

# UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería

***Robot Explorador***

***Módulo:***

***Visión por Computadora***

***Procesamiento y Análisis de Imágenes***

Por

Stefanie María del Rosario

Castillo González

Trabajo de graduación presentado para optar al grado  
académico de licenciada en Ingeniería Electrónica

Guatemala, 2008



***Robot Explorador***

***Módulo:***

***Visión por Computadora***

***Procesamiento y Análisis de Imágenes***

# UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería

***Robot Explorador***

***Módulo:***

***Visión por Computadora***

***Procesamiento y Análisis de Imágenes***

Por

Stefanie María del Rosario

Castillo González

Trabajo de graduación presentado para optar al grado  
académico de licenciada en Ingeniería Electrónica

Guatemala, 2008

## PREFACIO

El 4 de julio de 1997 la NASA aterrizó el primer vehículo explorador en Marte, conocido como *Sojourner*. La misión del Mars Pathfinder fue la primera misión en aterrizar en Marte un vehículo explorador autónomo y tenía el objetivo de explorarlo a un menor costo. El diseño e implementación de robots exploradores tuvo gran auge desde esa fecha. En 2004 se envió otro robot explorador, llamado *Spirit*, el cual recorrió la superficie Marciana y logró contestar muchas preguntas no resueltas acerca de la historia del planeta rojo. Estos vehículos exploradores tenían un computador a bordo que controlaba todos sus movimientos. Éste empleaba un microprocesador Rad 6000 de 32 bits, operaban a una velocidad de 20 millones de instrucciones por segundo y poseía una memoria, programable en C, de 1 gigabyte.

Dada la gran distancia a Marte, no es posible controlar el robot desde la tierra por lo que es necesario que pueda controlarse de forma autónoma. El control del robot fue realizado por la computadora que llevaba a bordo. Ésta procesaba la información que le era retroalimentada por sensores e imágenes. Estas imágenes eran provistas tanto de un satélite en órbita, como de cámaras colocadas sobre el robot, que simulaban vista estereoscópica.

Esta fue la inspiración que tuvieron los estudiantes de 5to año de electrónica en 2005 para presentar el proyecto de graduación llamado “Diseño y construcción de un sistema explorador robotizado y autárquico”. El robot construido, propiedad de la Universidad del Valle de Guatemala, es llamado “*Rover*”. El sistema de control del *Rover* se basaba en el procesamiento de la información provista por sensores colocados sobre el robot e imágenes provistas por una cámara que simulaba ser un satélite. Dicha cámara estaba colocada sobre el terreno sobre el cual se movería el robot.

El proyecto “Robot explorador” nació con el afán de dar seguimiento a este trabajo, sin embargo se decidió darle otro enfoque al robot, dado que Guatemala es un país que no tiene la capacidad de enviar robots a explorar planetas vecinos. Hay otras necesidades que atender antes de pensar en explorar otros planetas.

El problema de la gran cantidad de basura que se acumula en lugares públicos es real. Después de un feriado o fechas festivas, las playas y demás lugares públicos, terminan repletos de basura. El guatemalteco, en general, no tiene la cultura de preservar el medio ambiente.

Es por esto que se pensó cambiar el objetivo general del robot explorador “*Rover*” y darle una aplicación que resuelve una necesidad real del país. La nueva aplicación del robot es un *robot limpia playas*, cuyo objetivo será limpiar las playas Guatemaltecas de manera autónoma. El robot podrá ser colocado sobre una playa y este recogerá la basura que se encuentre sobre ella.

Las funciones básicas son las mismas, el robot es autómata, el sistema de control y visión retroalimenta al sistema operativo para que este procese dicha información y tome decisiones respecto a los movimientos robot. Adicionalmente con los nuevos módulos integrados al robot: los paneles solares y brazo robótico, permitirán que el robot pueda funcionar sin gran consumo de energía y pueda recolectar la basura. El descrito anteriormente es el proyecto de graduación “*Robot explorador*” presentado en 2008 por cuatro estudiantes para optar al grado académico de licenciados en Ing. Electrónica.

# CONTENIDO

	Página
Lista de figuras .....	i
Lista de tablas .....	iii
Resumen .....	iv
<i>CAPITULOS</i>	
I. Introducción .....	1
A. Antecedentes .....	2
i. Visión por Computadora .....	2
ii. Robots Limpiadores de Playas .....	8
iii. Megaproyecto UVG “ROVER” 2005 .....	9
a. Sub-módulo: Procesamiento de imágenes (Mansilla, 2005)....	10
b. Sub-módulo: Control a través de internet (Mijangos, 2005)....	11
B. Planteamiento del Problema .....	12
II. Objetivos .....	13
III. Marco Teórico .....	15
A. Cámaras de Red.....	15
i. Axis 205 .....	16
ii. AxisMediaControl.dll .....	17
B. Redes WLAN .....	18
i. Controlador de Ethernet: ENC28J60 .....	20
ii. MikroC: SPI Ethernet Library .....	22
iii. WinSock .....	23
C. Formatos de imagen .....	23
i. Formato Vector .....	23
ii. Formato Raster .....	24
a. JPEG .....	25
b. BMP .....	32
D. Librería vic32.dll .....	36

E.	Localización y Reconocimiento de objetos en imágenes .....	37
i.	Filtros digitales .....	38
a.	Filtros Pasa baja .....	39
1)	Filtro de Mediana .....	40
b.	Filtro Pasa alta .....	40
F.	Aproximación de distancia de objetos en imágenes .....	41
i.	Modelo Pinhole .....	41
IV.	Delimitación del Tema .....	44
V.	Diseño Experimental y Resultados .....	47
1.	Evaluación del funcionamiento actual del sistema de visión por computadora y del sistema de control a través de internet .....	49
1.1	Diseño Experimental .....	49
1.2	Resultados .....	49
2.	Manejo técnico de la cámara de red Axis 205 .....	51
2.1	Diseño Experimental .....	51
2.2	Resultados .....	52
3.	Acceso remoto a la cámara de red a través de una WLAN:	
AxisMediaControll.dll	.....	53
3.1	Diseño Experimental .....	53
3.2	Resultados .....	54
4.	Elección de formato en el que se procesaría la imagen: JPEG, BMP .....	54
4.1	Diseño Experimental .....	54
4.2	Resultados .....	54
5.	Transformación de una imagen de JPEG a BMP: Vic32.dll .....	55
5.1	Diseño Experimental .....	55
5.2	Resultados .....	55
6.	Lectura del archivo binario de una imagen en formato BMP: rover.dll .....	56
6.1	Diseño Experimental .....	56
6.2	Resultados .....	57
7.	Localización de obstáculos y eliminación de ruido en la imagen .....	57
7.1	Diseño Experimental .....	57
7.2	Resultados .....	58



8. Aproximación de la distancia entre la cámara y un objeto en la imagen .....	60
8.1 Diseño Experimental .....	60
8.2 Resultados .....	60
9. Comunicación Inalámbrica entre el robot y la estación central: IEEE 802.3 ..	63
9.1 Diseño Experimental .....	63
9.2 Resultados .....	63
10. Protocolo de comunicación entre estación central y PIC maestro .....	64
10.1 Diseño Experimental .....	64
10.2 Resultados .....	65
11. Definición de objetos a recolectar: basura removible .....	70
11.1 Diseño Experimental .....	70
11.2 Resultados .....	71
12. Algoritmo de búsqueda del Rover .....	72
12.1 Diseño Experimental .....	72
12.2 Resultados .....	72
13. Pruebas finales: Integración de módulos .....	73
13.1 Diseño Experimental .....	73
13.2 Resultados .....	73
VI. Discusión .....	74
VII. Conclusiones y Recomendaciones .....	81
Bibliografía .....	vii
Apéndice .....	viii
Glosario .....	ix

## LISTA DE FIGURAS

<i>Figura</i>	<i>Página</i>
1. Proceso en el que una imagen es almacenada digitalmente .....	3
2. Huella dactilar, FBI utiliza técnicas de visión por computadora .....	4
3. Reconocimiento de matrículas, una aplicación de visión por computadora..	5
4. FP Developments utiliza un sistema de multivisión .....	7
5. Personas limpiando el Puerto de San José .....	9
6. Rover UVG, vehículo creado en UVG en 2005 .....	9
7. Rover Sojourner, vehículo creado por NASA .....	11
8. Cámara de Red: Axis 205 .....	17
9. Empaquetado del ENC28J60 ` .....	21
10. Conexiones externas requeridas por el ENC28J60 .....	22
11. Fortalezas y deficiencias de Raster y Vector .....	23
12. Imagen en formato <i>vector</i> .....	24
13. Imagen en formato <i>raster</i> .....	25
14. Ecuaciones para conversión entre los modelos de color RGB y YUV .....	26
15. Diagrama de bloques de la compresión de imágenes raster: JPEG .....	27
16. Ejemplo de compresión de <i>down sampling</i> en una imagen .....	28
17. Matriz de la transformada discreta de cosenos .....	29
18. Matriz de la componente Y de una fotografía en formato YUV .....	30
19. Matriz después de aplicada la transformada discreta de cosenos .....	30
20. Matriz de cuantización de componente Y en el estándar JPEG .....	31
21. Matriz de cuantización de componentes Cr, Cb en el estándar JPEG .....	31
22. Matriz cuantizada de la transformada de cosenos .....	32
23. Estructura de un archivo BMP .....	32
24. Estructura del File header .....	33
25. Estructura del Info Header .....	34
26. Ordenamiento de los colores en una estructura de 16 bits por pixel .....	35
27. Cambios suaves y bruscos .....	38
28. Aplicación de filtros digitales .....	38

29. Ejemplos de frecuencia baja y alta en una imagen .....	39
30. Efecto de distintas máscaras sobre imágenes efecto de borroso .....	39
31. Efecto de filtro mediana sobre una imagen .....	40
32. Efecto de filtro pasa alta sobre una imagen .....	41
33. Modelo Pinhole para una cámara .....	42
34. Relaciones de triángulos en un modelo Pinhole .....	42
35. Relación de Triángulos en modelo pin-Hole .....	43
36. Imagen enviada por la cámara en formato JPEG (test.jpg) .....	56
37. Imagen transformada a formato BMP (test.bmp) .....	56
38. Imagen después de ser aplicado el filtro BlueYellowScale (blueyellf.bmp). .....	58
39. Imagen después de ser aplicado el filtro de mediana .....	59
40. Conjunto de fotografías que permiten la medición de distancia .....	61
41. Conjunto de figuras que permiten la determinación de la distancia del objeto en la imagen` .....	62
42. Áreas por las cuales se escanea la fotografía .....	63

## LISTA DE TABLAS

<i>Tabla</i>	<i>Página</i>
1. Características relevantes cámara Axis 205 .....	17
2. Requerimientos de la máquina donde se ejecute AMC .....	18
3. Código enviados por el PIC maestro hacia la estación central .....	65
4. Códigos enviados desde la estación central hacia el PIC maestro .....	66

## RESUMEN

El proyecto realizado en 2005 por los estudiantes de 5to año de ingeniería electrónica, llamado “Diseño y construcción de un sistema explorador robotizado y autárquico”, es el que se continúa ahora en 2008 con el megaproyecto llamado *Robot Explorador*.

En 2005 se desarrolló un robot explorador de Marte construyendo un vehículo, implementando una estación remota, controlándolo a través de Internet y por un sistema de inteligencia artificial que le permite al robot tomar decisiones. Ahora en 2008, el presente Megaproyecto, *Robot Explorador*, pretende realizar mejoras a los sistemas de control y visión por computadora del robot, añadir dos nuevos módulos, Energía alternativa por celdas solares y Brazo Robótico, y darle una nueva aplicación al robot explorador la cual es *Robot Limpia playas*.

El presente informe corresponde al módulo de visión por computadora del robot y se concentra en el procesamiento y análisis de imágenes, provistas por una cámara colocada sobre el robot, para determinar la trayectoria del mismo y a la vez controlar al brazo robótico para recolectar la basura; la transmisión de datos inalámbricamente por la red WLAN de la estación central hacia el robot y viceversa; y la implementación de la interfaz de usuario en la estación central.

En específico, el módulo de visión por computadora debió transmitir del robot hacia la estación central una imagen tomada por la cámara de red colocada sobre el mismo, esto inalámbricamente a través de una WLAN. La imagen es procesada para localizar obstáculos en ella y la información obtenida de dicho procesamiento es retroalimentada al robot, vía la misma red WLAN, hacia el microcontrolador (PIC) de telecomunicaciones colocado sobre el robot, que a su vez traslada la información hacia el microcontrolador maestro del Rover. El robot cambia sus parámetros y trayectoria, según lo indiquen los resultados del procesamiento de imágenes. Además, se implementó un método para poder medir la distancia entre el robot y la *basura* con el fin de que el robot se aproxime a ella y ordene al brazo que recoja el objeto, asegurando cierta proximidad entre el brazo y la basura. Finalmente, se diseñó e implementó un algoritmo que determina el comportamiento de robot, permitiéndole buscar y recolectar basura en un ambiente tipo laboratorio.

# Capítulo 1

## Introducción

*“Mil rutas se apartan del fin elegido, pero hay una que llega a él: la perseverancia”*

*Michel E. de Montaigne*

---

Un robot autómatas es aquel que puede realizar la tarea para el cual fue diseñado sin ayuda humana. Es así como la *ingeniería electrónica* y mecánica pretende relevar al humano de tareas triviales y/o repetitivas, dejando que éste se ocupe de tareas que realmente exigen y requieren toda la capacidad de su inteligencia.

La institución que desarrolla con la más alta tecnología *robots autómatas exploradores* en América, es la NASA, institución gubernamental de E.E.U.U. responsable de los programas espaciales de dicho país. La NASA implementa, en conjunto con universidades estadounidenses de gran prestigio, robots exploradores autómatas, llamados *Rover*, los cuales son enviados a explorar planetas vecinos. Estos robots son parte de la clase Discovery de la NASA, que pretende fomentar nuevos métodos de exploración de planetas vecinos. El sistema de control diseñado para estos robots les permite, sin ayuda humana, entrar a la atmósfera del planeta destino, descender a través de ella, aterrizar exitosamente en el planeta, explorarlo y enviar información de interés a los científicos en la

tierra, utilizando transmisión satelital. No cabe duda que esta institución implementa robots exploradores con la más alta tecnología existente.

En 2005 los estudiantes de quinto año de Ingeniería Electrónica de la Universidad del Valle, notaron la importancia de fomentar el desarrollo de tecnologías en el país y optaron por presentar como trabajo de graduación, el diseño e implementación de un robot explorador que imita a los robots *Rover* que envía la NASA a explorar planetas vecinos. El proyecto fue seguido de cerca por el diseñador de los exploradores de la NASA e hizo notar su sorpresa y aprobación al observar el excelente trabajo realizado por los compañeros graduados en 2005.

El presente proyecto retoma el trabajo realizado en 2005, con el fin de adentrarse más en el campo de los sistemas robotizados autárquicos. El Megaproyecto con el nombre de *Robot explorador* propone: *el diseño e implementación del sistema de control del robot autónoma Rover*, propiedad de la Universidad del Valle de Guatemala, para la exploración de playas y recolección de basura en ellas. Este es un proyecto retador que involucra, principalmente la *ingeniería de inteligencia artificial* y la *ingeniería de control*, con el fin de implementar un sistema de control para el robot autónoma. Esta no es una tarea fácil, y se requiere de conocimientos de la electrónica, mecánica, procesamiento digital de señales, microcontroladores y teoría de control.

### *A. Antecedentes*

A continuación se presenta la información existente previo al desarrollo del presente trabajo de graduación, acerca de los temas involucrados: *Visión por computadora*, *Robots limpia playas* y del robot *Rover diseñado en 2005* en UVG.

#### *i. Visión por Computadora*

El control de sistemas y procesos conlleva: el obtener información del medio mediante sensores, el procesamiento y análisis de la información provista por los mismos y el dar una respuesta al medio, mediante algún actuador del sistema. La disciplina denominada ‘Visión por computadora’ es aquella que tiene como objetivo el control de

sistemas y/o procesos mediante el procesamiento de imágenes digitales provistas por una cámara. (20)

Para analizar dichas imágenes, es necesario que éstas estén representadas en un lenguaje binario. En su más simple descripción, una imagen digital no es más que una larga cadena de unos y ceros que representan a los elementos discretizados que conforman a la imagen. El instrumento (cámara) tiene una serie de lentes que enfocan la luz sobre una lámina de semiconductores. Una manera simple de entender esto es pensar en ellos como un arreglo bidimensional de celdas solares. Estos sensores convierten las distintas longitudes de onda de la luz en carga eléctrica, para luego traducir la magnitud de dicha carga a un número binario mediante un convertidor análogo-digital. Es así como se obtiene una representación binaria de una imagen del mundo real. Véase una descripción más detallada del proceso en la figura 1. (21)



**Figura 1 - Proceso en el que una imagen es almacenada digitalmente (1)**



La importancia en la actualidad de los sistemas que implementan *visión por computadora* para el control de alguna parte del sistema, es notable al nombrar algunas de las más relevantes aplicaciones de la disciplina. Estas se detallan a continuación:

- 1) *Restauración de imágenes*: El procesamiento de imágenes que tiene como objetivo eliminar o corregir píxeles que no pertenezcan a ella, basándose en la comparación de un píxel con los píxeles aledaños a él. (22)
- 2) *Reconocimiento de patrones*: Es el análisis de imágenes, dirigido al reconocimiento de patrones. El ejemplo más claro está aplicado a criminalística, las autoridades generalmente necesitan reconocer a personas y/o objetos, que cumplen ciertas características, dentro de una enorme base de datos de imágenes. Se desarrollan programas a los que se les provee una foto muestra del objeto, la cual posee ciertos patrones o rasgos característicos, para que este busque compatibilidades con las fotografías que almacena su base de datos. Otro ejemplo es el reconocimiento de caracteres, OCR (*optical character recognition*) proceso que se realiza para reconocer caracteres, como letras, números o símbolos, dentro de una imagen. Esto es tan común que ya no se cae en cuenta de este tipo de procesamiento en los documentos que se trabajan cotidianamente, por ejemplo las aplicaciones PDF, scanners y otros editores de texto realizan este tipo de análisis de imagen. (22)



**Figura 2 - Huella dactilar, FBI utiliza técnicas de visión por computadora para reconocer similitudes de patrones de huellas dactilares (2)**

3) *Identificación de objetos*: Es el procesamiento dirigido a comparaciones específicas de una fotografía con otra. Por ejemplo, el buscar un objeto de color y tamaño específicos dentro de una fotografía, reconocimiento de caras, de huellas dactilares, placas de automóviles, etc. Una aplicación muy útil para el manejo de tiendas son programas que reconocen y cuentan la cantidad de personas que hay un lugar, determinando la afluencia de clientes y el interés del cliente en ciertas áreas específicas, lo cual conlleva el reconocer la cantidad de mujeres, hombres y niños en la imagen. El sistema de control de tráfico de ciudades grandes (varias ciudades estadounidenses y europeas) utiliza cámaras en las calles para determinar la cantidad de automóviles que transitan por ellas durante distintas horas del día. Con dicha información se pueden hacer planes para el manejo del tráfico en horas pico. Permite también, reconocer choques de automóviles o sucesos semejantes, con lo cual se actuar rápidamente y desviar el tráfico a otras calles de la ciudad. También se pueden reconocer las placas de los carros, permitiendo buscar en la ciudad algún automóvil, cosa que sería casi imposible para un operador, inclusive si se tiene a varios operadores monitoreando las pantallas de las cámaras de la ciudad día y noche. (22)



**Figura 3 - Reconocimiento de matrículas, es una de las aplicaciones de visión por computadora. Conlleva el reconocimiento OCR de los números de la matrícula en una fotografía y permite la localización de automóviles en una ciudad a través del sistema de cámaras instalado en las calles. (3)**

El término *Visión de Máquina* es la aplicación de visión por computadora a la industria y manufactura. Mientras el término *Visión por computadora* abarca en general el análisis y procesamiento de imágenes, el término *Visión de máquina* se enfoca en el control de sistemas industriales mediante el procesamiento de imágenes. Una de las aplicaciones más comunes es la inspección de manufactura. Ésta puede ser de una infinidad de productos como chips, semiconductores, automóviles, comida, farmacéuticos, plásticos, etc. Se tienen cámaras inspeccionando líneas de manufacturado y ensamblaje que tienen la capacidad de juzgar la calidad del producto desarrollado.

Para enfatizar la importancia del tema tratado en el presente trabajo de graduación y de su amplia aplicación en sistemas industriales, se mencionarán a continuación algunos casos específicos en donde la *Visión de máquina* ha ayudado a controlar o supervisar procesos industriales.

*Schneider Electric* produce mas de 2000 distintos PCB's de circuitos. La calidad de cada PCB debe estar 100% asegurada antes de que éste pase a ensamblaje. El sistema de visión implementado permite que cada PCB sea inspeccionado y revisado, según su referencia entre las 2000 posibles, antes de pasar a la siguiente etapa. (1)

*Josef Manner Group* es una empresa que produce el famoso Mannerschnitte waffer alemán. La empresa se asegura que sólo los productos que estén en perfectas condiciones y tengan un nivel uniforme de cubierta de chocolate, salgan al mercado, asegurando y manteniendo el éxito del producto. Un sistema de visión por computadora se asegura que cada waffer que sale de la planta de producción este en perfecto estado. (2)

*FP Developments* es una empresa farmacéutica que tiene en su línea de distribución de producto un sistema de multicamaras que permite reconocer el frasco a través de su etiqueta y lo envía la máquina distribuidora de medicamento adecuada. Lo complicado de este sistema es la forma del frasco, si se hiciera de manera manual se debería tener varios operadores supervisando el proceso y leyendo cada etiqueta para poder enviarlo a la distribuidora adecuada o en el caso de utilizar un sistema primitivo de *Visión por*

*Computadora* se debería tener un instrumento que gire el frasco para permitir a un escaner tomar una imagen del mismo y realizar reconocimiento OCR o una cámara que rotara alrededor del frasco para poder tener una vista completa de la etiqueta. Sin embargo con el sistema implementado de multicamaras, la etiqueta se reconstruye a partir de muchas imágenes tomadas desde distintos ángulos y luego se utiliza OCR para el reconocimiento del texto y del código de barras. Sistema muy eficiente y que le ha funcionado a la empresa para distribuir correctamente el producto. (3)



**Figura 4 - FP Developments utiliza un sistema de multivisión para controlar que los frascos de medicina sean llenados correctamente (6)**

*Nissan*, la compañía de vehículos, se enfrentó a un serio problema cuando el sistema actual de posicionamiento de vidrio, no podía posicionar el nuevo tamaño de vidrio para dos de sus más cotizados vehículos. El sistema, llamado ‘centralizador’, localiza seis puntos en la orilla de los vidrios y posiciona el vidrio en una posición centralizada. Es por eso que era necesario un nuevo sistema de visión que pudiera proveer información precisa de los nuevos vidrios y además que pudiera adaptarse fácilmente a futuros cambios en los diseños. (4)

*INVOTEC* es una empresa que desarrolla instrumental médico. El sistema de visión diseñado remueve extractos de hule y otros materiales, de la pieza manufacturada. El sistema de visión dirige un láser indicándole el camino por el cual debe cortar. (5)

*Cognex* es una de las grandes empresas que implementa *Visión por computadora* controlando sistemas mediante retroalimentación visual de cámaras. Ha dado soluciones a problemas reales en la industria. *Vision Systems* es una revista que publica artículos referentes al diseño de sistemas de visión en la industria y comercio en general.

Por lo antes mencionado, es notable que el sistema de control de muchos sistemas industriales es implementados utilizando *Visión por computadora*. Este tema viene teniendo auge desde los años 90s y las aplicaciones del mismo incrementan día a día.

## ii. *Robots Limpiadores de Playas*

Alrededor del mundo existen muchas comunidades que se han asentado aledañas al mar debido a la creciente densidad de población. Adicionalmente, muchos vacacionistas visitan las playas más concurridas, lo cual ha causado que la contaminación de las mismas sea un problema real en la mayoría de países del mundo. Lamentablemente los métodos para limpiar playas ya no son suficientes y requieren del esfuerzo de muchas personas, ya que generalmente las zonas son de difícil acceso y la tarea es exhaustiva (9). Es por ello que se propone la búsqueda de soluciones a este problema mediante el uso robots autómatas dedicados a la limpieza. Este tipo de robots pertenecen a la robótica de servicio, que tiene como objetivo el diseño y construcción de robots con aplicaciones prácticas y de utilidad directa para la sociedad como, entre otras, la limpieza, exploración y vigilancia.

La idea general de un robot limpia playas es un robot autónomo, con sensores y mecanismos de control pre-programados, que lleve a cabo la tarea de limpieza de playas. Este debe reconocer y recoger basura de tamaño y peso limitados al tipo de brazo que tenga.



**Figura 5 - Personas limpiando el Puerto de San José después de que acaba el verano y todos los guatemaltecos regresan a sus tareas cotidianas (9)**

*iii. Megaproyecto UVG “ROVER” 2005*

El Megaproyecto desarrollado en 2005 por los estudiantes de 5to año de Ingeniería Electrónica de la Universidad del Valle de Guatemala (UVG) tuvo como objetivo el desarrollo e implementación del proyecto de graduación denominado ‘*Diseño y construcción de un sistema explorador robotizado y autónomo*’. Ésta iniciativa dio inicio al proyecto del Depto. De Ing. Electrónica de la UVG, denominado ‘ROVER’.



**Figura 6 – Rover UVG, vehículo creado en UVG en 2005 (11)**

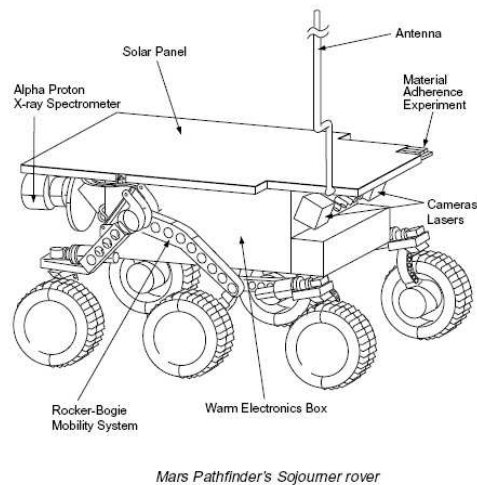
Este proyecto tenía como objetivo crear un robot que cumpliera, en su mayoría, con las especificaciones del robot que la NASA envió en la misión *Mars Pathfinder: Sojourner* lanzada en 1997 con el fin de explorar Marte de una forma más económica que las anteriores misiones que aterrizaron en dicho planeta. Esta misión fue exitosa ya que aterrizó en Marte un vehículo robot con ruedas (vehículos llamados *Rover*) y este exploró la superficie Marciana, enviando información a la tierra, durante los 30 días que duró su misión. El vehículo contaba instrumentos que permitieron el análisis del terreno y con dos cámaras llamadas *ojo izquierdo* y *ojo derecho* para poder realizar una exploración estereoscópica del terreno. Adicionalmente un satélite observaba el vehículo y enviaba las imágenes obtenidas al robot.

El Megaproyecto Rover 2005 contiene dos módulos que involucran el uso de cámaras. El sub-módulo de procesamiento de imágenes y el sub-módulo de control a través de Internet. El primero involucra una cámara externa al robot, simulando un satélite, con el cual se localizaba el vehículo y se determinaba el camino por el cual podía desplazarse. El segundo sub-módulo involucra una cámara posicionada sobre el vehículo, la cual tenía como objetivo que el usuario pudiese observar el terreno desde la perspectiva del robot. A continuación se detallan un poco más los módulos expuestos anteriormente.

#### *a. Sub-módulo: Procesamiento de Imágenes (Mansilla, 2005)*

Este sub-módulo fue el trabajo de graduación presentado por Juan Fernando Mansilla en 2005 y pertenece al módulo de inteligencia artificial del robot *Rover*. El sub-módulo se concentró específicamente en el procesamiento de una imagen tomada del terreno sobre el cual se desplaza el vehículo, que permite generar un mapa en archivo de texto (\*.txt) de unos y ceros, indicando con unos las posiciones donde existe un obstáculo y con ceros las posiciones donde el carro puede desplazarse libremente. Este mapa es provisto al sub-módulo de Algoritmo de Trayectoria, del mismo módulo (Inteligencia Artificial) y permite a éste determinar el camino óptimo para que el vehículo se desplace a la coordenada que le fue indicada. Mansilla implementó una aplicación en la cual se obtiene una imagen digital del terreno, se identifican los posibles obstáculos, mediante una variación al método de detección de contornos Canny, se realiza el archivo de

texto y se transmite la información al sub-módulo correspondiente para el cálculo de la trayectoria más eficiente. La aplicación proporciona la ubicación aproximada del explorador sobre el terreno, información que es utilizada para la detección de errores en el desplazamiento del vehículo. (23)



**Figura 7 - Rover Sojourner, vehículo creado por NASA y aterrizado en Marte en 1996 (12)**

Es importante recalcar que en este proyecto la cámara simulaba un satélite, tal y como en parte fue controlado el Rover Sojourner que se envió a Marte.

*b. Sub-módulo: Control a través de Internet (Mijangos, 2005)*

Este sub-módulo fue el trabajo de graduación presentado por Luis Mijangos en 2005 y pertenece al módulo de telecomunicaciones del robot *Rover*. Ese sub-módulo se concentró en hacer posible que un usuario controlase el *Rover* desde Internet. Para ello, se fabricó una aplicación Web utilizando la plataforma Java, se elaboró la consola de control del vehículo en un applet de Java y se desarrolló un sistema servidor-cliente para permitir el manejo multiusuario del Rover. Además de esto, se implementó la transmisión de video a través de una cámara de red instalada sobre el vehículo para ayudar al usuario a dirigir el vehículo, al tener una vista desde la perspectiva del robot. (24)



## B. Planteamiento del Problema

Como estudiantes del último año de ingeniería electrónica de la Universidad del Valle de Guatemala queremos aplicar los conocimientos adquiridos a lo largo de cinco años de carrera, para plantear una solución concreta al problema de recolección de basura en playas en el país, a través del diseño e implementación del sistema de control del robot autómatas *Rover*. La nueva propuesta de Megaproyecto, retoma el proyecto 2005<sup>1</sup> y pretende optimizar la circuitería, mejorar el sistema de control y visión por computadora, agregar dos módulos más al robot y además propone una nueva aplicación para el mismo, *robot limpia playas*.

La nueva aplicación del *Rover* enfrenta un problema real: la problemática de la contaminación de playas en el país. La presente propuesta propone el diseño y construcción de autómatas que limpien playas. El robot desarrollado es un prototipo de lo que sería un *robot limpia playas*, ya que no está del todo preparado para enfrentar las condiciones de humedad y polvo que se encuentran en una playa. Sin embargo, la propuesta implementada es un prototipo eficiente y funcional a nivel de laboratorio de un robot de éste tipo. Además, aporta una aplicación realista para un robot explorador en países sub-desarrollados, los cuales no están en la capacidad de enviar robots a explorar planetas, a corto o mediano plazo.

El aporte al país es inminente ya que el desarrollo de robótica e inteligencia artificial es un tema que no se trata ni en las universidades ni en la industria. Como futuros ingenieros graduados de la Universidad del Valle tenemos los conocimientos y las habilidades necesarias para diseñar robots con las capacidades básicas de un robot explorador diseñado por instituciones de gran prestigio, como lo es la NASA, tomando en cuenta el limitado tiempo de diseño y presupuesto.

El presente trabajo de graduación pretende llevar a cabo el diseño e implementación del sistema de visión por computadora, implementado en el robot *Rover*. El sistema de visión por computadora le permitirá al robot movilizarse por un ambiente tipo laboratorio esquivando obstáculos y recolectando basura.

---

<sup>1</sup> Diseño y Construcción de un sistema explorador robotizado y autárquico

# Capítulo 2

## Objetivos

*“No hay barco que zarpe, para el que no sabe a que puerto se dirige”*

*San Agustín*

---

### A. OBJETIVOS GENERALES

1. Desarrollar un proyecto multidisciplinario en el que se puedan aplicar todos los conocimientos y habilidades adquiridas durante la carrera.
2. Desarrollar un proyecto de robótica que incentive el desarrollo de nuevas tecnologías en el país.
3. Optimizar y mejorar el Megaproyecto, *Diseño y Construcción de un Sistema Explorador Robotizado y Autárquico*, iniciado en UVG<sup>1</sup> en 2005.

---

<sup>1</sup> Universidad del Valle de Guatemala

4. Crear un sistema robotizado, autárquico y auto sostenible que mediante la realización de tareas, mejore la calidad de vida humana.
5. Utilizar recursos renovables para proveer de energía eléctrica al robot.
6. Construir un brazo robótico que permita al robot hacer tareas simples que puedan mejorar el tipo de actividades realizadas por el mismo.
7. Crear un sistema de Inteligencia Artificial y reconocimiento de imágenes que permita al robot moverse sin necesidad de control humano.

## B. OBJETIVOS ESPECÍFICOS

1. Transmitir una imagen, por la red WLAN, del robot hacia la estación central.
2. Establecer un canal de comunicación digital de datos, a través de la red WLAN, entre la estación central y el robot.
3. Implementar funciones que analicen la imagen adquirida y puedan determinar la localización de obstáculos en ella.
4. Diseñar e implementar un método para medir distancias entre un objeto visto en la imagen y la posición desde donde se tomó dicha imagen.
5. Diseñar e implementar un algoritmo que permita al robot recoger basura en un ambiente tipo laboratorio y esto lo haga esquivando obstáculos.

# Capítulo 3

## Marco Teórico

*“Si quieres construir un edificio que llegue hasta el cielo, piensa primero en poner los cimientos”*

*Agustín de Hipona*

---

En el presente capítulo se describirá todo el material investigado y estudiado, que fue de utilidad en el desarrollo del presente trabajo de graduación.

### *A. Cámaras de Red*

Una cámara de red es aquella que transmite una secuencia de imágenes a través de una red LAN o WLAN. Estas contienen un sistema operativo y servidor web propio, por lo que no necesitan estar conectadas a una computadora para cumplir su cometido. Esta es la diferencia principal entre una *cámara de red* y una *webcam*. La *cámara de red* es un ente autónomo, *i.e.* funciona por si solo, en cambio las *webcam* necesitan estar conectadas a una computadora para transmitir imágenes a través de la red. El protocolo que utilizan para comunicación a través de la red generalmente, es TCP/IP. La cámara de red tiene su propio IP, seguridad integrada y un sistema operativo que funciona como cerebro de la misma. En

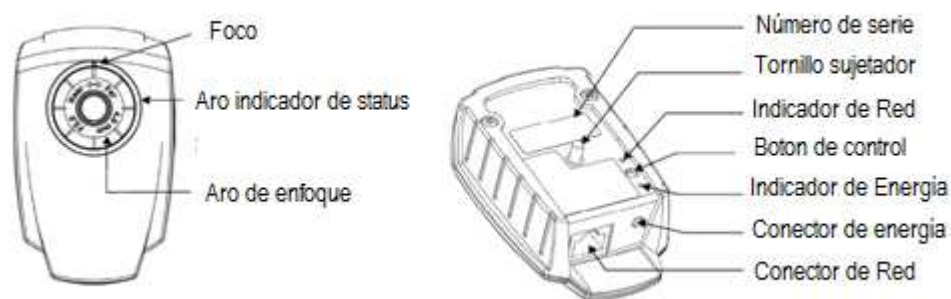
general, estas cámaras son pequeñas computadoras y su objetivo es enviar fotografías a través de la red a una dirección IP específica. (25)

El uso de este tipo de dispositivos se ha popularizado en los últimos años en aplicaciones tales como vigilancia doméstica, seguridad, monitoreo de bebés, monitoreo de procesos industriales, etc. Un usuario podría conectar la cámara en la entrada de su casa y monitorear a las personas que entran y salen de ella. Hay cámaras tan “inteligentes” que inclusive pueden enviar un mensaje de texto al celular o al correo electrónico del usuario, informándole que ha habido movimiento en el área que la cámara vigila. Distintos modelos de estas cámaras tienen la capacidad de detectar movimiento, accionar alarmas o salidas de relé para activar algún dispositivo, puede también ejecutar ‘scripts’, que son programas ejecutables relativamente sencillos, ocupan poco espacio en memoria y son generalmente archivos de texto. Las nuevas cámaras de red tienen incorporada visión nocturna. Estas cámaras permiten al usuario ver video en tiempo real de un lugar, desde su computadora conectada a la red. Las aplicaciones son muchas y es por ello que estas cámaras han tenido auge durante los últimos años. (26)

#### *i. Axis 205*

La cámara de Red Axis 205, es una cámara de red que distribuye la compañía AXIS. Este modelo en particular es un modelo discontinuado. Esta cámara está diseñada para conectarse a una red Ethernet y requiere que se le asigne una dirección IP, ya sea automáticamente a través del protocolo DHCP (Dynamic Host Configuration Protocol) que generalmente tiene los enrutadores o manualmente. (25)

Es una cámara pequeña diseñada para oficinas y casas. Este modelo de cámara transmite video en tiempo real y permite que éste sea visto desde cualquier Web-browser. Ésta es la cámara que utilizó el módulo de *Control a través de internet* para que el usuario tuviese una vista desde la perspectiva del *Rover*. El precio del modelo adquirido hace 3 años para el proyecto 2005: Axis 205, así como el modelo que vino a sustituirlo: Axis 206, es bastante accesible. (25)



**Figura 8 - Cámara de Red: Axis 205 (16)**

Las características más relevantes de la cámara se presentan en la tabla 1.

**Tabla 1 – Características relevantes cámara Axis 205 (25)**

Sensor de Imagen	CMOS de barrido progresivo
Ancho focal	4.0 mm
Compresión de video	JPEG
Resoluciones	640x480, 320x240, 160x120
Frecuencia de imagen	Hasta 30 imágenes por segundo
Velocidad de obturación	1/2 - 1/10.000 segundos
Seguridad	Multiusuario con contraseña
Procesadores y memoria	CPU: RISC de 32 bits, RAM: 16 MB, Flash: 4 MB
Alimentación	5,0 – 5,5 V CC ; Mín. 400 mA
Condiciones de funcionamiento	5 - 40 °C (41 - 104 °F) / Humedad relativa: 20 - 80 %
Protocolos compatibles	IP, HTTP, TCP, ICMP, UDP, SMTP, FTP, DHCP, UPnP™, Bonjour, ARP, DNS, DynDNS, NTP
Dimensiones	85 x 55 x 34 mm (3,35" x 2,17" x 1,34")
Peso	177 g (6,24 oz), soporte incluido, sin unidad de alimentación

## ii. *AxisMediaControl.dll* [AMC]

Este control es utilizado para visualizar video proveniente de una cámara de red Axis en cualquier explorador de internet. AMC es un componente ActiveX que permite ver video de una cámara Axis en aplicaciones de Visual Basic. Este controlador incluye la librería *AxisMediaContol.dll*, que tiene métodos y funciones que permiten controlar la

cámara desde una aplicación hecha por el usuario. Los requerimientos mínimos de la máquina donde se ejecuten los métodos de la librería se incluyen en la tabla 2. (25)

**Tabla 2 – Requerimientos de la máquina donde se ejecute AMC (25)**

	Motion JPEG	MPEG/H.264
Internet Browser	Internet Explorer 6.0 or later	Internet Explorer 6.0 or later
Operating System	<ul style="list-style-type: none"> <li>• Microsoft Windows 2000</li> <li>• Microsoft Windows 2003 Server</li> <li>• Microsoft Windows XP Professional</li> <li>• Microsoft Windows Vista</li> <li>• Microsoft Windows 2008 Server</li> </ul>	<ul style="list-style-type: none"> <li>• Microsoft Windows 2000</li> <li>• Microsoft Windows 2003 Server</li> <li>• Microsoft Windows XP Professional</li> <li>• Microsoft Windows Vista</li> <li>• Microsoft Windows 2008 Server</li> </ul>
CPU	Intel Pentium III 500 MHz or higher (1 GHz recommended), or comparable AMD processor	Intel Pentium III 1 GHz or higher, or comparable AMD processor
Memory	128 MB RAM	128 MB RAM
Graphics Card	16 MB video memory (32/64 MB recommended for high resolution/color depths)	64 MB video memory
Software		DirectX 9.0 or higher

El control incluye un conjunto de decodificadores y aplicaciones para interactuar con la cámara. AAC (*advanced audio coding*), formato de audio digital, es uno de los decodificadores que incluye. Utilizado en el caso se quiera transmitir audio desde la cámara Axis, a través de la red local. Este control puede ser descargado de la página de Axis: [www.axis.com](http://www.axis.com) (25)

### *B. Redes WLAN*

WLAN (Wireless Local Area Network) es una red de comunicación que utiliza radiofrecuencia para la transmisión de datos inalámbricamente. Permite reemplazar por conexiones inalámbricas los cables que conectan a la red las PCs, portátiles u otro tipo de dispositivos, dotando a los equipos de movilidad. Se rige bajo el estándar 802.11 de la IEEE (Institute of Electrical and Electronic Engineer) y se caracteriza por trabajar en bandas de frecuencia exentas de licencia de operación. Es la versión inalámbrica de redes LAN. (27)

El protocolo TCP/IP es un protocolo de comunicación utilizado en transmisiones por redes LAN y WLAN. Básicamente son reglas que deben seguir los equipos conectados a

una red para intercambiar eficientemente información a través de la misma. Cada servicio que se utiliza en internet se comunica utilizando uno o más puertos. Por ejemplo, el correo electrónico utiliza el puerto 25 para enviar y el puerto 110 para recibir. El puerto 80 es el que se usa para desplegar páginas web en general y es el que utilizan las cámaras de red para comunicarse. (28)

Todo equipo conectado a una red WLAN que utiliza el protocolo TCP/IP, tiene una dirección MAC propia única que es una dirección física asignada por la empresa que lo manufacturó y necesita una dirección IP que identifique al equipo dentro de la red. Si una computadora necesita extraer o enviar información a alguna otra, necesita saber la dirección IP de la computadora destino. Hay dos formas de asignar una dirección IP: manual o automáticamente. El sistema que asigna IP automáticamente, por tiempo limitado, se denomina DHCP (Dynamic Host Configuration Protocol). (28)

Existen dos tipos de direcciones IP: las globales y las privadas que se utilizan dentro de una red privada, aumentando la cantidad de personas que se pueden conectar a la red global. Un dispositivo con dirección IP local no puede acceder a otro dispositivo en su red por si solo, necesita una puerta de salida, conocida como “router” o “gateway”, para poder hacerlo. (27)

En la puerta de salida, la red local (LAN) se puede conectar con la red extensa (WAN). La compuerta necesita tener una dirección pública de internet, y esta a su vez puede servir a miles de computadoras en su red interna (LAN) para acceder a la red extensa. Este método de acceso a internet es conocido como NAT-routing (*Network Address Translation*) y fue desarrollado por CISCO. Este es el mecanismo que utilizan los routers IP para intercambio de paquetes entre dos redes, generalmente: la red local (LAN) y la red extensa (WAN). (28)

Son dos las reglas que dominan este método de acceso a internet:



- 1) Una dirección IP privada no puede ser accedida desde el internet público.
- 2) Una computadora que tiene dirección IP privada necesita un punto de salida (Gateway) para acceder al internet.

Una IP es un identificador numérico de 4 bytes para IPv4 o 16 bytes para IPv6, cuyo objetivo es identificar a un dispositivo dentro de una red. Este identificador le permite al dispositivo comunicarse con otros dispositivos conectados dentro o fuera de su red. IPv4 fue creada antes que IPv6 pero debido a la creciente expansión de redes, los 4 bytes (32 bits) para especificar direcciones fueron resultaron ser muy poco, por lo que hubo necesidad de implementar IPv6 utilizando 16 bytes (128 bits). Sin embargo el más utilizado en la actualidad es IPv4. (27)

Recordar 4 cifras de hasta 3 dígitos para ingresar a una dirección de internet es poco práctico. A las personas les es difícil recordar 4 números, es más sencillo recordar palabras. Es de esta necesidad que se propuso un sistema llamado DNS. El DNS (Domain name system) es una base de datos montada en servidores y su función es traducir una cadena de caracteres a la dirección IP correspondiente. (27)

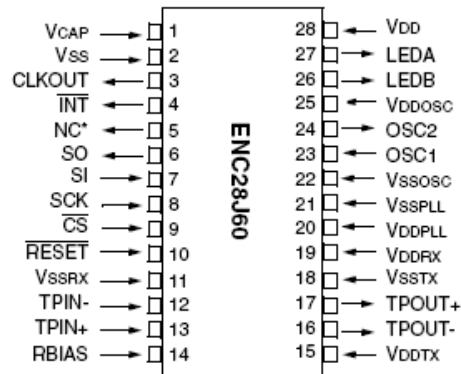
#### *i.* Controlador de Ethernet ENC28J60

El ENC28J60 es un controlador de Ethernet con interfaz SPI. Está diseñado para funcionar como interfaz Ethernet, *IEEE 802.3*, para cualquier controlador con equipado con interfaz SPI. Este cumple con todas las especificaciones del estándar *IEEE 802.3*. La comunicación con el microcontrolador es realizado mediante cuatro pines. Tres de ellos son del módulo de comunicación SPI, *input*, *output*, *clock*. El otro es una salida *i/o* que se conecta al pin de  $\sim CS$  (*chip select*) del controlador.

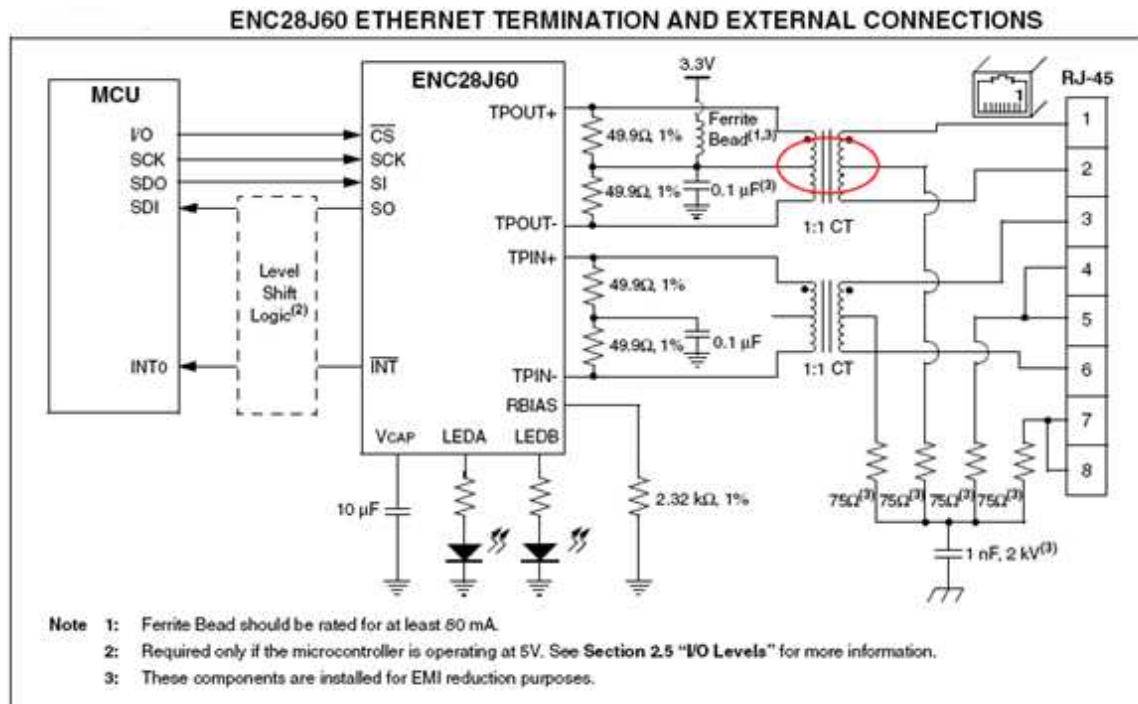
Entre sus características:

- Soporta conexión 10BASE-T con detección y corrección de polaridad incluida.
- Soporta los modos Full-Duplex y Half-Duplex.

- Retransmisión automática en caso de colisión.
- Produce relleno automático y el chequeo de tipo CRC para las tramas que envía.
- Ignora automáticamente paquetes erróneos o cuya dirección IP no coincide con la dirección del dispositivo.
- Interfaz SPI con velocidades de reloj hasta 20MHz. Notar que si la velocidad es baja, el dispositivo es más propenso a perder paquetes.
- Buffer de 8K para transmisión y recepción.
- Manejo tipo FIFO de la recepción de paquetes
- Soporta paquetes tipo UNICAST, MULTICAST y BROADCAST
- Tiene configurada wake-up en recepción de paquetes.
- Leds programables para indicar conexión de RED, TX, RX, colisiones y full/half duplex status.

**28-Pin SPDIP, SSOP, SOIC****Figura 9 - Empaquetado del IC ENC28J60**

El controlador requiere de componentes externos para funcionar. Uno de ellos, no mostrado en la figura 10 es un reloj de 25MHz. Los componentes externos que se le deben conectar se muestran en la figura 10.



**Figura 10 - Conexiones externas requeridas por ENC28J60**

*i. MikroC: SPI Ethernet Library*

La librería, SPI Ethernet Library de MikroC<sup>1</sup>, esta diseñada para simplificar el manejo del ENC28J60. La librería funciona con cualquier PIC que tenga integrado el módulo SPI y que contenga más de 4Kb de memoria ROM.

La librería maneja:

- Protocolo IPv4.
- Protocolo de ARP
- Protocolo de echo ICMP
- Protocolo UDP
- Protocolo TCP

<sup>1</sup> Software para programar microcontroladores PIC desarrollado por la compañía MikroElectronica

## ii. WinSock

*Windows Socket* (WinSock) es un control utilizado en aplicaciones de Windows para comunicarse con otras máquinas utilizando el protocolo TCP/IP o UDP/IP. El control se llama MSWINSCK.OCX. La librería del control se llama *winsock.dll* y contiene métodos para facilitar al programador la interacción entre un programa y una red que utilice dichos protocolos de comunicación. (29)

## C. Formatos de imagen

Una fotografía está compuesta por un conjunto de unidades discretas. Existen dos formatos gráficos y cada uno de ellos define ésta ‘unidad’ discreta. Los formatos son *vector* y *raster*. En la figura 11 se presenta una tabla con las fortalezas y deficiencias de cada formato. (15)

	<i>Raster Graphics</i>	<i>Vector Graphics</i>
Display speed	X	
Image quality	X	
Memory usage		X
Ease of editing		X
Display independence		X

**Figura 11 - Fortalezas y deficiencias de los dos tipo de formato de gráficos: Raster vs Vector.**

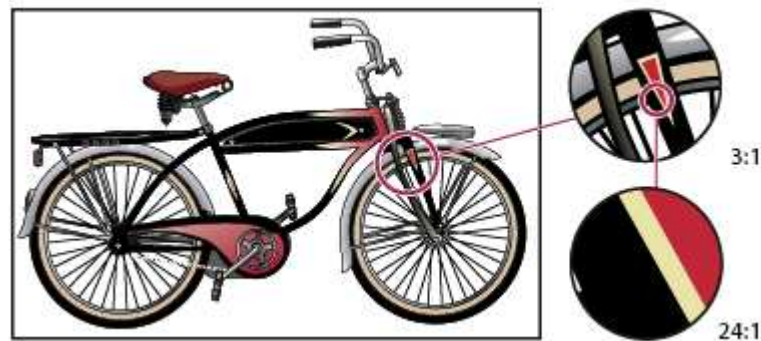
**Se indica con una x la debilidad del formato respecto a dicha característica (15)**

Nótese que *raster* tiene deficiencias en los aspectos de calidad de imagen y velocidad de despliegue. El formato *vector* utiliza mucha memoria para almacenar imágenes y la forma en la que representa una imagen es mucho más compleja de como lo hace el formato *raster*. (15)

## i. Formato Vector

El formato *vector* utiliza formas u objetos vectoriales y funciones geométricas como unidades fundamentales para describir una imagen. En general son líneas y curvas definidas por objetos matemáticos, llamados vectores. Estos describen a la imagen según

sus características geométricas. Dado que el formato utiliza funciones geométricas para describir áreas en una imagen, lo hace poco útil para la representación de imágenes que no tengan formas fácilmente representables con figuras geométricas. Sin embargo este tipo de imágenes se utiliza ampliamente en diseño gráfico ya que, dado que la imagen esta definida por figuras con bordes bien definidos, es posible modificar la imagen con libertad sin perder detalle ni claridad. Imágenes en formato *vector* mantienen los bordes cuando se les cambia de tamaño, se imprimen o se guardan en archivos. Esto por esto que los gráficos vectoriales son la mejor opción para ilustraciones como logotipos o publicidad que se utilizarán en varios tamaños (distintas escalas). Se puede decir entonces que éste formato de imagen es *independiente de la resolución*. Sin embargo, imágenes muy complejas tardan mucho tiempo en ser codificadas y decodificadas. Ejemplos de formato *vector* son SVG y SWF; ambos producen gran calidad de gráficos. (17)



**Figura 12 – Imagen en formato *vector* vista a distintas escalas (17)**

## ii. *Formato Raster*

El formato *raster* representa una imagen como un *arreglo* de dos dimensiones, donde cada elemento representa el color a ser desplegado en un área de la imagen. Cada elemento del array es una unidad elemental del formato, llamada *pixel*. Estas son imágenes denominadas *mapas de bits*, técnicamente imágenes *rasterizadas*. Cada pixel tiene una ubicación y color específicos. La calidad y nitidez de la imagen es directamente proporcional a la cantidad de pixeles que se utilicen para representar la misma, pero esto a la vez aumenta la cantidad de memoria que ocupa. Por lo que si se quiere reducir el tamaño en memoria que ocupa la imagen es necesario desechar información de la misma, causando

pérdida de calidad y nitidez. Este tipo de imágenes pierden detalle fácilmente al cambiarlas de tamaño. Al cambiar la escala de una imagen *raster*, ésta no conserva su nitidez. Se puede decir entonces que este formato de imagen es *dependiente de la resolución*. Sin embargo, las imágenes de mapa de bits son el medio electrónico más utilizado para las imágenes de tono continuo, tales como fotografías, puesto que pueden representar de manera eficaz los colores y sombras en la imagen. Todas las cámaras digitales utilizan algún tipo de formato *raster* para representar fotografías. Este tipo de formato utiliza espacio en memoria, según el tipo de compresión utilizada. Entre los formatos raster más conocidos y utilizados están: GIF, JPEG, PNG y BMP. (15)



**Figura 13 - Imagen *raster* vista a diferentes escalas (17)**

Del formato *raster* antes mencionado se detallará la estructura que manejan dos distintos tipos de compresión de imágenes *raster*: JPEG y BMP.

#### *a. JPEG*

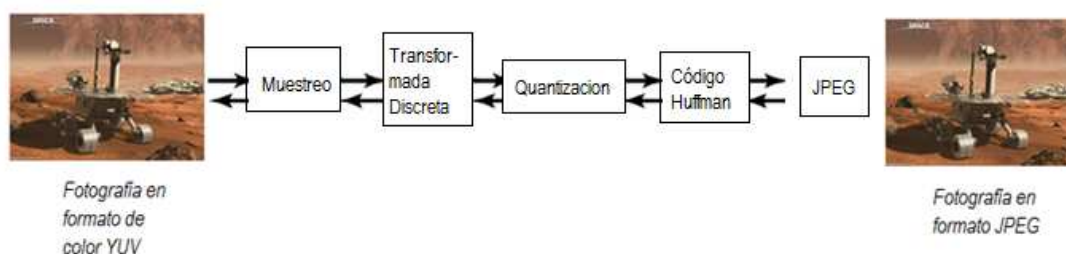
Es un tipo de compresión de imágenes *raster*. Este tipo de compresión pierde la calidad de la imagen al reducir el tamaño que ésta ocupa en memoria. En general este tipo de compresión es denominado '*lossy*', lo que significa que la información que se descarta, para permitir la compresión, se pierde permanentemente. Lo que permite a JPEG reducir el tamaño de la imagen sin que el ojo humano lo note, es el desechar información que el ojo humano no puede percibir. Este estándar de compresión fue desarrollado por "*Joint Photographic Experts Group*", de allí el nombre: *JPEG*. Este formato es el más utilizado en redes, para representar imágenes, por su reducido tamaño. (15)

Una imagen es representada por una cámara utilizando tres componentes: RGB. Debe recordarse que JPEG es un tipo de compresión, no de formato de imagen, por lo que se debe tener la imagen a comprimir en su representación RGB. El primer paso para la compresión es convertir el sistema en el que esta representada la imagen. Convertir de RGB de tres componentes: R (*rojo*), G (*green*) y B (*blue*) a YUV, también de tres componentes: Y, Cb y Cr. En la figura 14 se muestran las ecuaciones de conversión entre los dos modelos de color: YUV y RGB. (15)

$$\begin{aligned}
 Y &= 0.299R + 0.587G + 0.114B \\
 Cb &= -0.1687R - 0.3313G + 0.5B + 2^{\text{Presicion de muestreo}/2} \\
 Cr &= 0.5R - 0.4187G - 0.0913B + 2^{\text{Presicion de muestreo}/2} \\
 R &= Y + 1.402 Cr \\
 G &= Y - 0.34414 \left( Cb - 2^{\frac{\text{Presicion}}{2}} \right) - 0.71414 (Cr - 2^{\text{Presicion}}) \\
 G &= Y + 1.722 \left( Cb - 2^{\frac{\text{Presicion}}{2}} \right)
 \end{aligned}$$

**Figura 14 - Ecuaciones para conversión entre los modelos de color RGB y YUV (15)**

YUV es un sistema para representar el conjunto de colores que forman una imagen, utilizando tres componentes. Utiliza 24 bits para describir el color de un pixel. Cada una de las tres componentes se representa con 8 bits. La componente Y es la *luminancia*, que almacena la información del brillo del pixel, y es válido decir que Y representa la imagen en escala de grises. Tanto Cb como Cr son componentes de *crominancia*, que describen el contenido de color del pixel. En específico Cb representa lo azulado y Cr lo rojizo de la imagen. (15)



**Figura 15 - Diagrama de bloques de la compresión de imágenes raster: JPEG (15)**

En el modelo YUV, un pixel se describe definiendo los tres parámetros: Y, Cb y Cr. La componente Y almacena la mayor cantidad de información, y se podría decir que es la componente más significativa, de mayor peso. JPEG aprovecha esta característica y utiliza más bits para representar dicha componente de los que utiliza para representar las otras dos. (15)

Nótese en la figura 15 el proceso que conlleva una compresión JPEG después de tener la imagen ya en su representación por el sistema de colores YUV:

1. Muestrear
2. Transformada discreta de Cosenos
3. Quantización
4. Codificación Huffman

Al *muestrear*, se realiza un proceso denominado '*down sampling*', lo que se traduce en dar distintos pesos a cada componente Y, Cr y Cb. Es decir, la componente Y, la más significativa, se muestrea a una frecuencia mayor que la frecuencia a la que se muestrean las otras dos componentes. Se entenderá mucho mejor el concepto observando la figura 16. Una imagen JPEG disminuye la cantidad de bytes necesaria para almacenar una imagen, al muestrear cada componente a distintas frecuencias. Nótese que la compresión es más de un 60%. (15)



<u>CADENA DE PÍXELES ORIGINAL</u>												No. total de píxeles: 36
Y = 3 Cb = 4 Cr = 7	Y = 4 Cb = 12 Cr = 8	Y = 3 Cb = 9 Cr = 9	Y = 13 Cb = 10 Cr = 5	Y = 11 Cb = 8 Cr = 7	Y = 9 Cb = 6 Cr = 4	Y = 8 Cb = 7 Cr = 2	Y = 15 Cb = 11 Cr = 6	Y = 10 Cb = 8 Cr = 3	Y = 12 Cb = 0 Cr = 7	Y = 9 Cb = 6 Cr = 6	Y = 12 Cb = 6 Cr = 4	
<u>IMAGEN COMPRIMIDA</u>												No. total de píxeles: 12
Y = 3	Y = 3	Y = 11	Y = 8	Y = 10	Y = 9							
Cb = 4    Cr = 7		Cb = 8    Cr = 7		Cb = 8    Cr = 3								

**Figura 16 - Ejemplo de compresión de *down sampling* en una imagen**

Cada canal se trata de manera independiente en la compresión. ‘Canal’ se refiere a las componentes del modelo de colores utilizado, en YUV estas son Y, Cb y Cr. La imagen se subdivide en áreas de 8x8 píxeles. Dado que se tratará cada canal por individual, se tienen tres representaciones de la imagen, donde cada pixel es representado por 8 bits. Nótese también que se espera que la matriz de píxeles de las componentes Cb y Cr sea más pequeña que la matriz de píxeles de la componente Y. (15)

A cada elemento de la matriz de un componente se le aplica la *transformada discreta de cosenos* para convertirlo al dominio de la frecuencia. La transformada discreta de cosenos es el proceso fundamental de la compresión JPEG. Esta transformada mapea de  $R[n \text{ píxeles}] \rightarrow R[n \text{ píxeles}]$ , es decir mapea una fotografía de n píxeles en una fotografía de n píxeles. Es una transformada cuya inversa está definida y es utilizada para recuperar la imagen original; esta inversa es llamada ‘Transformada Inversa discreta de cosenos’. (15)

La transformada discreta de cosenos en dos dimensiones, se muestra en las siguientes ecuaciones. Nótese que esta es la transformada de interés, ya que la imagen es un matriz de dos dimensiones. (15)

$$T[i, j] = c(i, j) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} V[y, x] \cos \frac{(2y+1)i\pi}{2N} \cos \frac{(2x+1)j\pi}{2N}$$

$$c(i, j) = \frac{2}{N}, i \text{ and } j \neq 0$$

$$c(i, j) = \frac{1}{N}, i \text{ or } j = 0$$

Para compresión JPEG el valor de N siempre es 8, como se definió anteriormente, cada canal de la imagen se subdivide en pequeñas matrices de 8x8 pixeles. (15)

La definición de la transformada inversa de cosenos, para dos dimensiones es la siguiente:

$$V[y, x] = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} c(i, j) T[i, j] \cos \frac{(2i+1)j\pi}{2N} \cos \frac{(2x+1)i\pi}{2N}$$

Esta es muy importante también ya que las imágenes JPEG se comprimen para ser almacenadas en memoria pero cuando se requiere visualizarlas, estas deben ser descomprimidas. Para comprimir las se utiliza la transformada discreta de cosenos y para descomprimirlas se utiliza la transformada inversa discreta de cosenos. (15)

La matriz de la transformada es la que se muestra en la figura 17.

$$M = \begin{bmatrix} \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} & \frac{1}{\sqrt{8}} \\ \frac{1}{2} \cos \frac{1}{16} \pi & \frac{1}{2} \cos \frac{3}{16} \pi & \frac{1}{2} \cos \frac{5}{16} \pi & \frac{1}{2} \cos \frac{7}{16} \pi & \frac{1}{2} \cos \frac{9}{16} \pi & \frac{1}{2} \cos \frac{11}{16} \pi & \frac{1}{2} \cos \frac{13}{16} \pi & \frac{1}{2} \cos \frac{15}{16} \pi \\ \frac{1}{2} \cos \frac{2}{16} \pi & \frac{1}{2} \cos \frac{6}{16} \pi & \frac{1}{2} \cos \frac{10}{16} \pi & \frac{1}{2} \cos \frac{14}{16} \pi & \frac{1}{2} \cos \frac{18}{16} \pi & \frac{1}{2} \cos \frac{22}{16} \pi & \frac{1}{2} \cos \frac{26}{16} \pi & \frac{1}{2} \cos \frac{30}{16} \pi \\ \frac{1}{2} \cos \frac{3}{16} \pi & \frac{1}{2} \cos \frac{9}{16} \pi & \frac{1}{2} \cos \frac{15}{16} \pi & \frac{1}{2} \cos \frac{21}{16} \pi & \frac{1}{2} \cos \frac{27}{16} \pi & \frac{1}{2} \cos \frac{33}{16} \pi & \frac{1}{2} \cos \frac{39}{16} \pi & \frac{1}{2} \cos \frac{45}{16} \pi \\ \frac{1}{2} \cos \frac{4}{16} \pi & \frac{1}{2} \cos \frac{12}{16} \pi & \frac{1}{2} \cos \frac{20}{16} \pi & \frac{1}{2} \cos \frac{28}{16} \pi & \frac{1}{2} \cos \frac{36}{16} \pi & \frac{1}{2} \cos \frac{44}{16} \pi & \frac{1}{2} \cos \frac{52}{16} \pi & \frac{1}{2} \cos \frac{60}{16} \pi \\ \frac{1}{2} \cos \frac{5}{16} \pi & \frac{1}{2} \cos \frac{15}{16} \pi & \frac{1}{2} \cos \frac{25}{16} \pi & \frac{1}{2} \cos \frac{35}{16} \pi & \frac{1}{2} \cos \frac{45}{16} \pi & \frac{1}{2} \cos \frac{55}{16} \pi & \frac{1}{2} \cos \frac{65}{16} \pi & \frac{1}{2} \cos \frac{75}{16} \pi \\ \frac{1}{2} \cos \frac{6}{16} \pi & \frac{1}{2} \cos \frac{18}{16} \pi & \frac{1}{2} \cos \frac{30}{16} \pi & \frac{1}{2} \cos \frac{42}{16} \pi & \frac{1}{2} \cos \frac{54}{16} \pi & \frac{1}{2} \cos \frac{66}{16} \pi & \frac{1}{2} \cos \frac{78}{16} \pi & \frac{1}{2} \cos \frac{90}{16} \pi \\ \frac{1}{2} \cos \frac{7}{16} \pi & \frac{1}{2} \cos \frac{21}{16} \pi & \frac{1}{2} \cos \frac{35}{16} \pi & \frac{1}{2} \cos \frac{49}{16} \pi & \frac{1}{2} \cos \frac{63}{16} \pi & \frac{1}{2} \cos \frac{77}{16} \pi & \frac{1}{2} \cos \frac{91}{16} \pi & \frac{1}{2} \cos \frac{105}{16} \pi \end{bmatrix}$$

**Figura 17 - Matriz de la transformada discreta de cosenos (15)**

Recuérdese que una transformación matricial en álgebra lineal conlleva la multiplicación del objeto a transformar por la matriz de transformación.

$M$  es una matriz ortogonal, lo que quiere decir que el producto de la matriz  $M$  y su transpuesta es la matriz identidad  $I$ . Nótese que la frecuencia de los cosenos de la matriz crece a medida que se acerca al lado inferior derecho, mientras que las frecuencias tienden a cero a medida que se acercan al término superior izquierdo.

Ahora se supondrá que se tomó una fotografía, se cambió de representación de colores RGB a YUV, se subdividió en tres canales: Y, Cr y Cb y que una de ellas, en específico la matriz de la componente Y, luce como la figura 18.

58	45	29	27	24	19	17	20
62	52	42	41	38	30	22	18
48	47	49	44	40	36	31	25
59	78	49	32	28	31	31	31
98	138	116	78	39	24	25	27
115	160	143	97	48	27	24	21
99	137	127	84	42	25	24	20
74	95	82	67	40	25	25	19

**Figura 18 - Matriz de la componente Y de una fotografía en formato YUV (15)**

La matriz después de aplicada la transformada de coseno es la que se muestra en la figura 19.

-603	203	11	45	-30	-14	-14	-7
-108	-93	10	49	27	6	8	2
-42	-20	-6	16	17	9	3	3
56	69	7	-25	-10	-5	-2	-2
-33	-21	17	8	3	-4	-5	-3
-16	-14	8	2	-4	-2	1	1
0	-5	-6	-1	2	3	1	1
8	5	-6	-9	0	3	3	2

**Figura 19 – Matriz después de aplicada la transformada discreta de cosenos de la matriz de la componente Y (15)**

Nótese cómo la transformada amplifica las frecuencias bajas y atenúa las frecuencias altas. Es aquí donde JPEG pierde información al comprimir la imagen. La estrategia de JPEG es desechar las frecuencias altas. El ojo humano no puede detectar fácilmente cambios bruscos de color o luminosidad en las fotografías, por lo que al eliminar las frecuencias altas de la imagen, es decir los cambios bruscos de colores en la imagen, se desecha información que el ojo humano no puede percibir.

El estándar JPEG especifica las matrices a utilizar para cuantizar cada componente YUV. En el caso de la componente Y, el estándar especifica la matriz mostrada en la figura 20.

Para *cuantizar* se debe dividir la matriz de coeficientes presentada en la figura 19 dentro de la matriz presentada en la figura 20 (definida en el estándar), dado que es la matriz de la componente Y.

$$\begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

**Figura 20 - Matriz especificada por el estándar JPEG para cuantizar la componente Y del modelo YUV (15)**

17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	66	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99

**Figura 21 - Matriz especificada por el estándar JPEG para cuantizar las componentes Cr y Cb del modelo YUV (15)**

Una vez cuantizada la matriz, figura 22, es mucho más obvio ver que la transformada de cosenos desecha las frecuencias altas, ya que se obtiene una matriz cuya diagonal inferior está casi en su totalidad conformada por ceros.

Para almacenar los componentes de la matriz, se utiliza una estructura de *arreglo* lineal con un ordenamiento en *zig-zag*, con el fin de dejar todos los ceros al final del arreglo. Es decir el archivo JPEG que contenga estos pixeles en ella, se leería así:

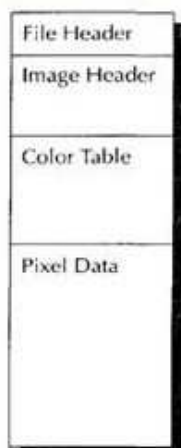
18 -9 -3 -8 1 -3 1 -2 4 -2 4 0 3 -1 0 1 1 0 -1 -1 0 0 0 -1 (39 ceros)

-38	18	1	-3	-1	0	0	0
-9	-8	1	3	1	0	0	0
-3	-2	0	1	0	0	0	0
4	4	0	-1	0	0	0	0
-2	-1	0	0	0	0	0	0
-1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

**Figura 22 - Matriz cuantizada de la transformada de cosenos (15)**

La *codificación* es el último proceso de la conversión JPEG. La codificación que utiliza este formato es la codificación de Hoffman, este es una codificación perfecta. El estándar también incluye tablas donde especifica cómo será la codificación.

#### ESTRUCTURA DE ARCHIVO



La decodificación es el proceso inverso al anteriormente descrito.

#### b. BMP

El formato BMP es el más simple de los formatos raster y es muy comúnmente utilizado en análisis de imágenes, dado que guarda la información en el modelo de colores RGB sin ninguna compresión. Soporta imágenes de 1, 4, 8, 16, 24 y 32 bits. El más comúnmente utilizado es el de 24 bits por pixel, que es el estándar de Windows.

**Figura 23 - Estructura de un archivo BMP (15)**

Nótese en la figura 23 que la estructura del formato BMP es bastante simple y consta de cuatro campos en los cuales se describe el tamaño, formato, compresión y la información de píxeles de la imagen, las cuales se describirán a continuación.

1) File Header:

Todo archivo BMP comienza con este encabezado que lo identifica como tal. La función principal de este campo es servir como un identificador para que el sistema operativo lo reconozca como archivo BMP. El formato de este campo es el que se muestra en la figura 24. (15)

Las características importantes de este encabezado son:

- Los primeros dos bytes del archivo deben ser los caracteres ASCII “B” y “M” indicando que es un archivo BMP.
- El tamaño del archivo (si se puede) debe ser comparado con el valor de bfSize ya que este campo indica el tamaño total del archivo.
- Los dos siguientes bytes deben ser cero bfReserved1 y bfReserved2.
- El último campo bfOffbits indica el offset desde el inicio del archivo hasta donde comienzan los datos de pixel. Este dato es importante ya que es utilizado para leer la data de píxeles. (15)

BITMAPFILEHEADER Structure	Field Name	Size in Bytes	Description
	bfType	2	Contains the characters "BM" that identify the file type
	bfSize	4	File size
	bfReserved1	2	Unused
	bfReserved2	2	Unused
	bfOffBits	4	Offset to start of pixel data

**Figura 24 - Estructura del File header (15)**

## 2) Info Header:

Sigue a la estructura de File Header y se puede encontrar en dos posibles formatos: BitMapInfoHeader y BitMapCoreHeader. El Info header es más común encontrarlo en Windows mientras que el Core header es más común encontrarlo en Mac OS. El formato del Info Header se puede observar en la figura 25 y generalmente es este el que se encuentra en la mayoría de BMPs. Dado que no hay un estándar, se reconoce cuál de dichas estructuras contiene un archivo BMP, revisando los primeros 4 bytes (biSize) del Info Header, en donde se indica el tamaño en bytes de este campo.

- El tamaño de BitMapInfoHeader es de 40 bytes.
- El tamaño de BitMapCoreHeader es de 12 bytes.

Los campos importantes de ésta estructura son los que indican las dimensiones del archivo y cantidad de bits que describen a un pixel.

	Field Name	Size	Description
BITMAPINFOHEADER Structure	biSize	4	Header size—Must be at least 40
	biWidth	4	Image width
	biHeight	4	Image height
	biPlanes	2	Must be 1
	biBitCount	2	Bits per pixel—1, 4, 8, 16, 24, or 32
	biCompression	4	Compression type—BI_RGB=0, BI_RLE8=1, BI_RLE4=2, or BI_BITFIELDS=3
	biSizeImage	4	Image Size—May be zero if not compressed
	biXPelsPerMeter	4	Preferred resolution in pixels per meter
	biYPelsPerMeter	4	Preferred resolution in pixels per meter
	biClrUsed	4	Number of entries in the color map that are actually used
	biClrImportant	4	Number of significant colors

**Figura 25 - Estructura del Info Header (15)**

## 3) Color Table (Paleta):

Ésta estructura se encuentra, si es que existe, después de la estructura de Image Header. Puede que se añada o no dependiendo de la definición del archivo y puede existir en dos distintos formatos. Si la imagen es descrita con 1, 4 u 8 bits, la paleta consiste de un

arreglo de  $2^{\text{bit\_count}}$ . Si la imagen está descrita con 24 bits, la paleta no es incluida y por defecto el orden de los pixeles será: B (blue), G (green) y R (red).

#### 4) Pixel Data:

Este campo sigue al ‘color table’ o en su ausencia, sigue al Image Header. Para asegurarse de la localización del inicio de este campo, se debe revisar la información de bfOffbits del File header. Los pixeles de la imagen están ordenados de izquierda a derecha, de abajo hacia arriba. El número de filas está dado por el campo de biHeight y el número de columnas está dado por biWidth. La cantidad biBitCount indica la cantidad de bits por fila. El formato de los bytes de pixel data esta definido por la cantidad de bits describen a un pixel.

- i. Imagen de 1, 4 u 8 bit por pixel: La cantidad de bits que describan a un pixel representa un índice de la paleta de colores. En el caso de ser monocromo (1 bit) solo se tienen dos colores: negro y blanco. En el caso de tener 4 bits por pixel, la paleta es de 16 colores. En el caso de 8 bits por pixel, la paleta es de 256 colores.
- ii. Imagen de 16 bits por pixel: Cada pixel es representado por dos bytes. La intensidad de cada color es representado con 5 bits con el bit más significativo inutilizado, como lo muestra la figura 26.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROJO						AZUL					VERDE				

**Figura 26 - Ordenamiento de los colores en una estructura de 16 bits por pixel (15)**

- iii. Imagen de 24 bits por pixel: Cada pixel es representado por tres bytes y cada uno de estos describe uno de los colores en RGB. El orden es el predefinido Blue, Green, Red.



### *D. Librería vic32.dll*

La librería ‘Victor Image Processing Library’ es una DLL (Dynamic Link Library). Ésta es un conjunto de métodos y funciones compiladas en un archivo binario. Sus funciones pueden ser invocadas desde cualquier aplicación. Este tipo de librerías tienen el beneficio que sus métodos se cargan solamente bajo la demanda de funciones. (30)

Esta librería, vic32.dll, está dirigida hacia el procesamiento de imágenes, por lo que contiene métodos para trabajar con ellas y que facilitan mucho la tarea del programador. La librería puede ser utilizada para crear aplicaciones que trabajen con imágenes en plataformas Windows. (30)

Entre sus principales características están:

- Abrir y salvar imágenes en formato BMP/GIF/JPEG/PCX/PNG/TGA/TIFF
- Procesamiento de imágenes en escala de grises y de color
- Combinar y comparar imágenes
- Cambiar el tamaño y orientación de una imagen
- Análisis de imágenes utilizando histogramas, correlaciones, ordenamiento de pixeles
- Despliegue de imágenes
- Permite manejar dispositivos tipo TWAIN (cámaras web). (30)

La librería contiene:

VIC32.DLL	Librería dedicada a procesamiento de imágenes, abrir y salvar imágenes
VICFX.DLL	Es una librería adicional que permite el análisis de múltiples imágenes
VICSTATS.DLL	Análisis de Imágenes
VICTW32.DLL	Soporte para dispositivos TWAIN

La interfaz a todo el paquete es el archivo: VICLIB.DLL (30)

### *E. Localización y Reconocimiento de objetos en imágenes*

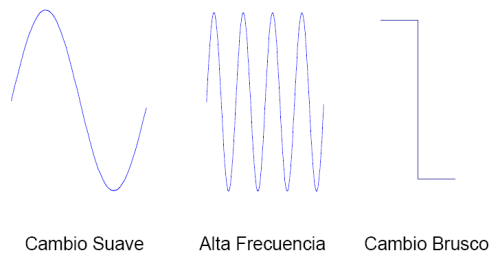
La localización y reconocimiento de objetos es una aplicación de la disciplina de *visión por computadora*. Es la tarea de encontrar un objeto dado en una imagen o secuencia de imágenes (video). La dificultad de esta tarea es que el objeto en una imagen puede verse desde distintas perspectivas, con diferentes tamaños, escala o rotación. Una tarea realmente retadora, actualmente para los eruditos en *visión por computadora*, es reconocer el objeto en las descritas circunstancias y en especial cuando éste está parcialmente obstruido. Un humano podría realizar todas estas tareas sin gran dificultad y, dado que esta disciplina pretende imitar la percepción visual humana, es necesario proponer rutinas de procesamiento que aproximen dicho comportamiento. (32)

Para reconocer objetos en imágenes se necesita en primera instancia localizarlos y en ella, i.e. determinar las coordenadas x,y de su punto medio, determinar su alto y ancho en pixeles para luego aplicar análisis sobre esa área de la imagen. Todo esto con el fin de determinar si el objeto localizado cumple con ciertos patrones con lo cual se le reconocería como el objeto de interés. (32)

El formato de imagen generalmente utilizada para fines de reconocimiento de objetos en imágenes es *raster* ya que permite el análisis de la imagen mediante sus unidades discretas ‘pixeles’, que corresponden a un área específica de la fotografía. Aunque se quisiera analizar una imagen en formato *vector*, no existen cámaras que manejen este tipo de formato. (32)

Para realizar las tareas de *localización y reconocimiento* de objetos en imágenes, le es necesario al programador utilizar ciertas herramientas de análisis, en general llamadas: *filtros digitales*. Un filtro es selectivo en frecuencia. Puede ser que preserve cambios suaves de tonalidades en pixeles adyacentes (*filtros pasa baja*), cambios bruscos (*filtros pasa alta*) o un valor intermedio entre estos (*filtros pasa banda*). (32)

Nótese en la figura 27 como se percibe un cambio suave y un cambio brusco.

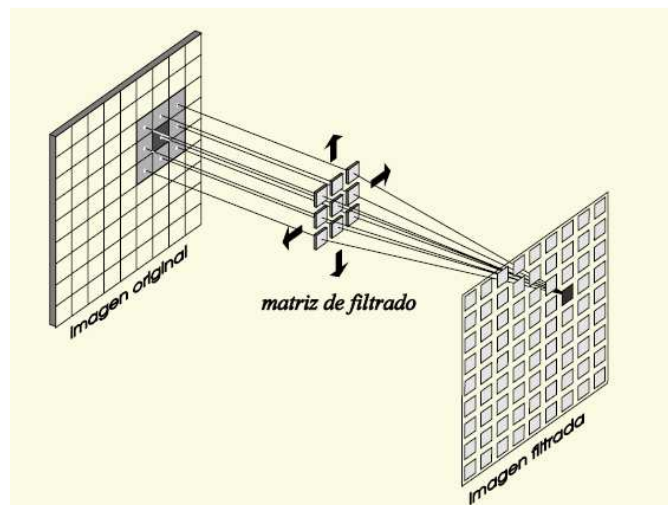


**Figura 27 - Cambios suaves y bruscos (15)**

*i. Filtros digitales*

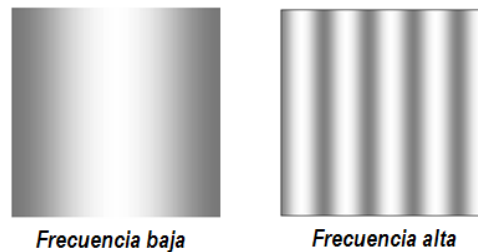
Una herramienta muy útil en procesamiento de imágenes es el uso de filtros, ya que permiten preservar cierta información en la imagen y desechar otra. Se busca generalmente conservar características en la imagen que cumplen con determinado patrón. (31)

Un filtro, ya sea digital o analógico, busca preservar determinadas *frecuencias* dentro de la imagen digital. El uso de *mascaras* es muy común en la aplicación de filtros digitales. Básicamente una matriz de pixeles en una imagen es convolucionada con una matriz de constantes a la que se le llama *mascara*. Nótese en la figura 28 cómo se convoluciona la matriz de pixeles con la máscara M. (31)



**Figura 28 – Aplicación de filtros digitales (18)**

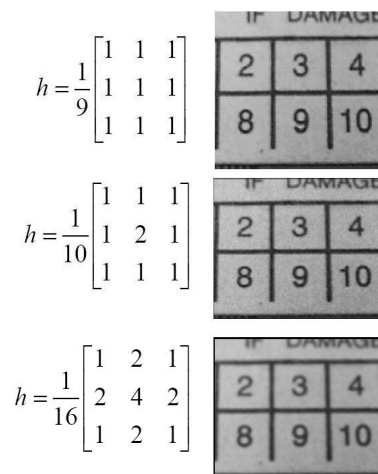
Es un tanto inusual pensar en una *imagen de frecuencia baja* o de *frecuencia alta*. Por ello, se muestran ejemplos en la figura 29 de lo que se percibiría en una imagen como frecuencia alta o baja.



**Figura 29 - Ejemplos de frecuencia baja y alta en una imagen (18)**

*a. Filtros pasa baja*

Un filtro pasa baja atenúa frecuencias altas. En lenguaje de imágenes digitales se refiere a: cambios leves de tonalidad de pixeles adyacentes. Esto corresponde a superficies homogéneas. Un filtro pasa baja digital es descrito por una máscara que al realizar la convolución con una matriz de pixeles logra el efecto de atenuar frecuencias altas. La imagen se torna más borrosa mientras mayor sea la atenuación. Nótese en la figura 30 el distinto efecto de las matrices de convolución sobre la imagen. (32)

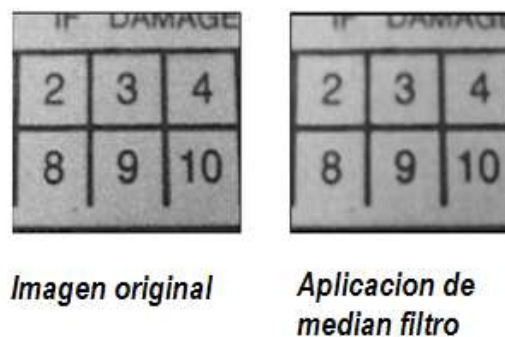


**Figura 30 - Efecto de distintas máscaras sobre imágenes efecto de borroso (18)**

### *1) Filtro de Mediana*

Un filtro de mediana (“median filter”) es un filtro pasa baja y sin embargo funciona un tanto diferente a lo descrito anteriormente. No es simplemente una convolución sino se tiene un poco más de procesamiento. Se ordenan todos los píxeles de la imagen cubiertos por la máscara de mayor a menor y se le asigna a todo el conjunto de píxeles el valor central de dicho ordenamiento. (33)

Una característica importante de este filtro pasa baja es que no degrada tanto la imagen como otros filtros de este tipo. El efecto que tiene es eliminar píxeles cuyo entorno no sea de la misma naturaleza del píxel en estudio. Véase el efecto que tiene un filtro de mediana sobre una imagen en la figura 31. (33)

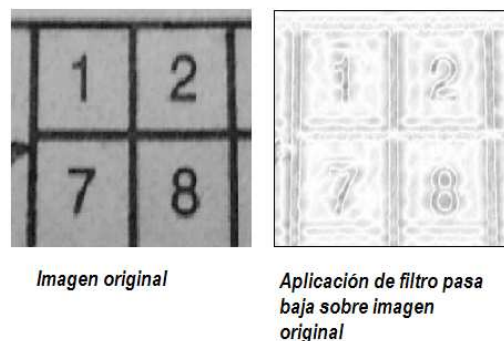


**Figura 31 - Efecto de filtro mediana sobre una imagen (18)**

### *b. Filtros pasa alta*

Un filtro pasa alta preserva frecuencias altas. Esto se refleja en una imagen como cambios bruscos de tonalidad en píxeles adyacentes y que corresponden a los bordes de un objeto. (32)

Este tipo de filtros acentúa los cambios bruscos en la imagen. Para un mejor entendimiento véase la figura 32 donde se muestra el efecto de un filtro pasa alta aplicado a una imagen. (32)



**Figura 32 - Efecto de filtro pasa alta sobre una imagen (18)**

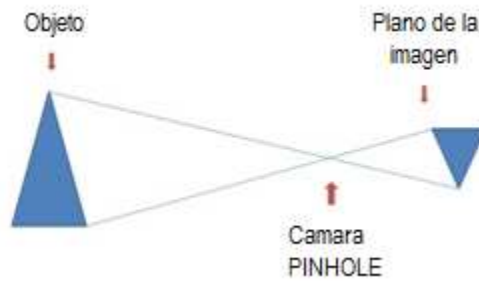
### *F. Aproximación de distancia de objetos en imágenes*

Este tema de la disciplina de *visión por computadora* tiene por objetivo determinar aproximadamente la distancia entre la cámara y un objeto. Esta tarea está íntimamente relacionada con las tareas de detección de obstáculos y reconocimiento de formas. La estimación de distancias puede mejorar el control y navegación de un robot. Para estas tareas es necesario modelar la cámara de alguna manera ya que se tiene que incluir el modelo de la cámara dentro de cálculos pertinentes para determinar distancia de la cámara al objeto.

#### *i. Modelo Pinhole*

Al capturar una imagen 3D mediante una en un plano 2D utilizando una cámara es preciso plantear el problema de medición de distancias utilizando un modelo que describa a la cámara. El modelo utilizado y sobre el cual se trabajaron las ecuaciones para el cálculo de distancia es el llamado: “pin-hole model” (en español “modelo estenopéico”). Este modelo es uno de los más utilizados en visión robótica, por su simplicidad. Éste modelo desprecia el diámetro del lente de la cámara. Los tamaños típicos de lente de una cámara comercial están entre 35-65mm. En este caso, la cámara utilizada en el proyecto es Axis 205, la cual tiene un lente con diámetro de 4 mm. Es por eso que se consideró válida la aproximación del diámetro del lente de la cámara, a un punto infinitesimal. Mientras más pequeño sea el diámetro del lente, más fina será la imagen, pero se requerirá más tiempo de exposición para obtenerla. (19)

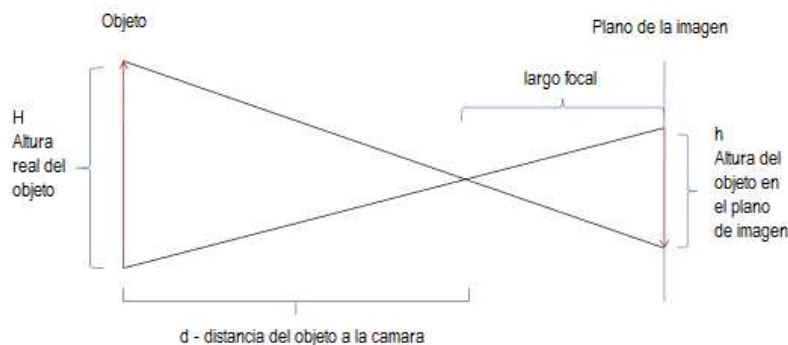
El pin-hole es el punto donde los rayos de luz provenientes de la escena son refractados y proyectados en el plano de la imagen. El objeto en el plano de imagen aparece invertido. Obsérvese la idea a grandes rasgos en la figura 33. (19)



**Figura 33 - Modelo Pinhole para una cámara (19)**

Este modelo de la cámara aproxima su lente (foco) como un punto infinitesimal por el cual penetra la luz. Esta es plasmada sobre un plano que se encuentra por detrás del foco y se denomina el *plano de imagen*. La imagen plasmada en dicho plano aparece escalada e invertida de cómo se encuentra el objeto en la realidad. (19)

En la figura 34 se muestra la asignación de variables en el modelo y cómo el modelo simplifica el problema de medición de distancias a un problema de triángulos. (19)



**Figura 34 - Relaciones de triángulos en un modelo Pinhole (19)**

Para computar la distancia se encuentra una relación entre los triángulos mostrados, la cual se presenta en la figura 35. (19)

$$\frac{H}{d} = \frac{h}{f}$$

**Figura 35 – Relación de Triángulos en modelo pin-Hole**



# Capítulo 4

## *Delimitación del Tema*

*“Nadie puede saber el límite de sus fuerzas hasta que las pone a prueba”*

*Goethe*

---

El megaproyecto *Robot explorador* se realizó durante los últimos tres semestres de la carrera de Ingeniería Electrónica, período comprendido entre Julio 2007 y Noviembre 2008. Ésta es la continuación del megaproyecto desarrollado en 2005 llamado *Diseño y construcción de un sistema explorador robotizado y autártico*. El equipo de trabajo que daría continuación al proyecto, esta conformado por 4 estudiantes de Ing. Electrónica. Cada estudiante es responsable de un módulo:

- a. Control
- b. Energía Alternativa
- c. Brazo Robótico
- d. Visión por computadora

El objetivo de dar seguimiento al proyecto es mejorar el sistema de Control y Visión por computadora del robot y agregarle dos módulos nuevos: Energía Alternativa por celdas solares y Brazo robótico. Además que se cambió el objetivo general del robot; de un robot explorador del terreno marciano a un robot limpia playas.

Respecto al módulo de Visión por computadora, tema del presente trabajo de graduación, se realizaron tres tipos de actividades a lo largo de los 18 meses que duró la realización del proyecto de graduación, los cuales culminaron con la mejora del sistema de visión por computadora del robot. Estas actividades son:

- I. Investigación de Antecedentes
- II. Diseño
- III. Implementación

El presente trabajo de graduación comprende la investigación realizada, diseño e implementación del módulo de *Visión por computadora* del robot explorador autómatas *Rover*.

Se concentra en el procesamiento y análisis de imágenes, provistas por una cámara colocada sobre el robot, para determinar la trayectoria a seguir del mismo, al desplazarse por un ambiente tipo laboratorio, evadiendo obstáculos que se encuentren en su camino.

El algoritmo de detección de obstáculos requiere que la luz sea uniforme en el terreno que analiza y que los obstáculos sean mucho más grandes que la basura que recoge.

El módulo de *Visión por computadora* debió establecer un canal de transmisión de datos inalámbrico, a través de una red WLAN, entre la estación central y el robot. Esto fue implementado utilizando microcontroladores PIC y el control WinSock de Visual Basic.

El módulo de *Visión por Computadora* controla al robot procesando imágenes en la estación central, transmitiendo la información obtenida del análisis, a través de la red WLAN, al PIC de telecomunicaciones en el ROVER que a su vez transfiriere la información recibida al PIC maestro del *Rover*, el cual mueve al robot. El PIC maestro del *Rover* es controlado por el *módulo de control*.

El módulo de *Visión por Computadora* le indica al brazo robótico, el momento en el que debe recoger basura, asegurándole una distancia prudente para que el brazo pueda alcanzarla. Toda instrucción dirigida al brazo robótico es transmitida al PIC maestro del *Rover* para que éste la transmita a su vez al PIC que controla el brazo. Para este fin se desarrollaron algoritmos que le permiten al sistema de visión por computadora verificar cierta proximidad entre el robot y la basura objetivo. También se desarrollaron algoritmos que permiten verificar que en efecto, el brazo recoja la basura cuando se le indique. El brazo es dirigido y controlado por el *módulo de brazo robótico*.

Todo procesamiento se realiza en la estación central, la cual no es más que una computadora que se encuentra fuera del robot. Los algoritmos que procesan la imagen fueron implementados en C++ y compilados en una DLL. Estos algoritmos se invocan desde una GUI, en la estación central, desarrollada en Visual Basic. La GUI transmite la información hacia un PIC de telecomunicaciones en el Rover a través de una red WLAN.

En resumen, las tareas a realizar por el modulo descrito en el presente trabajo de graduación son las siguientes:

1. Analizar una imagen provista por la cámara colocada sobre el Rover, para encontrar basura y obstáculos en ella, y así poder dirigir al robot.
2. Desarrollar un algoritmo de detección de obstáculos y aproximación de la distancia entre un objeto en la imagen y la cámara.
3. Establecer un canal de transmisión de datos inalámbrico utilizando una red WLAN.
4. Controlar al robot y al brazo mecánico mediante el desarrollo de una rutina para recolectar basura.

# Capítulo 5

## *Diseño experimental y Resultados*

*“Si buscas resultados distintos, no hagas siempre lo mismo.”*

*Albert Einstein*

---

El megaproyecto *Robot Explorador*, en específico el módulo de *Visión por Computadora*, fue desarrollado durante los últimos 18 meses de la carrera de Ing. Electrónica en la UVG. Durante dicho período se desarrolló una serie de experimentos con el objetivo final de implementar el sistema de visión por computadora de un *robot explorador autónoma, recolector de basura*.

A continuación se presentan los varios experimentos que culminaron con la mejora del sistema de visión del Rover. Se explican detalladamente las herramientas, estrategias y métodos utilizados en cada uno de ellos, así como los resultados parciales de cada experimento.

Los experimentos presentados guardan un orden cronológico y fueron sucesivos en el desarrollo global de éste proyecto.

### Lista de Experimentos:

1. *Evaluación del funcionamiento actual del sistema de visión por computadora y del sistema de control a través de internet*
2. *Manejo técnico de la cámara de red Axis 205*
3. *Acceso remoto a la cámara de red a través de una WLAN*
4. *Elección de formato en el que se procesaría la imagen: JPEG, BMP*
5. *Transformación de una imagen de JPEG a BMP: Vic32.dll*
6. *Lectura del archivo binario de una imagen en formato BMP: Rover.dll*
7. *Localización de obstáculos y eliminación de ruido en la imagen*
8. *Aproximación de la distancia entre la cámara y un objeto en la imagen*
9. *Comunicación Inalámbrica entre el robot y la estación central: IEEE 802.3*
10. *Protocolo de comunicación entre estación central y PIC maestro*
11. *Definición de objetos a recolectar: basura removible*
12. *Algoritmo de búsqueda del Rover*
13. *Pruebas finales: Integración de módulos*

A continuación se detalla el diseño experimental y los resultados obtenidos de los experimentos realizados:

## ***1. Evaluación del funcionamiento actual del sistema de visión por computadora y del sistema de control a través de internet***

---

### ***1.1 Diseño experimental***

Se contactó a Juan Fernando Mancilla. Él fue el estudiante que en el proyecto 2005 trabajó en el módulo de procesamiento de imágenes. Se ejecutaron y analizaron los algoritmos que Mancilla había desarrollado en 2005 para procesar la imagen adquirida por una cámara web colocada sobre el terreno sobre el cual se movilizaría el robot. Con ellos se desarrolló un programa que manipula y permite procesar las imágenes obtenidas de por cámara web tipo TWAIN. Todo esto se realizó con el objetivo de evaluar la eficacia del método utilizado por Mancilla.

Paralelamente, se contactó a Luis Mijangos. Él fue el estudiante que en el proyecto 2005 trabajó en el módulo de control del robot a través de internet. El interés de conocer el trabajo realizado por Mijangos fue debido a que en su trabajo utiliza la cámara de red colocada sobre el *Rover* para transmitir video en tiempo real a la aplicación web del usuario, con el objetivo de que éste tenga una vista desde la perspectiva del robot.

Las herramientas utilizadas fueron: Visual Studio, en específico visual Basic, la cámara de red Axis 205, una computadora con tarjeta de red e *internet Explorer* instalado y una webcam *lifecam vx-1000*.

### ***1.2 Resultados***

Los resultados obtenidos de las reuniones con Mancilla fueron en su mayoría fueron descripciones verbales de su trabajo, entre ellas, sus limitantes, sus debilidades, fortalezas, recomendaciones, justificaciones y estrategias del porque había realizado el procesamiento de imágenes en 2005 tal como lo hizo. Además se tuvo una sesión en donde se explico y demostró detalladamente el funcionamiento del algoritmo construido en 2005. Éste efectivamente convertía una imagen adquirida por la cámara web, a tonalidades de grises y mediante el método *cany*, se reconocían bordes en ella. Una característica importante, que destacó Mancilla acerca de su trabajo, es que el procesamiento se realizaba con imágenes

tomadas desde una webcam suspendida sobre el terreno por el cual se movilizaría el robot. La webcam capturaba imágenes simulando un satélite, que en la misión serviría para localizar al robot en el terreno.

Mancilla utilizaba un módulo ActiveX en Visual Basic, llamado “*csXImage*” para adquirir imágenes de la webcam y creó una aplicación ejecutable en Visual Basic la cual ordenaba a la webcam, conectada por USB, tomar una fotografía de 24 bits. Seguido esta fotografía era convertida a escala de grises de 8 bits, se le aplicaba un filtro Gaussiano para eliminar información no requerida, llámese *ruido*, se le aplica un filtro Sobel para detección de contornos y se genera un mapa en archivo de texto de ceros ‘0’ y unos ‘1’ los cuales indican nodos transitables y no transitables respectivamente. Este archivo era enviado al módulo de control para determinar la trayectoria a seguir.

Respecto a las técnicas que utilizó Mancilla: El filtro Gaussiano es utilizado para suavizar las imágenes, su efecto es el de hacer que los pixeles tiendan a verse como sus vecinos. Este filtro elimina o reduce los cambios bruscos de intensidad entre pixeles adyacentes, por lo que se podría clasificar como un filtro pasa baja. El filtro Sobel utiliza una aproximación discreta de la definición de la derivada direccional para la detección de contornos. Además la inteligencia artificial estaba separada en varios módulos cuyo conjunto permitía al robot movilizarse en el terreno, entre ellos: módulo de visión por computadora, módulo de algoritmo de trayectoria, módulo de sensores y módulo de control.

Como ya se mencionó se desarrolló una aplicación para evaluar la eficiencia del método de Mancilla. La aplicación desarrollada permitía mover una cámara colocada sobre una estructura y permitía analizar las imágenes provistas por la cámara. El proyecto constaba de tres partes fundamentales:

1. El diseño y construcción de una estructura para mover la cámara tanto vertical como rotacionalmente a lo largo de 180°.
2. El diseño y construcción de la interfaz de usuario en la computadora. Esta interfaz consta de dos recuadros donde se colocan las dos fotografías que se toman automáticamente cuando el usuario quiere tomar una vista *panorámica*, lo que significa

que una fotografía se toma a 45 grados y la otra en los 135 grados del rango de vista de la cámara. Además existe la función *reconocimiento de objetos* en la cual la cámara baja al nivel donde se encontrarían los objetos, toma la fotografía y luego sube otra vez para estar lista para la siguiente fotografía panorámica.

3. Se implementó que la cámara pudiera transmitir video y adicionalmente el usuario tiene control de la estructura por lo que puede mover la cámara. Adicionalmente se incluyó una función, aparentemente inservible en este proyecto, que era un área de reconocimiento de píxeles en la fotografía, el usuario puede colocar en cualquier lado de las dos fotografías su mouse y se indica el desglose de las componentes RGB del color que tiene dicho píxel.

El tema del trabajo de graduación presentado por Luis Mijangos en 2005 fue el control del robot vía internet. En el trabajo de graduación se plantea la solución que se encontró en ese entonces para controlar el robot vía Internet. Mijangos desarrolló una aplicación Web utilizando la plataforma *Java* desde la cual se podría controlar el robot manualmente. Además se implementó la transmisión de video a través de la cámara Axis 205 instalada sobre el vehículo para ayudar al usuario a mover el robot.

Se programaron dos sesiones con Mijangos, en una de las cuales se conectó la cámara web a la tarjeta de red de una laptop y se pudo visualizar video utilizando el *Internet Explorer*.

## **2. Manejo técnico de la cámara de red Axis 205**

---

### **2.1. Diseño experimental**

Se debía aprender a utilizar la cámara de red para poder disponer de una imagen para analizar en la estación central. La razón de ser del presente experimento es el configurar la cámara de red, tales como direcciones TCP/IP y MAC, y acceder a ella a través de una red. Con el fin de poder acceder a ella cuando esta esté conectada a una red. Se buscó el manual del modelo: Axis 205, sin embargo como este modelo ya estaba



descontinuado se tuvo que hacer equivalencias utilizando la hoja de especificaciones de otra cámara de su generación.

## 2.2. Resultados

Los datos técnicos relacionados al modelo de cámara de red, provistos por la compañía Axis se pueden encontrar en el capítulo de *marco teórico* (ver pag.15).

Se conectó la cámara a un router inalámbrico y se transmitió video por la red WLAN utilizando la aplicación que trae por default la cámara, accesible a través del directorio ftp de la cámara. El video pudo ser visto exitosamente utilizando el explorador de internet de una computadora conectada a la red inalámbrica.

Algunos datos importantes respecto a la conexión y configuración de la cámara Axis 205: la cámara Axis 205 se alimenta con 5V DC; una vez se conecta la cámara a la alimentación se debe esperar aproximadamente 15 segundos para que los Leds colocados a la par del lente se enciendan cambiando primero a rojo después a naranja y finalmente a verde, indicando que la cámara ya está funcionando.

Los pasos a seguir para configurar la cámara son los siguientes:

1. Se debe conectar la cámara a 5V DC. Cuando se haga esto una serie de Leds que se encuentra en la parte de atrás indicará que la cámara detecto la corriente y los Leds de la parte frontal de la cámara comenzaran a cambiar de color (rojo – naranja – verde).
2. Se debe reiniciar la cámara. En la parte trasera de la cámara en medio de dos Leds se encuentra un botón negro, este se debe presionar, hasta que los Leds que se encuentran alrededor del lente se enciendan. Esto indica que la cámara ya está funcionando.
3. Se conecta el cable de red. El otro Led que se encuentra a la par del Led que indica el encendido, en la parte trasera de la cámara, deberá comenzar a titilar indicando que la cámara está conectada a la red.

4. Se debe Iniciar el cmd en Windows para poder configurar la dirección IP de la cámara con las siguientes líneas:

```
arp -s 192.168.0.4 00-40-8c-6A-3d-dd
```

Esta instrucción le esta asignando al dispositivo cuya dirección MAC es la *00-40-8c-6A-3d-dd* la dirección IP: *192.168.0.4*. Esta dirección es una recomendación del manual de instalación de la cámara. La dirección MAC de la cámara se encuentra en la parte de atrás de la misma.

Luego presionar ENTER. Y escribir:

```
ping -l 408 -t 192.168.0.4
```

Se comienza a llamar a la cámara, esperando respuesta. Comando con el cual verificamos que la configuración y asignación de dirección IP ala cámara se haya realizado exitosamente. Si se recibe una respuesta de ping significa que la cámara fue configurada exitosamente, si no, repetir pasos del 1 al 4.

5. Finalmente, presionar el botón de RESET, desconectar alimentación DC, esperar 5 segundos y volver a conectar la cámara aún presionando RESET.

Ahora la cámara ya está configurada y se puede acceder a ella con cualquier browser de internet, escribiendo la dirección IP que se le asigno, en este caso: *192.168.0.4*

### ***3. Acceso remoto a la cámara de re a través de una WLAN: AxisMediaControll.dll***

---

#### ***3.1. Diseño experimental***

Se requería utilizar una plataforma de programación para desarrollar la aplicación. Se investigó en la compañía que realizó la cámara *Axis* para obtener algún tipo de información. Se encontró que la compañía proveía de una librería que permitía controlar la cámara de red: *AxisMediaControll.dll*. Con ella, se realizó una aplicación en Visual Basic que toma automáticamente una fotografía de la cámara *Axis 205* desde el *Rover* y a través de una red WLAN y la almacena en una carpeta en formato JPG.

### 3.2. Resultados

La librería que provee la compañía es *AxisMediaControll.dll*. Ésta es descrita en el *Marco Teórico* pág. 17. La aplicación se desarrolló en *Visual Basic*. Al ejecutar la aplicación se toma automáticamente la fotografía, accediendo a la cámara a través de la red WLAN. Se almacena la fotografía en la carpeta en donde se ejecuto la aplicación, con el nombre de “*test.jpg*”. Se tiene también un método de “*pausa*” utilizado para que le de tiempo al hardware de la cámara para prepararse antes de capturar la imagen. Ésta fue la aplicación base sobre la cual se desarrollaron el resto de algoritmos.

## 4. ***Elección de formato en el que se procesaría la imagen: JPEG, BMP***

---

### 4.1. Diseño experimental

La cámara de red Axis 205 entrega una imagen en formato JPEG al programa en la estación central, este tipo de compresión genera un archivo pequeño, permitiendo que sea fácilmente transportado por la red. Es necesario conocer cómo esta estructurado el archivo binario de la imagen JPEG. Por ello, se investigó la compresión JPEG. Paralelamente se buscó otras soluciones y se pensó en el formato genérico de Windows: BMP.

### 4.2. Resultados

Ver capítulo de *Marco Teórico* pág. 23 para la explicación en detalle de dichos formatos. Al concluir con dicha investigación se determinó que el formato JPEG es muy complicado para leer y por lo tanto, para procesar la imagen, sin embargo el formato BMP presenta la información referente al color del pixel sin ningún tipo de compresión y además en un formato muy simple.

## 5. Transformación de una imagen de JPEG a BMP: *Vic32.dll*

---

### 5.1. Diseño experimental

Dado que se determinó se debía transformar la imagen de formato JPEG a formato BMP, la estrategia a utilizar para ello fue: buscar un método comercial o alguna librería comercial que lo hiciera, dado que la transformación de imágenes no es el tema central del presente trabajo. Se realizó una exhaustiva investigación de librerías comerciales o métodos que realizaran dicha conversión.

### 5.2. Resultados

Realizando investigación se encontró una librería muy famosa en los ámbitos de procesamiento de imágenes, llamada *vic32.dll*. Las características de la librería se muestran en el capítulo *III. Marco Teórico* pág. 36.

Con ella, se realizó un programa que convierte la imagen JPEG a formato BMP, en Visual Basic utilizando la librería comercial, llamada “*vic32.dll*”. Esta es una librería provista por la compañía *Catenary*. Los archivos DLL (dynamic link library) son un aspecto importante de Windows. Un DLL contiene funciones a las que puede llamar un programa durante su ejecución. Un DLL es una biblioteca de funciones con las que se puede vincular el programa dinámicamente, es decir, mientras se ejecuta como se realizó en el programa desarrollado. El resultado final, fue la conversión de la imagen provista por la cámara en formato JPEG a formato BMP de 24 bits. Para ello se utilizaron cuatro métodos de dicha librería: *jpeginfo*, *allocimage*, *loadjpg* y *savebmp*.

Fue necesaria la declaración de los métodos, en un *módulo* del proyecto de Visual Basic. Esto es necesario para importar los métodos de la DLL y poderlos ejecutar desde el programa. Además se declararon dos estructuras, “*imgdes*” y “*jpegData*”, también utilizadas por la librería.

Los resultados de dicho procesamiento se muestran en las figuras 36 y 37.



Figura 36 - Imagen enviada por la cámara en formato JPEG (test.jpg)



Figura 37 - Imagen transformada a formato BMP (test.bmp)

## 6. Lectura del archivo binario de una imagen en formato BMP: rover.dll

---

### 6.1. Diseño experimental

Existe una librería muy útil al leer archivos binarios, ésta es provista por Windows, llamada *stdio.h*. Esta librería permite manipular la lectura de archivos binarios. Se pensó desarrollar una librería en C++ que permitiese leer el archivo binario de la imagen, utilizando *stdio.h*, al invocar sus métodos desde Visual Basic. Esta librería lleva el nombre de *rover.dll*. Los métodos que leen el archivo binario la imagen BMP fueron implementados en C++.

## 6.2. Resultados

La implementación en Visual Basic, lee el archivo binario de la imagen invocando métodos de la librería *rover.dll* creada en C++

La librería de métodos para leer archivos BMP, fue realizada en C++. Los métodos fueron compilados en una DLL. Ésta se crea en visual C++ 6.0, con los comandos file > new > Win32 Dynamic-Link Library (dll) > Create an empty DLL Project. A este proyecto en blanco se debe agregar un “C++ Source File (\*.cpp)”: file > new > C++ source file. En dicho \*.cpp se deben escribir los métodos a compilar en la DLL.

La implementación permitió leer la imagen y modificarla como se quisiera. Previo a esto se debió investigar la estructura del formato de archivo de una imagen BMP. Ver III. Marco Teórico pág. 32.

## 7. ***Localización de obstáculos y eliminación de ruido en la imagen***

---

### 7.1. Diseño experimental

La estrategia utilizada fue: observación. Se encontró un patrón en los colores de los pixeles que contenían un área con obstáculo y un área de paso. Utilizando el software *Photoshop* se pudo apreciar como en un pixel, perteneciente a un área de obstáculo, predominaban los colores oscuros, en especial la componente azul dominaba sobre las otras dos componentes. Por el contrario al ser un pixel de área de paso, la componente amarilla predominaba sobre las otras 2. Así se procedió a realizar un algoritmo en C++ que utilizara dichas características vistas en las componentes para reconocer pixeles de área de obstáculo o pixeles de área de paso. La estrategia se basó en dar pesos distintos a cada una de las componentes de un pixel en la fotografía. En la imagen que se obtuvo del procesamiento de reconocimiento de obstáculos, se tenía un efecto conocido en el área de procesamiento digital como de *sal y pimienta*. Éste debía ser eliminado, por lo que se analizaron distintos filtros que realizarían el trabajo.

## 7.2. Resultados

Como ya se dijo, se noto en la fotografía, transformada a formato BMP, que los pixeles tenían distintos valores en sus tres componentes RGB, según se tenía un lugar de paso u obstáculo. Las componentes azul, verde y roja al ser un pixel de obstáculo, tenían valores pequeños, generalmente por debajo de 100. Y cuando se trata de un lugar de paso estas tomaban valores altos, generalmente arriba de los 150. Mediante el análisis de un patrón se desarrolló el algoritmo que se llamó *BlueYellowScale ()* dentro de la librería *rover.dll*. La parte crucial del algoritmo se presenta a continuación.

```
while (!feof(archivo))
{
    fread(&Paleta,3,1,archivo);          // leer una secuencia de 3 bytes, RGB
    intensidad = (30*int(Paleta.Red) + 60*int(Paleta.Green) + 10*int(Paleta.Blue)); // asignar pesos
    gris = int(intensidad/100);           // normaliza el nuevo color
    fwrite(&gris,3,1,salida);             // almacena el nuevo color en la imagen
}
```

El objetivo del algoritmo es asignar cierto peso a cada una de las componentes que representan la imagen. La componente azul se acentuaba más en un área de obstáculo y la verde en un área de paso. Una vez aplicado dicho filtro se obtuvo una imagen donde se indicaban con tonos azules los obstáculos y con tonos amarillos los lugares de paso como se muestra en la figura 38.



Figura 18 - Imagen después de ser aplicado el filtro BlueYellowScale (blueyellf.bmp)

Para eliminar el efecto de *sal y pimienta* que tiene la fotografía mostrada en la figura 38 se aplicó un filtro reductor de ruido. El filtro de mediana es un filtro útil para reducir el ruido de sal y pimienta de las imágenes. Es un filtro digital, no lineal, utilizado para extraer el ruido de las imágenes. La idea básica es revisar un pixel y decidir, al evaluar sus alrededores, si este es representativo a la señal. Los filtros de suavizado lineales o filtros pasa baja tienden a "difuminar los bordes" a causa de que las altas frecuencias de una imagen son atenuadas. Se aplicó sobre la imagen el filtro de mediana de la librería *vic32.dll* llamado *load\_medianfilter\_save()*.

Se desarrollo tambien un filtro de mediana propio, pero dado que el algoritmo desarrollado era demasiado lento, en el rango de 1-3 min dependiendo de el radio de filtrado, se opto por utilizar el filtro provisto en la librería *vic32.dll*, el cual es mucho mas veloz. Sin embargo se dejara el código del otro filtro realizado, en caso sea requerido, en los anexos adjuntos en CD.

Nótese en la figura 39 que con la imagen obtenida después de ser aplicado el filtro de mediana, se obtiene una imagen con los bordes de objetos bien definidos. Por lo que esta lista para pasar a la siguiente etapa de procesamiento.



Figura 39 - Imagen después de ser aplicado el filtro de mediana



## 8. *Aproximación de la distancia entre la cámara y un objeto en la imagen*

---

### 8.1. *Diseño experimental*

Se debió inicialmente encontrar un modelo matemático para representar la cámara y poder incluirla dentro de un cálculo que permitiese la aproximación de distancias. Se requería de un modelo simple que permitiese un cálculo relativamente sencillo. Se investigaron los distintos modelos que representan a una cámara y se eligió un modelo sencillo que describía la cámara con el suficiente detalle para el fin de éste trabajo de graduación.

### 8.2. *Resultados*

El modelo utilizado y sobre el cual se trabajaron las ecuaciones para el cálculo de distancia es el llamado: “pin-hole model” (en español “modelo estenopéico”) descrito en el capítulo III. Marco Teórico (pág. 41). Este modelo es uno de los más utilizados en visión robótica, por su simplicidad. Este modelo desprecia el diámetro focal y lo aproxima como un punto infinitesimal. En este caso, la cámara utilizada en el proyecto es la Axis 205, tiene un lente de diámetro de 4mm, y es por esto que se considero como válida, la aproximación del diámetro del lente de la cámara, a un punto infinitesimal. La ecuación que describe la relación de triángulos del modelo es el siguiente:

$$\frac{H}{d} = \frac{h}{f}$$

Para determinar la distancia existente entre un objeto en la imagen y la cámara que adquirió la imagen fue necesario plantear dos ecuaciones con dos incógnitas, el sistema de ecuaciones se presenta a continuación:

Fotografía 1

$$\frac{H}{d_o} = \frac{h_1}{f}$$

Fotografía 2

$$\frac{H}{d_o + D} = \frac{h_2}{f}$$

$$H = d_0 \frac{h_1}{f} \quad H = \frac{h_2}{f} (d_0 + D)$$

$$d_0 h_1 = h_2 (d_0 + D)$$

$$d_0 [cm] = \frac{h_2 [pix] D [cm]}{h_1 [pix] - h_2 [pix]} = \frac{h_2 D}{h_1 - h_2} [cm]$$

Nótese que no necesitamos la distancia focal ya que se cancela en los cálculos. Importante ya que en la datasheet de la cámara esta información no aparece.  $h_1$ ,  $h_2$  deben ser medidos en pixeles.

Se incluyen a continuación dos resultados, la primera con una cámara digital comercial y la segunda con la cámara Axis del ROVER.

Distancia real = 30cm de la cámara al objeto

$h_1 = 180$  pixels,  $h_2 = 132$  pixels,  $D = 10$ cm

Distancia aproximada 27.5cm.

Fotografía 1



Fotografía 2



Figura 40 - Conjunto de fotografías que permiten la medición de distancias

Distancia real = 90cm de la cámara al objeto

$h_1 = 270$  pixels,  $h_2 = 241$  pixels,  $D = 10$ cm

Distancia aproximada 83.1cm

Fotografía 1



Fotografía 2

**Figura 41 - Conjunto de figuras que permiten la determinación de la distancia del objeto en la imagen**

El procedimiento para la determinación de distancias es el siguiente:

1. Se toman las dos fotografías una separada de la otra 10cm
2. Se procesan ambas
  - a) Se convierten las .jpg a .bmp
  - b) Se les aplica el primer filtro pasa altas, para detección de objetos (BlueYellowScale() de la librería rover.dll)
  - c) Se les aplica un filtro pasa bajas (medianfilter()); de la librería vic32.dll) con el objetivo de eliminar el ruido de la imagen después de haberla convertido a tonos azules y amarillos.
3. Se escanean las dos imágenes en busca del primer obstáculo que se encuentre en frente del robot por áreas mostradas por la plantilla de la figura 40. El área central esta comprendida desde 256px a 384px, es aproximadamente desde un 0.4 aun 0.6 del ancho de la fotografía (640px).

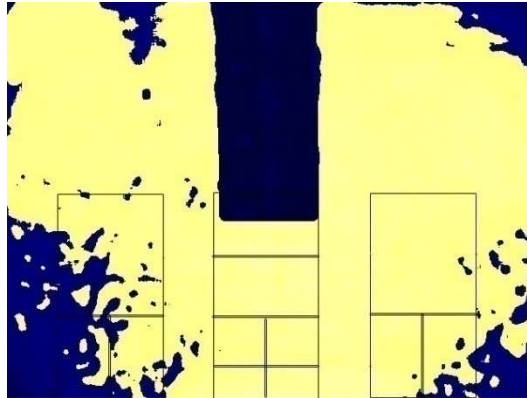


Figura 42 - Áreas por las cuales se escanea la fotografía

Finalmente se determinan  $h_1$  y  $h_2$ , para computar la distancia  $d_0$ .

## 9. Comunicación Inalámbrica entre el robot y la estación central: IEEE 802.3

---

### 9.1. Diseño experimental

Se debía establecer un canal de comunicación entre la estación central y el robot para retornar la información adquirida del análisis de la imagen. Se requería que dicha comunicación fuera inalámbrica, ya que es poco práctico que ésta fuera alámbrica ya que el robot debería acarrear dicho cable por el terreno por donde se moviliza. Se pudo haber construido un canal de comunicación propio, sin embargo se pensó que se podría tener un canal más eficiente si se utilizaba uno que ya estuviese definido y cuyos componentes estuviesen en el alcance del presupuesto. Se pensó en utilizar el estándar IEEE 802.3 en su versión inalámbrica: IEEE 802.11, éste es *wireless Ethernet*. Entre los componentes necesarios para establecer este tipo de comunicaciones estaban un router inalámbrico y una tarjeta computadora con tarjeta compatible al estándar IEEE 802.11. Adicionalmente se encontró un módulo que permitiese este tipo de conexión al PIC de telecomunicaciones del robot y a la vez en el programa de Visual Basic en la estación central

### 9.2. Resultados

Se investigó y se encontró el controlador ENC28J60 que permite conectar un PIC con módulo SPI a una *Ethernet IEEE 802.3*. Del lado de la estación central se utilizó una

laptop que incluía una tarjeta de red compatible con el protocolo *IEEE 802.11* la cual se conectaba a la red inalámbrica del router colocado sobre el robot.

Se estableció exitosamente una comunicación, a través de una red WLAN utilizando el protocolo UDP/IP, entre el programa en la computadora central y el PIC maestro del Rover utilizando como intermedio un PIC de red (ENC28J60) y un PIC de telecomunicaciones (18F2520). En Visual Basic se utilizó el socket WinSock, descrito en el *Marco teórico* (ver pág. 23), para comunicar el programa con la red WLAN. Los PIC intermediarios, en el Rover, son una interfaz entre la red WLAN y el PIC maestro del Rover. Al router se conecta el PIC de red y éste a su vez se conecta con el PIC de telecomunicaciones utilizando el módulo SPI. El PIC de telecomunicaciones utilizado fue el PIC18F2520 de 28 pines. Este traduce a USART la información recibida por SPI del PIC de red, para el PIC maestro del *Rover*. El PIC de red utilizado fue el ENC28J60. Este es un controlador de Ethernet con interfaz SPI. Diseñado para funcionar como interfaz Ethernet para cualquier controlador con equipado con interfaz SPI. Este cumple con todas las especificaciones del estándar IEEE 802.3. La comunicación con el microcontrolador es realizado mediante los pines del módulo de SPI.

## ***10. Protocolo de comunicación entre estación central y PIC maestro***

---

### ***10.1 Diseño experimental***

El protocolo de comunicación debía ser funcional tanto para el módulo de control como para el módulo de visión por computadora. Éste fue realizado en conjunto entre ambos módulos. Se debía definir el modo de entablar una conexión entre la estación central y el PIC de telecomunicaciones. Además se implementó el chequeo de errores para que el canal de comunicaciones funcionara adecuadamente.

## 10.2 Resultados

Se trabajó junto al módulo de control para definir el protocolo a utilizar. El protocolo será orientado a byte, no orientado a conexión con ACK. Las instrucciones dirigidas hacia el brazo fueron elegidas según sería el algoritmo de funcionamiento general.

**Tabla 3 - Código enviados por el PIC maestro del ROVER hacia la estación central**

Código binario	Código decimal	Procedencia	Instrucción
11011100	220	Rover	Movimiento Terminado
11011101	221	Rover	Error en el tiempo de realización del movimiento
11011110	222	Rover	Error por la solicitud de movimiento y hay problema con la temperatura
11011111	223	Rover	Error por la solicitud de movimiento hacia adelante y hay un obstáculo
	...	x	
11100001	225	Rover	Error en la temperatura (movimiento o espera) motor 1
11100010	226	Rover	Error en la temperatura (movimiento o espera) motor 2
11100011	227	Rover	Error en la temperatura (movimiento o espera) motor 3
11100100	228	Rover	Error en la temperatura (movimiento o espera) motor 4
11100101	229	Rover	Error en la temperatura (movimiento o espera) motor 5
11100110	230	Rover	Error en la temperatura (movimiento o espera) motor 6
	...	x	
11101000	232	Rover	Error en un obstáculo (movimiento o espera) delantero izquierda
11101001	233	Rover	Error en un obstáculo (movimiento o espera) delantero central
11101010	234	Rover	Error en un obstáculo (movimiento o espera) delantero derecho
11101011	235	Rover	Error en un obstáculo (movimiento o espera) agujero izquierdo
11101100	236	Rover	Error en un obstáculo (movimiento o espera) agujero derecho
	...	x	
11110000	240	Rover	Posición Actual (el siguiente dato tiene la posición)
11110001	241	Rover	Posición Inicial (el siguiente dato tiene la posición)
11110010	242	Rover	Posición Final (el siguiente dato tiene la posición)
	...	x	
11110110	246	Brazo	Movimiento terminado: Agarrar y levantar
11110111	247	Brazo	Movimiento terminado: Estirar y búsqueda
	...	x	
11111100	252	Rover	Detener al Rover, cancelar toda instrucción previamente enviada
11111111	255	Rover	Error en la comunicación (la confirmación no fue la esperada)

*Nota: x = no utilizadas*

Tabla 4 – Códigos enviados desde la estación central hacia el PIC maestro del ROVER

Código binario	Código decimal	Destino	Instrucción
101	5	Rover	Confirmación de la instrucción enviada.
1000	8	Rover	Solicitud de la posición actual
1010	10	Rover	Solicitud de giro hacia la Derecha de 0 a 345 grados en intervalos de 15
...	...		
100001	33		
100010	34	Rover	Solicitud de giro hacia la Izquierda de 0 a 345 grados en intervalos de 15
...	...		
111001	57		
111011	59	Rover	Solicitud para ir hacia adelante de 5 a 100 centímetros en intervalos de 5
...	...		
1001111	79		
1010000	80	Rover	Solicitud para ir hacia atrás de 5 a 100 centímetros en intervalos de 5
...	...		
1100100	100		
	...	x	
1101110	110	Rover	Solicitud de todas las temperaturas medidas en el explorador
1111000	120	Rover	Solicitud de los obstáculos alrededor del explorador
	...	x	
10000010	130	Brazo	Agarrar y levantar con ACK
10000011	131	Brazo	HOME
10000100	132	Brazo	Estirar y búsqueda con ACK
	...	x	
11111111	255	Rover	Error en el carácter de confirmación.

*Nota: x = no utilizadas*

A continuación se describen los comandos utilizados para comunicación entre el modulo de control y visión por computadora.

### ***Instrucciones entre módulo de control y visión por computadora***

Se debió llegar a un acuerdo de las señales que podría el módulo de *Visión por computadora* enviar al módulo de control, para mover al Rover.

Se definieron 5 comandos:

#### *1. mover(+d) con ACK*

Comando utilizado para mover al Rover hacia delante.

- Limites de distancias que se puede mover el ROVER: Min distancia = 5cm, Max distancia = 80cm.
- Instrucciones de protocolo: 60-79, donde cada instrucción incrementa +5cm la posición del Rover.
- El ACK implementado deberá indicar al módulo de *Visión por computadora* que el Rover ha terminado la tarea que le fue asignada.

#### *2. mover(-d) con ACK*

Comando utilizado para mover al Rover hacia atrás, generalmente para los 10 cm que necesita el método de medir distancias.

- Limites de distancias que se puede mover el ROVER: Min distancia = 5cm, Max distancia = 80cm.
- Instrucciones de protocolo: 81-100, donde cada instrucción es un decremento de -5cm a la posición del Rover.



- El ACK implementado deberá indicar al módulo de *Visión por computadora* que el Rover ha terminado la tarea que le fue asignada.

### 3. *stop*

Comando de paro al Rover (motores y brazo), no importando si este ha cumplido o no la tarea que le fue asignada o tiene otra tarea en proceso o espera.

### 4. *giro(derecha) con ACK*

Comando utilizado para girar al Rover hacia la derecha de la cámara.

- Limites de giro del ROVER: Min giro =  $15^{\circ}$ , Max giro =  $345^{\circ}$ .
- Instrucciones de protocolo: 11-33, donde cada instrucción es un giro de  $+15^{\circ}$  sobre su eje a la derecha, a la posición original del Rover.
- El ACK implementado deberá indicar al módulo de *Visión por computadora* que el Rover ha terminado la tarea que le fue asignada.

### 5. *giro(izquierda) con ACK*

Comando utilizado para girar al Rover hacia la izquierda de la cámara.

- Limites de giro del ROVER: Min giro =  $15^{\circ}$ , Max giro =  $345^{\circ}$ .
- Instrucciones de protocolo: 35-57, donde cada instrucción es un giro de  $+15^{\circ}$  sobre su eje a la izquierda, a la posición original del Rover.

- El ACK implementado deberá indicar al módulo de *Visión por computadora* que el Rover ha terminado la tarea que le fue asignada.

### ***Instrucciones entre el módulo de brazo robótico y visión por computadora***

Esta rutina tenía limitaciones en cuanto a que se debió llegar a un acuerdo de las señales que podría el módulo de *Visión por computadora* enviar al módulo del brazo, para recoger basura. Realmente las órdenes no son enviadas directamente del módulo de *Visión por computadora* al módulo del brazo, sino del módulo de *Visión por computadora*, al *módulo de control* y al *módulo del brazo*. Se definió que el módulo de inteligencia artificial solo se comunicara con el módulo de control, para tener centralizado el control del robot. Se acercará al Rover a aproximadamente 20cm de la basura, tal que quede a medio alcance el brazo.

Se definieron 3 comandos:

#### ***1. Agarrar y levantar con ACK***

Comando utilizado cuando el módulo de AI (Artificial Intelligence) ya aproximó que el objeto esta centrado enfrente del robot y en el rango de alcance del brazo. Por lo que envía este comando al brazo para que barra en línea recta desde su mínimo hasta su máximo de alcance y al detectar el objeto lo levante a una posición media, donde la cámara pueda ver la pinza del brazo. Cuando el brazo cree haber levantado la basura levanta el objeto a la posición media, envía ACK de que ya lo levanto a posición media, entonces la cámara toma una fotografía revisando que la basura ya no este en el suelo y la pinza del brazo tenga algo agarrado.

- Limites de alcance del brazo: Min alcance = 5cm, Max alcance = 35cm.
- Instrucción de protocolo: instrucción = 130, ACK = 246

## 2. *Home*

Comando que se envía al asegurar que el brazo ya recogió basura, por lo que se envía este comando. El brazo debe regresar al Rover, depositar la basura en el basurero que llevará el Robot y finalmente regresar a su posición inicial.

- Instrucción de protocolo: instrucción = 131

Se notaron ciertas características claves

1. El brazo deberá estar posicionado exactamente en el centro y arriba de la cámara.
2. El brazo debe incluir un sensor de presión de algún tipo ya que el solo debe poder agarrar el objeto.
3. Se debe definir la distancia máxima y mínima de barrido del brazo en línea recta

## ***11. Definición de objetos a recolectar: basura removable***

---

### *11.1 Diseño experimental*

Se debían tomar en cuenta las limitaciones para el módulo de visión por computadora como el módulo del brazo robótico. Las limitaciones y capacidades de cada módulo definieron el tipo de basura que podría recolectar el robot. Se evaluaron los tipos de basura que generalmente son encontradas en las playas después de días festivos o que el guatemalteco promedio bota en lugares públicos.

## 11.2 Resultados

Esto fue discutido entre los responsables del módulo de control, módulo de brazo robótico y módulo de visión por computadora, ya que era necesario conocer y tomar en cuenta las limitaciones que tenía cada módulo respecto a espacio y capacidad.

Los objetos definidos como basura son:

- Bolsas
  - Tipo “Bolsas metálicas”
- Cajitas pequeñas
  - Tipo “cigarro”: cartón
- Papel corrugado
  - Hojas “no blancas”: papel
- Latas
  - Tipo “aguas gaseosas”: metálicas

## ***12. Algoritmo de búsqueda del Rover***

---

*12.1 Diseño experimental*

*12.2 Resultados*

---

### ***13. Pruebas finales: Integración de módulos***

---

*13.1 Diseño experimental*

*13.2 Resultados*

# Capítulo 6

## Discusión

*“Un camino de mil millas comienza con un paso “*

*Benjamín Franklin*

---

El proyecto *Robot explorador* nace con la iniciativa de continuar el desarrollo de proyectos existentes en la universidad. Naturalmente el primer paso era, analizar las bases es decir analizar el proyecto desarrollado en 2005 y encontrar puntos clave en donde se podrían realizar mejoras. El presente módulo se relaciona con los módulos de Inteligencia Artificial y Control. Se evaluaron los trabajos de graduación correspondientes a cada módulo descrito y se encontró de interés el trabajo realizado por dos módulos. Estos fueron los trabajos de graduación denominados *Procesamiento de Imágenes* y *Control del Sistema explorador a través de Internet*, los licenciados responsables son Mancilla y Mijangos respectivamente. Se programaron reuniones con ambos licenciados. De la reunión con Mancilla y Mijangos se sacaron premisas importantes que dieron lugar al desarrollo del proyecto.

A diferencia de cómo lo realiza Mancilla en 2005, la nueva propuesta es colocar la cámara sobre el vehículo y no suspendida sobre el terreno, simulando un satélite, como se hacía en 2005. El procesamiento realizado por Mancilla está dirigido a crear un mapa del terreno, por el cual se movilizará el *Rover*, el cual es enviado posteriormente al módulo de control para que éste pueda tomar decisiones. Sin embargo, con el nuevo posicionamiento de la cámara, no se puede tener una visión global de todo el terreno, por lo que el procesamiento de imágenes deberá ser distinto. Un cambio relevante es que dado que la imagen se tomará desde el robot, es necesario controlar la cámara remotamente por el programa, es decir no puede estar contactada vía USB con la computadora central, como lo estaba en la versión de Mancilla. Esto da lugar a la necesidad de comunicar inalámbricamente al robot con la estación central, la cual tendrá la computadora que procesará la imagen.

Al desarrollar el proyecto que evaluaría la eficacia de los algoritmos desarrollados por Mancilla en 2005, se pudo notar la baja calidad de las imágenes provistas por la cámara web, adicionalmente se consideró un método poco práctico si se quiere tener la cámara sobre el robot en movimiento, asunto que con el cual Mancilla no tuvo problema alguno ya que su cámara simulaba un satélite y no estaba montada sobre el robot. Los algoritmos de análisis eran adecuados y apropiados para la aplicación al igual que cumplían con los objetivos del megaproyecto 2005.

El trabajo presentado por Mijangos en 2005, fue de interés al presente trabajo de graduación ya que involucra la cámara de red Axis 205, instalada sobre el robot. La razón de ser de dicha cámara en 2005 era presentar al usuario la vista desde la perspectiva del robot. La presente propuesta involucra el realizar todo procesamiento de imágenes del *Rover* desde el robot, utilizando dicha cámara de red. Esto fue una ventaja al hablar de costos ya que ésta cámara ya era parte del proyecto. Adquiría imágenes con la suficiente precisión para poder ser éstas analizadas. Se decidió utilizar ésta cámara para no incurrir en gastos extra y además comprobando que ésta era apta para los fines del módulo de *visión por computadora*.



Se decidió colocar la cámara que analizaría las imágenes sobre el robot ya que ésta simularía la visión del robot y facilitaría mucho el control del mismo si se analiza el terreno desde el punto de vista del robot.

La nueva propuesta, tema del presente trabajo de graduación, consiste en unir los dos sub-módulos previamente mencionados. En su esencia la nueva propuesta, elimina la cámara que simulaba al satélite y dirige todo procesamiento de imágenes a ser realizado con las imágenes provistas por la cámara de red que se encuentra sobre el Robot. Esto implica que las imágenes sean enviadas a una computadora localizada fuera del robot para realizar el procesamiento correspondiente, denominada *Estación de control*. La aplicación en la estación central permite al robot funcionar en modo autónomo y o en modo manual, en el que un usuario controla al robot. Esta nueva modalidad elimina la necesidad de realizar una exhaustiva calibración y difícil posicionamiento de la cámara respecto al terreno donde se movilizará el robot como era anteriormente.

Una vez elegida la estrategia para el control del robot se configuró la cámara Axis para acceder a ella al estar conectada en una red de área local (LAN), y así poder tomar una fotografía desde una computadora conectada a la misma red. La cámara se conectó a un router, que se encontraba sobre el robot y que a su vez se conectaba la computadora en la estación central para que pudieran comunicarse. A la cámara se le asignó una dirección IP fija clase C, dado que es la más común a asignar cuando se tienen redes de área local domiciliarias. La computadora al igual que el router en la estación central debía estar bajo la misma dirección de red, es decir sus direcciones IP debían coincidir en los primeros 3 bytes.

Se buscó apoyo en la compañía que construyó la cámara para el manejo de la misma. Se encontró que esta compañía provee una librería muy útil, que cabalmente permite el acceso a la cámara remotamente a través de una red. Ésta fue la primera interacción con el manejo de librerías. Es importante de mencionar que ésta introducción abrió la mente a pensar en compilar los métodos en librerías y solamente utilizarlos en la aplicación GUI. Tal como se hizo posteriormente con la librería creada en este proyecto llamada *rover.dll*. La librería provista por la compañía permitió que el manejo de la cámara

fuese relativamente sencillo. Con ella se implementó en Visual Basic una aplicación que permitía conectarse a la cámara y ver video o adquirir fotografías.

La imagen recibida de la cámara estaba en formato JPEG. La investigación realizada acerca de éste formato se muestra en el Marco Teórico, pág. 23 y determinó que aunque JPEG es el formato ideal para transmitir imágenes a través de la red es un formato bastante complejo de leer y decodificar. En contraste BMP es un formato genérico, muy fácil de leer y de procesar, por lo que se consideró el formato adecuado para realizar el análisis de la imagen.

Para transformar la imagen de formato JPEG a BMP se utilizó la librería *vic32.dll*, ésta es una librería comercial. La transformación de la imagen causó que ésta aumentara su tamaño. La fotografía en formato BMP obtenida es de buena calidad y de la misma cantidad de pixeles que la fotografía JPEG original. El formato de colores utilizado para almacenar la fotografía en BMP es RGB. Por lo que la descripción de un pixel consta de tres componentes: *rojo*, *verde* y *azul*. El algoritmo de conversión almacena la imagen en el mismo directorio desde donde se ejecuta el programa.

El siguiente paso requería que se leyera la imagen ya transformada, como archivo binario. Pareció muy apropiado el uso de librerías para implementar y compilar dichos métodos para luego solo invocarlos en Visual Basic. La estrategia elegida para implementar estos métodos, fue programar en C++, compilarlos en una DLL e invocar los métodos desde la aplicación que ya se venía desarrollando en Visual Basic. Todo esto por la facilidad de implementación de métodos y funciones en C++ y la facilidad de elaborar una GUI en Visual Basic. La librería que se utilizó para dichos fines es *stdio.h*.

Para reconocer objetos se desarrolló el algoritmo *BlueYellowFilter()*. Éste filtro lee todo el archivo binario de la imagen en formato BMP dando distintos pesos a cada componente RGB. Una vez aplicado el filtro la imagen obtenida marca con tonos amarillos las áreas de paso, mientras que las áreas de obstáculo son representadas con tonos azules. Después de aplicarle a la imagen en formato BMP, el filtro *BlueYellowScale()*, ésta tiene un

efecto, que se conoce en terminología de procesamiento de imágenes como, “sal y pimienta”. Se necesita que la imagen tenga bien definidos los contornos de un obstáculo para pasar a la siguiente etapa del procesamiento. Por lo que se debe eliminar este efecto que no permite definir bien los bordes de un objeto. El efecto se puede interpretar como “ruido” en la imagen. Para procesar una imagen es necesario reducir el ruido en ella antes de realizar otro tipo de procesamiento, tal como lo es localización de objetos.

El suavizado o filtrado pasa baja de una imagen elimina los cambios bruscos de intensidad de píxeles adyacentes en una imagen, es decir, atenúa las frecuencias altas, mientras se mantienen las bajas y medias frecuencias. El filtrado se alcanza mediante la convolución de la matriz del filtro con una matriz de píxeles de la imagen. Es fácil ver en la máscara que la convolución es simplemente la suma ponderada de los píxeles vecinos al píxel evaluado. Promediar tiende a eliminar los valores extremos de un grupo, así deben ser reconocidos como ruido los píxeles cuyo vecindad promedio, no coincida con el color de dicho píxel. Cuanto más grande es la máscara, mayor es el efecto de suavizado y mayor el tiempo de cómputo requerido. La pérdida de los detalles pertenecientes a las altas frecuencias es lo que caracteriza a todos los filtros pasa baja o de suavizado.

De entre la multitud de máscaras de filtros pasa bajas destaca especialmente la máscara de mediana, que es la que efectúa el promedio de los valores del entorno. Este filtro reemplaza el valor de un píxel por la media de los valores del mismo píxel y sus de sus vecinos en un radio que se predetermina. El filtro de mediana es un filtro útil para reducir el ruido de sal y pimienta de las imágenes. Es un filtro digital, no lineal, utilizado para extraer el ruido de las imágenes. La idea básica es revisar un píxel y decidir, al evaluar sus alrededores, si este es representativo a la señal. Los filtros de suavizado lineales o filtros pasa baja tienden a "difuminar los bordes" a causa de que las altas frecuencias de una imagen son atenuadas. Se aplicó sobre la imagen el filtro de mediana de la librería *vic32.dll* llamado *load\_medianfilter\_save()*. Después de ser aplicado dicho filtro se obtuvo una imagen cuyos bordes estaban bien definidos.

Para determinar la distancia entre un objeto y la cámara que adquirió la imagen fue necesario modelar la cámara de alguna manera. El modelo elegido por su simpleza fue el modelo *pinhole*. Con él se trabajaron dos ecuaciones, habiendo llegado a la conclusión que se tenían dos incógnitas en la ecuación de triángulos del modelo:  $H$  y  $d$ ; por lo que se necesitaba resolver dos ecuaciones para determinar sus valores. Fue así como se pensó en tomar dos fotografías a distintas distancias, y resolver el sistema de ecuaciones para encontrar la distancia entre el robot y el objeto en cuestión. La variable  $H$  se mide en píxeles y  $d$  en cm. La distancia que se mueve la cámara de un lugar a otro para tomar las dos fotografías es de 10cm. Mientras mayor sea esta distancia el error que podemos llegar a tener en la aproximación será peor. Se eligieron los 10cm que se mueve la cámara para tomar las dos fotografías. Entre las limitaciones del método están el hecho que el objeto debe estar totalmente alineado enfrente de la cámara. Requerimiento que podía ser cumplido al medir distancias de objetos alineados totalmente frente al robot.

El canal de comunicación se debió establecer inalámbricamente ya que era muy impráctico que el robot acarrease un cable consigo. Se prefirió elegir un canal ya establecido, en lugar de construir uno propio ya que se requería que dicho canal fuera eficiente, lo cual hubiese requerido de tiempo y una ardua evaluación de los campos que incluiría el protocolo de haber sido diseñado e implementado como parte del proyecto. El estándar elegido fue el *IEEE 802.3 (Ethernet)* en su versión inalámbrica conocida como *IEEE 802.11 (Wireless Ethernet)*. Se incorporó un router, con un alcance máximo de 50m a la redonda, sobre el robot que permitiría la transmisión de datos; a éste se conectaba la computadora en la estación central y permitía la comunicación entre todos los dispositivos conectados a dicha red: cámara Axis 205, PIC de telecomunicaciones y computadora en la estación central.

La aplicación utilizada para el control del robot se montó sobre el protocolo UDP/IP, en la capa de aplicación. Éste tipo de protocolo permitió la transmisión de datos eficientemente. Los errores en la transmisión eran mínimos si no, nulos. El protocolo implementado incluía comandos que el PIC maestro del Rover re-transmitiría al brazo robótico cuando se le indicase la recolección de basura.

Con el fin de tener un control centralizado, toda instrucción dirigida al control del Rover es transmitida hacia el PIC maestro del *Rover*. De éste modo, en todo momento el PIC maestro del *Rover* sabrá que esta ocurriendo con el robot. Se determinó se debían apagar los sensores de proximidad del Rover cuando éste se este aproximando a la basura. Si la basura es alta, se detectará como obstáculo no contemplado y el módulo de control realiza un paro de emergencia. Los sensores detectan obstáculos a una distancia máxima de 10cm y mínima de 1cm. Por el momento no se implementará para simplificar el modelo.

Respecto al control del brazo por parte del módulo de *visión por computadora*, éste solo le informará que recolecte basura cuando se posicione al robot a 20cm de distancia de la basura y ésta esté alineada frente al robot. El módulo del brazo robótico deberá recogerlo por si solo.

La definición de basura removible debió tomar en cuenta las limitaciones de los módulos de control, visión por computadora y del brazo robótico, ya que todos los módulos debían exponer sus limitaciones respecto a este tema. El módulo de brazo robótico, debido a la fuerza que tiene el mismo. El módulo de visión por computadora debido al alcance del algoritmo implementado. Finalmente la estructura del Rover debía poder soportar la basura que éste recogería. Fue así como se llevo a un consenso y se estableció el tipo de basura que el robot podría recoger.

# Capítulo 7

## Conclusiones y Recomendaciones

*“Para el logro del triunfo siempre ha sido indispensable pasar por la senda de los sacrificios”*

*Simón Bolívar*

---

### CONCLUSIONES

#### GENERALES

- i. Se desarrolló exitosamente un proyecto multidisciplinario en el que se puso en práctica los conocimientos adquiridos durante los cinco años de carrera.
- ii. Se desarrolló la optimización de un robot explorador autónoma que cumple con las características básicas de robots de este tipo. El robot es autónoma, auto sostenible y cumple con la tarea para el cual fue diseñado sin ayuda humana.

- iii. Se implementó exitosamente un sistema de *Visión por computadora* para controlar un robot explorador *limpia playas*, el cual recorre un terreno evadiendo obstáculos y recolectando basura.

### *ESPECIFICAS*

- iv. Se transmitió exitosamente una imagen tomada desde el robot a través de una red WLAN, hacia el programa en la estación central. Para ello se utilizó la cámara de Red Axis 205 con la cual se provee una librería que permite el control de la misma. La imagen era recibida en una aplicación en Visual Basic.
- v. Se estableció exitosamente un canal de transmisión inalámbrico entre el robot y la estación central a través de una red WLAN. Para ello se debió programar en un PIC la comunicación entre éste y el módulo ENC28J60 que se conectaba a la red Ethernet. En el lado de la computadora central se debió utilizar el módulo *winsock* para utilizar el protocolo UDP/IP en Visual Basic.
- vi. Se diseño e implementó exitosamente un método de detección de objetos en imágenes. Éste se basa en asignar distintos pesos a cada una de las tres componentes que describen el color de un pixel en el modelo de colores RGB.
- vii. Se diseño e implementó exitosamente un método para medir distancia aproximada entre el robot y un obstáculo. Se utilizó el modelo *pinhole* de una cámara con el cual, a través de una relación de triángulos y la adquisición de dos fotografías, se pudo determinar con un 90% de aproximación la distancia entre un objeto en una imagen y la cámara que adquirió la imagen.
- viii. Se diseño e implementó exitosamente un algoritmo que permite al robot movilizarse por su entorno recogiendo basura. El algoritmo hace tomar fotografías con cierta frecuencia a la cámara colocada sobre el robot para reconocer cuando se aproxima a un obstáculo.

- ix. Todo procesamiento realizado a la imagen hace el uso de la aplicación de filtros a la imagen. Al procesar una imagen digital, estos son la piedra angular de la disciplina de *visión por computadora*. Cualquier transformación que se le requiera hacer a la imagen se resume en la convolución de la imagen con la matriz del filtro elegido.

## RECOMENDACIONES

- x. Las condiciones en las que se trabajó el presente trabajo de graduación no son similares a las que se encontrarían en una playa. La luz del sol afectaría mucho el algoritmo de detección de objetos implementado, principalmente porque la cámara no detectaría grandes cambios de luz y ésta es la piedra angular de la técnica de localización de objetos. Además, al estar el robot en un ambiente húmedo puede empañar el lente de la cámara lo que alteraría las fotografías. Se recomienda atención a este detalle al llevar al Rover a un escenario tipo playa.
- xi. La posición actual de la cámara fue elegida por la estrategia que se eligió para mover al Rover. Algo importante a notar, es que la cámara está muy cerca del suelo. Para el escenario tipo laboratorio en el que se probó el Rover durante la implementación de este megaproyecto, no se encontró razón alguna por la cual la cámara no pudiese estar en la posición en la que está sobre el Rover. Sin embargo, se debe tomar en cuenta que al cambiar de escenario, a uno de playa, la cámara estaría muy cerca de la arena. Es de conocimiento general que la sal y humedad de la arena son enemigos íntimos de la electrónica. Es por ello que se recomienda se tome en consideración esta dificultad y se hacen tres sugerencias para la solución de dicho problema. Primero: cubrir la cámara con algún cubículo de cristal tal que la arena no llegue a ella. Segundo: se adquiera una cámara apta para ambientes exteriores. Tercero: se cambie de posición a la cámara a una posición en donde se encuentre más alejada del suelo. En el caso se elija la primera opción es posible que sobre el cubículo se refleje la luz o se tenga manchas en la fotografía, causadas por el deterioro del cubículo o es posible que la cámara no pueda adquirir fotografías



útiles si el material del cual está hecho el cubículo refleja demasiada luz (se debería elegir un material poco reflejante). En el caso de elegir la segunda opción se incurriría en gastos extra, ya que dichas cámaras tienen un precio elevado. Se debe tomar en cuenta que de elegir la tercera opción como la solución al problema, las fórmulas implementadas en los algoritmos deben cambiar, dado que ya no se puede aproximar que el objeto se observa como si estuviese alineado con la cámara, se deberá incluir en el modelo un ángulo de línea de vista entre el centro de la imagen de la imagen adquirida y el lente focal de la cámara.

- xii. Respectó al filtro que localiza obstáculos (cambios de intensidad de luz): *BlueYellowFilter*. No se probó, pero es probable que debido al color de la arena, café-rojizo, esta sea detectada como “obstáculo” en lugar de un camino transitable. En este caso se necesarió un ajuste de las constantes por las cuales se multiplica cada componente RGB en el filtro *BlueYellowFilter* para compensar la presencia de colores oscuros en la arena.
- xiii. Según la uniformidad de luz en el escenario en el cual se movilice el Rover, es necesario ajustar previamente las constantes del filtro: *BlueYellowFilter*, dado que estas constantes son dependientes de la cantidad de luz en el terreno. Para evitar hacer esto manualmente cada vez que cambien la uniformidad de luz en el terreno se recomienda que esto se realice mediante una constante retroalimentación de las celdas solares. Estas deberán reportar la cantidad de luz presente en el terreno y acorde a este dato se deberá ajustar las constantes del filtro. Se recomienda que el dato reportado por las celdas solares como la cantidad de intensidad de luz en el terreno en determinado momento, sea adquirido cuando el vector normal al plano de celdas solares apunte hacia arriba 90 grados de un vector paralelo al suelo.
- xiv. La idea que inspiró el presente módulo fue la implementación de un algoritmo llamado “Neural Networks” o “Redes neurales” en español. El algoritmo permite que un robot pueda “aprender” de sus conocimientos pasados, utilizando la base de datos que posee actualmente y formas de comportamiento a situaciones similares

ocurridas en el pasado. Se recomienda se implemente esto. El Rover debería tener una base de datos con la cual reconoce basura mediante comparación de patrones o rasgos. La idea detrás de “Redes neurales” es que el robot sólo, pueda agrandar su base de datos y determinar acciones según las haya tomado en situaciones similares en el pasado. Se recomienda se implemente este algoritmo para hacer del Rover un robot aún más autómatas.

- xv. Dado que este era un primer intento de concretizar al Rover como un robot recoge basura, los algoritmos se realizaron pensando en que cumplirían una acción definida y no pensando en el consumo de recursos de memoria o tiempo de procesamiento. Se cree que los algoritmos pueden ser mucho más eficientes, si se tiene el tiempo para analizarlos y encontrar maneras óptimas para realizar sus tareas, economizando tiempo de procesamiento. Se recomienda se optimicen los algoritmos del sistema de visión por computadora.
- xvi. El alcance del Router actualmente instalado es de 50m radiales por el alcance de la señal de radio frecuencia del *router*, sin embargo al soltar al robot en una playa el alcance del robot debe ser mayor, ya que la estación de control generalmente estará más alejada del robot que 50m a la redonda. Por lo que se recomienda que se cambie el router a uno con mayor alcance.

# BIBLIOGRAFÍA

1. **Willson, Andrew.** Multicamera system reads vial labels. *Vision Systems Design*. [Online] 2008. [http://www.vision-systems.com/articles/article\\_display.html?id=281365](http://www.vision-systems.com/articles/article_display.html?id=281365).
2. **Cognex Corp.** Inspection of up to 2000 different types of circuit breaker. *Cognex*. [Online] 2008. <http://www.cognex.com/ApplicationsIndustries/TypesOfApplications/default.aspx?id=74>.
3. **Cognex Corp..** Defect-free packaging for distribution. *Cognex*. [Online] 2008. <http://www.cognex.com/ApplicationsIndustries/TypesOfApplications/default.aspx?id=74>.
4. **Cognex Corp.,.** Nissan saves time and money with Cognex vision. *Cognex*. [Online] 2008. <http://www.cognex.com/ApplicationsIndustries/TypesOfApplications/default.aspx?id=76>.
5. **Cognex, Corp.,.** Laser-cutting path corrections and inspection. *Cognex*. [Online] 2008. <http://www.cognex.com/ApplicationsIndustries/TypesOfApplications/default.aspx?id=78>.
6. **Palacio de Minería.** Torneo Mexicano de Robotica. *5o Torneo Mexicano de Robots Limpiadores Terrestres y Acuáticos*. Mexico DF : s.n., 2008.
7. **Cognex Corp.** Defect-free packaging for distribution. *Cognex*. [Online] 2008. <http://www.cognex.com/ApplicationsIndustries/TypesOfApplications/default.aspx?id=74>.
8. **Cognex, Corp.** Inspection of up to 2000 different types of circuit breaker. *Cognex*. [Online] 2008. <http://www.cognex.com/ApplicationsIndustries/TypesOfApplications/default.aspx?id=74>.
9. **Frostburg Faculty.** Peripherals Input Output Hardware. [Online] 11 26, 2006. Imagen. <http://faculty.frostburg.edu/cosc/htracy/COSC100/PERIPHERALS/DC04-37.jpg>.
10. **El economista.** elEconomista.es. [Online] 10 2007. Imagen. [http://www.eleconomista.es/imag/\\_v2/istock/huella-dactilar.jpg](http://www.eleconomista.es/imag/_v2/istock/huella-dactilar.jpg).
11. **BuyaBussiness.** Buy a Bussiness. *South african Bussiness for sale*. [Online] 2008. Imagen. [http://www.buyabusiness.co.za/images/businesses/1032\\_250px-FrenchNumberPlates\\_CopyrightKaihsuTai.jpg](http://www.buyabusiness.co.za/images/businesses/1032_250px-FrenchNumberPlates_CopyrightKaihsuTai.jpg).
12. **Paredes, Carlos.** Limpian playa y calles del Puerto San José, Escuintla. *Prensa Libre*. Departamentales, 2008, Vol. 17 de marzo de 2008.
13. **Departamento de Ingeniería en Electrónica, UVG.** Sitio Web del Departamento de Ingeniería en Electrónica. [Online] 2007. <http://www.uvg.edu.gt/uvgtronik/>.
14. **Space History.** Word press. [Online] 12 04, 1996. <http://todayinspacehistory.wordpress.com/2007/12/04/december-4-1996-first-successful-mars-rover-sojourner-was-launched-toward-mars/>.
15. **Miano, John.** *Compressed Image File Formas: JPEG, PNG, GIF, XBM, BMP*. Massachusets : Addison Wesley Longman, Inc., 1999. 0-201-60443-4.
16. Axis Cooperation. [Online] 2008. [www.axis.com](http://www.axis.com).
17. Adobe Illustrator CS3. *Acerca de los formatos vectoriales*. s.l. : Adobe, 2007. Vol. Adoebe Illustrator help.
18. **R., C. Pinilla.** Filtrado Selectivo de Imágenes. s.l. : Universidad de Jaén, 2000.

19. **Studio, Microsoft Robotics.** Microsoft Robotics Developer Center. [Online] 2008. <http://msdn.microsoft.com/en-us/library/bb727252.aspx>.
20. Computer Vision Home Page. [Online] 02 2004. <http://www.cs.cmu.edu/~cil/vision.html>.
21. **Lowe, David.** The Computer Vision Industry. [Online] 01 1998. <http://www.cs.ubc.ca/spider/lowe/vision.html>.
22. **Pollefeys, Marc.** Computer Vision. [Online] 2006. <http://www.cs.unc.edu/Research/vision/comp256/>.
23. **Mansilla, Juan Fernando.** *Diseño y construcción de un sistema explorador robotizado y autónomo. Submódulo de procesamiento de imágenes.* Guatemala : UVG, 2005.
24. **Mijangos, Luis Guillermo.** *Diseño y construcción de un sistema explorador robotizado y autónomo, Submódulo de Control del Sistema Explorador a través de Internet.* Guatemala : UVG, 2005.
25. **Axis Corporation.** Axis Communication. *Tecnología de las cámaras de red.* [Online] 2007. [http://www.axis.com/products/video/camera/about\\_cameras/netcam\\_tech.es.htm](http://www.axis.com/products/video/camera/about_cameras/netcam_tech.es.htm).
26. **Abox.** Cámaras de Red Axis. [Online] 2007. <http://www.abox.com/productos.asp?pid=558>.
27. **Proxim.** Proxim. [Online] 2007. <http://sss-mag.com/pdf/proximwhatwlan.pdf>.
28. **Huidobro, José Manuel.** Wi-Fi. [Online] 2007. <http://www.monografias.com/trabajos14/wi-fi/wi-fi.shtml>.
29. **Stardust Technologies, Inc.** Sockets. [Online] 04 27, 1997. <http://www.sockets.com/>.
30. **Catenary.** Catenary. [Online] 2008. <http://www.catenary.com/>.
31. **GameDev.** Matrix convolution filters. [Online] 2007. <http://www.gamedev.net/reference/programming/features/imageproc/page2.asp>.
32. **Russell S., Norvig P.** *Artificial intelligence, a modern approach.* 2ed. 2003.
33. **UMA.** The Median Filter Problem. [Online] 2003. <http://tracer.lcc.uma.es/problems/mfp/mfp.html>.

# APENDICE

En el disco adjunto se incluyen los archivos de proyecto de Visual Basic, C++ y MikroC así como los archivos \*.exe y \*.hex.

La librería *rover.dll* implementada en C++ tiene los siguientes métodos que permiten el análisis de imágenes.

**1. *Funcion BlueYelowScale:***

Lee y devuelve la imagen en tonos azules y amarillos

Un obstaculo es identificado por el color azul del pixel

Un paso libre es identificado por el color amarillo del pixel

**2. *Funcion MedianFilter:***

Ajusta el valor de un pixel promediándolo con el valor de los pixeles aledaños a un radio r.

r: indica el radio de pixeles promediados

**3. *Funcion Scan:***

Funcion que busca en las tres areas del centro de la fotografia

llamando a la funcion LookArea

Las areas son: 30%, 70%, MID, UP

**4. *Funcion LookArea:***

Funcion que busca obstaculos en un area especificada

por ys->ye & xs->xe

ys,ye: inicio y final de area en eje y

xs,xs: inicio y final de area en eje x

MD: booleano si se requiere que se mida la distancia al objeto o no

**5. *Funcion Measure:***

Funcion que mide la altura del objeto localizado

sp: y start point, valor y desde el que comienza a buscar la altura  
del objeto encontrado

xi,xf: valor x inicial y final en la que busca el final del objeto

o: offset del archivo en donde comienza pixel data

w,h: width y height de la fotografia

\*archivo: puntero al archivo medianf.bmp, fotografia a analizar

**6. *Distancia:***

Funcion que mide la distancia de un objeto al ROVER, h1, h2 y D son parámetros

# GLOSARIO

ADC: *Analog Digital Converter*, es un circuito que convierte una señal continua (análoga) a una discreta (Binaria).

AMC: *Axis Media Controll*, este es un control que provee la compañía Axis para visualizar video proveniente de alguno de sus modelos desde alguna aplicación hecha por el usuario.

Applet: Un applet es un componente de una aplicación que se ejecuta en el contexto de otro programa. Generalmente es una aplicación gráfica.

Axis 205: Es una cámara de red modelo 205 de la compañía Axis.

CCD: *Charge-Coupled Device*. Es un sensor que contiene una matriz de capacitores enlazados o acoplados los cuales transfieren su carga eléctrica a uno o varios de los condensadores que estén a su lado. La aplicación más usual de este dispositivo es como sensor en cámaras digitales, estos poseen una matriz de capacitores que pueden pensarse como pequeñas celdas solares las cuales registran la imagen adquirida por una cámara. Cada *celda* corresponde a un pixel en la imagen, *i.e.* cada celda captura la longitud de onda correspondiente a ese pixel en la imagen.

DHCP: *Dynamic Host Configuration Protocol*. Es un protocolo de capa de enlace de datos que permite a los nodos en una red obtener sus parámetros de configuración automáticamente. Se trata de un protocolo cliente-servidor, en el que generalmente un servidor posee una lista de direcciones IP dinámicas y las va asignando a los clientes conforme éstas van estando libres

Dirección IP: *Internet Protocol*. Es un número que identifica a un nodo dentro de una red que utiliza el protocolo IP. IP es un protocolo de capa de red.

DLL: *Dynamic Linking Library*. Una DLL es una librería de enlace dinámico la cual no es más que métodos y funciones, compiladas en un archivo binario para que estos puedan ser invocados por cualquier otro programa que incluya en el proyecto a dicha librería.

DNS: *Domain Name System*. Una base de datos distribuida y jerárquica que almacena información asociada a nombres de dominio en redes como Internet. Permite traducir a un lenguaje coloquial de letras las direcciones IP utilizadas en la red como identificadores de equipo.

Dirección MAC: *Medium Access Control address*. Es un identificador de 48 bits (6 bytes) que corresponde de forma única a una tarjeta o interfaz de red. Es individual, cada dispositivo tiene su propia dirección MAC determinada y configurada por el IEEE (los últimos 24 bits) y el fabricante (los primeros 24 bits) utilizando el OUI.

DSP: *Digital Signal Processing*. Es un sistema basado en un procesador o microprocesador que realiza operaciones sobre datos digitales adquiridos. En este caso los datos digitales sobre los cuales se realizará el procesamiento son los números binarios asignados a cada pixel en una imagen.

GUI: *Graphical User Interface*. Esta es una interfaz gráfica que permite a personas interactuar con dispositivos electrónicos.

IEEE: *Institute of Electrical and Electronics Engineers*. Es una asociación de profesionales Eléctricos y Electrónicos mundial dedicada a la estandarización. Es la mayor asociación internacional sin fines de lucro formada por profesionales de las nuevas tecnologías, como ingenieros eléctricos, ingenieros en electrónica, científicos de la computación, ingenieros en informática e ingenieros en telecomunicación.

IEEE 802.3: *Ethernet*. Es un protocolo de comunicación para redes de área local (LAN) que define la capa física y la capa de MAC del protocolo OSI. La capa física puede ser de 3 tipos: cable coaxial, par trenzado o fibra óptica. Existen en velocidades 10Mbps, 100Mbps, 1Gbps hasta 100Gbps.

IEEE 802.11: *Wireless Ethernet*. Es un protocolo de comunicación para redes de área local (LAN) define la capa física y la capa MAC del protocolo OSI. En la capa física se transmite a través de ondas electromagnéticas.

JPEG: *Joint Photographic Experts Group*. Es un algoritmo de compresión de imágenes que elimina frecuencias altas en una imagen aprovechando que el ojo no puede percibirlas.

LAN: *Local Area Network*. Es un grupo de computadoras y dispositivos que comparten un mismo canal de comunicación alámbrico que les permite comunicarse entre si.

NASA: *National Aeronautics and Space Administration*. Es una institución de los Estados Unidos, que es la agencia gubernamental responsable de los programas espaciales.

OCR: *Optical character recognition*. Es un sistema que permite reconocer, al ser escaneado un documento, texto escrito a mano, a computadora o con máquina de escribir. Este lo transforma a texto editable y por la computadora.

PCB: *Printed Circuit Board*. Es el medio para sostener mecánicamente y conectar eléctricamente componentes electrónicos, a través de rutas o pistas de material conductor, grabados sobre hojas de cobre laminadas sobre un sustrato no conductor.

PIC: *Peripheral Interface Controller*. Son una familia de microcontroladores tipo RISC fabricados por Microchip Technology Inc.

Relé: Es un dispositivo electromecánico, que funciona como un interruptor controlado por un circuito eléctrico en el al energizar un electroimán, se acciona un juego contactores que cierra o abre un circuito.

Router: Es un dispositivo de hardware que permite la interconexión de red de computadoras que opera en la capa tres (nivel de red). Este dispositivo permite asegurar el enrutamiento de paquetes entre redes o determinar la ruta óptima que debe tomar el paquete de datos.

Rover: Rover de exploración de Marte es una misión de la NASA no tripulada enviada para explorar Marte. La misión incluye el envío de dos *Rovers* (robots) para explorar la superficie y la geología de Marte de manera autónoma.

SD: *Secure Digital* es un formato de tarjeta de memoria flash. Se utiliza en dispositivos portátiles tales como cámaras fotográficas digitales.

Sojourner: Es el primer *Rover* enviado a explorar Marte en 1997.

Spirit: Es un robot *Rover* que forma parte del Programa de Exploración de Marte. La nave aterrizó con éxito en el planeta Marte el 4 de enero de 2004.

TCP/IP: *Transmission-Control-Protocol / Internet Protocol*. Es el conjunto de dos protocolos que permiten la transmisión de datos entre computadoras conectadas a una red. El protocolo TCP define la capa de transporte y el protocolo IP define la capa de red del modelo OSI. El protocolo TCP se caracteriza por ser orientado a conexión.

UDP/IP: *User Datagram Protocol / Internet Protocol*. Es el conjunto de dos protocolos que permiten la transmisión de datos entre computadoras conectadas a una red. El protocolo UDP define la capa de transporte y el protocolo IP define la capa de red del modelo OSI. El protocolo UDP se caracteriza por ser no orientado a conexión.

UVG: *Universidad del Valle de Guatemala*. Universidad en la ciudad de Guatemala con más de 40 años de trabajo en educación superior en Guatemala.

Visión por computadora: Es la disciplina que tiene como objetivo el control de sistemas y/o procesos mediante el procesamiento de imágenes digitales provistas por una cámara.

Visión de máquina: El término denota la aplicación de visión por computadora a la industria y manufactura.

WAN: *Wide Area Network*. Es un tipo de red de computadoras capaz de cubrir distancias desde unos 100 hasta unos 1000 km, dando el servicio de comunicación de equipos y computadoras a un país o un continente.

WinSock: *Windows Sockets*. Es un control utilizado en aplicaciones de Windows para comunicarse con otras máquinas utilizando el protocolo TCP/IP o UDP/IP.

WLAN: *Wireless Local Area Network*. Es una red de comunicación que utiliza radiofrecuencia para la transmisión de datos inalámbricamente.