106
Figura 5.19	Proceso para erosionar una imagen	107
Figura 5.20	Imagen luego de la operación de cierre	108
Figura 5.21	Pseudo-Código del algoritmo de segmentación	109
Figura 5.22	Algoritmo de relleno por difusión	109
Figura 5.23	Imagen Binaria	111
Figura 5.24	Resultados del primer escaneo	111
Figura 5.25	Resultados del segundo escaneo	112
Figura 5.26	Localización de núcleos	112
Figura 5.27	Etiquetado de los diferentes núcleos	113
Figura 5.28	Detección de uniones	114
Figura 5.29	Código de 8-cadena	114
Figura 5.30	Primer paso del algoritmo de trazado de Moore	115
Figura 5.31	Búsqueda en sentido horario	116
Figura 5.32	Búsqueda de un tercer punto	116
Figura 5.33	Búsqueda de un cuarto punto	117
Figura 5.34	Fin del algoritmo de Moore	117
Figura 5.35	Diferentes formas con sus valores de circularidad	118
Figura 5.36	Ventana de monitoreo de la aplicación	120
Figura 5.37	UML para el algoritmo de la aplicación	120

INTRODUCCIÓN

De las leguminosas de grano seco, tenemos el frijol de palo o gandul, el cual es una menestra cultivada durante todo el año principalmente en el norte del Perú (Entre la región Lambayeque y Piura existen alrededor de 1500 hectáreas de este producto, con una producción por año que va de 7 a 13 mil toneladas). Gracias a la ola de consumo de productos naturales y nutritivos, sumada al boom de la exportación de menestras en la región Lambayeque, se registraron entre los años 2001 – 2002, las exportaciones de este frijol crecieron en 322%, llegando a representar el 1.26% de las exportaciones agrícolas peruanas. Hoy en día la exportación de menestras cayó en un 17% respecto al periodo anterior 2014 según la Asociación de Exportadores ADEX, debido a una menor producción por la falta de lluvias y cambio de temperatura, que afectaron los rendimientos y la calidad de los frijoles; sin embargo la menestra más exportada en el primer cuatrimestre del año 2015 fue el pallar que sumo US\$ 2.7 millones, luego le siguió el frijol de palo con US\$ 2.5 millones.

Las principales empresas envasadoras de este producto son Procesadora, Gandules, SV, Pulses Perú, Sun Packers y Congemar, siendo el principal destino de las menestras peruanas (frescas y conservadas) fue Estados Unidos, que las importó por US\$ 3.3 millones, con una caída de -9% y una concentración de 31% de los envíos totales; le sigue Colombia con US\$ 884 mil con un incremento de 391%; luego Turquía, Ecuador, Líbano, Canadá, España, Italia, Reino Unido, Panamá, Brasil, Trinidad y Tobago, Japón entre otros. A las grandes empresas mencionadas se han sumado varias pequeñas y medianas empresas (principalmente en el norte del país) que todavía cuentan con un sistema artesanal de selección y control de calidad del producto.

CAPITULO 1

1. DEFINICIONES PRELIMINARES

A continuación se presentarán las definiciones más importantes relacionadas con el desarrollo de este trabajo de tesis, se tratará temas como la visión artificial, sus aplicaciones, entre otros.

1.1 Visión Artificial

La visión artificial describe la deducción automática de la estructura y propiedades de un mundo tridimensional posiblemente dinámico, bien a partir de una o varias imágenes bidimensionales del mundo. Las imágenes pueden ser monocromáticas (de niveles de gris) o colores, pueden provenir de una o varias cámaras e incluso cada cámara puede estar estacionaria o móvil.

Las estructuras y propiedades del mundo tridimensional que queremos deducir en visión artificial incluyen no sólo sus propiedades geométricas, sino también sus propiedades materiales. Ejemplos de propiedades geométricas son la forma, tamaño y localización de los objetos. Ejemplos de propiedades de los materiales son su color, iluminación, textura y composición. Si el mundo se modifica en el proceso de formación de la imagen, necesitaremos inferir también la naturaleza del cambio, e incluso predecir el futuro.

La entrada a un sistema de visión artificial es una imagen obtenida por un elemento de adquisición, mientras que su salida es una descripción de la escena, la cual ha sido obtenida a partir de la imagen. Por un lado, esta descripción debe estar relacionada de algún modo con aquella realidad que produce la imagen y, por el otro, debe contener toda la información requerida para la tarea de interacción con el medio ambiente que se desea llevar a cabo, por ejemplo mediante un robot.

1.2 Aplicaciones de la Visión Artificial

El amplio espectro de aplicaciones cubierto por la visión artificial, se debe a que permite extraer y analizar información espectral, espacial y temporal de los distintos objetos. La información espectral incluye frecuencia (color) e intensidad (tonos de gris). La información espacial se refiere a aspectos como forma y posición (una, dos y tres dimensiones). La información temporal comprende aspectos estacionarios (presencia y/o ausencia) y dependientes del tiempo (eventos, movimientos, procesos).

La mayoría de las aplicaciones de la visión artificial podemos clasificarlas por el tipo de tarea, entre las que mencionaremos a continuación:

- **La medición o calibración;** se refiere a la correlación cuantitativa con los datos del diseño, asegurando que las mediciones cumplan con las especificaciones del diseño. Por ejemplo, el comprobar que un cable tenga el espesor recomendado.
- **La detección de fallas;** es un análisis cualitativo que involucra la detección de defectos o artefactos no deseados, con forma desconocida en una posición desconocida. Por ejemplo, encontrar defectos en la pintura de un auto nuevo, o agujeros en hojas de papel.
- **La verificación;** es el chequeo cualitativo de que una operación de ensamblaje ha sido llevada a cabo correctamente. Por ejemplo, que no falte ninguna tecla en un teclado, o que no falten componentes en un circuito impreso.
- **El reconocimiento;** involucra la identificación de un objeto con base en descriptores asociados con el objeto. Por ejemplo, la clasificación de cítricos (limones, naranjas, mandarinas, etc.) por color y tamaño.
- **La identificación;** es el proceso de identificar un objeto por el uso de símbolos en el mismo. Por ejemplo, el código de barras, o códigos de perforaciones empleados para distinguir hule de espuma de asientos automotrices.
- **El análisis de localización;** es la evaluación de la posición de un objeto. Por ejemplo, determinar la posición donde debe insertarse un circuito integrado.

Con el incremento de potencia de los microprocesadores, las aplicaciones que pueden resolverse con éxito utilizando sistemas de visión están creciendo rápidamente. Una aplicación puede exigir una o más funciones de procesamiento de imágenes, que cuando se combinan crean una solución. La gama de detección es muy amplia e incluye:

- **Forma o apariencia: (Control de la conformidad)**

Los sistemas actuales, por lo general, comienzan con una operación de medición bidimensional para establecer el desplazamiento necesario para que el objeto se encuentre en una posición ideal. Posteriormente, se llevan a cabo las operaciones de comparación con un objeto patrón o la caracterización geométrica de su forma. De esta manera, se puede decidir si el objeto analizado cumple o no con los requisitos preestablecidos.

- **Detección de defectos: (Elementos discretos)**

La detección de defectos puede ser considerada como un caso especial de control de la conformidad, en el que los objetos no ofrecen formas específicas caracterizables. Como ejemplos, citaremos la inspección de baldosas cerámicas en la etapa previa al esmaltado para detectar defectos en la superficie como rugosidades, o el uso de infrarrojos para la inspección de defectos de botellas de vidrio.

- **Detección de defectos: (Tejidos o laminados)**

La característica de muchos sistemas de este tipo es que se debe examinar grandes áreas a gran velocidad, en busca de defectos relativamente pequeños. El material producido en bobinas incluye la chapa de acero, papel, películas fotográficas, vidrio, hoja de plástico y los textiles. Algunos de estos sistemas utilizan el escaneado láser en lugar de cámaras lineales para formar la imagen.

- **Control por colores:**

La inspección por color se utiliza ampliamente en la industria farmacéutica para confirmar que la tableta o píldora de color se ha colocado en el embalaje correcto.

- **Medidas unidimensionales o bidimensionales:**

Las mediciones bidimensionales se pueden realizar mediante la superposición de calibres ópticos sobre la imagen del producto. La medición por este método es muy rápida porque no hay movimiento mecánico, y el posicionamiento exacto del objeto a medir no es crítico (el sistema puede determinar el desplazamiento X e Y del objeto y el grado de rotación). Algunos sistemas ofrecen interpolación sub-píxel que, en condiciones adecuadas, hace que el sistema sea capaz de medir y, en particular, detectar cambios o diferencias en la medición, de milésimas en lugar de las centésimas que ofrecen otros métodos de metrología. El número de píxeles de la imagen se puede aumentar mediante el uso de cámaras de muy alta resolución y obtener así mediciones aún más precisas.

- **Luz estructurada y otras técnicas de triangulación:**

Si una banda fina de luz se dirige a una superficie tridimensional y se observa con una cámara en un ángulo diferente al del dispositivo de iluminación, la forma aparente de la banda puede ser usada para inferir la forma de la superficie expuesta a lo largo de la longitud de la banda. Mediante la exploración la banda a través de la superficie, se puede generar un mapa tridimensional del objeto observado. A menudo es conveniente usar un láser como fuente de luz, dado que puede ser fácilmente refractado para formar un plano de luz.

- **Técnicas tridimensionales:**

También se puede obtener información tridimensional mediante el uso de visión estereoscópica automatizada.

- **Reconocimiento de caracteres:**

El reconocimiento de caracteres se puede basar en técnicas de correlación simple (comparación), sin conocimientos especializados de la forma en que

los caracteres individuales se forman, o se pueden usar algoritmos más avanzados de inteligencia artificial.

Los sistemas de visión también pueden ser usados para leer códigos de barras con un mayor grado de confianza que un escáner láser. Incluso si una gran parte del código de barras está oculto por suciedad, un sistema de visión, por lo general, puede leerlo, mientras que un lector de códigos de barras convencional. Además de códigos de barras, los sistemas de visión también se usan para leer códigos matriciales de dos dimensiones, donde los códigos pueden contener mucha más información que un código de barras unidimensional.

- **Reconocimiento de piezas o componentes del producto:**

El reconocimiento de piezas o componentes puede ser usado para identificar qué partes deben ser sometidas a que proceso. Por ejemplo, un sistema de visión puede discernir entre las diferentes piezas de un elemento e instruir a un robot para el correcto montaje del elemento.

- **Guiado predeterminado:**

El guiado predeterminado se caracteriza por una situación en la que una cámara azimutal toma una instantánea de la escena y el sistema de visión dirige un robot para que recoja o deje un objeto en una posición concreta. A continuación, el robot obra a ciegas. Una aplicación típica incluye tareas de paletización de objetos pesados, tales como bloques de motor y cigüeñales, pero también puede incluir trabajos mucho más ligeros tales como el embalaje chocolates.

- **Guiado continuo:**

El guiado continuo implica una cámara montada sobre el brazo o la mano del robot y el camino del robot está continuamente corregido por el sistema de visión. Una aplicación común es el guiado de la soldadura por arco, pero la técnica puede utilizarse también para controlar el camino la aplicación de selladores o pegamentos por robot.

1.3 Visión Artificial en la Robótica

El sentido de visión en los robots industriales es el elemento esencial que permite a estos presentar características de adaptabilidad, flexibilidad y capacidad de reorganización.

La visión juega un papel muy importante en los sistemas de manipulación automática que merezcan el calificativo de inteligentes y en general en los sistemas flexibles de manufactura, pues permite la retroalimentación sensorial fina que hace ampliar las capacidades de los robots.

La visión artificial proporciona la descripción del estado que guardan los elementos del puesto de trabajo, así como su evolución en el tiempo, información que el sistema de control del robot utiliza en la generación y modificación de sus planes de trabajo, en el monitoreo de la ejecución de tareas y en la detección de errores e imprevistos.

El empleo de sistemas de visión artificial en la manipulación controlada sensorialmente permite, por un lado resolver problemas de conocimiento a priori del ambiente, precisión, costo y fiabilidad y por otro lado, permite a los robots industriales evolucionar en ambientes variables.

En este contexto, un Sistema de visión artificial debe realizar las siguientes funciones:

- Reconocimiento de piezas o conjuntos, así como sus posiciones de equilibrio.
- Determinación de la posición y orientación de piezas con relación a una referencial base.
- Extracción y ubicación de rasgos significativos de las piezas, con objeto de establecer servomecanismos visuales que permitan su manipulación robotizada.
- Inspección en línea y verificación de que el proceso ha sido realizado satisfactoriamente (control de calidad sin contacto).

Por otro lado, las aplicaciones de la visión artificial en los robots controlados sensorialmente, son básicamente los siguientes:

- Manipulación de objetos aislados acarreados por bandas transportadoras: normalmente las piezas tienen una posición de equilibrio única y presentan una proyección fácilmente identificable (aun cuando esté en contacto con otra pieza) mediante técnicas de reconocimiento.
- Manipulación de objetos acomodados aleatoriamente en contenedores: en este caso los objetos presentan una proyección no única, parcialmente oculta, su posición y orientación son aleatorias. Por tanto requiere de una potencia de cálculo mayor y de algoritmos de tratamiento de imágenes más sofisticados.
- Ensamble: esta tarea se resume en los siguientes 4 puntos: permite identificar piezas, tomarlas y presentarlas en una forma predeterminada para ensamblarlas con otras piezas; provee retroalimentación visual en el posicionamiento dinámico de las herramientas de ensamble; provee control continuo del órgano terminal sobre la trayectoria deseada; y permite la inspección en línea.

1.4 Visión Artificial en la Industria

La visión artificial aplicada a la industria abarca la informática, la óptica, la ingeniería mecánica y la automatización industrial. A diferencia de la visión artificial académica, que se centra principalmente en máquinas basadas en el procesamiento de imágenes, las aplicaciones de visión artificial industrial integran sistemas de captura de imágenes digitales, dispositivos de entrada/salida y redes de ordenador para el control de equipos destinados a la fabricación tales como brazos robóticos. Los sistemas de visión artificial se destinan a realizar inspecciones visuales que requieren alta velocidad, gran aumento, funcionamiento las 24 horas del día o la repetitividad de las medidas. Además también se tiene aplicaciones de la visión artificial en la industria como se describe a continuación:

- **Industrias de la Alimentación:**

En el sector de la alimentación es una industria en que la automatización industrial ha tardado más en introducirse debido a la diversidad de formas que presentan los productos naturales. Con los sistemas de inspección por visión artificial que se puede desarrollar, se podrá mejorar los procesos productivos de la industria, de la alimentación, ya se manipule carne, pescado, verduras, frutas, hortalizas, lácticos o bien productos ya elaborados.

Por ejemplo:

a) Para el control de calidad del cierre de latas de sardinas, puede tener fallos en el cierre, especialmente de las latas de aluminio. Un defecto en el cierre de una lata de conserva que llega al mercado, puede provocar efectos en la salud pública, por no citar la mala imagen de marca que esto genera.

Un fabricante de conservas, concienciado con este problema, ha instalado un sistema de inspección del cierre de la lata por visión artificial que verifica todo el contorno del cierre de cada una de las latas que produce. Esta instalación ha permitido reubicar a los 24 operarios que realizaban esta tarea, necesarios para cubrir el ritmo de producción (200 latas/min) de las 4 líneas instaladas.

Un sistema de visión artificial de última generación, compuesto por 6 cámaras sincronizadas, es el encargado de inspeccionar el cierre de latas en línea, es decir, controla el 100% de la producción en tiempo real, sin alterar el ritmo de producción. La inspección del cierre se realiza a una velocidad de hasta 600 latas/min. y es capaz de detectar los defectos más pequeños como se muestra en las siguientes figuras:

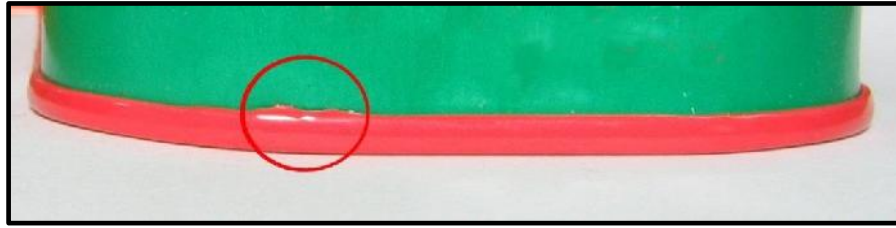


Figura 1.1 Defecto de falso cierre.

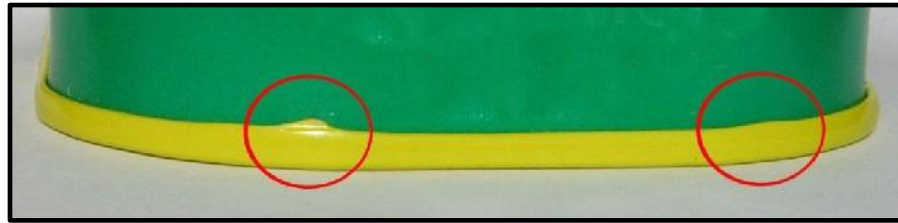


Figura 1.2 Defecto de pico y rebabas.



Figura 1.3 Defecto de pestañas golpeadas.

Las principales ventajas del uso de la visión artificial aplicada al control de calidad son:

- **Inspección del total de la producción:** Debido a la elevada cadencia que puede alcanzar la inspección con cámaras, la visión artificial para el control de calidad se puede usar para inspeccionar cada una de las piezas producidas en lugar de hacer muestreos aleatorios de los lotes.
- **Repetibilidad:** Con la visión industrial para el control de calidad, cada una de las inspecciones es realizada bajo las mismas condiciones, asegurando así la repetibilidad de cada uno de los parámetros de aceptación del producto.

- **Consistencia:** La inspección automatizada evita la subjetividad característica del ser humano y el cansancio, por lo que cada una de las piezas es consistente a nivel de calidad con todas las demás.
- **Reducción de costes:** Son evidentes las ventajas del control de calidad al final del proceso productivo respecto a la reducción de costes, servirá a su cliente productos libres de defectos, evitando devolución de lotes y mejorando su imagen. Con la inspección de los parámetros de calidad con cámara en puntos intermedios en su proceso productivo, le permitirá detectar con antelación piezas defectuosas y retirarlas antes de ser ensambladas, evitando así el rechazo completo del producto final. También obtendrá beneficios debido a la recolocación de recursos humanos previamente destinados a asegurar la calidad del producto.

b) Versatilidad en la línea de producción para la fabricación de salchichas, para satisfacer la demanda del mercado de productos variados, un fabricante de salchichas ha instalado un sistema robotizado controlado por visión artificial para adaptar la línea de producción a los diferentes tamaños y variedades demandados por el consumidor rápidamente.

El sistema consiste en dos subsistemas de visión industrial. Uno se encarga de guiar a un robot para que escoja la salchicha adecuada para proceder a su envasado. Posteriormente, otro subsistema de visión, verifica que el paquete este completo y permite su sellado.

- **Industria de Automoción:**

Los elevados requisitos de control de calidad y automatización industrial que el sector de la automoción se impone a sí mismo como a sus proveedores, hacen de este sector un gran consumidor de soluciones de visión artificial.

Por ejemplo:

a) Para el control de calidad de piezas de plástico del elevallunas,

Como proveedor de empresas de automoción, la calidad de producto pasa a un primer plano. Un fabricante de molduras de plástico entrega a su cliente piezas libres de defectos con una confianza

del 100%. El fabricante de molduras de plástico instaló un sistema de visión artificial al final de su proceso productivo para comprobar defectos geométricos y la falta o exceso de material plástico en cada una de las piezas entregadas.

b) Guiado de robots para colocar tuercas en las ruedas, para aumentar la velocidad de una cadena de montaje, una empresa ha automatizado el sistema de colocación de tuercas en las ruedas durante el proceso de producción. La compañía utiliza el sistema de visión artificial para determinar la posición de la rueda, incluyendo las posiciones frontal y trasera y de lado a lado, además de tres ejes rotacionales. Con dicha información el robot mueve el dispositivo de apriete de tuercas y lo sitúa en posición para apretar las tuercas.

Se trata de un complejo trabajo manual debido tanto al tamaño y peso del dispositivo del apriete, como a la necesidad de colocar las tuercas en dos ruedas en aproximadamente 40 segundos. El vehículo es colocado en posición por el transportador, y las ruedas tienen libre movimiento para rotar, inclinarse y girar.

La aplicación cuenta con dos sistemas de visión para situar cada rueda. Una vez que el sistema identifica el tipo de la rueda, se utiliza un software para buscar círculos y determinar el centro del eje. Una herramienta de visión inspecciona el centro de la llanta para determinar el ángulo de rotación de la rueda. La información es enviada al controlador de un robot, que ajusta los ángulos de inclinación y giro de la rueda y rota el dispositivo de apriete para ajustar el ángulo. Después dicho dispositivo es guiado directamente sobre los espárragos y aprieta las tuercas con el par de torsión adecuado

- **Industria de la Electrónica:**

La industria de la electrónica se caracteriza por el elevado ritmo de producción y automatización. La visión artificial se aplica en diferentes áreas del proceso productivo, desde la manipulación e identificación de componentes hasta el control de calidad.

Por ejemplo:

a) Identificación de componentes en el montaje de interruptores, La mayor parte del proceso de producción de interruptores se realiza manualmente. La inspección por parte de humanos puede llevar a errores debidos al cansancio o la repetibilidad de las operaciones. Un fabricante de interruptores decidió incorporar un sistema de visión artificial para comprobar que cada pieza es la correcta y que cada producto se empaqueta con el código de referencia que le corresponde. Este fabricante a reducido a cero la devolución de pedidos debido a errores en la marcación.

b) Control de calidad final en la inspección de píxeles en pantallas LCD, la gran demanda de pantallas LCD, tanto para ordenadores como para televisores, requiere unos altos niveles de calidad para satisfacer a los consumidores. Las pantallas LCD sufren, al final del proceso de producción, estrictos controles de calidad. Para determinar que todos los píxeles que componen la pantalla LCD funcionan correctamente se usa un sistema de visión artificial de alta resolución.

- **Industria Farmacéutica:**

En la industria farmacéutica se tiene los siguientes ejemplos:

a) Dosificación de medicamentos personalizada, a través de guiado de robots y trazabilidad. La unidad de farmacia de un hospital tiene una gran responsabilidad en la correcta distribución de medicamentos. Estos se sirven en monodosis sin desprecintar y se reenvasan para hacerlos llegar al paciente destinatario sin posibilidad de errores. Para llevar a cabo esta tarea, el hospital instalo un sistema automatizado que corta los blíster y codifica las monodosis con código matricial para garantizar su trazabilidad. Estos medicamentos en formato monodosis son almacenados para su posterior reenvasado y etiquetado para que lleguen al paciente al que le han sido prescritos.

El mecanismo de corte consiste en un láser guiado por un sistema de visión artificial, dada la variabilidad de los formatos de blisters existentes.

La dosificación posterior se realiza mediante un robot y es comprobada por un sistema de visión.

- b) **Control de envasado de blíster**, a través de control de presencia. En el envasado de píldoras, pastillas o cápsulas no cabe el error. Por eso, durante este proceso un sistema de visión artificial, equipado con una iluminación especial que evita los reflejos, comprueba la correcta colocación de los medicamentos y que el blíster está completo.

1.5 Usos de la Visión Artificial

Las aplicaciones de visión artificial se dividen en tres grandes categorías:

- Control de procesos
- Control de calidad
- Aplicaciones no industriales

Las inspecciones realizadas por los seres humanos, a menudo no pueden cumplir con los requisitos de la industria moderna respecto a la velocidad de producción, calidad de producto y costes de producción. Los humanos se cansan, cometen errores y los criterios que se aplican durante las inspecciones son inevitablemente subjetivos. En algunos casos, no es humanamente posible llevar a cabo la tarea de inspección debido a las condiciones ambientales. Las cámaras y los sistemas que componen un sistema de visión artificial, por el contrario, llevan a cabo las mediciones con una precisión constante y a un ritmo que es establecido por el propio proceso de producción. Estas ventajas han llevado a una creciente aceptación de la visión por industrias de todo el mundo.

Aplicaciones de esta tecnología en instalaciones alrededor del mundo incluyen:

- La inspección de la calidad óptica de las pantallas de los televisores y los ordenadores.
- La inspección de la calidad de la pintura durante la fabricación de automóviles.
- La inspección de los billetes de banco durante la impresión.

- Comprobación de circuitos electrónicos.
- Comprobación de que los envases farmacéuticos estén completos.
- Inspección para asegurarse de que las botellas estén llenas.
- Comprobación de defectos en azulejos en la industria de la cerámica.

La visión artificial se refiere a la interpretación automática de imágenes de escenas reales con el fin de controlar o vigilar máquinas o procesos industriales. Las imágenes pueden ser de luz visible, rayos X o infrarroja, y puede incluso derivarse de información ultrasónica.

1.6 Componentes de un Sistema de Visión Artificial

Un moderno sistema de visión industrial artificial consta de:

- **Un sistema de iluminación:**

Una buena iluminación es especialmente importante para la toma de imágenes de los productos en una línea rápida de producción, aunque algunas aplicaciones pueden utilizar la luz ambiente.

- **La lente de la cámara:**

La correcta selección de lentes es importante para alcanzar una solución óptima.

- **Uno o más cámaras para adquirir las imágenes:**

Las cámaras pueden ser analógicas, pero el precio de las cámaras digitales está disminuyendo, de modo que estas se están usando más a menudo.

- **Un dispositivo de interfaz para transferir las imágenes al ordenador.**

- **Un procesador de imagen, ordenador o cámara inteligente:**

Una opción es utilizar cámaras inteligentes que integran el procesamiento de imágenes dentro de la propia cámara, evitando la necesidad de transferir imágenes a un ordenador externo. La velocidad de proceso de estas cámaras es inferior a la de un ordenador y existen aplicaciones en las que estas no son adecuadas.

- **Una interfaz para notificar el resultado del análisis a un operador:**

Se puede notificar de este resultado mediante una señal electrónica que opera un mecanismo de rechazo.

La imagen de entrada -una matriz bidimensional de niveles energéticos (por ejemplo, luz) se divide en elementos de imagen, conocidos como píxeles. Estas forman filas y columnas que abarcan toda la zona de la imagen y representan los niveles de gris en una imagen monocromática o la codificación de color en una imagen en color. Un pixel no puede ser subdividido en regiones de menor nivel de gris o color. Este proceso es un tipo de digitalización espacial. Para cada píxel, la información del nivel de energía también debe ser digitalizada, es decir los niveles analógicos (variable continua) producidos por la cámara deben ser representados por un número finito de pasos. En muchas aplicaciones es suficiente digitalizar una imagen monocroma con 8 bits por píxel, lo que equivale a 256 pasos, para representar el nivel de gris de cada píxel. En aplicaciones más exigentes puede ser necesario digitalizar a 14 bits (ó 16384 niveles). Las imágenes en color son más complejas y pueden ser representados en diferentes formatos. Las imágenes en color normalmente contienen tres veces más información que una imagen monocromática.

Algunos sistemas de visión no utilizan una cámara matricial, en su lugar se usa una cámara lineal que produce una sola línea o fila de píxeles. La imagen bidimensional se genera a medida que el objeto pasa bajo la cámara lineal, aprovechando su movimiento, normalmente generado por una cinta transportadora. Uniendo las distintas filas de píxeles obtenidas a diferentes intervalos de paso, se obtiene una imagen bidimensional.

1.7 Cómo Funciona la Visión por Computador

Los pasos básicos son la segmentación y el análisis, que son esencialmente algoritmos de software y funciones que se ejecutan en el procesador.

- **Segmentación:**

Se trata de decidir qué partes de la imagen necesitan interpretación o análisis y cuáles no. A menudo es posible y necesario refinar la segmentación. Por ejemplo, si la aplicación debe encontrar arañazos o grietas en un vidrio, la primera fase de segmentación suele encontrar

elementos que son las propias grietas o arañazos además de suciedad y polvo. A menudo se perfecciona este resultado para eliminar de la imagen la suciedad y el polvo de los elementos a analizar.

- **Análisis:**

Una vez que la imagen ha sido satisfactoriamente segmentada, el procesador puede hacer una serie de pruebas y mediciones en los elementos de interés que aparecen en la escena. Existen multitud de algoritmos que nos ayudan a obtener los resultados deseados. Entre ellos, cabe destacar sus funcionalidades:

- Detección de formas
- Análisis geométrico
- Comparación con patrones
- Medición de objetos
- Detección de objetos difusos

Estos algoritmos, usados estratégicamente, nos permiten decidir si una pieza se ha fabricado correctamente o no, guiar un robot hacia un objeto concreto o reajustar la maquinaria de producción automáticamente.

1.8 Justificación de la Inversión en Visión Artificial

Hay muchas justificaciones para la utilización de la visión artificial:

- **Costes de los materiales:**

En la mayoría de las aplicaciones, evitar la producción de piezas defectuosas mediante el uso de un sistema de visión industrial tendrá un período de amortización muy corto. Para evitar que se fabriquen piezas defectuosas, el sistema de inspección automática, ya sea muestreando el 100% en la línea de producción o bien usado fuera de línea tomando muestras, debe formar parte del control estadístico de procesos (SPC) del sistema productivo. Esto significa que el sistema indica cuando un parámetro de control deriva hacia el límite de tolerancia, o es simplemente demasiado errático. El sistema de visión puede tomar medidas correctivas antes de que el límite sea superado.

- **Costes de la mano de obra:**

La reducción de la mano de obra es también un importante ahorro de costes, ya que muchas de las tareas realizadas por la visión industrial pueden sustituir a personas directamente. Además, deben ser considerados los ahorros en selección de personal, prestaciones sociales y los aumentos salariales anuales.

- **Costes de la calidad:**

La creciente conciencia del costo de la calidad a través de las normas ISO9001, significa que el uso de la visión artificial puede ofrecer un estándar más objetivo, fiable y consistente en la inspección de productos.

El ahorro en la optimización del uso de materiales, seguimiento de la calidad de los proveedores y garantía de calidad de los productos acabados pueden llevar a ahorros tanto tangibles como intangibles. El costo de los trabajos de reparación en garantía se puede reducir y además se mejora la confianza de los clientes consiguiendo pedidos recurrentes y una mayor cuota de mercado.

CAPITULO 2

2. ETAPAS DE UN SISTEMA DE VISIÓN ARTIFICIAL

En un sistema de visión artificial se incluyen diversas técnicas, tales como el procesamiento de imágenes (captura, transformación, codificación de imágenes) o como el reconocimiento de formas (teoría estadística de decisiones, enfoques sintácticos y neuronales aplicados a la clasificación de patrones). En este tipo de sistemas, además se incluyen técnicas de modelado geométrico y procesos de conocimiento. En vista a. esto, en este capítulo se trata de describir las etapas a considerar en el sistema de visión a implementar.

2.1 Adquisición de la Imagen

El sistema permite la captura y/o adquisición de la imagen está formada por los siguientes elementos:

2.1.1 Cámara

Es el dispositivo encargado de transformar las señales luminosas que aparecen en la escena, en señales analógicas capaces de ser transmitidas por un cable coaxial. Se divide en dos partes, el sensor, que captura las propiedades del objeto en forma de señales luminosas y lo transforma en señales analógicas, y la óptica que se encarga de proyectar los elementos adecuados de la escena ajustando una distancia focal adecuada.

Los sensores de visión usados más recientemente son los basados en matrices de dispositivos acoplados por carga CCD; estos transductores proporcionan una señal con amplitud proporcional a la luminosidad de la escena y realizan una digitalización espacial completa en dos dimensiones (líneas y columnas), pues descomponen la imagen en una matriz de puntos.

La codificación de la brillantez de cada elemento de imagen o pixel, obtenido de la digitalización espacial, se hace generalmente en 8 bits, mientras que la resolución de la discretización espacial de una imagen puede ser por ejemplo de 320x240 pixeles.

La tecnología CCD interline transfer (IT) y frame interline transfer (FIT) identifica el tipo de CCD, cada uno de ellos tiene aspectos positivos y negativos. En la práctica el fabricante tiene optimizado el diseño y el tipo de CCD usado raramente determina el funcionamiento completo de la cámara.

- **CCD tipo IT:**

La tecnología IT (interline transfer) tiene registros separados, protegidos de la luz con una máscara de aluminio opaco ópticamente, como se observa en la figura 2.1. Las cargas proporcionales al contenido de la escena se acumulan en cada elemento del arreglo del sensor. Durante el intervalo vertical, los paquetes de carga son desplazados al arreglo de almacenamiento adyacente. Luego, los elementos del sensor ahora vacíos, capturan el próximo campo mientras la información del arreglo de almacenamiento se transmite fuera para formar la señal de video de salida.

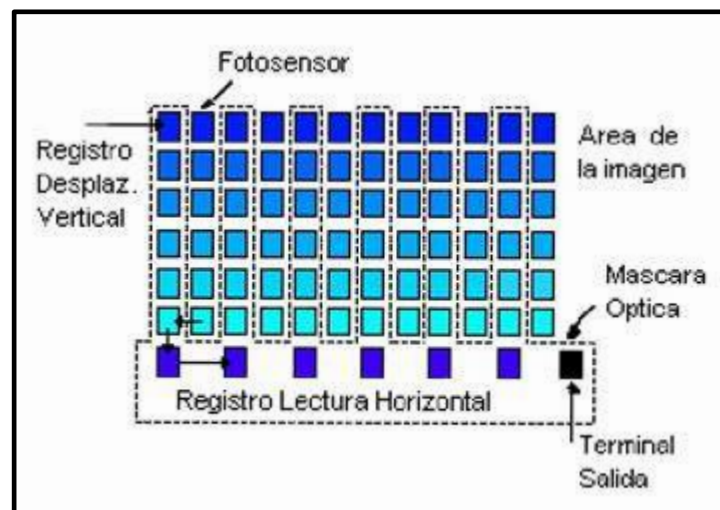


Figura 2.1 CCD tipo IT.

- **CCD tipo FIT:**

La parte superior de este dispositivo opera como un CCD IT. Sin embargo las cargas son rápidamente desplazadas desde el registro de almacenamiento interline al registro de almacenamiento protegido totalmente. Los paquetes de carga son mantenidos en el registro interline solo por un corto tiempo.

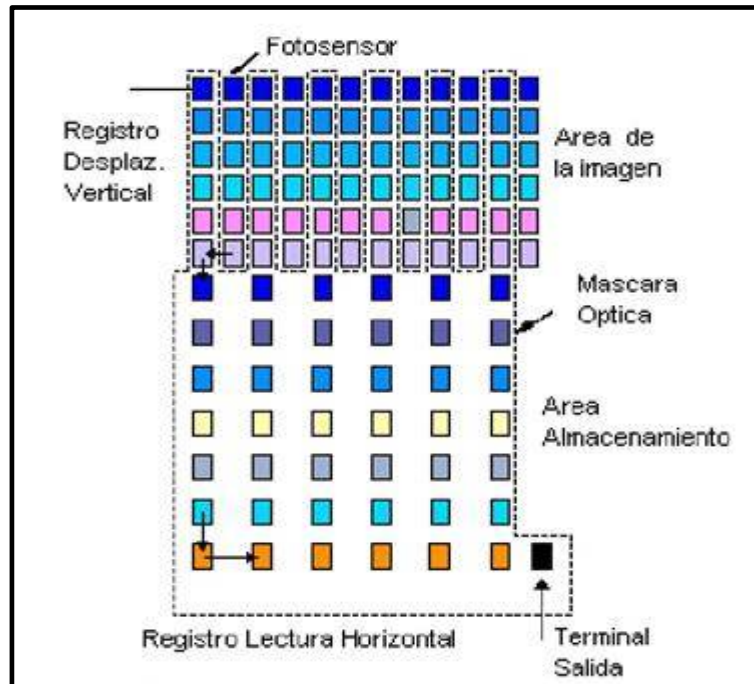


Figura 2.2 CCD tipo ITF.

2.1.2 Digitalizador

El digitalizador (frame grabber), es el encargado de transformar la señal de vídeo, cualquiera que sea el formato utilizado (NTSC, PAL), en una señal digital capaz de ser capturada, almacenada en memoria y procesada por una computadora. Las principales características de las tarjetas digitalizadoras son precio, controlabilidad, resolución, velocidad y almacenaje, en el sentido de si los algoritmos de visión pueden tener acceso rápido y fácilmente a los datos.

Existen tarjetas que proporcionan sus propios buffers de memoria y otras que utilizan la memoria del ordenador (vía DMA). Muchas de ellas permiten un preprocesamiento previo de las imágenes, donde el número de tareas implementadas en hardware es muy variable.

La resolución de las tarjetas digitalizadoras y la de las cámaras (sensor) no tiene por qué coincidir. Por lo tanto, es importante (sobre todo cuando se emplean técnicas de medición) saber que ocurre con los puntos que faltan o sobran.

2.1.3 Sistema de Iluminación

La iluminación juega un papel vital dentro de cualquier sistema de visión artificial ya que proporciona las condiciones ópticas bajo las cuales se lleva a cabo la adquisición de la imagen. No obstante, a pesar de su enorme trascendencia, la selección de un buen medio de iluminación suele ser una de las áreas más frecuentemente descuidadas.

La aplicación de una iluminación adecuada a la naturaleza del objeto a inspeccionar y a sus condiciones ambientales, será determinante en el éxito de una aplicación de inspección automática. Un enfoque para un sistema de inspección que no preste suficiente atención a la iluminación supondrá una mayor complicación en el análisis e interpretación de la imagen captada. La iluminación es, por tanto, un factor que afecta radicalmente a la complejidad de los algoritmos de visión y a la fiabilidad del sistema. Es indiscutible que el éxito de una aplicación industrial depende tanto del diseño de la iluminación como del procesamiento de imagen.

2.1.3.1 La luz

El comprender el comportamiento físico de la luz es importante ya que los datos de una imagen en sí no son más que la distribución espacial de la luz de una escena. El principio de cómo la información se transfiere de un objeto a un sensor se basa en cómo interactúan los fotones con el material del objeto observado.

Si el objeto modifica la luz entrante de manera que los rayos salientes sean diferentes de los rayos incidentes, entonces se dice que el objeto origina contraste. Éste es el principio básico de todos los sistemas de visión ya sean biológicos o artificiales. Si el objeto no modifica el rayo de luz incidente de una manera apreciable, entonces dicho objeto no es visible.

La luz es la radiación electromagnética que estimula nuestra respuesta visual. Se expresa como una distribución de energía espectral, donde es la longitud de onda perteneciente al espectro electromagnético visible (de 380 nm a 770 nm). La luz que se recibe desde un objeto opaco puede ser escrita como:

$I(\lambda) = R(\lambda) * L(\lambda)$	Ecuación 2.1
--	---------------------

Donde:

- $R(\lambda)$: Factor de reflexión del objeto
- $L(\lambda)$: Distribución de energía incidente

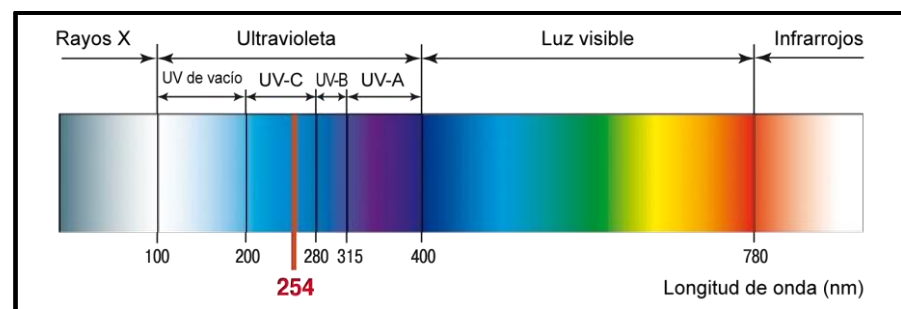


Figura 2.3 Espectro electromagnético

El diseñador de un sistema de visión dispone de gran cantidad de tecnologías en cuanto a fuentes de iluminación que puede emplear bajo distintas configuraciones. El objetivo va a ser siempre optar por aquella solución que aumente al máximo el

contraste en la imagen obtenida, de forma que las características del objeto a inspeccionar sean más fácilmente identificables. Este contraste se detecta al ser reflejada la luz desde la superficie de los objetos y por tanto, a la hora de elegir de un sistema de iluminación, es fundamental saber que al iluminar hay una parte de la luz que es absorbida y otra parte que es reflejada o transmitida y esta la que constituye la entrada al sistema de visión.

Cuando la luz encuentra un obstáculo en su camino choca contra la superficie de este y una parte es reflejada. Si el cuerpo es opaco el resto de la luz será absorbida. Si es transparente una parte será absorbida como en el caso anterior y el resto atravesará el cuerpo transmitiéndose. Así, podemos considerar tres fenómenos:

- Reflexión
- Absorción
- Transmisión



Figura 2.4 División del rayo de luz

- Reflexión:

La reflexión de la luz se produce cuando esta choca contra la superficie de separación de dos medios diferentes ya sean gases,

líquidos o sólidos y está regida por la ley de la reflexión. La dirección en que sale reflejada la luz viene determinada por el tipo de superficie. Si es una superficie brillante o pulida se produce la reflexión regular en que toda la luz se refleja en una única dirección. Si la superficie es mate la luz se dispersa en todas direcciones y se conoce como reflexión difusa como ocurre por ejemplo en una hoja de papel no satinado. Y, por último, está el caso intermedio, la reflexión mixta, en que predomina una dirección sobre las demás. Este tipo de reflexión aparece en superficies metálicas sin pulir, barnices, papel satinado, etc. El factor de reflexión de una superficie se da como el cociente de la luz reflejada entre la radiación del incidente:

$\text{Factor de Reflexión} = \frac{\text{Luz Reflejada}}{\text{Luz Incidente}}$	Ecuación 2.2
--	--------------

El factor de reflexión es un número entre cero y uno, aunque se presenta a veces como un porcentaje.

- **Absorción:**

Cuando la luz choca con un objeto opaco, una parte de la luz que incide es absorbida por la superficie y otra reflejada. Las longitudes de onda que son reflejadas son las que determinan el color que percibimos. La absorción del material se expresa como el cociente de la luz absorbida con la radiación del incidente:

$\text{Factor de Absorción} = \frac{\text{Luz Absorbida}}{\text{Luz Incidente}}$	Ecuación 2.3
--	--------------

El factor de absorción es también un número entre cero y uno, y se da a veces como porcentaje. Un material con alta absorción, tal como un pedazo de filtro negro tendrá valores próximos a uno. La absorción es un proceso muy ligado al color. Las longitudes de

onda que son reflejadas son las que determinan el color que percibimos. En los extremos, si el cuerpo la refleja toda aparece como blanco y si las absorbe todas como negro. Un objeto es rojo porque refleja longitudes de onda en torno al rojo y absorbe todas las demás. Si se ilumina el mismo objeto con luz azul lo veremos negro porque el cuerpo absorbe esta componente y no refleja ninguna.

COLOR	LONGITUDES DE ONDA (nm)
Violeta	380 – 435
Azul	435 – 495
Verde	495 – 565
Amarillo	565 – 590
Naranja	590 – 625
Rojo	625 - 780

Tabla 2.1 Longitud de onda reflejada del color

- Transmisión:

La transmisión se puede considerar una doble refracción. Si pensamos en un cristal; la luz sufre una primera refracción al pasar del aire al vidrio, sigue su camino y vuelve a refractarse al pasar de nuevo al aire. Si después de este proceso el rayo de luz no es desviado de su trayectoria se dice que la transmisión es regular como pasa en los vidrios transparentes. Si se difunde en todas direcciones tenemos la transmisión difusa que es lo que pasa en los vidrios translúcidos.

Este fenómeno se presenta en materiales que son transparentes, translúcidos, u opacos con los agujeros físicos. En estos tipos de materiales, una cierta cantidad de radiación del incidente (I) que llega a una superficie será transmitida a través del material. El factor de transmisión del material se representa como el cociente de la luz transmitida con la radiación del incidente.

$\text{Factor de Transmisión} = \frac{\text{Luz Transmitida}}{\text{Luz Incidente}}$	Ecuación 2.4
--	---------------------

El factor de transmisión es un número entre cero y uno, y se da a veces como porcentaje. Un material con buena transmisión, tal como un pedazo de cristal óptico de calidad, puede alcanzar valores próximos a la unidad.

- Relación entre la reflexión, la absorción y la transmisión:

Sobre un objeto real la luz interaccionará combinando los distintos mecanismos dependiendo del factor de reflexión, absorción, y transmisión que presente el objeto. De cualquier forma y en todo caso, hay una relación que va a cumplirse siempre: la conservación de energía. La suma de todas las formas de radiación saliente debe ser igual a la cantidad de radiación entrante que afecta al material bajo inspección.

$\text{Luz Incidente} = \text{Luz Reflejada} + \text{Luz Absorbida} + \text{Luz Transmitida}$	Ecuación 2.5
---	---------------------

Como normalmente se iluminan objetos opacos se tendrá la siguiente relación:

$\text{Luz Incidente} = \text{Luz Reflejada} + \text{Luz Absorbida}$	Ecuación 2.6
--	---------------------

Cada material tiene un cierto factor característico de reflexividad, de absorción, y de transmisión que permite que se modifique la luz incidente y que pase la información a un detector. La diferencia en alguno de estos parámetros entre dos materiales va a posibilitar discriminarlos como distintos. Por el contrario, si los dos materiales presentan índices muy similares en los tres índices será imposible diferenciarlos visualmente.

La interacción de la luz con la materia que se ha expuesto, aunque ilustra el comportamiento básico de la luz y es suficiente para el diseño de sistemas de iluminación, se ha realizado de manera muy simplista. Hay factores que no se han considerado que hacen que los mecanismos de interacción de la luz sean más complicados: los parámetros de la reflexión, absorción, y transmisión varían en función de la longitud de onda de la luz incidente, y en función del ángulo de incidencia de la luz.

2.1.3.2 Características de las Fuentes de Iluminación

En las aplicaciones de visión artificial, la selección de una fuente de iluminación depende de numerosos factores: geometría y características superficiales del objeto a inspeccionar, velocidad de la aplicación, hostilidad del entorno, que hacen que tengamos que optar por sistemas de iluminación con unas características determinadas de distribución espectral, de patrón de radiación (puntual, lineal o hemisférico) y de distribución de la luz (focalizada, difusa o colimada) además de considerar otros factores como robustez, estabilidad y coste. A continuación veremos a qué se refieren estas características para poder establecer el perfil del sistema más idóneo entre la multitud de fuentes y dispositivos de iluminación.

- Nivel de Luz:

En principio todo sistema de visión artificial industrial debe estar apantallado bajo un carenado que evite en lo posible cualquier entrada de luz ambiente. La luz ambiente de las plantas industriales es muy variable y va a suponer una perturbación que con toda seguridad hará fracasar al sistema. No obstante, algunas veces hay problemas para apantallar completamente la aplicación. En estos casos será especialmente importante disponer de una fuente de luz con buena intensidad para que las

variaciones ambientales queden enmascaradas por la mayor potencia de la iluminación y no tengan efecto sobre la imagen.

Otra situación donde también es preciso tener muy en cuenta el nivel de luz es cuando se trabaja en líneas donde los productos se desplazan a alta velocidad.

En estas aplicaciones es necesario emplear velocidades de obturación elevadas en la cámara para no obtener imágenes movidas. Cuando se emplean velocidades altas de obturación se requiere luz más intensa pues la escena observada es expuesta al sensor durante un periodo de tiempo más corto. En aplicaciones de alta velocidad hay que tener muy en cuenta este parámetro a la hora de seleccionar la fuente de iluminación y optar por aquella que proporcione una luz suficientemente potente como son los flashes.

- Distribución Espectral:

La distribución espectral hace referencia al conjunto de longitudes de onda que emite la fuente de luz. Un cambio en la distribución espectral de la fuente de iluminación origina cambios sustanciales en la imagen capturada por la cámara. Si se trata de una aplicación en color la luz debe ser blanca, no puede ser monocromática pues distorsionaría los colores de la escena. Ya hemos visto que el color percibido por un captador depende de la frecuencia de la luz con que iluminamos.

La distribución de la fuente de luz debe estar contenida dentro de la respuesta espectral de la cámara porque de otra forma la luz proyectada no excitaría el sensor de imagen. En el mercado existen fluorescentes, lámparas halógenas, láseres, LEDs o fibra óptica (utilizando filtros de color) con distintas respuestas espectrales.

Por otra parte, aunque la mayor parte de las aplicaciones de visión artificial utilizan el intervalo del espectro visible, hay otras

zonas del espectro no visibles al ojo humano, como el infrarrojo al que sí son sensibles las cámaras, que están siendo utilizados en nuevos campos de aplicación. Por ejemplo, la utilización de LEDs NIR (infrarrojo cercano) y filtros que bloqueen la radiación visible en la óptica de la cámara proporciona muy buenos resultados en aplicaciones monocromáticas donde sea difícil de aislar completamente al sistema de la luz ambiente. De esta forma las variaciones en lumínicas del entorno no van a afectar tanto a la aplicación, ya que la luz presente en las plantas industriales no contiene una componente infrarroja significativa.

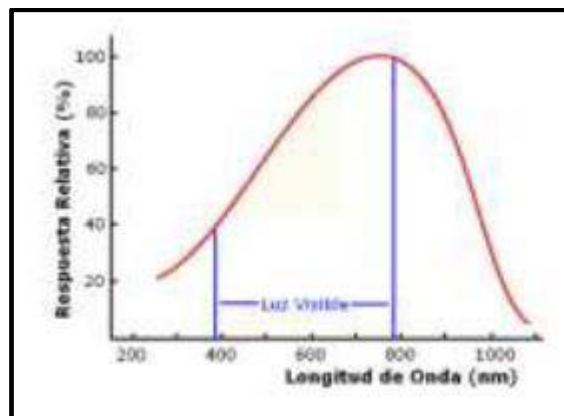


Figura 2.5 Sensibilidad espectral de un sensor CCD típico

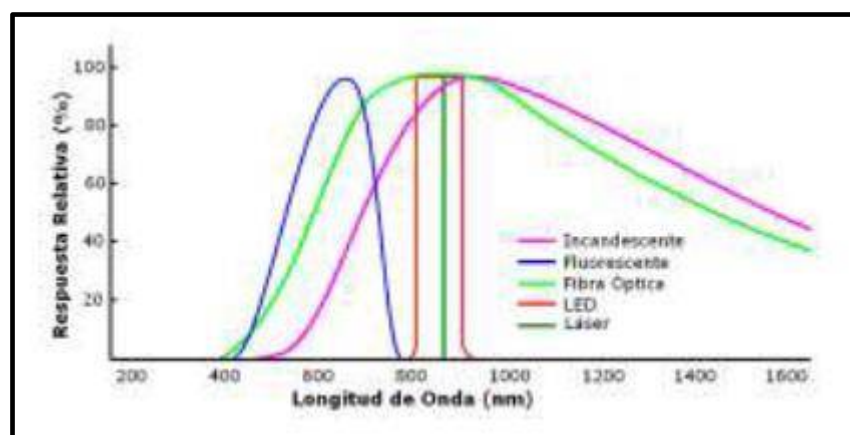


Figura 2.6 Distribución espectral de varias fuentes de luz

- **Patrón de Radiación:**

Las fuentes de luz pueden también clasificarse de acuerdo con su patrón de radiación que puede ser puntual, lineal, anular, superficie o hemisférico. Los patrones rectilíneos son especialmente utilizados en inspección de productos continuos como chapa en trenes de laminación, rollos de papel y de tela, etc. Ya que las elevadas velocidades de producción de estos productos requieren el empleo de cámaras lineales

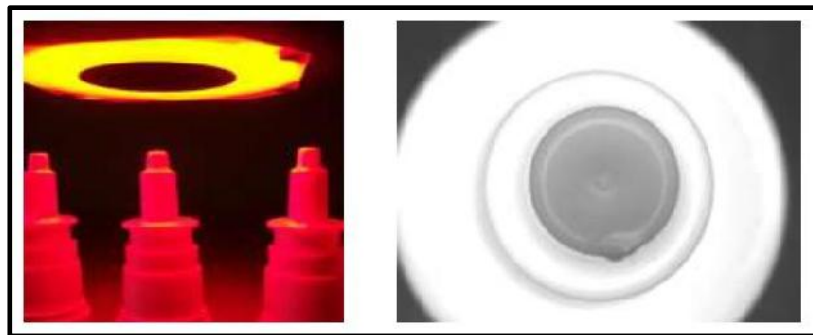


Figura 2.7 Patrón anular en una inspección de válvulas para farmacia

Los patrones anulares y hemisféricos son muy útiles cuando se trabaja con superficies brillantes. Los anulares permiten proyectar sobre superficies planas una luz intensa y homogénea si se observa desde el eje del anillo. Si se trata de una superficie brillante que no es plana o de superficies muy reflexivas para obtener una imagen sin sombras es preciso recurrir a un patrón hemisférico. El resultado es similar a si el objeto fuera observado bajo un cielo completamente cubierto en el que la posición del sol es imposible de adivinar. Con este tipo de iluminación la luz llega desde todas las direcciones reduciendo al mínimo la aparición de sombras y brillos sobre los objetos.

- **Estabilidad:**

El primer aspecto que hay que considerar en lo que respecta a la estabilidad en la iluminación es que la escena no se debe ver

afectada por variaciones de la luz ambiente. Siempre se debe apantallar el entorno en el que se va a trabajar el sistema de visión, instalando un carenado que evite las perturbaciones de la luz exterior. Existen aplicaciones de guiado de robots con visión, donde a veces es difícil aislar totalmente al sistema de la iluminación ambiente. En estos casos habrá que emplear una fuente de luz potente con objeto de que las perturbaciones procedentes de las variaciones ambientales queden enmascaradas o bien iluminar fuera del espectro visible por ejemplo en el infrarrojo cercano.

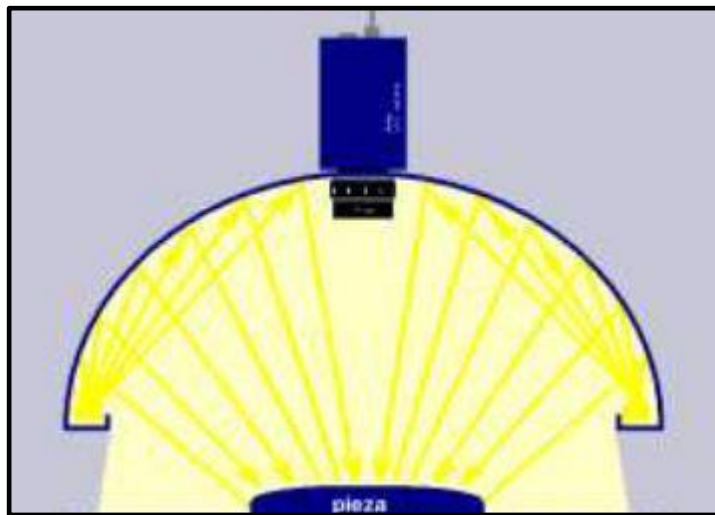


Figura 2.8 Patrón de luz hemisférico

Hay que tener especial cuidado cuando se emplean velocidades de obturación en las cámaras por debajo de los 20 milisegundos pues aparece una fluctuación en la intensidad de la imagen si la fuente de luz es alimentada directamente con la intensidad de la red eléctrica. Tal es el caso de las lámparas incandescentes de tungsteno tradicionales y los fluorescentes estándar. Esto origina que la intensidad media de la imagen fluctúe, haciendo prácticamente imposible un procesamiento fiable de la imagen. En estos casos es preciso alimentar las lámparas incandescentes

con corriente continua y si se utilizan fluorescentes incorporar un balasto de alta frecuencia.

Por otro lado hay que considerar que las fuentes de iluminación se van degradando con el tiempo disminuyendo su eficiencia lumínica. Esta pérdida de luz alcanza el 30% en fuentes incandescentes tras unos pocos miles de horas de funcionamiento. Incluso los LED que es la fuente con menor degradación en el tiempo, pueden fluctuar en su intensidad de luz debido a las variaciones en voltaje, corriente y temperatura. Por tanto, es fundamental al desarrollar los algoritmos de tratamiento de imagen no hacer nunca la suposición de que la luz va a ser perfectamente constante. En caso de que se requiriera una iluminación especialmente estable existen sistemas que implantan una realimentación de la luz proporcionada para mantener la intensidad lumínica de salida inamovible.

- Robustez:

Un sistema de iluminación tiene que tener una esperanza de vida aceptable en un entorno industrial. Debe ser capaz de resistir las vibraciones y golpes siempre presentes en estos entornos. Por ejemplo las tradicionales lámparas incandescentes se muestran como la opción menos robusta por su corta esperanza de vida así como por su fragilidad de los filamentos ante vibraciones o choque. Por este motivo las lámparas incandescentes han sido desplazadas por otras fuentes de luz mucho más robustas y en la actualidad sólo son utilizadas donde se requiere una gran potencia lumínica sobre grandes áreas.

En esta línea es una buena práctica en aquellos sistemas que empleen fuentes con una esperanza de vida más corta y que se degradan con el tiempo, como las lámparas incandescentes y los fluorescentes, implantar un programa de sustitución regular y seguir la política, dado que su coste no es excesivo, de reemplazarlas cuando se encuentran a media vida.

2.1.3.3 Configuración para la Iluminación

Se ha indicado que cuando se desea captar con una cámara un objeto o una determinada característica, la luz proporcionada por la fuente de iluminación y que es reflejada por el objeto debe entrar por el objetivo de la cámara. Por tanto, la apariencia del objeto en la imagen no dependerá sólo de la fuente de luz empleada sino también va a depender radicalmente de la posición de la fuente con respecto al objeto y a la cámara.

Existe un conjunto de técnicas de iluminación basadas en la utilización de distintas configuraciones geométricas de la fuente de luz con respecto al objeto y a la cámara que son empleadas frecuentemente en los sistemas de visión artificial.

- Retroiluminación Difusa:

Si para el reconocimiento o medida de un producto sólo se precisa su silueta, se puede obtener de forma muy sencilla y estable muy buenas imágenes iluminando con una fuente difusa a contraluz. Esta configuración, conocida como retroiluminación difusa produce imágenes de gran contraste entre el objeto y el fondo que son fácilmente procesables. Esta iluminación también resulta muy adecuada cuando se pretende medir el grado de porosidad de ciertas sustancias, como tejidos, láminas de corcho, o también cuando se desea determinar el nivel de líquido o presencia de algún componente interno en envases transparentes.

Debe tenerse cuidado, especialmente en las aplicaciones de metrología de no sobresaturar el sensor de la cámara con la fuente de iluminación pues en este caso el objeto parecerá más pequeño de lo que realmente es. Habrá que reducir la apertura o la velocidad del obturador hasta que desaparezca la sobresaturación. Si se trata de una aplicación de metrología fina para la retroiluminación será preciso recurrir a una fuente de iluminación

colimada que garantice la emisión de rayos de luz paralelos al eje óptico y que por tanto no va a falsear el perfil de la pieza observada.

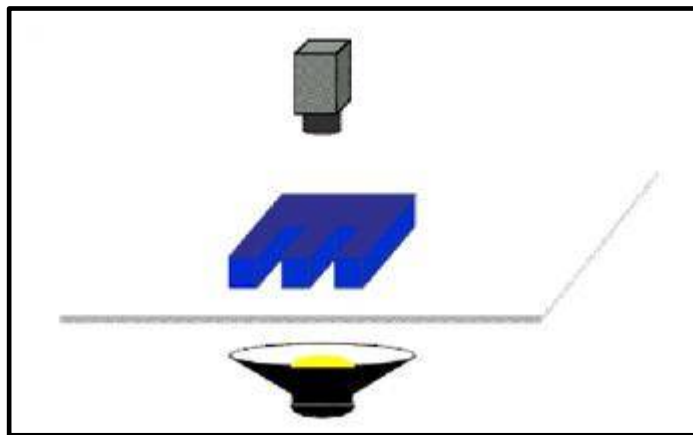


Figura 2.9 Retroiluminación difusa

- Iluminación Frontal:

Cuando se requieren más detalles del producto que la simple silueta lo más sencillo es proyectar una o más luces sobre el objeto desde el mismo lado que la cámara. Esta técnica se conoce como iluminación frontal y en su configuración más típica las luces forman 45 grados con el eje óptico de la cámara que se encuentra sobre el objeto.

Si las superficies son brillantes esta configuración no es la más conveniente pues hace aparecer brillos y zonas oscuras que dificultan el procesamiento de imagen.

En este caso para la iluminación frontal es preciso emplear fuentes de iluminación difusa con patrones anulares o hemisféricos para

que los haces luminosos incidan sobre el objeto desde todas las direcciones. Los anillos son a menudo la mejor opción cuando se trabaja con objetos circulares aunque también resultan muy útiles para observar objetos pequeños o pequeñas áreas de grandes objetos.

Cuando se opta por la iluminación frontal es conveniente experimentar con el número de lámparas, su distancia al objeto y el ángulo respecto a la cámara.

Muchas veces también resulta útil comparar la iluminación directa con la difusa quitando los difusores y viendo el efecto en la imagen en vivo.

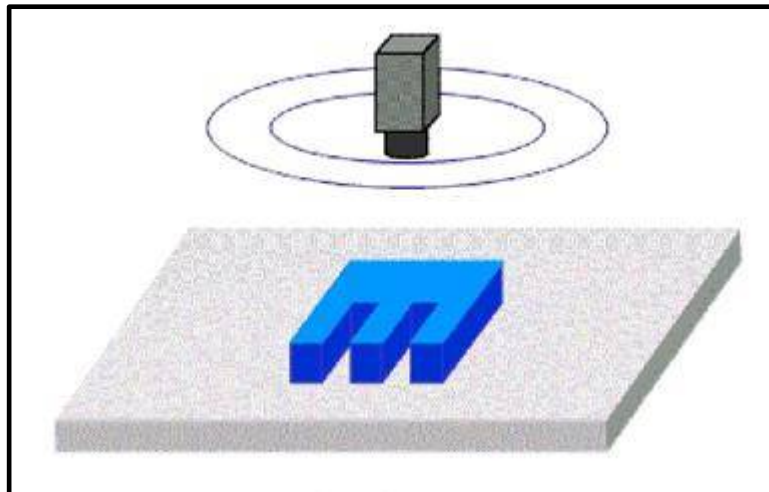


Figura 2.10 Iluminación Frontal

- Iluminación Direccional:

Consiste en una iluminación direccionada en algún sentido en el espacio para destacar una característica concreta del objeto. La principal virtud es la creación de sombras sobre el objeto, lo que puede ayudar a aumentar el contraste de partes tridimensionales y obtener la consiguiente información 3D.

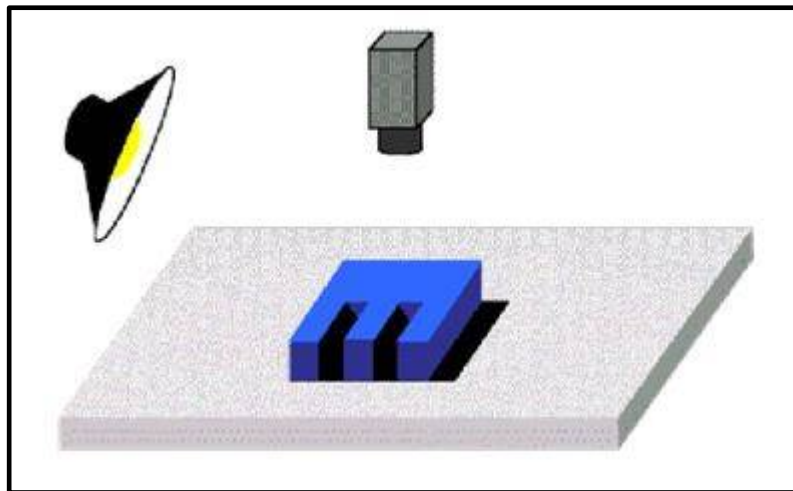


Figura 2.11 Iluminación Direccional

- Iluminación Estructurada:

Consiste en proyectar sobre la pieza unos patrones de luz conocidos modulados (proyección de puntos, franjas o rejillas sobre la superficie de trabajo) y observando la luz reflejada, que también viene modulada, obtener información sobre la estructura de la superficie del objeto, la cual puede ser reconstruida mediante triangulación. Las fuentes de luz empleadas deben de ser especiales pues deben ser capaces de emitir luz estructurada y suelen ser láseres. Se usa para reconstrucciones 3D de objetos y para conocer su forma.

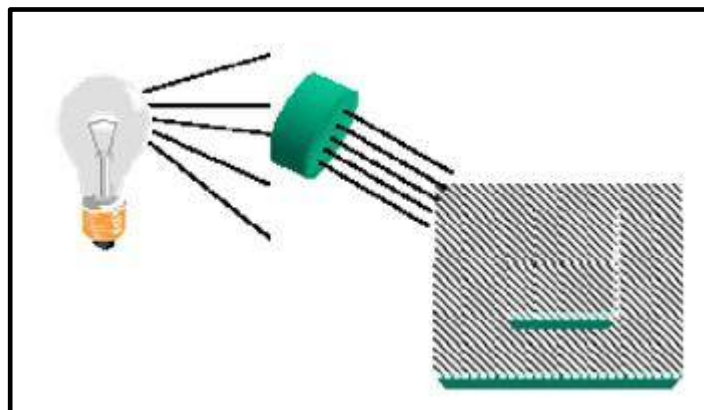


Figura 2.12 Iluminación Estructurada

2.1.4 Fuentes de Iluminación

Por otro lado debemos prestar atención también a las diferentes fuentes de iluminación.

- **Lámparas incandescentes:**

Es la fuente de iluminación más común y consiste en un filamento de tungsteno o halógeno-tungsteno. Como ventaja tiene que existe gran variedad de potencias y como desventaja, que reduce su luminosidad con el tiempo, lo que puede provocar problemas en algunos sistemas de visión.

- **Tubos fluorescentes:**

Más eficaces que las lámparas y suministran una luz más difusa, que es bueno para piezas muy reflectoras. Existe una gran variedad, tanto en forma (circulares, lineales), como en tamaño con lo que son ampliamente utilizados.

- **Fibra óptica:**

Para iluminar zonas de difícil acceso o extremadamente pequeñas. Proporciona iluminación constante.

- **Láser.**

Empleados para una iluminación con luz estructurada, ya que el láser es capaz de emitir luz estructurada con un control adecuado. Tiene el inconveniente de presentar un mal comportamiento frente a superficies que absorben luz.

2.2 Preprocesamiento de la Imagen

Una etapa importante de la VA es el preprocesamiento de imágenes, es decir, la transformación de la imagen original en otra imagen en la cual hayan sido eliminados los problemas de ruido granular de cuantización o de iluminación espacialmente variable. La utilización de estas técnicas permite el mejoramiento de las imágenes digitales adquiridas de acuerdo a los objetivos

planteados en el sistema de visión artificial.

A continuación sólo se mencionara las técnicas de preprocesamiento empleado en el presente trabajo.

2.2.1 Conversión a Escala de Grises

En esta parte se trata la conversión de una imagen en color a escala de grises, el equivalente a la luminancia de la imagen. Como sabemos el ojo percibe distintas intensidades de luz en función del color que se observe, esto es debido a la respuesta del ojo al espectro visible la cual se puede observar en la figura 2.13, por esa razón el cálculo de la escala de grises o luminancia de la imagen debe realizarse como una media ponderada de las distintas componentes de color de cada pixel.

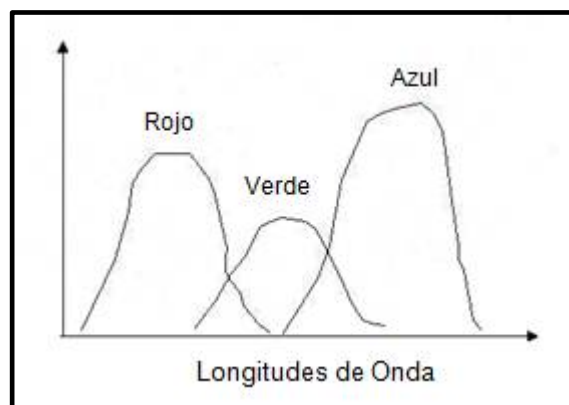


Figura 2.13 Componentes RGB de una imagen

La ecuación de la luminancia es la expresión matemática de ese fenómeno, y los factores de ponderación de cada componente de color nos indican la sensibilidad del ojo humano a las frecuencias del espectro cercanas al rojo, verde y azul.

$Y = 0.3 * R + 0.5 * G + 0.11 * B$	Ecuación 2.7
------------------------------------	--------------

Por tanto, para realizar esta conversión basta con aplicar la ecuación 2.7 a cada pixel de la imagen de color, entonces resultará una nueva matriz de un byte por pixel que daría la información de luminancia.

2.2.2 Filtrado Espacial

El empleo de máscaras espaciales para el procesamiento de las imágenes, se denomina frecuentemente filtrado espacial, y las propias máscaras se denominan filtros espaciales. Dentro del filtrado espacial, existen los filtros suavizantes, que se emplean para hacer que la imagen aparezca algo borrosa y también para reducir el ruido.

- **Filtro Pasa Bajo:**

Para un filtro espacial de 3x3 (grado 3), la construcción más simple consiste en una máscara en la que todos los coeficientes sean iguales a 1. Sin embargo, la respuesta, en este caso es, la suma de los niveles de gris de los nueve píxeles, lo que hace que el resultado quede fuera del rango válido de gris [0,255]. La solución consiste en cambiar la escala de la suma, dividiéndola por el grado de la máscara al cuadrado, en este caso por 9. La figura 2.14 muestra la máscara resultante.

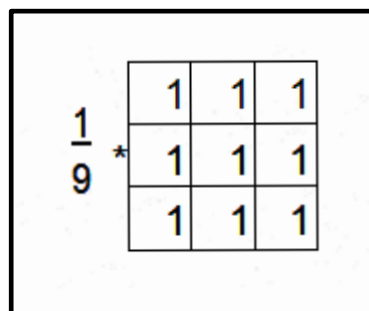


Figura 2.13 Componentes RGB de una imagen

- **Filtro de Mediana**

Una de las principales dificultades del método de suavizamiento, es que afecta los bordes y otros detalles de realce. Cuando el objetivo es reducir el ruido, el empleo de los filtros de mediana representa una posibilidad alternativa. En este caso, el nivel de gris de cada pixel se reemplaza por la mediana de los niveles de gris en un entorno de este pixel, en lugar del promedio, como lo hace el filtro pasa bajo.

Este método es particularmente efectivo cuando el patrón de ruido consiste en componentes fuertes y de forma puntiaguda, y la característica que se desea preservar es la agudeza de los bordes.

La mediana m de un conjunto de valores es tal que la mitad de los valores del conjunto quedan por debajo de m y la otra mitad por encima. Con el fin de realizar el filtro de mediana, en el entorno de un pixel, primero se deben extraer los valores del pixel y de su entorno, determinar la mediana y asignar este valor al pixel. Por ejemplo, para un entorno de 3×3 , con los valores que se observan en la figura 2.14, se realizan los siguientes pasos:

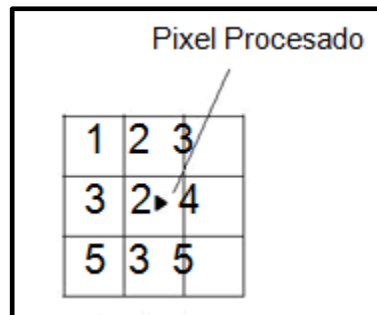


Figura 2.14 Entorno 3x3

Se almacenan los valores en un vector:

$X[1] = 1, X[2] = 2, X[3] = 3, X[4] = 3, X[5] = 2, X[6] = 4, X[7] = 5, X[8] = 3, X[9] = 5$
--

Se hace un ordenamiento en el vector, por valor de nivel de gris:

$X[1] = 1, X[2] = 2, X[5] = 2, X[3] = 3, X[4] = 3, X[8] = 3, X[6] = 4, X[7] = 5, X[9] = 5$
--

Entonces, el valor de la mediana corresponde a la posición 4, con el

valor 3.

Suponiendo que una imagen está afectada por un cierto error por pixel (debido a ruido en la cámara, ruido shot, etc.), se puede aplicar un operador de proximidad por media de 5x5 para conseguir una mejora notable de la imagen de entrada. A diferencia de un operador de aproximación normal, el filtro de media no causa aplanamiento en los bordes, sino que se limita a eliminar píxeles únicos aleatorios.

2.3 Segmentación de la Imagen

La segmentación es el proceso mediante el cual una imagen se descompone en regiones o elementos que pueden corresponder a objetos o parte de objetos. El proceso de segmentación se encarga de evaluar si cada pixel de la imagen pertenece o no al objeto de interés. Esta técnica de procesamiento de imágenes idealmente genera una imagen binaria, donde los píxeles que pertenecen al objeto se representa con un 1, mientras que los que no pertenecen al mismo se representan con un 0. Este tipo de particionamiento está basado en el análisis de alguna característica de la imagen, tal como los niveles de gris o la textura. A continuación describiremos el método de segmentación basado en la umbralización.

2.3.1 Segmentación Basado en Píxeles:

Este método de segmentación toma en cuenta sólo el valor de gris de un pixel, para decidir si el mismo pertenece o no al objeto de interés. Para ello, se debe encontrar el rango de valores de gris que caracterizan dicho objeto, lo que requiere entonces la búsqueda y el análisis del histograma de la imagen.

El objetivo de este método, es el de encontrar de una manera óptima los valores característicos de la imagen que establecen la separación del objeto de interés, con respecto a las regiones que no pertenecen al mismo; debido a esta característica y si los valores de gris del objeto y del resto de la imagen difieren claramente, entonces el histograma mostrará una distribución bimodal, con dos máximos distintos, lo que

debiera generar, la existencia de una zona del histograma ubicada entre los dos máximos, que no presenten los valores característicos, y que idealmente fuera igual a cero, con lo cual se logrará una separación perfecta entre el objeto y la región de la imagen que lo circunda, al establecer un valor umbral ubicado en esta región del histograma. Por lo tanto cada pixel de la imagen, es asignado a una de dos categorías, dependiendo si el valor umbral es excedido o no.

Si el valor del histograma ubicado entre los dos máximos, es distinto de cero, las funciones de probabilidad de los valores de gris del objeto y de la región restante, se solaparán, de tal manera que algunos pixeles del objeto deberán ser tomados como pertenecientes a la región circundante y viceversa. Conocida la distribución de la función de probabilidad de los pixeles del objeto y de la región circundante, es posible aplicar análisis estadístico en el proceso de buscar un umbral óptimo, con el número mínimo de correspondencias erróneas. Estas distribuciones pueden ser estimadas por histogramas locales, los cuales solamente incluyen las regiones correspondientes de la imagen.

En imágenes industriales tenemos generalmente un alto contraste lo cual permite aplicar segmentaciones muy simples como el mencionado, para distinguir dos clases de regiones en la imagen: región objeto y región fondo.

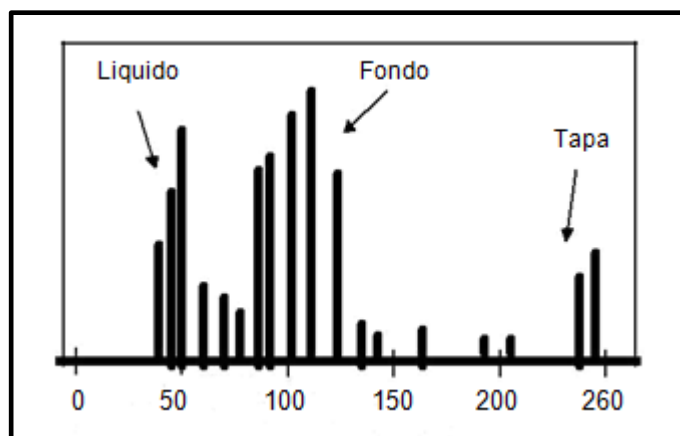


Figura 2.15 Histograma de la imagen de una botella

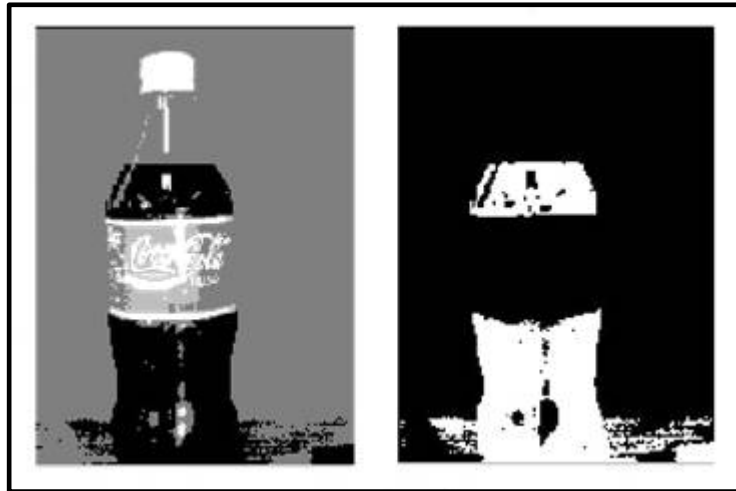


Figura 2.16 Segmentación de la imagen de una botella

En la imagen de la figura 2.15, el histograma nos indica que el fondo (gris claro) es lo que ocupa más espacio en la imagen, porque es el que contiene mayor número de píxeles en la escala de grises 125 (aproximadamente). Luego tenemos el líquido (refresco, color negro) y la tapa y partes de la etiqueta (color blanco). Si aplicamos un umbral a esta imagen, a partir del límite que establece el líquido (aproximadamente nivel 50 de la escala de grises), todo lo demás se convierte en color negro (0), y el líquido se convierte en color blanco (1). De esta forma, la imagen pasa a ser binaria, y el líquido queda claramente separado de todo lo demás que aparece en la imagen.

2.3.2 Operaciones Morfológicas:

Las operaciones morfológicas son métodos para procesar imágenes binarias basado sobre formas. Estas operaciones toman una imagen binaria como entrada y dan como resultado una imagen binaria como salida. El valor de cada píxel en la imagen de salida está basado sobre el correspondiente píxel de entrada y sus vecinos. Dentro de las operaciones morfológicas tenemos la dilatación y erosión, las cuales serán tratadas a continuación.

La dilatación adiciona píxeles a los límites del objeto (es decir los cambia

de off a on), y la erosión remueve pixeles sobre los límites del objeto (los cambia de on a off).

- **Dilatación:**

Sea A y B conjuntos de Z^2 y \emptyset representando al conjunto vacío, la dilatación de A por B, se representa $A \oplus B$, en la ecuación (2.8)

$A \oplus B = \{x / (B)_x \cap A \neq \emptyset\}$	Ecuación 2.8
--	--------------

Por tanto, el proceso de dilatación consiste en obtener la reflexión de B sobre su origen, después, cambiar esta reflexión por x. La dilatación de A por B es entonces, el conjunto de todos los desplazamientos x, tales que B y A se solapen en al menos un elemento distinto de cero. Basándose en esta interpretación, la ecuación (2.8) se puede volver a representar mediante la ecuación (2.9):

$A \oplus B = \{x / [(B)_x \cap A] \subseteq A\}$	Ecuación 2.9
---	--------------

Al conjunto B, se le conoce normalmente como el elemento de estructura de la dilatación. Como vemos, se toma el elemento de estructura B como una máscara de convolución. Aunque la dilatación se basa en operaciones de conjunto, mientras que la convolución se basa en operaciones aritméticas, el proceso básico de mover a B respecto a su origen, y desplazarlo después sucesivamente de tal forma que se deslice sobre el conjunto (imagen) A, es análogo al proceso de convolución expuesto anteriormente.

Los componentes del conjunto B (elemento estructurador), pueden ser ceros o unos. La cantidad de ceros y/o unos determina, en conjunto con el tamaño del elemento estructurador, el efecto que produce su utilización en las operaciones morfológicas.

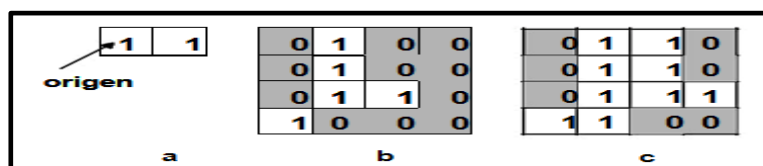


Figura 2.17 Ejemplo de dilatación. (a) elemento estructural. (b) imagen original. (c) resultado de la dilatación.

Al usar el elemento estructurador que se muestra en la figura 2.17a, y si se tiene como imagen de entrada la figura 2.17b y después de aplicar el proceso de dilatación, se obtiene la imagen de salida ilustrada en la figura 2.17c.

El estado de cualquier pixel dado en la imagen de salida es determinado aplicando una regla a los vecinos del correspondiente pixel en la imagen de entrada. La siguiente regla define la operación para la dilatación:

Para dilatación, si cualquier pixel en los vecinos del pixel de entrada es ON, el pixel de salida es ON. De otra manera el pixel de salida es OFF.

- **Erosión:**

Para los conjuntos A y B de Z^2 , la erosión de A por B representada por $A \ominus B$, se define como la ecuación (2.9).

$A \ominus B = \{x / (B)_x \subseteq A\}$	Ecuación 2.10
---	----------------------

Donde se dice, que la erosión de A por B es el conjunto de todos los puntos x tal que B, trasladado por x, está contenido en A. Como en el caso de la dilatación, la ecuación (2.10) no es la única definición de la erosión.

Sin embargo, esta ecuación, normalmente, es más adecuada en implementaciones prácticas de la morfología. Si se toma como ejemplo el elemento estructurador que se muestra en la figura 2.18a, y la imagen de entrada de la figura 2.18b, se obtiene la imagen de salida que se muestra en la figura 2.18c, en la cual el proceso de erosión ha sido aplicado.

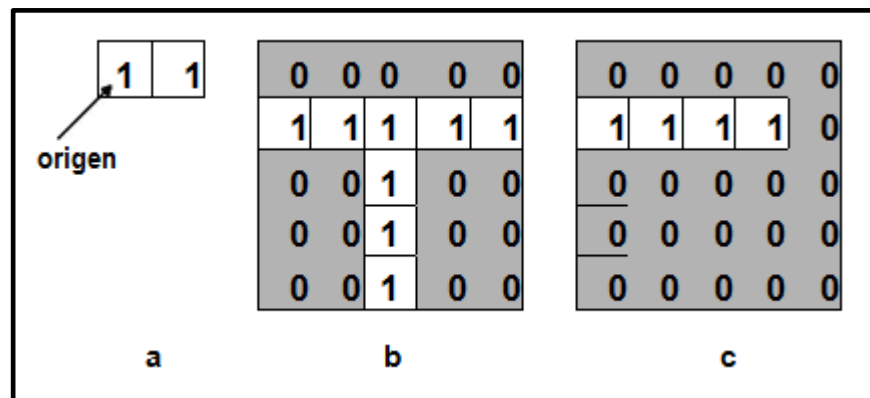


Figura 2.18 Ejemplo de erosión. (a) elemento estructural. (b) imagen original. (c) imagen erosionada.

El estado de cualquier pixel dado en la imagen de salida es determinado aplicando una regla a los vecinos del correspondiente pixel en la imagen de entrada. La siguiente regla define la operación para la erosión:

Para erosión, si cada pixel en el vecino del pixel de entrada esta ON, el pixel de salida estará ON. De otra manera el pixel de salida está OFF.

2.3.3 Etiquetado:

En la práctica llegar a la imagen binaria no suele ser suficiente para realizar una descripción adecuada, por lo que el proceso de segmentación se prolonga aplicando diversas técnicas sobre este tipo de imágenes.

Existe una gran cantidad de técnicas de análisis para imágenes binarias, con propósitos tan variados, entre estas técnicas está el contar, etiquetar objetos y filtrado de objetos según su tamaño.

Una de las operaciones más comunes en visión es encontrar las componentes conectadas dentro de una imagen. En la figura 2.19 vemos un ejemplo de una imagen y su imagen de componentes conectadas, en donde se ha etiquetado con un número a las componentes conectadas

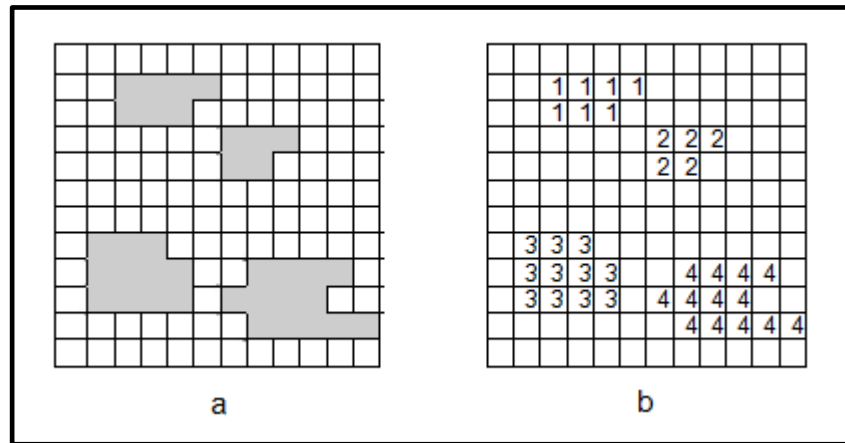


Figura 2.19 (a) Imagen original. (b) Etiquetado.

Se ha de disponer de una definición consistente de conectividad para demarcar todos los objetos en imágenes binarias y ser capaces de idear algoritmos para etiquetarlos y contarlos.

El etiquetado es una técnica que, partiendo de una imagen binaria, nos permite etiquetar cada uno de los objetos conectados presentes en la imagen. Esto va a posibilitar:

- Distinguir los objetos respecto del fondo (propiedad intrínseca de la imagen binaria).
- Distinguir un objeto respecto de los demás objetos.
- Conocer el número de objetos en la imagen.

En general, los algoritmos para etiquetar dan buenos resultados con objetos convexos, pero presentan problemas cuando aparecen objetos que tienen concavidades (formas en U), como se observa en la figura 2.20 donde diferentes partes de un mismo objeto pueden acabar con etiquetas distintas, incluso pueden aparecer colisiones de etiquetas.

En este sentido el peor caso que puede plantearse es un objeto con forma de espiral.

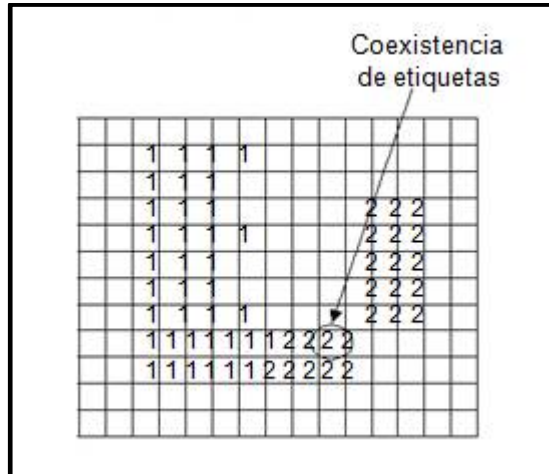


Figura 2.20 Imagen en forma de U, etiquetada erróneamente.

2.3.4 Filtro de Tamaño:

El filtrado de tamaño en imágenes binarias está orientado, principalmente con dos objetivos:

- Eliminación de ruido: debido al proceso de umbralización pueden aparecer regiones en la imagen binaria que son consecuencia del ruido. Tales regiones son normalmente pequeñas.
- Reducir el número de objetos de la imagen (hay objetos de un determinado tamaño que no interesan para la aplicación): En algunas ocasiones resulta interesante eliminar de una imagen binaria aquellos objetos que no superen un tamaño determinado, o por el contrario, eliminar los objetos que si superen este tamaño.

Los objetos son filtrados teniendo en cuenta su tamaño o área o utilizando las operaciones de erosión y dilatación como se analizó anteriormente.

Sin embargo, en muchas aplicaciones es conocido que los objetos de interés son de un tamaño mayor que T píxeles (área). Todas las componentes de la imagen que tienen un tamaño igual o menor que T

pixeles se eliminan; para ello se cambian los correspondientes pixeles al valor del fondo (0).

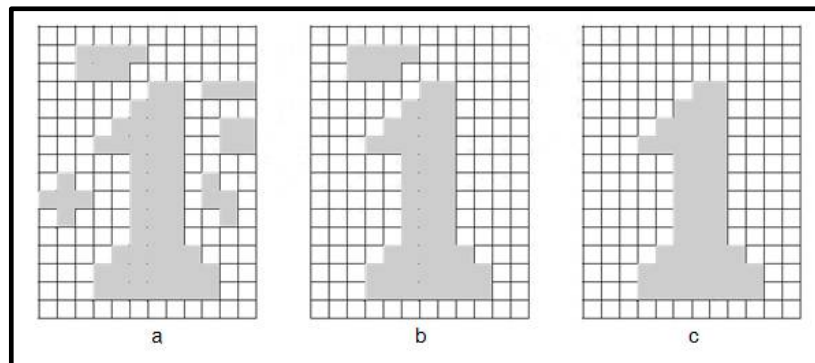


Figura 2.21 (a) Imagen original. (b) Filtrado de área $T=6$, (c) Filtrado de área $T=10$

2.4 Descripción

Con el objeto de poder analizar una imagen será necesario tener una descripción en la memoria de la computadora. Esta descripción no es otra cosa que un modelo que representa las características de la imagen de relevancia para los fines del sistema de visión artificial específicos. Los modelos que describen una imagen pueden ser estadísticas, estructurales, etc. pero todos ellos se obtienen a partir de imágenes previamente segmentadas y analizadas. En el tipo de imágenes utilizados en robótica, la información relativa se encuentra en el contorno, entonces los modelos utilizados para describir este tipo de imágenes representaran tan solo contornos presentes en la imagen, y entre los más usuales tenemos a las cadenas de códigos.

2.5 Reconocimiento

Finalmente, una vez que se han realizado las etapas de preprocesamiento, segmentación y extracción de características, se procede a realizar el reconocimiento.

La fase de reconocimiento consistirá en la clasificación de los objetos de interés, presentes en la imagen de acuerdo a sus modelos respectivos. Este punto se tratará más detalladamente en los siguientes capítulos, debido a su importancia en el desarrollo en este trabajo de investigación.

CAPITULO 3

3. DESCRIPCIÓN Y EXTRACCIÓN DE CARACTERÍSTICAS

Para la descripción de los objetos, deben de ser independientes del tamaño, localización u orientación del objeto, y además deben ser suficientes para discriminar objetos entre sí.

Los descriptores se basan en la evaluación de alguna característica del objeto, por ejemplo:

- **Descriptores unidimensionales:**
Códigos de cadena, perímetro, forma del perímetro.
- **Descriptores bidimensionales:**
Área, momentos de inercia, etc.
- **Descriptores específicos:**
Número de agujeros, área de agujeros, posición relativa de agujeros, rasgos diferenciadores de un objeto, etc.

Para encontrar la manera óptima de representar la información original que describe a cada uno de los patrones basado en los descriptores mencionados inicialmente, se realiza el proceso conocido como extracción de características; dicho proceso consiste en reducir la cantidad de información, que representa a cada uno de los patrones, obteniendo de esta forma, un vector de características que represente de la mejor manera posible al patrón original.

3.1 Propiedades en la Extracción de Características

- a. **Discriminación:** valores numéricos diferentes para objetos de clases diferentes.
- b. **Fiabilidad:** cambios numéricos pequeños para objetos de una misma clase, es decir los objetos de una misma clase deberán representar la menor dispersión.
- c. **Incorrelación:** la independencia de las características equivale al principio de la parsimonia, decir lo mismo con la máxima economía de términos. Nunca debe utilizarse características que dependan fuertemente entre sí, ya que no añaden información.

- d. Cálculo en Tiempo Real:** este es un requisito que puede llegar a ser determinante en ciertas aplicaciones de tiempo real, ya que las características deben calcularse en un tiempo aceptable.
- e. Invarianza:** frente a transformaciones geométricas como rotación, traslación, escalado.
- f. Dimensionalidad:** el tamaño del vector de características debe de ser menor que la del patrón original. Las características deben representar una codificación óptima de la entrada, perdiendo la información que no sea muy importante, se debe reflejar lo esencial del objeto no lo accesorio.

Especialmente, cuando se realiza el reconocimiento de patrones utilizando redes neuronales, la extracción de características debe tratar de obtener un vector de características con una dimensionalidad mucho menor a la del patrón original, puesto que un vector con una dimensionalidad más pequeña que sea dado como entrada a la red neuronal tiene varios beneficios. En primer lugar, la cantidad de pesos que deben de ser aprendidos es menor, y en segundo lugar, al tener menos pesos, el tiempo de entrenamiento puede ser reducido considerablemente.

3.2 Extracción de Características

Se refiere al proceso de extraer algunas medidas numéricas e información relevante de los datos en bruto de los patrones suministrada por los sensores (representación inicial), para la clasificación. Por otro lado, se define también como el proceso de formar un conjunto de características (dimensión “n”), partiendo de los datos de entrada (“m>n”).

Dado un conjunto de patrones n-dimensionales:

$X = [x_1 \ x_2 \ x_3 \ \dots \ x_n]^T$	Ecuación 3.1
---	---------------------

De ello se trata de conseguir un nuevo conjunto de características con d-dimensionales:

$Y = [y_1 \ y_2 \ y_3 \ \dots \ y_d]^T ; \ d \leq n$	Ecuación 3.2
--	---------------------

Para lograr las relaciones anteriores, se pueden realizar de dos formas:

3.2.1 Reduciendo la dimensionalidad de los datos:

Si los patrones son de alta dimensionalidad, el coste computacional asociado a la clasificación puede ser muy alto. Como veremos, muchos clasificadores están basados en cálculos de distancias y estos cálculos pueden depender de forma cuadrática respecto a la dimensionalidad de los patrones. Como otra consideración computacional hay que considerar el espacio de almacenamiento adicional que supone guardar los valores de nuevas variables. Además, algunas de las variables pueden ser redundantes con otras y no aportar información adicional.

La técnica dedicadas a seleccionar las variables más relevantes se denomina selección de características y reducen la dimensionalidad de los patrones. Este proceso puede esquematizarse como se aprecia en la figura 3.1, en el que un módulo selector recibe patrones n-dimensionales (para el ejemplo “n = 100”) y proporciona como resultado las “d” variables más significativas (para el ejemplo “d=3”) de acuerdo a algún criterio a optimizar. En este caso “d<n” y el conjunto de las “d” variables seleccionadas es un subconjunto del conjunto original de variables.

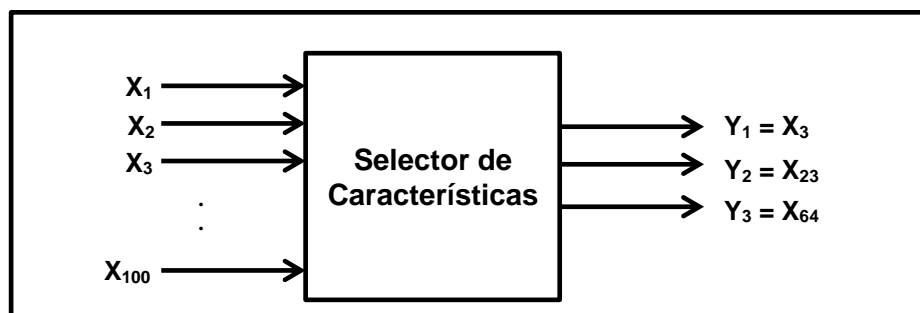


Figura 3.1 Selección de las 3 variables más significativas de un conjunto de 100.

3.2.2 Cambiando el espacio de representación:

El objetivo es obtener una nueva representación de los patrones en la que los agrupamientos aparezcan bien separados si son de diferente clase y que haya un agrupamiento por clase. Esto puede conseguirse aplicando alguna transformación sobre los datos originales.

Estas técnicas reciben el nombre de extracción de características y producen un nuevo conjunto de variables. Este proceso puede esquematizarse como se aprecia en la figura 3.2, en el que un módulo extractor recibe patrones n-dimensionales (Para el ejemplo “n=100”) y proporciona como resultado nuevos patrones n-dimensionales de acuerdo a algún criterio a optimizar.

Es posible que las nuevas variables estén implícitamente ordenadas, por lo que proporcionan, adicionalmente un procedimiento de selección. En el caso “d=n” y las variables seleccionadas no forman un subconjunto del conjunto original de variables.

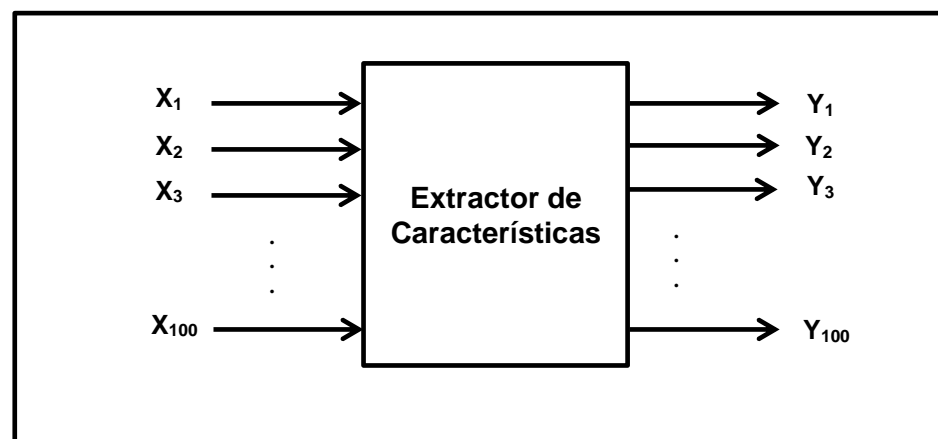


Figura 3.2 Transformación de las 100 variables.

Los vectores resultantes del proceso de selección o extracción se denominan características, aunque nosotros utilizaremos el término patrón a no ser que queramos resaltar explícitamente que se ha producido un proceso de selección o extracción de características.

3.3 Métodos de Descripción

3.3.1 Descriptores de Forma:

Parten de una información binaria de pertenencia de un pixel al objeto.

En este tipo tenemos los descriptores siguientes:

- **Descriptores de Contorno:**

Información binaria de pertenencia al contorno: código de cadenas, signaturas, aproximaciones poligonales, representación polar, esqueletización, descriptores de Fourier.

- **Descriptores de Región:**

Información binaria de pertenencia al interior del objeto, no sólo con los del contorno: momentos (área, centro de gravedad, momentos invariantes), esqueletos, descriptores topológicos.

Una imagen binaria generalmente es representada como una matriz, la cual contiene información tanto del objeto como del fondo. Una representación más compacta de la imagen, se puede obtener al almacenar en una estructura de datos, la información de los pixeles que acotan al objeto. De esta forma, bastaría con realizar análisis u operaciones sobre la estructura de datos, para realizar la extracción de los parámetros que definen el objeto, o para aplicar operaciones de modificación de la forma del mismo. Las técnicas que se basan en la utilización de estructuras de datos, para almacenar los puntos que acotan al objeto, son las técnicas de codificación de contornos.

3.3.1.1 Código de Cadena

Una de las técnicas utilizadas en la extracción de contornos de imágenes a las cuales se les ha aplicado un procedimiento de segmentación, es la del seguimiento de contornos para su codificación. Un algoritmo de seguimiento de contornos examina todas las direcciones establecidas por el código de cadena, el cual se aprecia en la figura 3.3.

La codificación del contorno en código de cadena, resultará en una estructura de datos que representa la frontera de una imagen binaria sobre un mallado discreto, en cuyos campos se indica la dirección de los píxeles de la frontera. El primer paso es el de escoger sobre el contorno de la imagen, el pixel inicial cuyas coordenadas deben ser almacenadas en la estructura; para luego seguir el contorno en la dirección de las agujas del reloj. Si el contorno es seguido con ayuda de una vecindad de 4, habrán cuatro posibles direcciones que seguir, mientras que si es seguido con una vecindad de 8 existirán ocho direcciones, como muestra la figura 3.4. Al utilizar este código, el contorno puede entonces ser codificado con 2 o 3 bits. Los códigos de cadena representan de una manera compacta las imágenes binarias.

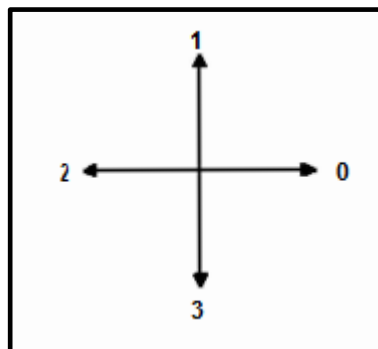


Figura 3.3 Código de cadena – Vecindad de 4.

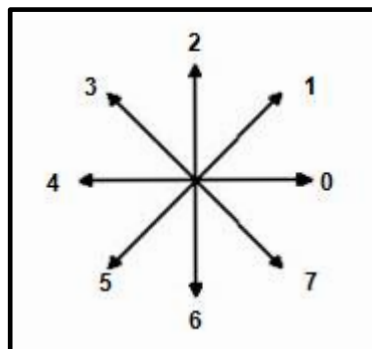


Figura 3.4 Código de cadena – Vecindad de 8.

Una vez que se ha revisado como realizar la representación de objetos (en la figura 3.5 y figura 3.6), la idea fundamental, es la estudiar ciertos parámetros que puedan dar una representación adecuada de la forma del objeto.

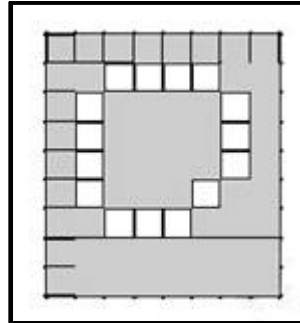


Figura 3.5 Codificación de una frontera.

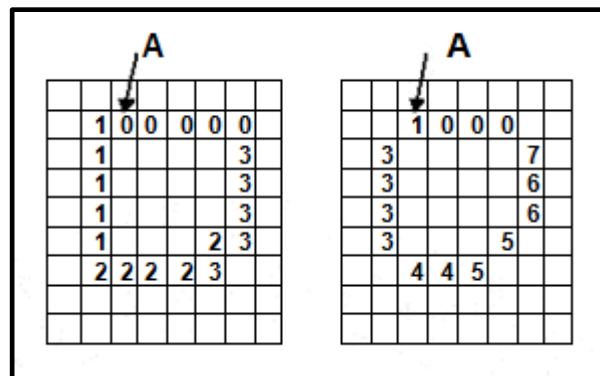


Figura 3.6 Conectividad 4 y Conectividad 8

3.3.1.2 Área

En una imagen digital, el área de un objeto está dada por el número de píxeles que representan al mismo, por lo tanto el cálculo del área se realiza contando el número de píxeles.

Si una imagen está representada por un contorno codificado en código de cadena, el cálculo del área se realiza por un algoritmo que trabaja en forma similar a la integración numérica.

3.3.1.3 Perímetro

Es un parámetro geométrico que al igual que el área, puede ser calculado a partir del código de cadena. Para realizar este cálculo, es necesario contar la longitud del código, y tomar en consideración que los pasos en direcciones diagonales deben ser multiplicados por un factor igual a raíz cuadrada de dos. El perímetro p de un contorno codificado en código de cadena se expresa según la siguiente ecuación:

$p = n_e + n_0\sqrt{2}$	Ecuación 3.3
-------------------------	--------------

Donde:

- n_e : número de pasos pares del código.
- n_0 : número de pasos impares del código.

3.3.1.4 Circularidad

Es uno de los parámetros geométricos utilizados en la comparación de objetos que son observados desde diferentes distancias, ya que el mismo no depende del tamaño del objeto. La circularidad es un número adimensional definido según la siguiente ecuación:

$c = \frac{p^2}{A}$	Ecuación 3.4
---------------------	--------------

Donde:

- p : perímetro.
- A : área.

La invariante área y perímetro simplemente determina el área y el perímetro de la imagen a través de la siguiente ecuación:

$Invariante = \frac{A}{p^2}$	Ecuación 3.5
------------------------------	--------------

3.3.1.5 Momentos Generales

La teoría de los momentos proporciona una interesante y útil alternativa para la representación de formas de objetos. Si tenemos un objeto en una región que viene dado por los puntos en los que $f(x,y) > 0$, definimos el momento de orden p, q como:

$m_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x^p y^q f(x,y) dx dy$	Ecuación 3.6
---	--------------

El interés de estos momentos generales desde el punto de vista de la caracterización discriminante de los contornos de los objetos es que, estos contornos pueden modelarse como un tipo especial de funciones $f(x, y)$ acotadas y por ende se pueden calcular los momentos generales. Aquí radica su aplicabilidad en el reconocimiento de formas, dada una función acotada $f(x,y)$ existe un conjunto de momentos generales y viceversa.

Dado un conjunto de momentos generales se puede reconstruir una función $f(x,y)$ única, simbólicamente:

$f(x,y) \leftrightarrow \{m_{pq}\}; p, q = 0, 1, \dots, \infty$	Ecuación 3.7
---	--------------

Obviamente esta correspondencia biunívoca no representaría ningún interés práctico, a menos que se pudiera reducir el número de momentos generales a una cantidad manejable.

Particularizando este enfoque basado en la descripción de una función acotada $f(x,y)$ mediante un conjunto finito de sus momentos generales al problema del reconocimiento automático de los contornos (es decir formas cerradas) de los objetos en una imagen digital, es preciso pasar de una integral doble a un doble sumatorio, puesto que la función $f(x,y)$

es ahora una función $I(x,y)$ acotada que toma valores distintos de cero únicamente en el contorno del objeto y en su interior.

Los momentos generales discretos de la función $I_0(x,y)$ está dado por la siguiente relación:

$m_{pq} = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} x^p y^q I_0(x,y) \quad ; \quad 0, 1, 2, \dots, \infty$	Ecuación 3.8
--	--------------

Dependiendo de cómo se defina $I_0(x,y)$ se tendrá cuatro posibles momentos generales

$I_0(x,y) = 1 \text{ contorno}$ $I_0(x,y) = 0 \text{ resto}$	Ecuación 3.9
--	--------------

$I_0(x,y) = I(x,y) \text{ contorno}$ $I_0(x,y) = 0 \text{ resto}$	Ecuación 3.10
---	---------------

$I_0(x,y) = 1 \text{ objeto}$ $I_0(x,y) = 0 \text{ resto}$	Ecuación 3.11
--	---------------

$I_0(x,y) = I(x,y) \text{ objeto}$ $I_0(x,y) = 0 \text{ resto}$	Ecuación 3.12
---	---------------

Se puede definir una función acotada $I_0(x,y)$ que tome los valores (1,0) o bien valores ($I(x,y)$,0). Siendo $I(x,y)$ el nivel de intensidad luminosa en la escala de grises. Igualmente se puede establecer el campo de existencia de la función $I_0(x,y)$ bien como el contorno del objeto o bien como el contorno más su interior.

Los momentos generales que se obtienen a partir de su contorno, aunque lo caracterizan adecuadamente, sufren el grave defecto de ser muy sensibles al ruido y a pequeñas variaciones en la forma de un

contorno (siendo preciso tener un mayor número de momentos generales para robustecer la descripción del contorno). Por el contrario, los momentos generales basados en el contorno más su interior presentan una mayor robustez.

Para la obtención de características discriminantes de un contorno cerrado es habitual trabajar con la tercera opción, es decir con los momentos generales basados en la función $I_0(x,y)$:

$I_0(x,y) = 1 \text{ objeto}$ $I_0(x,y) = 0 \text{ resto}$	Ecuación 3.13
--	---------------

Un momento de gran interés es el de orden cero (el orden de un momento viene dado por la suma de los índices p y q) que coincide con el área del objeto:

$m_{00} = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} I_0(x,y)$	Ecuación 3.14
---	---------------

El área física se puede calcular sin más que multiplicar m_{00} por el área física de un pixel, lo cual obliga a un calibrado previo de la cámara. Afortunadamente, desde el punto de vista del reconocimiento de objetos veremos que no es necesario pasar a las dimensiones físicas del objeto, ya que es posible obtener momentos invariantes a traslaciones, giros y tamaños relativos de los objetos.

Los momentos de orden uno, m_{01} y m_{10} junto con m_{00} , el cual determinan el llamado centro de gravedad del objeto.

$x = \frac{m_{10}}{m_{00}} = \frac{\sum \sum x I(x,y)}{\sum \sum I(x,y)}$	Ecuación 3.15
---	---------------

$y = \frac{m_{01}}{m_{00}} = \frac{\sum \sum y I(x,y)}{\sum \sum I(x,y)}$	Ecuación 3.16
---	---------------

3.3.2 Descriptores de Textura:

3.3.2.1 Niveles de gris

Se empieza a partir del histograma del objeto.

3.3.2.2 Distribución espacial de niveles

Se empieza a partir de la información de distribución espacial del nivel de gris.

3.4 Momentos Invariantes

El inconveniente de los momentos generales reside en el hecho de que varía con la posición del objeto dentro del plano de la imagen, así como con el tamaño relativo del objeto, que depende de la distancia entre la cámara y el objeto.

Los cambios en la posición del objeto se deben exclusivamente a las operaciones de giro o traslación. En cuanto a los cambios de tamaños se puede modelar como una operación de homotecia. Por tanto, vamos a transformar sucesivamente los momentos generales para hacerlos invariante a giros y homotecias.

3.4.1 Momentos Invariante a Traslaciones

Los momentos generales se pueden hacer invariantes a traslaciones en el plano sin más que referirlos al centro de gravedad (x_{cg}, y_{cg}) , obteniéndose los llamados momentos centrales:

$u_{pq} = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} (x - x_{cg})^p (y - y_{cg})^q I_0(x, y)$	Ecuación 3.17
--	---------------

Puesto que el centro de masas (x_{cg}, y_{cg}) ocupa siempre la misma posición relativa respecto a todos los puntos del objeto, los momentos centrales no varían ante traslaciones de los objetos. El cálculo directo de

los momentos centrales es posible en función de los momentos generales

$u_{pq} = \sum_{r=0}^p \sum_{s=0}^q (r)^p (s)^q (-x_{cg})^r (-y_{cg})^s m_{p-r, q-s}$	Ecuación 3.18
---	---------------

Así aplicando esta expresión se obtendría los siguientes momentos centrales:

$u_{00} = m_{00}$	Ecuación 3.19
-------------------	---------------

$u_{01} = u_{10} = 0$	Ecuación 3.20
-----------------------	---------------

$u_{11} = m_{11} - \frac{m_{10}m_{01}}{m_{00}}$	Ecuación 3.21
---	---------------

$u_{20} = m_{20} - \frac{m_{10}^2}{m_{00}}$	Ecuación 3.22
---	---------------

$u_{02} = m_{02} - \frac{m_{01}^2}{m_{00}}$	Ecuación 3.23
---	---------------

$u_{03} = m_{03} - \frac{3m_{01}m_{02}}{m_{00}} + \frac{2m_{01}^3}{m_{00}^2}$	Ecuación 3.24
---	---------------

$u_{30} = m_{30} - \frac{3m_{10}m_{20}}{m_{00}} + \frac{2m_{10}^3}{m_{00}^2}$	Ecuación 3.25
---	---------------

$u_{21} = m_{21} - \frac{m_{01}m_{20}}{m_{00}} - \frac{2m_{10}m_{11}}{m_{00}} + \frac{2m_{10}^2m_{01}}{m_{00}^2}$	Ecuación 3.26
---	---------------

$u_{12} = m_{12} - \frac{m_{10}m_{02}}{m_{00}} - \frac{2m_{01}m_{11}}{m_{00}} + \frac{2m_{01}^2m_{10}}{m_{00}^2}$	Ecuación 3.27
---	---------------

Aunque no se obtengan unos momentos estrictamente invariantes ante cambios del tamaño relativo de un objeto, es decir invariantes a homotecias, suele emplearse la siguiente normalización, en donde los momentos centrales normalizados de orden $p+q$ se define según la siguiente ecuación:

$n_{pq} = \frac{u_{pq}}{u_{00}}$	Ecuación 3.28
----------------------------------	---------------

Donde:

$y = \frac{p + q + 2}{2} ; \quad p + q = 2, 3, \dots, \infty$	Ecuación 3.29
---	---------------

Como ya dijimos anteriormente, el problema práctico fundamental es determinar un número mínimo, pero suficiente, de momentos invariantes que permitan caracterizar cada uno de los objetos del universo de trabajo del sistema de reconocimiento automático.

Para evaluar el grado de reconstrucción que se obtendría de un conjunto finito y truncado de momentos. Este proceso de reconstrucción depende de la forma o contorno de cada objeto.

Cuando más irregular sea este, en general se necesitara un mayor número de momentos para su reconstrucción.

Aunque no se ha dicho, es preciso subrayar que en la obtención del subconjunto de momentos invariantes discriminantes se comienza siempre desde los momentos de menor a mayor orden. Esto es porque es en los momentos de menor orden donde están concentrados las principales características de las formas de un contorno cerrado. Los momentos de orden superior contribuyen en menor grado a la caracterización de la forma.

3.4.2 Momentos Invariante a Traslaciones, Rotacionales y Homotecias

De los momentos de segundo y tercer orden, pueden derivarse 7 de los llamados "momentos invariantes" que, no dependen del tamaño ni la posición del objeto, pudiendo ser usados para la identificación de objetos. El usuario puede decidir cuáles de estos momentos invariantes tendrán mayor significación en su propia aplicación. El siguiente conjunto de momentos invariantes se puede obtener usando únicamente los momentos centrales normalizados de orden 2 y 3:

$\varphi_1 = n_{20} + n_{02}$	Ecuación 3.30
-------------------------------	---------------

$\varphi_2 = (n_{20} - n_{02})^2 + 4n_{11}^2$	Ecuación 3.31
---	---------------

$\varphi_3 = (n_{30} - 3n_{12})^2 + (3n_{21} - n_{03})^2$	Ecuación 3.32
---	---------------

$\varphi_4 = (n_{30} + 3n_{12})^2 + (n_{21} + n_{03})^2$	Ecuación 3.33
--	---------------

$\begin{aligned} \varphi_5 = & (n_{30} - 3n_{12})(n_{30} + n_{12})[(n_{30} + n_{12})^2 - 3(n_{21} + n_{03})^2] \\ & + (3n_{21} - n_{03})(n_{21} \\ & + n_{03})[3(n_{30} + n_{12})^2 - (n_{21} + n_{03})^2] \end{aligned}$	Ecuación 3.34
---	---------------

$\begin{aligned} \varphi_6 = & (n_{20} - n_{02})[(n_{30} + n_{12})^2 - (n_{21} + n_{03})^2] + 4n_{11}(n_{30} \\ & + n_{12})(n_{21} + n_{03}) \end{aligned}$	Ecuación 3.35
---	---------------

$\begin{aligned} \varphi_7 = & (3n_{21} - n_{03})(n_{30} + n_{12})[(n_{30} + n_{12})^2 - 3(n_{21} + n_{03})^2] \\ & + (3n_{21} - n_{03})(n_{21} \\ & + n_{03})[3(n_{30} + n_{12})^2 - (n_{21} + n_{03})^2] \end{aligned}$	Ecuación 3.36
---	---------------

Este conjunto de momentos resulta ser invariante a la traslación, la rotación y el cambio de escala. Otra ventaja de los momentos es su gran facilidad de cálculo, sobre todo aplicándolos a imágenes binarias, ya que pueden ser calculados óptimamente a velocidades elevadas.

3.5 Cálculo de los Momentos generales a partir del código de Cadena.

Una vez planteado y desarrollado el fundamento teórico de la aplicación de los momentos de una función bidimensional acotada $I_0(x,y)$, vamos a ver cómo realizar su cálculo a partir, precisamente del código de cadena del contorno de $I_0(x,y)$.

El teorema de Green declara que si una función $f(x,y)$ es integrable sobre una región A y puede ser descompuesta como la suma de las derivadas de dos funciones, a través de la siguiente ecuación:

$f(x,y) = \frac{\partial Q(x,y)}{\partial x} - \frac{\partial P(x,y)}{\partial y}$	Ecuación 3.37
--	---------------

A partir de la ecuación anterior se obtiene lo siguiente:

$\iint_A f(x,y) dx dy = \int_C P(x,y) dx + Q(x,y) dy$	Ecuación 3.38
---	---------------

Dónde:

- C : es una curva cerrada (muchas pequeñas curvas finitas)
- La función $f(x,y)$ es el momento central $x^p y^q$.

La región A esta limitada por la curva C , tal como se representa en la figura 3.4; en donde el convenio de los ejes coordenados cartesiano no influyen en el resultado del teorema y que establece la siguiente ecuación 3.39.

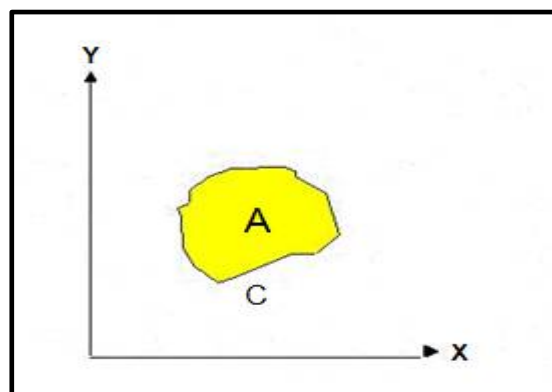


Figura 3.7 Región continua y acotada.

$\iint_A \left(\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y} \right) dx dy = \int_C P dx + Q dy$	Ecuación 3.39
---	---------------

El teorema permite pasar de una integral doble extendida sobre la región A, a una integral curvilínea sobre una curva cerrada C. Este resultado nos va permitir calcular los momentos generales, que están definidos mediante integrales dobles, como integrales curvilíneas extendidas sobre el contorno de un objeto, es decir utilizando el código de cadena que es una aproximación poligonal del contorno.

$P = -x^p y^{q+1}$ $Q = x^{p+1} y^q$	Ecuación 3.40
--------------------------------------	---------------

Entonces:

$\frac{\partial P}{\partial y} = -(q+1)x^p y^q$ $\frac{\partial Q}{\partial x} = (p+1)x^p y^q$	Ecuación 3.41
--	---------------

Por tanto se obtiene:

$\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y} = (p+q+2)x^p y^q$	Ecuación 3.42
--	---------------

Sustituyendo la **ecuación 3.6** en la **ecuación 3.3** y recordando la definición de momento general de orden **p+q** dada por la ecuación **3.4**, se obtiene la siguiente ecuación:

$m_{pq} = \iint_A x^p y^q dx dy = \frac{1}{p+q+2} \int_C -x^p y^{q+1} dx + x^{p+1} y^q dy$	Ecuación 3.43
--	---------------

En donde la función $I_0(x,y)$, asociada al objeto vale la unidad en el propio objeto (es decir, en la región A ocupada por el) y cero en el resto del plano.

La integral curvilínea de la **ecuación 3.37** puede calcularse en función del código de cadena del contorno del objeto. Para eso consideramos un tramo del contorno de un objeto dentro de una imagen digital:

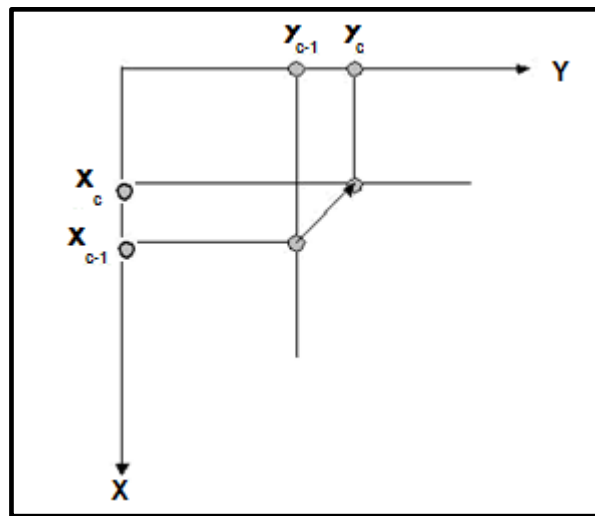


Figura 3.8 Tramo de un pixel.

La ecuación de una recta genérica que une dos pixeles consecutivos del contorno es:

$y - y_c = \frac{dy_c}{dx_c}(x - x_c) = m_c(x - x_c)$	Ecuación 3.44
---	---------------

Siendo:

$dy_c = y_c - y_{c-1}$	Ecuación 3.45
------------------------	---------------

$dx_c = x_c - x_{c-1}$	Ecuación 3.46
------------------------	---------------

Donde los (x_c, y_c) y (x_{c-1}, y_{c-1}) son las coordenadas de dos pixeles consecutivos del contorno. Despejando de la ecuación 3.44, se obtiene la siguiente ecuación:

$y = y_c + m_c(x - x_c)$	Ecuación 3.47
--------------------------	---------------

Permite expresar la integral curvilínea de la ecuación 3.43, que da el momento general de orden $p+q$ en función de x exclusivamente. Para ver esto vamos a calcular el momento general de orden cero m_{00} , de la siguiente manera:

$m_{00} = \frac{1}{p+q+2} \int_c x^{p+1} y^q dy - x^p y^{q+1} dx$	Ecuación 3.48
---	---------------

$m_{00} = \frac{1}{2} \int_c x dy - y dx = \frac{1}{2} \int_c m_c x - [y_c + m_c(x - x_c)] dx$	Ecuación 3.49
--	---------------

En la **figura 3.6**, se ve el contorno de un objeto definido por el código de cadena permite expresar la integral curvilínea de la **ecuación 3.48** como una sumatoria.

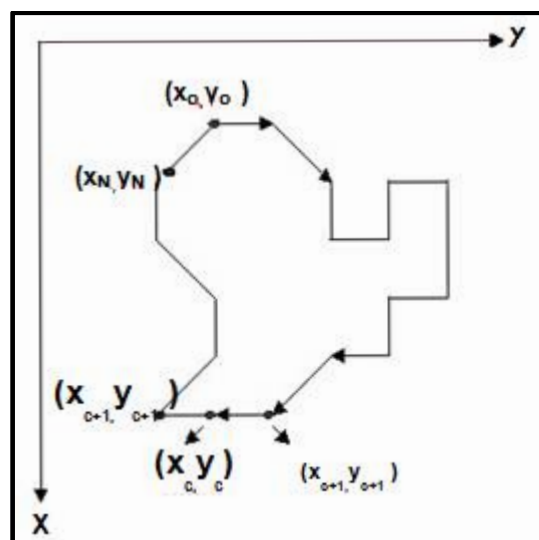


Figura 3.9 Curva del Área.

Se puede expresar de la siguiente manera a través de las siguientes ecuaciones:

$m_{00} = \frac{1}{2} \sum_{c=1}^N \int_{x_{c-1}}^{x_c} m_c x - [y_c + m_c(x - x_c)] dx$	Ecuación 3.50
--	---------------

$m_{00} = \frac{1}{2} \sum_{c=1}^N \int_{x_{c-1}}^{x_c} (m_c x_c - y_c) dx$	Ecuación 3.51
---	---------------

Observar que el índice sumatorio se refiere al orden del enlace del código de cadena, desarrollado en la **ecuación 3.51**.

$m_{00} = \frac{1}{2} \sum_{c=1}^N (m_c x_c - y_c)(x_c - x_{c-1})$	Ecuación 3.52
--	---------------

$m_{00} = \frac{1}{2} \sum_{c=1}^N \frac{dy_c}{dx_c} (x_c - y_c) dx_c$	Ecuación 3.53
--	---------------

$m_{00} = \frac{1}{2} \sum_{c=1}^N (x_c dy_c - y_c dx_c)$	Ecuación 3.54
---	---------------

Obtener la sumatoria de la ecuación 3.54 es de inmediato dado el código de cadena de un objeto; ya que se conoce las coordenadas de los pixeles y la siguiente tabla:

CC	0	1	2	3	4	5	6	7
dx _c	0	-1	-1	-1	0	+1	+1	+1
dy _c	+1	+1	0	-1	-1	-1	0	+1

Realizando: $A_c = x_c dy_c - y_c dx_c$, el momento de orden cero es:

$m_{00} = \frac{1}{2} \sum_{c=1}^N A_c$	Ecuación 3.55
---	---------------

Análogamente se puede obtener los momentos de mayor orden:

$m_{10} = \frac{1}{3} \sum_{c=1}^N A_c x_c - \frac{dx_c}{2}$	Ecuación 3.56
--	---------------

$m_{01} = \frac{1}{3} \sum_{c=1}^N A_c y_c - \frac{dy_c}{2}$	Ecuación 3.57
--	---------------

$m_{20} = \frac{1}{4} \sum_{c=1}^N A_c x_c^2 - x_c dx_c + \frac{1}{3} d^2 x_c$	Ecuación 3.58
--	---------------

$m_{02} = \frac{1}{4} \sum_{c=1}^N A_c y_c^2 - y_c dy_c + \frac{1}{3} d^2 y_c$	Ecuación 3.59
--	---------------

$m_{11} = \frac{1}{4} \sum_{c=1}^N A_c x_c y_c - \frac{1}{2} (x_c dy_c + y_c dx_c) + \frac{1}{3} (dx_c dy_c)$	Ecuación 3.60
---	---------------

$m_{30} = \frac{1}{-5} \sum_{c=1}^N A_c (x_c - \frac{dx_c}{2}) (x_c^2 - \frac{d^2 x_c}{2} - x_c dx_c)$	Ecuación 3.61
--	---------------

$m_{03} = \frac{1}{-5} \sum_{c=1}^N A_c (y_c - \frac{dy_c}{2}) (y_c^2 - \frac{d^2 y_c}{2} - y_c dy_c)$	Ecuación 3.62
--	---------------

CAPITULO 4

4. RECONOCIMIENTO DE PATRONES

4.1 Introducción

Una definición formal de Reconocimiento de Patrones es la siguiente: "es la categorización de datos de entrada en clases identificadas, por medio de la extracción de características significativas o atributos de los datos extraídos de un medio ambiente que contiene detalles irrelevantes". Matemáticamente hablando, la clasificación consiste en la partición del espacio n -dimensional definido por las características de un objeto, en varias regiones, donde cada región corresponde a una clase. Un sistema de reconocimiento de patrones tiene uno de los siguientes objetivos:

- Identificar el patrón como miembro de una clase ya definida (clasificación supervisada).
- Asignar el patrón a una clase todavía no definida (clasificación no supervisada, agrupamiento o clustering).

El reconocimiento automático de formas gira entorno a varios conceptos claves que se definen algunos a continuación:

- Patrón: un patrón es una descripción cuantitativa o estructural de un objeto o alguna entidad de interés.
- Clase: una clase de patrones es un conjunto de patrones que comparten algunas propiedades.
- Vector de características: la composición de varias características en un vector se denomina vector de características, esta contiene la medida de las características de un patrón; puede estar formado de números binarios o valores reales. Un vector de característica define puntos en un espacio n -dimensional.

4.2 Enfoque de un Sistema de Reconocimiento

En la literatura pueden encontrarse diferentes enfoques para este problema, motivadas por la diversidad de tareas de reconocimiento que

pueden abordarse. Sin embargo podemos identificar en general, 4 tipos de metodologías para el reconocimiento de patrones:

4.2.1 Heurísticas

Esta metodología es la que hace uso de la experiencia y la intuición humana. Por lo general, los sistemas que son desarrollados bajo estos métodos, están hechos a la medida del problema que se desea resolver.

4.2.2 Matemáticas

Este tipo de metodología hace uso de las propiedades comunes de los patrones y se basan en las reglas de clasificación formuladas en un marco matemático. Además, este enfoque se divide a su vez en otras dos categorías: las determinísticas y las estadísticas.

- **Determinísticas**

No requieren de propiedades estadísticas, pero son limitadas. Entre los métodos determinísticos podemos encontrar, por ejemplo, la clasificación por distancia Euclídea.

- **Estadística**

La aproximación más simple (y no por ello la menos eficiente) consiste en representar cada patrón mediante un vector de números, y cada clase por uno o varios patrones prototipo. Dado que existe variabilidad en las medidas registradas, cada componente del vector es una variable aleatoria y cada uno de sus valores es una realización de esa variable aleatoria.

Con esta aproximación un patrón no es más que un punto en el espacio de representación de los patrones, que es un espacio de dimensionalidad determinada por el número de variables consideradas.

Esta aproximación concluye que es razonable que los patrones pertenecientes a una misma clase estén cercanos en el espacio de representación mientras que aquellos que pertenezcan a clases diferentes deberían estar en diferentes

regiones del espacio de representación. Dentro de los métodos estadísticos podemos encontrar a los clasificadores basados en la regla de clasificación de Bayes.

4.2.3 Linguísticas

Las técnicas de este tipo reducen un objeto (habitualmente ya binarizado) a un conjunto de elementos estructurales o 'primitivas'. Si a esto unimos una sintaxis para relacionar estos elementos de forma espacial, obtenemos lo que se llama una representación sintáctica. Se trata de descomponer objetos complejos en términos de conjuntos de primitivas simples (como son arcos, ángulos, rectas, etc.) con reglas para describir la relación espacial entre ellas.

Hay que decir que es importantísimo el diseño de la sintaxis y de sus primitivas y se requiere habitualmente la especificación de un complejo conjunto de reglas que den una correspondencia unívoca entre la estructura y su representación. Además suelen surgir problemas de dependencia de la orientación de la estructura, tamaño, varias representaciones para una misma estructura, etc.

4.2.4 Redes Neuronales Artificiales

Las Redes Neuronales tienen muchas similitudes con el reconocimiento estadístico de patrones (REP) concerniente en la representación de datos y los principios de clasificación. La implementación práctica es sin embargo muy diferente. El modo de análisis implica la configuración de una red de neuronas artificiales y el entrenamiento de la red para determinar cómo las neuronas individuales pueden afectar uno a la otra. El modo de reconocimiento implica el envío de datos a través de la red y la evaluación a que clase se aproximará más.

4.3 Reconocimiento Estadístico de Patrones (REP)

El Reconocimiento Estadístico de Patrones es una disciplina relativamente madura hasta el punto de que existen ya en el mercado sistemas comerciales de reconocimiento de patrones que emplean esta técnica. En REP, un patrón se representa por un vector numérico de dimensión n ; de esta forma, un patrón es un punto en un espacio de características. n -dimensional.

Un REP funciona en dos modos diferentes: entrenamiento y reconocimiento. En modo de entrenamiento, se diseña el extractor de características para representar los patrones de entrada y se entrena al clasificador con un conjunto de datos de entrenamiento de forma que el número de patrones mal identificados se minimice. En el modo de reconocimiento, el clasificador ya entrenado toma como entrada el vector de características de un patrón desconocido y lo asigna a una de las clases o categorías.

El proceso de toma de decisiones en un REP se puede resumir como sigue. Dado un patrón representado por un vector de características

$X = (x_1, x_2, x_3 \dots x_n)'$	Ecuación 4.1
----------------------------------	--------------

Asignar a una de las c clases o categorías, c_1, c_2, \dots, c_c . Decimos que la densidad o probabilidad condicional $p(x | C_i)$ es la probabilidad de que la variable aleatoria C_i , sea precisamente x . Dependiendo del tipo de información disponible sobre las densidades condicionales de las clases, se pueden diseñar varias estrategias de clasificación. Si todas las densidades condicionales $p(x | C_i)$, $i=1,2,\dots,c$ se conocen, la regla de decisión es la de Bayes que establece los límites entre las diferentes clases. Sin embargo, en la práctica las densidades condicionales no se conocen y deben ser estimadas (aprendidas) partiendo de los patrones de entrada. Si se conoce la forma funcional de estas densidades pero no sus parámetros, el problema se llama de toma de decisión paramétrico. En caso contrario, estamos ante un problema de toma de decisión no paramétrico.

4.3.1 Representación de los Patrones

La entrada a un sistema de reconocimiento estadístico de patrones es un vector numérico que contiene los valores muestreados y cuantizados (o binarizados) de una serie de señales naturales. De una manera más formal, suponiendo patrones n -dimensionales, un patrón X es una variable aleatoria n -dimensional compuesta por n componentes x_1, x_2, \dots, x_n .

Con esta aproximación un patrón no es más que un punto en el espacio es una representación de los patrones que espacio de dimensionalidad determinada por el número de variables consideradas. Esta aproximación concluye que es razonable que los patrones pertenecientes a una misma clase estén cercanos en el espacio de representación mientras que aquellos que pertenezcan a clases diferentes deberían estar en diferentes regiones del espacio de representación.

4.3.2 Similitud entre Patrones

La tarea fundamental de un sistema de reconocimiento de patrones, es la de asignar a cada patrón de entrada una etiqueta. Dos patrones diferentes deberían asignarse a una misma clase si son similares y a clases diferentes si no lo son. La cuestión que se plantea ahora es la definición de una medida de similitud entre patrones.

Supongamos un sistema de adquisición perfecto (sin ruido), podemos asegurar que:

- La adquisición repetida del mismo patrón debería proporcionar la misma representación en el espacio de patrones. Por ejemplo, una cámara debería proporcionar siempre la misma imagen de la misma escena si las condiciones externas no cambiaran.
- Dos patrones diferentes deberían proporcionar dos representaciones diferentes.

- Una ligera distorsión aplicada sobre un patrón debería proporcionar una pequeña distorsión de su representación.

En definitiva, se supone que el proceso de adquisición es biunívoco y continuo. Estas consideraciones sugieren que si las representaciones de dos patrones están muy cercanas en el espacio de representación, entonces los patrones deben tener un alto grado de similitud. No obstante, no puede afirmarse tajantemente que a mayor distancia mayor disimilitud ya que la medida (absoluta) de distancia depende de la escala en la que se cuantifiquen las variables asociadas al patrón.

4.3.3 Variabilidad entre Patrones

La suposición de un sistema de adquisición perfecto no deja de ser eso, una suposición. Los sistemas de adquisición introducen, indefectiblemente cierta distorsión o ruido, lo que produce una variabilidad en la representación de los patrones. Aunque es posible controlar eficientemente en muchos casos esta distorsión mediante el calibrado de los sistemas de adquisición aparece otra fuente de variabilidad por la propia naturaleza de los patrones.

Con mucha frecuencia, patrones de una misma clase difieren, incluso significativamente. Un ejemplo sencillo es el de los sistemas de reconocimiento de caracteres OCR que pueden interpretar diversos tipos de letra, incluso caracteres escritos (bajo fuertes restricciones). Bien en el caso más simple, en el que se trata un solo tipo de letra, la variabilidad de los patrones se debe a factores tales como el granulado, color y calidad del papel o el tipo de tinta empleado.

La variabilidad intrínseca de los patrones hace que las representaciones tengan la forma de nubes de puntos en lugar de puntos individuales tal como se muestra en la **figura 4.1**. Esta última representación sería la de darse el caso de que no existiera

variabilidad entre los patrones. Estas nubes de puntos sugieren que patrones de una misma clase (similares) se representan cercanos (relativamente) en el espacio de representación mientras que patrones de diferentes clases (diferentes) se representan lejanos (relativamente) en ese espacio.

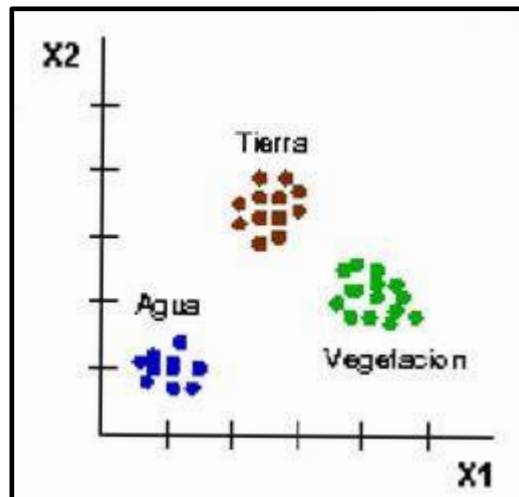


Figura 4.1 Variabilidad entre patrones.

Como vemos los patrones de una misma clase presentan cierta variabilidad natural. No obstante, deben estar (relativamente) cercanos en el espacio de representación y lejanos (relativamente) respecto a los patrones de otras clases.

Otra situación posible, y mucho más crítica, es la presencia de agrupamientos solapados. En la **figura 4.2** mostramos la representación de un conjunto de patrones pertenecientes a dos clases. Los patrones son bidimensionales y las variables consideradas son medidas de peso y altura de un conjunto de personas. Las clases informacionales son mujer y hombre. En este caso no es fácil la discriminación de estas clases con estos patrones.

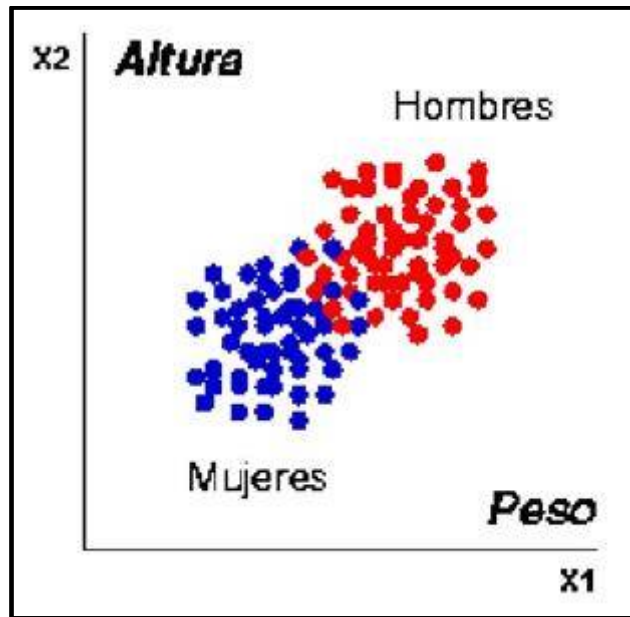


Figura 4.2 Agrupamientos solapados.

4.4 Diseño de un Reconocedor de Formas

- La primera etapa consiste en el establecimiento de las clases, en lo que podría denominarse como definición del universo de trabajo del sistema. En la mayoría de casos esta etapa es directa y trivial, el diseñador conoce perfectamente las clases de los objetos (piezas) que han de ser reconocidos.

Si suponemos que todos los patrones a reconocer son elementos potenciales de N clases distintas denotadas, $i = 1, 2, \dots, N$, llamaremos:

$\Omega = \{w_1, w_2, w_3, \dots, w_N\}$	Ecuación 4.2
--	--------------

Al conjunto de las clases. Conviene tener en cuenta que una clase es la denominación que se da a una clase conocida y con significado. Por ejemplo, un conjunto de clases sensato para el problema de clasificaciones de zonas en una área terrena sería:

$\Omega = \{agua, vegetación, suelo\}$	Ecuación 4.3
--	--------------

Resulta conveniente ampliar el conjunto Ω incorporando una nueva clase, llamada la clase de rechazo. Así, se define la clase de rechazo w_0 como una clase que se asigna a todos los patrones para los que no se tiene una certeza aceptable de ser clasificados correctamente en alguna de las clases de Ω . Se dice que:

$\Omega^* = \{w_1, w_2, w_3, \dots, w_N, w_0\}$	Ecuación 4.4
---	--------------

Dicha expresión es el conjunto extendido de clases informacionales. No obstante puede ocurrir que las clases puedan ser desconocidas a priori, en donde no está clasificado el universo de clases. En tales situaciones se recurre a técnicas denominadas de agrupación (clustering) o reconocimiento no supervisado. Estas técnicas difieren de las expuestas con anterioridad, ya que no existe una supervisión o reconocimiento externo que guíe el diseño de las funciones de discriminación.

- Una vez definida las clases, la siguiente operación consiste en la elección del vector de características. Esta etapa es crítica, y la bondad del sistema final estará completamente determinada por los rasgos (características) escogidos.

El vector característico, sin duda, constituye el elemento clave en un sistema de reconocimiento automático de formas. La elección de rasgos o características es muy dependiente de la aplicación concreta que se tenga, se trata más de un arte que de una ciencia y es la intuición y la experiencia quienes pueden guiar en su elección. En cualquier caso las características elegidas deben posibilitar que las clases sean disjuntas o separables.

- Una vez seleccionado el vector característico del sistema, la etapa final es ya puramente mecánica y se centra en el cálculo de las correspondientes funciones de decisión o funciones discriminantes. Dentro de las diferentes técnicas veremos solo algunas.

Una vez establecido el conjunto de clases se procede a la construcción del clasificador. Como puede intuirse, la construcción del clasificador no es una tarea trivial ni directa e involucra una serie de etapas:

- La elección del modelo.
- Aprendizaje (entrenamiento del clasificador).
- Verificación de los resultados.

Es muy importante señalar que estas etapas no deben verse de forma secuencial. Puede ocurrir que en un momento dado hay que volver atrás para replantearse alguno de los pasos dados, incluso el conjunto de clases informacionales. En la **figura 4.3** observamos el diagrama de flujo para el diseño de un reconocedor automático.

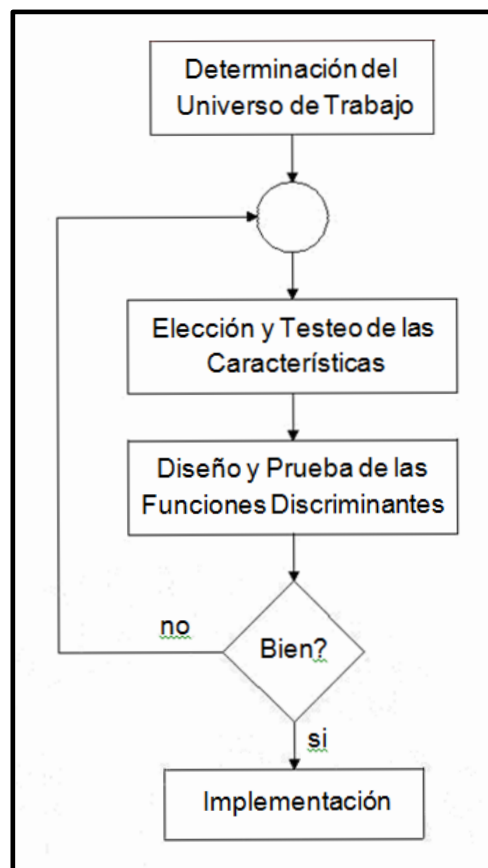


Figura 4.3 Diagrama de las etapas en el diseño del sistema de reconocimiento automático

4.5 Modelo de Clasificadores

Se suele utilizar indistintamente los términos aprendizaje y entrenamiento para referirse al proceso de construcción del clasificador. El aprendizaje en los clasificadores puede realizarse de dos maneras muy diferentes: aprendizaje Supervisado y aprendizaje no Supervisado.

4.5.1 Aprendizaje Supervisado

Un aprendizaje supervisado requiere disponer de un conjunto de patrones de los cuales se conoce su clase cierta. A este conjunto se le denomina conjunto de entrenamiento. Este tipo de entrenamiento se denomina entrenamiento supervisado y los clasificadores así obtenidos clasificadores supervisados.

En este caso se dispone de un conjunto de entrenamiento, los cuales han sido etiquetados previamente. Esta tarea la suele realizar un experto en el campo en el que se va a realizar el reconocimiento.

- **Modelo Paramétrico**

Suponemos que podemos modelar las clases mediante funciones de densidad de probabilidad conocidas: esta forma de abordar el problema se conoce como aproximación paramétrica. Como veremos, el aprendizaje se reducirá a la estimación de los parámetros que determinan las funciones de densidad de probabilidad de las clases. La herramienta fundamental que se usa será la estadística y particularmente la teoría de la decisión de Bayes.

Si se supone un completo conocimiento a priori de la estructura estadística de las clases, el aprendizaje se reduce a la estimación de los parámetros que determinan las funciones de densidad de probabilidad de las clases. Los clasificadores contruidos bajo esta suposición se conocen como clasificadores paramétricos.

En este planteamiento, los patrones de una clase dada son realizaciones de una distribución de probabilidad normal,

cuyos parámetros son el vector medio y la matriz de covarianza. Cada punto del espacio de representación tiene asociado un valor de densidad de probabilidad de pertenencia a cada clase.

Se dice que un punto X está en la región de decisión asociada a una clase si el valor de probabilidad de pertenencia a esa clase asociado a X es mayor que para cualquiera otra clase

- **Modelo No Paramétrico**

Si no se supone un determinado modelo estadístico, bien por desconocimiento o por la imposibilidad de asumir un modelo paramétrico adecuado, el problema resulta más complejo y se puede abordar desde diferentes perspectivas. Los clasificadores construidos sin esta suposición se conocen como clasificadores no paramétricos.

Cuando no puede suponerse un modelo paramétrico para las funciones de densidad de probabilidad asociadas a las clases se utilizan modelos no paramétricos para estimar las funciones de densidad de probabilidad. En estos casos la única información disponible para la clasificación es el conjunto de prototipos. El método más simple entre los no paramétricos es el del vecino más cercano, que consiste en etiquetar un patrón con la etiqueta del prototipo más cercano.

4.5.2 Aprendizaje No Supervisado

El aprendizaje no supervisado se realiza a partir de un conjunto de patrones del que no se conoce su clase cierta. Básicamente, se traduce en encontrar agrupamientos. El objetivo suele ser el de verificar la validez del conjunto de clases informacionales para una clasificación supervisada. Las técnicas utilizadas suelen denominarse métodos de agrupamiento o clustering.

CAPITULO 5

5. ALGORITMO DE PROCESAMIENTO DIGITAL DE IMÁGENES

5.1 Modelos de Color

Los modelos de color (o espacios o sistemas de color) son especificaciones de un sistema de coordenadas y un subespacio dentro de este sistema donde cada color es representado por un único punto, con la finalidad de facilitar las especificaciones de color.

5.1.1 Modelos RGB

El modelo RGB, cada color aparece representado en sus componentes espectrales primarios: rojo, verde y azul. Este modelo es basado en un sistema coordenado cartesiano. El subespacio de interés es el cubo mostrado en la figura 5.1, los colores rojo, verde y azul están en tres vértices. El color negro está en el origen y el blanco diametralmente opuesto a este. En la recta que une estos dos valores se encuentran todos los tonos de grises. Usualmente se normalizan los valores de R, G y B (es decir se han dividido entre el valor $R+G+B$) por lo que el cubo se convierte en un cubo unitario.

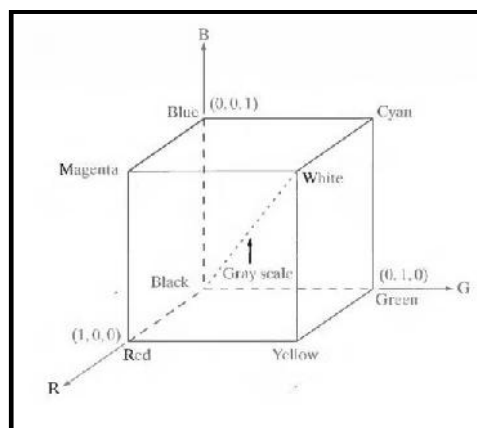


Figura 5.1 Modelo RGB

5.1.2 Modelo HSI

El modelo RGB, aun cuando es ideal para la implementación en un computador, no es el más adecuado para describir colores en términos que sean prácticos para la interpretación humana. Cuando vemos, por ejemplo, una vas, no nos referimos a su color por el porcentaje de rojo, azul y verde que lo componen. Los humanos describimos el color de un objeto mediante su tono (la descripción de un color puro como amarillo puro, rojo puro, etc.), saturación (el grado en que un color puro está “diluido” por luz blanca) y brillo. Al ser esta última cantidad algo muy subjetivo, en el presente sistema se usa la intensidad, que es una noción acromática.

El modelo HSI es una herramienta ideal para algoritmos basados en descripción de colores que son naturales e intuitivos para los humanos. El subespacio de este modelo consiste en una serie de círculos perpendiculares a un eje vertical que es la intensidad, la saturación es la distancia radial y el tono el ángulo que se forma con el eje del rojo.

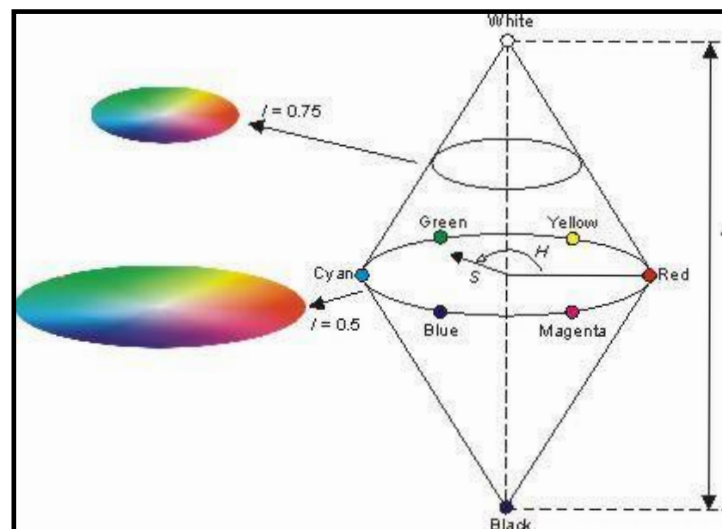


Figura 5.2 Modelo HSI

5.2 Filtrado de Imágenes

El proceso de filtrado es el conjunto de técnicas englobadas dentro del preprocesamiento de imágenes cuyo objetivo fundamental es obtener, a partir de una imagen origen, otra final cuyo resultado sea más adecuado para una aplicación específica mejorando ciertas características de la misma que posibilite efectuar operaciones del procesado sobre ella.

Los principales objetivos que se persiguen con la aplicación de filtros son:

- Suavizar la imagen: reducir la cantidad de variaciones de intensidad entre píxeles vecinos.
- Eliminar ruido: eliminar aquellos píxeles cuyo nivel de intensidad es muy diferente al de sus vecinos y cuyo origen puede estar tanto en el proceso de adquisición de la imagen como en el de transmisión.
- Realzar bordes: destacar los bordes que se localizan en una imagen.

Detectar bordes: detectar los píxeles donde se produce un cambio brusco en la función intensidad.

Por tanto, se consideran los filtros como operaciones que se aplican a los píxeles de una imagen digital para optimizarla, enfatizar cierta información o conseguir un efecto especial en ella.

5.2.1 Detección de Bordes

Los bordes de una imagen digital se pueden definir como transiciones entre dos regiones de niveles de gris significativamente distintos. Suministran una valiosa información sobre las fronteras de los objetos y puede ser utilizada para segmentar la imagen, reconocer objetos, etc.

La mayoría de las técnicas para detectar bordes emplean operadores locales basados en distintas aproximaciones discretas de la primera y segunda derivada de los niveles de grises de la imagen.

Operadores basadas en la primera derivada (Gradiente).

La derivada de una señal continua proporciona las variaciones locales con respecto a la variable, de forma que el valor de la derivada es mayor cuanto más rápidas son estas variaciones.

En el caso de funciones bidimensionales $f(x,y)$, la derivada es un vector que apunta en la dirección de la máxima variación de $f(x,y)$ y cuyo módulo es proporcional a dicha variación. Este vector se denomina gradiente y se define:

$$\nabla f(x,y) = \begin{bmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{bmatrix} \quad \text{Mag}[\nabla f(x,y)] = \sqrt{\left(\frac{\partial f(x,y)}{\partial x}\right)^2 + \left(\frac{\partial f(x,y)}{\partial y}\right)^2}$$
$$\theta = \arctan \frac{\frac{\partial f(x,y)}{\partial x}}{\frac{\partial f(x,y)}{\partial y}}$$

Figura 5.3 Vector gradiente.

En el caso bidimensional discreto, las distintas aproximaciones del operador gradiente se basan en diferencias entre los niveles de grises de la imagen. La derivada parcial $f_x(x,y)$ (gradiente de fila $G_f(i,j)$) puede aproximarse por la diferencia de píxeles adyacentes de la misma fila.

$$\frac{\partial f(x,y)}{\partial x} \approx \nabla_x f(x,y) = f(x,y) - f(x-1,y)$$

-1	1
----	---

Figura 5.4 Aproximaciones del vector gradiente.

La discretización del vector gradiente en el eje Y ($G_c(i,j)$), será:

$$\frac{\partial f(x,y)}{\partial y} \approx \nabla_y f(x,y) = f(x,y) - f(x,y-1)$$

-1
1

Figura 5.5 Discretización del vector gradiente.

El gradiente de la fila GF y de columna GC en cada punto se obtienen mediante la convolución de la imagen con las máscaras HF y HC, esto es:

$$G_F(i,j) = F(i,j) \otimes H_F(i,j)$$

$$G_C(i,j) = F(i,j) \otimes H_C(i,j)$$

Figura 5.6 Convolución de la imagen con las máscaras HF y HC.

La magnitud y orientación del vector gradiente suele aproximarse por la expresión:

$$|G(i,j)| = \sqrt{G_F^2 + G_C^2} \approx |G_F(i,j)| + |G_C(i,j)|$$

Figura 5.7 Magnitud del vector gradiente.

- **Operador Roberts**

Las máscaras utilizadas en este operador son:

Gradiente fila			Gradiente columna		
0	0	0	-1	0	0
0	0	1	0	1	0
0	-1	0	0	0	0

Figura 5.8 Máscaras del gradiente fila y columna

Obtiene buena respuesta ante bordes diagonales. Ofrece buenas prestaciones en cuanto a localización. El gran inconveniente de este operador es su extremada sensibilidad al ruido y por tanto tiene pobres cualidades de detección.

- **Operadores de Prewitt, Sobel y Frei-Chen.**

Los tres operadores pueden formularse de forma conjunta con las siguientes máscaras de convolución mostradas a continuación.

	Gradiente fila				Gradiente columna		
$\frac{1}{2+K}$	1	0	-1	$\frac{1}{2+K}$	-1	-K	-1
	K	0	-K		0	1	0
	1	0	-1		1	K	1

Figura 5.9 Máscaras de Prewitt, Sobel y Frei-Chen

En el operador Prewitt ($K=1$), se involucran a los vecinos de filas / columnas adyacentes para proporcionar mayor inmunidad al ruido.

El operador Sobel ($K=2$), se supone que es más sensible a los bordes diagonales que el de Prewitt aunque en la práctica hay poca diferencia entre ellos.

El Frei-Chen ($K= \sqrt{2}$), el gradiente es el mismo para bordes verticales, horizontales y diagonales.

5.2.2 Vecindad de Moore

La vecindad de Moore de un pixel P es el conjunto de 9 píxeles que comparten con P un vértice o una arista. También se les conoce como 8-vecinos o vecinos indirectos.

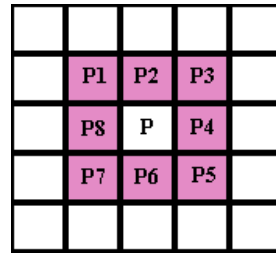


Figura 5.10 Vecindad de Moore de un pixel P

5.3 Fase de Construcción

5.3.1 Captura de la Imagen Digital

Para captar la imagen digital del producto a procesar se utilizó una cámara web modelo Creative P1001 y se utilizó iluminación LED que no afecte el circuito de control a base de sensores infrarrojos.

Para el manejo de la cámara web desde el Borland C++ Builder se utilizaron las APIs de Windows. Un API (Application Programming Interface) es el conjunto de funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción. Las APIs permiten:

- Depuración y manejo de errores
- E/S de dispositivos
- Varias DLL, procesos e hilos
- Comunicación entre procesos
- Manejo de la memoria
- Almacenamiento
- GDI (interfaz para dispositivos gráficos) de Windows

En el presente caso se utilizó la librería “Video for Windows”

(vfw.h) que permite el uso de APIs de Windows para el manejo de periféricos de Video.

A rasgos generales el procedimiento para manejar la cámara web es el siguiente:

- Creación de un “handle”, el cual se puede considerar como un tipo especial de puntero que permite a un programa o sistema operativo controlar objetos o bloques de memoria.
- Creación de una “Ventana de Captura” donde se visualizará los datos de la cámara web configurada y asociar esta ventana con el handle antes creado.
- Comunicación con la cámara web mediante la función SendMessage que permite enviarle órdenes al handle de la ventana de captura. Así, se puede controlar la conexión, desconexión, velocidad de refresco, etc.
- Copia de cierto frame de la ventana de captura al portapapeles, para luego ser reescalada (se eligió la resolución de 320 píxeles x 240 píxeles) y guardada en formato bmp.

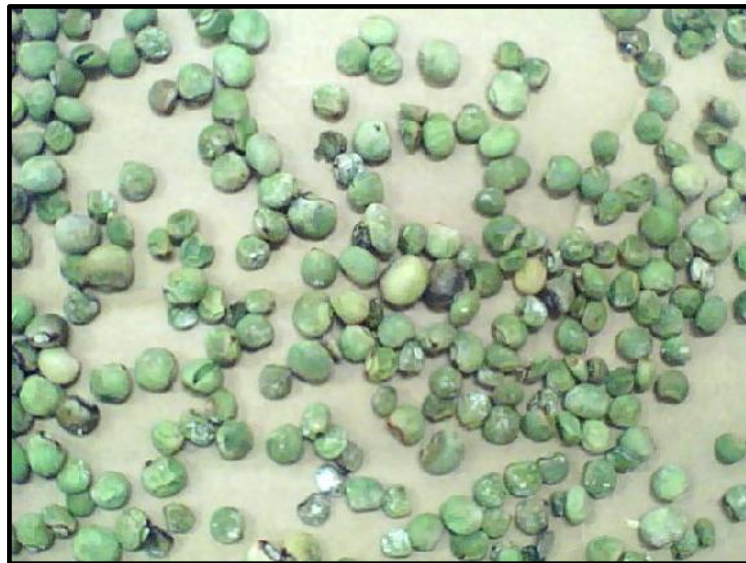


Figura 5.11 Imagen capturada de una cámara web.

5.3.2 Procesamiento de la Imagen

Luego de captada la imagen los pasos que se eligieron para el procesamiento son los siguientes:



Figura 5.12 Secuencia de procesos para procesar la imagen.

5.3.2.1 Conversión a escala de grises

Para el manejo de Imágenes se utilizó la clase `Graphics::TBitmap` utilizando la función `ScanLine` que permite la obtención de los valores RGB de una fila en formato de la estructura `RGBQUAD` (una estructura de 4

bytes cuyos elementos son R, G y B (cada uno es un byte) y alpha del píxel).

Como se explicó antes para la conversión a escala de grises se crea una matriz auxiliar donde los valores R, G y B son los tres iguales al promedio de los valores R, G y B originales.

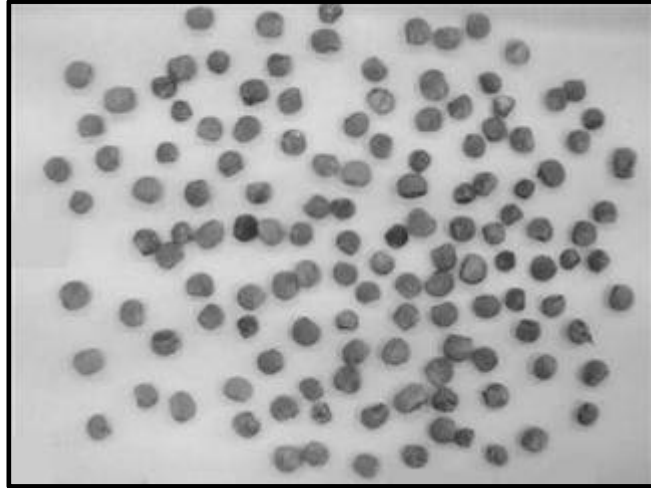


Figura 5.13 Imagen transformada a escala de grises.

5.3.2.2 Suavizado con Filtro Pasabajo

Se implementó una función genérica de convolución que toma como argumentos una matriz m_{ij} de 3x3 (la máscara) y un factor de división d , tal que la matriz resultado z_{ab} es:

$z_{ab} = \frac{1}{d} \sum_{i=1}^3 \sum_{j=1}^3 (m_{ij})(z_{a+i-2,b+j-2})$	ecuación 5.1
--	---------------------

Para el caso del filtro pasabajo la máscara es:

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Y el factor de división es 9

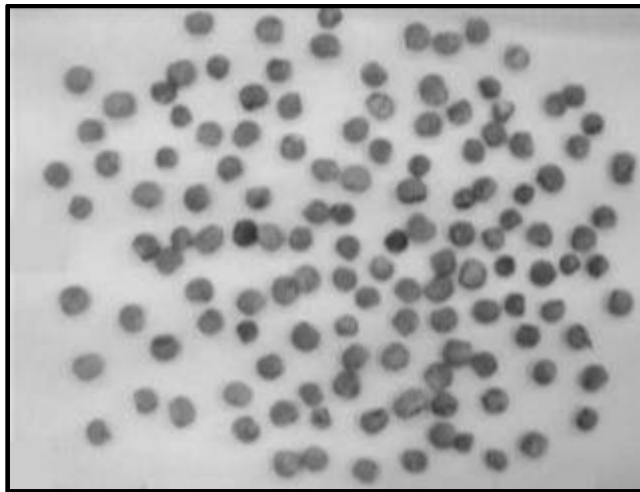


Figura 5.14 Imagen con filtro Pasa Bajo.

5.3.2.3 Binarización

Para la binarización se determinó experimentalmente mediante el análisis del histograma de la imagen que se muestra en la figura 5.14., y mediante la evaluación cualitativa de diferentes valores de umbral que el valor adecuado (en un rango de 0 a 255) es de 150.

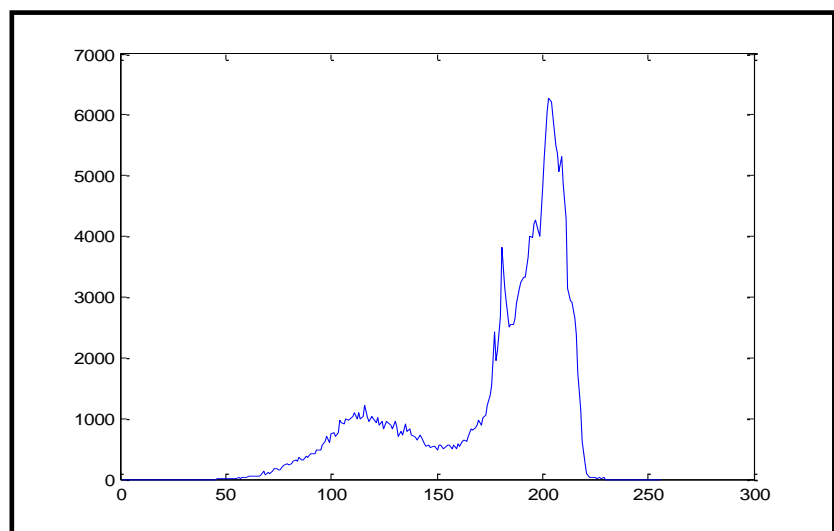


Figura 5.15 Análisis del Histograma de la Imagen.

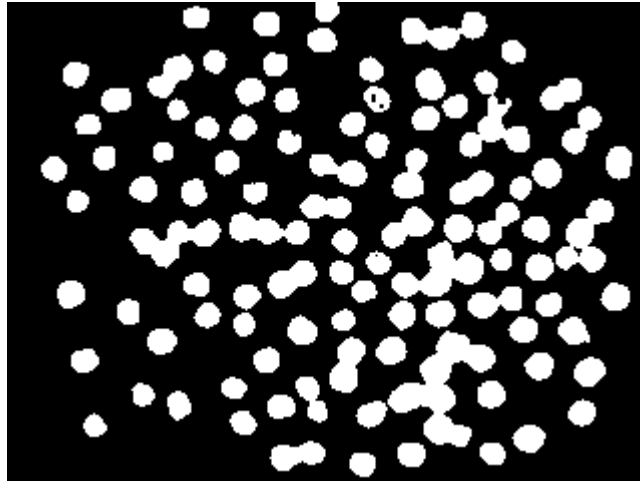


Figura 5.16 Resultado de la Binarización.

5.3.2.4 Operación de Cierre

La operación de cierre es parte de las operaciones morfológicas, las cuales tienen la propiedad de responder selectivamente a la forma local de las distintas estructuras de la imagen. La operación de cierre es la aplicación sucesiva de otras dos operaciones morfológicas básicas: dilatación y erosión.

Dilatación:

Es una operación que corresponde con la idea intuitiva de “agrandar” una región. Toda operación morfológica necesita de una matriz binaria $H(i,j)$ llamada “elemento estructural” el cual tiene un “punto base” que se utiliza como su origen de coordenadas.

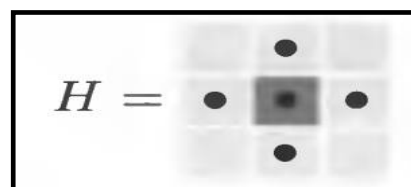


Figura 5.17 Elemento estructural (punto base es el centro).

La operación de dilatación de la imagen I mediante el elemento estructural H se define como:

$I \oplus H = \{(p + q) \text{para todo } p \in I \text{ y } q \in H\}$	ecuación 5.2
---	--------------

Intuitivamente esto corresponde a “copiar” el elemento estructural y colocarlo con su punto base en cada pixel con valor 1 de la imagen original I .

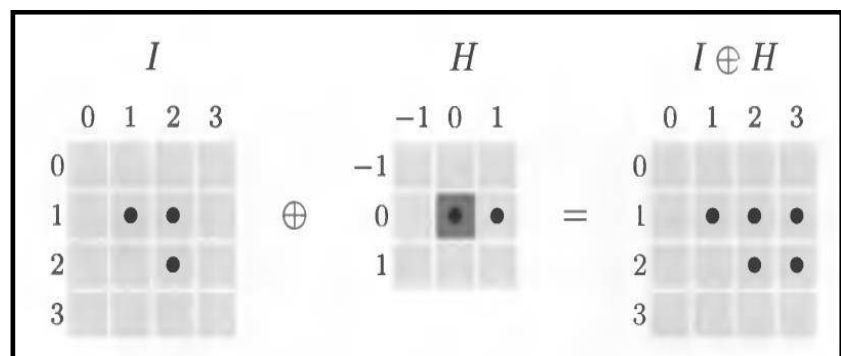


Figura 5.18 Proceso para dilatar una imagen.

Erosión

Coincide con la idea intuitiva de reducción o encogimiento de una región. Se define como:

$I \ominus H = \{p \in \mathbb{Z}^2 (p + q) \in I, \forall q \in H\}$	Ecuación 5.3
---	--------------

En términos prácticos, la erosión es una operación cuasi-inversa a la dilatación y se puede implementar siguiendo estos pasos:

- Invertir la imagen (los 1 en 0 y viceversa).
- Calcular la traspuesta H^* de H .
- Dilatar la imagen invertida con H^* como elemento estructural.
- Volver a invertir el resultado.

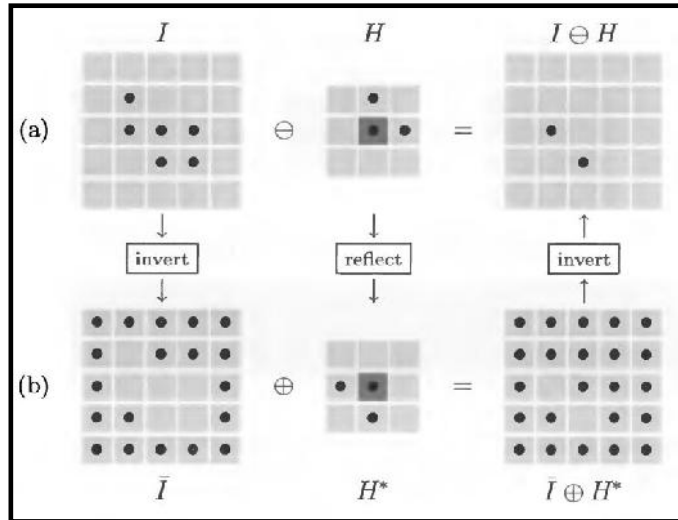


Figura 5.19 Proceso para erosionar una imagen.

La operación utilizada aquí es una operación compuesta de cierre que se define como:

$I \cdot H = (I \oplus H) \ominus H$	ecuación 5.4
--------------------------------------	--------------

Es decir, se realiza una dilatación y luego una erosión (en ese orden). Esta operación tiene como efecto eliminar agujeros existentes en regiones sólidas que tengan tamaño menor al del elemento estructural. En el presente caso, se eligió el elemento estructural clásico:

$$H = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Se observa que luego de la binarización existen pequeños agujeros (como en el objeto de la parte superior en la figura 5.16); al aplicar la operación de cierre se eliminan estos agujeros que pueden interferir en operaciones futuras (Como detección de contornos o cálculo del área)

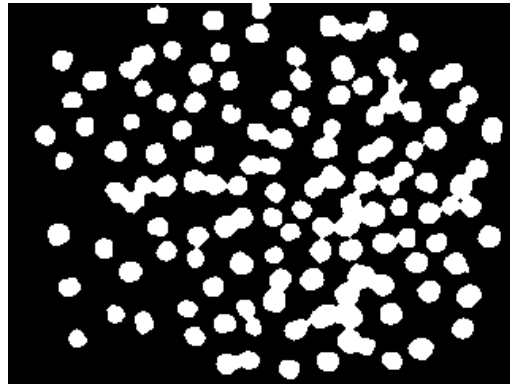


Figura 5.20. Imagen luego de la operación de cierre.

5.3.2.5 Segmentación Inicial de Regiones

Consiste en etiquetar con un mismo valor en una matriz auxiliar todos los píxeles que corresponden a un mismo objeto. El algoritmo elegido para marcar las regiones fue el Algoritmo de relleno por difusión (versión a lo ancho (“Breadth-first”). La idea de este algoritmo es la búsqueda de un pixel de objeto (valor=1) no marcado y luego se visitan y etiquetan el resto de los píxeles vecinos de esta región. El nombre del algoritmo se debe a que el algoritmo recuerda un flujo de agua que se difunde y llena una región a partir del pixel inicial. A continuación se presenta el pseudo-código del algoritmo:

```

ENTRADA: imagen I, punto inicial (u, v) , etiqueta_actual
Crear una cola vacía Q
Agregar (u, v) a la cola Q
Mientras Q no esté vacío hacer
    Sacar elemento frontal de la cola
     $(x, y) \leftarrow$  Sacar de la cola Q
    Si  $(x, y)$  está dentro de los límites de la imagen y  $I(x, y) = 1$ 
    entonces
         $I(x, y) \leftarrow$  etiqueta_actual
        Agregar  $(x+1, y)$  a la cola Q
        Agregar  $(x, y+1)$  a la cola Q
        Agregar  $(x, y-1)$  a la cola Q
        Agregar  $(x-1, y)$  a la cola Q

```

Figura 5.21. Pseudo-Código del Algoritmo de Segmentación.

En la siguiente imagen aparecen los resultados de la localización de las regiones de la imagen de la figura 5.19. Se muestra cada región diferente con un color aleatorio y el fondo en blanco.

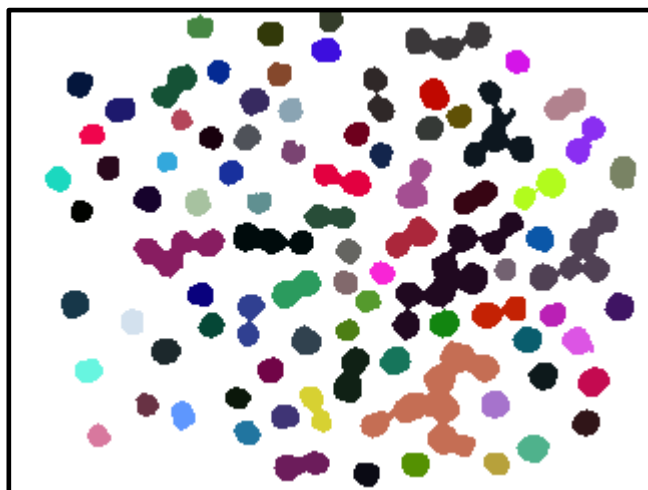


Figura 5.22. Algoritmo de Relleno por difusión.

5.3.2.6 Segmentación Objetos Solapados

Como se observa en la figura 5.19, existen algunos objetos que al estar unido, son tomados por el algoritmo de segmentación como una sola región. Esto lleva a un conteo erróneo de la cantidad de productos y una clasificación errada de estos mismos. Para solucionar este problema se eligió aplicar el algoritmo propuesto.

El algoritmo presenta una gran eficacia cuando se tienen varios objetos del mismo tipo con pocas zonas de solapado entre ellos. Los pasos del algoritmo son los siguientes:

- Aplicar la transformada distancia a la imagen.
- Localizar los núcleos de los objetos.
- Dilatar repetidamente los núcleos hasta encontrar zonas de solapamiento.

Transformada Distancia

La Transformada Distancia consiste en reemplazar el valor de un píxel en una región con la mínima distancia entre este y un píxel de fondo. Para ejecutar esta transformada se realizan dos escaneos. El primero de izquierda a derecha y de arriba abajo. Sea $f(i, j)$ el valor original del píxel, se asignan los valores según la fórmula:

$s(i, j) = \begin{cases} 0, & \text{Si } f(i, j) = 0 \\ \min\{(s(i, j - 1) + 1), (s(i - 1, j) + 1)\} & \text{Si } f(i, j) = 1 \end{cases}$	ecuación 5.5
--	--------------

Luego se realiza un segundo escaneo de derecha a izquierda y de abajo hacia arriba y se reasignan los valores de acuerdo a:

$d(i,j) = \begin{cases} 0, & \text{Si } s(i,j) = 0 \\ \min\{(d(i+1,j)+1), (d(i,j+1)+1), s(i,j)\} & \text{Si } s(i,j) \neq 0 \end{cases}$	ecuación 5.6
--	--------------

A continuación se muestra un ejemplo de dos objetos solapados. El objeto se muestra como 1 y el fondo como 0.

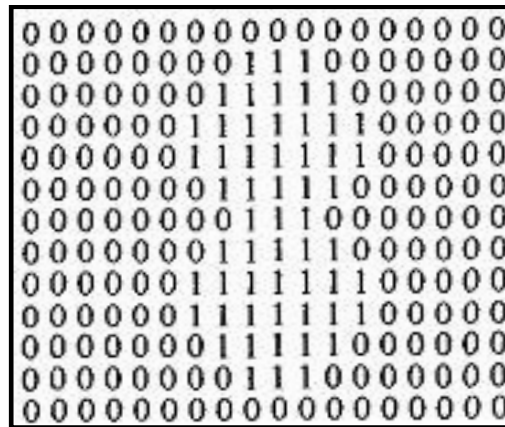


Figura 5.23. Imagen Binaria.

A continuación se muestra los resultados del primer escaneo:

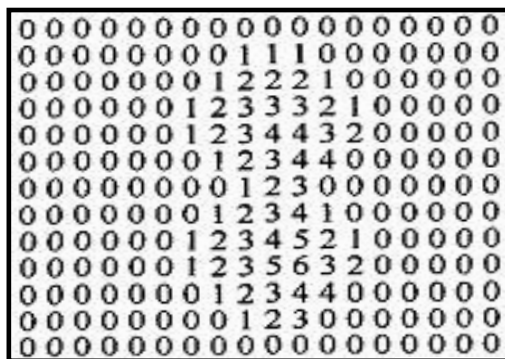


Figura 5.24. Resultados del primer escaneo.

A continuación se muestra los resultados del segundo escaneo:



Figura 5.25. Resultados del segundo escaneo.

Localizar los Núcleos de los Objetos

Los núcleos de los objetos son los que tienen la mayor distancia a los bordes de la región. Para hallarlos se realiza otro escaneo con la siguiente fórmula:

$d(i,j) = \begin{cases} d(i,j), & d(i,j) \geq \max\{vecindad\ Moore\} \\ 1, & \text{en otros casos} \end{cases}$	ecuación 5.7
--	--------------

Luego los núcleos son etiquetados para diferenciarlos por regiones.



Figura 5.26. Localización de núcleos.



Figura 5.27. Etiquetados de los diferentes núcleos.

Dilatación Repetida

Luego se dilatan las áreas de los núcleos varias veces hasta que se restauran las regiones originales y a su vez no pueden ser alteradas. Las uniones son marcadas con -1. Si $k = 2, 3, \dots$ son las etiquetas de los núcleos se aplica por cada núcleo la siguiente ecuación:

$G_k(i, j) = \begin{cases} k, & d(i, j) = k \\ k, & d(i, j) = 1 \wedge (\text{Algún vecino} = k) \\ -1, & d(i, j) \neq k \wedge d(i, j) > 1 \wedge (\text{Algún vecino} = k) \\ d(i, j), & \text{en otros casos} \end{cases}$	Ecuación 5.8
---	--------------

A continuación se muestra el ejemplo anterior con las uniones ya detectadas (se muestran cómo *).

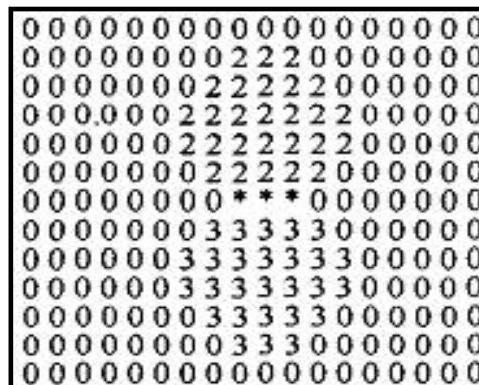


Figura 5.28. Detección de uniones.

5.3.2.7 Reconocimiento de Contornos

Para el almacenamiento de los contornos de las diferentes regiones se eligió guardarlos en su forma de código de 8-cadena (también llamados códigos de Freeman). En este tipo de codificación se guarda un punto de inicio x_s y se almacena una sucesión de cambios direccionales. Así para la sucesión de puntos $c_R = \{x_0, x_1, \dots, x_{M-1}\}$ con $x_i = (u_i, v_i)$ los elementos de la 8-cadena $c'_R = \{c'_0, c'_1, \dots, c'_{M-1}\}$ dados por

$c'_i = \text{dir}(\Delta u_i, \Delta v_i)$	ecuación 5.9
---	--------------

Donde:

$(\Delta u_i, \Delta v_i) = \begin{cases} (u_{i+1} - u_i, v_{i+1} - v_i), & \text{para } 0 \leq i < M - 1 \\ (u_0 - u_i, v_0 - v_i), & \text{para } i = M - 1 \end{cases}$	ecuación 5.10
--	---------------

Y donde $\text{dir}(\Delta u_i, \Delta v_i)$ se da en la siguiente tabla:

Δu	1	1	0	-1	-1	-1	0	1
Δv	0	1	1	1	0	-1	-1	-1
$\text{dir}(\Delta u_i, \Delta v_i)$	0	1	2	3	4	5	6	7

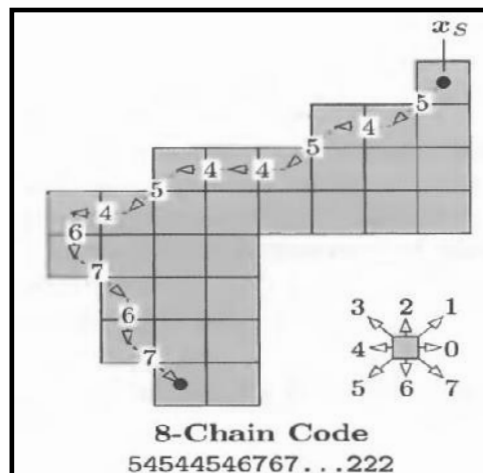


Figura 5.29. Código de 8-cadena.

En cuanto a la detección propiamente dicha de los contornos se utilizó el Algoritmo de Trazado de Moore que consiste en:

- Dada una Imagen Binaria (fondo=0 y objetos=1) escanear la imagen secuencialmente y elegir un pixel inicial con valor 1.
- A partir de este pixel “visitar” en sentido horario (o antihorario siempre y cuando se mantenga el mismo criterio durante todo el algoritmo) cada pixel de la vecindad de Moore hasta encontrar otro con valor 1. Se debe empezar el registro en la dirección en que se entró al pixel.
- Continuar hasta que se ingrese nuevamente al pixel inicial en la misma dirección en que se entró originalmente (a este criterio de parada se le llama criterio de parada de Jacob).

A continuación se muestra gráficamente este algoritmo (los pixeles de fondo son blancos y los de objeto verdes. Los píxeles de contorno se marcan con rojo):

- En este caso se escanea de izquierda a derecha y de arriba hacia abajo y se marca el punto inicial (rojo) al que se ingresó por la izquierda.

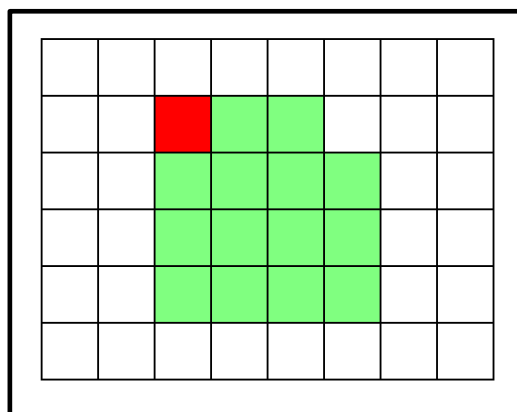


Figura 5.30. Primer paso del algoritmo de trazado de Moore.

- Empezando en la dirección de ingreso se busca en sentido horario hasta encontrar otro punto de objeto

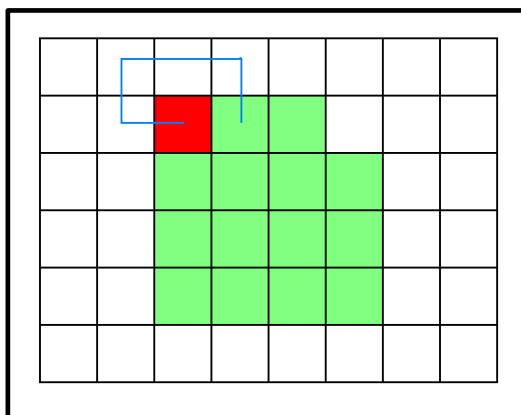


Figura 5.31. Búsqueda en sentido horario.

- Se repite el paso anterior buscando nuevos puntos de contorno.

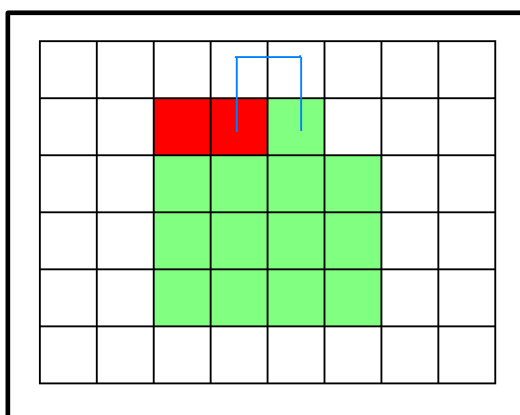


Figura 5.32. Búsqueda de un tercer punto.

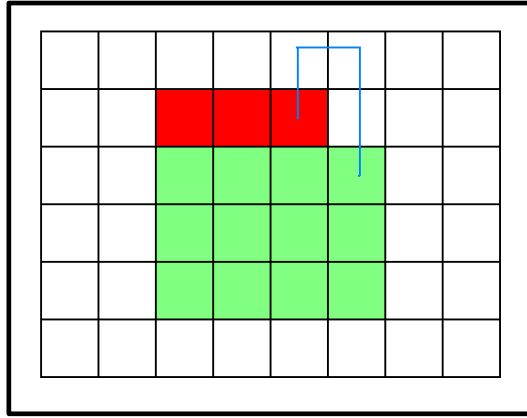


Figura 5.33. Búsqueda de un cuarto punto.

- El algoritmo termina cuando se ingresa nuevamente al pixel inicial en la misma dirección con que se ingresó originalmente.

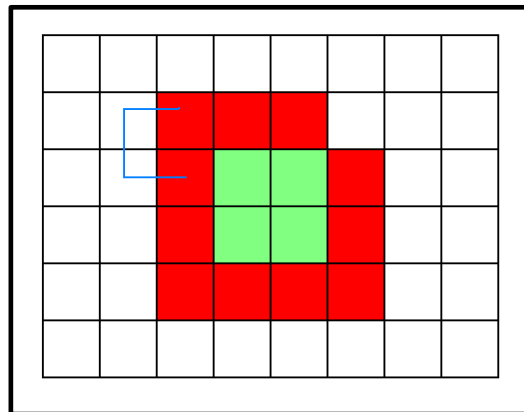


Figura 5.34. Fin del Algoritmo de Moore.

5.3.2.8 Extracción Reconocimiento de Contornos

Las características relevantes para las regiones del presente proyecto son el perímetro y el área.

El área se define simplemente como el número total de píxeles que componen la región en cuestión. Para el perímetro es necesario partir del código de 8-cadena $c'_R = \{c'_0, c'_1, \dots, c'_{M-1}\}$, y se define como:

$\text{Perímetro}(R) = \sum_{i=0}^{M-1} \text{longitud}(c'_i)$	ecuación 5.11
--	---------------

Con

$\text{longitud}(c'_i) = \begin{cases} 1, & \text{para } c = 0,2,4,6 \\ \sqrt{2}, & \text{para } c = 1,3,5,7 \end{cases}$	ecuación 5.12
---	---------------

La explicación de estos valores radica en que las direcciones pares son avances en sentido horizontal o vertical (de un pixel de longitud) y los impares son avances diagonales (donde $\sqrt{2}$ es la longitud de la diagonal del cuadrado de 1x1).

5.3.2.9 Extracción Reconocimiento de Contornos

Para la segmentación por Forma (diferenciar al producto de elementos extraños) se tomó como referencia la circularidad de una región definida como:

$\text{Circularidad}(R) = 4\pi \frac{A(R)}{[P(R)]^2}$	ecuación 5.13
---	---------------

Donde $A(R)$ es el área de la región y $P(R)$ es el perímetro. Este parámetro resulta en un máximo de 1 para una región perfectamente redonda y en el rango $[0,1)$ para otras formas.

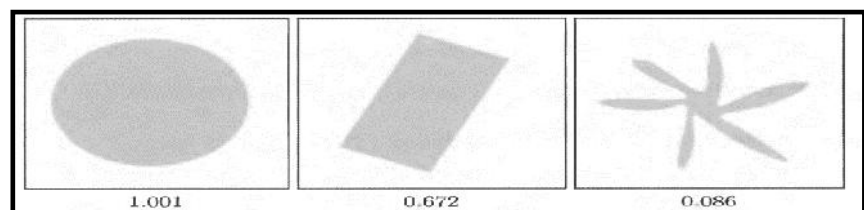


Figura 5.35. Diferentes formas con sus valores de circularidad.

La circularidad será cercana a la unidad para los productos deseados y mucho menor para objetos extraños.

5.3.2.10 Extracción Reconocimiento de Contornos

Luego de localizar los elementos extraños, se pasa a realizar segmentación por color de los productos en búsqueda de zonas que indiquen productos en mala calidad y, al mismo tiempo, desechando algún objeto extraño que por su circularidad haya podido superar la segmentación por forma.

Para este proceso se utiliza el modelo de color HSI y se analiza principalmente la componente H (tono) que es la que contiene la información de Color. En esta etapa se buscan valores de H alrededor del verde (cerca de 120°) y se desechan principalmente valores cercanos a tonos pardos (cerca de 30°).

5.3.3 Resultados

Todos los algoritmos fueron implementados de manera exitosa en el IDE C++ Builder. A continuación se muestra una impresión de pantalla del formulario final:

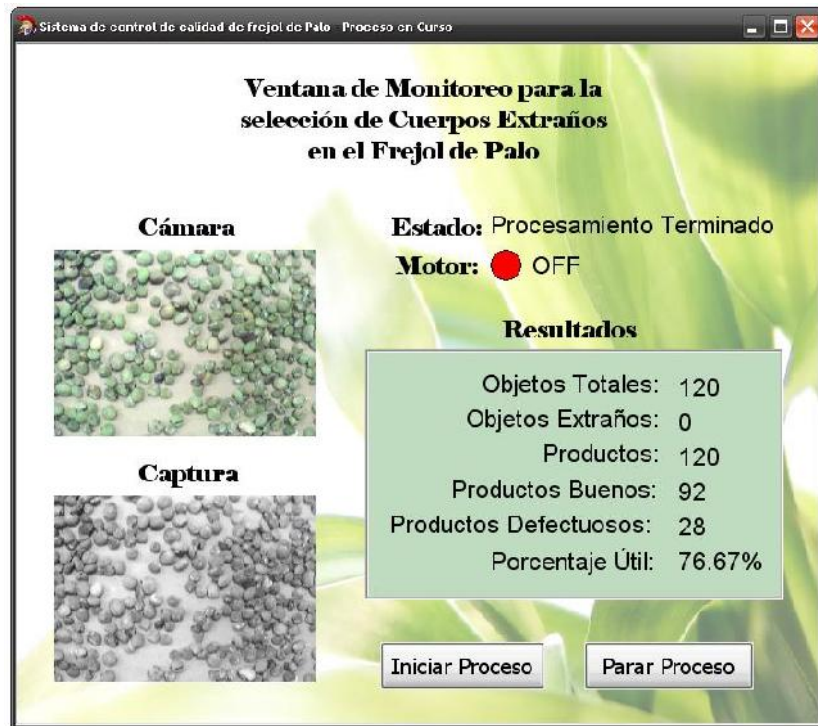


Figura 5.36. Ventana de monitoreo de la aplicación.

Se muestra a continuación el diagrama UML de las clases usadas en la implementación:

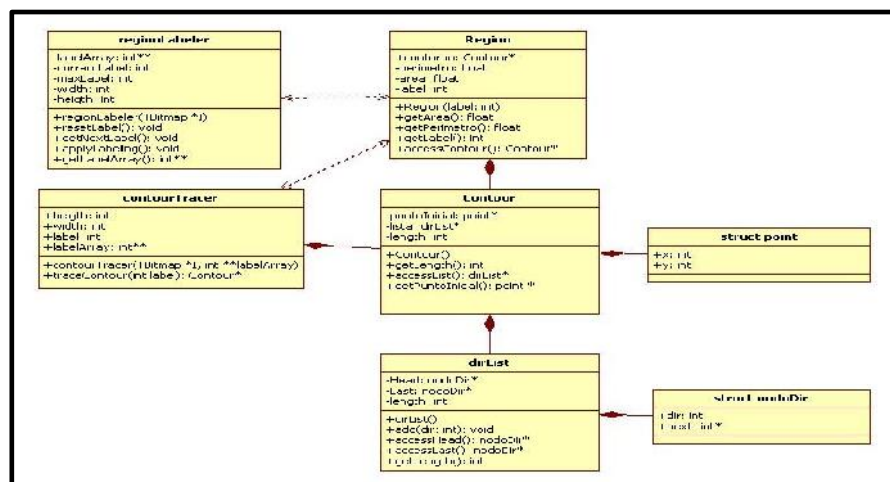


Figura 5.37. UML para el algoritmo de la aplicación.

- **Struct point:** es una estructura que representa un punto en el plano cartesiano.
- **Struct nodoDir:** es cada uno de los eslabones de un código de 8-cadena.
- **Clase dirList:** se utiliza para crear códigos de 8-cadena en la detección de contornos.
- **Clase Contour:** es la abstracción de cada contorno detectado el cual posee longitud, un código de 8-cadena y un punto inicial.
- **Clase contourTracer:** crea un objeto con los métodos necesarios para implementar el algoritmo de Trazado de Moore y retornar un objeto Contour por cada región.
- **Clase Región:** es la abstracción de cada región detectada, con su contorno (objeto Contour), su etiqueta (label), su perímetro y su área.
- **Clase regionLabeler:** crea un objeto con los métodos necesarios para realizar la detección de regiones mediante el algoritmo de relleno por difusión y detectar los elementos solapados.

CAPITULO 6

6. CONCLUSIONES Y RECOMENDACIONES

6.1 Conclusiones

- Se logró implementar exitosamente los algoritmos para la detección de cuerpos extraños de las leguminosas de grano seco.
- Se determinó que la mejor forma de identificar los objetos extraños es mediante la segmentación por forma evaluando la circularidad de las regiones halladas.
- Se determinó que el modelo de color HSI es el idóneo para la segmentación por color y la discriminación de los productos aptos.
- Se usó el lenguaje de programación C++, debido a la velocidad de procesamiento y por el uso nativo de memoria dinámica.
- El uso de los sensores infrarrojos se debió principalmente a su facilidad de uso, a su poco consumo de energía y a su disponibilidad comercial.

6.2 Recomendaciones

- Para lograr la captación correcta de la imagen y el funcionamiento adecuado del sensor infrarrojo es necesario el aislamiento del sistema de la luz natural, debido al contenido de luz infrarroja en la luz solar. Además, se necesita una iluminación cuya irradiación sea coherente en el espectro de longitud de onda.
- Para la implementación de un producto comercial final, se hace necesario el diseño de un sistema embebido específico con un dispositivo capaz de hacer procesamiento digital de señales, ya sea un DSP, FPGA, VIRTEX, etc.
- Si es necesaria la implementación del proyecto en un ambiente industrial muy ruidoso, es recomendable el uso de un enlace de fibra óptica en vez del cable serial normal, con su correspondiente conversión de protocolo serial a fibra óptica.

ANEXOS

Archivo: “clasecontornos.cpp”

```
#pragma hdrstop
#include "clasecontornos.h"
#include <Jpeg.hpp>
#pragma package(smart_init)
#define RAIZ2 1.4142

typedef struct punto{
    int x;
    int y;
} puntoXY;

class labelContornos{
private:
    int W;
    int H;
    int maxLabel;
    int **labelArray;
    int **pixelArray;
    float *perimArray;
    int **boxArray;

public:
    labelContornos(Graphics::TBitmap *I,int max,int **mb,float
    *perimetrosArray, int **cajasArray, int **final);
    int findNextPoint(puntoXY *pt,int dir,int label);
    void traceContour (int xs,int ys,int label);
    void findAllContours ();
    void expand(int **mb);
```

```

void getArrayFinal(int** contornosArray);
};

labelContornos::labelContornos(Graphics::TBitmap *I,int max,int
**mb,float *perimetrosArray, int **cajasArray, int **final){
    W= I->Width;
    H=I->Height;
    maxLabel=max;
    perimArray=perimetrosArray;
    boxArray=cajasArray;
    labelArray=(int**)malloc(((unsigned)(H+2))*(sizeof(int*)));
    for (int y=0; y < H+2; y++) {
        labelArray[y]=(int*)malloc(((unsigned)(W+2))*sizeof(int));
        for (int x=0; x < W+2; x++) {
            labelArray[y][x]=0;
        }
    }
    expand(mb);
    findAllContours();
    getArrayFinal(final);
}

void labelContornos::expand(int **mb){
    pixelArray=(int**)malloc(((unsigned)(H+2))*(sizeof(int*)));
    for (int y=0; y < H+2; y++) {
        pixelArray[y]=(int*)malloc(((unsigned)(W+2))*sizeof(int));
        for (int x=0; x < W+2; x++) {
            if(x==0 || x==W+1 || y==0 || y==H+1){
                pixelArray[y][x]=0;
            }else{
                pixelArray[y][x]=mb[y-1][x-1];
            }
        }
    }
}

```

```

    }
}
}

```

```

void labelContornos::getArrayFinal(int** contornosArray){
    for (int y=0; y < H+2; y++) {
        for (int x=0; x < W+2; x++) {
            if(!(x==0 || x==W+1 || y==0 || y==H+1)){
                contornosArray[y-1][x-1]=labelArray[y][x];
            }
        }
    }
}

```

```

int labelContornos::findNextPoint(puntoXY *pt,int dir,int label){
    int delta[8][2]={1,0},{1,1},{0,1},{-1,1},{-1,0},{-1,-1},{0,-1},{1,-1}};
    for (int i=0; i < 7; i++) {
        int x=pt->x + delta[dir][0];
        int y=pt->y + delta[dir][1];
        if (pixelArray[y][x]==0) {
            labelArray[y][x]=-1;
            dir=(dir+1)%8;
        }else{
            pt->x=x;
            pt->y=y;
            break;
        }
    }
    return dir;
}

```

```

void labelContornos::traceContour (int xs,int ys,int label){
    int xT,yT;
    int xP,yP;
    int xC,yC;
    puntoXY *pt=(puntoXY*)malloc(sizeof(puntoXY));
    pt->x=xs;
    pt->y=ys;
    int dNext=findNextPoint(pt,0,label);
    xP=xs; yP=ys;
    xC=pt->x;
    yC=pt->y;

    xT=pt->x;
    yT=pt->y;

    bool done=(xs==xT && ys==yT);

    while ((!done)){
        labelArray[yC][xC]=label;
        if ((dNext%2)==0) {
            perimArray[label-2]+=1;
        }else{
            perimArray[label-2]+=RAIZ2;
        }

        pt->x=xC;
        pt->y=yC;
        int dSearch=(dNext+6)%8;
        dNext=findNextPoint(pt,dSearch,label);
        xP=xC;
        yP=yC;
        xC=pt->x;

```

```

        yC=pt->y;
        done=(xP==xs && yP==ys && xC==xT && yC==yT);
        if(!done){
            }
        }
    }

void labelContornos::findAllContours (){
    for (int i=2; i <= maxLabel; i++) {
    {
        for (int y=1; y < H+1; y++) {
            for (int x=1; x < W+1; x++) {
                if (pixelArray[y][x]==i) {
                    traceContour(x,y,i);
                    labelArray[y][x]=i;
                    goto eti;
                }
            }
        }
        eti:
    }
}

```

Archivo: dip.cpp

```
#pragma hdrstop
#include <stdio.h>
#include <math.h>
#include "dip.h"
#include <Jpeg.hpp>
#include <Clipbrd.hpp>
//-----

#pragma package(smart_init)
#define NULL 0
#define MIN(A,B) ((A) < (B) ? (A) : (B))
#define MAX(A,B) ((A) > (B) ? (A) : (B))
//float porcentaje;
//bool resultado;

void procesamiento1 ( Graphics::TBitmap *I,float *porcentaje,bool
*resultado);
void rgb2gray(Graphics::TBitmap *I);
void pasabajo(Graphics::TBitmap *I);
void binarizacion(Graphics::TBitmap *I);
void mascara(Graphics::TBitmap *I,int mask[3][3], int divisor);
void sobel1(Graphics::TBitmap *I);
void sobel2(Graphics::TBitmap *I);
void sobeltotal(Graphics::TBitmap *I, float ***matrizhsi);
void deteccion( Graphics::TBitmap *I, float ***matrizhsi);
int etiquetado( Graphics::TBitmap *I, float ***matrizhsi);
void llenado( Graphics::TBitmap *I);
float porcentaje1( Graphics::TBitmap *I, int azules);
//float *rgb2hsi( Graphics::TBitmap *I );
```



```

void rgb2gray( Graphics::TBitmap *I ){
    int x,y;

    RGBQUAD *pixelactual;
    RGBQUAD *pixeltemporal;
    pixelactual=NULL;
    pixeltemporal=NULL;
    int H = I->Height;
    int W = I->Width;
    int Bytes = W*4;
    char prom;

    void *buffer=malloc (Bytes);

    for (y=0; y<H; y++)
    {
        pixelactual = (RGBQUAD *)I->ScanLine[y];
        pixeltemporal = (RGBQUAD *)buffer;
        for (x=0; x<W; x++,pixelactual++,pixeltemporal++)
        {
            prom=(pixelactual->rgbRed+pixelactual->rgbBlue+pixelactual-
>rgbGreen)/3;

            pixeltemporal->rgbRed=prom;
            pixeltemporal->rgbBlue=prom;
            pixeltemporal->rgbGreen=prom;
        }
        Move (buffer, I->ScanLine[y], Bytes);
    }

    free (buffer);
}

```

```
//-----
void pasabajo( Graphics::TBitmap *I ){
    int d=9;
    int m[3][3]={{1,1,1},{1,1,1},{1,1,1}};
    mascara(I,m,d);
}

//-----

void binarizacion( Graphics::TBitmap *I ){
    int x,y;
    rgb2gray(I);
    RGBQUAD *pixelactual;
    RGBQUAD *pixeltemporal;
    pixelactual=NULL;
    pixeltemporal=NULL;
    int H = I->Height;
    int W = I->Width;
    int Bytes = W*4;
    char prom;

    void *buffer=malloc (Bytes);
    I->PixelFormat = pf32bit;    /* make sure the bitmap can be accessed
as RGBA */
    for (y=0; y<H; y++)
    {
        pixelactual = (RGBQUAD *)I->ScanLine[y];
        pixeltemporal = (RGBQUAD *)buffer;
        for (x=0; x<W; x++,pixelactual++,pixeltemporal++)
        {
            if (pixelactual->rgbBlue<55) {
                pixeltemporal->rgbRed=0;
            }
        }
    }
}
```

```

        pixeltemporal->rgbGreen=0;
        pixeltemporal->rgbBlue=0;
    }else{
        pixeltemporal->rgbRed=255;
        pixeltemporal->rgbGreen=255;
        pixeltemporal->rgbBlue=255;
    }
}
Move (buffer, l->ScanLine[y], Bytes);
}
free (buffer);
}

```

//-----

//Sobel Horizontal

```

void sobel1(Graphics::TBitmap *I){
    int d=1;
    int m[3][3]={{-1,-2,-1},{0,0,0},{1,2,1}};
    mascara(I,m,d);
}

```

//-----

//Sobel Vertical

```

void sobel2(Graphics::TBitmap *I){
    int d=1;
    int m[3][3]={{-1,0,1},{-2,0,2},{-1,0,1}};
    mascara(I,m,d);
}

```

//-----

//Sobel Final

```

void sobeltotal( Graphics::TBitmap *I){
    int x,y;
    rgb2gray(I);
    Graphics::TBitmap *I2 = new Graphics::TBitmap();
    I2->Assign (I);
    sobel1(I);
    sobel2(I2);
    //show waiting cursor, julio
    RGBQUAD *pixelactual;
    RGBQUAD *pixelactual2;
    RGBQUAD *pixeltemporal;
    pixelactual=NULL;
    pixeltemporal=NULL;
    int H = I->Height;
    int W = I->Width;
    int Bytes = W*4;
    void *buffer=malloc (Bytes);

    for (y=0; y<H; y++)
    {
        pixelactual = (RGBQUAD *)I->ScanLine[y];
        pixelactual2 = (RGBQUAD *)I2->ScanLine[y];
        pixeltemporal = (RGBQUAD *)buffer;
        for (x=0; x<W; x++,pixelactual++,pixelactual2++,pixeltemporal++)
        {
            //pixeltemporal->rgbRed=sqrt(pow(pixelactual-
            >rgbRed,2)+pow(pixelactual2->rgbRed,2));
            //pixeltemporal->rgbGreen=sqrt(pow(pixelactual-
            >rgbGreen,2)+pow(pixelactual2->rgbGreen,2));
            //pixeltemporal->rgbBlue=sqrt(pow(pixelactual-
            >rgbBlue,2)+pow(pixelactual2->rgbBlue,2));
        }
    }
}

```

```

        pixeltemporal->rgbRed=abs(pixelactual-
>rgbRed)+abs(pixelactual2->rgbRed);
        pixeltemporal->rgbBlue=abs(pixelactual-
>rgbBlue)+abs(pixelactual2->rgbBlue);
        pixeltemporal->rgbGreen=abs(pixelactual-
>rgbGreen)+abs(pixelactual2->rgbGreen);
    }
    Move (buffer, l->ScanLine[y], Bytes);
}

free (buffer);
delete l2;
}

//-----

void mascara(Graphics::TBitmap *l,int mask[3][3],int divisor){
int x,y,result;
    RGBQUAD *A11, *A12, *A13, /* input */
        *A21, *A22, *A23,
        *A31, *A32, *A33;
    RGBQUAD *O;          /* output */
    int H = l->Height;
    int W = l->Width;
    int Bytes_per_Row = W*4;
    RGBQUAD **ImageBuffer = (RGBQUAD **)malloc ((unsigned)H *
sizeof(RGBQUAD *));

    l->PixelFormat = pf32bit; /* make sure the bitmap can be accessed
as RGBA */
    for (y=0; y<H; y++)
        ImageBuffer[y] = (RGBQUAD *)malloc (Bytes_per_Row);

```

```

for (y=1; y<H-1; y++)
{
    A11 = (RGBQUAD *)I->ScanLine[y-1];    A12 = A11+1;    A13 =
A12+1;
    A21 = (RGBQUAD *)I->ScanLine[y];    A22 = A21+1;    A23 = A22+1;
    A31 = (RGBQUAD *)I->ScanLine[y+1];    A32 = A31+1;    A33 =
A32+1;
    O = ImageBuffer[y];
    O->rgbRed=A21->rgbRed;    O->rgbGreen=A21->rgbGreen;    O-
>rgbBlue=A21->rgbBlue;
    O++; /*advance to center */
    for (x=1; x<W-1;
x++,O++,A11++,A12++,A13++,A21++,A22++,A23++,A31++,A32++,A33+
+)
    {
        result = ((mask[0][0]*A11->rgbRed+mask[0][1]*A12-
>rgbRed+mask[0][2]*A13->rgbRed+mask[1][0]*A21-
>rgbRed+mask[1][1]*A22->rgbRed+mask[1][2]*A23-
>rgbRed+mask[2][0]*A31->rgbRed+mask[2][1]*A32-
>rgbRed+mask[2][2]*A33->rgbRed)/divisor);
        result = MIN(255,MAX(result,0));
        O->rgbRed = result;
        result = ((mask[0][0]*A11->rgbGreen+mask[0][1]*A12-
>rgbGreen+mask[0][2]*A13->rgbGreen+mask[1][0]*A21-
>rgbGreen+mask[1][1]*A22->rgbGreen+mask[1][2]*A23-
>rgbGreen+mask[2][0]*A31->rgbGreen+mask[2][1]*A32-
>rgbGreen+mask[2][2]*A33->rgbGreen)/divisor);
        result = MIN(255,MAX(result,0));
        O->rgbGreen = result;

        result = ((mask[0][0]*A11->rgbBlue+mask[0][1]*A12-

```

```

>rgbBlue+mask[0][2]*A13->rgbBlue+mask[1][0]*A21-
>rgbBlue+mask[1][1]*A22->rgbBlue+mask[1][2]*A23-
>rgbBlue+mask[2][0]*A31->rgbBlue+mask[2][1]*A32-
>rgbBlue+mask[2][2]*A33->rgbBlue)/divisor);
        result = MIN(255,MAX(result,0));
        O->rgbBlue = result;
    }

    O->rgbRed=A23->rgbRed; O->rgbGreen=A23->rgbGreen; O-
>rgbBlue=A23->rgbBlue;
}

/* transfer results */
for (y=1; y<H-1; y++)
    Move (ImageBuffer[y], l->ScanLine[y], Bytes_per_Row);

for (y=0; y<H; y++)
    free(ImageBuffer[y]);

free (ImageBuffer);
}

//-----

//Deteccion

void llenado( Graphics::TBitmap *I)
{
    int xder=-1;
    int xizq=0;
    int H = I->Height;
    int W = I->Width;
    int x,y,z;

```

```

    RGBQUAD *pixelactual;
    pixelactual=NULL;
    for (y=0; y < H; y++)
    { pixelactual = (RGBQUAD*)I->ScanLine[y];
      for (x=0; x < W; x++)
      {
        if (pixelactual->rgbRed==255)
        {
          xizq=x;
          break;
        }
      }
      for (x=W-1; x >=0; x--)
      {
        if (pixelactual->rgbRed==255)
        {
          xder=x;
          break;
        }
      }
      for (x=xizq; x <= xder; x++)
      {
        pixelactual->rgbRed=255;
        pixelactual->rgbGreen=255;
        pixelactual->rgbBlue=255;

      }
      xizq=0;
      xder=-1;
    }
  }

```


/////////////////////////////////HSI////////////////////////////////

```
void rgb2hsi( Graphics::TBitmap *I, float ***matrizhsi)
{
    RGBQUAD *pixelactual;
    RGBQUAD *pixeltemporal;
    pixelactual=NULL;
    pixeltemporal=NULL;
    int H = I->Height;
    int W = I->Width;
    int x,y,z;
    for (y=0; y<H; y++){
        pixelactual = (RGBQUAD*)I->ScanLine[y];
        for (x=0; x<W; x++,pixelactual++){
            if(pixelactual->rgbRed==pixelactual->rgbBlue && pixelactual-
>rgbRed==pixelactual->rgbGreen){
                matrizhsi[0][y][x]=0;
                matrizhsi[1][y][x]=0;
                matrizhsi[2][y][x]=(float)pixelactual->rgbRed;
            }else{
                float R=(float)pixelactual->rgbRed;
                float G=(float)pixelactual->rgbGreen;
                float B=(float)pixelactual->rgbBlue;
                float r=R/(R+G+B);
                float g=G/(R+G+B);
                float b=B/(R+G+B);
                matrizhsi[0][y][x]=      acos((r-(0.5)*(g+b))/(sqrt(pow((r-
g),2)+(r-b)*(g-b)))));
                if (b>g) {
                    matrizhsi[0][y][x]=2*3.14159-matrizhsi[0][y][x];
                }
            }
        }
    }
}
```

```

        matrizhsi[0][y][x]=matrizhsi[0][y][x]*(180.00/3.14159);
        matrizhsi[1][y][x]=1-3*(MIN(r,MIN(g,b)))/(r+g+b);
        matrizhsi[2][y][x]=(R+G+B)/(765);
    }
    }
    }
}

```

```

//-----

```

```

//Deteccion

```

```

void deteccion( Graphics::TBitmap *I, float ***matrizhsi)
{
    int x,y,z;
    RGBQUAD *pixelactual;
    pixelactual=NULL;
    int H = I->Height;
    int W = I->Width;
    int A[3][240][320];
    int R,G,B;
    for(y=0;y<H;y++)
    {
        pixelactual = (RGBQUAD *)I->ScanLine[y];
        for(x=0;x<W;x++)
        {
            R=pixelactual->rgbRed;
            A[0][y][x]=R;
            G=pixelactual->rgbGreen;
            A[1][y][x]=G;
            B=pixelactual->rgbBlue;
            A[2][y][x]=B;
        }
    }
    for (y=0; y<H; y++)

```

```

    { for (x=0; x<W; x++)
        { matrizhsi[0][y][x]=matrizhsi[1][y][x]=matrizhsi[2][y][x]=0;
            if(A[0][y][x]==255&&A[1][y][x]==255&&A[2][y][x]==255)
                break;
        }
    }
    for (y=H-1; y>=0; y--)
    { for (x=W-1; x>=0; x--,pixelactual--)
        { matrizhsi[0][y][x]=matrizhsi[1][y][x]=matrizhsi[2][y][x]=0;
            if(A[0][y][x]==255&&A[1][y][x]==255&&A[2][y][x]==255)
                break;
        }
    }
}

//-----

//Etiquetado

int etiquetado( Graphics::TBitmap *I, float ***matrizhsi)
{ int x,y;
  int H = I->Height;
  int W = I->Width;
  int azules3=0;
  RGBQUAD *pixeltemporal;
  pixeltemporal=NULL;
  for (y=0; y<H; y++)
  {
    pixeltemporal = (RGBQUAD *)I->ScanLine[y];
    for (x=0; x<W; x++,pixeltemporal++)
    {
      if(matrizhsi[0][y][x]>2.5 && matrizhsi[0][y][x]<40)

```

```

        {
            pixeltemporal->rgbRed=0;
            pixeltemporal->rgbGreen=0;
            pixeltemporal->rgbBlue=255;
            azules3=azules3+1;
        }

    }
}
return azules3;
}
//-----

//Etiquetado
float porcentaje1( Graphics::TBitmap *I, int azules)
{
    int x,y,blancos=0;
    int H = I->Height;
    int W = I->Width;
    RGBQUAD *pixeltemporal;
    pixeltemporal=NULL;
    for (y=0; y<H; y++)
    {
        pixeltemporal = (RGBQUAD *)I->ScanLine[y];
        for (x=0; x<W; x++,pixeltemporal++)
            if(pixeltemporal->rgbRed==255)
                blancos=blancos+1;
    }
    float coc=100.00*((float)azules/(float)blancos);
    //float p=((float)(floor(100*coc)))/100.00;

    return coc;
}

```

```
}
```

```
//-----
```

```
void procesamiento1 ( Graphics::TBitmap *I,float *porcentaje,bool  
*resultado)
```

```
{
```

```
int H = I->Height;
```

```
int W = I->Width;
```

```
float p;
```

```
int x,y,z,azules2=0;
```

```
float ***matrizhsi=(float***)malloc((unsigned)3*sizeof(float**));
```

```
for (z=0; z < 3; z++)
```

```
{ matrizhsi[z]=(float**)malloc((unsigned)H*sizeof(float*));
```

```
for (y=0; y < H; y++)
```

```
matrizhsi[z][y]=(float*)malloc((unsigned)W*sizeof(float));
```

```
}
```

```
////////////////////////////////
```

```
Graphics::TBitmap *I2 = new Graphics::TBitmap();
```

```
I2->Assign( I );
```

```
rgb2gray( I2 );
```

```
pasabajo( I2 );
```

```
binarizacion( I2 );
```

```
llenado( I2 );
```

```
rgb2hsi( I, matrizhsi);
```

```
azules2=etiquetado( I , matrizhsi);
```

```
p=porcentaje1(I2,azules2);
```

```
(*porcentaje)=p;
```

```
if(p>15.00)
```

```

    { *resultado=false;}
    //////////////////////////////////
    for (int z=0; z < 3; z++)
        for (int y=0; y < H; y++)
            free(matrizhsi[z][y]);

    for (int z=0; z < 3; z++) {
        free(matrizhsi[z]);
    }
    free(matrizhsi);

    delete l2;
}

```

Archivo: "regiones.cpp"

```
#pragma hdrstop
#include <Jpeg.hpp>
#include "regiones.h"
#include "queue.cpp"
//-----

#pragma package(smart_init)

class labelRegiones{
private:
    int **labelArray;
    int width;
    int height;
    int currentLabel;
    int maxLabel;
    colaPuntos *cola;
    Graphics::TBitmap *I;

public:
    labelRegiones(Graphics::TBitmap *Im,int **arreglo,int conex);
    void makeLabelArray();
    void aplicarLabel(int conex);
    void resetLabel();
    int getNextLabel();
    void setLabel(int u, int v, int label);
    int getMaxLabel();
    colaPuntos * getCola();
    void floodFill(int x, int y, int label);
    void floodFill2(int x, int y, int label);
```

```
};
```

```
labelRegiones::labelRegiones(Graphics::TBitmap *Im,int **arreglo,int
conex){
    l=Im;
    width = Im->Width;
    height = Im->Height;

    labelArray =(int**)malloc((unsigned)height*sizeof(int*));
    for (int j = 0; j < height; j++) {
        labelArray[j]=(int*)malloc((unsigned)width*sizeof(int));
    }
    for (int i = 0; i < width; i++) {
        for (int j = 0; j < height; j++) {
            labelArray[j][i]=arreglo[j][i];
        }
    }
    aplicarLabel(conex);
    for (int i = 0; i < width; i++) {
        for (int j = 0; j < height; j++) {
            arreglo[j][i]=labelArray[j][i];
        }
    }
}
```

```
void labelRegiones::makeLabelArray(){
    RGBQUAD *pixelactual;

    for (int j = 0; j < height; j++) {
        pixelactual = (RGBQUAD *)l->ScanLine[j];
        for (int i = 0; i < width; i++,pixelactual++) {
            if (pixelactual->rgbRed==0){
```



```

        labelArray[j][i] = 0;
    }else{
        labelArray[j][i] = 1;
    }
    }
}
}

void labelRegiones::aplicarLabel(int conex){
    resetLabel();
    for (int y = 0; y < height; y++) {
        for (int x = 0; x < width; x++) {
            if (labelArray[y][x]==1) {
                int label = getNextLabel();
                if (conex==4) {
                    floodFill(x, y, label);
                }else{
                    floodFill2(x, y, label);
                }
            }
        }
    }
}

void labelRegiones::resetLabel() {
    currentLabel = -1;
    maxLabel = -1;
}

int labelRegiones::getNextLabel() {

```

```

    if (currentLabel < 1){
        currentLabel = 2;
    }else{
        currentLabel = currentLabel + 1;
    }
    maxLabel = currentLabel;
    return currentLabel;
}

void labelRegiones::setLabel(int u, int v, int label) {
    if (u >= 0 && u < width && v >= 0 && v < height){
        labelArray[v][u] = label;
    }
}

void labelRegiones::floodFill(int x, int y, int label){
    cola=new colaPuntos();
    cola->insertar(x,y);

    while (!(cola->estaVacia())) {
        Punto *n;
        n=cola->sacar();
        int u=n->x;
        int v=n->y;
        if ((u>=0) && (u<width) && (v>=0) && (v<height) &&
labelArray[v][u]==1) {
            setLabel(u, v, label);
            cola->insertar(u+1,v);
            cola->insertar(u,v+1);
            cola->insertar(u,v-1);
            cola->insertar(u-1,v);
        }
    }
}

```

```

    }
}

void labelRegiones::floodFill2(int x, int y, int label){
    cola=new colaPuntos();
    cola->insertar(x,y);
    while (!(cola->estaVacia())) {
        Punto *n;
        n=cola->sacar();
        int u=n->x;
        int v=n->y;
        if ((u>=0) && (u<width) && (v>=0) && (v<height) &&
labelArray[v][u]==1) {
            setLabel(u, v, label);
            cola->insertar(u+1,v);
            cola->insertar(u,v+1);
            cola->insertar(u,v-1);
            cola->insertar(u-1,v);
            cola->insertar(u+1,v+1);
            cola->insertar(u-1,v-1);
            cola->insertar(u+1,v-1);
            cola->insertar(u-1,v+1);
        }
    }
}

int labelRegiones::getMaxLabel(){
    return maxLabel;
}

colaPuntos * labelRegiones::getCola(){
    return cola;
}

```

BIBLIOGRAFIA

- Amorós Eduardo, Díaz Diógenes y León Carlos. **Toma de Decisiones para Negocios: Casos Prácticos.** USAT.
- Gonzalez Rafael y Richard Woods. **Digital Image Processing.** Second Edition.
- Castleman Kenneth. **Digital Image Processing.**
- Joyanes Luis e Ignacio Zahonero. **Programacion en C.**
- Gonzalez Rafael. **Digital Image Processing using MATLAB**
- Burger Wilhelm y Mark Burge. **Digital Image Processing. An Algorithmic Introduction Using JAVA**
- Bingham Liu, Xiuduan Fang, Weizhi Wang y Zheng Zhiyong. **Automatic Separation of Overlapping Objects**
- <http://wiki.sumaqperu.com/es/Frijol>
- http://www4.ujaen.es/~satorres/practicas/practica3_vc.pdf
- http://www.imageprocessingplace.com/downloads_V3/root_downloads/tutorials/contour_tracing_Abeer_George_Ghuneim/moore.html