
# UNIVERSIDAD
# LASALLISTA BENAVENTE

ESCUELA DE INGENIERÍA EN COMPUTACIÓN
CON ESTUDIOS INCORPORADOS A LA
**UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO**
CLAVE: 8793-16


# "PROCESAMIENTO Y FILTRADO DE IMÁGENES EN LOS SISTEMAS DE VISIÓN ARTIFICIAL"

# TESIS

QUE PARA OBTENER EL TITULO DE:
## INGENIERO EN COMPUTACIÓN

PRESENTA:
## JESÚS AVILA ESTRADA

ASESOR:
**ING. CARLOS A. HERNÁNDEZ VILLANUEVA**

CELAYA, GTO. DICIEMBRE DEL 2007

Son tantas personas las que me han apoyado, contribuido y motivado en la conclusión de este trabajo, y que a su vez son parte importante en mi formación tanto profesional como humana, por ello agradezco a cada una de ellas, especialmente:

A Dios por darme la vida, una familia, por permitirme concluir mis estudios y la oportunidad de desarrollarme en todos los ámbitos.

A mis padres Donaciano Ávila Sánchez y Celia Estrada López, por darme su amor, por inculcarme valores y apoyarme incondicionalmente sobre todo en los momentos más difíciles.

A mis hermanos Rafael, Angélica, Laura, Claudia y Karolaine, por ser para mí un ejemplo de superación y por todo su amor y apoyo.

A mi novia Claudia Jiménez Juárez por su amor, paciencia y por apoyarme en todos los momentos difíciles.

A todos mis maestros, por compartir su tiempo, conocimientos y experiencias con el afán de hacer de mí un verdadero profesionista.

A mi asesor Ing. Carlos A. Hernández Villanueva, por ayudarme con sus conocimientos a poder encausar este proyecto y dar conclusión al mismo.

A todos mis compañeros y amigos, por compartir todas y cada una de las experiencias y conocimientos durante nuestra estancia en las aulas.

# ÍNDICE

pág.

## CAPÍTULO TERCERO

### SISTEMAS DE VISIÓN ARTIFICIAL



## CAPÍTULO IV

### DISEÑO Y CONSTRUCCIÓN DE UN SVA BÁSICO DE SEGUNDO NIVEL



### CONCLUSIONES

### BIBLIOGRAFÍA

# INTRODUCCIÓN

La visión artificial es una técnica basada en la adquisición de imágenes, generalmente en dos dimensiones, para luego procesarlas digitalmente mediante algún tipo de CPU (computadora, microcontrolador, DSP, etc.), con el fin de extraer y medir determinadas propiedades adquiridas de la imágenes. Se trata por tanto, de una tecnología que combina las computadoras con las cámaras de video para adquirir, analizar e interpretar imágenes de una forma equivalente a la inspección visual humana.

Actualmente se aplica en diversos procesos científicos y militares, extendiéndose su uso además, en un amplio rango de sectores industriales como por ejemplo para la automatización de tareas anteriormente reservadas para la inspección visual humana. La gran demanda de esta tecnología en el sector industrial es debido a que se trata de una tecnología especialmente útil en labores de inspección o supervisión, siendo los sistemas de visión artificial cuantitativamente más objetivos y consistentes que la inspección humana.

Un segundo aspecto complementario de estas técnicas es la automatización derivada de esta inspección y la posibilidad de actuar sobre el proceso de fabricación o inspección modificando parámetros de la máquina de producción.

Las técnicas de visión artificial, como demuestra su gran uso en la industria, son particularmente apropiadas para la realización de trabajos visuales altamente repetitivos que sean fatigosos o difíciles de realizar para un operario, especialmente cuando este trabajo es ineficiente o costoso en términos económicos o de tiempos.

Hasta hace poco tiempo los sistemas encargados de automatizar los procesos dentro de la industria, eran en su mayoría dirigidos o gobernados por el uso de sensores, esto hacía que las maquinas fueran un tanto limitadas, pues no podían ser autónomas, es decir tener la capacidad de tomar decisiones por si mismas para resolver un problema, sin la intervención de un operador.

Esta fue una de las principales necesidades que dieron origen a la idea de realizar sistemas capaces de llevar acabo tareas en las que no se requiriera la presencia de un operador, y además que dichos sistemas tuvieran la capacidad de poder dar múltiples soluciones a los problemas y modificar las respuestas con solo alterar un algoritmo de comportamiento.

Debido a que la información visual es una de las principales fuentes de datos del mundo real, resulta útil el proveer a una computadora del sentido de la vista (a partir de imágenes tomadas con cámaras digitales o analógicas), que junto con otros mecanismos como el aprendizaje, hagan de esta una herramienta capaz de detectar y ubicar objetos en el mundo real, el cual es el objetivo principal de la Visión por Computadora.

La visión por computadora incluye muchos campos de desarrollo, pero en general es un área que usa las técnicas para estimar los detalles en imágenes, relacionar las mediciones de dichos detalles a la geometría de objetos en el espacio e interpretar la información geometría.

Una parte importante de la visión por computadora o también conocida como la Visión Artificial, es el procesamiento de imágenes, que se refiere a las técnicas para transformar las imágenes para obtener información necesaria.

La hipótesis que se plantea en esta tesis, comprende la idea de que es más sencilla la obtención de información de los objetos en el entorno de un Sistema (Robot o máquina), empleando el procesamiento y filtrado de las imágenes captadas por una cámara, que si solamente se utilizaran sensores (movimiento, calor, proximidad, etc.).

Para ello, en el Capítulo Primero, se empezarán a estudiar los conceptos básicos del procesamiento y filtrado de imágenes, los diferentes modelos de color y términos que respectan a la visión computacional, así como el estudio de las diferentes herramientas, en el que se da una breve descripción de las ventajas y desventajas de cada una de ellas. También se dan ejemplos de las diferentes áreas de aplicación de la Visión Artificial.

En lo que se refiere al Capítulo Segundo, se estudiarán a fondo las herramientas con las que cuenta Matlab (Image Processing Toolbox e Image Acquisition Toolbox), así como ejemplos de las principales funciones para realizar tareas en el manejo de imágenes y la captura de imágenes a través de un dispositivo de entrada.

En el Tercer Capítulo se estudiarán a fondo los Sistemas de Visión Artificial en lo que se refiere a los elementos que los integran y las diferentes etapas que los constituyen. Y para concluir con la investigación en el Cuarto Capítulo se describe un SVA de segundo nivel o también conocido como de nivel medio, en su parte teórica y física y algunos ejemplos de las diferentes aplicaciones que se les pueden dar a los principios en los que se basa dicho sistema, esto con la finalidad de comprobar lo planteado en la hipótesis.

# CAPÍTULO PRIMERO

# INTRODUCCIÓN AL PROCESAMIENTO Y FILTRADO DE IMÁGENES.

## 1. 1. CONCEPTOS BÁSICOS

### Definición de imagen

Una imagen puede ser definida como una función bidimensional *f(x,y)*, donde *x* e *y* son coordenadas espaciales, y la amplitud de *f* en cualquier par de coordenadas se denomina intensidad o nivel de gris de la imagen en el punto, cuando *x y* y los valores de la amplitud de *f* son todos finitos, cantidades discretas.

### Procesamiento de imágenes

El procesamiento de imágenes es el mejoramiento de la calidad de las imágenes para su posterior interpretación o utilización. En el procesamiento de imágenes interviene el uso de la computadora.

Una imagen digital está compuesta de un número finito de elementos, cada uno de ellos teniendo una particular localización y valor. Esos elementos se llaman elementos pictóricos, elementos unidad de imagen o más comúnmente llamados píxeles.

**Píxel:** Es la unidad más pequeña en la representación de una imagen.

**Visión Computacional**

La visión Computacional es el estudio de los procesamientos de imágenes, para entenderlas y construir máquinas con capacidades similares a las humanas.

Su función principal es localizar y reconocer el ambiente mediante el procesamiento de las imágenes.

El objetivo de la Visión Computacional es extraer ciertas características de una imagen para su descripción e interpretación mediante la computadora.

## 1. 2. LOS MODELOS DE COLOR

Un modelo de color se refiere al esquema basado en colores ya sean primarios o secundarios y de estos se obtienen las diferentes gamas. La mayoría de los modelos de color que se usan hoy se orientan hacia hardware como monitores o impresoras o simplemente aplicaciones de manipulación de color. El modelo orientado a hardware más común es el RGB (rojo-verde-azul), el modelo CMY (cian-magenta-amarillo) para impresoras a color y el YIQ que es el Standard para la difusión de TV, en este caso la Y corresponde a la luminosidad y la I y Q son dos componentes cromáticas.

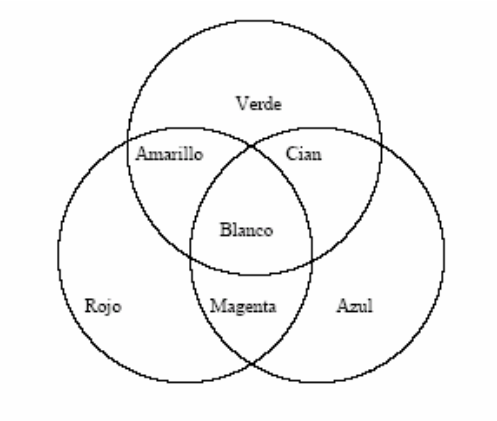



Fig. 1.1 Suma de colores primarios.[1]

Para la manipulación de color se usan normalmente los modelos HSI (matriz, saturación e intensidad) y HSV (matriz, saturación, valor).

### 1. 2. 1 Modelo RGB

En el modelo RGB cada color aparece en sus componentes primarias espectrales rojo, verde y azul. Este sistema esta basado en coordenadas cartesianas.

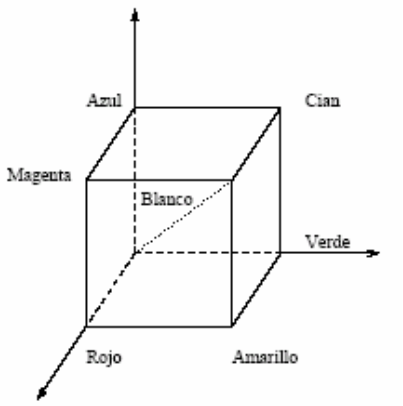



Fig.1.2 Cubo de color RGB.[2]

[1] http://www.diinf.usach.cl/~dmery/imagenes/proc.htm
[2] http://www.diinf.usach.cl/~dmery/imagenes/proc.htm

Las imágenes en el modelo RGB están formadas por tres planos de imágenes independientes, cada una de los colores primarios.

Por ejemplo los monitores RGB al momento de proyectar una imagen, se combinan en la pantalla de fósforo para producir una imagen de color compuesta. Por tanto el uso del modelo RGB esta orientado al procesamiento de imágenes que están expresadas en términos de los tres planos de colores. La mayoría de las cámaras para adquirir imágenes a color utilizan el formato RGB. En la figura 1.3 se puede ver una imagen descompuesta en los tres planos RGB

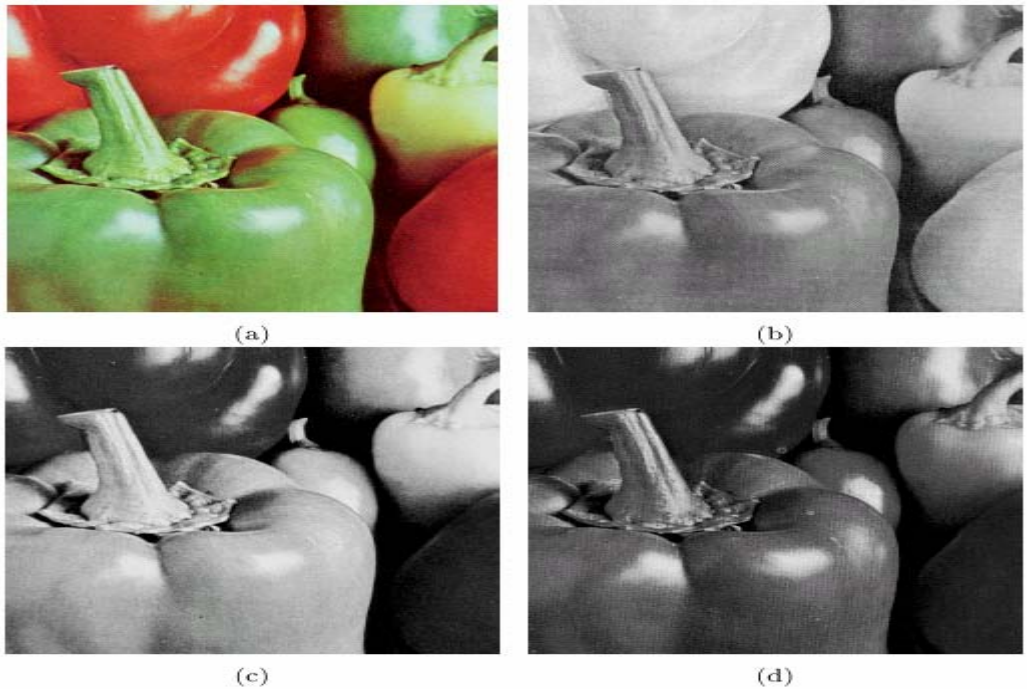



Fig. 1.3 (a) Imagen Original, (b) Plano R, (c) Plano G, (d) Plano B .[3]

[3] http://www-dbv.informatik.uni-bonn.de/abstracts/will.icip00.htm

### 1. 2. 2 Otros Modelos

#### Modelo CMY

Este modelo a diferencia del anterior, esta basado en los colores secundarios Cyan, Magenta y Amarillo. Este formato es utilizado principalmente por las impresoras a color.

#### Modelo YUV

Este formato se utiliza para los Standard de TV NTSC, PAL y SECAM. La Y representa la componente de Blanco y Negro, y la información de color de U y V se combinan para dar como resultado una imagen a color.

#### Modelo YIQ

El modelo YIQ se usa en las emisiones de TV a color. Básicamente YIQ es una recodificación de RGB para la eficiencia en la transmisión y así tener compatibilidad con los standares de la TV monocromo. La componente Y del sistema YIQ proporciona toda la información de video que se requiere para una TV monocromo.

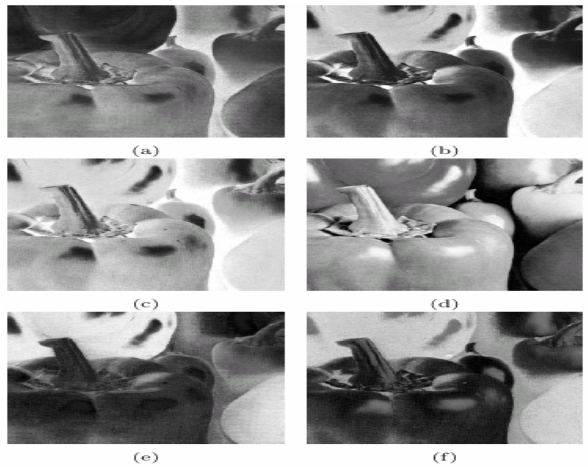



Fig. 1.4 (a) Plano C del modelo CMY, (b) Plano M del modelo CMY, (c) Plano Y del modelo CMY, (d) Plano Y del modelo YIQ, (e) Plano I del modelo YIQ, (f) Plano Q del Modelo YIQ.[4]

[4] http://www-dbv.informatik.uni-bonn.de/abstracts/will.icip00.htm

## 1. 3 PASOS PARA EL PROCESAMIENTO DE IMÁGENES

La visión artificial lleva asociada una enorme cantidad de conceptos relacionados con el hardware y software y también con desarrollos teóricos.

El primer paso es el proceso de adquirir una imagen digital. Para ellos necesitamos sensores y la capacidad para digitalizar la señal producida por el sensor. El sensor puede ser una cámara video de color o monocromo que produce secuencias de imágenes, puede ser también un scanner que produce líneas al instante o cámaras de fotografía digital.

Una vez digitalizada la imagen, el paso siguiente es el pre-procesamiento de dicha imagen. El objetivo del pre-procesamiento es mejorar la imagen de forma que nuestro objetivo final tenga mayores posibilidades de éxito.

El paso siguiente es la segmentación, que tiene por objetivo dividir la imagen en las partes que la constituyen o los objetos que la forman.

La salida del proceso de la segmentación arroja como resultado una serie de datos o bien las fronteras de la región de cada objeto.

El último paso es el reconocimiento y la interpretación. El reconocimiento es el proceso que asigna una etiqueta a un objeto basada en la información que proporcionan los descriptores. La interpretación consiste en asignar significado al conjunto de objetos reconocidos.

La figura 1.5 muestra gráficamente todos los pasos que intervienen en el procesamiento de una imagen.

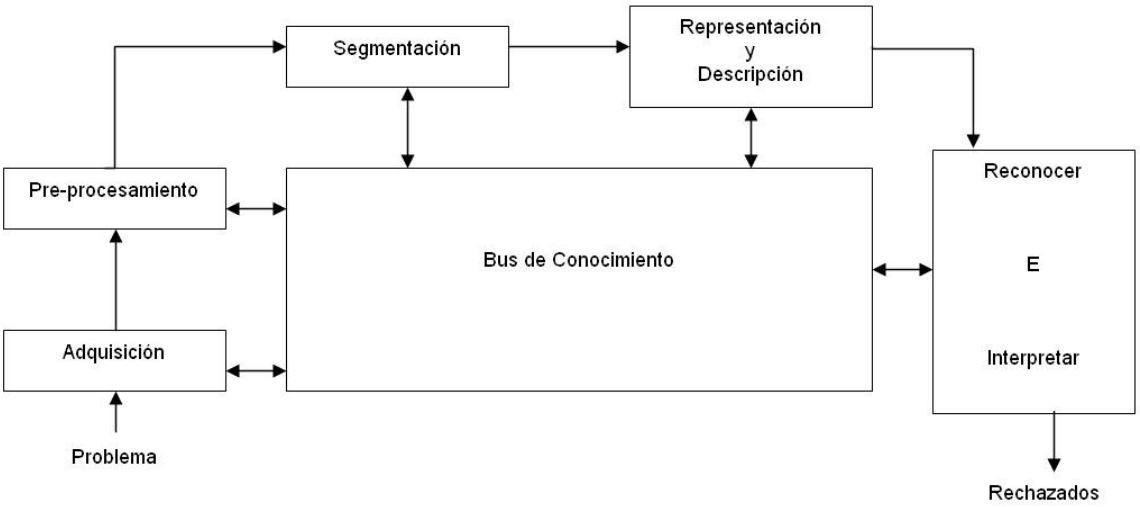



Fig. 1.5 Pasos para el procesamiento de una imagen.[5]

## 1. 4 HERRAMIENTAS PARA EL PROCESAMIENTO DE IMÁGENES

En la actualidad la necesidad de poder procesar una imagen para obtener los resultados deseados ha ido incrementando con el uso de la PC, y esto va en incremento ya que cada día más personas tienen acceso a los servicios que ofrece una computadora.

Existen en el mercado infinidad de herramientas que tienen como finalidad la edición de imágenes que van desde fotografías caseras hasta imágenes de tomas satelitales, estas herramientas aunque sirven para el procesamiento, pero en su diversidad tiene diferentes enfoques.

En el estudio que se realiza, trata de enfocarme a tres de las herramientas que son conocidas, usadas y que engloban las diferentes aplicaciones que se le puede dar al procesamiento de imágenes.

[5] http://www.eeng.dcu.ie/~whelanp/vsg/outline.html

El primer paquete que se analiza es: COREL DRAW; el conjunto de software de gráficos Corel es de los más potente hasta hoy en el mercado CorelDRAW® 13 Graphics Suite es la ultima de las versiones de este software. Respaldado por una década de galardonada potencia creativa, este producto ofrece herramientas de diseño de páginas web, ilustraciones, edición de imágenes, pintura y animación vectorial. El proceso de diseño es sencillo ya que cuenta con una interfaz mejorada y amigable, nuevas herramientas de creación, más compatibilidad, una mayor capacidad de personalización de funciones y nuevas opciones profesionales de producción final. Todo lo que ofrece CorelDRAW® en un solo paquete ha hecho de este uno de los software de mayor demanda en lo que a la edición de imágenes digitales se refiere.

Algo que se puede destacar de este paquete es, que se pueden hacer eventos programados con una interfaz hacia Visual Basic.

Otro de los paquetes es Adobe Photoshop, que es la competencia más fuerte de Corel, ya que están enfocados al diseño y edición profesional, la versión CS2 de Photoshop es la ultima versión la cual incluye nuevos filtros, efectos para fotos y es más flexible para todo tipo de usuarios.

Se puede decir de Adobe Photoshop es el software estándar de edición de imágenes profesional y el líder de la gama de productos de edición de imágenes digitales. Las innovadoras herramientas creativas ayudan a que el proceso de edición de imágenes sea más fácil.

Es muy adaptable a los métodos de trabajos ya que se puede personalizar Photoshop de acuerdo a las necesidades de quien lo maneja. Además, gracias a unos procesos de edición, tratamiento y gestión de archivos más eficaces, el procesamiento es mucho más rápido.

### 1. 4. 1 Image Processing Toolbox de Matlab

El enfoque de esta herramienta va más allá de solo el procesamiento de la imagen, ya que se pueden obtener otros datos o parámetros de una imagen.

El enfoque de MatLab con las imágenes no difiere mucho de los dos paquetes antes mencionados, solo que el IMAGE PROCESSING TOOLBOX no cuenta con un ambiente grafico para la aplicación de filtros, sino que es a través de funciones definidas, y aquí es donde entra la programación.

En comparación con los otros dos software antes mencionados Image Processing Toolbox es el más apropiado para la implementación de un SVA (Sistema de Visión Artificial), debido a los datos que podemos obtener al procesar una imagen y que cuenta con un compilador propio del lenguaje C++ que es un estándar, puesto que es compatible con Microsoft Visual C++, Borland C++ Builder, entre muchos otros, y esto permite la creación de los algoritmos necesarios para el comportamiento del SVA.

Al hacer el estudio de las herramientas antes mencionada fue posible observar que a pesar de que los 3 paquetes tienen como finalidad el procesamiento y filtrado de imágenes, pero tanto Corel Draw como Photoshop están constituidos como herramientas para diseñadores, a diferencia de la herramienta de Matlab (Image Processing Toolbox) que está más orientada a la ingeniería y para la construcción de un Sistema de Vision Artificial. Los dos primeros paquetes citados, son limitados para la implementación de algoritmos ya que aunque manejan programación en Visual Basic, pero está mas orientada a la programación de Macros y no a la programación de acciones de reconocimiento de patrones.

Al adoptar el IPT (Image Processing Toolbox) fue necesario compararlo con otras alternativas comerciales, que tengan como finalidad los SVA, y algunas de las herramientas que hoy se encuentran en el mercado son las siguientes:

### 1. 4. 2 Sistemas Matrox

*Ventajas*

- Sistemas profesionales de gran calidad
- Gran potencia de cálculo en su hardware.
- Funciones muy optimizadas con bajos tiempos de proceso.
- Fácil desarrollo de sistemas multicámara en tiempo real con sincronismos externos.
- Gran desarrollo de drivers para todo tipo de cámaras (analógicas y digitales).
- Herramientas de ayuda para el desarrollo de aplicaciones.

*Desventajas*

- Las librerías de funciones son cerradas, no se proporciona el código.
- Utilizan sistemas de seguridad con llave para poder desarrollar código.
- Funciones desarrolladas únicamente para Windows.
- Sistemas muy caros tanto desde el punto de vista hardware como software.
- Se necesita renovar las licencias anualmente para poder desarrollar código.

*Productos*

Ofrece un conjunto de herramientas software para programadores. Librerías de funciones optimizadas con un API específico para su hardware. Utilidades para definir el interfaz para cualquier cámara.

**MIL (Matrox Imaging Library)**

Librería de funciones para adquisición, transferencia y presentación de imágenes así como para procesamiento y análisis de las mismas.

**MIL-Lite**

Librería de funciones para adquisición, transferencia y presentación de imágenes.

**Matrox inspector.**

Aplicación para realización de pruebas rápidas mediante el acceso interactivo a un gran conjunto de operaciones de imagen.

**Matrox 4Sight-II**

Es una computadora industrial con una tarjeta de adquisición de imágenes que permite la captura, procesamiento, presentación de imágenes y comunicación con el exterior (ethernet, USB, 1394, puertos serie, puertos paralelo, etc.) de una forma sencilla.

Aplicaciones:

- Sistemas de visión industriales
- Procesamiento de imágenes médicas
- Aplicaciones de vigilancia


Fig. 1.6 Sistema Matrox 4Sight-II[6]

### 1. 4. 3 Librería Open CV

Librería de funciones de visión de libre distribución desarrollada por Intel.

*Ventajas:*

- Software libre donde se proporciona el código de las funciones
- Gran cantidad de funciones de análisis de imágenes
- Aplicables con sistemas operativos Windows y Linux

*Desventajas:*

- Los drivers con los frame-grabbers se deben gestionar aparte.
- No se garantiza que el código de las funciones sea el más óptimo.
- Las actualizaciones y el control de la librería no es tan estricto como en las de pago.

[6] www.depeca.uah.es/docencia/doctorado/cursos04_05/82854/docus/CursoVision10.pdf


Fig. 1.7 Gama de Productos Open CV de Intel[7]

### 1. 4 .4 Sistemas Omron

Características generales.

- Sistemas de visión basados en hardware destinados a la captura y tratamiento de imágenes.
- Orientados principalmente a aplicaciones de control de calidad, posicionamiento y medida de piezas u objetos.
- Extensa gama de producto que facilita su adaptación a todo tipo de aplicaciones.
- Sistemas sencillos y compactos con iluminación incorporada en los modelos F10 y F30.
- Sistemas modulares de alta funcionalidad con posibilidad de conexión a varias cámaras, programables, comunicación con otros dispositivos, etc.
- Procesamiento en Binario y Escala grises (256 niveles).
- Resolución de 512 (H) x 484 (V) Pixeles.
- Permite el uso de sistemas de iluminación inteligente.
- Conexión de 1 o 2 cámaras (unidad de dos cámaras).
- Nueve métodos de medida implementados.
- Función de Compensación de Posición.
- 16 Escenas de Trabajo y 16 Regiones de Inspección por escena con salida independiente + salida OR de todas ellas.

[7] www.depeca.uah.es/docencia/doctorado/cursos04_05/82854/docus/CursoVision10.pdf

Fig. 1.8 Sistema Onron F-150 V3[8]

### 1. 4. 5 Halcon

Es un conocido software de visión artificial que es usado por varias empresas alrededor de todo el mundo. Esta creado para facilitar el ahorro de tiempo y dinero a desarrolladores que quieran trabajar en visión artificial.

La arquitectura flexible del HALCON facilita el desarrollo rápido de aplicaciones de visión artificial y análisis de imagen. HALCON proporciona una biblioteca extensa con más de 1100 operadores. Tiene un funcionamiento excepcional para análisis de grises, análisis de forma, metrología, calibración 3D y visión binocular. HALCON asegura una inversión confiable ya que cuenta con una amplia gama de sistemas de operaciones e interfaces que corresponden a más de 25 sistemas de adquisición de imagen.

[8] www.depeca.uah.es/docencia/doctorado/cursos04_05/82854/docus/CursoVision10.pdf

HALCON no requiere un software alterno para el tratamiento o conversión a imágenes 3D, puesto que puede sacar el modelado de las figuras en tres dimensiones únicamente compilando el archivo CAD en que se encuentre el diseño de la figura o imagen, para posteriormente pasarla al procesado de fronteras de los objetos.

### 1. 4. 6 Siemens Pro-Vision

Características

- Software abierto para una configuración flexible
- Desarrollo y carga de programas ejecutables, transferencia de imágenes, test en línea y optimización.
- Aprendizaje de cada dato en línea vía RS232 o PROFIBUS DP
- Memorización opcional de la última imagen de defecto
- Almacenamiento residente del software runtime con librerías y drivers en el sistema de destino
- Programación en C++ de nuevas funciones como OCXs bajo WINDOWS
- Con Provisión, los programas de usuario son independientes del hardware debido al compilador residente.

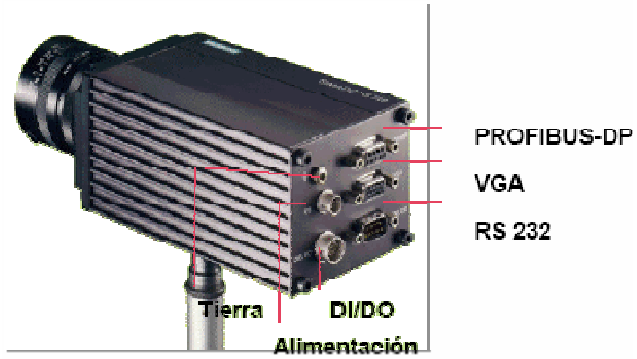



Fig. 1.9 Sistema Siemens Simatic VS 710 basado en PRO-VISION[9]

[9] www.depeca.uah.es/docencia/doctorado/cursos04_05/82854/docus/CursoVision10.pdf

Las funcionalidades que ofrece Pro-Vision para la implementación de un Sistema de Visión Artificial completo, van desde el sensor de video hasta la unidad de procesamiento sin necesidad del uso de la PC, todo comandado por un PLC de la familia Siemens como se puede observar en la figura 1.10.

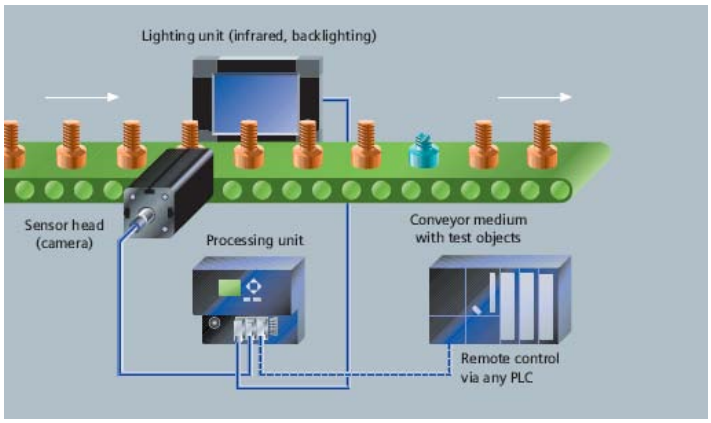



Fig. 1.10 Banda de producción comandada por Siemens Pro-Vision Vs710[10]

La figura 1.11 muestra las partes que componen el sistema pro-vision junto con el dispositivo de entrada de video.

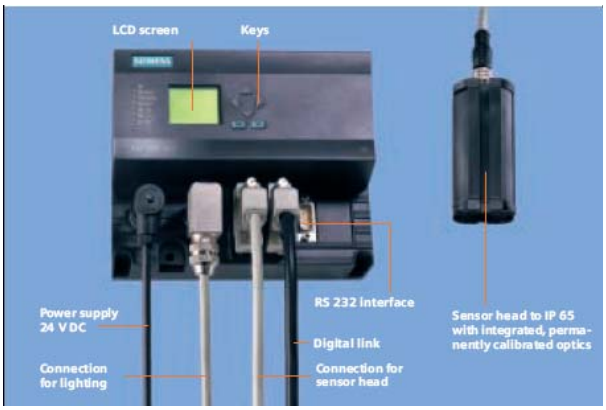



Fig. 1.11 Hardware del sistema Pro-Vision de Siemens[11]

[10] http://www.siemens.be/eit/microautomation/downloads/SIMATIC_VS110_EN.pdf
[11] http://www.siemens.be/eit/microautomation/downloads/SIMATIC_VS110_EN.pdf

## 1. 5 APLICACIONES EN DIFERENTES ÁREAS DE ESTUDIO

La información visual juega un papel muy importante en la comunicación humana, de ahí se desprende el gran interés mostrado por el desarrollo de sistemas para el tratamiento digital de imágenes. Sus aplicaciones son fundamentales e imprescindibles en multitud de campos: ingeniería, física, biología, medicina, etc. A continuación se enuncian algunas aplicaciones del procesamiento y filtrado de imágenes en las diferentes áreas de estudio.

### Biología

En los campos de la biología y biomedicina, el procesamiento de imágenes se usa para analizar visualmente las muestras biológicas. Hay varios casos donde el análisis de muestras ha sido completamente automatizado usando procesamiento de imágenes. De cara a la realización de un buen análisis, se mejoran mediante algoritmos, aquellas características inidentificables y/o poco nítidas de la imagen. La identificación automática, clasificación, categorización y análisis de ADN pueden ser realizadas por el procesamiento de imágenes.

### Procesado de Documentos

Una recolección y procesado automático de documentos e imágenes se presenta útil para bancos y compañías de seguros. Estos documentos están digitalmente comprimidos y guardados. Se detectan e identifican automáticamente la información impresa sobre cheques y otros documentos contables.

### Automatización en la industria

El procesamiento de imágenes se usa para la inspección y supervisión automática en líneas de producción de grandes empresas. Este sistema reduce mucho el error humano al tiempo que proporciona estabilidad y precisión en la producción.

**Diagnostico a partir de Imágenes Médicas**

Los rayos X y las proyecciones CT médicas se digitalizan para examinar áreas internas del cuerpo. Un número determinado de proyecciones CT se combinan y digitalizan automáticamente para producir imágenes 3-dimensionales.

**Teledetección**

Un satélite fotografía la corteza exterior de la Tierra a intervalos regulares y las imágenes se usan para analizar condiciones de crecimiento del cultivo, distribución de la vegetación y para exploraciones de recursos naturales. También la corteza de la Tierra puede ser convertida en un modelo 3-dimensional usando imágenes 2d tomadas por satélites.

**Efectos de video/film**

La industria cinematográfica usa una extensa variedad de técnicas de procesado de imágenes para producir efectos visuales especiales. Las imágenes irreales y otras escenas costosas son artificialmente producidas usando técnicas de Informática Gráfica o/y Procesamiento de Imágenes Digitales.

**Identificación**

Mediante biometría (reconocimiento de huellas dactilares, palmas de la mano, caras, etc.).

**Robótica o Autómatas**

En esta área es donde en la actualidad ha tomado más auge el procesamiento y filtrado de imágenes y concretamente los Sistemas de Visión Artificial; hoy en día se está dotando a los robots o autómatas de sistemas de visión que internamente hacen un procesamiento de las imágenes obtenidas y

así llevan acabo el reconocimiento según los parámetros establecidos en los algoritmos.

### Mundial de fútbol de Robots (FIRA)

FIRA y RoboCup son emprendimientos internacionales dedicados a la organización de campeonatos de fútbol de robots. FIRA (Federation of International Robot-soccer Association) fue fundada en junio de 1997 con el principal objetivo de acercar la ciencia y la tecnología de los robots a las nuevas generaciones, a través del fútbol de robot. Con objetivos similares, RoboCup busca motivar la investigación y desarrollo de soluciones a un problema conocido. Una amplia gama de tecnologías pueden integrarse para resolver este problema.

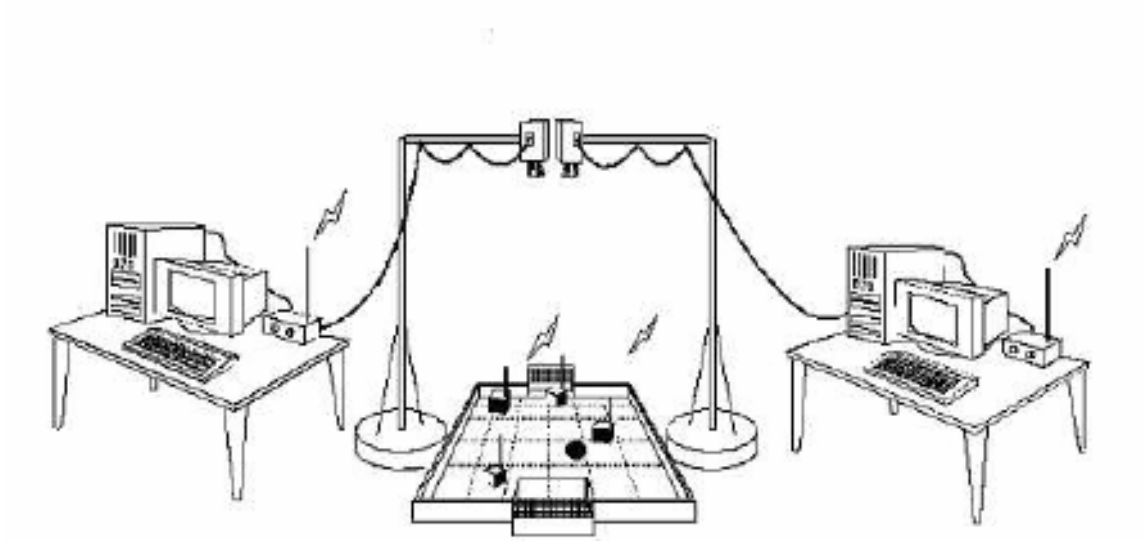


Fig. 1.11 Sistema Completo para control de Robots.[12]

[12] www.depeca.uah.es/docencia/doctorado/cursos04_05/82854/docus/CursoVision10.pdf

Con el objetivo de identificar los robots y la pelota dentro del campo de juego, se utiliza un sistema de visión.

Cada equipo ubica una cámara o sensor sobre su medio campo, incluyendo la línea central, de forma que no sea necesario trasladarla en la segunda mitad del juego, cuando se cambien las canchas. Si ambos equipos deciden ubicar sus cámaras en el centro, deberían ubicarse juntas, tan cerca como sea posible y equidistante del centro.

La altura mínima es de dos metros. La especificación de las reglas del juego es similar a las de la FIFA, ya que los robots también pueden cometer faltas y son sancionados, el sistema completo se muestra en la figura 1.10

Cada robot es independiente, tiene energía propia y un mecanismo que le permite desplazarse. Sólo está permitida la comunicación inalámbrica para todas las interacciones entre la computadora y el robot.

Dentro de este contexto se desarrollan tres proyectos de grado: uno dedicado a la construcción de robots de bajo costo, otro al desarrollo de estrategias para dirigir el comportamiento de los robots, y por ultimo el modulo de visión que es el que nos interesa.

En la figura 1.12 se muestra más claramente las etapas del Sistema de Visión empleado para controlar a los jugadores robots.

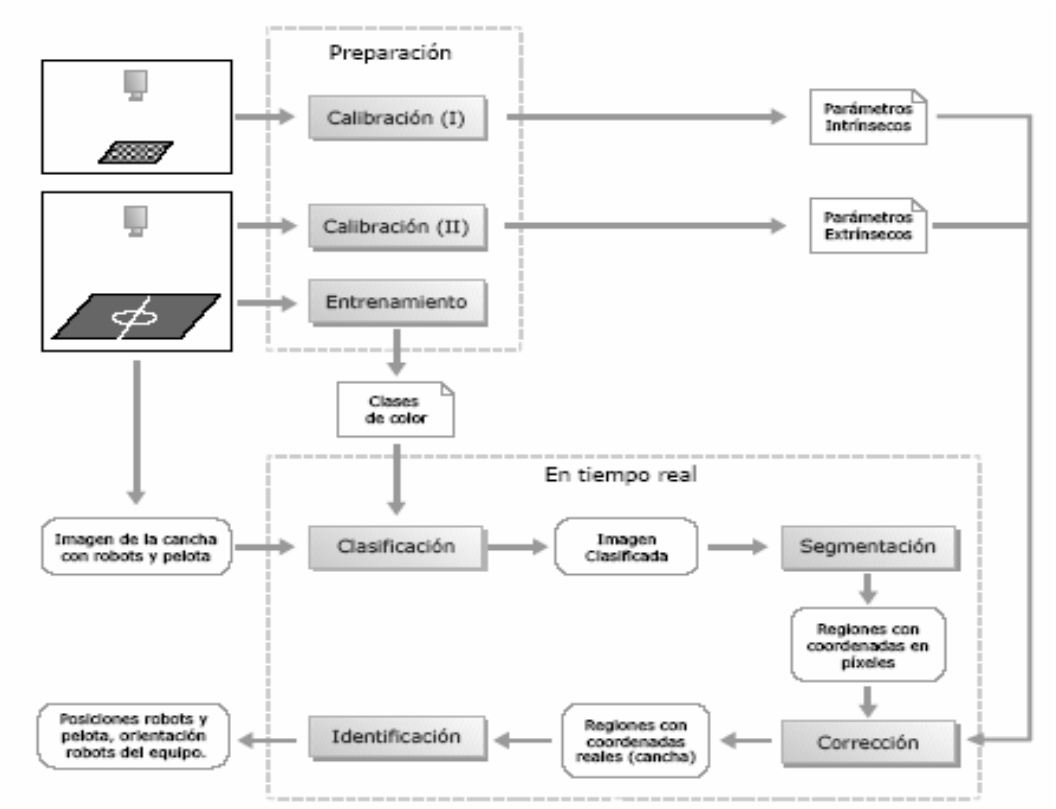



Fig. 1.12 Etapas del Sistema de Visión utilizado en la FIRA.[13]

[13] http://cvpr2000.cs.uiuc.edu/

# CAPÍTULO SEGUNDO

# IMAGE PROCESSING & IMAGE ACQUISITION TOOLBOX

## 2. 1 INTRODUCCIÓN AL IPT (IMAGE PROCESSING TOOLBOX)

Image Processing Toolbox es una colección de funciones incluidas en el paquete Matlab, las cuales permiten realizar múltiples tareas entre las que destacan:

- Transformación de Imágenes Espaciales.
- Operaciones Morfológicas.
- Filtrado Lineal y diseño de filtros para transformación.
- Análisis de Imágenes.
- Registro de regiones distorsionadas y operaciones de mejora.

Algunos conceptos tareas básicas en el procesamiento de imágenes son:

- Leer y desplegar una imagen
- Verificar y saber interpretar como se carga una imagen en el espacio de trabajo.
- Optimizar el histograma de ecualización en una imagen.
- Escribir una imagen en disco.
- Obtener información de un archivo de imagen.

Antes de empezar a conocer el Image Processing Tool, se debe tener en cuenta que el procesamiento de las imágenes es un poco diferente que en los paquetes convencionales, ya que se realizan por medio de comandos.

La siguiente imagen muestra el entorno del MatLab:

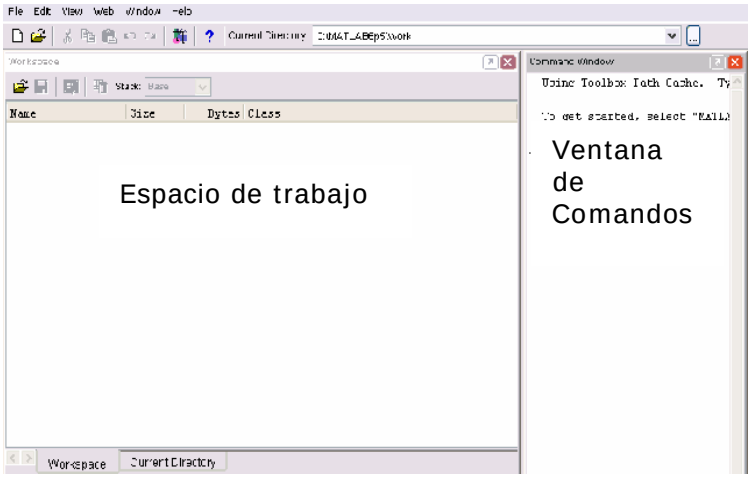



Fig. 2.1 Entorno de Matlab[1]

Matlab cuenta con una ventana de comandos en donde se hacen las llamadas a funciones, y para facilitar el manejo de las funciones trae incluida una ayuda detallada que contiene todos los comandos y la sintaxis correcta, así como ejemplos con explicación paso a paso.

El potencial de Image Processing Toolbox recae en la facilidad de incorporarse con otras herramientas de MatLab, en este caso se va a conjugar el procesamiento y filtrado de imágenes de IPT con la captura y tratado de video que ofrece Image Acquisition Toolbox.

[1] Ventana del Programa Matlab

## 2. 2 CONCEPTOS BÁSICOS EN IPT

**Binary Image (BW)**: Imagen que contiene solo pixeles blanco y negro.

**Image Type**: Define la relacion entre un el valor de un arreglo y el color de un píxel.

**Indexed image(X)**: Imagen cuyos valores de píxeles son indices dentro del mapa de color RGB.

**Intensity image(I):** Consiste en una imagen de intensidad en escala de grises.

**Multiframe image(4-D)**: Arreglo de imágenes que contiene mas de una imagen.

**RGB Image(RGB):** Imagen en la que cada píxel es especificado por tres valores, uno para el componente rojo, uno para el verde y otro para el azul.

Los tipos de imágenes utilizados en el Image Proccesing Toolbox de Matlab son los siguientes:

- Binary
- Index
- Intensity
- RGB

La tabla mostrada en la Fig. 2.2 muestra las diferentes imágenes y los tipos de datos de Image Proccesing Toolbox. Si es una matriz, cómo son interpretados cada uno de los elementos de una matriz, ya sea como píxeles de color dependiendo del tipo de imagen o como una matriz de unos y ceros cuando la imagen es procesada a escala de grises o Binaria.

| Tipo de imagen | Tipo de Datos | Interpretación |
|---|---|---|
| Binary | Logical | Arreglo de unos y ceros |
| Index | Double | Arreglo de enteros en el rango [1, p]. El mapa de color asociado es el arreglo px3 de punto flotante con valores en el rango [0, 1]. |
| | uint8 o uint16 | Arreglo de enteros en el rango [1, p-1]. El mapa de color asociado es el arreglo px3 de punto flotante con valores en el rango [0, 1]. |
| Intensity | Double | Arreglo punto flotante. El rango tipico es [0, 1]. El mapa de color, comúnmente escala de grises es un arreglo px3 de punto flotante en el rango [0, 1]. |
| | uint8 o uint16 | Arreglo de enteros. El rango tipico es [0, 255] ó [0, 65535]. El mapa de color, comúnmente escala de grises es un arreglo px3 de punto flotante en el rango [0, 1]. |
| RGB (trae color) | Double | Arreglo de punto flotante mxnx3 con rango [0, 1] |
| | uint8 o uint16 | Arreglo de enteros mxnx3 en el rango [0, 255] o [<0, 65535] |

Fig. 2.2 Tabla de tipos de imagen y datos del Image Proccesing Toolbox[2]

[2] Tabla elaborada por el Autor de la Tesis

## 2. 3 FUNCIONES BASICAS DE IPT

- Leer Imagen

Ima=imread(‘nombre de imagen.ext’);

Ejemplo:

RGB=imread(‘football.jpg’);

En el ejemplo anterior se carga en una variable llamada “RGB” una imagen con extensión jpg.

- Desplegar imagen

Imshow(‘nombre de variable’);

Ejemplo:

Imshow(RGB);

En el ejemplo anterior se muestra la imagen contenida en la variable llamada “RGB”.

Otra forma de visualizar una imagen es el comando imview: Imview(RGB);

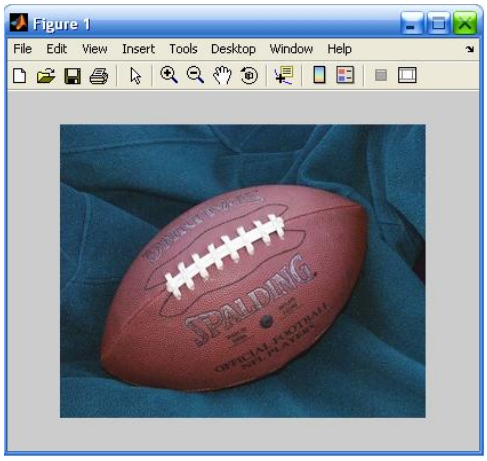



Fig. 2.3 Vista Previa de Imagen.[3]

[3] Image Proccesing ToolBox

- Obtener información de un archivo de imagen:

Para obtener información de una imagen se emplea el comando imfinfo(‘nombre del archivo.ext’) ;

Ejemplo:

Info=imfinfo(‘football.jpg’);.

La variable info contiene la información que es arrojada por la función imfinfo() y el resultado es el siguiente

```
info =
Filename: 'C:\MATLAB7\toolbox\images\imdemos\football.jpg'
FileModDate: '01-Mar-2001 09:52:38'
FileSize: 27130
Format: 'jpg'
FormatVersion: ''
Width: 320
Height: 256
BitDepth: 24
ColorType: 'truecolor'
FormatSignature: ''
NumberOfSamples: 3
CodingMethod: 'Huffman'
CodingProcess: 'Sequential'
Comment: {}
```

- Desplegar Histograma de Imagen:

Para mostrar el histograma de una imagen se emplea el comando

imhist(‘nombre de variable’);

I=imread('pout.tif');

Imhist(I);

El ejemplo anterior da como resultado el siguiente histograma:

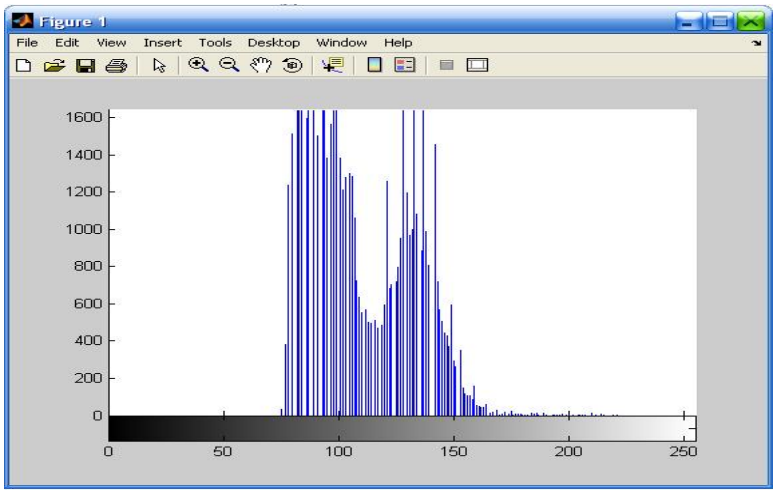



Fig. 2.4 Histograma de Imagen[4]

- Escribir una imagen en un archivo de imagen:

Para guardar una variable de imagen en un archivo de imagen se emplea el comando imwrite(var,’nombre de archivo.ext’)

RGB=imread(‘football.jpg’);

imwrite(RGB,'C:/nuevo.jpg')

En el ejemplo anterior se crea un archivo “jpg” con nombre “nuevo”.

[4] Image Proccesing ToolBox

## 2. 4 APLICACIÓN DE FILTROS

### 2. 4. 1 Edge (Deteccion de Fronteras)

La función EDGE toma una imagen (I) como una entrada y devuelve una imagen binaria (BW) (blanco y negro) del mismo tamaño inicial, la que esta representada con 1`s las fronteras y con 0`s donde no hay nada.

EDGE soporta 6 diferentes métodos para la detección de fronteras en una imagen y son los siguientes:

- Método Sobel
- Método Prewitt
- Método de Roberts
- Método Laplace-Gaussiano
- Método zero-cross
- Método Cany

#### 2. 4. 1. 1 Método Sobel

En el metodo de Sobel se especifica un umbral de intensidad (Thresh) este valor se ignoran todas las fonteras que no sean mayores al valor de umbral especificado. En caso de no especificar dicho valor, la funcion EDGE selecciona automáticamente un valor.

Imagen=Edge(ImagenOrigen,‘sobel’,[Thresh])

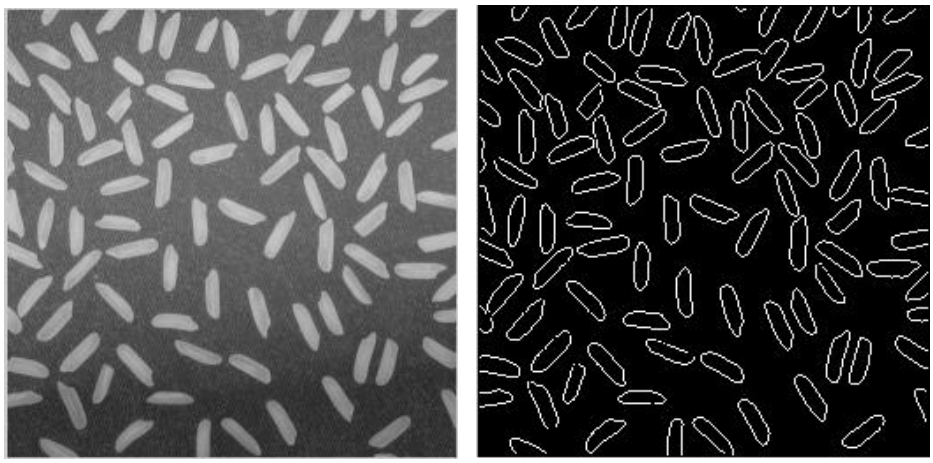
Ejemplo.

Leer una imagen en una variable (I) de tipo UNIT8

```
I=imread('rice.png');
```

Asigna a la variable EI el resultado de la imagen procesada con el método sobel. El valor de umbral se asigna automáticamente.

EI=edge(I,'sobel',[ ]);



(a) (b)

Fig. 2.5 (a) Imagen Original, (b)Imagen Procesada con el Método Sobel[5]

### 2. 4. 1. 2 Método Prewitt

Imagen=Edge(ImagenOrigen,‘prewitt’,[Thresh])

Ejemplo.

Leer una imagen en una variable (I) de tipo UNIT8

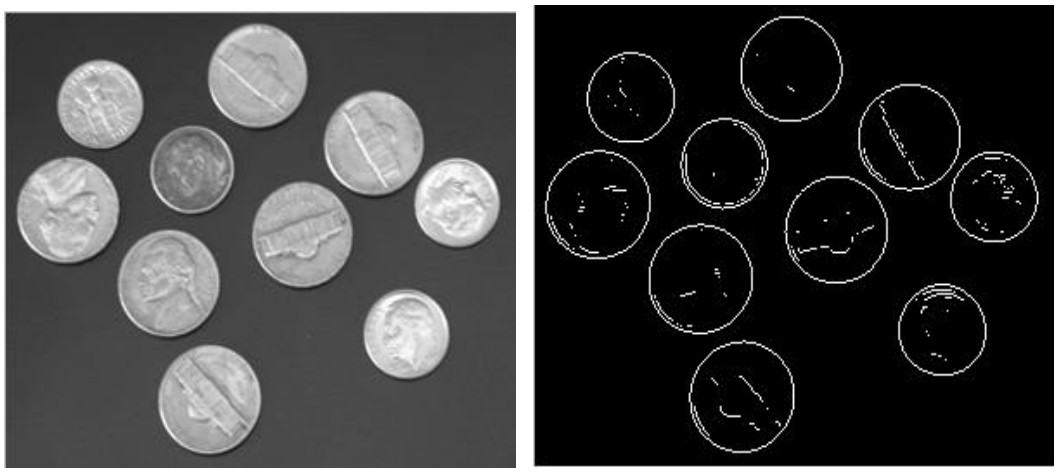
I=imread('coins.png');

Asigna a la variable EI el resultado de la imagen procesada con el método Prewitt. El valor de umbral se asigna automáticamente.

EI=edge(I,'prewitt');

[5] Image Proccesing ToolBox

El resultado del procesamiento es el siguiente:



(a) (b)

Fig. 2.6 (a) Imagen Original, (b)Imagen Procesada con el Método Prewitt[6]

### 2. 4. 1. 3 Método de Roberts

Imagen=Edge(ImagenOrigen,'roberts',[Thresh])

Ejemplo.

Leer una imagen en una variable (I) de tipo UNIT8

```
I=imread('coins.png');
```

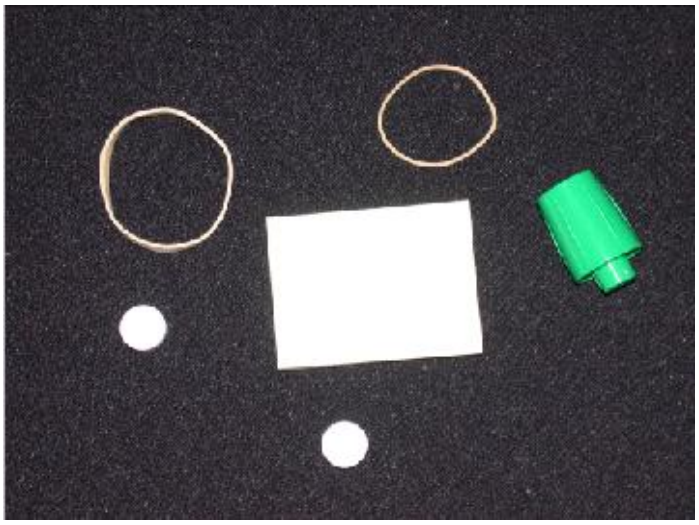
En este caso tendremos que echar mano de la función im2bw que convierte una imagen 2x o 3x a una imagen a blanco y negro, esto con la finalidad de poder aplicar después la función EDGE, de otra manera no seria posible.

```
I2=im2bw(I);
```

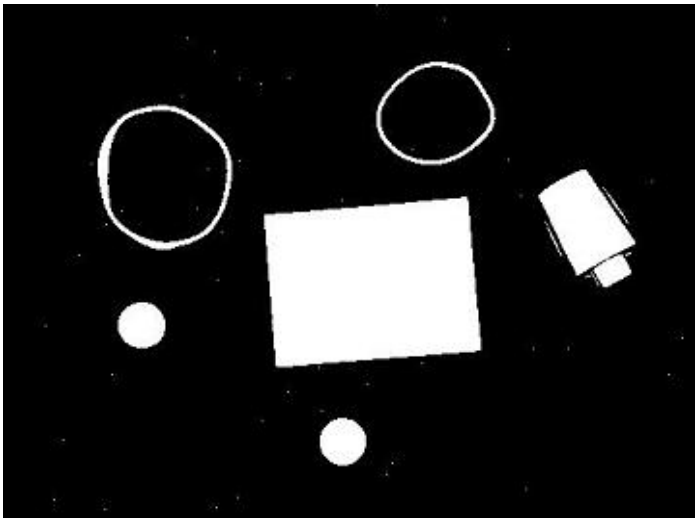
[6] Image Proccesing ToolBox

Asigna a la variable EI el resultado de la imagen procesada I2 con el método Roberts. El valor de umbral se asigna automáticamente.

```
EI=edge(I2,'roberts');
```



(a)



(b)

Fig.2.7 (a) Imagen Original, (b) Imagen Procesada en escala de Grises[7]

[7] Image Proccesing ToolBox

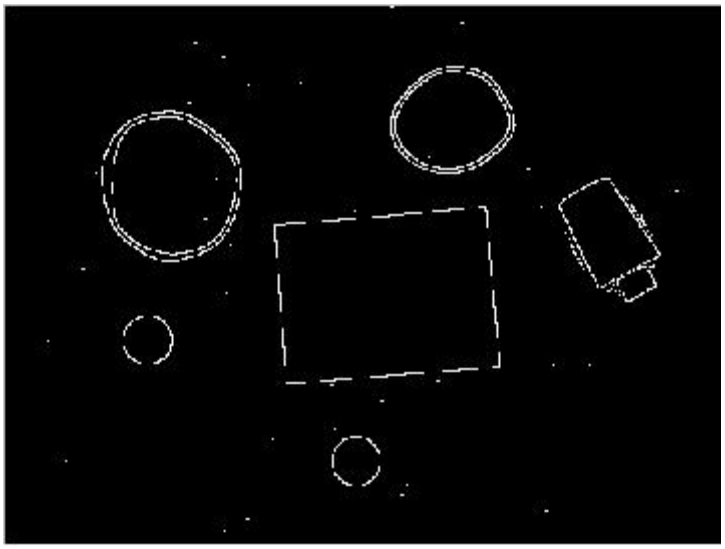


Fig. 2.7 Imagen Procesada con el método de Roberts[8]

### 2. 4. 1. 4 Método Laplace-Gaussiano

Imagen=Edge(ImagenOrigen,'log',[Thresh])

Ejemplo.

Leer una imagen en una variable (I) de tipo UNIT8

I=imread('pears.png');

También en este ejemplo debemos emplear la función im2bw que convierte una imagen 3x a una imagen a blanco y negro, después la función EDGE.

I2=im2bw(I);

Asigna a la variable EI el resultado de la imagen procesada I2 con el método Laplace-Gausiano (log). El valor de umbral se asigna automáticamente.

EI=edge(I2,'log');

[8] Image Proccesing ToolBox

El resultado de aplicar el filtro Laplace-Gaussiano es el siguiente:

Fig. 2.8 Imagen Original que va a ser procesada con método Gaussiano[9]

Esta imagen al ser asignada a una variable se almacena como una imagen de tres dimensiones 486x732x3 y la funcion EDGE no puede procesar esta imagen hasta hacerla de dos dimensiones, una vez aplicando la funcion im2bw obtenemos una variable de dos dimensiones 486x732 y la imagen resultante a dos colores es la siguiente:

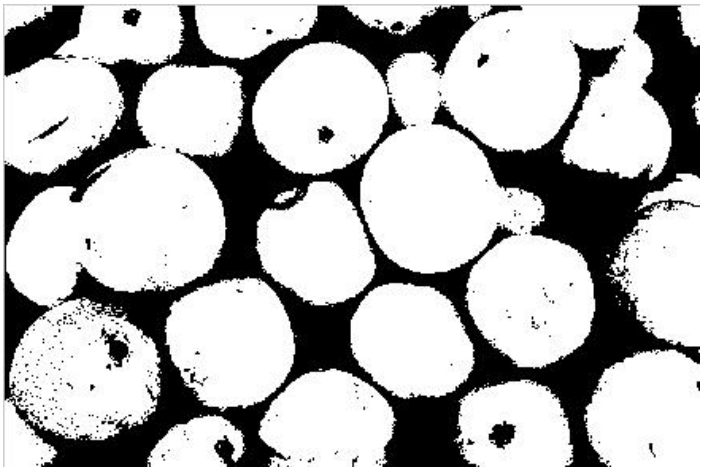


Fig. 2.9 Imagen bidimensional (bw)[10]

[9] Image Proccesing ToolBox
[10] Image Proccesing ToolBox

Una vez que tenemos la variable I2 cargada con la imagen procesada a 2 colores, ahora si podemos aplicar el método Laplace-Gaussiano.

La imagen resultante es depositada en la variable EI de clase lógica de dos dimensiones 486x732. Al desplegar la imagen EI el resultado es el siguiente:

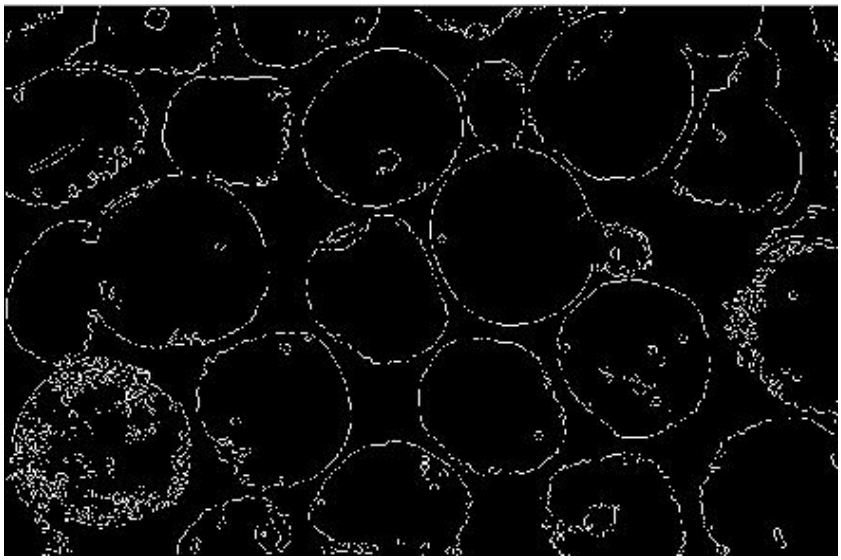


FIg. 2.10 Imagen Procesada con el método LAPLACE-GAUSIANO[11]

No existe un método ideal o uno mejor que otro en la detección de fronteras, ya que dependiendo del tipo imagen que deseamos filtrar, las características que deseamos extraer de la imagen origina o la aplicación de los resultado arrojado por el procesamiento de la imagen, son algunos de los aspectos que dan la pauta para elegir un método de detección de bordes o fronteras; en unos casos es más eficiente un método que otro, ya sea por la iluminación de la imagen, el tipo de objetos contenidos en la imagen, y la nitidez de la imagen

[11] Image Proccesing ToolBox

## 2. 5 IMAGE ACQUISITION TOOLBOX

Image Acquisition ToolBox(IAT) es una colección de funciones que incrementan el potencial de Matlab en lo que a tratamiento de imágenes se refiere, ya que por ser esta una herramienta flexible podemos obtener imágenes en tiempo real desde varios dispositivos que van desde una simple webcam hasta los frame grabers de alta resolución, entre las operaciones que cabe resaltar esta el poder configurar llamadas a funciones cada que ocurra ciertos eventos, también el poder tener una vista previa de video (video preview), así como trabajar directamente con el espacio de datos, es decir cargar variables con datos directamente de la captura de video.

Es posible conjugar las diferentes herramientas de Matlab para poder desarrollar un sistema completo de Visión Artificial, entre las cuales podemos mencionar: Image Processing Toolbox, Data Acquisition Toolbox, Instrument Control Toolbox e Image Acquisition Toolbox.

El IAT incluye adaptadores que proporcionan soporte para varios equipos de adquisición de imágenes profesionales como son Coreco Imaging, Data Translation y Matrox.

El IAT también puede ser conectado con cámaras digitales que soporten el IIDC 1394 que son especificaciones para cámaras digitales (DCAM) desarrolladas por 1394 Association.

La especificación DCAM describen interfases genéricas para el intercambio de datos con camaras digitales IEEE 1394 (Firewire) que son a menudo empleadas en aplicaciones científicas.

Como un plus, IAT soporta varios dispositivos de adquisición que están basados en WDM (Windows Driver Model) como son las cámaras vía USB y las

basadas en IEEE 1394 (FireWire, iLINK), WebCam y Tarjetas decodificadoras para TV.

### 2. 5. 1 Instalación de dispositivos de adquisición de imagen

Para instalar un dispositivo Frame Graber es necesario seguir los pasos que a continuación se mencionan:

1. Instalar el Frame Graber en la PC.
2. Instalar los driver y software de control necesarios para el dispositivo que proporciona el fabricante.
3. Conectar la cámara al Frame Graber.
4. Verificar que la cámara este funcionando apropiadamente al ejecutar el software de aplicación que viene con la cámara y hacer un video preview para comprobar el correcto funcionamiento.

Para los dispositivos genéricos para la adquisición de imágenes bajo Windows, como lo son los USB y los IEEE, no es necesario hacer las pruebas como con los frame grabers, ya que solo basta con configurar el dispositivo, como comúnmente se hace al conectar un nuevo dispositivo a Windows.

En caso de que desconozcamos que tipo de dispositivo tenemos para la adquisición de imágenes podemos hacerlo a través del comando *imaqhwinfo,* este nos muestra de manera resumida que tipo de adaptador tenemos instalado. Un ejemplo de cómo despliega la información es el siguiente:

```
imaqhwinfo
ans =
InstalledAdaptors: {'winvideo'}
```

MATLABVersion: '7.0 (R14)'
ToolboxName: 'Image Acquisition Toolbox'
ToolboxVersion: '1.5 (R14)'

En este ejemplo Matlab detectó un dispositivo de Windows, que en efecto es una WebCam así como la versión de Matlab y la de IAT.

Una vez que se ha identificado el dispositivo, es necesario saber ciertas características del dispositivo de adquisición como lo es la resolución, en que puerto está instalado, así como el formato de video que soporta.

Para esto Matlab también nos proporciona dicha información haciendo uso del *imaqhinfo,* solo que ahora vamos a especificar el identificador del dispositivo como se muestra en el siguiente ejemplo.

```
dev_info = imaqhwinfo('winvideo',1)

dev_info =

DefaultFormat: 'RGB24_352x288'
DeviceFileSupported: 0
DeviceName: 'ICM532A'
DeviceID: 1
ObjectConstructor: 'videoinput('winvideo', 1)'
SupportedFormats: {1x10 cell}
```

El ejemplo muestra que el dispositivo Windows, soporta un formato por default RGB con una resolución de 325x288 píxeles, y es reconocida por el sistema operativo y por matlab, como ICM5324A.

Para poder visualizar todos los formatos que soporta el dispositivo basta con introducir el siguiente comando:

```
celldisp(dev_info.SupportedFormats)
```

Los formatos soportados por el dispositivo son los siguientes:

```
ans{1} =I420_352x288       ans{2}=I420_320x240
ans{3} = I420_160x120      ans{4} = I420_176x144
ans{5} = I420_640x480      ans{7} = RGB24_320x240
ans{8} = RGB24_160x120   ans{9} = RGB24_176x144
ans{10} = RGB24_640x480
```

El IAT emplea dos tipos de objetos para Adquisición de imágenes:

- Video Input Objects VIO
- Video Source Objects VSO

Para entender mejor el significado de los objetos, lo ideal es describir cada uno de ellos.

Video Input Objects representan la conexión entre Matlab y el dispositivo de adquisición de imágenes.

Video Source Objects representan una o mas colecciones de datos que son tratados como una entidad, ya que cuando se crea un Video Input Objec automáticamente IAT crea un o mas Video Source Object asociados a dicho VIO.

Para crear un VIO, basta con llamar a la función *videoinput()* especificando el nombre del adaptador, el ID del dispositivo y el formato que se quiero emplear.

El siguiente ejemplo muestra la creación de un VIO sin especificar el formato por lo que IAT toma el formato por default para la creación de el objeto.

```
vid = videoinput('winvideo',1);
```

Ahora tenemos en el espacio de trabajo cargado el objeto vid y se muestra como la siguiente figura.

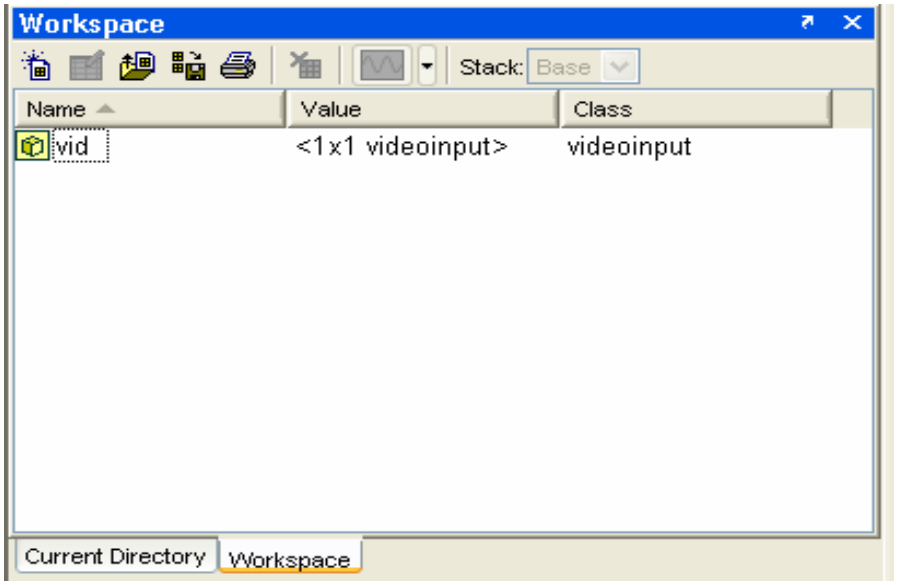



Fig. 2.11 Variable de Video cargada en Espacio de trabajo[12]

Una vez que se ha creado el VIO Matlab esta lista para comenzar a la captura de imágenes, pero antes de comenzar a capturar la secuencia de imágenes, es recomendable hacer un Video Preview para verificar que el

[12] Programa Matlab

dispositivo este captando de manera correcta lo que deseamos captar, para esto emplearemos el comando *preview(),* esta función despliega una ventana en pantalla con el video en tiempo real que en ese momento esta captando el dispositivo, en caso de modificar las características de la imagen o las propiedades de la cámara, automáticamente se verán reflejados en la ventana de preview.

El siguiente ejemplo muestra el Video Preview del dispositivo cargado en espacio de trabajo llamado vid.

```
preview(vid);
```

El resultado de aplicar la función es el siguiente:


Fig.2.10 Imagen Previa del diapositivo[13]

[13] Image Acquisition Toolbox.

Para detener el Video Preview y cerrar la ventana se hace con el comando *closepreview().* El siguiente ejemplo muestra la sintaxis de dicho comando.

Closepreview(vid);

Una vez que se ha calibrado y enfocado de manera correcta la cámara y el objeto que va a ser captado, pasamos a la captura de escenas (frames), ya sea, guardarlos en una variable matricial o también se capturar en un archivo directamente al disco duro.

Para iniciar a capturar frames en una variable matricial, se hace mediante la función *Start()* por ejemplo.

start(vid);

Donde *vid* es un objeto previamente creado como se vio anteriormente, aquí es donde se van a vaciar todos los frames una vez ejecutada la instrucción hasta que ocurra un evento que interrumpa la captura.

A estos eventos que pueden interrumpir la captura se les denomina *triggers* y en Matlab hay varios tipos de triggers y estos se pueden definir para que después de determinado tiempo, de un determinado número de frames, después de algún error, se dispare el trigger y finalice la captura de manera automática.

El número de frames para capturar es por default de 10 frames por trigger, pero para especificar un número diferente de frames se emplea el siguiente comando.

```
set(vid,'FramesPerTrigger',100)
```

En el ejemplo se definen 100 frames por trigger, y de esta manera se puede indicar la cantidad de frames que se desea capturar por trigger. Pero a pesar de esto, no es posible captar demasiados frames por trigger porque todos los frames antes de ser vaciados en el objeto de video, están contenidos en la memoria buffer y esto va a depender de la cantidad de memoria que tenga la computadora en la que se este corriendo la aplicación, porque de otra manera ocurriría un desbordamiento.

Para evitar el desbordamiento es conveniente vaciar continuamente los frames al objeto de video y liberar la memoria para continuar con la captura de los siguientes frames.

Cuando estamos capturando es útil conocer cuantos frames ha sido captados en un determinado momento, y sobre todo cuando los intervalos de frames son grandes como por ejemplo si capturamos 200 frames por trigger y queremos saber en determinado momento cuantos frames hasta el momento están capturados no es necesario detener el proceso, sino que se recurre a la función FramesAcquired.

```
vid.FramesAcquired
```

Al ejecutar esta función nos devuelve el número de frames que hasta ese momento han sido capturadas.

En ocasiones resulta útil poder extraer un frame al momento de la captura para visualizar las características del objeto capturado y más cuando dicho objeto se encuentra en movimiento. Para esto Matlab cuenta con la función llamada getsnapshot que inmediatamente al ser ejecutada, asigna el frame que este siendo captado a una variable para tener así una imagen simple con ese frame.

El siguiente ejemplo muestra el uso de la funcion getsnapshot.

```
frame = getsnapshot(vid)
```

Donde frame es la variable que va a contener la imagen y vid es el objeto de video.

### 2. 5. 2 Captura de frames a Video AVI

Mientras un objeto de tipo video esta corriendo bajo un proceso, resulta útil poder capturar los frames captados en un archivo de video para posteriormente manipular las imágenes, en caso de que no sea necesario hacerlo en tiempo real.

Para llevar acabo el respaldo de los frames capturados a un archivo AVI es necesario seguir lo punto que a continuación de muestran:

1. Se crea un archivo AVI en el que se va a traspasar la información de video y esto se realiza utilizando la función *avifile*. Por ejemplo

si creamos un archivo AVI llamado mi_archivo.avi, la sintaxis completa sería:

```
aviobj = avifile('mi_archivo.avi');
```

2. Posteriormente se configura el objeto que contiene el archivo AVI, en este caso se va a modificar el valor *Quality* que comprende la calidad de captura en el archivo AVI. Por ejemplo:

```
aviobj.Quality=50;
```

En este ejemplo indicamos que la calidad de video sea baja, con la finalidad de que el archivo resultante no ocupe demasiado espacio en el disco duro.

3. Se crea un objeto de video, como ya se vio anteriormente.

```
vid = videoinput('matrox',1);
```

4. Después es necesario hacer algunos ajustes a la configuración del objeto de video, para indicar que se va a capturar hacia un archivo de video.

```
vid.LoggingMode = 'disk&memory';
vid.DiskLogger = aviobj;
vid.TriggerRepeat = 3;
```

En el ejemplo anterior se modifica la propiedad LogginMode donde se indica que va a ser de tipo *disk&memory,* esto es de la memoria buffer al archivo AVI, en la propiedad DiskLogger, se indica el nombre del objeto que creamos anteriormente con el nombre del archivo AVI y también se modificó

la propiedad TriggerRepeat en la que se refiere al intervalo de frames en la que se va a hacer el refresh de la captura.

5. Por ultimo, se prosigue a arrancar el objeto de video para que una vez realizados los ajustes, comience a hacer la captura de los frames a la memoria y posteriormente al archivo AVI que creamos.

Para poder verificar que el archivo AVI se creo de manera satisfactoria, se puede hacer mediante el siguiente código:

```
if(exist('mi_archivo.avi')==2)
disp('Archivo AVI fue creado satisfactoriamente.')
```

La función *exist* devuelve como valor 2 cuando encuentra el archivo, en caso contrario retorna un valor diferente a 2, y una vez encontrado el archivo envía un mensaje en pantalla informando que la operación tuvo éxito.

Para finalizar la captura en el archivo AVI, es necesario detener el proceso a través del comando *close()* como se muestra a continuación.

```
aviobj = close(vid.DiskLogger);
```

Siempre en conveniente remover del espacio de trabajo los objetos de video y las variables de referencia, ya que ocupan memoria, que una vez concluido el proceso de captura, es recomendable liberar.

En nuestro caso solo creamos un objeto de video llamado *vid* y la forma de liberar la memoria es la siguiente:

```
delete(vid);
clear vid;
```

# CAPÍTULO TERCERO

# SISTEMAS DE VISIÓN ARTIFICIAL

## 3. 1 INTRODUCCIÓN A LOS SISTEMAS DE VISIÓN ARTIFICIAL

Para algunos autores la Visión artificial también la han denominado Visión Computacional, y se refiere principalmente a la reconstrucción de diferentes propiedades de una escena o imagen, tales como campos de movimiento, color, profundidad, etc., para destacar aspectos que sean de nuestro interés.

"**La visión artificial** describe la deducción automática de las estructuras y propiedades de un mundo tridimensional, posiblemente dinámico, a partir de una o varias imágenes bidimensionales de él".[1]

"**La visión artificial** puede ser definida como los procesos de obtención, caracterización e interpretación de información de imágenes tomadas de un mundo tridimensional."[2]

De las definiciones anteriores se pude concluir que la Visión Artificial es el estudio de los procesos para obtención de imágenes del mundo tridimensional,

[1] Definición según Nawal http://www.ual.es/~fguindos/VA.html
[2] K.S.FU, MacGraw Hill. ROBOTICA control, detección, visión e inteligencia. Pág. 306

para la posterior definición e interpretación de las propiedades de dichas imágenes.

Un sistema óptico forma una imagen de un objeto o conjunto de objetos tridimensionales. Este es el propósito de un sistema de visión artificial: obtener de esta imagen la información necesaria y útil para la ejecución de una tarea.

En el caso más simple, la información se refiere solamente a la posición y orientación de un objeto aislado; en otros casos se deben reconocer los objetos y determinar sus relaciones, que relación tiene entre si.

Entre las capacidades visuales que posee el ser humano se encuentra la de poder seguir objetos. Aunque lo hagamos de forma inconsciente, mediante esta capacidad podemos realizar múltiples actividades tales como leer, caminar, conducir, etc. El seguimiento visual está estrechamente ligado con muchas de las tareas que realizamos.

La introducción de estas capacidades en los sistemas visuales artificiales es una de las áreas de investigación en la visión por computadora y la robótica de hoy en día.

Al igual que sucede con el ser humano, la capacidad de visión proporciona a un sistema como puede ser un robot dotado de algún mecanismo de percepción, poder responder a su entorno de forma inteligente tal y como lo haría una persona.

El uso de visión y otros esquemas de percepción están motivados por la constante necesidad de aumentar la flexibilidad y los campos de aplicación de dichos sistemas, esto es poder contar con maquinas inteligentes que respondan a estímulos en este caso visuales y que además puedan aprender nuevas

tareas lo que hace que no sean limitadas y que con el tiempo vayan ampliando el numero de soluciones que puede dar cada vez que se presente un problema.

La visión se considera como la más potente capacidad sensorial de un robot y como es de suponerse los sensores, conceptos y hardware de proceso asociados con la visión artificial son considerablemente más complejos que los asociados con las actividades de percepción habituales como pueden ser el uso de sensores de proximidad, y sensores de contacto.

La visión artificial (VA) se ha construido, en gran parte, sobre tres campos relacionados:

- El procesamiento de imágenes.
- El reconocimiento de objetos (o formas).
- El análisis de escenas.

La VA tiene que ver con la producción de nuevas imágenes a partir de las ya existentes.

## 3. 2 FUNDAMENTOS BASICOS DE LOS SVA

Como anteriormente se había mencionado, estos procesos también suelen denominarse “Visión por computadora”, pueden a su vez ser subdivididos en seis áreas principales:

- Captación
- Pre-procesamiento
- Segmentación
- Descripción
- Reconocimiento
- Interpretación

La Captación: Es el proceso a través del cual se obtiene una imagen visual mediante dispositivos de entrada como pueden ser cámaras de video, scanners, tarjetas digitalizadotas, etc.

El Pre-procesamiento: Incluyes las técnicas tales como la reducción de ruido y realce de detalles, en otras palabras es optimizar las imágenes captadas para obtener mejores resultados.

La Segmentación: Es el proceso en el cual se divide la imagen en objetos que sean de nuestro interés.

La Descripción: Mediante este proceso se obtienes características por ejemplo, tamaño, forma, posición, etc., convenientes para diferencial un objeto de otro.

El Reconocimiento: Es el proceso que identifica los objetos de la segmentación (por ejemplo, una tornillo, una llave, una botella, etc.).

La Interpretación: Se encarga de asociar un significado a un conjunto de objetos reconocidos.

Es conveniente agrupar a los SVA en base a la clasificación anterior según su complejidad en tres niveles:

- Visión de Bajo Nivel
- Visión de Medio Nivel
- Visión de Alto Nivel

Debido a que no existen fronteras claras entre los tres niveles en un SVA, se emplean los siguientes criterios para categorizar cada uno de ellos.

La Visión de Bajo nivel se asocia a aquellos procesos que son primarios en el sentido de que pueden ser considerados como reacciones automáticas sin necesidad de algún tipo de inteligencia. Algunas tareas que se realizan en este nivel pueden ser: Percepción y Pre-procesamiento, es decir la obtención de una imagen y la eliminación del ruido de dicha imagen.

En lo que se refiere a la Visión de Nivel Medio son asociados los procesos que extraen, caracterizan y etiquetan componentes de la imagen que se obtiene por la visión de bajo nivel. Algunas de las tareas de este nivel pueden ser la segmentación, la descripción y el reconocimiento.

La Visión de Alto Nivel se refiere a procesamientos que tratan de emular la cognición, todo esto se hace mediante la implementación de algoritmos.

## 3. 3 DIAGRAMA A BLOQUES DE UN SVA

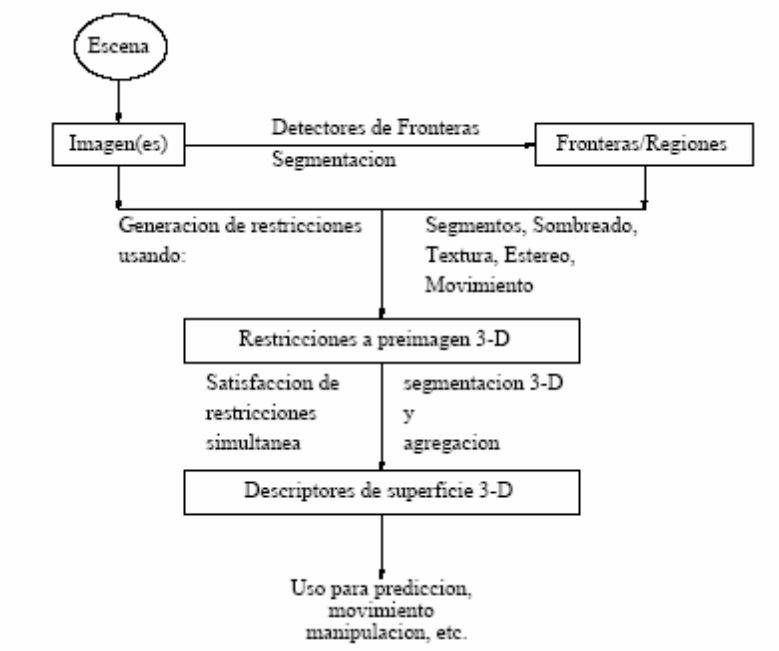



Fig. 3.1 Ejemplo típico de Diagrama a Bloques de un SVA.[3]

[3] http://www.diinf.usach.cl/~dmery/imagenes/proc.htm

Un sistema de visión artificial consta de forma general de un conjunto de subsistemas capaces de realizar las funciones de:

- Adquisición de imágenes.
- Análisis de imágenes.

La ADQUISICIÓN DE IMÁGENES consta de:

- Iluminación
- Óptica + sensor
- Digitalizador

Por parte del ANÁLISIS DE IMÁGENES podemos mencionar:

- Preprocesado
- Segmentación
- Descripción
- Reconocimiento

### 3. 3. 1 Adquisición de imágenes

Conjunto de operaciones que se efectúan para transformar la Información luminosa de una escena en una señal digital.

Este proceso permite almacenar una escena (imagen) en memoria o disco de forma digitalizada.

Las operaciones necesarias en la adquisición de imágenes son:

- Iluminación de los objetos que van a ser captados para que la imagen obtenida sea mas nítida.

- Adquisición de la imagen a través de un sensor (transforma información luminosa en eléctrica analógica) y su óptica asociada (lente).
- Digitalización: conversión analógico-digital a través de un ADC (la señal eléctrica analógica se transforma en digital).

La figura 3.2 muestra de manera clara y sencilla los elementos de un SVA:

### 3. 3. 2 Esquema de un SVA

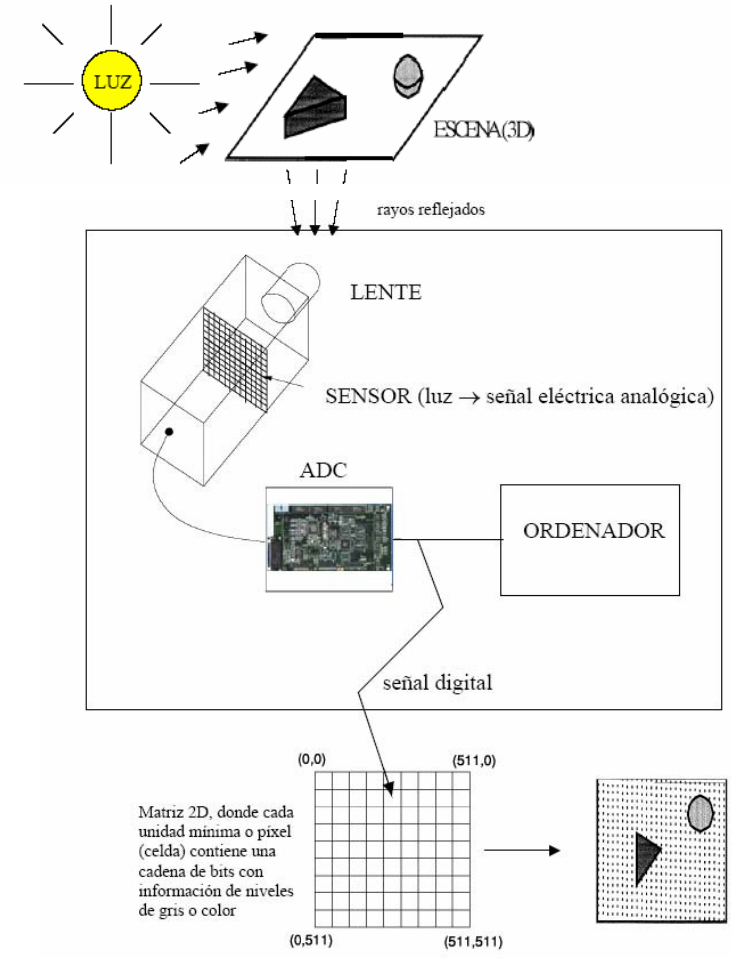



Fig. 3.2 Esquema General de un SVA.[4]

[4] http://www.diinf.usach.cl/~dmery/imagenes/proc.htm

Para visualizar la señal digital según el diagrama anterior, ésta debe pasar por un DAC e iría conectada a un monitor.

### 3. 3. 3 Análisis o procesado de imagen

El análisis o procesado de una imagen engloba a todas aquellas técnicas que permiten extraer una información explícita (posición, tamaño...) de los objetos que componen la imagen o escena.

Estas técnicas se pueden englobar en cuatro bloques:

*Preprocesado:* Conjunto de técnicas encaminadas a realizar una mejora de la imagen digitalizada.

Consiste en modificar las características de la imagen digitalizada:

- Nivel de gris
- Contraste
- Eliminación de ruido
- Realce de características
- Transformaciones geométricas

*Segmentación:* Proceso que divide una escena en partes constituyentes u objetos, los cuales se extraen de la imagen para su posterior reconocimiento y análisis.

Muchas de las técnicas de segmentación se basan en el análisis de:

- Discontinuidad
- Similitud

*Descripción:* consiste en extraer una serie de características (descriptores) de un objeto para reconocerlo.

Los descriptores, a ser posible, deben ser independientes de la localización, orientación y cambios de escala de los objetos de una escena.

Deben contener suficiente información para distinguir un objeto de otro.

**Reconocimiento:** Consiste en un proceso de etiquetado e identificación.

Identificar cada objeto de una escena y asignarle una etiqueta.

Identificar cada objeto segmentado en función de un conocimiento previo (inteligencia artificial).

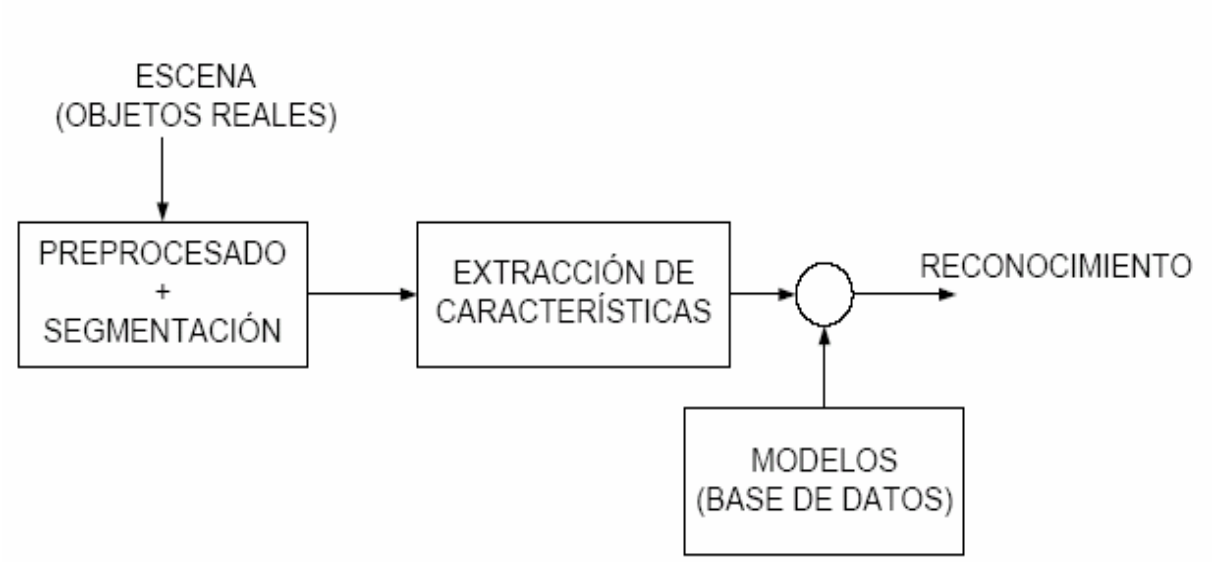



Fig. 3.3 Esquematización del Reconocimiento.[5]

[5] http://www.mathworks.com/access/helpdesk/help/toolbox/images/images.shtml

Para llevar a cabo cada una de las fases de un sistema de VA se requiere un hardware específico.

Los componentes fundamentales (hardware) de un sistema de VA son:

- Sistema de iluminación
- Sistema de captación: óptica + sensor (ambos generalmente integrados dentro de una cámara).
- Conversor analógico-digital (ADC): puede estar dentro de la cámara (cuando la cámara tiene salida digital), integrada incluso en el sensor de la imagen dentro del ordenador (necesaria cuando la cámara tiene salida analógica). En este último caso, el ADC está dentro de una tarjeta llamada tarjeta de adquisición de vídeo (o capturadora de vídeo o frame-grabber).
- Ordenador central, compuesto por:
  - *Tarjeta de adquisición de vídeo, capturadora de vídeo o frame-grabber* (“imprescindible” si la señal proviene de una cámara con salida analógica).
  - Funciones: Recoger la señal analógica o digital procedente de la cámara, convertir la señal cuando es analógica en digital, almacenar la señal digital en memoria.

Muchas tarjetas comerciales incluyen:

- Memoria propia.
- Procesador propio (y software de procesado).
- Conversores digital-analógico y analógico-digital.
- Presentación en pantalla de la imagen.

*C.P.U* (Unidad central de proceso), con funciones:

- Control central.
- Operaciones especificas de la aplicación.

*Memoria central:* Memoria que utiliza el ordenador para almacenar programas de aplicación y datos (imagen digitalizada...).

*Tarjetas de entrada/salida (I/O) y controladoras*:Tarjetas de comunicación con los periféricos.

- Tarjetas vía serie RS-232, USB, firewire...
- Tarjetas vía paralelo.
- Comunicaciones a redes locales.
- Controladoras de monitores.
- Impresoras.

*Periféricos:*

- Monitores de vídeo.
- Impresoras.

*Entorno industrial*:(robot, actuadores industriales...).

## 3.4 SISTEMAS HARDWARE DE VISIÓN ARTIFICIAL

La figura 3.4 muestra una Cámara con salida analógica o digital conectada a PC por medio de frame grabber (capturadora de vídeo) que pueden ir directamente instalada en un puerto PCI o USB; esta tarjeta tiene la capacidad de convertir video Analógico a Digital o viceversa, ya que cuenta con un procesador independiente y memoria RAM.

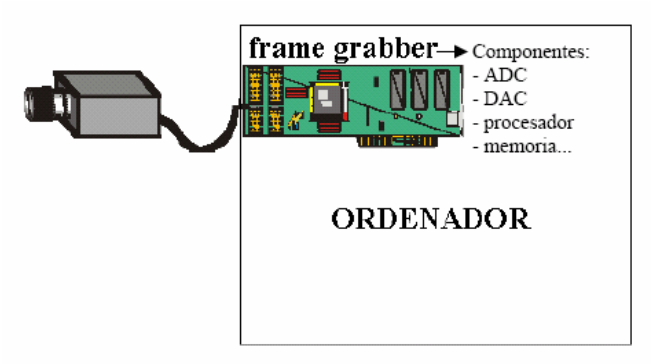



Fig. 3.4 Hardware de un SVA.[6]

La figura 3.5 se muestra una Cámara con salida digital conectada a PC "directamente" mediante el puerto IEEE-1394,Ethernet o USB 2.0 que es de alta velocidad y la misma cámara ya pasa a la PC, el video en formato digital sin la necesidad de una tarjeta de captura.

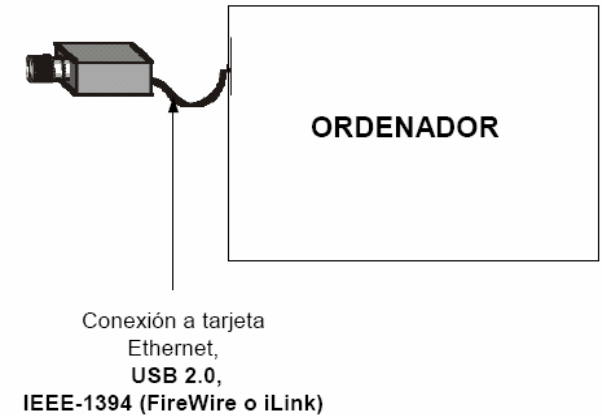



Fig. 3.5 Conexión Directa a través de IEEE.[7]

## 3. 5. SISTEMAS DE ILUMINACIÓN

La iluminación es un factor muy importante en el proceso de visión ya que una iluminación adecuada puede simplificar considerablemente los algoritmos de detección y extracción de las características de los objetos que forman la imagen.

[6] http://www.mathworks.com/access/helpdesk/help/toolbox/images/images.shtml
[7] http://www.mathworks.com/access/helpdesk/help/toolbox/images/images.shtml

Para cada aplicación debe estudiarse el sistema de iluminación mas adecuado con el fin de realzar los aspectos que se pretenden estudiar por VA.

La importancia de las técnicas de iluminación puede suponer un 70% de un proyecto de análisis automático de imágenes.

### 3.5.1 Aspectos a considerar en la iluminación

**Precisión del sistema:**

- Tamaño del defecto a detectar.
- Precisión con la que se desean detectar los defectos (tolerancia).

**Tipo de escena:**

- Tamaño del objeto a estudiar.
- Posiciones fijas o indeterminada del objeto en la escena.

**Profundidad de enfoque:**

- Distancia (Z) del objeto.

### 3.5.2 Características ópticas del objeto

- Superficie lisa (ángulo incidente = ángulo reflejado).
- Superficie rugosa (reflectancia difusa: ángulo reflejado en dirección aleatoria)
- Reflectancia direccional (reflectancia en mayor o menor grado según la dirección).
- Sensibilidad a ciertas longitudes de onda.
- Superficie plana, curva, irregular...etc.

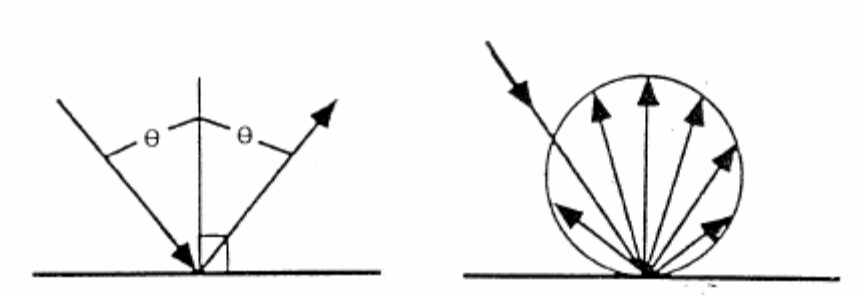



Fig. 3.6 Reflectancia.[8]

### 3. 5. 3 Fuentes de Luz

- Incandescente (estándar y halógena)
- Fluorescente
- Led.
- Láser
- Luz estroboscópica

Fig. 3.7 Fuentes de Luz.[9]

**Iluminación incandescente (estándar y halógena)**

**Ventajas:**

- Equipos de bajo coste, fáciles de utilizar.
- La radiación luminosa a veces se puede ajustar con un aerostato.

[8] http://www.mathworks.com/access/helpdesk/help/toolbox/images/images.shtml
[9] http://www.cs.mu.oz.au/research/vislab/

Con halógenos se consigue luz parecida a la natural a bajo coste.

**Desventajas:**

- El tiempo de vida depende de las condiciones de trabajo.
- Para incrementar la vida útil se recomienda trabajar al 75% de su tensión nominal.
- Generan mucho calor, por ello, en ocasiones, deben ir encerradas en reflectores (también, a veces, se usan espejos
- disipadores de calor para que este no afecte a los objetos u otras partes del sistema).
- Elevado consumo para el flujo luminoso que entregan (muchas pérdidas por calor).
- A 50 Hz pueden aparecer picos de intensidad luminosa. Conviene utilizar una fuente de alimentación continua.

**Aplicaciones:**

- Iluminación en pequeñas superficies (hogar).
- Se utilizan en escenas con iluminación frontal.

**Fuentes fluorescentes**

**Ventajas:**

- Alta eficiencia (lúmenes/vatio) y mínima disipación térmica.
- Pueden tener diversos colores.
- Existen multitud de tamaños y formas.
- Emiten luz difusa (en todas direcciones).
- Larga vida útil.
- Es preferible trabajar a frecuencias elevadas.

**Desventajas:**

- La iluminación varía con la temperatura (de hecho no trabajan correctamente a bajas temperaturas).
- La iluminación disminuye con el tiempo.

**Aplicaciones:**

- Se utilizan para iluminar escenas grandes.

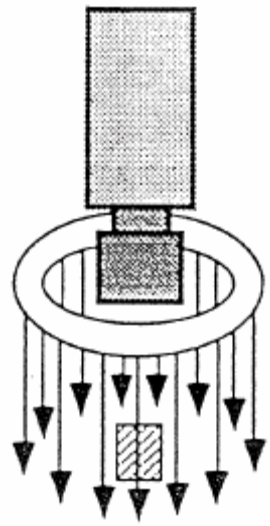


Fig. 3.8 Luz Fluorescente.[10]

**L E D**

**Ventajas:**

- Luz monocromática, a diferentes longitudes de onda.
- Componentes de larga vida y alto rendimiento o eficiencia luminosa.
- Necesitan una fuente de alimentación de baja potencia con corriente constante.

[10] http://www.mathworks.com/access/helpdesk/help/toolbox/images/images.shtml

**Desventajas:**

- Poca intensidad luminosa, que se puede aumentar con matrices de led’s (nivel luminoso más alto pero aún caros).
- Ligeras diferencias en longitud de onda e intensidad entre led's diferentes.

**Aplicaciones:**

- Sistemas de señalización vial, de iluminación arquitectónica, ornamental y museos, paneles publicitarios.
- Se pueden utilizar array's de led's para iluminaciones a contraluz.

**Láser**

**Ventajas:**

- Luz monocromática, fuertemente direccionada.
- Son posibles diversos patrones de luz: puntos, parrilla, líneas...

**Desventajas:**

- Precio elevado.

**Aplicaciones:**

- Medidas de características de los objetos.
- Curvas multidimensionales.
- Profundidad en líquidos.
- Espesores.
- Medidas en tres dimensiones:
- Coordenadas x,y,z.

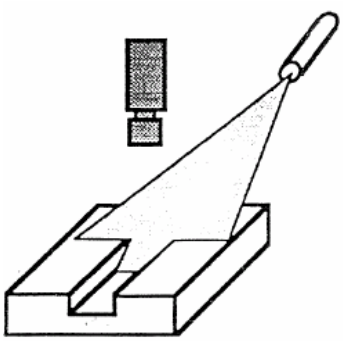


Fig. 3.9 Ejemplo Luz de Láser.[11]

**Luz Estroboscópica (FLASH)**

Consiste en emitir una gran intensidad de iluminación en un espacio de tiempo muy corto (µs) y que apenas depende de la luz ambiente (flash). Se pueden utilizar lámparas de descarga de Xenón para generarla.

Puede hacerse intermitente a una velocidad muy alta como por ejemplo: 1000 Hz (1000 flashes por segundo). Deben estar sincronizados con el sistema de adquisición de imágenes.

**Aplicaciones:**

Inspecciones de objetos en movimiento.

### 3. 5. 4 Técnicas de iluminación

La iluminación de la imagen es un factor importante ya que de ella depende la fidelidad con que se capten las imágenes, y la complejidad del algoritmo que se va a emplear para el modelo de la visión.

La iluminación arbitraria no suele ser aceptable ya que se obtienen imágenes de bajo contraste, reflexiones, especulares, sombras y resaltar detalles que no nos interesan.

[11] http://www.mathworks.com/access/helpdesk/help/toolbox/images/images.shtml

Un sistema de luces bien diseñado ilumina una imagen de forma que la complejidad en la imagen percibida sea mínima, aumentándose a su vez la información necesaria para la extracción y detección del objeto.

Existen cuatro principales técnicas para la iluminación usadas para el área de trabajo de del SVA y son las siguientes:

- Iluminación Difusa
- Retro Iluminación
- Iluminación Estructural
- Iluminación Direccional

El uso de la iluminación Difusa se suele emplear para objetos caracterizados por superficies suaves y regulares. Este tipo de iluminación se utiliza en aplicaciones donde las características de la superficie son importantes.

La Retro Iluminación produce imágenes en blanco y negro (sin matices). Esta técnica es apropiada en aplicaciones las cuales las siluetas de los objetos son superficies para el reconocimiento u otras medidas.

La siguiente figura es un ejemplo de la retro iluminación:

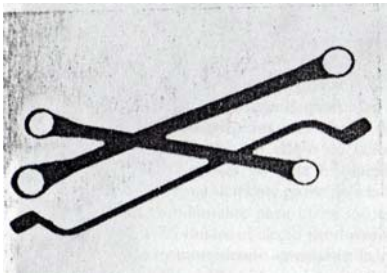


Fig. 3.10 Imagen percibida con Retro Iluminación.[12]

[12] K.S.FU, MacGraw Hill. ROBOTICA control, detección, visión e inteligencia. Pág. 318

La iluminación estructural se sirve de la proyección de los puntos, franjas o rejillas sobre la superficie de trabajo. Esta técnica tiene dos ventajas importantes.

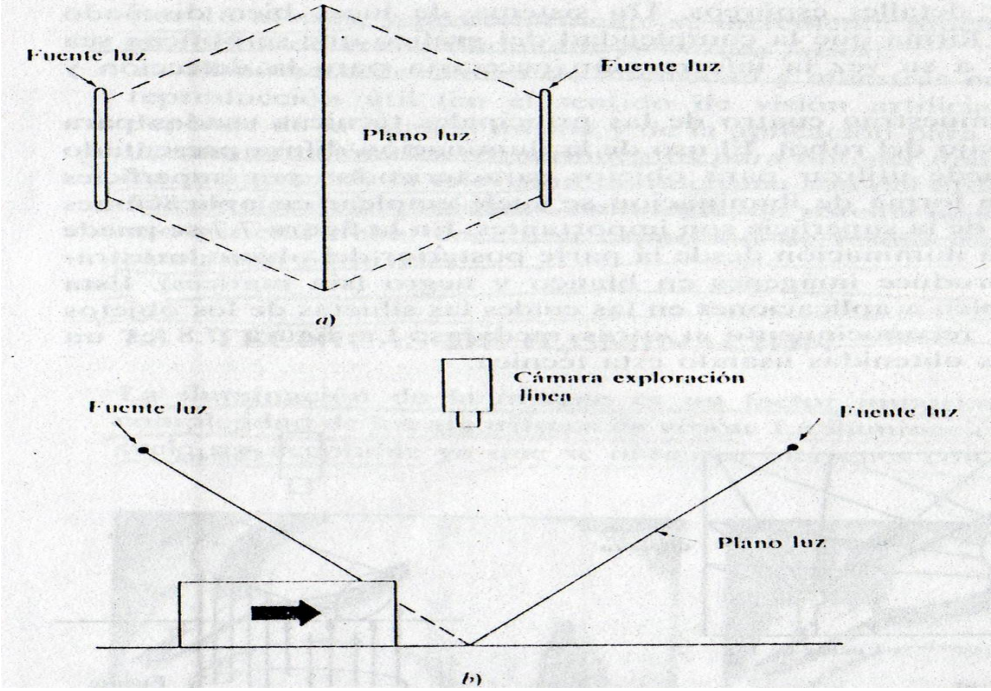



Fig. 3.11 Esquema de Iluminación Estructural.[13]

En prime lugar establece un patrón de luz conocido sobre la superficie de trabajo y la diferencias con este patrón indican la presencia de un objeto, simplificándose así el problema de detección del objeto. En segundo lugar, analizando la forma en que el patrón de luz es distorsionado, es posible obtener información de las características tridimensionales del objeto.

La técnica de iluminación direccional es útil para la inspección de la superficie de los objetos. Los defectos en la superficie, tales como hoyos y arañazos, pueden ser detectados usando un haz de luz altamente direccional (por ejemplo, un haz láser) y medir el grado de dispersión. Para

[13] K.S.FU, MacGraw Hill. ROBOTICA control, detección, visión e inteligencia. Pág. 319

superficies sin desperfectos se difumina un poco de luz en dirección hacia la cámara. Por otro lado, la presencia de un desperfecto suele inspeccionar la cantidad de luz dispersada hacia la cámara, facilitando así la detección del defecto.

Las figuras 3.12, 3.13 y 3.14, que se muestran a continuación contienen ejemplos de las formas básicas de iluminación:

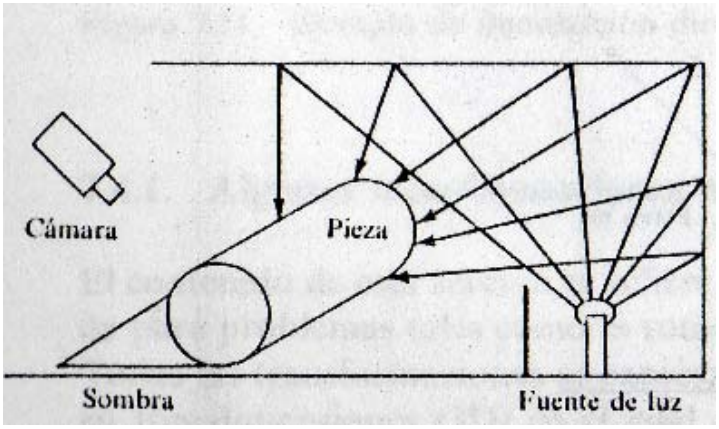
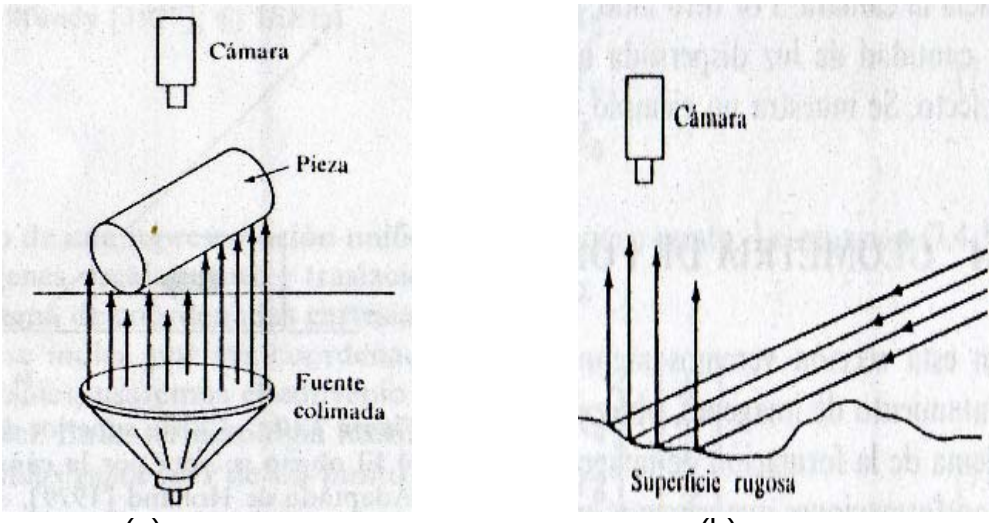



Fig. 3.12 Iluminación con superficie reflectora.[14]




(a) (b)

Fig. 3.13 (a) Iluminación con fuente colimada, (b) Iluminación en superficie rugosa[15]

[14] K.S.FU, MacGraw Hill. ROBOTICA control, detección, visión e inteligencia. Pág. 316
[15] K.S.FU, MacGraw Hill. ROBOTICA control, detección, visión e inteligencia. Pág. 316

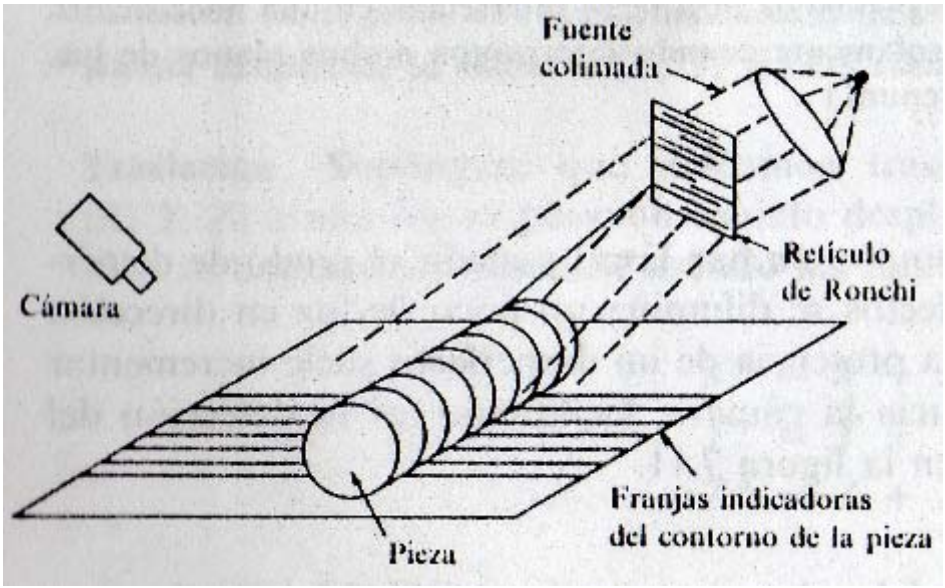



Fig. 3.14 Iluminación con rejilla o película y fuente colimada[16]

[16] K.S.FU, MacGraw Hill. ROBOTICA control, detección, visión e inteligencia. Pág. 316

# CAPÍTULO CUARTO

# DISEÑO Y CONSTRUCCIÓN DE UN SISTEMA DE VISION ARTIFICIAL BÁSICO DE SEGUNDO NIVEL

## 4. 1 MARCO TEORICO DEL PROYECTO

El proyecto que se describe en este capítulo consiste en un sistema de visión artificial que detecta las fronteras entre varios objetos (figuras regulares), y así comprobar que un sistema inteligente puede ser completamente funcional sin el uso de sensores, apoyado únicamente en los principios de la visión artificial y comprender la importancia de la iluminación dentro de los SVA.

Este sistema se considera dentro del nivel medio o segundo nivel, en el que en base a una imagen captada por el dispositivo de entrada de video, que en nuestro caso consta de una webcam, se obtienen varios frames con imágenes que posteriormente serán tratadas con las herramientas del Matlab como lo son: Image Proccesing Toolbox e Image Acquisition Toolbox.

Una vez obtenidas las imágenes y almacenadas dentro del espacio de trabajo de Matlab, son procesadas para la obtención de una imagen en escala de grises (blanco y negro), una vez que tenemos dicha imagen se prosigue a obtener los bordes de la imagen para diferenciar a los objetos del fondo, ya que

solamente nos interesa conocer la ubicación de los objetos que son captados por la cámara.

Al obtener las fronteras de cada objeto captado por la cámara, resultará más sencillo conocer la ubicación dentro del espacio de trabajo puesto que solo obtendremos una matriz con los valores de las fronteras.

También en este sistema se podrá comprobar la efectividad de dos o tres tipos de iluminación y los resultados obtenidos por las diferentes ubicaciones de la fuente luminosa.

Se va a aplicar luz halógena blanca y amarilla, pero la proyección de la luz es la que determinará la calidad de la captura al momento de obtener la imagen, y al aplicar los filtros se obtendrá una imagen más nítida y parámetros más exactos de la información de cada objeto.

Normalmente, la visión artificial se utiliza para tomar decisiones a partir de la información proporcionada por el sistema de adquisición de imágenes y las transformaciones y operaciones realizadas con ellas.

La información extraída se puede considerar como un vector que recoge las características o rasgos diferenciadores de la imagen analizada, dicho vector consiste en una variable de tipo matriz, que contiene ceros y unos; los ceros representan los bordes de los objetos y los unos representan el fondo de la imagen.

En el caso de que se trate de una aplicación de medición, este vector recoge todas aquellas medidas que se desean obtener pues una vez que obtenemos la matriz de la imagen, es sencillo determinar las dimensiones de las objetos captados por la cámara. En aplicaciones de inspección, y sobre

todo, en aplicaciones de clasificación, este vector es el conjunto de datos con los que ha de trabajar un reconocedor o clasificador encargado de extraer las conclusiones posibles a partir del vector de entrada.

Para el diseño del clasificador es necesaria una etapa de selección de características y una etapa de aprendizaje o entrenamiento que almacene los conocimientos adquiridos en un banco de datos, para poder dar solución a todos los posibles problemas que se presenten en un futuro.

Generalmente, se usa el prototipo de un sistema clasificador como evaluador del conjunto de características en pruebas de calidad, dado que este método proporciona una mayor confiabilidad y mejores resultados, aunque sea costoso de momento, pero a la larga trae muchos beneficios.

En el reconocimiento de formas aplicado a la visión artificial, se utilizan técnicas de reconocimiento geométrico de formas, como el aprendizaje supervisado (se conoce la clase a la que pertenece cada vector) en condiciones estadísticas o algoritmos de clasificación no supervisados o *clustering* y, además, en las redes neuronales, siendo estas últimas especialmente interesantes por su capacidad de aprendizaje y de adaptación.

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

Por tal motivo el sistema cuenta con una etapa que después de la captura hace el pre-procesado eliminando los ruidos captados por la cámara, en este caso, no interesa conservar la información del color, así que se aplica el filtrado en escala de grises perdiendo de esta manera las propiedades del color, y de esta manera el margen de error en la detección de características en los limites es menor. Cuando ya se dispone de la imagen capturada y filtrada, es necesario aislar o separar los objetos de interés de la escena. Se pretende por tanto dividir una imagen en diferentes regiones, o dicho de otra forma, detectar automáticamente los bordes entre los elementos o regiones.

Las operaciones de segmentación de una escena dependen de la misma y de la información que se busque dentro de la imagen.

## 4. 2 ANÁLISIS DEL PROYECTO

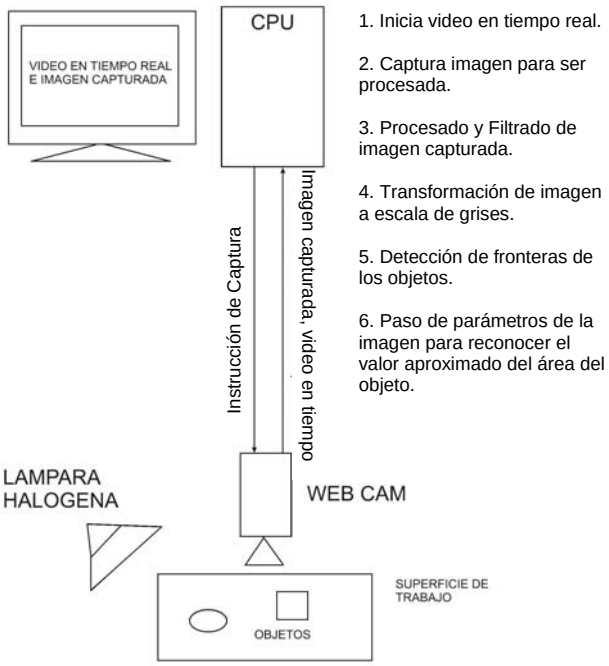



Fig. 4.1 Esquema general del proyecto[1]

[1] Diagrama elaborado por el Autor de la Tesis

### 4. 2. 1 Diseño y construcción de la etapa física

Esta etapa es muy importante pues es el punto de partida, ya que la obtención de imágenes de calidad depende, de los factores que se involucran en la captura, en los cuales cabe mencionar: el tipo de luz, posición de la luz, intensidad de luz, tipo de cámara, resolución de cámara, etc.

Se comenzara por detallar la superficie de trabajo, la cual debe contar con el aislamiento necesario para que la captura no se vea afectada por otras fuentes externas de luz.

En el proyecto se empleó una caja de cartón como contenedor, con las dimensiones que se que muestran en la figura 4.2, para evitar que el paso de la luz pudiera provocar ruido en las imágenes, se colocó una fuente de luz en la parte superior (tapa de arriba de la caja), que en este caso es una lámpara de halógeno, y también en esta parte superior de la caja se montó el dispositivo de captura de imágenes que para los fines del proyecto fue suficiente emplear una webcam convencional con conexión USB 2.0 y que soportaba una resolución máxima de 640x480 pixeles .

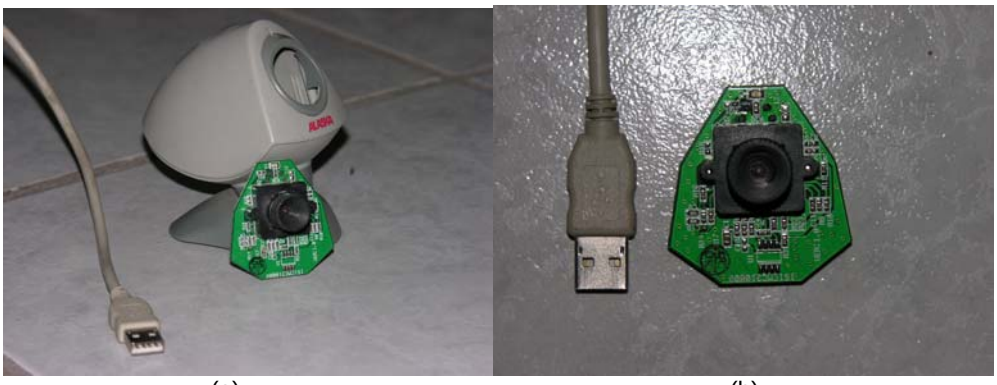

Fig. 4.2 Dimensiones del contenedor[2]

[2] Dimensiones del contenedor, Imagen del Autor de la Tesis 1_0807AVI.jpg

El interior del contenedor fue pintado de color blanco con la finalidad de que las paredes del contenedor reflejen la luz y así se aproveche más la iluminación.

Fig. 4.3 Interior del contenedor [3]

Para fijar correctamente la cámara Webcam, en la tapa de la caja, fue necesario sacar el circuito de la cámara de su carcasa original, como se muestra en la figura 4.4(a), y solamente se instaló dicho circuito que ya trae fija la cámara, como se puede apreciar en la figura 4.4(b).



(a) (b)

Fig. 4.4 (a) Webcam con carcasa y conector USB, (b) Circuito de la Webcam y conector USB [4]

[3] Interior de la caja de cartón pintada, Imagen del Autor de la Tesis 2_0807AVI.jpg
[4] Webcam,circuito y conector USB, Imagen del Autor de la Tesis 3_0807AVI.jpg

Para la iluminación adentro del espacio de trabajo, se empleó una lámpara halógena de 50w (figura 4.5), y se incorporó en la parte superior del contenedor aun lado del dispositivo de captura. Este tipo de lámpara tiene la capacidad de no esparcir la luz en todas las direcciones, sino que va orientada únicamente a un punto.

(a) (b)

Fig. 4.5 (a) Lámpara de halógeno con sus partes, (b) Lámpara de halógeno armada [5]

La tapa del contenedor fue adaptada con 2 orificios, para el ensamble de la lámpara de halógeno y del dispositivo de captura como se muestra en la figura 4.6(a); estando la cámara al centro y la fuente luminosa ligeramente desplazada, para evitar que la cámara sea deslumbrada por la luz halógena y así evitar que haya ruido en al momento de hacer las tomas de los objetos contenidos en el área de captura como se muestra en la figura 4.6 (b).

[5] Webcam,circuito y conector USB, Imagen del Autor de la Tesis 3_0807AVI.jpg

(a) (b)

Fig. 4.6(a) Vista de lejos de la tapa del contenedor, (b) Lámpara y webcam montadas en la tapa[6]

Hasta aquí la parte física del sistema de visión artificial quedó completa, restando solamente hacer los ajustes de la cámara y las primeras tomas desde el interior del contenido.

La figura 4.7 muestra como quedó el sistema ya completo visto desde el exterior.

Fig. 4.7 Parte física del Sistema de Visión Artificial [7]

[6] Tapa con lámpara y circuito de webcam, Imagen del Autor de la Tesis 4_0807AVI.jpg
[7] Parte física del Sistema de Visión completo, Imagen del Autor de la Tesis 5_0807AVI.jpg

Para los primeros ajustes del dispositivo de captura se emplearon objetos convencionales, sin importar de momento el color, la dimensión o la distancia entre ellos, esto únicamente se hizo para verificar la resolución de operación de la cámara y el buen funcionamiento y posición de la lámpara, así como determinar el mapa de color mas optimo para las condiciones de iluminación.

### 4. 2. 2 Etapa de Pre-Procesamiento

Para iniciar la evaluación de la configuración del dispositivo de entrada, es necesario previamente iniciar el entorno de Matlab y dar de alta un objeto de tipo video, empleando la siguiente línea de comando.

Vid= videoinput('winvideo')

En donde Vid es la variable de tipo video, y la palabra que esta entre comillas 'winvideo' es el tipo de dispositivo que se esta conectando al software, en este caso la webcam. Al introducir esta línea código en la ventana de comandos del Matlab, aparece la variable cargada en el espacio de trabajo como se muestra a continuación.

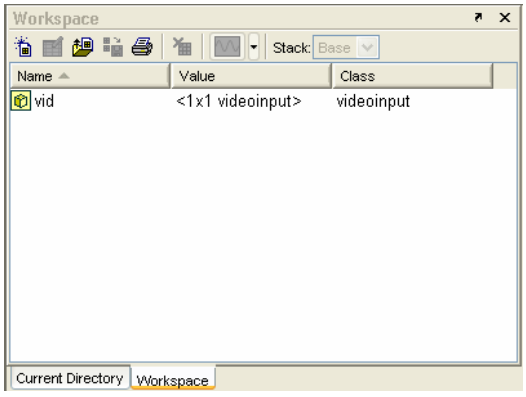



Fig. 4.8 Espacio de trabajo de Matlab[8]

[8] Programa Matlab

Enseguida es necesario ver la información del dispositivo de video para ver que valores posee por default, y así confirmar que son los indicados. Se empleo el comando dev_info = imaqhwinfo('winvideo',1) asignando a una variable la información del dispositivo de tipo winvideo y desplegar la información del dispositivo de entrada, el resultado de este comando fue el siguiente:

dev_info =

DefaultFormat: 'RGB24_352x288'
DeviceFileSupported: 0
DeviceName: 'ICM532A'
DeviceID: 1
ObjectConstructor: 'videoinput('winvideo', 1)'
SupportedFormats: {1x10 cell}

La resolución del dispositivo de captura es de 352X288, estas unidades están dadas en píxeles, con una calidad de color de 24 bits que esta basada en el modelo de color RGB. Otra resolución que admite la cámara es la 640X480 que emplea mas píxeles para mostrar la imagen pero en ocasiones esta resolución puede crear un poco de distorsión dependiendo en gran parte al tipo de iluminación con la que se cuente.

Una vez que se conocieron los valores con los cuales se esta trabajando, se hicieron las pruebas de video, para ver en tiempo real como son captados algunos objetos que se colocaron sobre el espacio de captura.

Para comenzar a captar en tiempo real lo que esta enfocado por la cámara se empleó el comando preview(), este comando, como se mencionó anteriormente, no comienza la captura de frames sino que únicamente da

una vista previa, esto es importante para poder dar la ubicación correcta al dispositivo de captura.

A continuación se muestra en la figura 4.8 la primera imagen captada por la cámara colocada en el espacio de captura e iluminada con la lámpara de halógeno.


Fig. 4.8 Imagen captada por la cámara[9]

En la figura 4.8 se puede ver el espacio captado por la webcam y los objetos colocados sobre este espacio de trabajo, después de la primera toma, hay ciertos aspectos a considerar como: si la intensidad de la luz es la adecuada, si la ubicación de la cámara con respecto a la fuente luminosa es la correcta, así como las proporciones captadas por el espacio de trabajo.

[9] Image Acquisition ToolBox de Matlab

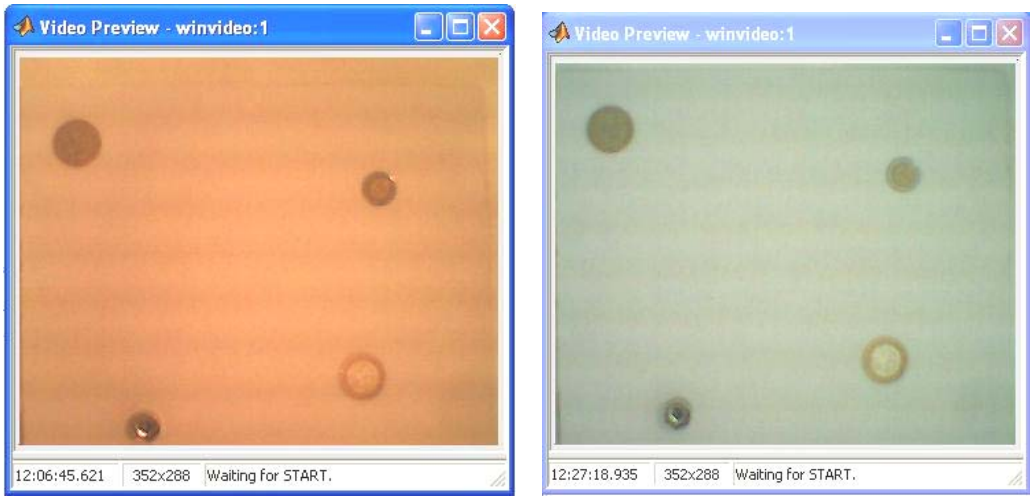
En la primera toma, se nota que la luz de la lámpara hizo que todo lo captado por la cámara se perdiera un poco, con respecto al color del fondo del contenedor. Este es uno de los primeros problemas que se presento en esta etapa de pre-procesamiento, por lo que se va a cambiar la lámpara de luz halógena, por una lámpara de luz blanca convencional, de esta manera tendremos un punto de comparación con respecto a la primera toma.


Fig. 4.9 Imagen captada por la camara con luz blanca[10]

Al comparar ambas imágenes tenemos como resultado que hay más nitidez en los objetos al emplear la luz blanca que con la luz halógena como se aprecia en la figura 4.9. Cabe mencionar que es posible procesar la imagen captada con luz amarilla para obtener los resultados óptimos, es decir la primera toma, mejorarla a través del procesamiento, pero esto

[10] Image Acquisition ToolBox de Matlab

provoca un trabajo extra y posteriormente puede hacer más laborioso el diseño del algoritmo de respuesta para el sistema de visión.




(a) (b)

Fig. 4.10 (a) Imagen Captada con Luz amarilla, (b) Imagen captada con luz blanca[11]

Otro aspecto importante fue la equivalencia entre píxeles y centímetros, que se determina según la distancia de la cámara con el espacio de captura, así como la magnitud de dicho espacio.

Para determinar la razón de píxeles con centímetros se realizaron los siguientes cálculos:

$$x = \frac{352\,pxl * 1cm}{9.2cm} = 38.2 \text{ pxl} \qquad y = \frac{288\,pxl * 1cm}{7.57cm} = 38.0 \text{ pxl}$$

$$\textit{Razón en Píxeles en X} = \frac{\text{Re}\,solucion\_en\_X * 1cm}{Magnitud\_en\_centímetros\_en\_X}$$

[11] Image Acquisition ToolBox de Matlab

$$\textit{Razón en Píxeles en Y} = \frac{\text{Re}\,solucion\_en\_Y * 1cm}{Magnitud\_en\_centímetros\_en\_Y}$$

La cámara alcanzaba a captar un cuadro de 9.2 cm x 7.57 cm, y en píxeles según la resolución de la cámara es 352 pxl x 288 pxl, al aplicar una regla de tres obtenemos el cálculos para las componentes en x y y, lo que nos da una equivalencia de que cada 38 píxeles de la imagen es igual a 1cm, de aquí se desprenden dos términos los cuales son: mundo real, que comprende todo lo que se percibe realmente en el espacio de captura es decir, las dimensiones en la realidad no son las mismas que en el mundo visual que es todo lo captado por el dispositivo de video y que desprende diferentes dimensiones. En este caso el cuadro captado por la cámara es de 9.2cm x 7.57cm y ese mismo cuadro medido en el espacio de captura es de 28cm x 25 cm.

Pero al momento de captar la imagen, debe ser considerada con la razón antes mencionada tomando como referencia la equivalencia de píxeles con respecto a los centímetros.

En la figura 4.11 se muestra la división de la imagen captada, cada cuadro tiene como medida 1cm o 38 píxeles.

$$x = \frac{352\,pxl * 1cm}{28cm} = 12.5714 \text{ pxl} \qquad y = \frac{288\,pxl * 1cm}{25cm} = 11.520 \text{ pxl}$$

Estos cálculos fueron realizados tomando en cuenta las dimensiones obtenidas del recuadro captado por la cámara, pero en el mundo real.

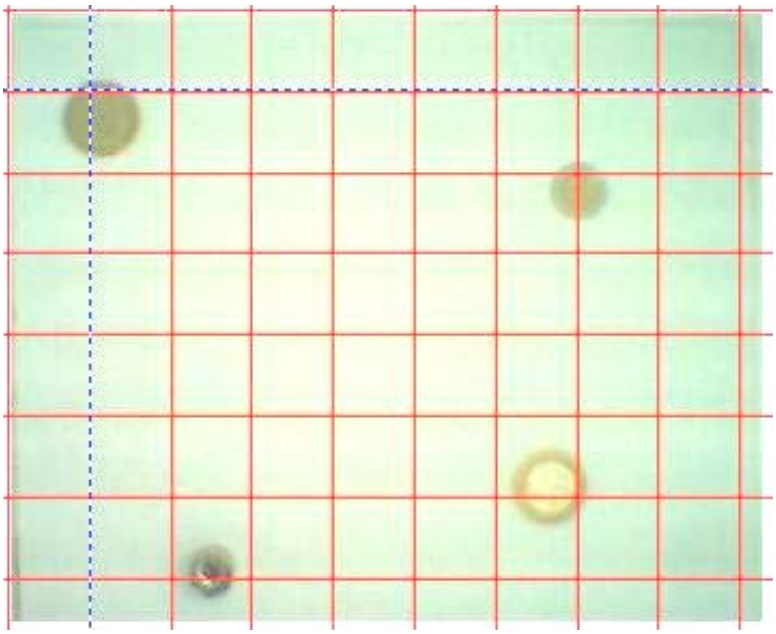


Fig. 4.11 Imagen captada por la cámara y dividida en centímetros[12]

En el mundo real la imagen captada tuvo una razón de que cada centímetro equivale 12.5714 píxeles, esto hace que el calculo tenga mas margen de error de un +-10%, debido a las decimales, por lo que fue mas conveniente adoptar la equivalencia de 1cm=38 pxl correspondiente al mundo visual.

Una vez que se hicieron las consideraciones respecto a la escala y resolución de la imagen captada, fue necesario tomar en cuenta el tiempo de espera entre cada imagen al momento de ser captada por la cámara, es decir los cuadros por minuto y de igual manera verificar el tamaño de memoria temporal o buffer, destinada para el almacenamiento de los cuadros captados en una secuencia de imágenes.

Para tener una referencia de la configuración por default y ver que valores se tenían que modificar se empleó el comando get() para enlistar

[12] [12] Image Acquisition ToolBox de Matlab

todas las características de configuración del objeto de video creado dentro del espacio de trabajo de Matlab, al aplicar dicho comando los resultados arrojados fueron los siguientes.

| General Settings: | Trigger Settings: | Acquisition Sources: |
|---|---|---|
| FramesAcquired = 0 | InitialTriggerTime = [] | SelectedSourceName =input1 |
| FramesAvailable = 0 | TriggerCondition = none | Source = [1x1 videosource] |
| FramesPerTrigger = 10 | TriggerFrameDelay = 0 | |
| Logging = off | TriggerRepeat = 0 | |
| LoggingMode = memory | TriggersExecuted = 0 | |
| ReturnedColorSpace = rgb | TriggerSource = none | |
| ROIPosition = [0 0 352 288] | TriggerType = immediate | |
| Running = off | | |
| Tag = | | |
| Timeout = 10 | | |
| Type = videoinput | | |
| UserData =[] | | |
| VideoFormat=RGB24_352x288 | | |
| VideoResolution = [352 288] | | |

Las características más relevantes son los frames por trigger o evento que por default son 10 frames por trigger, es decir que cada diez imágenes o cuadros captados se vacía la memoria buffer, esto es importante cuando se hacen capturas de objetos en movimiento. Para el proyecto fue conveniente ampliar el número de frames en el muestreo al comenzar la captura, debido a que 10 frames son captados en milisegundos, y es poco tiempo para realizar el respaldo de la imagen antes de ser borrada de la memoria buffer.

En la característica LoggingMode tiene asignado el valor memory es decir, los frames que son captados se depositan en la memoria temporal; otra opción es que se pueden guardar los frames un archivo de película o en varios archivos de imagen para hacer una secuencia.

Para configurar el número de frames que se van a almacenar en cada evento o tiempo de captura, se empleó el comando:

set(variable,'FramesPerTrigger',# de frames).

En el proyecto se consideraron 100 frame por evento hasta antes de que se saturara la memoria temporal y se disparara el trigger.

Set(vid,'FramesPerTrigger',100)

### 4. 2. 3 Etapa de Captura

Una vez que se configuró en número de Frames antes de cada trigger, se prosiguió a colocar objetos dentro del espacio de trabajo, que en este caso constaba de figuras geométricas bidimensionales.

El objetivo del sistema de visión artificial es captar los objetos contenidos dentro del espacio de trabajo y procesar las imágenes para obtener los contornos y traducir los resultados del procesamiento en una variable de matriz bidimensional.

Esta información contenida en la matriz, puede ser empleada para conocer las dimensiones, el número de objetos contenidos en el espacio de trabajo, el perímetro y área de cada figura, etc.

Para dar comienzo a la captura, es necesario inicializar un objeto de tipo video que este caso fue la variable *vid*, previamente cargada en el espacio de trabajo del entorno de Matlab.

Antes de que se iniciara lo que era propiamente el almacenamiento de frames en la memoria virtual, fue necesario dar una ultima vista a la posición de la webcam, empleando el comando preview(); esto se hizo con la finalidad de que alguno de los objetos no estuviera fuera del área de captura o que estuvieran demasiado cercanos a las fronteras del alcance de la cámara.

El resultado de la vista previa de la cámara se muestra en la siguiente figura.

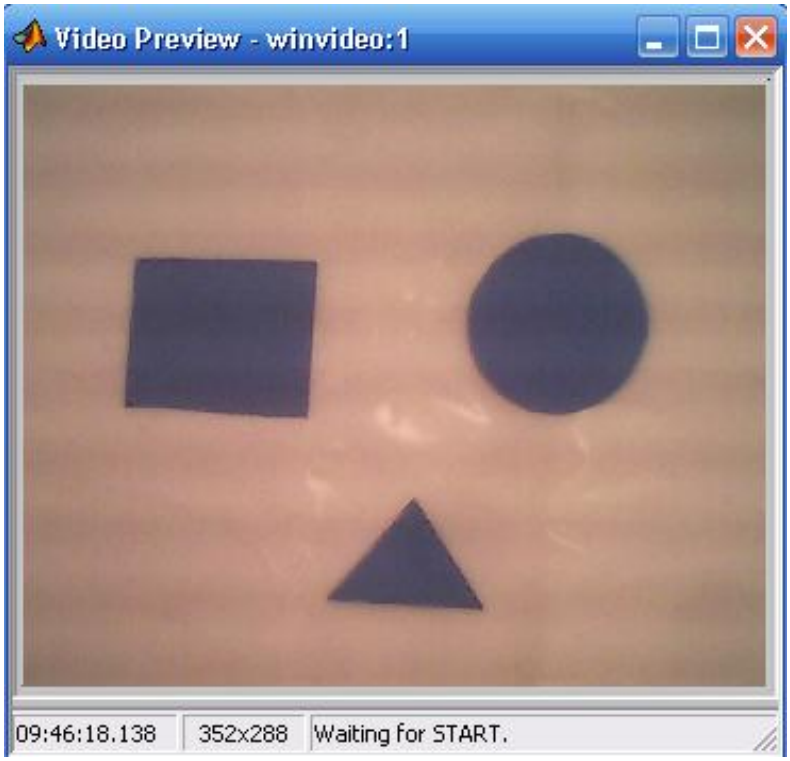



Fig. 4.11 Captura de Objetos en Espacio de trabajo[13]

Una vez que se colocaron los objetos dentro del espacio de captura, se prosiguió a capturar dichos objetos, en este caso como no son objetos que están en movimiento, solo fue necesario captar una sola vez los objetos

[13] Image Acquisition ToolBox de Matlab

enfocados por la cámara en determinado tiempo, una vez comenzada la captura.

Primero se inicializó un objeto de video "*vid*", para dar paso a que se almacenaran en la memoria los frames, esto se logró mediante el comando

Start(vid);

Una vez que se inicializó el objeto de video, fue necesario captar en determinado instante la imagen que estaba siendo recibida a través del dispositivo de video, para esto se empleó el comando getsnapshot, que asigna a una variable de tipo matricial la imagen más reciente que en el momento que se ejecuto la función, estaba alojada en la memoria.

frame=getsnapshot(vid);

Al ejecutar la función inmediatamente la variable frame adoptó el valor matricial de la imagen en tres dimensiones y en espacio de trabajo apareció la variable ya cargada con el valor, como se muestra en la siguiente figura.

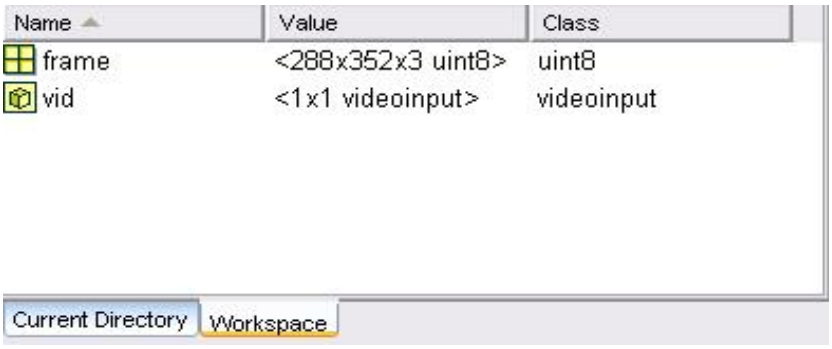



Fig. 4.12 Variables en Espacio [14]de trabajo de Matlab

[14] Espacio de trabajo de Matlab

Una vez cargada la variable frame con el valor de la imagen captada por la cámara, fue necesario guardar el valor de la variable frame en un archivo de imagen, con la finalidad de facilitar el siguiente paso que fue el de procesamiento.

Para guardar una variable cargada en el espacio de trabajo en un archivo de imagen se emplea la función imwrite, en este caso se empleó de la siguiente manera.

```
Imwrite(frame,'c:\captura1.jpg');
```

En la función se especifica el nombre de variable que contiene la información y el nombre con la ubicación que va a tener el archivo, así como el formato que se va a emplear para la imagen, que en este caso se empleó el formato jpg.

Una vez que se ejecutó la función, se creó un archivo de imagen JPG que facilitó la etapa del procesamiento, y de esta manera se pudieron limpiar los frames que estaban alojados en la memoria.

### 4. 2. 4 Etapa de Procesamiento

En esta etapa se emplearon varios pasos para llegar al resultado deseado, y así haber obtenido los parámetros necesarios.

Una vez se que obtuvo la captura del dispositivo de entrada en un archivo de imagen JPG, se prosiguió a cargar dicha imagen a una variable dentro del espacio de trabajo de Matlab para que fuera tratada mediante los filtros correspondientes.

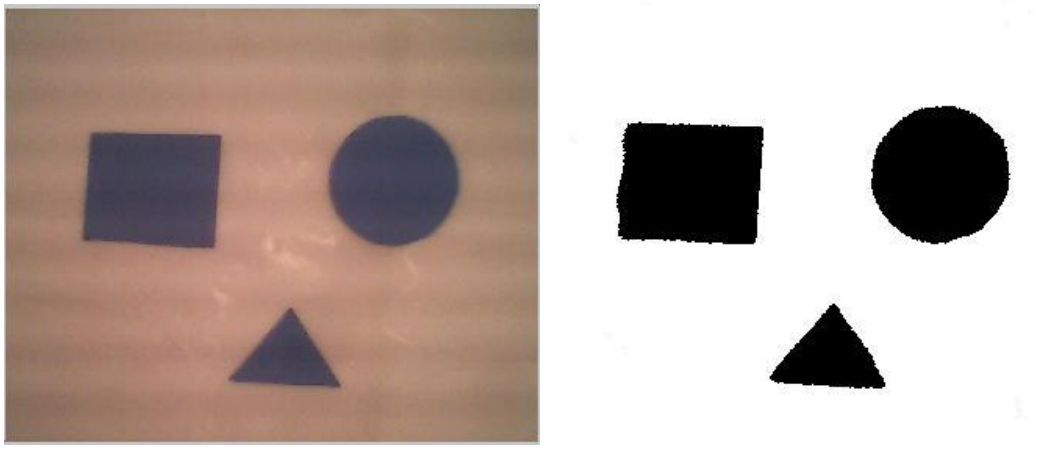
Para ellos se empleo el comando imread y así lograr cargar la imagen que se había guardado en la etapa de captura al entorno de Matlab.

Imagen=imread('c:\captura1.jpg')

Una vez que se cargó la imagen en el espacio de trabajo, se prosiguió a modificar la imagen en su mapa de color, y pasarlo únicamente a la escala de grises para facilitar así la diferenciación de los objetos con respecto al fondo.

Mediante la función "bw" se pierde la información de color y únicamente se conservan las escalas de grises, al aplicarle a la imagen dicha función se obtuvo el siguiente resultado.

Imagen2=bw(Imagen);



(a) (b)

Fig. 4.13 Imagen Original captada por la webcam, (b) Imagen Procesada en escala de Grises[15]

[15] Image Acquisition ToolBox de Matlab

Una vez que se aplico la función “bw” y se cargo a la variable Imagen2, se obtuvo la imagen con fondo blanco y los objetos rellenados de color negro, esto facilitó en gran manera la detección de fronteras ya que únicamente se limito 2 colores.

Para la detección de fronteras se empleo la función EDGE que como se vio a lo largo del capítulo 2, tiene como función extraer únicamente los bordes de los objetos en una imagen y pasar dicha imagen a una imagen vectorial.

En este caso se empleo el método Sobel, para la extracción de las fronteras y el resultado se guardó en una variable matricial denominada ImgRes, al aplicar la función EDGE se obtuvieron lo siguiente resultados en la imagen:

```
ImgRes=Edge(Imagen2,’Sobel’);
```

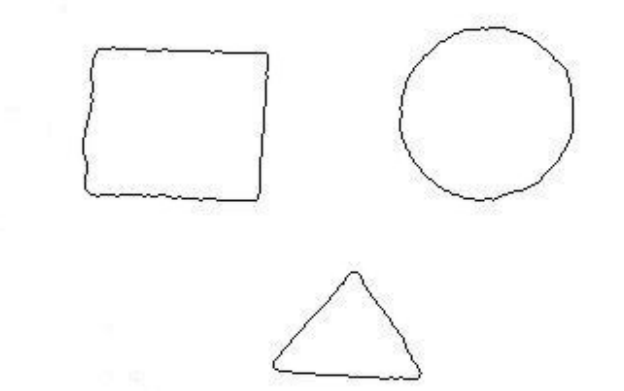


Fig. 4.14 Imagen Procesada mediante la función EDGE[16]

[16] Image Acquisition ToolBox de Matlab

La matriz obtenida en la variable ImgRes, ya puede ser manipulada para conocer aspectos mediante el uso de píxeles en 1 y 0, y así poder conocer características tales como la distancia entre los objetos, el perímetro, el área, etc.

El proyecto que se propuso en esta tesis, pretende dar las bases para obtener los parámetros previos a la elaboración de un algoritmo y como facilitar el diseño y la elaboración de los mismos mediante una buena iluminación, configuración y demás aspectos a considerar en el montaje del proyecto.

## 4. 3 APLICACIONES DEL PROYECTO

Los principios básicos del proyecto descrito, actualmente tienen múltiples aplicaciones en el ramo de la investigación tales como la biología, aeronáutica, automatización, robótica entre otros a continuación se presentan algunos ejemplos de Sistemas de visión artificial en las que se emplea en procesamiento y filtrado de imágenes.

1.- Análisis de imágenes médicas: En las aplicaciones medicas, dentro del diagnostico se emplea para procesar imágenes como puede ser una radiografía, ultrasonido, etc., y así percibir ciertos aspecto que no son visibles por el ojo humano. Como por ejemplo fisuras en los huesos en una radiografía

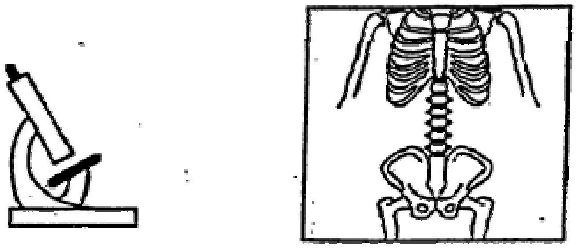


Fig. 4.15 SVA en la Medicina en diagnósticos[17]

[17] http://www.inteligenciaartificial.cl/ciencia/software/ia/reconocimiento_de_imagens.htm

2.- Interpretación de fotografía aérea: Los SVA en la actualidad se emplean en las tomas aéreas ya sea desde un avión o incluso desde las tomas realizadas por los satélites, con las finalidad de estudiar principalmente la estructuras en la tierra como son las fallas geológicas, las montañas, etc.

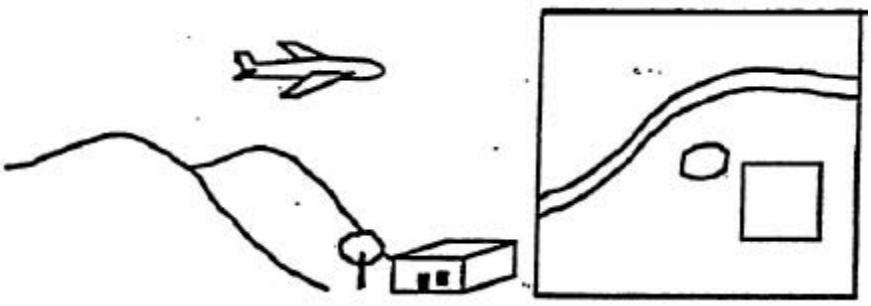


Fig. 4.16 SVA en las tomas Aéreas[18]

3.-Exploración y movilidad: Los SVA en la expiración se emplea en las imágenes que despliegan los radares y así tener una mejor visualización del territorio.

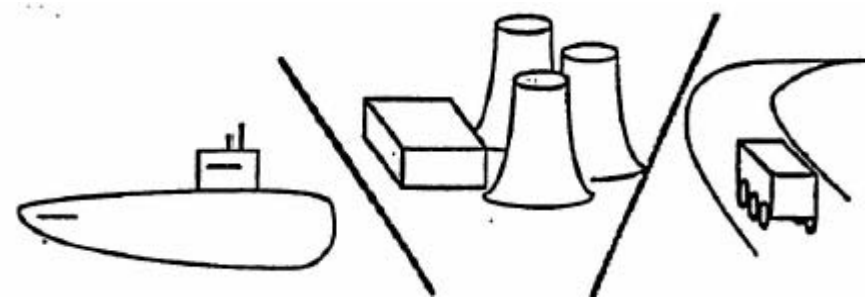


Fig. 4.17 SVA en la expiración[19]

[18] http://www.inteligenciaartificial.cl/ciencia/software/ia/reconocimiento_de_imagens.htm
[19] http://www.inteligenciaartificial.cl/ciencia/software/ia/reconocimiento_de_imagens.htm

4.- Manejo de materiales: Los robots en muchas industrias actualmente actúan mediante la Visión Artificial, ya que anteriormente las tareas las llevaban acabo, guiados por sensores que en muchas de las ocasiones eran limitados al desempeñar las tareas.

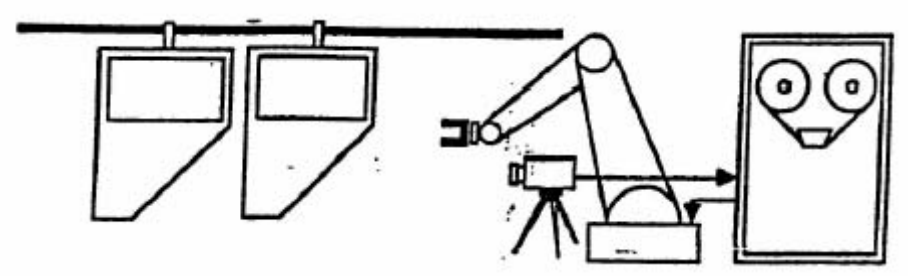


Fig. 4.18 SVA en el manejo de Materiales o Producto[20]

5.- Inspección: Actualmente hay muchos robots que dentro de la industria se guían mediante imágenes, para seleccionar de acuerdo a patrones previamente establecidos, aquellos materiales que son de calidad y aquellos que no.

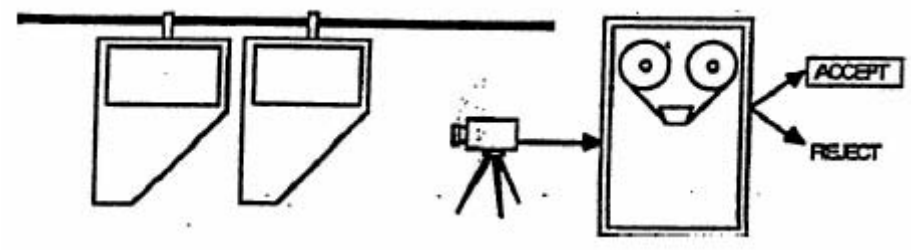



Fig 4.19 SVA en el Control de Calidad[21]

[20] http://www.inteligenciaartificial.cl/ciencia/software/ia/reconocimiento_de_imagenes.htm
[21] http://www.inteligenciaartificial.cl/ciencia/software/ia/reconocimiento_de_imagenes.htm

6.- Ensamblaje: Los SVA en el modelado de acuerdo a una imagen captada, tratan de reproducir exactamente lo que se capturo mediante la cámara de video.

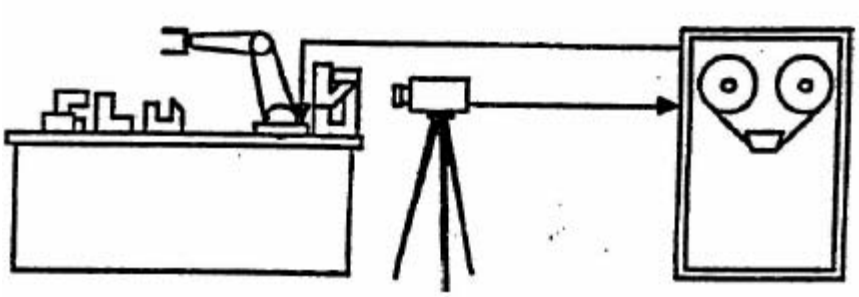


Fig. 4.20 SVA en el maquinado o reproducción de piezas[22]

7.- Navegación: En la actualidad existen robots que se guían dentro de la industria mediante imágenes, como por ejemplo seguir una línea de determinado color y es una manera de ser autónomo.

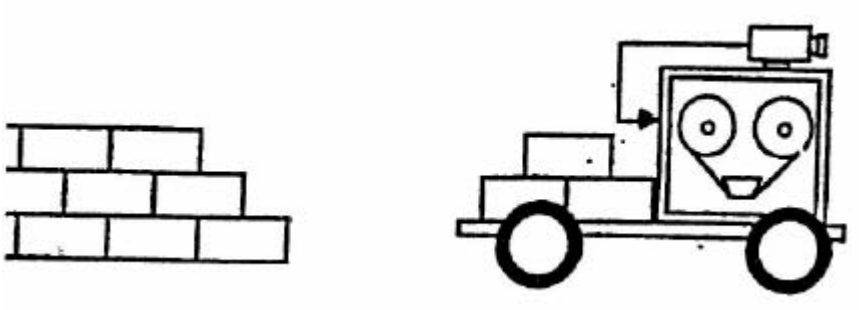


Fig. 4.21 Manejo mediante imágenes.[23]

[22] http://www.inteligenciaartificial.cl/ciencia/software/ia/reconocimiento_de_imagenes.htm
[23] http://www.inteligenciaartificial.cl/ciencia/software/ia/reconocimiento_de_imagenes.htm

# CONCLUSIONES

PRIMERA: El procesamiento y el filtrado de imágenes dentro de los Sistemas de Visión Artificial, juega un papel de gran importancia, ya que depende en gran parte de su complejidad o simplicidad, es decir, entre más claros sean los parámetros obtenidos del procesamiento, más sencillo será el algoritmo que determine el comportamiento del Sistema.

SEGUNDA: El manejo de un sistema inteligente, es mas completo si se emplea la visión, ya que da más libertad de manipular los datos de entrada, a diferencia de los sistemas que únicamente hacen uso de sensores de proximidad, calor, humedad, luminosidad, etc. Esto se pudo comprobar al diseñar el SVA de segundo nivel, pues al obtener la matriz de píxeles, fue posible determinar la información de la que se desprende, la distancia entre cada objeto, las dimensiones de cada objeto incluso hacer comparaciones entre los objetos, y todo eso con una simple imagen, algo que difícilmente se lograría con un sistema basado en sensores.

TERCERA: En la actualidad existen muchas opciones para poder implementar un SVA algunas de ellas muy costosas, pero, la propuesta que se hizo en el proyecto de esta tesis, tenía como finalidad llevar acabo un sistema de visión artificial complejo de bajo costo mediante el uso de la computadora y una webcam convencional, y se comprobó la efectividad de la aplicación, al

realizar una práctica de procesamiento de imágenes con un sistema de segundo nivel.

CUARTA: Día a día son cada vez más las aplicaciones .en las diferentes áreas de estudio, y después del estudio realizado se pudo concluir que la Visión Artificial es el futuro de los sistemas autómatas, es decir aquellos que pretenden imitar el comportamiento humano, así como también en las áreas científicas importantes entre las cuales cabe mencionar: Medicina, Geología, Astronomía, etc.

QUINTA: En México aún se está atrasado con respecto a la implementación de Sistemas de Visión Inteligentes, ya que solamente en algunas ciudades se cuenta con pequeños prototipos y hay poca documentación científica de investigación en nuestro país.

SEXTA: Hay aspectos de suma importancia en los SVA, entre los cuales cabe destacar, la iluminación, ya que una buena iluminación ahorra mucho trabajo al momento de procesar, y de esta manera se garantiza que al momento de capturarse una imagen la misma luz no produzca sombras que son como ruido al momento de la aplicación de algún filtro.

SÉPTIMA: Se pudo comprobar que Matlab es una de las mejores opciones al manejar imágenes para la obtención de parámetros, ya que ofrece todos los elementos necesarios para capturar, procesar y manipular todo tipo de imágenes, todo en un solo programa; también se pudo concluir que no es un paquete complicado, ya que incluye una ayuda con ejemplos e información completa de cada una de las funciones.

OCTAVA: La visión artificial es sumamente difícil, ya que una tarea que parece relativamente fácil para los humanos es compleja de llevar a cabo para las computadoras debido a todas las etapas necesarias para lograr que el sistema entienda una imagen.

NOVENA: La visión artificial es una herramienta poderosa en automatización, inspección y control de calidad que, aunque en ningún momento iguala la capacidad de la visión humana, garantiza la calidad de los productos en los procesos productivos e industriales.

DÉCIMA: El desarrollo de aplicaciones a través de interfases, es posible por las facilidades que ofrece Matlab para establecer conectividad con lenguajes de programación tales como Visual Basic, C++ e incluso ensamblador, esto se pudo comprobar mediante la aplicación desarrollada en Visual Basic.NET que determinaba el perímetro de las figuras que se encontraban dentro del espacio de captura.

## BIBLIOGRAFÍA

K.S. FU, *ROBOTICA: control, detección, visión e inteligencia,* Ed. McGraw-Hill, México, D.F., 2001. pp. 599.

## OTRAS FUENTES

http://www.diinf.usach.cl/~dmery/imagenes/proc.htm

www.depeca.uah.es/docencia/doctorado/cursos04_05/82854/docus/CursoVision10.pdf

http://www.ual.es/~fguindos/VA.html

http://www.diinf.usach.cl/~dmery/imagenes/proc28.htm

http://www.mathworks.com/access/helpdesk/help/toolbox/images/images.shtml

http://www.inteligenciaartificial.cl/ciencia/software/ia/reconocimiento_de_imagenes.htm

http://www-dbv.informatik.uni-bonn.de/abstracts/will.icip00.htm

http://www.cs.mu.oz.au/research/vislab/

http://www.eeng.dcu.ie/~whelanp/vsg/outline.html

http://cvpr2000.cs.uiuc.edu/

http://www.cs.cmu.edu/

http://www.cs.cmu.edu/~cil/vision.html

http://eia.udg.es/index.shtml.es

http://www.mathworks.com