ESCUELA POLITECNICA NACIONAL

FACULTAD DE INGENIERIA ELECTRICA

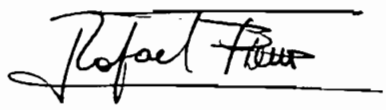
"SIMULACION Y ENSAMBLAJE DE UN PROTOTIPO PARA CONTROL Y NAVEGACION DE UN ROBOT MOVIL"

TESIS PREVIA A LA OBTENCION DEL TITULO DE:
INGENIERO EN ELECTRONICA Y CONTROL

LENIN MAX ANDRANGO MEJIA
NELSON GONZALO SOTOMAYOR OROZCO

QUITO, MAYO DE 1999

Certifico que el presente trabajo ha sido realizado en su totalidad por los Srs. Lenin Andrango y Nelson Sotomayor



Rafael Fierro B., PhD.
DIRECTOR DE TESIS

AGRADECIMIENTO

Un agradecimiento sincero a todas las personas que de una u otra manera colaboraron en el desarrollo de la presente Tesis, y de manera especial al Dr. Rafael Fierro por el tiempo y el esfuerzo brindados en ella.

DEDICATORIA

A mis padres que supieron brindarme su apoyo, a mis hermanos por su incondicional colaboración, y al grupo de amigos que me dio el impulso para llegar a la meta.

Lenin

A mis padres: que con su ejemplo, sacrificio e incondicional apoyo me han llevado por el camino del bien permitiéndome alcanzar mis metas y aspiraciones.

A mis hermanos: que me han apoyado en los momentos difíciles.

A mi esposa: que con su amor me ha dado las fuerzas necesarias para alcanzar esta meta.

A mi hijo(a): que a pesar de estar en el vientre de su madre me impulso a culminar este trabajo.

Nelson

INDICE

Capítulo 1

INTRODUCCION

1.1 ANTECEDENTES

Hasta hace pocos años, ni él más imaginativo empresario habría podido concebir lo que hoy es ya una realidad cotidiana, la entrada de los robots en el ámbito de la producción y la actividad humana.

Los robots han demostrado ser especialmente capaces de realizar actividades repetitivas, que para los seres humanos podrían resultar tediosas. Son también ideales para manejar materiales peligrosos en ambientes nocivos.

En los últimos años, el área de los robots móviles ha tenido una importante actividad y desarrollo, ya que ha sido el sistema preferido por investigadores en las áreas de Inteligencia Artificial, Control Inteligente e Instrumentación. Debido a su movilidad y manipulación, eventualmente podrían reemplazar a los humanos en sistemas de manufactura y servicios industriales. Cada día se reportan nuevas aplicaciones de robots móviles, como por ejemplo, en transporte de material en instalaciones de manufactura altamente automatizadas, vigilancia, exploración, aplicaciones militares y espaciales. Robots móviles podrían reemplazar a perros guías para ciegos, o sillas de ruedas autónomas, entre otras. Uno de los avances más importantes y que captó la atención mundial fue el envío exitoso de un robot móvil ('rober') para explorar y enviar información del planeta Marte.

La gran mayoría de proyectos que involucran la investigación con robots móviles cuentan con herramientas sofisticadas de simulación. Mediante simulación, se pueden probar diferentes algoritmos de control de alto y bajo nivel antes de implantarlos en el robot móvil. Lamentablemente, estas herramientas de simulación son muy costosas y generalmente desarrolladas en sistemas UNIX. Adicionalmente, las plataformas móviles disponibles en el mercado suelen tener costos prohibitivos y sistemas de control cerrados

que dificultan la experimentación de nuevos algoritmos de control y navegación.

La falta de herramientas de simulación para PCs que trabajan con Windows 95/NT, y el alto costo de plataformas móviles, motivó la realización de la presente tesis.

1.2 OBJETIVOS Y ALCANCE

Los objetivos y alcance de este trabajo son los siguientes:

Desarrollo de un programa en ambiente Windows 95/NT que simule un robot móvil de tracción diferencial. El programa permite editar el ambiente de trabajo del robot móvil, el cual estará constituido por paredes, corredores y obstáculos, estos últimos podrían ser localizados por el usuario.

Ensamblar un prototipo de robot móvil. En el sistema se implementarán tareas de navegación simples, tales como: marcha adelante, marcha atrás, movimientos de izquierda y derecha.

Se dotará al robot móvil con cierta inteligencia, éste será capaz de buscar alternativas diferentes de movimiento, con el objeto de evitar chocar con obstáculos que se encontraren en su trayectoria dentro de su ambiente de trabajo.

Se incorporará en el robot móvil, algunos sensores, para que pueda cumplir con actividades de exploración, y dar información del medioambiente que está explorando.

1.3 CONTENIDO

La tesis esta organizada de la siguiente forma:

El segundo capítulo, se presenta una introducción general de robots móviles, y se dan fundamentos de navegación, posicionamiento y el modelo cinemático del robot móvil de tracción diferencial.

Los capítulos tres y cuatro, tratan de Simulación de Robots Móviles, en el capítulo tres, se tiene información general sobre simuladores y ejemplos, mientras que el capítulo cuatro describe el simulador implementado en la tesis.

En los capítulos cinco y seis se detalla el ensamblaje de la plataforma móvil, por comodidad de presentación en el capítulo cinco se describe la parte de hardware, mientras que en el capítulo seis se describe el software, implementado en el microcontrolador a bordo del robot.

Finalmente en el capítulo siete se dan conclusiones, recomendaciones, así como una visión general sobre el trabajo futuro que se podría realizar para mejorar tanto el prototipo como el simulador.

Capítulo 2

FUNDAMENTOS BASICOS DE ROBOTS MOVILES

2.1 INTRODUCCION

Un robot puede manipular los objetos que están a su alcance. En la mayoría de las industrias que utilizan robots, estos se encuentran fijos en su lugar de trabajo, pero también existen las industrias que utilizan la movilidad de los robots, montados sobre rieles, en este caso el robot tiene un camino fijo sobre el cual moverse, por lo que sus movimientos son limitados. Pero no sólo en la industria son utilizados los robots móviles, estos se han utilizado también en ambientes militares, para exploración y reconocimiento de ambientes que podrían ser peligrosos para los humanos.

En los últimos años, en este campo se han realizado muchos estudios y se han desarrollado robots móviles, que para su movimiento utilizan ruedas y están equipados con sensores para su control y navegación.

2.2 CONFIGURACION DE PLATAFORMAS MOVILES

Entre las configuraciones más importantes se pueden incluir las siguientes:

2.2.1 Dirigidor de Ackerman

Es usado casi exclusivamente en la industria automotriz. Este diseño permite que la rueda interior gire con un ángulo ligeramente superior a la rueda exterior, como se puede apreciar en la figura 2.1, los ejes extendidos de las ruedas delanteras pasan por un punto concéntrico en el eje extendido de las ruedas traseras, el centro de rotación P_1 (ignorando el efecto de la aceleración centrífuga), por lo tanto los vectores de velocidad instantánea serán tangenciales a los arcos concéntricos de cada rueda. Esta configuración geométricamente debe satisfacer la ecuación de Ackerman:

$$\cot\theta_i - \cot\theta_o = \frac{d}{l}$$

Donde:

θ_i = ángulo relativo interior

θ_o = ángulo relativo exterior

l = separación de las ruedas

d = separación lateral de las ruedas

Por facilidad se asume un ángulo θ_{SA} ubicado en una rueda imaginaria en el centro de las dos ruedas, por lo que la ecuación será:

$$\cot\theta_{SA} = \frac{d}{2l} + \cot\theta_i \qquad o \qquad \cot\theta_{SA} = \cot\theta_O - \frac{d}{2l}$$

Este dirigidor provee una solución bastante precisa para una buena tracción en todo terreno, por lo que es muy utilizado en vehículos con motores de gasolina o diesel.

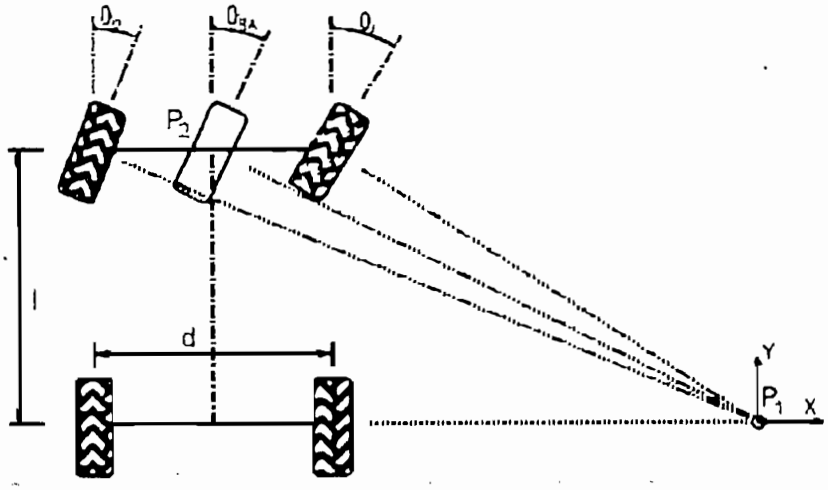


Figura 2.1 Dirigidor de Ackerman

2.2.2 Tracción de Triciclo

Es una de las configuraciones más simples de implementar, como se puede observar en la figura 2.2, consta de una rueda activa anterior y dos ruedas pasivas posteriores (o viceversa). La forma de control es similar al dirigidor de Ackerman discutido en la sección anterior.

El problema fundamental de este tipo de configuración, radica en que el centro de gravedad de la plataforma tiende a moverse lejos de la rueda anterior cuando cruza por un declive, ocasionando la perdida de tracción.

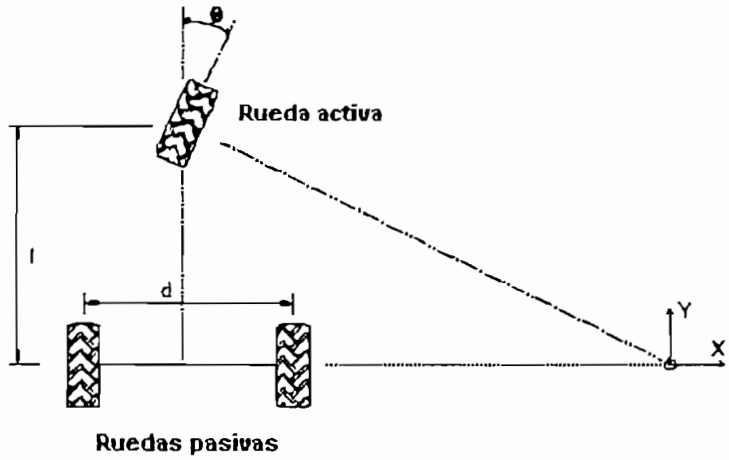


Figura 2.2 Tracción de Triciclo

2.2.3 Manejador Sincrónico

Este manejador se caracteriza por tres o más ruedas mecánicamente acopladas, que giran en la misma dirección y a la misma velocidad, esta sincronización se puede realizar de diferentes maneras, la más utilizada es colocar una cadena, banda o engranajes.

Un ejemplo ilustrativo es el Denning de Centinela [1], cuyo esquema ilustrativo se muestra en la figura 2.3. Donde la primera cadena provee el

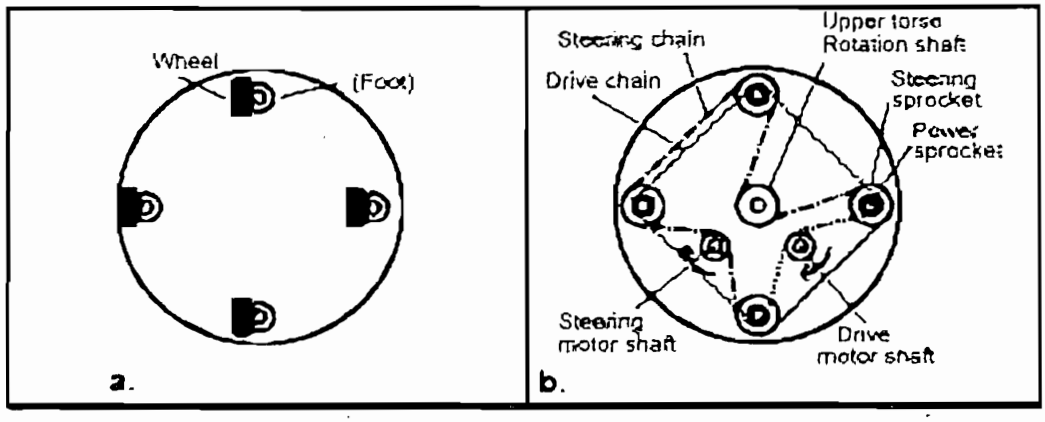


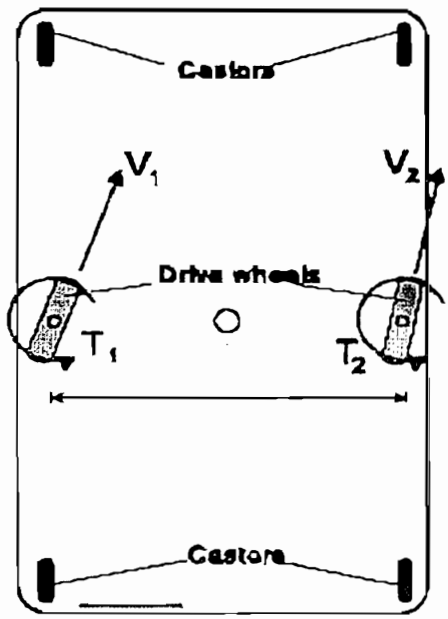
Figura 2.3 a. Vista inferior, b. Vista superior

torque necesario para la tracción, mientras que la segunda permite la rotación.

2.2.4 Vehículos de Múltiples Grados de Libertad

Los diseños de múltiples grados de libertad, tienen múltiples motores, por lo que el problema fundamental radica en la coordinación de estos actuadores. Un ejemplo de aplicación es la plataforma HERMES III, diseñada y construida por The Oak Ridge National Laboratory, el cual tiene 4 grados de libertad, controlados por cuatro motores independientes. (ver figura 2.4)

Figura 2.4. Plataforma de 4 grados de libertad



2.2.5 Vehículos de Múltiples Grados de Libertad con Enlace Flexible

Para superar los problemas descritos en el numeral anterior, se desarrollo un sistema de Multi grado de libertad [MDOF], mostrado en la fig. 2.5. Consiste en dos vehículos de tracción diferencial, unidos con un enlace flexible y dos articulaciones rotativas, para tener un total de 3 grados de libertad.

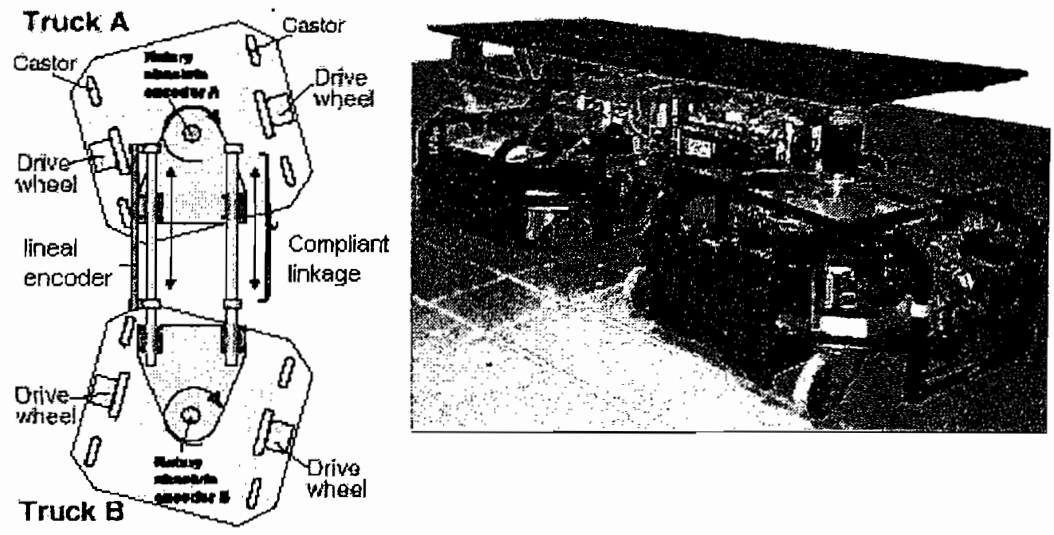


Figura 2.5 Vehículos de Múltiples grados de libertad con enlace flexible

2.2.6 Vehículos Tipo Oruga

Consiste en un vehículo de tracción diferencial, en el cual las ruedas van montadas sobre canales, similar al utilizado en maquinaria de construcción. Se emplea generalmente en robots móviles autónomos, ya que por su configuración pueden salvar casi cualquier irregularidad del terreno. Como se puede ver en la figura 2.6, la estructura esta montada sobre círculos concéntricos que permitirán al vehículo girar en su propio eje.

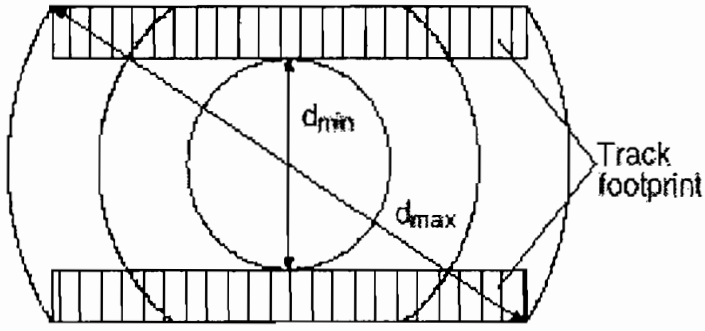


Figura 2.6 Estructura del vehículo de oruga.

En la figura 2.7, se puede ver al Romotec Andros V, implementado en la *Universidad de Michigan.*

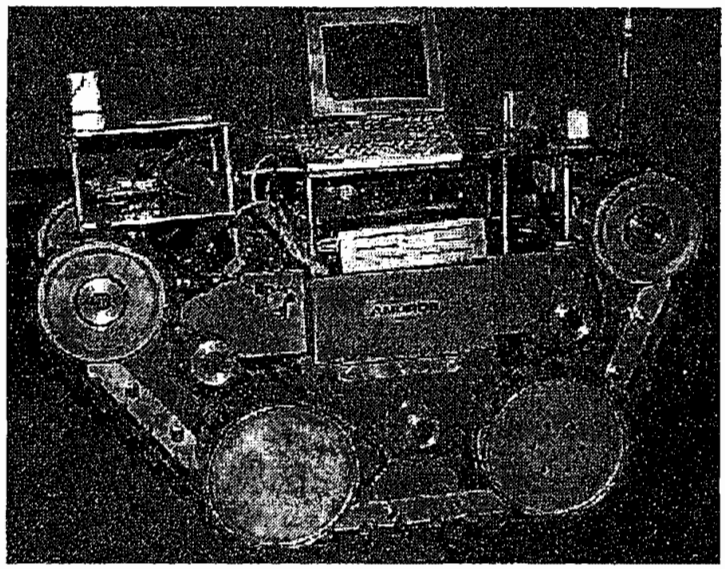


Figura 2.7 Romotec Andros V.

2.2.7 Vehículos de Tracción Diferencial

La figura 2.8 muestra la típica plataforma móvil de tracción diferencial, utilizado por la plataforma LabMate. En esta plataforma se colocan encoders ópticos en los dos motores, para determinar su posición utilizando simples ecuaciones geométricas. El método para determinar la posición se la analizará mas adelante.

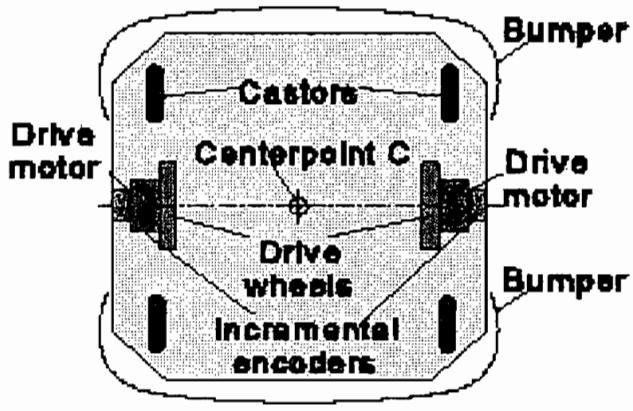


Figura 2.8 Vehículo de tracción diferencial

2.3 FUNDAMENTOS DE NAVEGACION DE ROBOTS MOVILES

Navegación es la ciencia (o el arte) de dirigir a un robot móvil en su entorno de trabajo. La navegación involucra tres tareas principales: el mapeo, la planeación y la conducción.

El proceso de navegación de robots móviles se puede resumir en el siguiente cuadro:

Medición:	*Sensar el entorno*
	Detectar objetos
	Ingresar comandos de usuario
Modelación:	*Mapa del entorno*
	Modelo de objetos
	Mapa de caminos
Percepción:	*Buscar caminos*
	Detectar situaciones de colisión
	Conocer el mapa
Planeación:	*Descomponer las tareas en subtareas*
	Seleccionar caminos
	Escoger alternativas cuando el camino es bloqueado
Acción:	*Navegar*
	Atravesar caminos evitando colisiones
	Control en base al modelo cinemático o dinámico del robot.

En el resto de la sección se examinarán técnicas para implementar subtareas de mapeo, planeación y conducción.

2.3.1 Sistemas de Guía

Son sistemas que cuentan con trayectorias predeterminadas. Algunos de estos sistemas almacenan una red de trayectorias en forma gráfica.

Los vehículos que utilizan estos sistemas son conocidos como AGVs (Automated Guided Vehicles) [2], y son empleados generalmente en la industria. Los AGVs siguen cables guías colocados en el piso, estos cables son detectados usando inducción. Adicionalmente existen otros métodos para definir los caminos alternativos, como por ejemplo: rayas pintadas, columnas magnéticas, líneas fluorescentes invisibles o rayos láser.

Estos sistemas están limitados a un número fijo de caminos previamente definidos. Los AGVs no pueden continuar su camino si encuentran un objeto en su trayectoria, hasta que el objeto sea retirado, y cuentan con un rudimentario sistema de detección de obstáculos, el cual los desactiva cuando chocan con un objeto. A pesar de estas limitaciones estos vehículos son usados en plantas donde este nivel de navegación es aceptable.

2.3.2 Odometría

La odometría es el cálculo de la posición y orientación del robot móvil. Para corregir los problemas de los robots con caminos definidos, algunos robots siguen un camino preprogramado, utilizando marcas en el piso para corregir su posición, algunos utilizan rayos láser o colocan marcas a lo largo de su trayectoria.

El control por odometría tiene varias fuentes de inexactitud: pobre mecanismo de alineación de ruedas, juego en los engranajes, ruido en los sensores, errores en la señal del sensor, deslizamientos de las ruedas por deformidades del piso.

2.3.3 Faros

Son similares a los controles remotos de TV, es decir utilizan señales infrarrojas, por lo tanto el robot tiene sobre si un sensor infrarrojo de 180° con el cual detecta los faros y corrige su trayectoria.

Son utilizados para corregir los errores de odometria. Los robots calculan su trayectoria y corrigen los errores de posición basándose en estos faros.

Para que el vehículo aprenda el camino, el operador lo lleva a lo largo de la ruta y especifica puntos de parada y bifurcaciones. El robot almacena el camino y la ubicación, para poder navegar solo por la ruta establecida, si se añade una nueva ruta que no haya sido almacenada, el robot se detiene hasta que el operador lo guíe por la nueva ruta.

2.3.4 Mapas de Medio Ambiente, Modelos y Estructura de Datos

Uno de los aspectos más importantes en el diseño de robots móviles, es lograr mayor autonomía, así ellos pueden navegar libremente en su ambiente de trabajo. Para lograr este objetivo es indispensable que se disponga de un mapa en una estructura de datos, y de los algoritmos necesarios para manipular esos datos y conocer de esta manera la posición del robot dentro de su ambiente.

Los robots móviles analizados hasta el momento, utilizan para su navegación faros o marcas colocadas en el piso, estos pueden ser reemplazados por visión artificial a través de sensores que podrían ser: de ultrasonido, rayos láser u otro tipo de sensor. La información que entregan estos sensores se compara con la del mapa y se determina así la posición del robot. Esa información también puede ser almacenada para crear nuevos mapas de ambientes desconocidos.

La estructura de datos ideal es aquella que divide al mapa en regiones de forma rectangular o cuadrada, haciendo así que una zona del mapa sea representada con un número en la estructura de datos. De esta manera para el robot móvil, el mapa del ambiente de trabajo es una lista de conexiones de pequeñas áreas representadas por números.

Algunos robots móviles almacenan en su memoria los datos de posición de cada segmento del mapa, con ello conocen su ubicación y tienen información de los segmentos cercanos, esta información es almacenada en matrices para mayor eficiencia, de esta manera se puede ampliar o disminuir el mapa solamente añadiendo o retirando segmentos (figura 2.9).

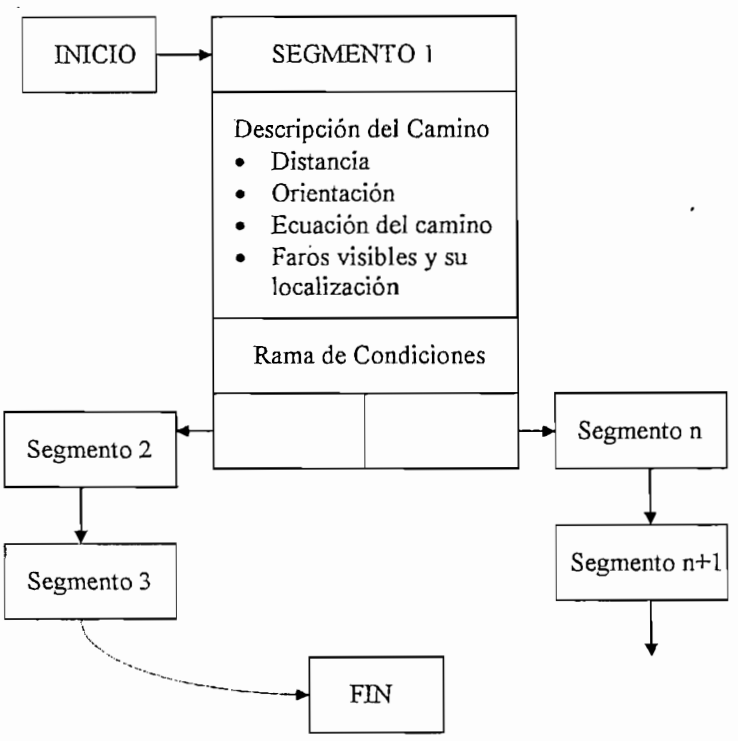


Figura 2.9. Lista de enlaces de segmentos de un mapa de medio ambiente

A continuación se analizan diferentes formas de realizar los mapas de medioambiente.

2.3.4.1 Información de Sensores

Utilizando sensores se puede recopilar la información necesaria para realizar el mapeo de un ambiente, esta información estaría constituida por puntos que son centros de circunferencias con los cuales se podría localizar los objetos y modelar un camino libre de obstáculos.

Esta estructura de datos esta compuesta por tres elementos: la posición (X,Y) y el radio de la circunferencia (ver figura 2.10).

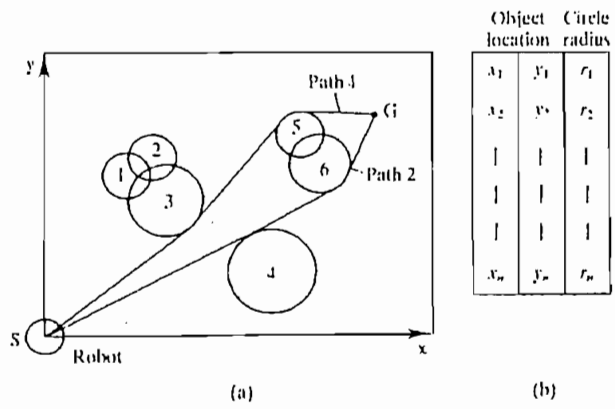


Figura 2.10. Mapa del medio ambiente utilizando círculos que representan objetos

2.3.4.2 Mapas de Espacio Libre

Un robot equipado con sensores puede moverse por un ambiente desconocido sin chocar con ningún objeto, y gravar los caminos alternativos de movimiento para construir un mapa espacial del entorno (ver figura 2.11). En el gráfico los puntos representan paradas que el robot hace para sensar el entorno. Todas las regiones con arcos pueden contener un objeto.

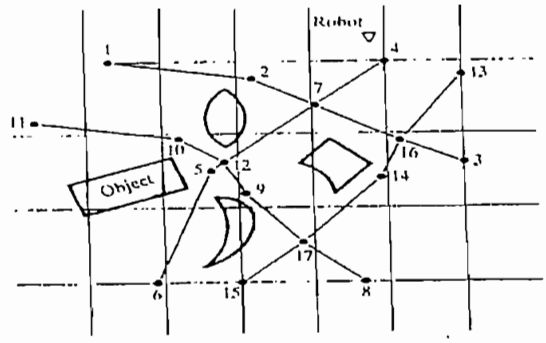


Figura 2.11. Gráfico espacial formado con información de sensores.

A partir del gráfico espacial se puede construir un diagrama de Voronoi (figura 2.12), que se forma dividiendo el gráfico espacial en regiones poligonales. Cada región esta asociada con un punto de parada, en estas regiones se tiene además información de la existencia o no de objetos. Así el robot móvil se puede mover seleccionando regiones vacías adyacentes hasta llegar a su meta.

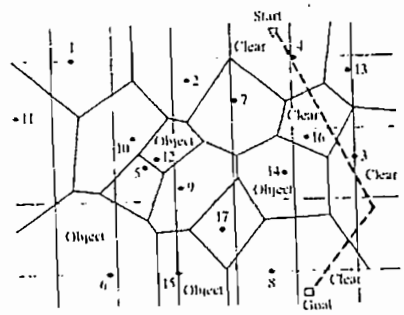


Figura 2.12. Diagrama de Voronoi, en el cual se muestra un camino óptimo de movimiento.

2.3.4.3 Mapas Globales y Locales

Mapas globales son aquellos que muestran de manera general las conexiones entre regiones de interés, mientras que los locales muestran los detalles del medio ambiente del robot móvil. En las aplicaciones reales de robots móviles, se utilizan los dos tipos de mapas, es decir un mapa global que estaría almacenado en su memoria, el cual se descompondría jerárquicamente en varios mapas locales que se almacenarían en un computador central.

2.3.4.4 Mapa de Objetos Orientados

Son utilizados frecuentemente cuando los ambientes de trabajo se conocen muy bien, estos mapas almacenan la ubicación de los objetos con las coordenadas absolutas de sus vértices. Lo que implica que las zonas donde no existen objetos son espacios libres. Cabe anotar que este método es muy limitado si los objetos están en movimiento.

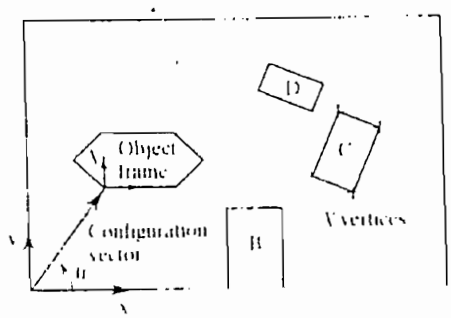


Figura 2.13 Mapa de objetos orientados.

2.3.4.5 Mapas de Espacio Compuesto

En los mapas de objetos y de espacio libre, se conocen únicamente uno de los dos, y el otro sale por añadidura, pero en algunas aplicaciones de robots móviles, puede ser necesario un conocimiento de los dos objetos y espacio libre, así mientras el robot viaja por el espacio libre, debe evitar chocar con los objetos, y en ocasiones podría tener que acercarse al objeto como parte de su tarea.

El método más común de realizar este tipo de mapa es el de la rejilla, que consiste en colocar una cuadrícula sobre el medioambiente de trabajo del robot.

En la figura 2.14 se puede observar como se localizan los objetos subdividiendo el ambiente de trabajo tantas veces como sea necesario hasta encontrar la ubicación exacta de objetos y espacio libre.

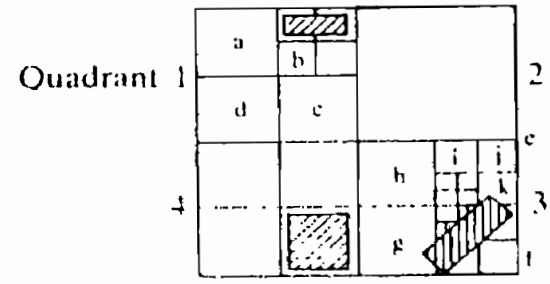


Figura 2.14 Estructura de cuadrícula que almacena una resolución múltiple

2.3.5 Planeación del Camino

Los requerimientos para planear el camino de un robot móvil se pueden resumir en los siguientes:

1. Encontrar un camino en el medioambiente de trabajo del robot, donde pueda navegar libremente sin chocar con ningún objeto.
2. Manejar la incertidumbre en el modelo sensado y los errores en la ejecución del camino.
3. Minimizar la posibilidad de un impacto con los objetos en el área de visión de los sensores del robot móvil, ubicando al robot lejos de los objetos.
4. Encontrar el camino óptimo para el robot móvil, desde su inicio hasta llegar a su meta.

Los requerimientos antes mencionados, no son aplicados en todas las situaciones, así en algunas aplicaciones se podrían aplicar un número mayor de requerimientos.

Para cumplir con estos requerimientos y de acuerdo a la forma en que se realizó el mapa del medioambiente del robot, se aplican diferentes tipos de algoritmos, mayor información encuentran en referencia [2].

2.4 METODOS DE POSICIONAMIENTO DE ROBOTS MOVILES

De lo disentido en la sección anterior, se concluye que se pueden aplicar diferentes métodos de posicionamiento, de acuerdo a la forma en que el robot móvil esté navegando. Así, para corregir su posición podría utilizarse por ejemplo encoders, corrección interna del error de posición, elementos auxiliares de navegación, sistemas de triangulación, etc.

En el resto de esta sección se hará referencia a algunos de los métodos de posicionamiento que se utilizan en robots móviles.

2.4.1 Ruedas Auxiliares y Encoders Básicos de Remolque

Para mejorar el error de posición de los robots móviles, se pueden colocar encoders en las ruedas del robot (como se ve en la figura 2.8), a través de ellos se obtiene la información de la posición del robot, con esa información se puede implementar un sistema de control de posición.

El encoder básico de remolque, como se puede ver en la figura 2.15, está constituido por dos ruedas auxiliares colocadas en un remolque, el cual se acoplaría al robot móvil. Igual que en el caso anterior, este encoder daría la información de la posición del robot.

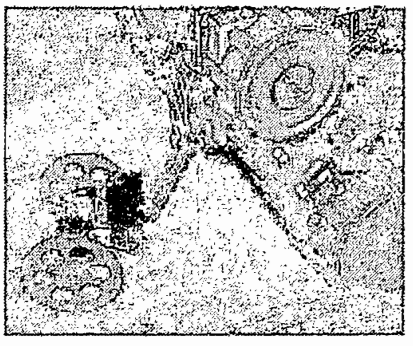


Figura 2.15. Encoder Básico de Remolque

2.4.2 Sistemas Activos de Navegación por Faros

En este tipo de sistemas, el robot móvil calcula su posición basándose en los faros guías colocados en su ambiente de trabajo. Este cálculo se puede realizar de dos formas: trilateración y triangulación.

La trilateración es la determinación de la posición del robot, sobre la base de la distancia medida desde el robot a los faros guías, generalmente se

colocan tres a más transmisores en el ambiente de trabajo y un receptor sobre el robot.

La triangulación se la realiza basándose en faros transmisores (generalmente infrarrojos) colocados en lugares conocidos, el robot detecta los faros con un receptor rotativo colocado sobre él.

El lector interesado encontrará mayor información sobre métodos de posicionamiento en la referencia [3].

2.5 MODELO CINEMATICO DEL ROBOT MOVIL DE TRACCION DIFERENCIAL

En esta sección se describe un método de control para determinar las velocidades rotacional y lineal del vehículo [3]. La estabilidad del sistema de control se verifica mediante la teoría de Lyapunov. Una de las dificultades de este problema reside en el hecho de que el vehículo posee solamente dos grados de libertad (velocidad lineal υ y velocidad angular ϖ) para el control del desplazamiento, sin embargo existen tres variables a controlar [x y θ], posición y orientación respectivamente. Otra dificultad es la no linealidad de la relación cinemática entre $(\upsilon, \varpi)^t$ y $(x,y,z)^t$.

2.5.1Formulación del Problema

Se tiene un robot móvil localizado en el plano 2D en el cual se define un sistema de coordenadas Cartesiano global. La postura del robot se representa por el vector de estado:

$$\mathbf{p} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \qquad (1)$$

donde θ es la orientación del robot respecto al eje x. El **0** denota posición nula $(0, 0, 2n\pi)^t$ donde n es un entero. Ya que el robot tiene la capacidad de

locomoción en el plano, la posición **p** es una función del tiempo *t*. Todas las localizaciones (x(*t*), y(*t*)) determinan el *camino* o *trayectoria*. Si las derivadas en tiempo x y y existen, $\theta(t)$ es:

$$\theta(t) = tan^{-1}\left(\frac{\dot{x}}{\dot{y}}\right) \tag{2}$$

El movimiento del vehículo está controlado por su velocidad lineal υ y la velocidad rotacional ω las cuales también son funciones del tiempo. La cinemática está definida por la matriz Jacobiana *J*:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ 0 \end{bmatrix} = \dot{p} = J \cdot q \equiv \begin{bmatrix} \cos\theta & 0 \\ sin\theta & 0 \\ 0 & 1 \end{bmatrix} \cdot q \tag{3}$$

donde: **q** = (υ, ω).

Esta cinemática es común a todas las clases de vehículos que no son omni-direccionales, es decir vehículos que no tienen múltiples grados de libertad (Por ejemplo, un automóvil, una bicicleta, un vehículo con dos llantas paralelas independientes, y un triciclo).

En el siguiente capítulo se presentará un ejemplo de simulación implementado en MATLAB, en el cual se utilizan los algoritmos de control descritos en [3], con la cual se verifica la validez del método.

Capítulo 3

SIMULACION DE ROBOTS MOVILES

3.1 INTRODUCCION

En los últimos años la investigación sobre robots móviles ha tenido un gran desarrollo, esto se debe principalmente al desarrollo de simuladores en los cuales se pueden probar diferentes algoritmos de control antes de ser implantados en el robot real. Estas sofisticadas herramientas se han desarrollado para sistemas UNIX, siendo sus costos elevados y en cierta manera prohibitivos, ya que se encuentran en el orden de los miles de dólares.

Por lo anotado anteriormente y por la poca difusión de simuladores en un sistema operativo WINDOWS, en la presente tesis se desarrolló un simulador con características similares a los de tipo comercial utilizando para ello el programa *Microsoft Visual C++* para Windows 95, el mismo que será detallado en el capítulo siguiente.

Tanto el prototipo como el simulador implementado utilizan el mismo sistema de navegación, por lo que utilizando el simulador se puede de una manera muy aproximada conocer el funcionamiento del prototipo en diferentes ambientes de trabajo. Sin embargo, cabe mencionar que los simuladores tienen limitaciones ya que no se pueden simular todos los tipos de sensores, y no se podría predecir un funcionamiento erróneo de los mismos, como por ejemplo malas medidas de sensores ultrasónicos debido a reflexiones [4].

3.2 IMPORTANCIA DE LA SIMULACION EN ROBOTS MOVILES

Un simulador es una herramienta muy importante con la cual se pueden resolver muchos problemas en el desarrollo de plataformas móviles, probando algoritmos sin correr ningún riesgo, ya que estos se ejecutan solamente en el computador y no en el robot real.

La importancia de la simulación puede ser mejor apreciada si se miran las dificultades que implican los diseños de plataformas móviles, cuyo sistema inteligente está basado en sensores. Con la información obtenida de los sensores se ejecutan acciones de control de movimiento. Lamentablemente estos sistemas acarrean errores debido al sistema de odometría tales como: patinaje de las ruedas, el valor óptimo del tiempo de muestreo y los problemas de estabilidad de los algoritmos de control. Mediante simulación, los investigadores pueden construir muchos ambientes de trabajo y monitorear la operación del robot por muchas horas, probando diferentes algoritmos y mejorándolos. La simulación hace que el uso del robot real sea mínimo, evitando varios problemas prácticos como por ejemplo la recarga de baterías o la reconfiguración de ambientes.

Lo indicado anteriormente se evidenció en el desarrollo de la tesis, ya que con el simulador se apreció algunas deficiencias en el sistema de navegación del prototipo, las cuales se las corrigió a tiempo.

3.3 ARQUITECTURA DE UN AMBIENTE DE SIMULACION

Un ambiente de simulación está constituido por módulos que realizan una actividad específica, y en conjunto hacen que trabaje de manera similar al robot real.

Un simulador para robots móviles tiene los siguientes módulos:

Simulación del Medio Ambiente.-
En el cual se pueden construir diferentes configuraciones del entorno del robot.

Simulación de los Sensores.-
Donde se modela de la manera más fiel posible el funcionamiento de los sensores utilizados.

Simulación del Robot.-

Está constituido por los algoritmos que modelan el funcionamiento de la plataforma móvil utilizada.

Interfaz Hombre Máquina.-

Es un elemento de alto nivel, el cual permite al usuario probar los módulos descritos anteriormente, y le da control total para manipularlos y realizar las pruebas necesarias para simular el funcionamiento del robot real. Información adicional se la puede encontrar en la referencia [4].

3.3.1 Arquitectura del Programa de Simulación *Pionero 1*

El programa de simulación tomó el nombre de la plataforma móvil (*Pionero1*), y se encuentra implementado por una programación orientada a objetos, esto genera un manejo modular de las características del simulador. Las partes principales son:

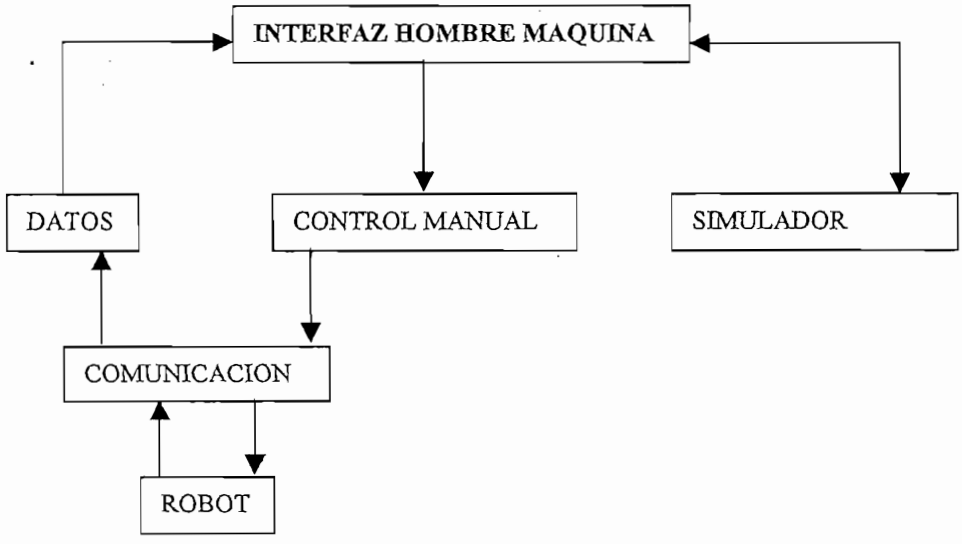


Figura 3.1 Arquitectura del Programa de Simulación *Pionero1*

La arquitectura del simulador desarrollado, como se muestra en la figura 3.1, consta de los siguientes módulos:

Interfaz Hombre Máquina.- Este es el medio por el cual los usuarios interactúan con el software. Sirve para enviar comandos y recibir información tanto del simulador como de la comunicación.

Simulador.- Donde se puede configurar un ambiente de trabajo con obstáculos. Un objeto robot es creado para navegar en el medio simulado el cual incluye la simulación de sensores de ultrasonido. Con las funciones del objeto robot se pueden simular características del robot real.

Control Manual.- Para tener un control manual del robot real, por medio de comandos desde el Interface de usuario y de una comunicación serial, se accede a los comandos de navegación en el controlador del robot real.

Datos.- El robot cuenta con módulos para adquirir datos del medio que le rodea, como temperatura y cantidad de luz. A través de un conversor A/D y de la comunicación, se pueden mostrar datos en el interface de usuario.

Comunicación.- Por medio de una comunicación serial se puede acceder a los datos y comandos de robot real y, por lo tanto, manejarlo desde el programa de usuario.

Robot.- Construcción mecánica del objeto móvil con capacidades de navegación simple. El robot tiene un software de control que le permite evadir obstáculos con total autonomía con ayuda de sensores de ultrasonido.

3.4 APLICACIONES

En esta sección se mostrarán dos aplicaciones de simulación, en primer lugar un simulador comercial llamado Nomadic Software Environment, desarrollado para el robot móvil Nomad 200, mostrado en la figura 3.2, este ambiente se divide en dos partes denominadas: On-board Programing Environment, y Host Programing Environment, la primera provee compatibilidad con la base UNIX del ambiente central, contiene una librería de funciones, que hacen que el ambiente central funcione de manera similar al robot real, y además tiene funciones de bajo nivel con las cuales se tiene acceso a los sensores y al control de los motores, la segunda parte es una interfaz gráfica en la que se aprecian cuatro módulos: Interfaz Robot - control central, Simulación del Robot, Interfaz gráfica para el usuario, comunicación cliente servidor, similares a las anotadas en la sección anterior. En la figura 3.3 se aprecia la arquitectura de este simulador.

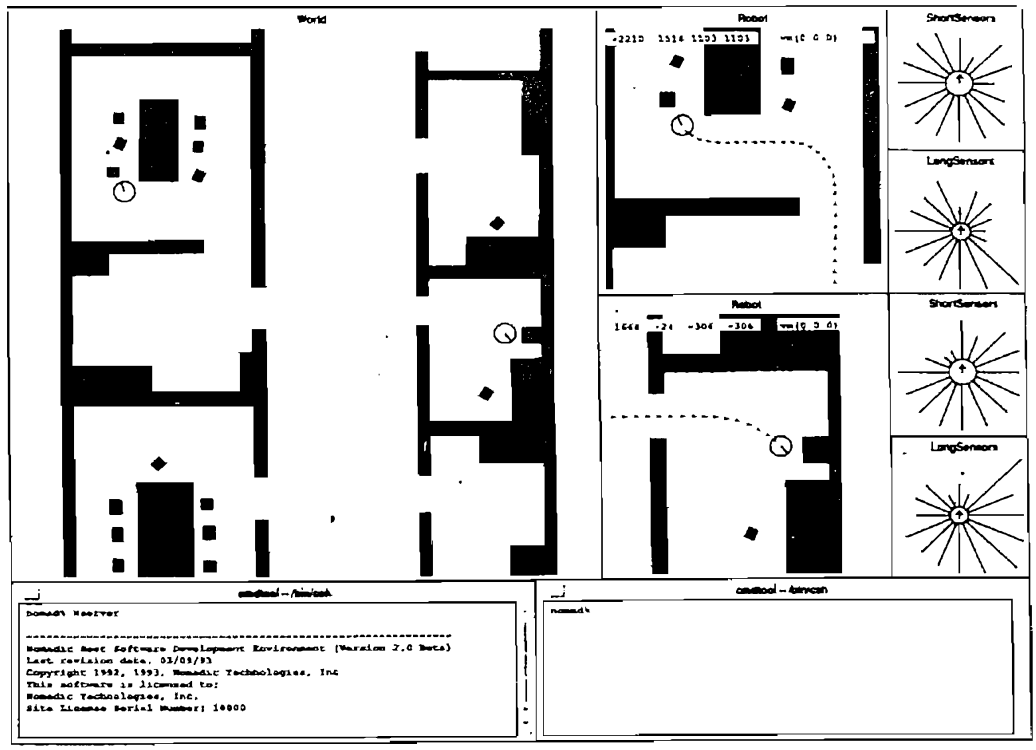


Figura 3.2 Simulador Nomadic

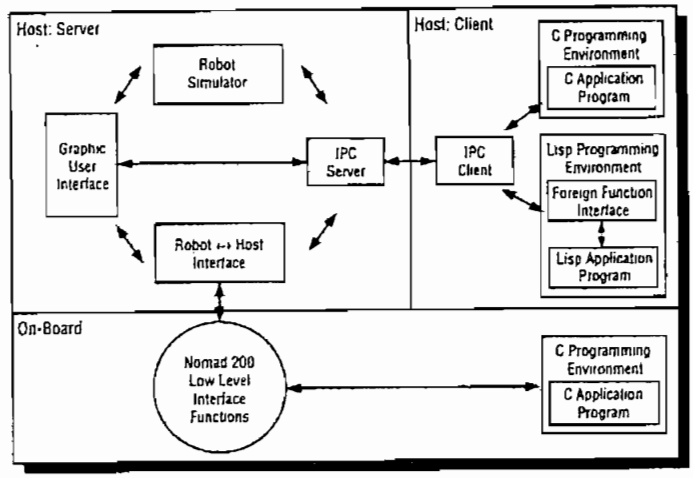


Figura 3.3 Arquitectura del Software de Simulación Nomadic

Finalmente se presenta una simulación implementada en MATLAB que permite verificar la validez del método de control para la plataforma móvil de tracción diferencial descrita por los autores en [3].

Por medio de funciones y archivos del Simulink se muestra tres ejemplos para la simulación. Los archivos son:

Demo01.m	archivo menú de selección
Robot1.m	archivo de ejemplo *función paso en Y*
Robot2.m	archivo de ejemplo *Recta*
Robot3.m	archivo de ejemplo *Círculo*

Para empezar, dentro del programa se hace un llamado al archivo del menú de selección con el que se puede escoger el ejemplo a simular. En la ventana de comandos se escribe demo01:

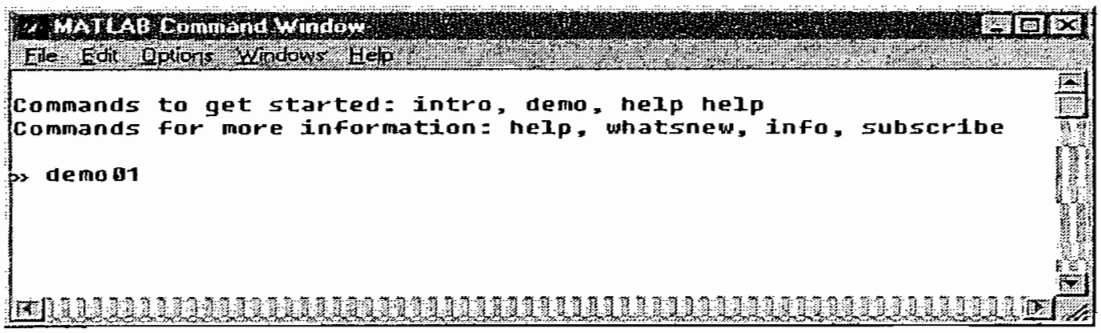


Figura 3.4 Ventana de Comamdos de Matlab

De inmediato aparece la ventana del menú:

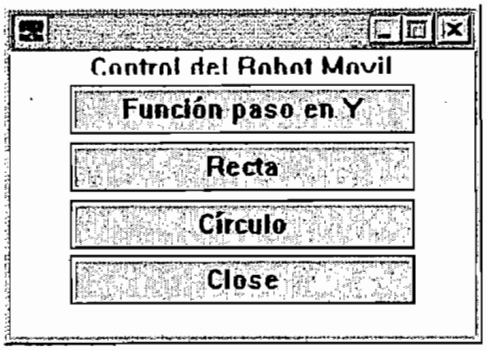


Figura 3.5 Ventana de Menu

El usuario puede escoger cualquiera de los ejemplos, aparecerá entonces la ventana donde se encuentra el diagrama de control y tres ventanas de ayuda, en el que después de correr la simulación, aparecen la respuesta en el tiempo de la dirección, eje X y eje Y, finalmente aparece una ventana adicional en el que se tiene la simulación en el espacio XY, que corresponde al camino recorrido por el robot. A continuación se describe cada ejemplo:

Función paso en Y: Para probar el control, al robot se lo mueve en dirección del eje X y luego se aplica una función paso en la entrada de control de manera que el robot siga la dirección del eje X pero con una variación en Y, se observa la respuesta del sistema en la figura 3.8.

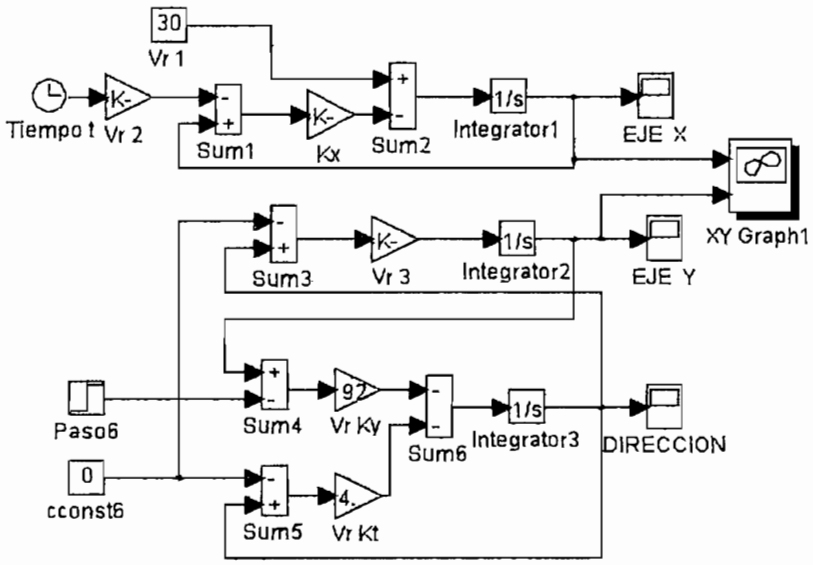


Figura 3.6 Diagrama de Control, ejemplo Función paso en Y

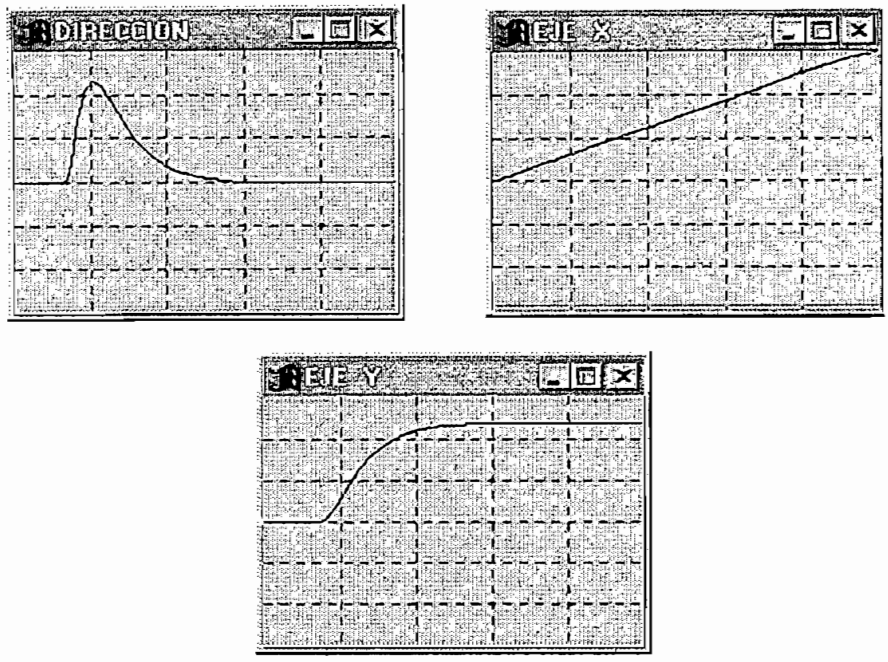


Figura 3.7 Ventanas de Ayuda

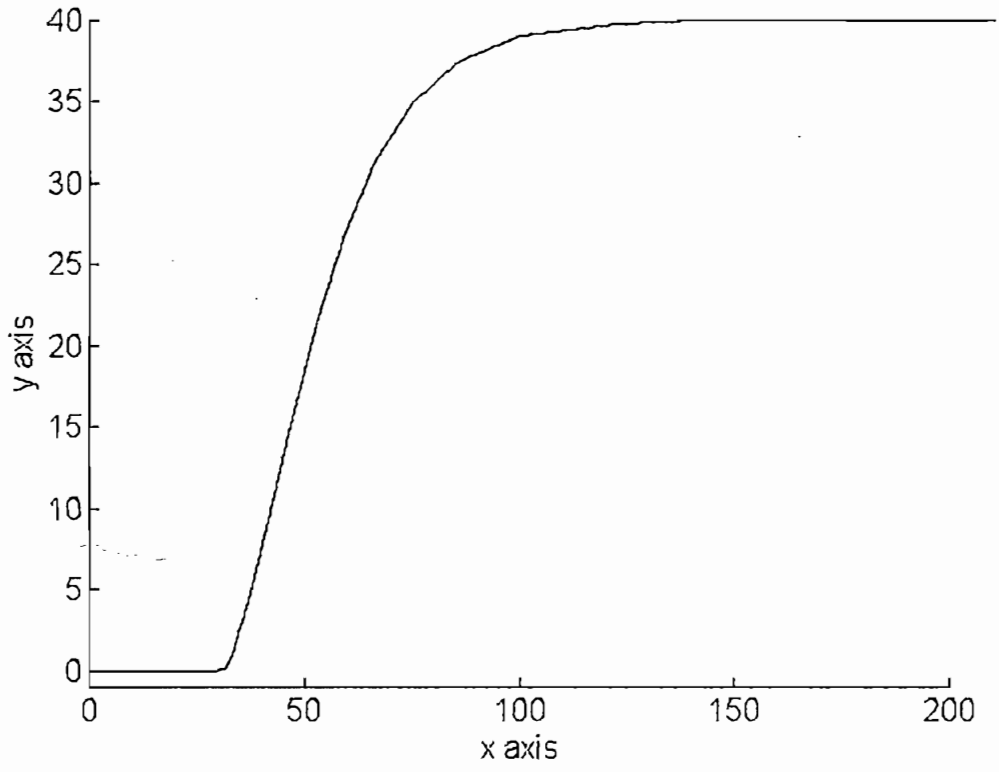


Figura 3.8 Camino recorrido por el Robot

Recta: En este caso, la señal de entrada se incrementa linealmente tanto en el eje X como en el Y. El robot parte con una dirección inicial de 0 grados respecto al eje X, por lo tanto en esta simulación se obtiene la respuesta de la dirección que toma el robot hasta alcanzar la señal de entrada, figura 3.11.

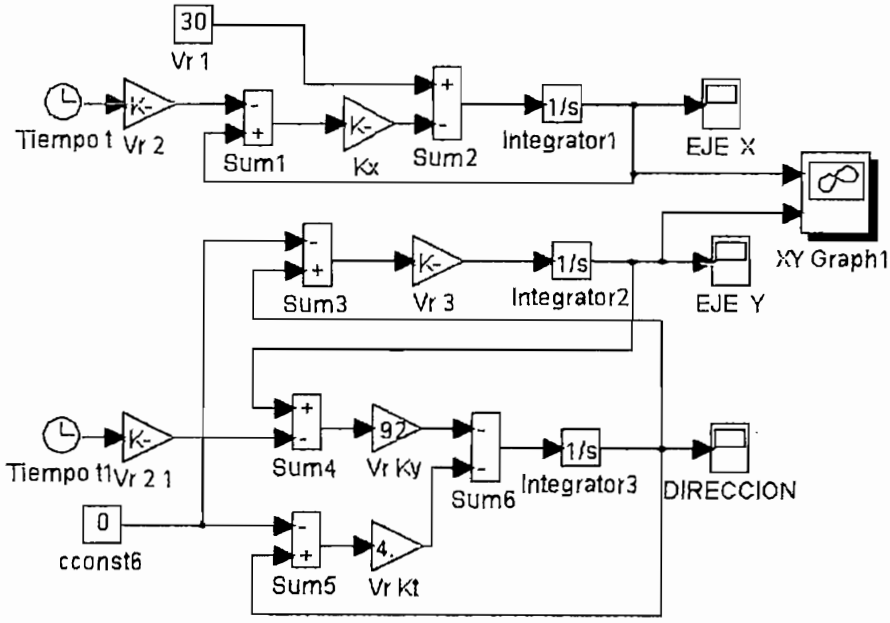


Figura 3.9 Diagrama de Simulación Ejemplo Recta

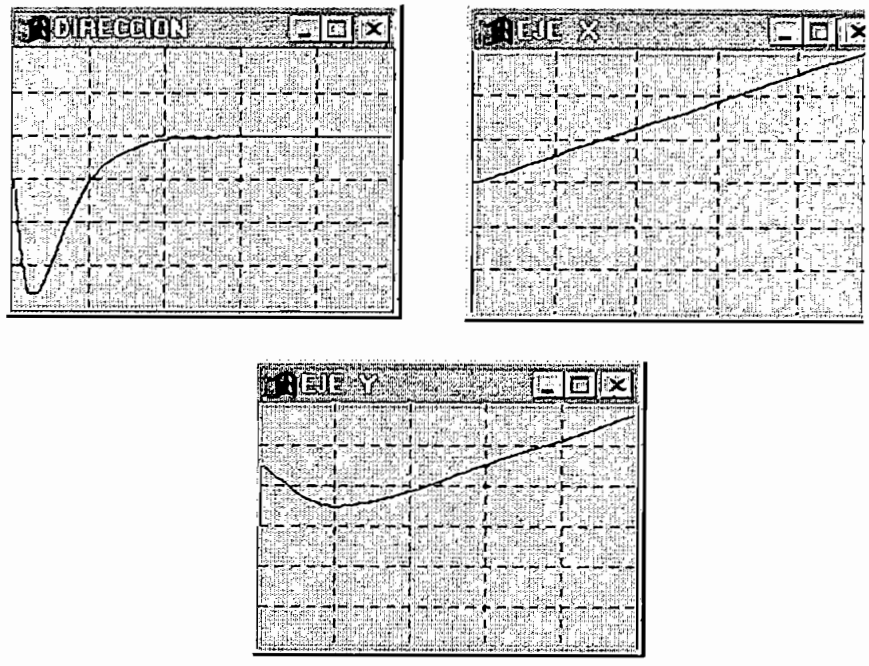


Figura 3.10 Ventanas de Ayuda

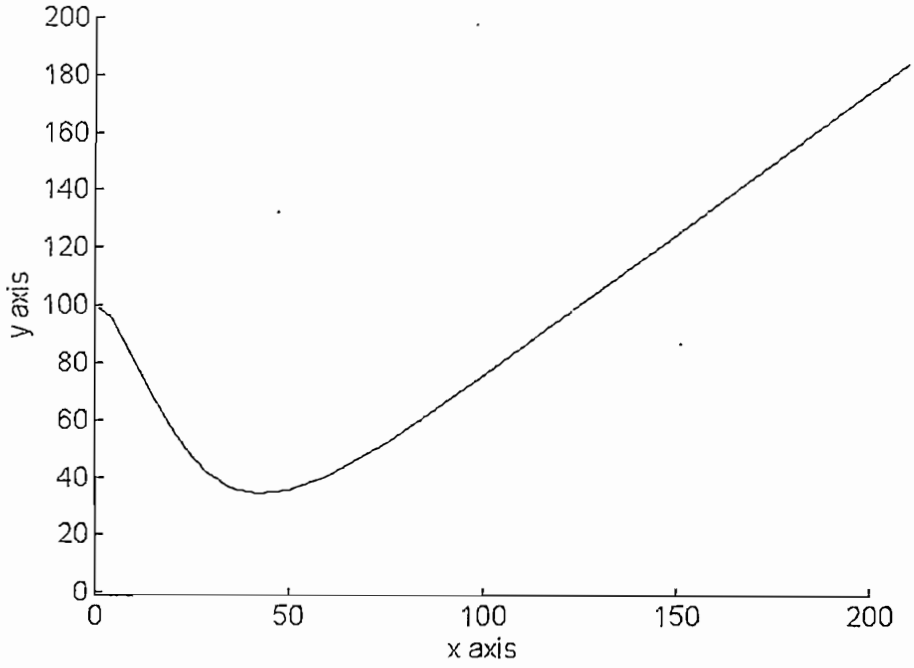


Figura 3.11 Camino recorrido por el robot

Círculo: En esta simulación se genera una entrada senoidal en el tiempo tanto para el eje X como para el Y, obteniendo así una señal de entrada circular. El robot parte con una dirección 0 (eje X). En los gráficos se observa como el control hace que el robot se ajuste a la señal de entrada (figura 3.16).

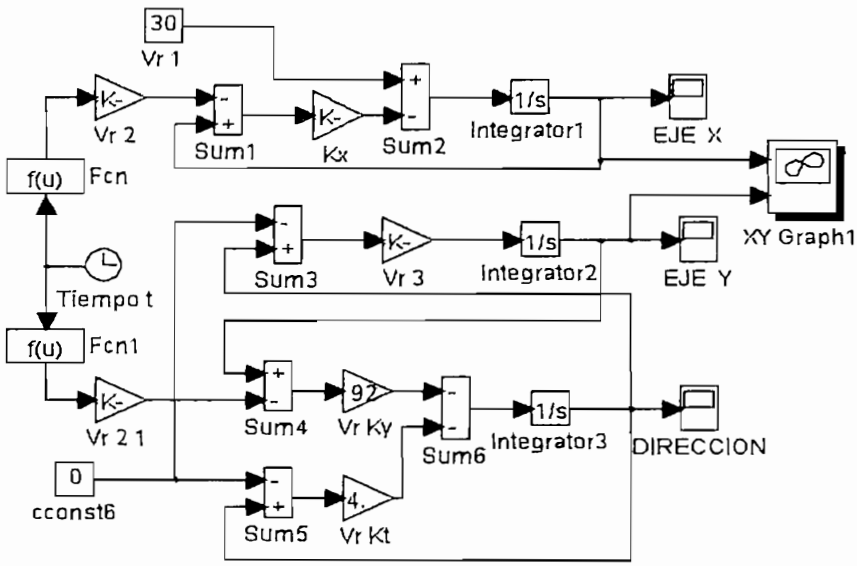


Figura 3.12 Diagrama de Simulación, Ejemplo Círculo

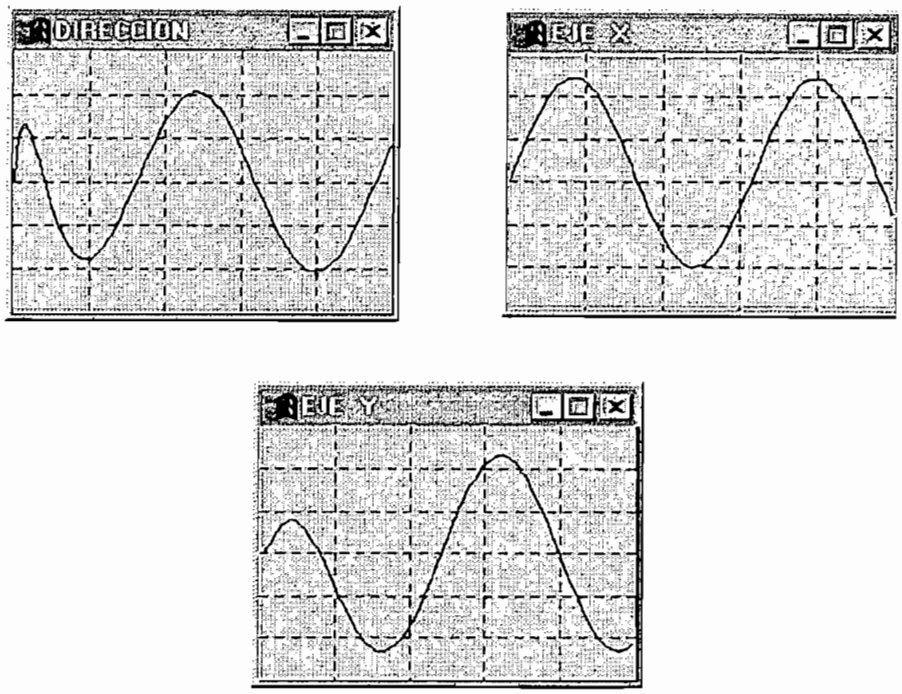


Figura 3.13 Ventanas de Ayuda

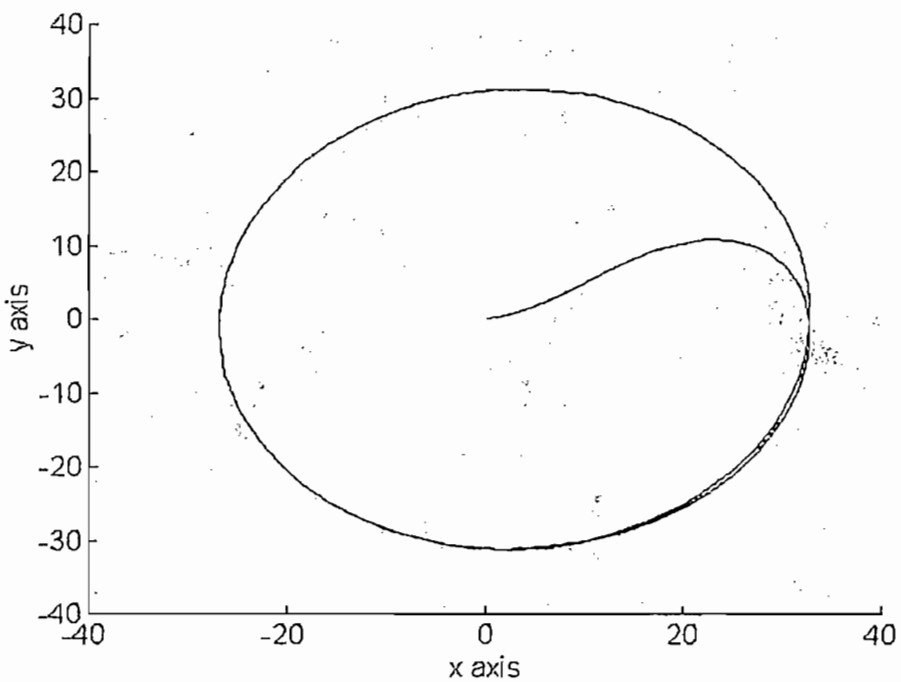


Figura 3.14 Camino recorrido por el Robot

Capítulo 4

4.1 DESCRIPCIÓN DEL PROGRAMA

El simulador está desarrollado en el lenguaje de programación orientado a objetos Visual C ++ utilizando las librerías MFC (Microsoft Fundation Class) para sistemas basados en ambiente Windows. Visual C++ utiliza una interfaz gráfica con un gran número de ayudas y herramientas denominado Developer Studio.

Para la creación de programas se puede recurrir a una ayuda llamada AppWizard. Este genera un completo juego de archivos iniciales con todas las características para la construcción de una aplicación básica y luego se le añaden funciones, comandos y datos para un propósito en particular, en este caso un simulador para el robot móvil *PIONERO 1*.

4.1.1 Estructura Básica del Programa

Al crear una aplicación con AppWizard se genera un proyecto que consta de las siguientes partes principales [15]:

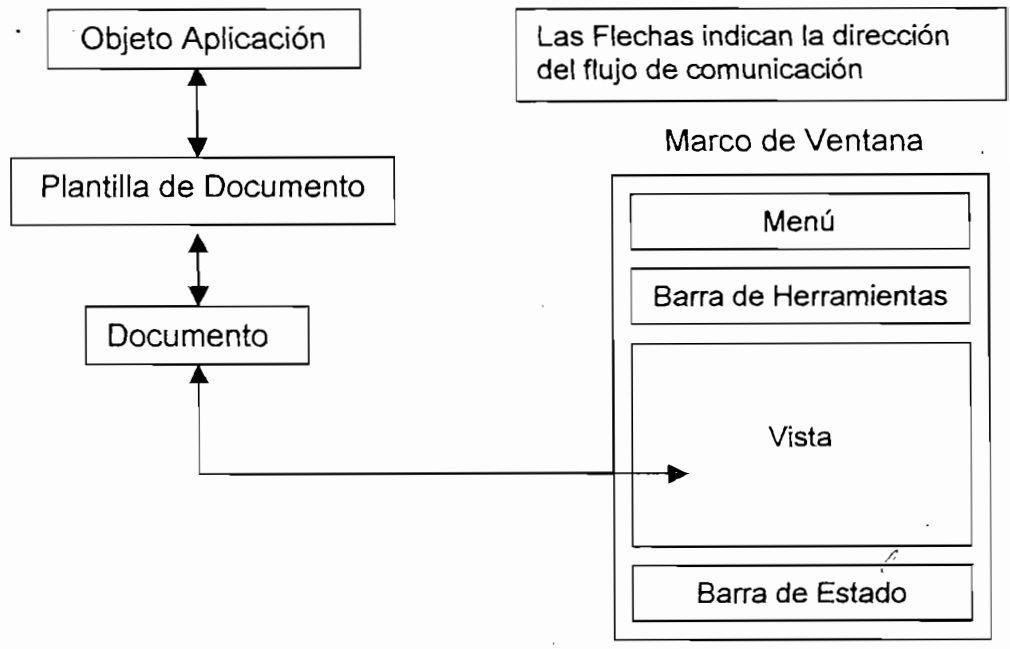


Figura. 4.1 Configuración Básica

- Objeto Aplicación
- Plantilla de Documento
- Objeto Documento
- Objeto Vista (View)
- Objeto Marco Principal de Ventana (Main Frame Window)

- Objeto Aplicación: controla todo acerca de los objetos y especifica el comportamiento de la aplicación así como la iniciación y término de la misma.

- Plantilla de Documento: una aplicación puede soportar más de un tipo de documento, por ejemplo hoja de cálculo, documento de texto o documentos gráficos. Cada uno de estos documentos tienen asociado una plantilla de documento y ésta especifica el tipo de recurso (por ejemplo: menú, icono, etc.) que será usado para cada tipo de documento.

- Objeto Documento: es el encargado de especificar el tratamiento de los datos de la aplicación. Un documento representa el conjunto de datos que el usuario típicamente abre con el comando *Abrir* y guarda con el comando *Guardar* del menú *Archivo.*

- Objeto Vista: éste maneja la interacción entre el usuario y los datos del documento. La Vista es una ventana hija que llena el área cliente del cuadro de ventana principal.

- Marco Principal de Ventana: cuadro que contiene al objeto Vista o Cuadro de Documento. En él se especifica el estilo y otras características del Marco_Principal de la Ventana.

Al utilizar la biblioteca de clases MFC de Visual C++, se crean todos los archivos necesarios que forman un esqueleto base con los objetos mencionados, y son el punto de partida para la creación de una aplicación.

Es necesario entonces añadir los objetos, funciones y comandos para que la aplicación responda a las necesidades para las que fue creada.

4.1.2 Estructura Personalizada del Programa

Para que la aplicación funcione como el Simulador del Robot Móvil se toma en consideración los siguientes puntos:

- Es necesario representar un ambiente gráfico que conste de un área donde el robot pueda desplazarse y puedan colocarse obstáculos. Estos pueden estar constituidos por formas geométricas básicas. Para tal efecto se crea la clase FORMA la que contiene los datos y funciones de las formas Rectángulo y Elipse que serán los obstáculos del ambiente gráfico del simulador.

- El Robot Móvil debe tener la capacidad de moverse en el medio y evitar los obstáculos con la simulación de los sensores de ultrasonido utilizados. En este caso se utiliza la clase MOVIL que consta de una figura que simula el robot real con funciones que permitan la animación, además incluye una representación de un área gráfica que simula el alcance de los sensores. Dentro de esta clase se define el comportamiento del móvil frente a un obstáculo reproduciendo el mismo algoritmo implementado en el robot real.

- Como el prototipo tiene la capacidad de control a distancia por medio de comunicación serial, se implementa un cuadro de dialogo para este fin en el que consta botones para la navegación y comandos para la lectura de sensores.

- Para ciertas características del simulador se incluyen ayudas para el manejo del mismo, en forma de cuadros de diálogo para fijar valores del simulador, como desplazamiento del robot y velocidad de animación, y configuración de los sensores.

4.1.3 Arquitectura del Software

La figura 4.2 muestra el esquema de la arquitectura funcional del software. Básicamente está conformado por una Interfaz Hombre Máquina que abarca todos los objetos gráficos con que el usuario puede interactuar como son: Barra de Herramientas, Cajas de Diálogo con información de datos que pueden ser fijados, Barras de Estado que indican características del simulador y un Menú.

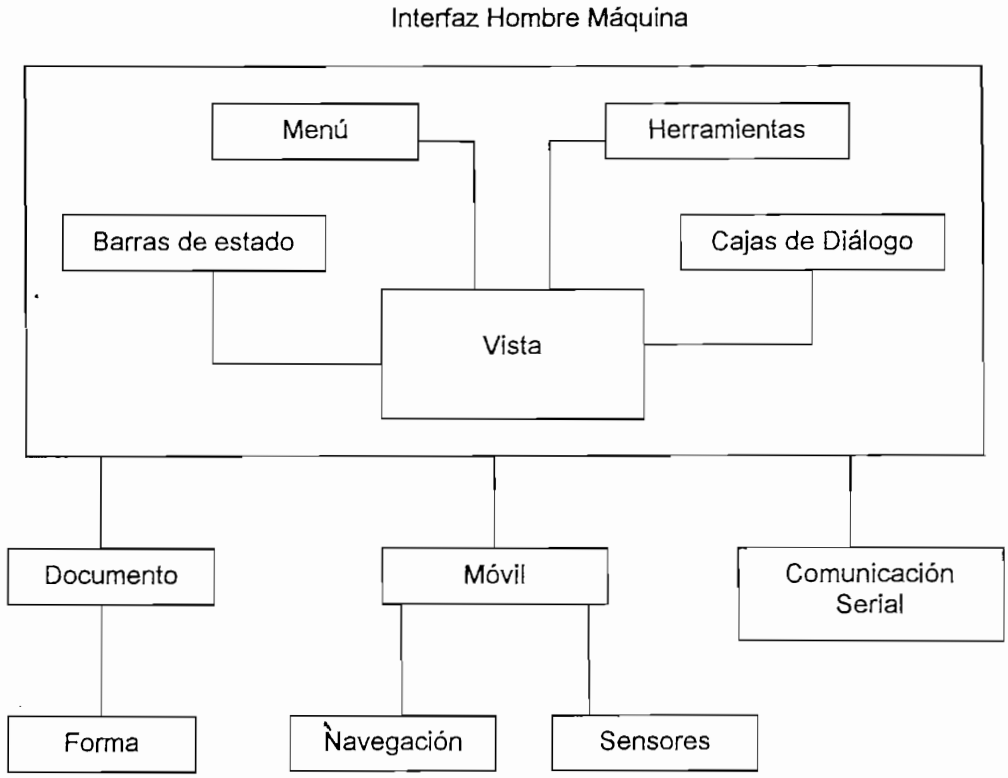


Figura. 4.2 Arquitectura del Software

Por otro lado se encuentra el Documento que guarda la información de los obstáculos que son de la clase Forma. También el objeto Móvil se encarga del proceso de navegación y la simulación de los sensores. La comunicación serial conecta al usuario (Computador Principal) con el Prototipo a través de una caja de diálogo que contiene controles, para la navegación manual o automática, e indicadores del estado de los sensores.

Para la descripción detallada de la arquitectura del simulador se dividirá en dos grupos: Interfaz Gráfica y Objetos del Simulador.

4.1.3.1 Interfaz Gráfica

- **Vista**

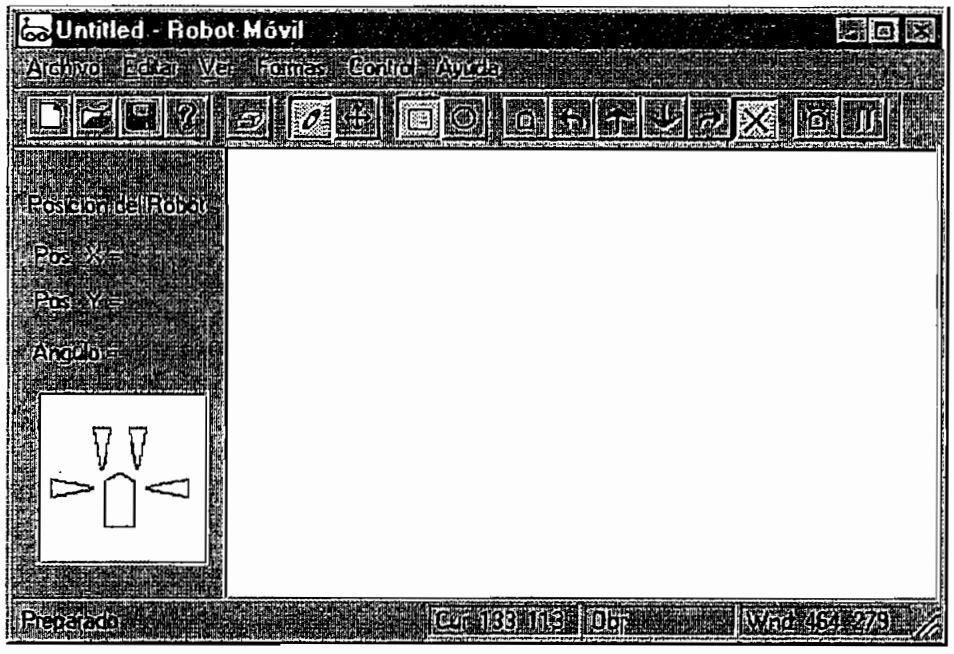


Figura 4.3 Interfaz Gráfica

Es una ventana que encierra el área de trabajo donde se crea un ambiente para la navegación del móvil.

- **Barra de Herramientas**

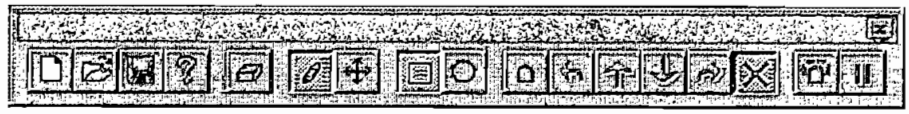


Figura 4.4 Barra de Herramientas

Conjunto de botones que ejecutan una función específica:

Funciones básicas: crear un nuevo ambiente, abrir o guardar un ambiente e Información del Programa.

Funciones de Edición: borrar todo, crear formas, cambiar el tamaño y posición de las formas, escoger tipos de formas (rectángulo y elipse).

Funciones de Navegación: navegación manual, dirección del movimiento (izquierda, adelante, atrás, derecha, parar), navegación automática y pausa.

- **Barras de Estado**

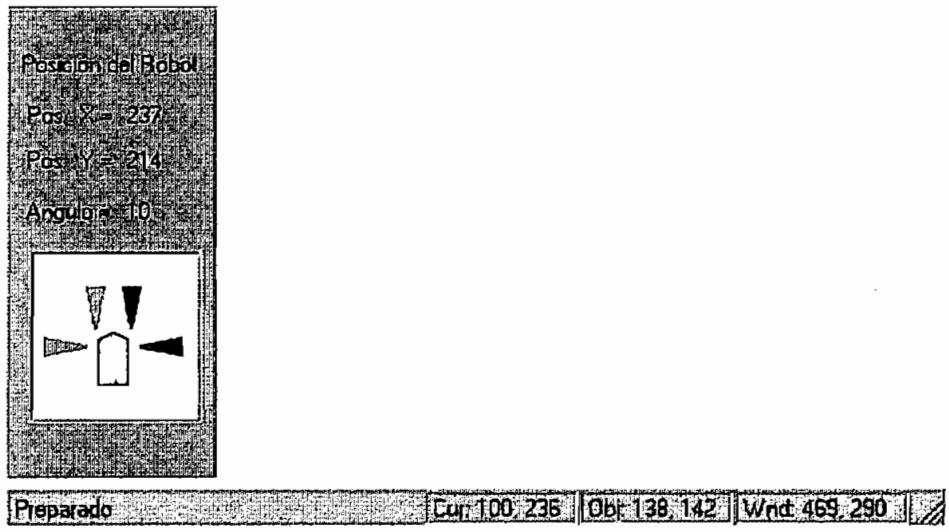


Figura 4.5 Barra de Estado

Muestran información de la posición del robot y el estado de los sensores. Además en la parte inferior se muestra la posición actual del cursor, el tamaño del último objeto (obstáculo) y el tamaño de la ventana.

- **Cajas de Diálogo**

Menú: Control – Retardo...

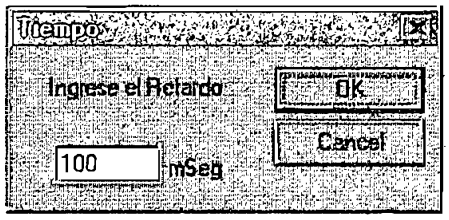


Figura 4.6 Control Retardo

En este Diálogo se puede establecer el tiempo de retardo para la animación. Este tiempo esta comprendido entre una posición y la posición siguiente después de un movimiento. La velocidad de desplazamiento también depende de las capacidades del computador.

Menú: Control – Desplazamiento...

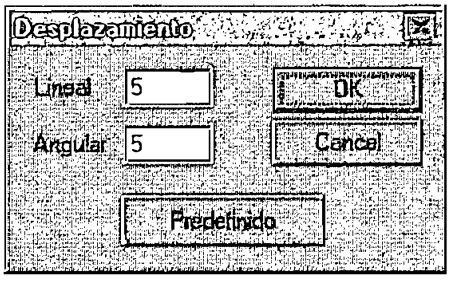


Figura 4.7 Control Desplazamiento

A través de éste Diálogo se puede fijar el desplazamiento, en pixeles, Lineal y Angular por separado dependiendo que movimiento se desea que sea más rápido.

Menú: Control – Aleatorio...

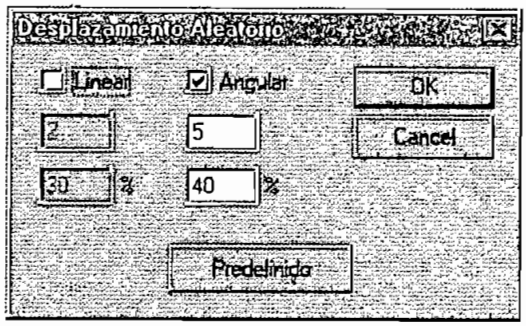


Figura 4.8 Control Desplazamiento Aleatorio

Como una ayuda para la simulación se puede añadir al desplazamiento del móvil cantidades de movimiento aleatorio (en pixeles), que se añaden al desplazamiento normal. La frecuencia de que ocurra este desplazamiento angular se puede valorizar entre el 0% y 100%. El desplazamiento aleatorio se puede habilitar o deshabitar independientemente.

Menú: Control – Sensores...

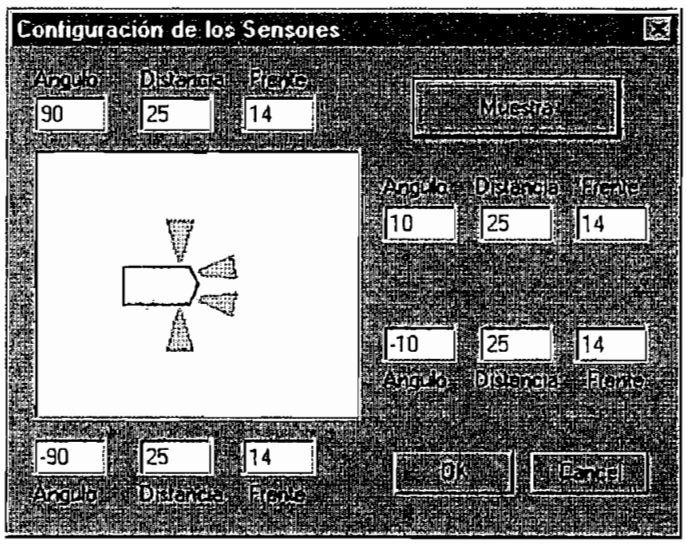


Figura 4.9 Configuración de Sensores

Es un cuadro de Diálogo un poco más complejo que los anteriores pero de mucha utilidad ya que puede modificar el alcance de los sensores para la simulación y ajustarse de acuerdo al robot real. Los parámetros se ajustan de acuerdo al siguiente gráfico:

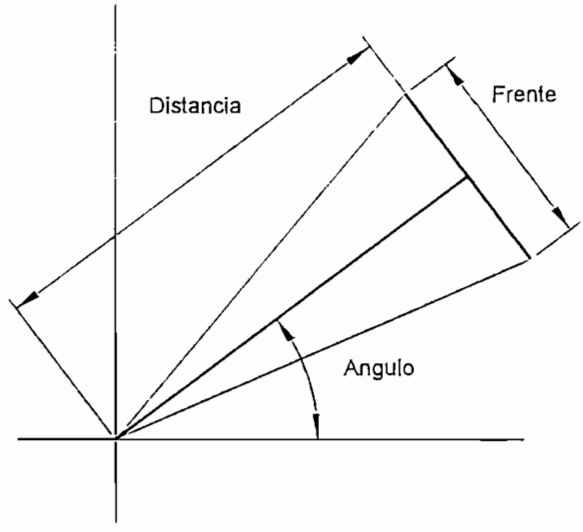


Figura 4.10 Datos del Sensor

- **Menús**

Los menús principales son:

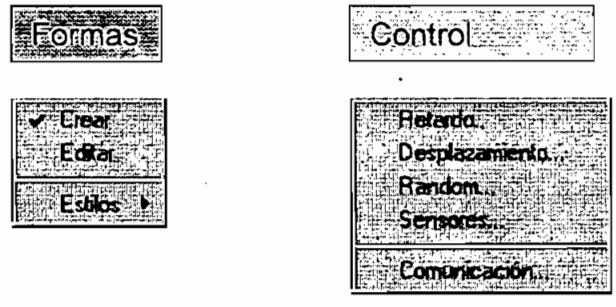


Figura 4.11 Menús

Con el menú Formas se puede escoger el modo de Crear o Editar, además de fijar el estilo de las Formas (Rectángulo, Elipse). En el menú control se encuentran todos los Cuadros de Diálogo vistos anteriormente y un cuadro adicional que es para la Comunicación Serial con el prototipo real.

4.1.3.2 Objetos del Simulador

- **Documento y Formas**

El objeto Documento es el encargado de administrar y guardar los datos necesarios para la visualización del ambiente de trabajo. Cada Obstáculo que se crea se guarda en una lista que se incrementa a medida que se crean más obstáculos. También mantiene datos a cerca del Estilo del objeto y del color. Existen además funciones que permiten el manejo y enlace de datos entre otros objetos. Las principales funciones son:

`PonObstac()`	Añade una Forma a la lista de Obstáculos.
`DeleteContents()`	Borra toda la lista de Obstáculos.
`HitTest()`	Comprueba si el mouse ha sido pulsado sobre un obstáculo.
`OnEditBorrar()`	Llama a la función `DeleteContents()`.
`OnEstiloRect()`	Especifica el estilo del Obstáculo a Forma Rectángulo.
`OnEstiloElipse()`	Especifica el estilo del Obstáculo a Forma Elipse.

La clase *Forma* consta de funciones y variables miembros que encierran las características de los objetos Obstáculo. Para dar formato a estos, guarda datos a cerca del Estilo, Color y Tamaño. Las funciones principales son:

`CForma()`	Función constructora de un obstáculo.
`Dibuja()`	Encargada del procedimiento para dibujar un objeto Obstáculo.

`Serialize()`	Función de la librería MFC encargada de guardar los obstáculos y sus características en un archivo.

El usuario interactúa con la interfaz gráfica que está controlada por el objeto Vista (View), y ésta a su vez interactúa con los demás objetos delegando los trabajos específicos para cada uno como por ejemplo la forma de dibujar los obstáculos y la forma de guardar los datos en un archivo.

- **Móvil**

Esta clase contiene las características operativas del objeto que representa al prototipo en forma de silueta que se desplaza por el área cliente del ambiente de trabajo. También contiene las funciones con el mismo algoritmo implementado en el robot real para la navegación. Además se encarga de la simulación del comportamiento de los sensores de ultrasonido implementado en el prototipo.

EL MOVIL: está constituido por una región definida por cinco puntos que representa la silueta del robot móvil. Existe concordancia entre las unidades reales, en centímetros, con los pixeles. Un pixel equivale a un centímetro como se muestra en la figura 4.12.

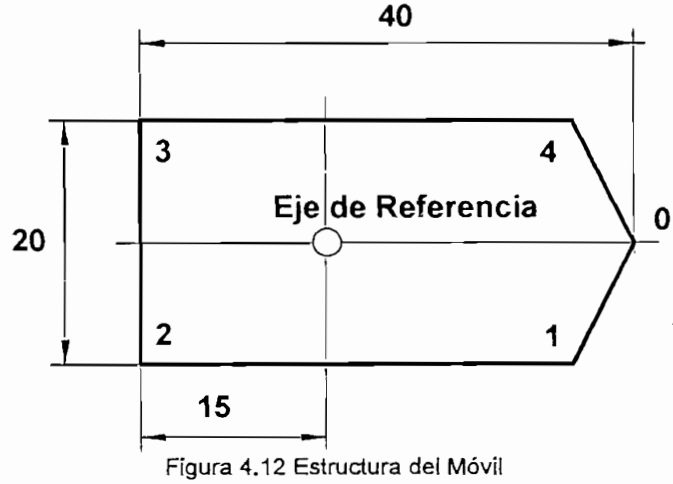


Figura 4.12 Estructura del Móvil

Cada uno de los puntos (0, 1, 2, 3, 4) que conforman el móvil está referido a un eje que representa el eje de rotación del prototipo cuando éste gira. Este eje sirve de referencia para el desplazamiento del móvil, por lo tanto la posición del mismo está dado por el punto del eje de referencia y la dirección con un ángulo como se muestra en la figura 4.13.

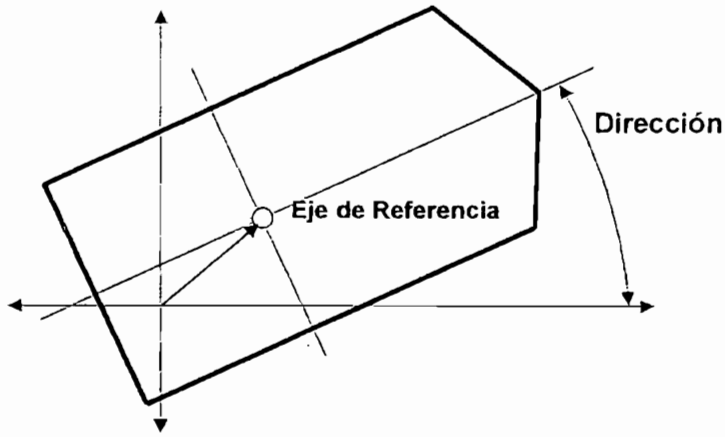


Figura 4.13 Desplazamiento del Móvil

Las posiciones de los puntos que conforman el móvil están guardadas en un arreglo o vector de datos. Por facilidad para el usuario en la modelación del robot, se utilizan coordenadas rectangulares para colocar cada punto respecto al eje de referencia y luego son transformadas a coordenadas polares, que son las que maneja el programa para desplazar al robot, ya que para realizar un giro a la izquierda o derecha se utilizan desplazamientos angulares.

Para encontrar las coordenadas de los vértices se utiliza la ayuda de un esquema como el siguiente:

```
// Configura Carro:
Ax[0] = 25; Ay[0] =  0;
Ax[1] = 20; Ay[1] =-10;
Ax[2] =-15; Ay[2] =-10;
```

```
Ax[3] =-15; Ay[3] = 10;
Ax[4] = 20; Ay[4] = 10;

// Punto 0 de sensores

Ax[5] = 15; Ay[5] = 10;
Ax[6] = 20; Ay[6] =  6;
Ax[7] = 20; Ay[7] = -6;
Ax[8] = 15; Ay[8] =-10;
```

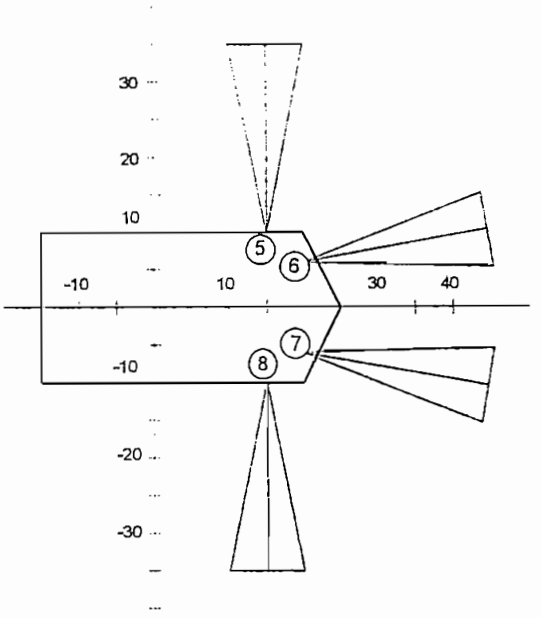


Figura 4.14 Mapeado del Móvil

Las principales funciones que determinan el comportamiento del Móvil son:

Iniciar()	Fija las variables con valores iniciales y transforma las coordenadas rectangulares, con las cuales se definieron los puntos, a coordenada polares, con las que trabaja el simulador.
PonPosicion()	Calcula la posición, después de un desplazamiento, de todos los puntos incluidos los sensores.
Mover()	Incrementa la posición con el valor del desplazamiento dependiendo de la dirección y luego llama a la función *PonPosicion()*.
DibujarMovil()	Selecciona el color del perímetro y del fondo de las figuras que comprenden el móvil y los sensores y dibuja las formas.

`Pol_Rec()`	Función que transforma de coordenas polares a rectancgulares.

LOS SENSORES: están representados por una figura triangular que muestra el alcance de los sensores de ultrasonido que están implementados en el robot real. Los sensores reales están acondicionados de tal manera que tienen la capacidad de ajuste de sensibilidad, que es la distancia de alcance de los sensores, y un ángulo de enfoque que puede ser fijado mecánicamente por tornillos.

De esta manera y por facilidad para el usuario, los datos que ingresan al sensor en el simulador están dados por tres magnitudes: distancia, ángulo y frente como se muestra en la figura 4.10.

Para simular el comportamiento del sensor real se aprovecha de la diferencia de colores que existe entre el área de trabajo (por donde puede desplazarse el móvil) y los obstáculos (por donde no puede pasar). Cada vez que se realiza un desplazamiento, el programa llama a la función `LeeSensores()` que informa si alguno de los cuatro sensores a encontrado un obstáculo haciendo una comparación del color leído en la posición del sensor con el color de fondo (área de trabajo) y si es diferente entoces señala obstáculo. Esta señalización es aprovechada por el algoritmo de control que toma una acción dependiendo que sensores están activados. Para los sensores se tienen las siguientes funciones:

`PonSensores()`	Transforma de las coordenadas polares a rectangulares para llenar la matriz de puntos que se desplazarán junto con el móvil.
`LeeSensores()`	Determina si alguno de los 4 sensores ha encontrado un obstáculo.

`VerSensores()` Función que determina la forma de visualizar los sensores dependiendo si está en el modo manual o automático.

EL CONTROL: para el desplazamiento del móvil se considera el mismo control de navegación que está implementado en el prototipo real.

En cada desplazamiento del robot, se llama a la función `Control()` que es parte del objeto Móvil. Esta función es la encargada de decidir hacia donde se moverá el robot dependiendo si un sensor está activo o no.

El algoritmo de la función `Control()` lee los sensores y realiza las siguientes tareas:

- Comprueba si uno de los sensores delanteros está activo (encontró un obstáculo), si es así, entonces llama a la función `Control_B()`.
- Comprueba si existe problema (cuando entra en un callejón), y llama a la función `Control_C()` mientras no exista salida (no puede girar a ningún lado).

`Control_B()`: Si puede girar a cualquier lado, escoge uno alternativamente. Si el sensor izquierdo está activo gira a la derecha, si el sensor derecho está activo gira a la izquierda y si ambos sensores están activos entonces existe problema (entró en un callejón).

`Control_C()`: Si el sensor izquierdo no está activo, entonces hay salida y gira a la izquierda. Si el sensor derecho no está activo hay salida y gira a la derecha. Si ambos sensores laterales están activos no hay salida y el desplazamiento es hacia atrás.

- **Vista**

El objeto Vista es el que mayor cantidad de funciones tiene ya que implementa el manejo de la Interfaz de Usuario y todos los comandos para que el programa funcione de acuerdo a nuestras necesidades. Las funciones principales son:

`OnDraw()`	Rutina principal que maneja la forma de dibujar los obstáculos y el móvil en la pantalla.
`OnTimer()`	Función encargada de cambiar de posición al móvil cada cierto tiempo. También llama a la función que muestra los sensores activos en la barra de diálogo.
`IniciarMovil()`	Fija los valores iniciales con que se presenta el móvil.
`CalculaMov()`	Utilizada para determinar la variación del desplazamiento teniendo en cuenta el efecto aleatorio.
`ActivarTimer()`	Activa el temporizador.
`ParарTimer()`	Detiene el temporizador por lo tanto la animación.
`MuestraSensorActivo()`	Encargado de indicar, con otro color, al sensor en la barra de diálogo cuando un sensor del móvil encuentra un obstáculo.

Cuando se pulsa un Boton

`OnBotonCrear()`	Pone en modo crear, establece la dirección parar, llama a la función *ParарTimer()*.

`OnBotonEditar()`	Pone en modo editar, y la dirección en parar.
`OnBotonMover()`	Pone en modo mover, la dirección en parar, llama a la función *ActivarTimer()* y cambia la forma de los sensores en la Barra de Diálogo.
`OnBotonAde()`	Pone en la dirección Adelante.
`OnBotonAtr()`	Pone en la dirección Atrás.
`OnBotonIzq()`	Pone en la dirección Izquierda.
`OnBotonParar()`	Pone en la dirección Parar.
`OnBotonSimular()`	Pone en modo simular, dirección adelante y llama a la función *ActivarTimer()*.
`OnBotonPausa()`	Este botón está activo cuando el modo es simular y sirve para detener la animación. Pone modo en pausa, dirección parar y llama a la función *ParарTimer()*.

Comandos del Ratón

`OnLButtonDown()`	Cuando se pulsa el botón izquierdo del ratón, si el modo es crear fija el primer punto del rectángulo que contendrá un obstáculo. Si el modo es editar llama a la función *HitTest()* para comprobar si puso sobre un obstáculo y lo enmarca.
`OnMouseMove()`	Indica en la barra de estado la posición del ratón. Si el modo es crear muestra un recuadro que indica el tamaño del obstáculo. Pone el tamaño del obstáculo en la barra de estado.

`OnLButtonUp()`	Si el modo es crear y tiene el recuadro contenido a un obstáculo, entonces añade un nuevo obstáculo en la lista que guarda el objeto documento.
`OnSetCursor()`	Cambia el tipo de cursor dependiendo en que modo se encuentre.

Comandos del Menú

Muestra las cajas de Diálogo Tiempo, Desplazameinto, Desplazamiento aleatorio, Configuración de sensores y el de Comunicación Serial. Si el botón OK es presionado se actualizan los valores para la simulación.

`OnControlRetardo()`

`OnControl Desplazamiento()`

`OnControlRandom()`

`OnControlSensores()`

`OnControlManual()`

- **Comunicación Serial**

Una de las capacidades del prototipo es que puede comunicarse con el computador para recibir comandos de navegación y enviar lectura de los sensores. Se utiliza la comunicación serial a 9600 bps a través de un cable que conecta al computador con el prototipo. Los comandos son enviados por medio de acción de botones, y se recibe datos que son mostrados a través de una Caja de Diálogo como se muestra en la figura 4.15.

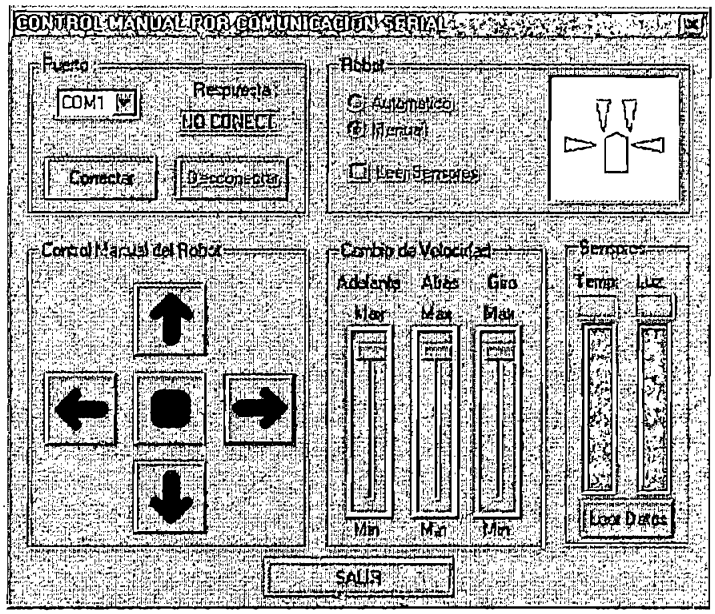


Figura 4.15 Diálogo para la Comunicación

La caja de diálogo se divide en áreas que agrupan ciertas características:

Puerto Permite escoger el puerto por donde se desea hacer la comunicación, además de botones para la conexión y desconexión del mismo.

Control Botones con flechas que indican la dirección a la que se desea mover el prototipo.

Velocidad El robot real posee la capacidad de cambiar la velocidad de los motores. Cuando se envía un comando para mover el prototipo se envía también un parámetro con el que se puede variar la velocidad de desplazamiento. La caja de diálogo permite escoger una velocidad hacia delante, otra hacia atrás y otra para los giros.

Sensores Son indicadores que muestran el nivel de temperatura y de luz de sensores incorporados en el robot real que sirven para medir parámetros del ambiente.

Robot Existe un control manual, donde el usuario tiene un total control de la navegación, y un control automático donde el usuario puede controlarlo pero el prototipo reacciona si encuentra un obstáculo. También puede visualizarse en la pantalla qué sensores han sido activados.

4.2 RESULTADOS

4.2.1 Interfaz de Usuario

Se trata de dar al usuario facilidades en el uso de herramientas, por este motivo se utilizan una barra con botones gráficos que muestran, si se pasa con el ratón sobre ellos, la función que realizan. Además se muestra en la parte inferior (Barra de Estado) una mayor explicación de la función de los botones.

La Barra de Diálogo (a la izquierda de la ventana) es una ayuda visual que muestra la posición del móvil en las coordenadas X e Y, relativas al área de trabajo, y la dirección del frente que está dada en grados.

Otras ayudas son los datos de posición del cursor, el tamaño del último obstáculo señalado y el tamaño de la ventana.

Para fijar ciertos valores como tiempo, desplazamiento, modificación de los sensores se utilizan cuadros de Diálogo.

Todas las herramientas mostradas son sencillas de utilizar si se conoce el funcionamiento de ventanas en un ambiente Windows.

4.2.2 Editor de Ambiente

Las herramientas para crear obstáculos son básicas: rectángulos y elipses, pero con la combinación de los mismos se pueden recrear ambientes relativamente complejos. Además se cuenta con la capacidad de guardar el ambiente de trabajo creado, con lo que se evita tener que reconstruirlo nuevamente.

Existe la capacidad de modificar el tamaño y la localización de los obstáculos a través de un recuadro llamado *tracker* que consta de 8 puntos que se encuentran en las esquinas y los puntos medios de los lados permitiendo una fácil modificación de un obstáculo. Para cambiar de posición un obstáculo solo necesita arrastrarlo al lugar deseado.

Los límites del área de trabajo son considerados obstáculos, por lo que no es necesario crear paredes que detengan al móvil cuando éste quiera sobrepasar los límites de la ventana.

4.2.3 Animación

Gracias a la capacidad de compilación que tiene Visual C++, se puede tener rutinas relativamente complejas a un costo muy pequeño en tiempo. A pesar de la forma como se dibuja el móvil y los sensores cada vez que realiza un movimiento, el programa lo muestra de una forma fluida y rápida, dependiendo también de las características del computador.

Para la animación se considera al móvil y a los sensores como puntos referidos a un punto que es el eje de giro del robot (Eje de Referencia). Para desplazar al móvil por la pantalla se le considera como un objeto puntual al que se le da una posición y dirección. El objeto móvil se encarga de hacer el cálculo de la nueva posición de cada uno de los puntos referidas al eje,

estos datos se guardan en una matriz que es la fuente de datos para dibujar al móvil y los sensores con diferentes colores.

Para el comando en modo manual, cada uno de los puntos que conforman el móvil es sensible a los obstáculos, es decir, si encuentra un obstáculo choca y se detiene.

4.2.4 Comunicación con el Robot

Además de los comandos básicos que son los de navegación, se muestran los datos de los sensores de temperatura y de luz en forma numérica y también en forma gráfica.

Existen controles más avanzados que son barras deslizantes que tienen la capacidad de fijar la velocidad con que se desplaza el prototipo.

Es interesante poder observar el comportamiento de los sensores de ultrasonido reflejándose la respuesta de ellos tanto en el robot real como en la consola de comunicación.

4.3 EJEMPLOS

Los siguientes ejemplos muestran la capacidad en la realización de ambientes de trabajo donde las figuras básicas (Rectángulos y Elipses) pueden representar paredes, puertas, maceteros, etc (Ejemplo 1), que pueden simular una oficina o laboratorio. En el Ejemplo 2, con la ayuda de sólo elipses se puede configurar un ambiente de trabajo relativamente complejo.

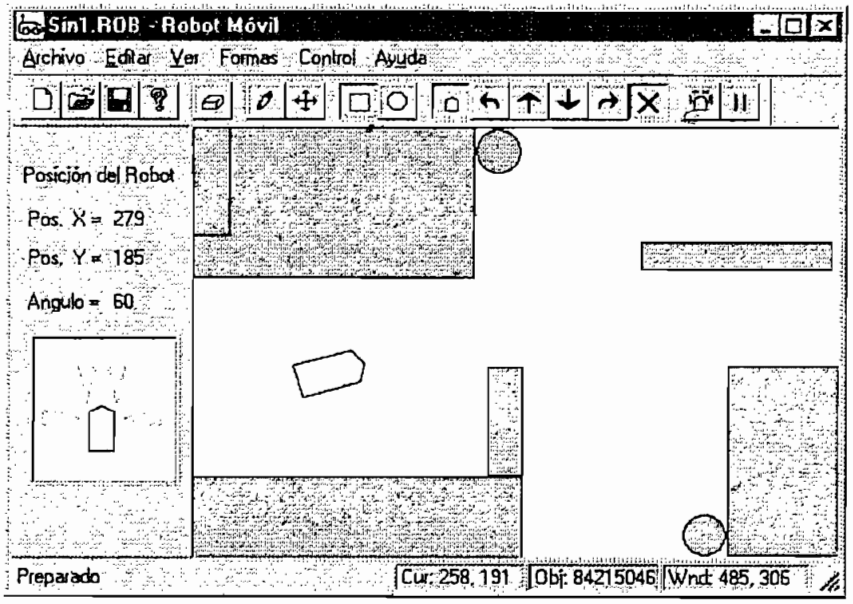


Figura 4.16 Ejemplo 1

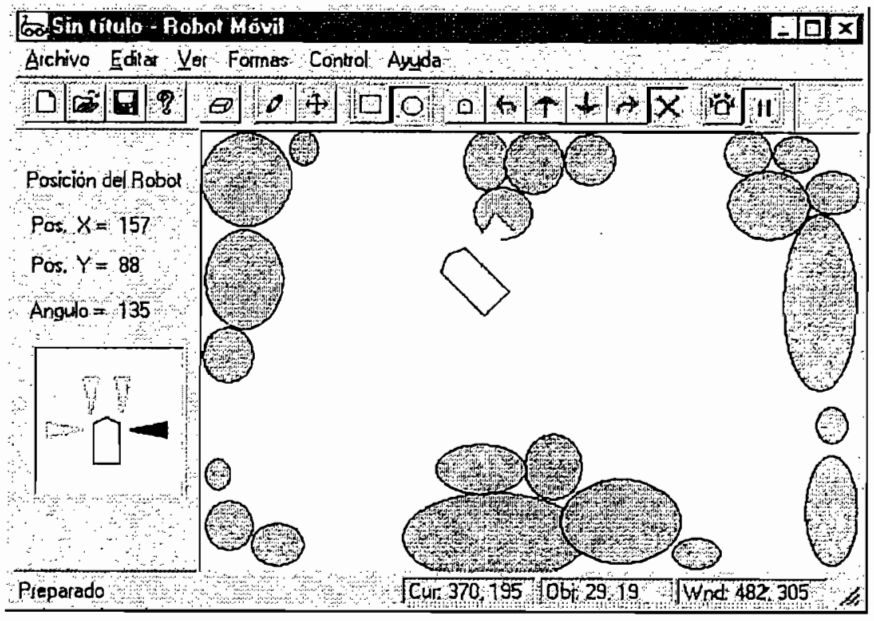


Figura 4.17 Ejemplo 2

Capítulo 5

ENSAMBLAJE DEL PROTOTIPO

En este capítulo se describe el ensamblaje de un prototipo de robot móvil de tracción diferencial. El prototipo utiliza elementos de bajo costo a diferencia de las plataformas móviles comerciales cuyos costos suelen ser prohibitivos.

A la plataforma móvil se la denominó ***Pionero1***, ya que constituye el primer trabajo en el campo de robots móviles desarrollado en la Escuela Politécnica Nacional. El robot será descrito como dos partes separadas inicialmente, las cuales fueron acopladas para conseguir un funcionamiento adecuado, estas partes son: el sistema mecánico y el sistema eléctrico. El sistema mecánico se lo adaptó de un juguete, del cual se aprovechó parte de la carrocería, el sistema de tracción diferencial y los motores de DC. El sistema eléctrico está constituido por un sistema microprocesado y por el sistema de sensores, para navegación y monitoreo. El software de navegación y comunicación, sin el cuál la plataforma no podría realizar ninguna acción, se describirá con detalle en el capítulo siguiente.

5.1 SISTEMA MECANICO

El sistema mecánico con el que originalmente se pensó realizar la plataforma estaba basado en dos motores de pasos y el sistema de tracción adecuado, pero debido a la dificultad de conseguir en el mercado local llantas a escala para la tracción, se optó por utilizar un juguete que se adapte en lo posible a los requerimientos de la plataforma móvil a ensamblarse. El único vehículo que se adaptaba a estos requerimientos fue una pala mecánica de juguete, similar a la utilizada como maquinaria pesada de construcción. Las partes utilizadas son: parte de la carrocería, el reductor formado por engranajes mostrados en el Anexo A, las llantas y dos motores de DC de 6 V.

En la plataforma se adaptaron interruptores de fin de carrera en los cuatro lados de la misma, con el objeto de proteger los motores y el sistema de engranajes de los reductores en caso de choque.

5.1.1 Cálculo de la Relación de Reducción del Sistema de Engranajes

El diagrama de bloques del sistema de transmisión se muestra en la figura 5.1.

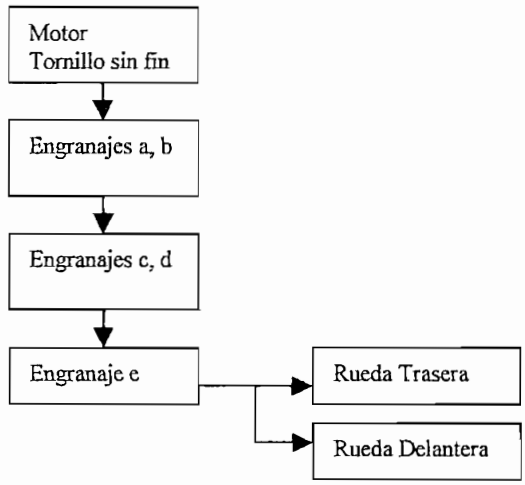


Figura 5.1 Diagrama de bloques de la transmisión

Para el cálculo se utilizarán las relaciones para transmisión por tornillo sin fin y transmisión doble, mostradas en las figuras 5.2 y 5.3 respectivamente.
Tornillo sin fin:

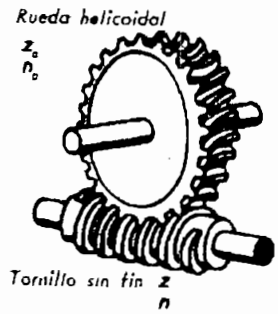


Figura 5.2 Transmisión por tornillo sin fin

Para este caso se tiene la siguiente relación: $n * z = n_a * z_a$ [6]

Donde:

n = revoluciones del tornillo sin fin

z = número de entradas del tornillo sin fin

n_a = revoluciones de la rueda helicoidal

z_a = número de dientes de la helicoidal

v = velocidad nominal del motor

Sabiendo que: z = 1, n = *v* y z_a = 28, entonces: n_a = *v* / 28

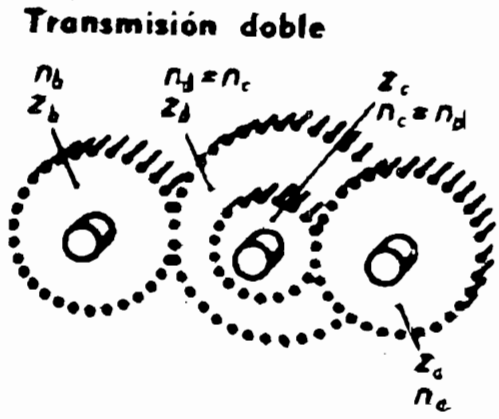


Figura 5.3 Transmisión doble

En este caso se tiene:

$$n_c = n_d = \frac{z_b * n_b}{z_c}$$

$$n_e = \frac{z_d * n_d}{z_e}$$

Donde:

n_c = revoluciones de la rueda C

z_c = número de dientes de la rueda

n_e = revoluciones del engranaje de la llanta

z_e = número de dientes del engranaje de la llanta

De donde se obtiene:

$$n_e = n_a \frac{z_b * z_d}{z_c * z_e}$$

Sabiendo que: $n_a = v/28$, $z_b = 8$, $z_c = 26$, $z_d = 10$, $z_e = 40$, se tiene:

$$n_e = v / 364$$

Sabiendo que n_e es la velocidad del engranaje en el que se encuentra acoplada la rueda trasera.

5.2 SISTEMA ELECTRICO

Con el objeto de darle autonomía a la plataforma móvil se montó en ella un sistema microprocesado, el cual recibe los datos de los sensores, y sobre la base de esa información se comanda a los motores de DC para que navegue con facilidad en su ambiente de trabajo. Los comandos también pueden ser enviados desde un computador central para realizar un control manual del robot. El sistema microprocesado esta constituido por la tarjeta de desarrollo para microcontroladores de la familia MCS-51, denominada MCPD51 [5]. En esta tarjeta, aprovechando su capacidad de direccionar 64K localidades de memoria externa a través del bus de direcciones de 16 bits, se ha incluido toda la circuitería para direccionar independientemente hasta 8 dispositivos de entrada y 8 dispositivos de salida, los dispositivos utilizados se muestran en la tabla 5.1.

Tabla 5.1 Mapa de Memoria del Microcontrolador

Direcciones	Dispositivo de Entrada	Dispositivo de Salida
0000H – 1FFFH	Pórticos SW0-SW7	Pórticos OUT0-OUT7
2000H – 3FFFH	Pórticos EXT-INT0 a EXT-INT7	Pórticos OUT8 – OUT15
4000H – 5FFFH	Sensores de Ultrasonido	Conversor D/A
6000H – 7FFFH	Conversor A/D	DISPONIBLE
8000H – 9FFFH	Memoria Eprom	----------
A000H – BFFFH	DISPONIBLE	DISPONIBLE
C000H – DFFFH	DISPONIBLE	DISPONIBLE
E000H – FFFFH	DISPONIBLE	DISPONIBLE

Los pórticos mencionados en la tabla, se encuentran disponibles en los conectores H2, H6 y H5 de la Tarjeta MCPD51, mayor información sobre los conectores se la encuentra en la referencia [5].

Para obtener la información del medio ambiente se utilizaron sensores de ultrasonido (ver anexo D), cuya información fue acondicionada a los requerimientos del software de navegación. Además se colocaron sensores adicionales de nivel de luz y temperatura, que se utilizarán para realizar monitoreo de estos parámetros en el medio ambiente de trabajo del robot.

5.2.1 Sensores de Ultrasonido

Los sensores de ultrasonido, son hoy en día los más utilizados en el campo de robots móviles, debido a su costo accesible y a la fácil implementación del acondicionador. Estos sensores pueden ser utilizados en varias áreas como: modelación del medio ambiente, detección de obstáculos, medición de distancias, detección de movimiento. Los sensores encontrados ampliamente en robots móviles, son los desarrollados por Polaroid para cámaras automáticas. Polaroid ofrece el transductor de ultrasonido y la tarjeta de interface (ver figura 5.4) a un costo cercano a los 100 dolares. El lector interesado en este tipo de sensores puede encontrar mayor información en [1].

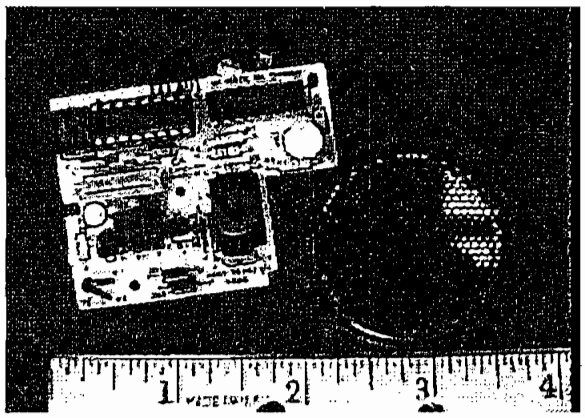


Figura 5.4 Transductor ultrasónico y tarjeta de inteface polaroid

Los sensores utilizados son de menor costo a los antes mencionados y están constituidos por un par transmisor - receptor (figura 5.5), que tienen las siguientes características:

Frecuencia: 40kHz ± 1.0kHz

Capacitancia: 2000 pf ± 20%

Transmisor:

Ancho de banda: 5.0 kHz/100dB

Nivel de presión: 112dB/40 ± 1kHz (0dB+0.0002 μbar)/10V SN/30cm/min

Receptor:

Ancho de banda: 5.0kHz @ -75dB

Sensibilidad min: 67dB/40 kHz ± 1.0kHz (0dB vs 1V μbar) R=3.9kΩ

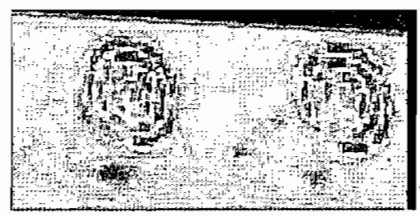


Figura 5.5 Sensores de ultrasonido, par transmisor - receptor

La información de los sensores proporcionada por el fabricante es muy escasa, por lo que fue necesario experimentar con ellos para obtener información de alcance, ángulo de visión y la separación óptima entre el transmisor y el receptor. Los resultados se los puede ver en la tabla 5.2 y en forma gráfica en la figura 5.6.

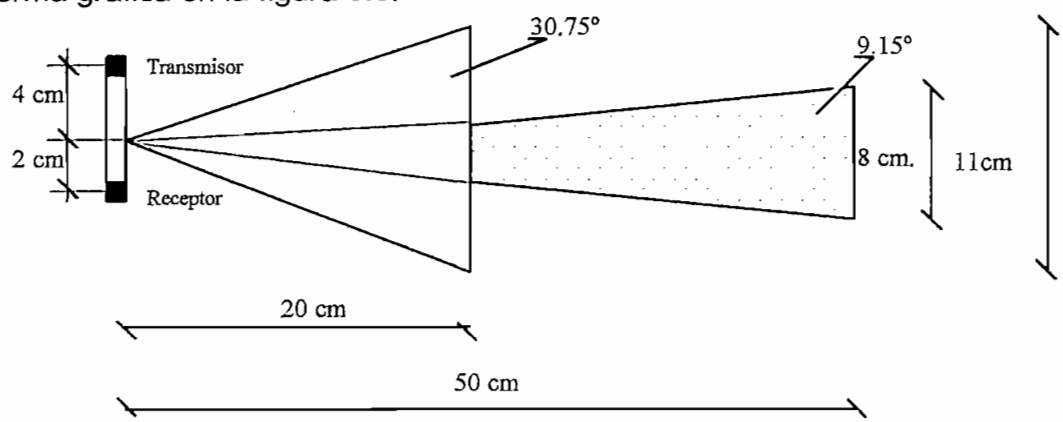


Figura 5.6 Angulo de visión de los sensores

Tabla 5.2 Resultados experimentales

Separación entre sensores	Amplitud del receptor, obstáculo a 20 cm	Amplitud del receptor, obstaculo a 50 cm
4 cm	800 mV	300 mV
5 cm	800 mV	300 mV
6 cm	900 mV	340 mV
9 cm	800 mV	280 mV
11 cm	600 mV	260 mV

De lo anterior se ve que la mayor sensibilidad se obtiene, separando al transmisor del receptor una distancia de 6 cm entre sus centros. El ángulo de visión varia con la distancia, así en la figura se indica el ángulo para dos distancias. De acuerdo a estos experimentos se optó por colocar dos pares transmisor - receptor de estos sensores en la parte delantera del robot, con el objetivo de darle una visión de todo el frente del robot y evitar de esta manera posibles choques. Por la forma del montaje de los sensores, se los puede colocar a diferentes ángulos respecto al eje de las ruedas, con el objetivo de experimentar en el prototipo diferentes configuraciones hasta encontrar la óptima. La ubicación de los sensores en el robot móvil se muestra en la figura 5.7.

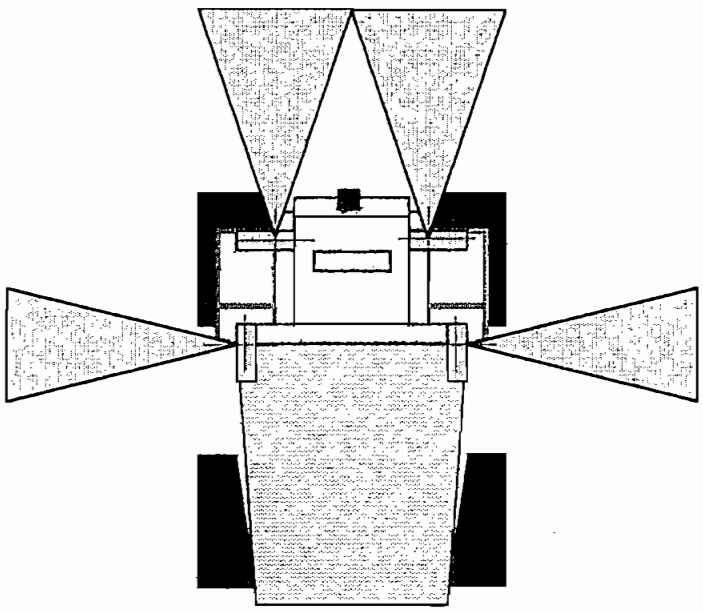


Figura 5.7 Ubicación de los sensores en el robot móvil

Para el funcionamiento de los sensores se implantó un oscilador de 40 kHz para el transmisor, para ello se experimentó con algunos diseños que no fueron satisfactorios ya que al tener los sensores un ancho de banda muy pequeño, una leve variación la frecuencia producía que ellos ya no trabajen adecuadamente y se haga necesaria una recalibración del oscilador. Los osciladores experimentados, los mismos que no se recomienda usar con sensores de estas características son los siguientes:

Oscilador de relajación con ujt, put

Oscilador aestable con 555

Oscilador de precisión con LM356 (decodificador de tono)

El problema encontrado con los osciladores antes mencionados fue la sensibilidad de los mismos a cambios de temperatura del medio. Por lo expuesto, la alternativa adecuada para estos sensores, por la estabilidad que presentan, fue la utilización de un cristal de cuarzo para generar la frecuencia de trabajo.

El oscilador se implementó utilizando un cristal de 5.18 MHz, y las compuertas (4011), la frecuencia generada por el oscilador se la introdujo en un divisor de frecuencia de 7 estados (4024) con el cual dividiéndola para 128, se obtuvo la frecuencia de 40.46875 KHz (ver figura 5.8), que está dentro del rango de frecuencia de trabajo de los sensores.

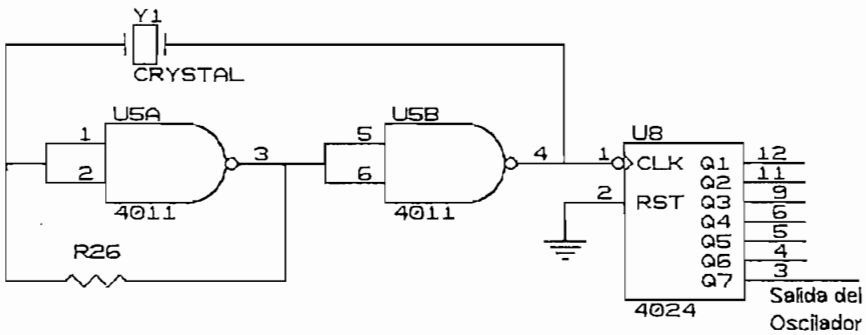


Figura 5.8 Oscilador de 40.46875 KHz.

El oscilador descrito anteriormente se colocó a través de un transistor para amplificar la señal y asegurar que el voltaje del oscilador se encuentre entre 0 y 5 voltios como se muestra en la figura 5.9.

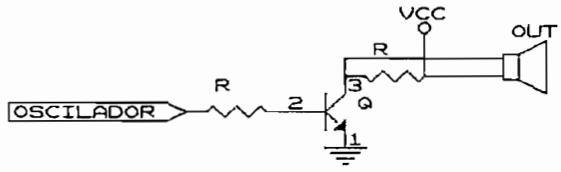


Figura 5.9 Amplificador de salida de ultrasonido

La señal del receptor se la acondicionó con amplificador de dos etapas. La primera, con una ganancia de 10, y la segunda con ganancia variable, dada por un potenciómetro (figura 5.10), el cual permite calibrar la sensibilidad de los sensores.

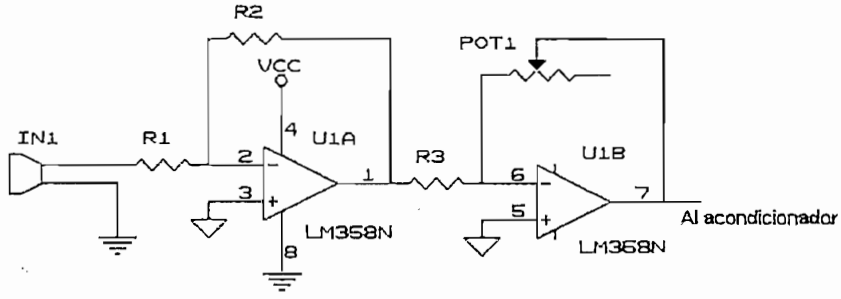


Figura 5.10 Amplificador de dos etapas

La señal amplificada se la acondiciona para hacerla compatible con la lógica TTL [8], es decir, que se ponga en 1 lógico si los sensores detectan y 0 lógico si no lo hacen, además se coloco indicadores del estado de los sensores, como se aprecia en la figura 5.11.

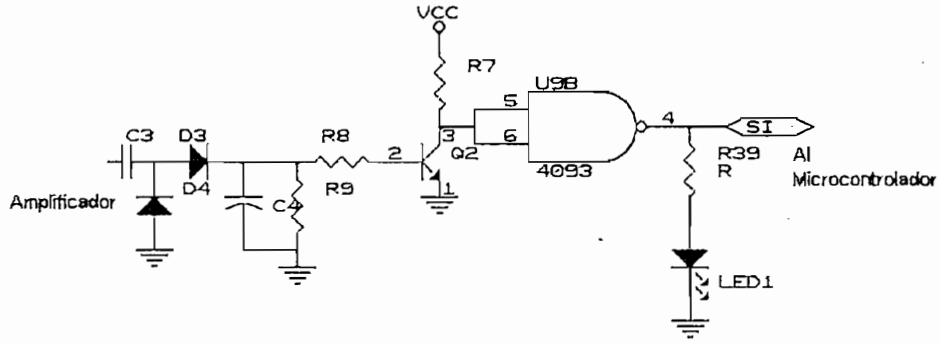


Figura 5.11 Acondicionador sensores laterales

La configuración anterior se utilizó para los sensores laterales que dan señales independientes una de otra, mientras que los dos sensores delanteros dan una sola señal al micro, por lo que se los hizo un OR lógico utilizando una compuerta NAND, y de la misma manera que los sensores laterales se colocó un indicador como se aprecia en la figura 5.12.

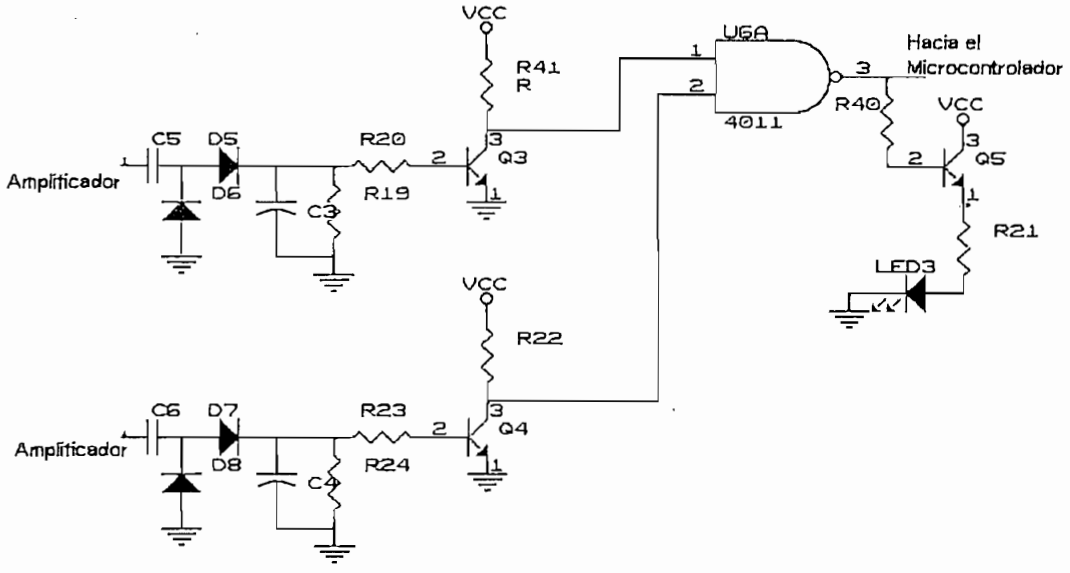


Figura 5.12 Acondicionador sensores delanteros

5.2.2 Sensores de Temperatura y Nivel de Luz

Además de los sensores de ultrasonido utilizados para navegación, en el robot móvil se montaron dos sensores para monitoreo del medio ambiente, un sensor de temperatura, implementado con el LM355 y un sensor de nivel de luz, dichos sensores ingresan al microcontrolador a través del conversor A/D ADC0804, que se encuentra disponible en la tarjeta MCPD51. Como la tarjeta cuenta con una sola entrada, fue necesario usar un MUX 4052, el mismo que cuenta con dos multiplexers de 4 a 1, simultáneos, de los cuales se utilizan únicamente dos entradas, quedando libres dos del primer multiplexer y las 4 del segundo, las entradas y salidas libres de este elemento se encuentran disponibles en el conector JP4 de la placa de acondicionamiento de sensores (Anexo C), para que en un futuro sea posible

añadir más sensores a la plataforma móvil a través del conversor A/D. Con el circuito implementado (figura 5.13), la temperatura varía linealmente con el voltaje, es decir si la temperatura varia de 0°C a 50°C, la salida del circuito variará de 0V a 5V, el sensor de nivel de luz esta constituido por un divisor de tensión con una fotoresistencia, cuya salida variará de 0V a 5V, de acuerdo al nivel de Luz.

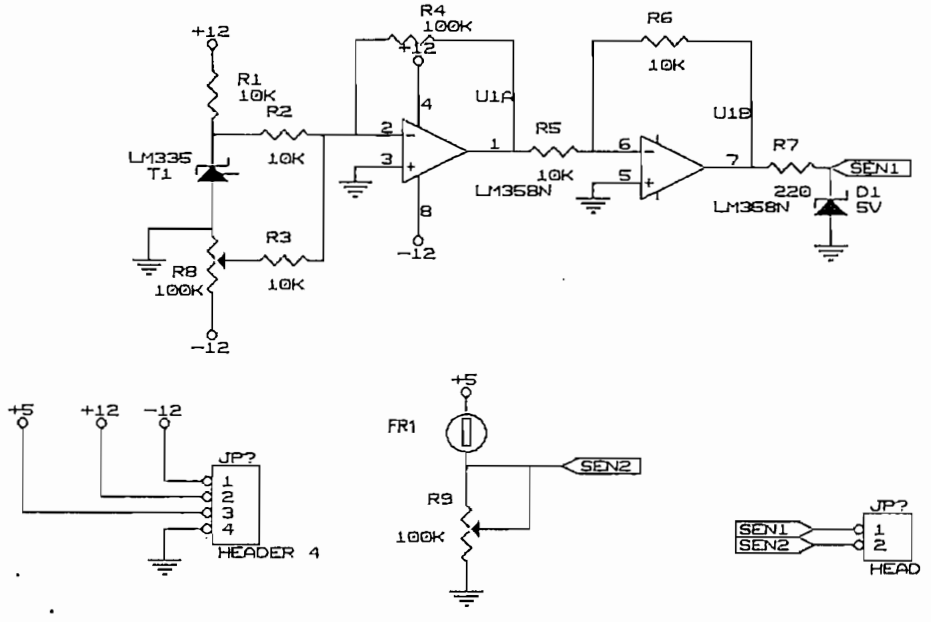


Figura 5.13 Sensores de Temperatura y Nivel de Luz.

5.2.3 Control de Motores

En el robot móvil desarrollado, era necesario realizar el control de velocidad e inversión de giro de los motores de DC. Para el control de velocidad se utilizó la configuración de un chooper mostrado en la figura 5.14, cuyo control se lo realiza con un Modulador por Ancho de Pulso (PWM), generado por el microcontrolador, y que sale a través pin 0 del puerto 1 (P1.0). Se optó por la configuración del chooper indicado, ya que la misma señal de control PWM se la utiliza para los dos motores de DC que deben funcionar independientemente uno del otro. La inversión de giro se la realizó con relés

cuyos contactos alimentan a los motores. La conexión y desconexión de los mismos es controlada por el micro a través del pin 2 y 3 del puerto 1 (P1.2; P1.3), como se muestra en la tabla 5.3. Debido a que el cambio de giro del motor se realiza con circulación de corriente, fue necesario colocar una configuración de diodos de recuperación rápida como se muestra en la figura 5.15, para dar un camino alternativo a la corriente durante la conmutación. Ya que la bobina de los relés utilizados se excita con 24 voltios, se utilizo optoacopladores para cambiar el nivel de referencia de las bobinas (figura 5.16), ya que para la señal de control se utiliza la fuente del micro que es de 5V. En el anexo C se tiene información detallada de los elementos de la placa para control de los motores.

Tabla 5.3 Asignación de pines del puerto 1 para control de motores

Pin	Acción
P1.0	PWM
P1.2	Señal de control M1
P1.3	Señal de control M2

Tabla 5.4 Acción de P1.2 y P1.3

ESTADO	1 LÓGICO	0 LÓGICO
P1.2 (SC1)	Alimenta al motor 1, con Voltaje negativo	Alimenta al motor 1, con Voltaje positivo
P1.3 (SC2)	Alimenta al motor 2, con Voltaje negativo	Alimenta al motor 2, con Voltaje positivo

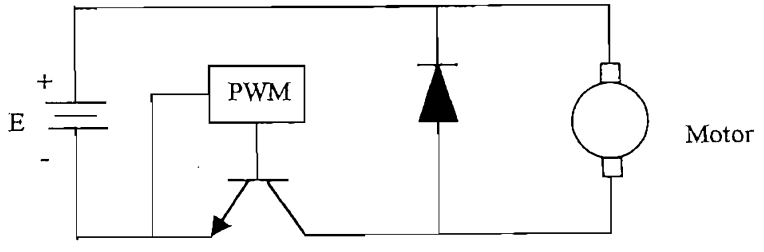


Figura 5.14 Control de velocidad de un motor DC

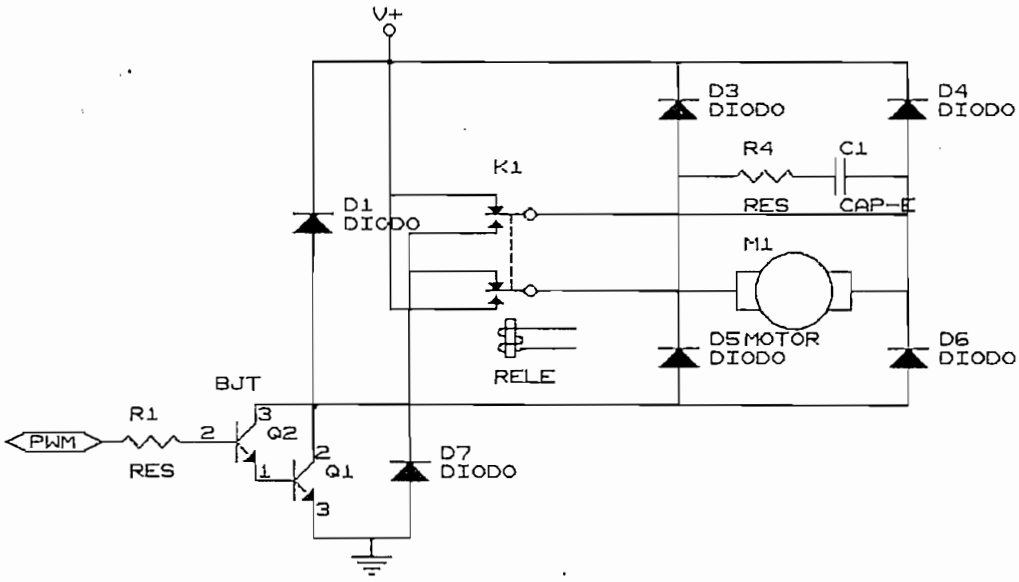


Figura 5.15 Control de velocidad de los motores de DC.

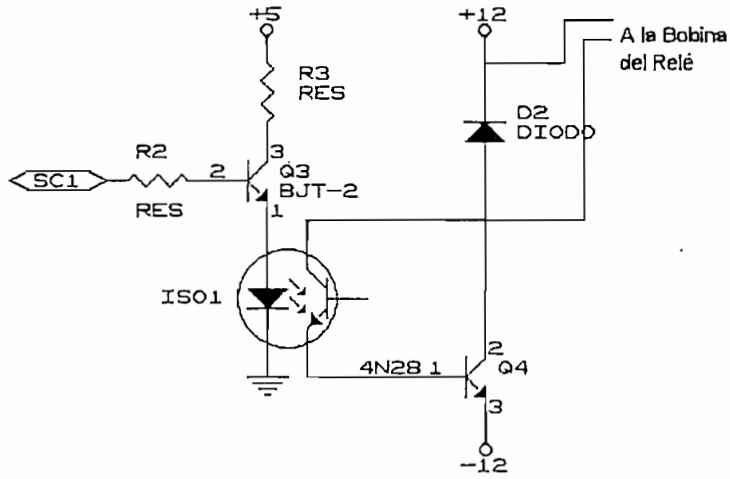


Figura 5.16 Circuito de Control para inversión de giro de los motores de DC.

5.2.4 Circuito de Vigilancia para el Microcontrolador (WDT)

La función del circuito de vigilancia, más conocido como Watch Dog Timer (WDT), es reinicializar el programa implementado en el microcontrolador, cuando este por cualquier circunstancia interna o externa deja de responder

adecuadamente. Este circuito informa al usuario el estado de reinicialización a través de un indicador luminoso colocado en la tarjeta de sensores.

Para este circuito se tomo como referencia un monoestable multivibrador [7] con el CI LM555, en el cual se utiliza la señal PWM generada por el microcontrolador como señal de control para el WDT, el circuito implementado y su diseño se muestran a continuación.

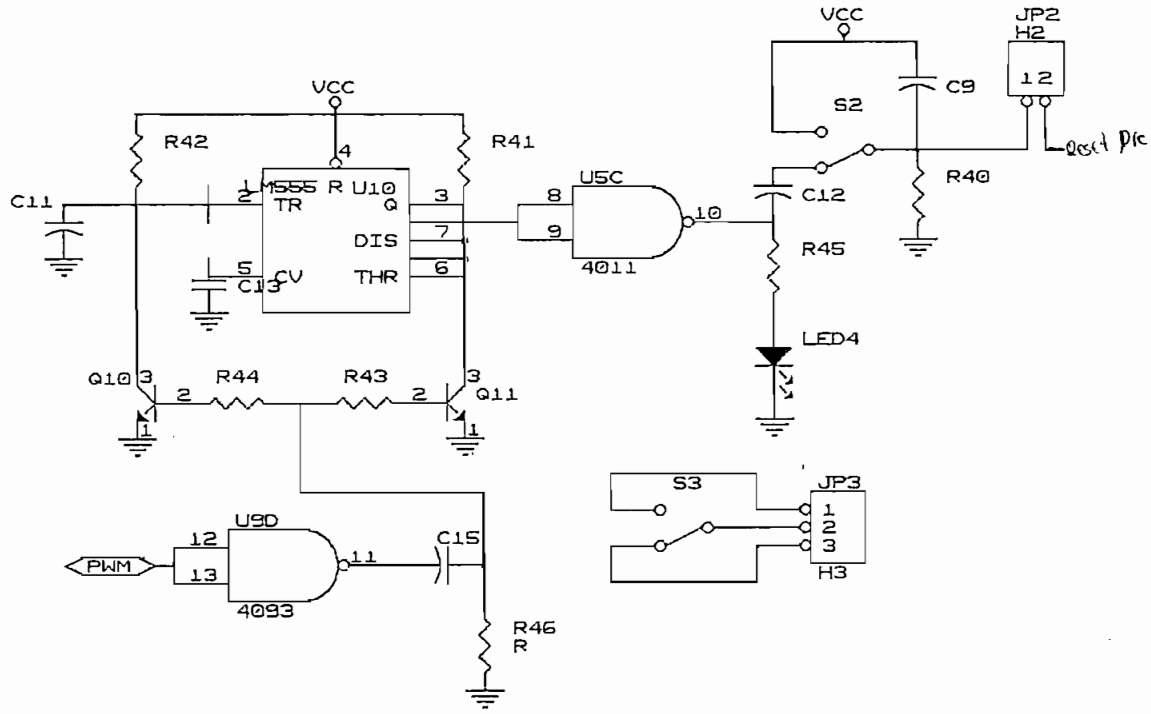


Figura 5.17 Circuito WDT

Para el diseño, se sabe que la frecuencia del PWM es 2KHz, por lo tanto el periodo es T_{PWM} = 0.5 ms. Es necesario dar un tiempo mayor al circuito WDT para que genere el RESET, entonces se asume que: T_{WDT} = 1ms.

Se conoce que para el LM555, el periodo es:

T = 1.1 RC

Entonces: R = T / 1.1 C

Sea C = 0.1 μF, entonces

R = 1ms / 1.1*0.1μF = 9000 Ω

Por lo que se coloca una resistencia de 10K

El usuario puede hacer que el circuito de vigilancia este conectado al reset del microcontrolador, a través de JP2, o simplemente sea un indicador, utilizando para este efecto el selector JP3, que se encuentra conectado al conector JP4 de la tarjeta MCPD51.

5.3 LISTA DE ELEMENTOS Y PRECIOS

En la tabla 5.5 se muestra el listado detallado de los elementos utilizados para ensamblar el prototipo con sus respectivos precios, para los elementos electrónicos los precios se tomaron del catálogo JAMECO de Agosto, 1998. Al final de la sección se muestra el valor total del prototipo en dólares.

Tabla 5.5 Lista de Elementos y Precios

DESCRIPCION	CANTIDAD	VALOR UNITARIO	VALOR TOTAL
Total Tarjeta MCPD51			**60**
Tarjeta MCPD51	1	35	35
Microcontrolador 8751H	1	25	25
Total Tarjeta Sensores1			**64,1723**
Tarjeta Sensores 1	1	9,2258	9,2258
Resistencias 1/4 W	44	0,0258	1,1352
Potenciómetros 2K	4	0,79	3,16
Condensadores electrolíticos	7	0,1814	1,2698
Condensadores cerámicos	5	0,1661	0,8305
Diodos 1N4007	8	0,0246	0,1968
Transistores NPN	10	0,0246	0,246
Leds	4	0,0595	0,238
Cristal 5.18 MHz	1	1,05	1,05
Operacionales LM358N	4	0,39	1,56
CI 4052	1	0,45	0,45
CI 4011	1	0,25	0,25
CI 4024	1	0,35	0,35
CI 4093	1	0,29	0,29
Timer LM555	1	0,29	0,29
Pulsador	1	0,45	0,45
Slide Switches	1	0,35	0,35
Heders	68	0,0264	1,7952

Sensores de Ultrasonido	4	7,95	31,8
Conectores 2 contactos	8	0,25	2
Conectores 4 contactos	2	0,35	0,7
Conectores 6 contactos	1	0,39	0,39
Cable apantallado	2	0,125	0,25
Zócalos 8 pines	5	0,375	1,875
Zócalos 14 pines	3	0,99	2,97
Zócalos 16 pines	1	1,05	1,05
Total Plataforma Móvil			**130,54**
Plataforma Móvil	1	68,2	68,2
Placa de acrílico	1	1	1
Switches 3 posiciones	2	1,25	2,5
Microswitches	4	0,4	1,6
Batería 6 V	1	25	25
Batería 9,5 V	1	30	30
Terminales	8	0,08	0,64
Jacks	2	0,05	0,1
Cable serial 10 mts	1	1,5	1,5
Total Tarjeta de Control de Motores			**16,0618**
Tarjeta de control de motores	1	3,7742	3,7742
Relés	2	2,25	4,5
Transistores TIP 110	4	0,227	0,908
Transistores NPN	4	0,0246	0,0984
Diodos 1N4148	14	0,05	0,7
Resistencias 1/4 W	8	0,0258	0,2064
Heders	8	0,0264	0,2112
Conectores de fuente	3	0,3	0,9
Condensadores cerámicos	2	0,1661	0,3322
Condensadores electrolíticos	1	0,1814	0,1814
Optoacopladores	2	0,35	0,7
Conectores 4 contactos	2	0,35	0,7
Zócalos 6 pines	2	0,375	0,75
Zócalos 16 pines	2	1,05	2,1
Total Tarjeta de Fuente			**13,7968**
Tarjeta de fuente	1	2	2
Regulador LM7805	1	1,69	1,69
Conversor 5 12-12	1	7,95	7,95
Conector de Fuente	1	0,3	0,3
Disipador	1	0,49	0,49
Conectores 4 contactos	3	0,35	1,05
Heders	12	0,0264	0,3168
Total Tarjeta Sensores 2			**8,2886**
Tarjeta Sensores 2	1	2	2
Sensor de Temperatura LM335	1	1,19	1,19
Fotocelda	1	1,79	1,79

Potenciometros 100K	2	0,79	1,58
Resistencias 1/4 W	7	0,0258	0,1806
Diodo zener 5.1 V	1	0,0246	0,0246
Operacionales LM358N	1	0,39	0,39
Headers	6	0,0264	0,1584
Conectores 2 contactos	1	0,25	0,25
Conectores 4 contactos	1	0,35	0,35
Zocalos	1	0,375	0,375
Otros Gastos	1	**18**	**18**
COSTO TOTAL	DOLARES	**310,8595**	

5.4 RESULTADOS

El ensamblaje del prototipo de robot móvil (Pionero1), fue más allá de los objetivos planteados originalmente, es así que ahora se cuenta con un prototipo de plataforma móvil similar a plataformas comerciales, que a diferencia de la ensamblada cuestan mucho dinero.

Los resultados obtenidos se pueden resumir en los siguientes:

Se cuenta con un prototipo de plataforma móvil de tracción diferencial, que puede cumplir tareas de navegación simples tales como: marcha adelante, marcha atrás y movimientos de izquierda y derecha, además tiene la capacidad de evadir obstáculos que se encuentren al alcance de sus sensores, por lo que puede navegar libremente por su ambiente de trabajo.

Ya que para su sistema microprocesado se utilizó la tarjeta MCPD51, se puede utilizar el interfaz de comunicación serial que ella posee, por lo tanto el prototipo puede comunicarse con un computador central. Con dicha comunicación se puede realizar un control directo del prototipo, es decir se puede suspender su autonomía y realizar un control manual desde un PC. Esta comunicación se la realiza utilizando un cable serial que tiene una longitud de 10 metros.

Se montó en el prototipo sensores de temperatura y nivel de luz, con los que se puede conseguir información del ambiente de trabajo del robot móvil. Dichos sensores ingresan al sistema a través del conector JP4 de la tarjeta de acondicionamiento de sensores, quedando libres en el mismo conector dos entradas en las que se podrían colocar dos sensores más para realizar tareas de monitoreo del medio ambiente.

El prototipo de plataforma móvil utiliza para su alimentación dos baterías recargables, una de 9,5 V utilizada para el sistema de control y una de 6 V que alimenta a los motores. Ambas baterías pueden ser recargadas sin ser desmontadas de la plataforma, a través del los conectores R1 y R2 respectivamente, dichos conectores se encuentran montados en las partes laterales de la plataforma.

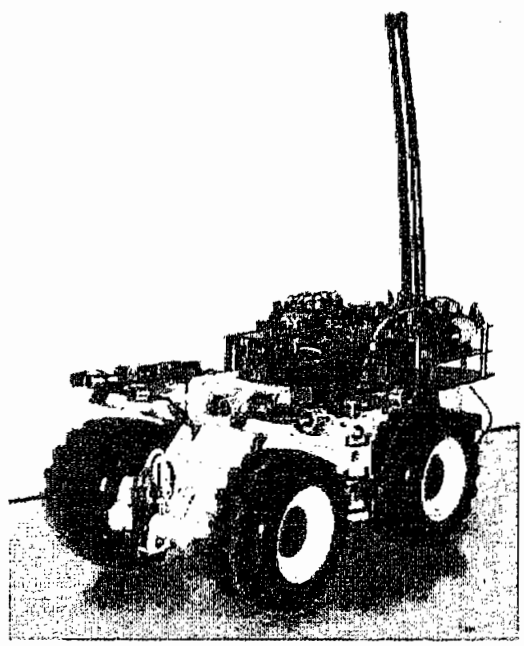


Figura 5.17 Robot Móvil *Pionero1*

Los detalles de los componentes de la plataforma se muestran en los anexos.

Capítulo 6

PROGRAMA DE CONTROL

El programa de control del robot móvil utiliza el lenguaje ensamblador (ASM51), para microcontroladores de la familia MCS-51. Con dicho programa de control, denominado *Sistema de Navegación*, el robot móvil tiene la inteligencia necesaria para evadir los obstáculos que se encuentren en su ambiente de trabajo y enviar datos de los sensores de monitoreo si son solicitados por el usuario.

El sistema de navegación es operado por el microcontrolador (modo autónomo) o por un operador desde un computador central (modo manual). En el modo autónomo el microcontrolador, en base a la información de los sensores de ultrasonido, controla el movimiento de los motores del robot móvil, la lógica utilizada para ello se describirá mas adelante. En el modo manual, el microcontrolador ejecuta las ordenes dadas por el usuario utilizando comunicación serial RS-232 a una velocidad de 9600 bps. El sistema de navegación está implementado con la arquitectura mostrada en la figura 6.1.

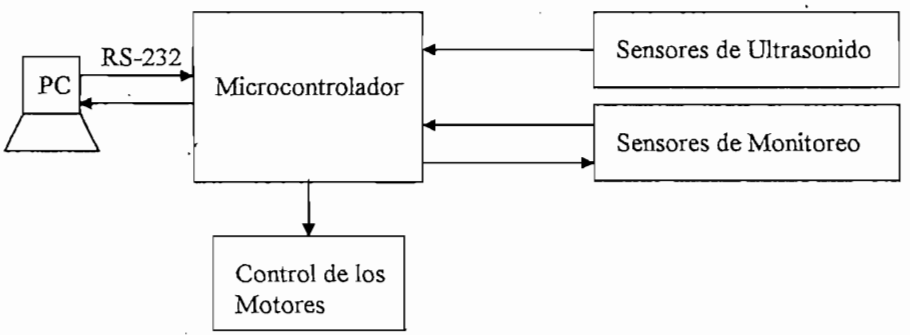


Figura 6.1 Arquitectura del Sistema de Navegación

6.1 DESCRIPCION DEL PROGRAMA

El programa ha sido desarrollado en forma modular. Los módulos que se encuentran en subrutinas dentro del programa del microcontrolador, se describen a continuación. Al final de la sección se mostraran los diagramas de flujo del programa, con lo cual el lector podrá observar como se relacionan los diferentes módulos.

6.1.1 Programa Principal

Aquí se definen las etiquetas y banderas utilizadas en el programa, se vectorizan las interrupciones utilizadas, se inicializan los registros de configuración de timers e interrupciones y se carga los valores para que el robot móvil inicie su movimiento hacia delante. Lo anotado anteriormente se realiza cada vez que se le aplique un reset al microcontrolador, luego de esto, el programa se coloca en un lazo infinito en el cual verifica si se encuentra en modo manual o autónomo. En el primer caso, el micro espera las órdenes del computador, caso contrario lee la información de los sensores delanteros, para verificar si existe o no un obstáculo enfrente, si existe, llama al módulo de evasión de obstáculos, y si no, vuelve al inicio del lazo.

6.1.2 Módulo de Evasión de Obstáculos

En este módulo se dan las directivas al módulo de control de motores para evitar chocar con los obstáculos. Esto lo hace consultando al módulo de lectura de sensores de ultrasonido, y basándose en esa información reacciona de la siguiente manera:

En el caso de tener camino libre a los dos lados del robot, es decir a la derecha y a la izquierda, el módulo dará la información para que el robot móvil gire a la derecha o izquierda en forma alternada, cada vez que el

sistema de navegación se encuentre en este estado, hasta que el sensor delantero le indique que el camino está libre y el robot avance hacia delante.

Cuando tiene camino libre a un solo lado, se dará la información al modulo de control para que el robot gire a ese lado, de igual manera hasta que el sensor delantero le indique que el camino está libre y el robot avance hacia adelante.

En el caso de no tener camino libre a ninguno de los dos lados, el robot avanzará hacia atrás hasta que pueda curvar a la derecha o a la izquierda y se encuentre en cualquiera de los estados antes mencionados.

6.1.3 Módulo de Control de Velocidad de los Motores

Aquí se genera una señal PWM, con la cual se realiza el control de velocidad de los motores de DC. El ancho de pulso de la señal PWM puede ser ingresada por el usuario desde el computador central, de esta manera se varia la velocidad del robot móvil desde su velocidad nominal hasta velocidades bajas.

6.1.4 Módulo de Lectura de Sensores

Este módulo lee la palabra en la cual se encuentra la información de los sensores laterales; activa o desactiva banderas que informan al sistema si los sensores detectaron o no obstáculos.

6.1.5 Módulo de Control de Giro de los Motores

Con la información proporcionada por el módulo de evasión de obstáculos, aquí se controla las bobinas de los relés que realizan el cambio de giro de los motores del robot móvil, y se carga el valor de velocidad que será utilizado por el módulo respectivo.

6.1.6 Módulo de Comunicación

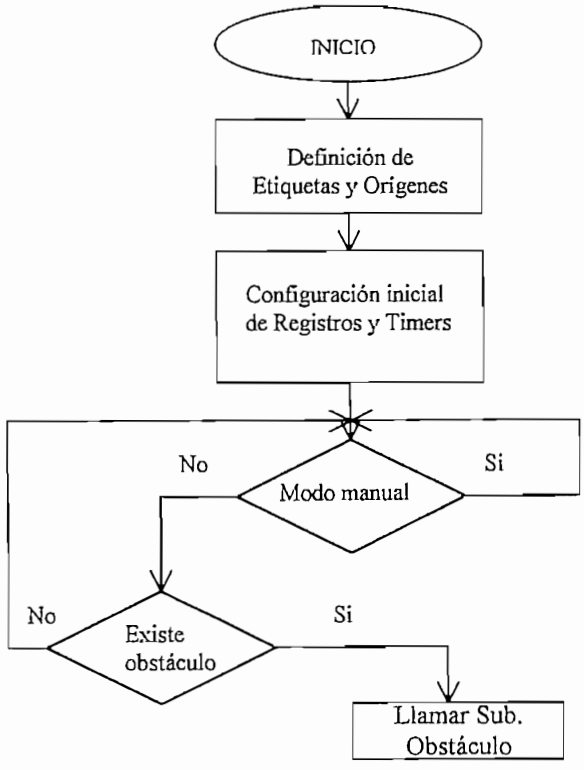
En este módulo se reciben, codifican y ejecutan las ordenes dadas por los usuarios desde el computador central, las mismas que deben cumplir con el protocolo de comunicación establecido.

6.1.7 Módulo de Confirmación de Comunicación

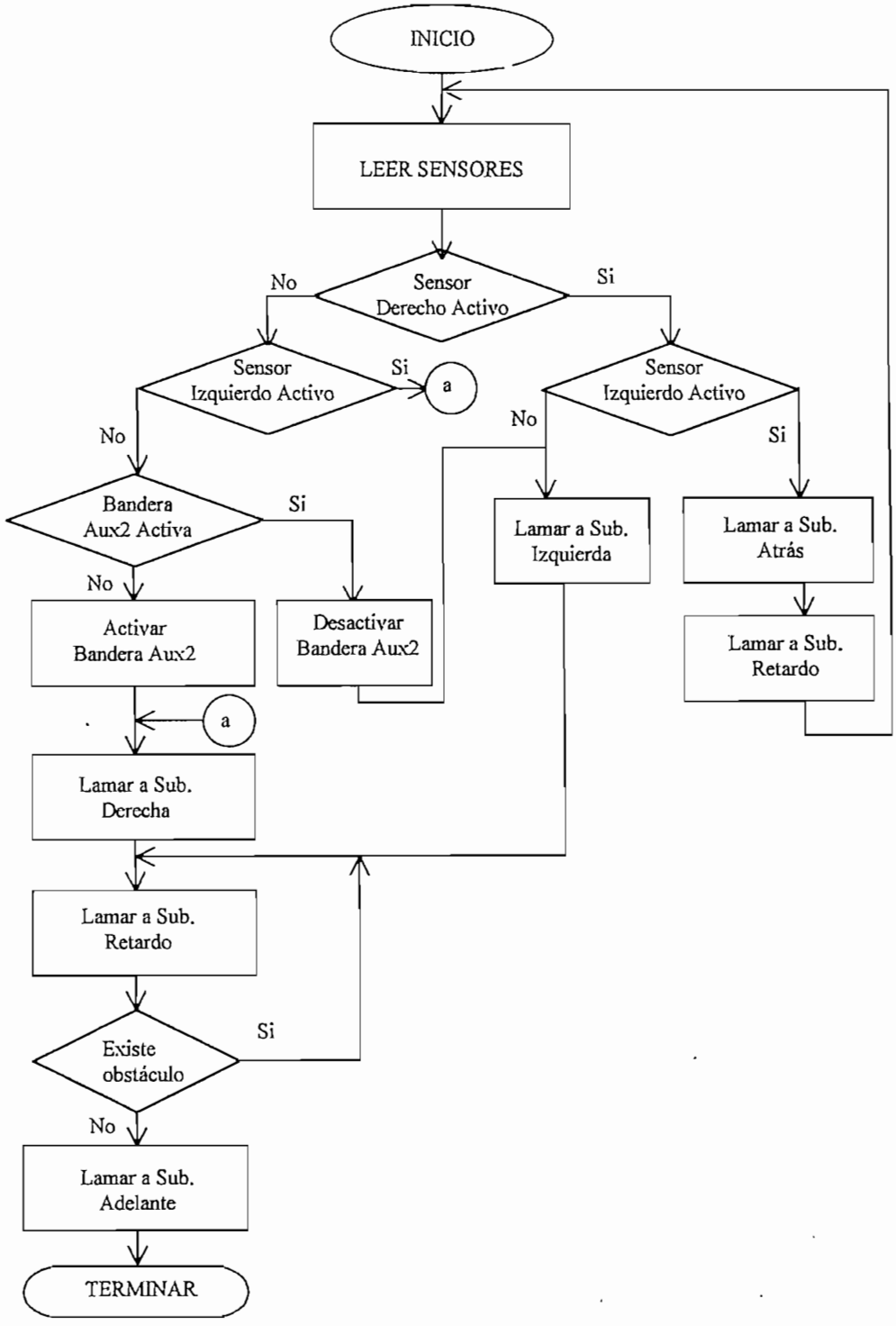
Este módulo esta asociado al anterior. En él se confirma la comunicación con el computador central y se envían la información de los sensores de monitoreo, cuando son requeridos.

6.1.8 Diagramas de Flujo del Programa de Control

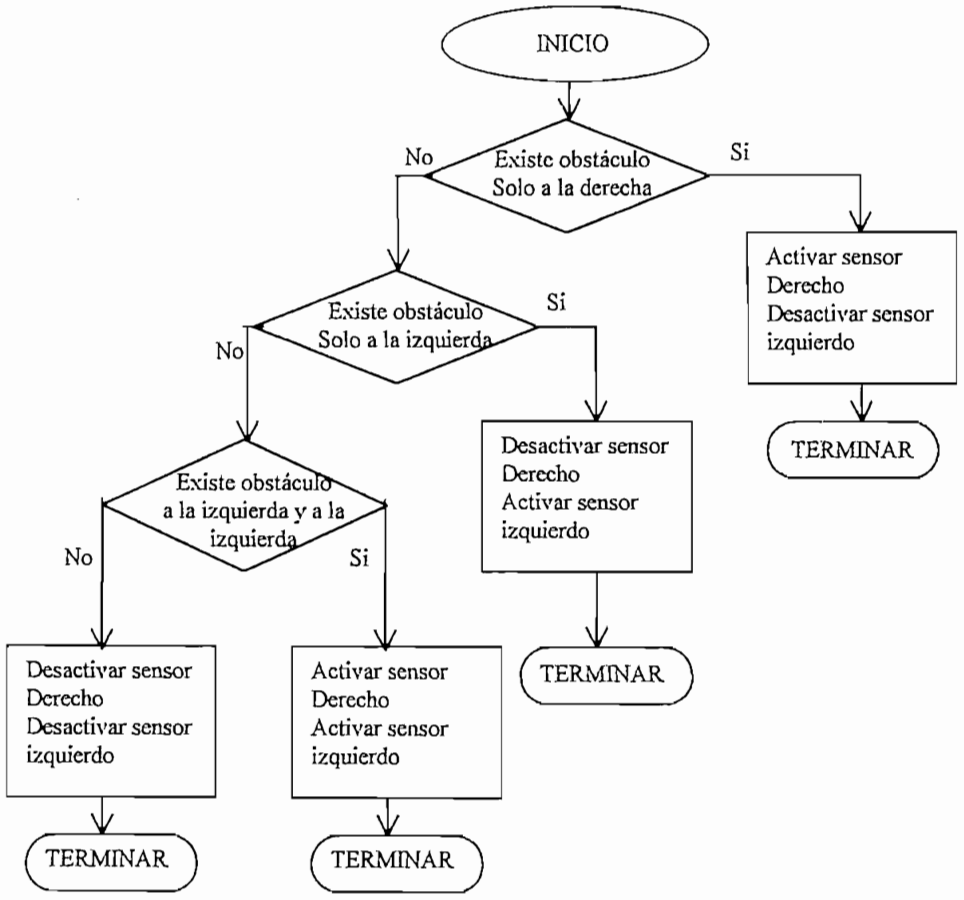
Programa Principal



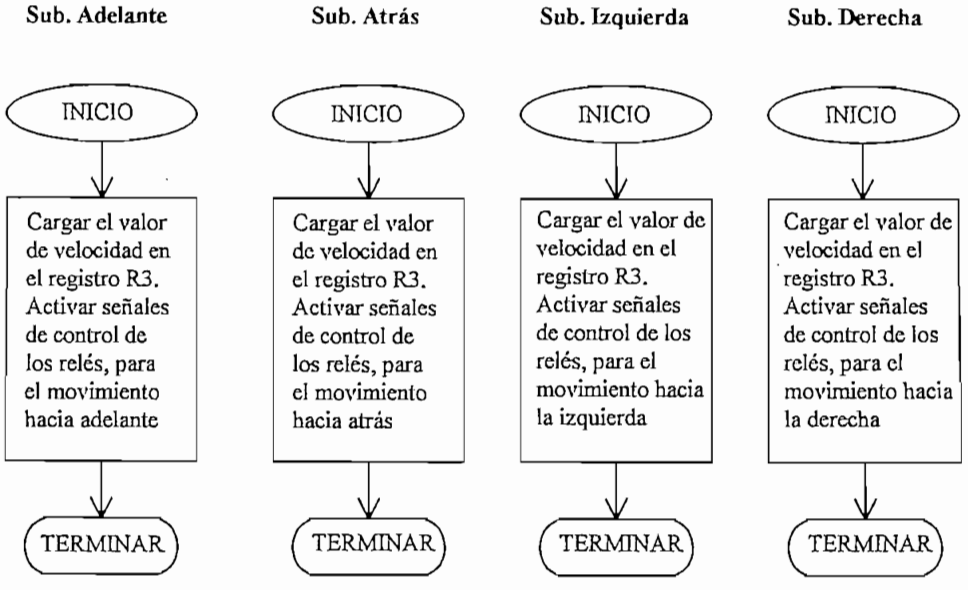
Subrutina Obstáculo



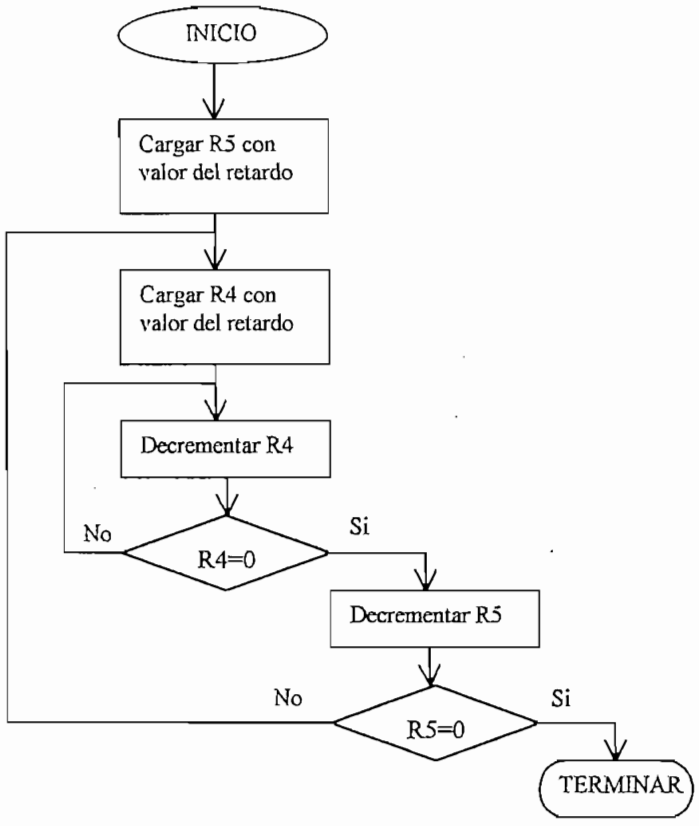
Subrutina Leer Sensores



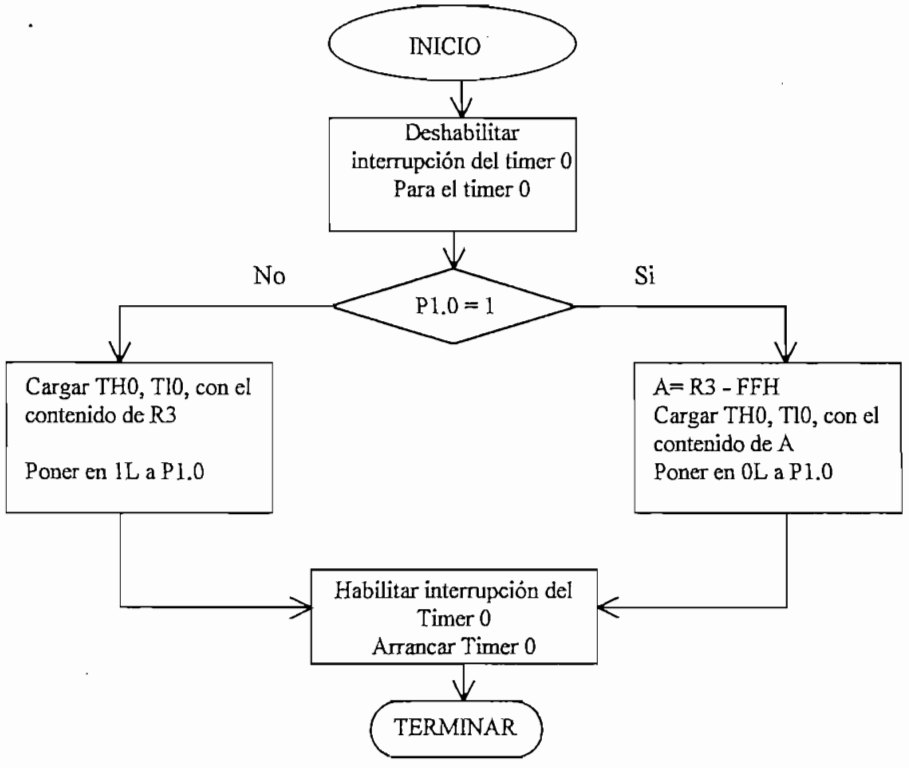
Subrutinas para el Movimiento del Robot



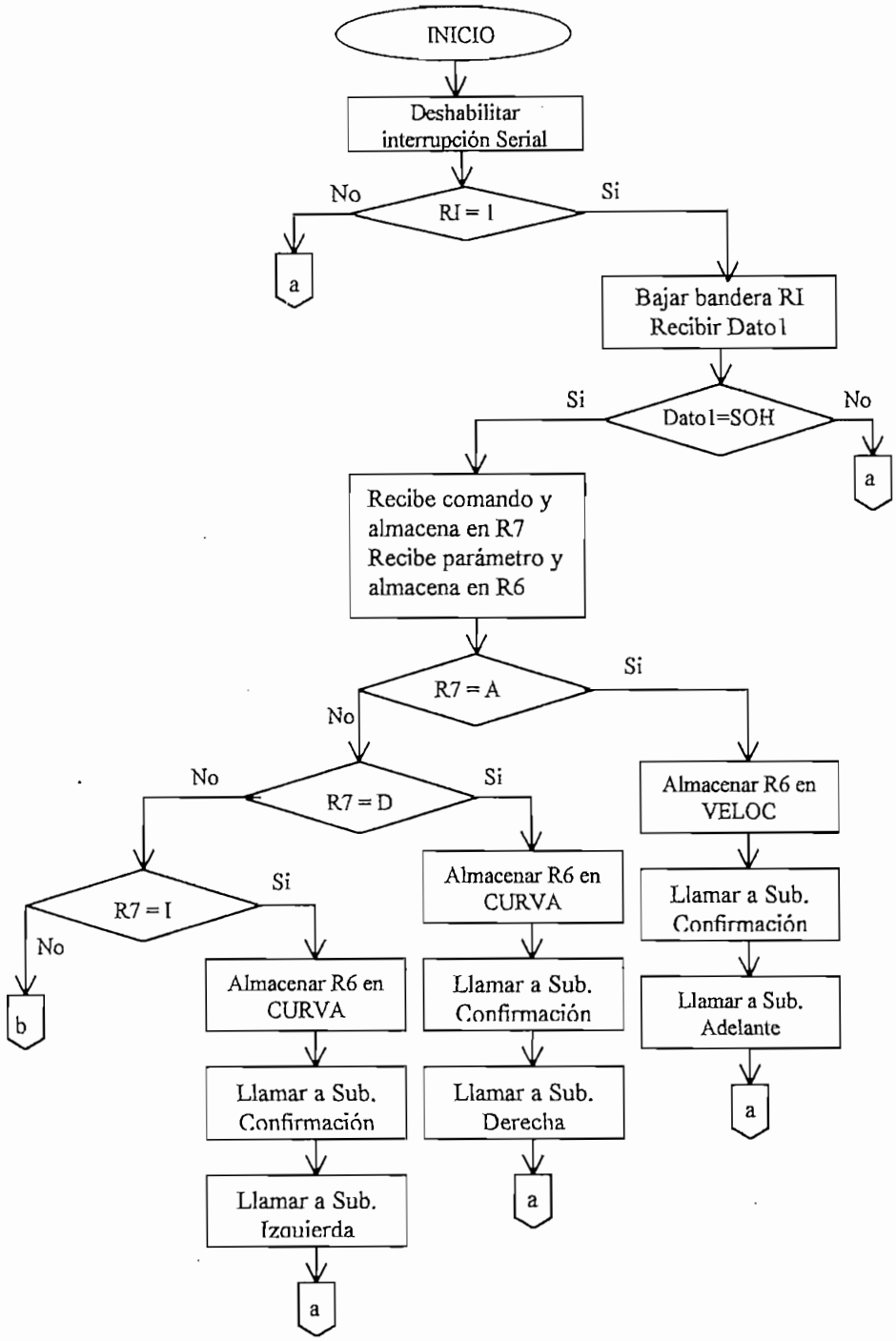
Subrutina Retardo

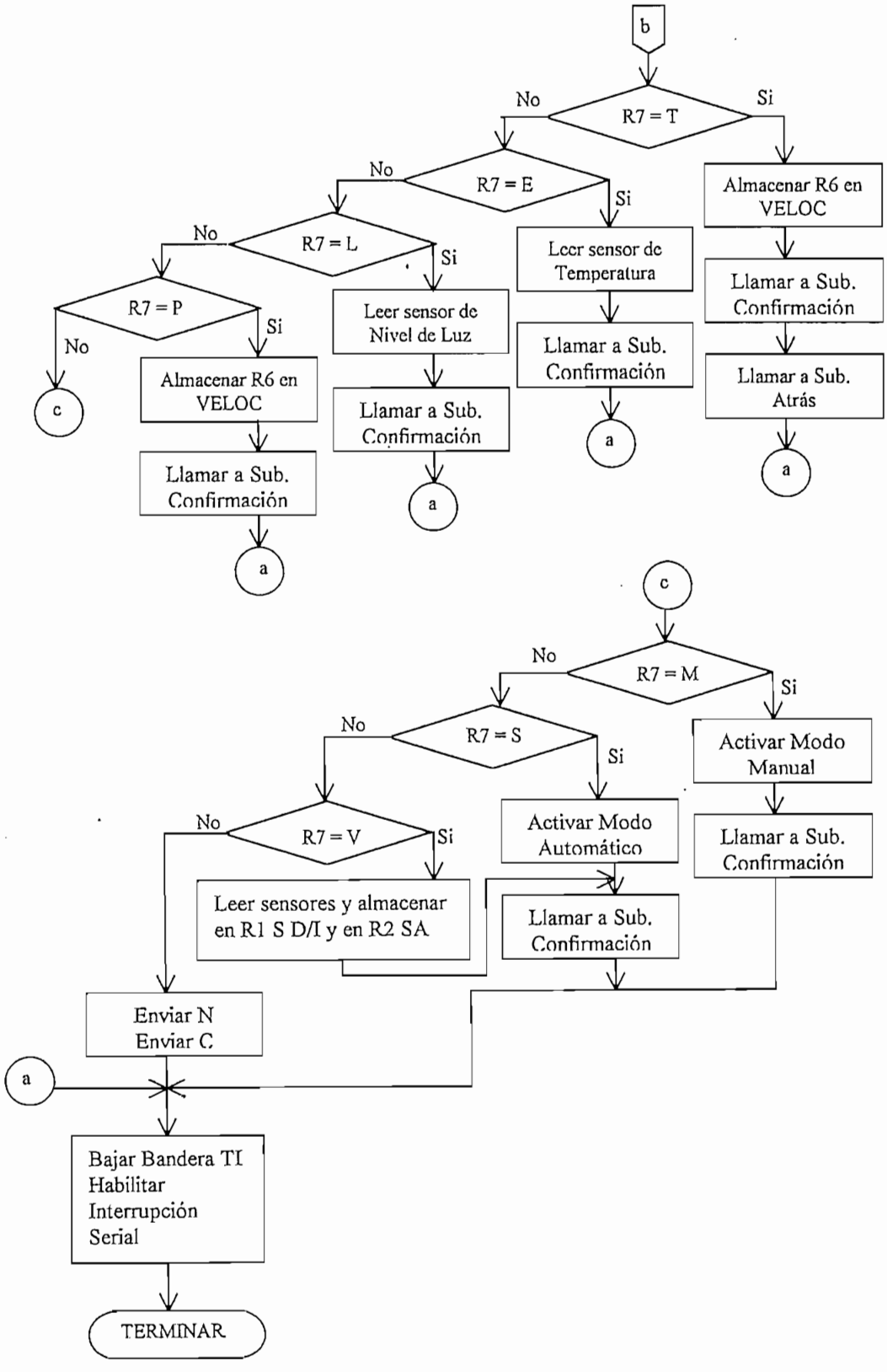


Rutina de Interrupción del Timer 0

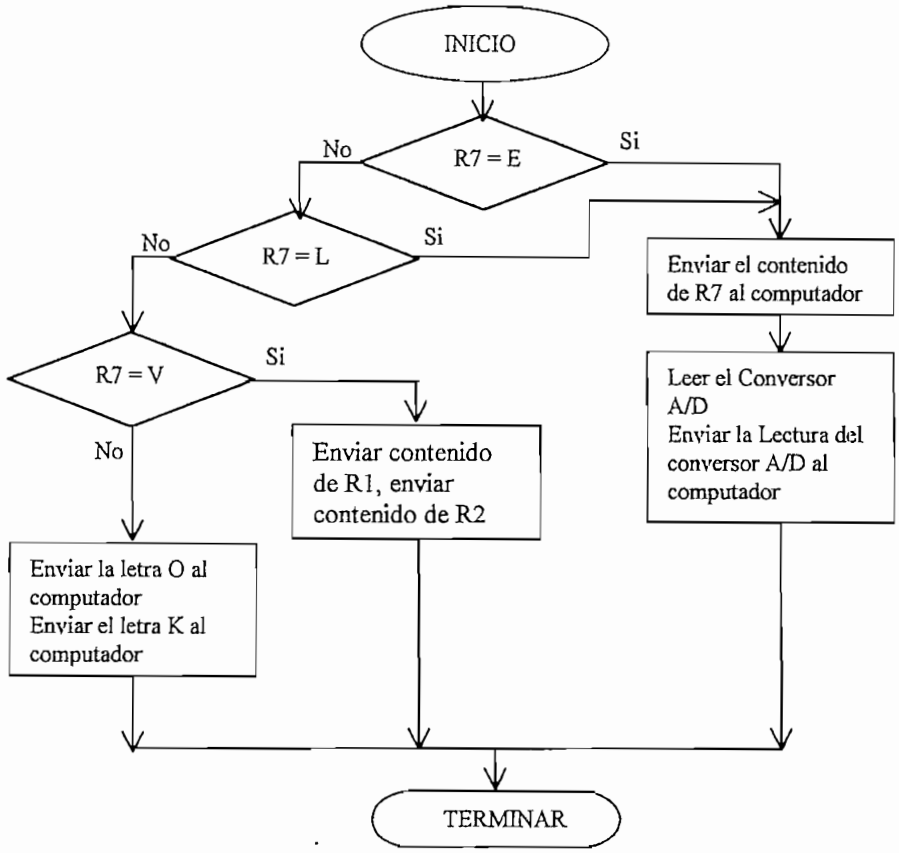


Rutina de la Interrupción Serial




b
R7 = T
No
Si
R7 = E
No
Si
R7 = L
No
Si
R7 = P
No
Si
c
Almacenar R6 en VELOC
Llamar a Sub. Confirmación
a
Leer sensor de Nivel de Luz
Llamar a Sub. Confirmación
a
Leer sensor de Temperatura
Llamar a Sub. Confirmación
a
Almacenar R6 en VELOC
Llamar a Sub. Confirmación
Llamar a Sub. Atrás
a
c
R7 = M
No
Si
R7 = S
No
Si
R7 = V
No
Si
Activar Modo Manual
Llamar a Sub. Confirmación
Activar Modo Automático
Llamar a Sub. Confirmación
Leer sensores y almacenar en R1 S D/I y en R2 SA
Enviar N
Enviar C
a
Bajar Bandera TI
Habilitar Interrupción Serial
TERMINAR

Subrutina Confirmación



6.2 ESTRUCTURA DE COMANDOS

Para realizar la comunicación entre el robot móvil y el computador central se estableció un protocolo de comunicación, utilizando la interfaz RS-232. Este protocolo contiene tres elementos: la cabecera, que indica el inicio de comunicación que será el # 1, el comando y el parámetro del mismo.

SOH ; COMANDO ; PARAMETRO

Manteniendo este protocolo, los comandos implementados son los siguientes:

1 ; A ; #	Movimiento hacia adelante
1 ; T ; #	Movimiento hacia atrás
1 ; D ; #	Giro a la Derecha
1 ; I ; #	Giro a la Izquierda
1 ; P ; #	Parar motores
1 ; M ; #X	Modo Manual
1 ; S ; #X	Modo Autónomo
1 ; E ; #X	Temperatura
1 ; L ; #X	Luz
1 ; V ; #X	Sensores de Ultrasonido

Movimiento hacia adelante (1;A;#)

Con este comando se da la orden al robot para que avance hacia adelante a la velocidad dada por un número entre 0 - 255, que constituye el ancho del pulso de la señal PWM.

Movimiento hacia atrás (1;T;#)

Este comando hará que el robot avance hacia atrás a una velocidad dada por el número, que estará comprendido entre 0 - 255.

Giro a la Derecha (1;D;#)

Hace que el robot gire a la derecha, a una velocidad dada por un número comprendido entre 0 - 255.

Giro a la Izquierda (1;I;#)

Hace que el robot gire a la izquierda, a una velocidad dada por un número comprendido entre 0 - 255.

Modo Manual (1;M;#X)

Cambia al robot móvil de modo autónomo a modo manual. Para este comando no se necesita parámetro, por lo que #X, representa a cualquier número, que es utilizado únicamente para cumplir con el protocolo.

Modo Autónomo (1;S;#X)

Cambia al robot móvil de modo manual a modo autónomo.

Temperatura (1;E;#X)

Pide al microcontrolador que se le envíe la lectura del sensor de temperatura.

Luz (1;L;#X)

Pide al microcontrolador que se le envíe la lectura del sensor de nivel de luz.

Sensores de Ultrasonido (1;V;#X)

Pide al microcontrolador se le envíe la lectura de los sensores de ultrasonido.

Todos los comandos antes descritos envían confirmación de comunicación al computador central, esta es enviada de la siguiente manera: para los comandos de movimiento y modo manual y autónomo, el microcontrolador envía: **O;K** que significa comunicación correcta. Mientras que para los comandos temperatura y luz, el microcontrolador envía: **E;#, y L;#** respectivamente, donde #, es un número de 0 - 255 que representa el nivel de la variable medida, y que físicamente es la lectura del conversor A/D a cuya entrada ingresa la señal de los sensores antes mencionados. En caso de los sensores de ultrasonido el microcontrolador envía dos números, el primero es la lectura de los sensores laterales y el segundo la lectura de los sensores de delanteros.

En el caso de iniciar comunicación, enviando un comando erróneo, el microcontrolador retorna **N;C** que significa ningún comando.

Capítulo 7

7.1 CONCLUSIONES Y RECOMENDACIONES

El trabajo realizado en esta tesis abarca mucho más de los objetivos planteados, es así que ahora se cuenta con un prototipo, al cual se le pueden añadir más dispositivos para navegación y monitoreo, sin hacer modificaciones en el mismo, ya que la tarjeta MCPD51 utilizada tiene dividido el mapa de memoria externa en dieciséis espacios de memoria, para colocar hasta ocho dispositivos de entrada y ocho dispositivos de salida, que serán tratados como memoria externa del microcontrolador. En el prototipo únicamente se están utilizando tres espacios de memoria para entradas y dos para salida.

Al utilizar sensores de ultrasonido cuyo ancho de banda es muy pequeño, es recomendable utilizar un circuito oscilador basándose en un cristal de cuarzo, ya que estos son muy estables y no ocasionan problemas a los sensores cuando la temperatura varía.

Cuando se trabaja con motores de DC pequeños, como los de los juguetes, hay que tener mucho cuidado con el ruido que este tipo de motores introducen en el sistema, por lo que es necesario realizar las protecciones necesarias para que este ruido no afecte a los componentes del sistema y provoque un mal funcionamiento en el mismo.

Existen algunas ayudas para la simulación, como la variación del tiempo entre cada movimiento del móvil. Esto permite variar la velocidad de la animación dependiendo de las características de velocidad del microprocesador y el computador en que se ejecute el programa. Una capacidad adicional comprende el uso de un factor aleatorio en el movimiento del móvil que simula deslizamientos de las llantas y variaciones bruscas de giro en el robot real. Existe también un cuadro de diálogo que permite simular el ajuste y calibración de los sensores, que es de gran ayuda para el estudio del comportamiento de los mismos.

Si se desea mejorar la comunicación, se pueden agregar registros de los sensores de temperatura y de luz, o comandos que después de pasar cierto nivel ejecuten una acción.

Existe cierta limitación para modelar a los sensores reales de ultrasonido en el simulador, por ejemplo las reflexiones múltiples o no - reflexiones que requerirían un algoritmo complejo. Otro tipo de limitación es que no se puede modelar exactamente el ambiente de operación, como son el tipo de material, textura de los objetos, etc.

Las herramientas para la creación de ambientes de trabajo de este simulador son simples pero pueden crear ambientes relativamente complejos. Para ambientes de mayor complejidad se necesita un código mucho más elaborado, por lo que se recomienda crear herramientas dependiendo del tipo de aplicación.

El campo de la Robótica Móvil, en nuestro país, debería ser desarrollado y adaptado a las necesidades de nuestra industria, para mejorar su rendimiento y abaratar los costos de producción. Pero no sólo en la industria puede ser utilizado, es así que este tipo de robots podrían utilizarse como exploradores de lugares nocivos para los humanos, o lugares a los que no se tiene un libre acceso como por ejemplo el interior de tuberías, en las cuales el robot móvil podría monitorearlas, detectar fallas y realizar arreglos de las mismas.

Es recomendable para robots móviles utilizar la mayor cantidad posible de sensores de ultrasonido, para que el robot tenga mayor información de su ambiente de trabajo y pueda realizar sus tareas de una manera más óptima y segura.

La realización de una herramienta de simulación que tenga el mismo sistema de control de un robot móvil ayuda para detectar fallas y realizar correcciones del sistema de control, ya que esta herramienta permite simular

muchos ambientes y verificar el funcionamiento del robot en poco tiempo y costo menor comparado con el mismo estudio en el robot real.

7.2 TRABAJO FUTURO

A continuación se dan sugerencias para mejorar el prototipo desarrollado en la presente tesis:

Para conseguir que el prototipo no esté atado al computador central con un cable, se sugiere realizar comunicación inalámbrica utilizando para este objeto radio modems, o una opción más económica que sería una comunicación Full Duplex FKS Modem utilizando los chips XR-2296 y XR-2211 de EXAR.

Se sugiere realizar el control de posición del prototipo, aumentando su grado de inteligencia, como por ejemplo el poder definir metas a las que el robot debe llegar evadiendo los obstáculos que encuentre a su paso.

Utilizando las entradas disponibles en el conector JP4 de la tarjeta de acondicionamiento de sensores, que se encuentra colocado en la entrada del conversor A/D de la tarjeta MCPD51, se puede colocar hasta cuatro sensores que podría utilizar el robot móvil para realizar tareas de navegación más avanzadas.

Se puede relacionar la comunicación con el simulador, haciendo que la información recopilada en la comunicación se muestre en la ventana del simulador, y de esta manera realizar el mapa del ambiente de trabajo del robot real.

Finalmente, se recomienda que se realice más trabajos de investigación en este fascinante campo de la Robótica Móvil, una área que sin duda tendrá un gran desarrollo en el próximo milenio.

Bibliografía

BIBLIOGRAFIA

[1] J. Borenstein, N. Everett and L. Feng, *Navigating Mobile Robots System and Techniques*, Wellestey, MA: AK Peters, 1996

[2] P. J. McKerrow, *Introduction to Robotics*, Sidney: Addison-Wesley Publishing Company, 1993.

[3] Y. Kanayama, Y. Kimura, F. Miyazaki and T. Noguchi, "A Stable Tracking Control Method for an Autonomous Mobile Robot," *Proc. IEEE Int. Conf. on Robot. and Automat.*, vol. 1, May 1990, pp. 384-389.

[4] Y. F. Zheng, *Recent Trends in Mobile Robots,* Singapore: World Scientific Publishing Co. Pte. Ltd., 1993.

[5] B. Ledesma, *Tarjeta MCPD51*: E.P.N. Quito, Septiembre 1992.

[6] J. Hermann, S. Eduard and L. Rolf, *Tablas para la Industria Metalúrgica*, Barcelona: Reverté, S.A., 1984.

[7] J. Carr, "Desinging with Digital IC's," *Proc. Radio Electronics*, vol. 57, Jan. 1986, pp. 71-99.

[8] Ediciones Nueva Lente, *"Montaje de un Sistema Emisor - Receptor de Ultrasonidos," Proc. Enciclopedia Práctica ELECTRONICA, vol 1, 1982, pp. 208-216.*

[9] M. Rashid, *Electrónica de Potencia*, México: Prentice Hall Hispanoamericana, S.A., 1993.

[10] P. Rivera, *Control de Máquinas Eléctricas*, E.P.N. Quito, 1997.

[11] J. González, *Introducción a los Microcontroladores,* España: McGraw-Hill / Interamericana de España, S.A., 1992

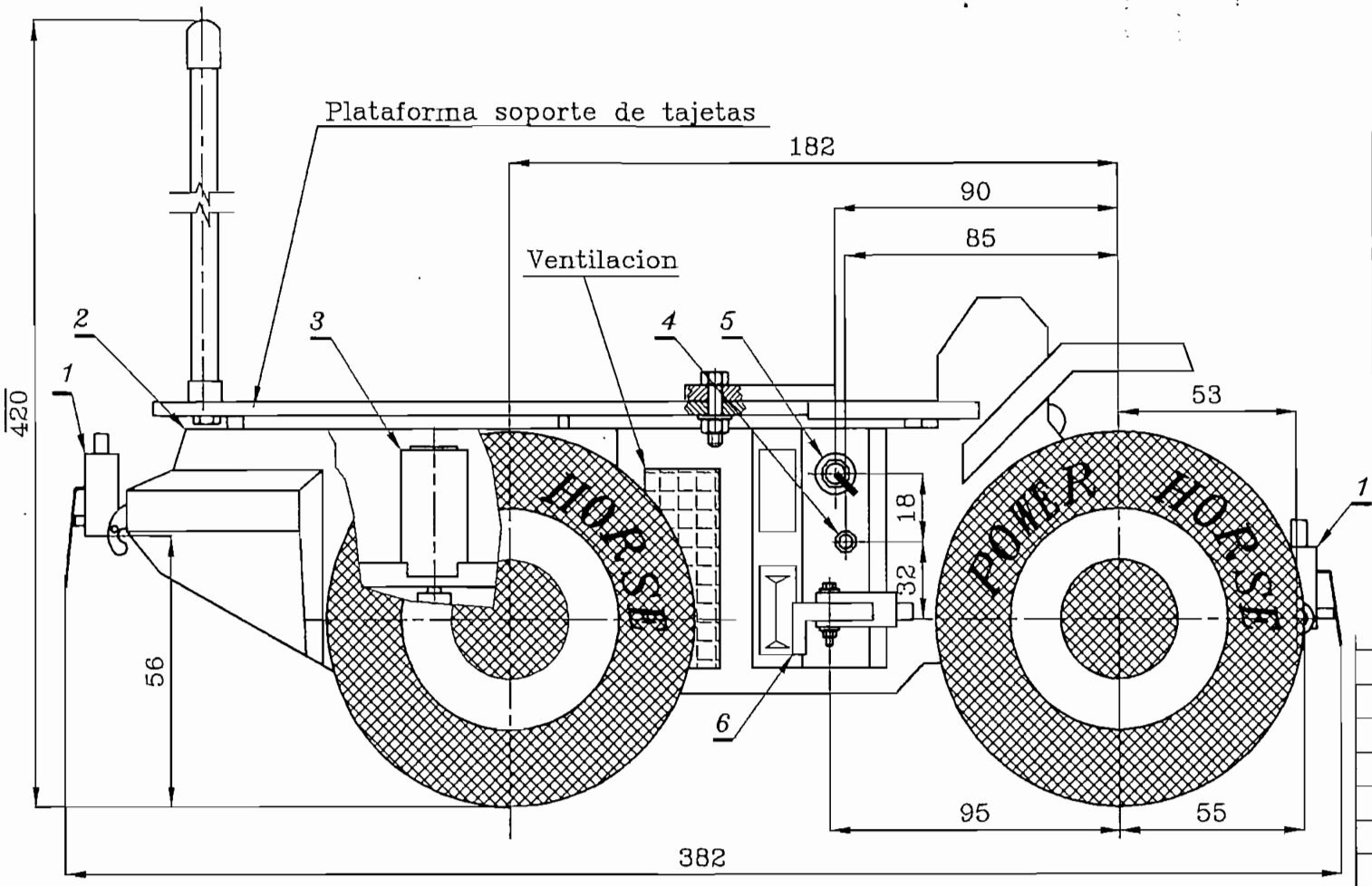
[12] B. Kernigham and D. Ritchie, *El Lenguaje de Programación C*, México, D.F.: Prentice-Hall Hispanoamericana, S.A., 1992.

[13] H. Schildt, *Aplique Turbo C++ para Windows,* España: McGraw-Hill Interamericana S.A., 1993.

[14] M. Andrews, *Aprenda Visual C++ Ya*, España: McGraw-Hill Interamericana S.A., 1996.

[15] Manual de Usuario, *Microsoft Visual C++ Versión 4.0*, Microsoft Corporation, 1995

Anexo A



ESPECIFICACIONES TECNICAS	
MOTORES	
VOLTAJE	6V
AMPERAJE	0,5–1 A

No	Cant	DESCRIPCION	OBSERVACIO
6	2	SWITCHES DE PROTECCION	
5	2	SELECTOR ON/OFF/RECARGA	Derecha fuente mot Izquierda fuente con
4	2	JACK PARA RECARGA	Derecha fuente mot Izquierda fuente con
3	2	MOTOR	0.5-1 A / 6
2	1	Bastidor New Bright	
1	2	SWITCHES DE PROTECCION	

Nota: los ejes de los interruptores (1) están alineados con los ejes medios del prototipo
Distribucion de caja de engranajes en hoja (AS002)
ancho total del prototipo 185 mm

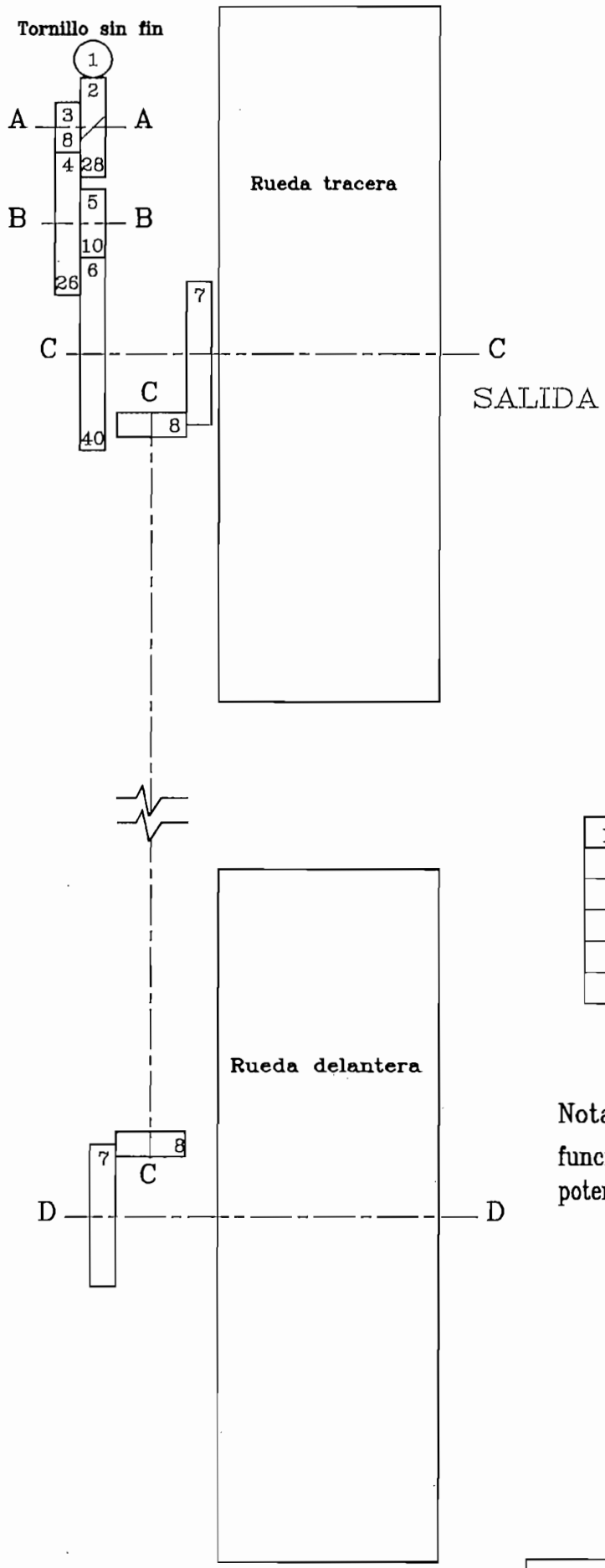
FACULTAD DE INGENIERIA ELECTRICA

A4

PLATAFORMA MOVIL
Pionero1

	Modificacion				DWG. No:	Stat:	Rev:
Date: 1998		Drawn: Andrango - Sotomayor			AS001	LC	
Dia:NEW BRIGHT		Date: 98/04					
Scale: 1 : 2		Andrango - Sotomayor				Sheet: 1	

DIAGRAMA ESQUEMATICO DEL REDUCTOR DE VELOCIDAD



No	No de dientes	Diámetro	Plano de fabricación
2	28	16 mm	A—A
3	8	8 mm	A—A
4	26	23 mm	B—B
5	10	11 mm	B—B
6	40	31 mm	C—C

Nota: los engranajes 7 y 8 funcionan como extensiones de potencia a las llantas delanteras

Hoja AS002

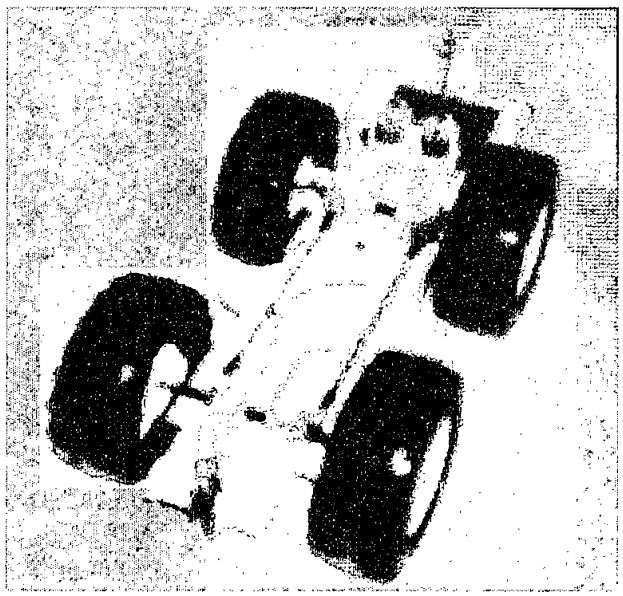


Figura A.1. Vista del Reductor de Velocidad y Sistema de Transmisión

Figura A.2. Carrocería Utilizada para el Robot Móvil

Anexo B

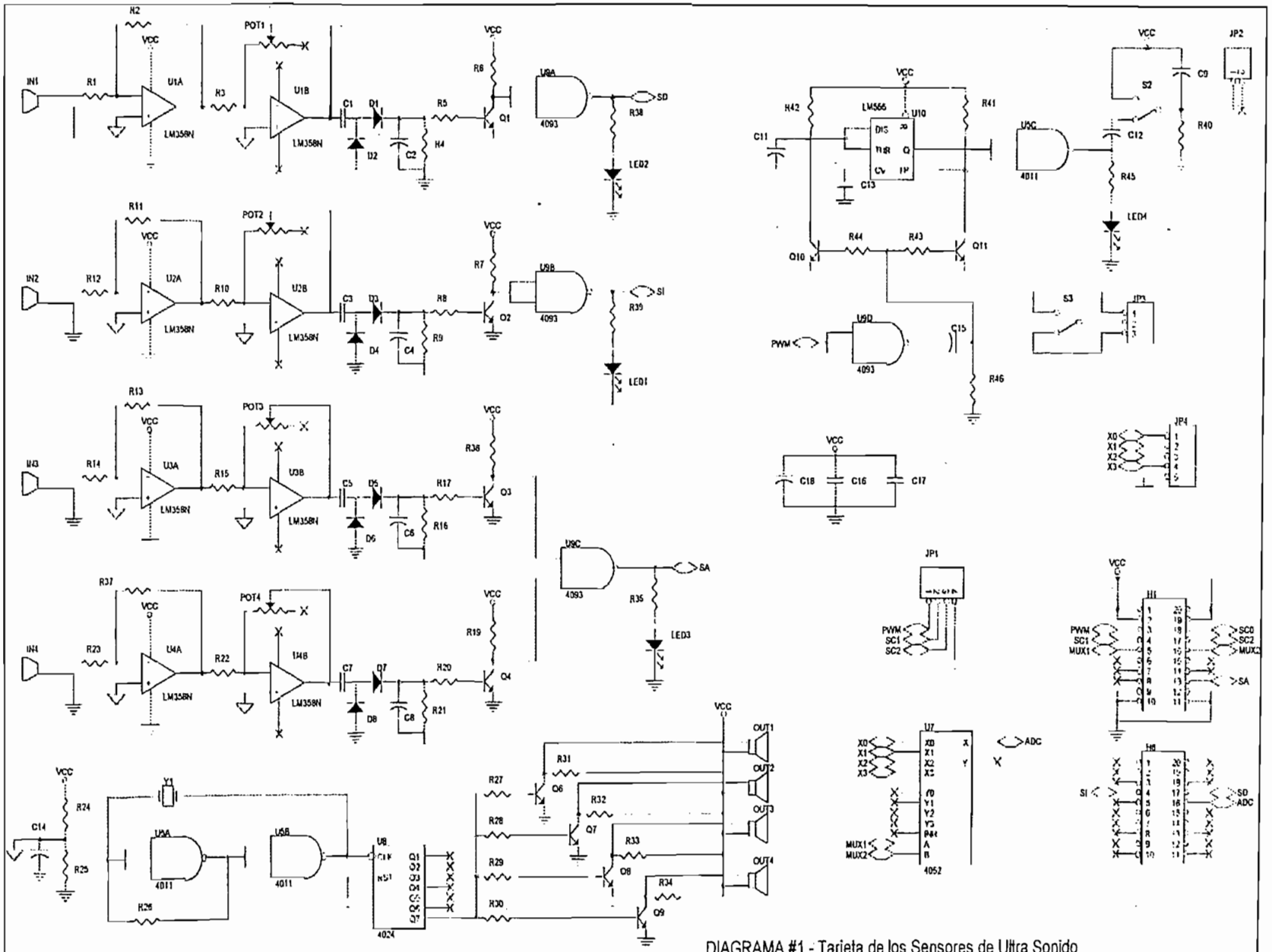


DIAGRAMA #1 - Tarjeta de los Sensores de Ultra Sonido

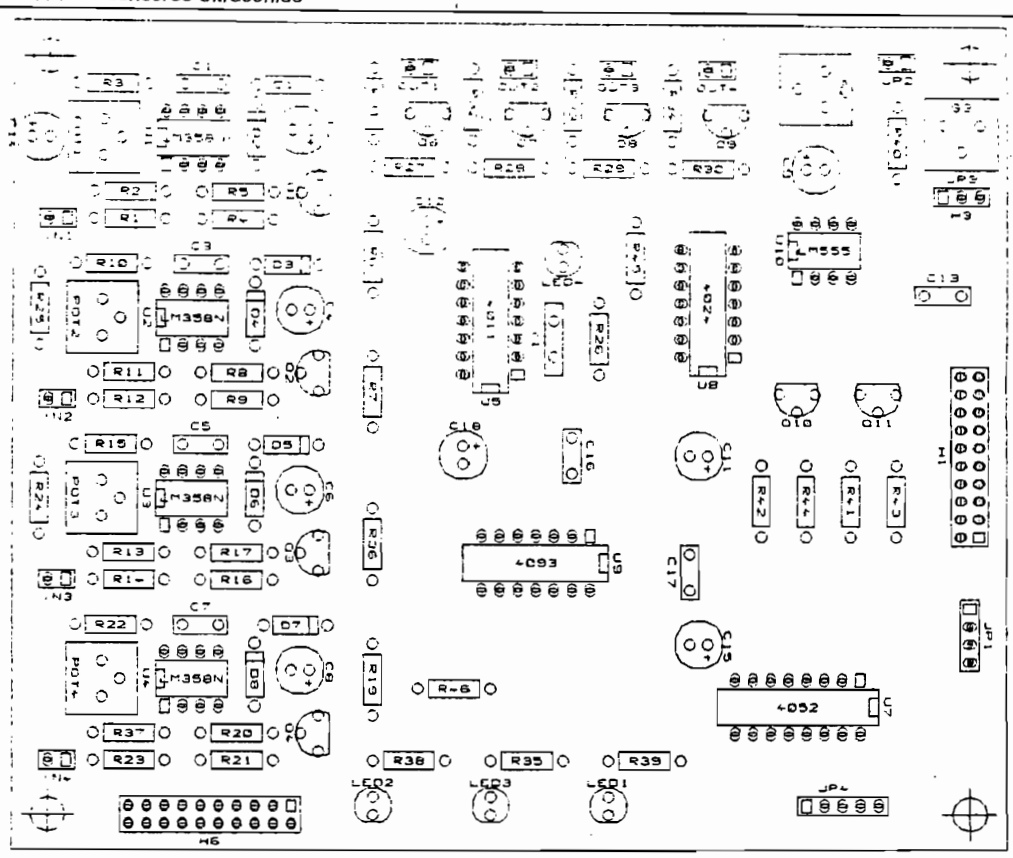


Figura B.1 Placa #1 - Componentes

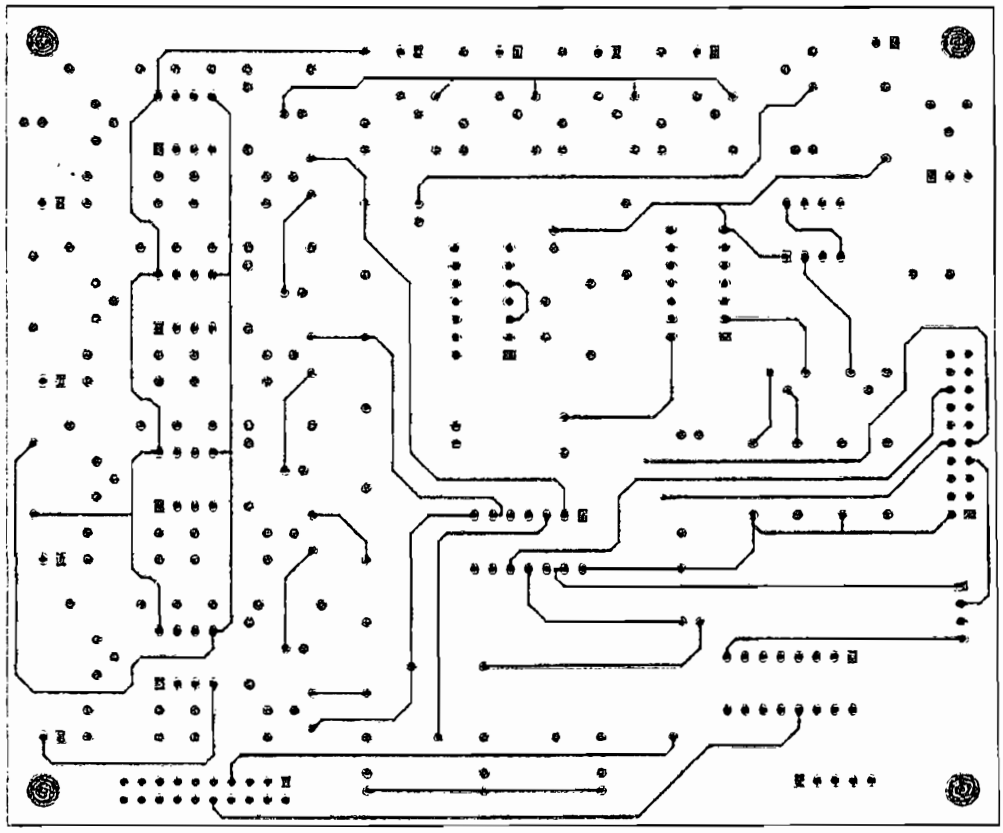


Figura B.2 Placa #1 – Pista Superior

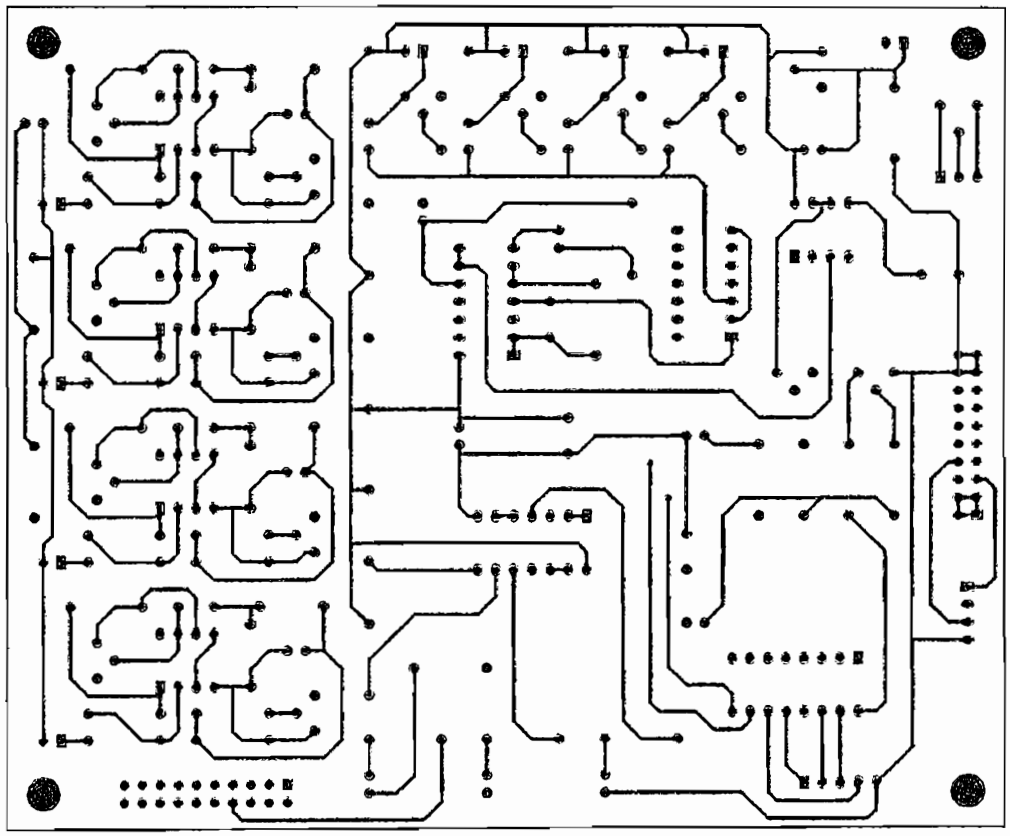


Figura B.3 Placa #1 – Pista Inferior

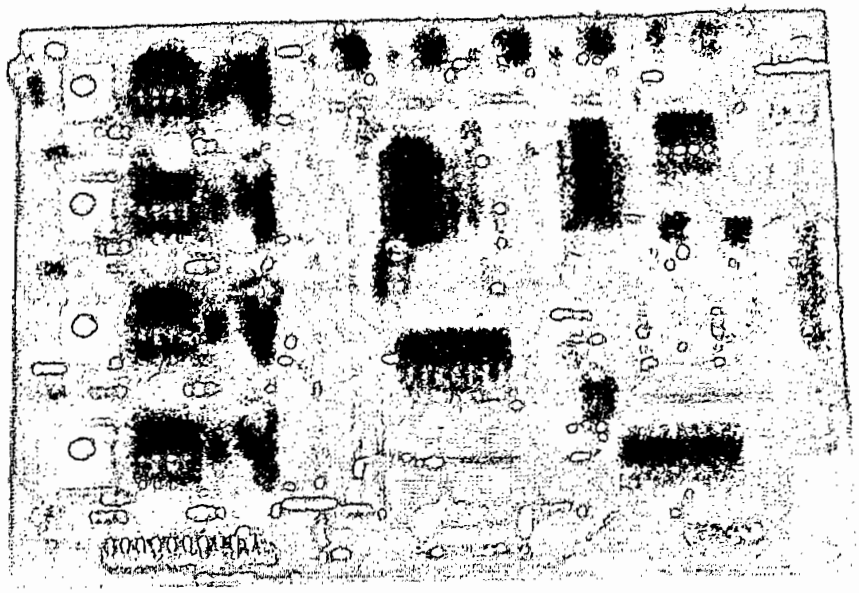


Figura B.4 Placa #1 Acondicionamiento de Sensores de Ultrasonido

Tabla B.1 Lista de Componentes

ELEMENTO	VALOR	DESCRIPCION
R1, R12, R14, R23, R41	10K	Resistencias de 1/4 W
R2, R11, R13, R37	100K	Resistencias de 1/4 W
R3, R10, R15, R22	5,6K	Resistencias de 1/4 W
R4, R9, R16, R21	3,9K	Resistencias de 1/4 W
R5, R8, R17, R20, R31, R32, R33, R34	1K	Resistencias de 1/4 W
R6, R7, R19, R36	4,7K	Resistencias de 1/4 W
R24, R25	100 Ohm	Resistencias de 1/4 W
R26	22K	Resistencias de 1/4 W
R27, R28, R29, R30	3,3K	Resistencias de 1/4 W
R35, R38, R39, R45	220 Ohm	Resistencias de 1/4 W
R40	8,2K	Resistencias de 1/4 W
R42	300 Ohm	Resistencias de 1/4 W
R43, R44	2K	Resistencias de 1/4 W
POT1, POT2, POT3, POT4	100K	Potenciometros de precisión
C1, C3, C5, C7, C16	0,1uF	Condensadores cerámicos
C2, C4, C6, C8	47uF	Condensadores electrolíticos
C9, C14	10uF	Condensadores electrolíticos
C12	100uF	Condensadores electrolíticos
C11	22uF	Condensadores electrolíticos
C15, C17	1uF	Condensadores electrolíticos
D1, D2, D3, D4, D5, D6, D7, D8		Diodos de señal 1N4007
Q1.. Q11		Transistores NPN, 2N3904
LED1, LED2, LED3, LED4		Leds
U1, U2, U3, U4		Operacionales, LM358N
U5		Compuertas NAND, 4011
U7		Differentials dual 4 Ch. Multiplexer, 4052
U8		Contador binario de 7 estados, 4024
U9		Compuertas NAND Schmitt Trigger, 4093
U10		Timer, LM555
Y1	5,18MHz	Cristal de cuarzo
S2		Pulsador NA, NC
S3		Selector

Anexo C

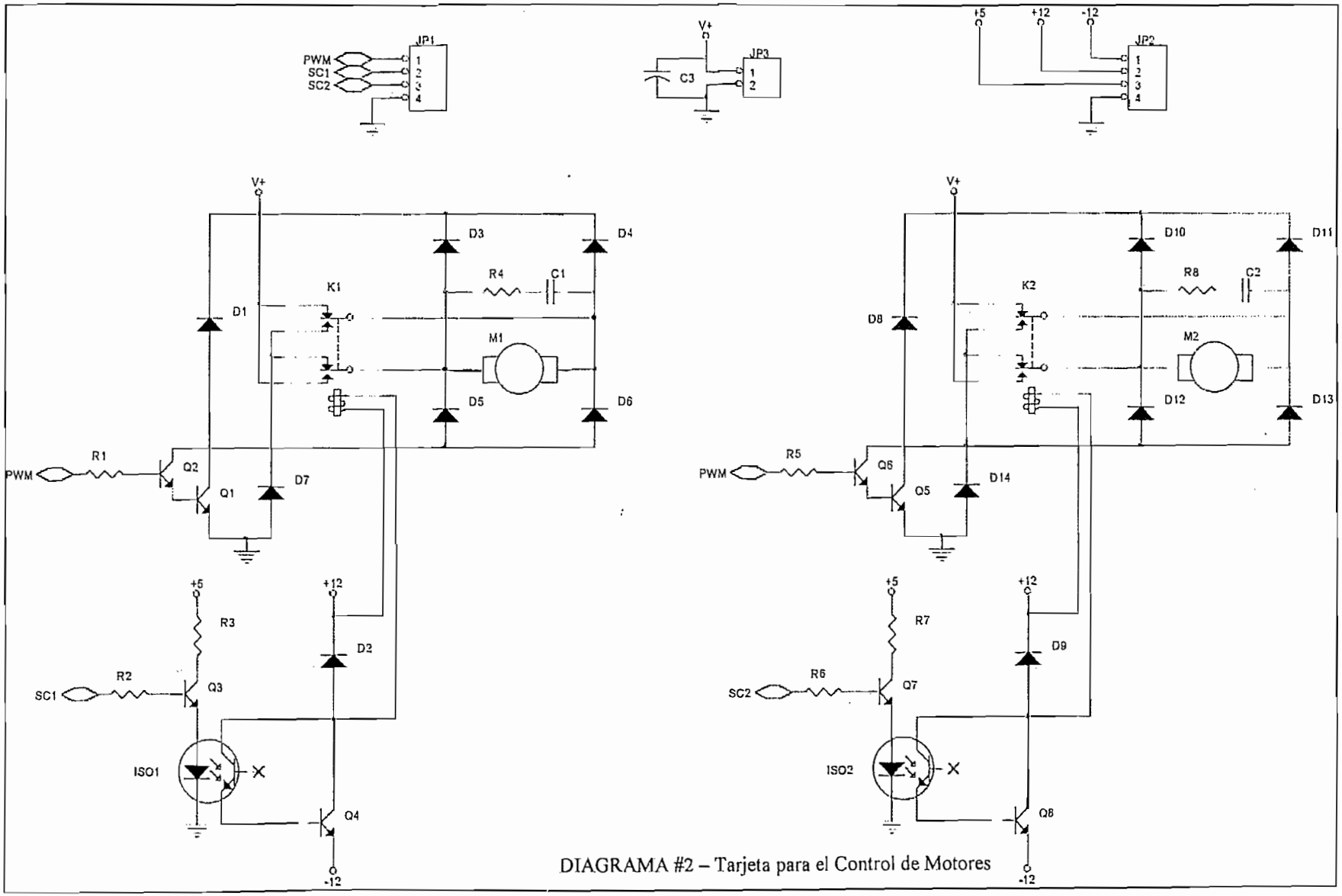


DIAGRAMA #2 – Tarjeta para el Control de Motores

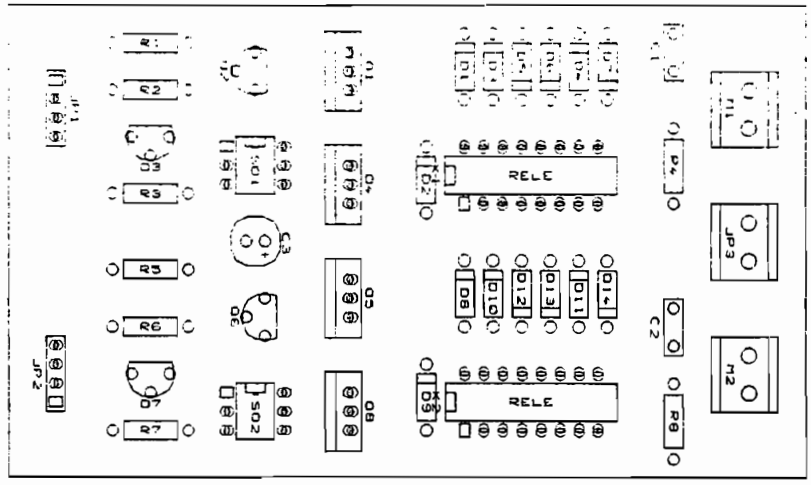


Figura C.1 Placa #2 - Componentes

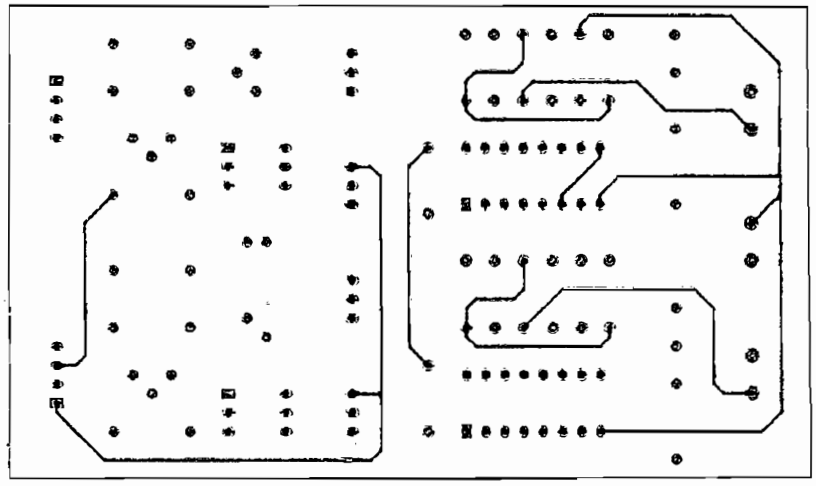


Figura C.2 Placa #2 – Pista Superior

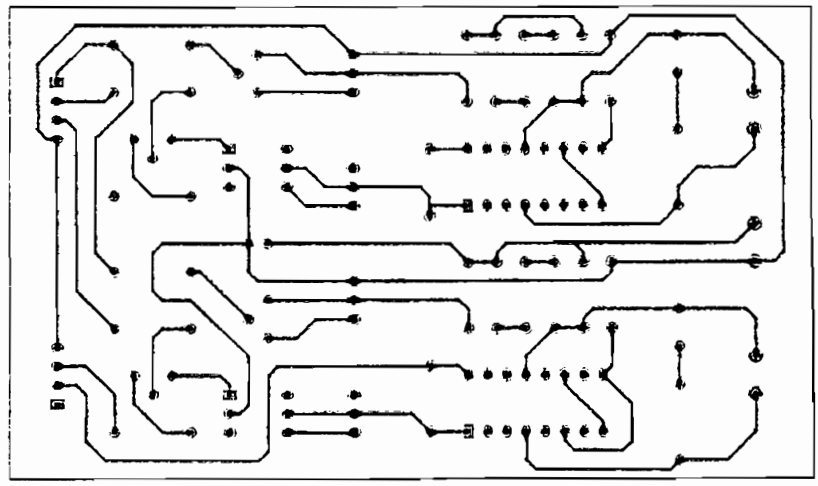


Figura C.3 Placa #2 – Pista Inferior

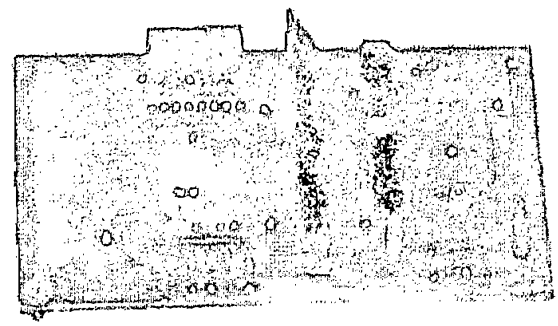


Figura C.4 Placa #2 – Control de Motores

Tabla C.1 Lista de Componentes

ELEMENTO	VALOR	DESCRIPCION
R1, R5	3,9K	Resistencias 1/4 W
R2, R6	10K	Resistencias 1/4 W
R3, R7	220 Ohms	Resistencias 1/4 W
R4, R8	150 Ohms	Resistencias 1/4 W
C1, C2	0,1 uF	Condensadores cerámicos
C3	10 uF	Condensador electrolítico
D1.. D14	1N4148	Diodos Si, Fast Switching
Q1, Q4, Q5, Q8		Transistores de Potencia TIP 110
Q2, Q3, Q6, Q7		Transistores NPN, 2N3904
SO1, SO2		OPTO, 4N28, NPN TRANS
RELE		Relé LMI 2E00

Anexo D

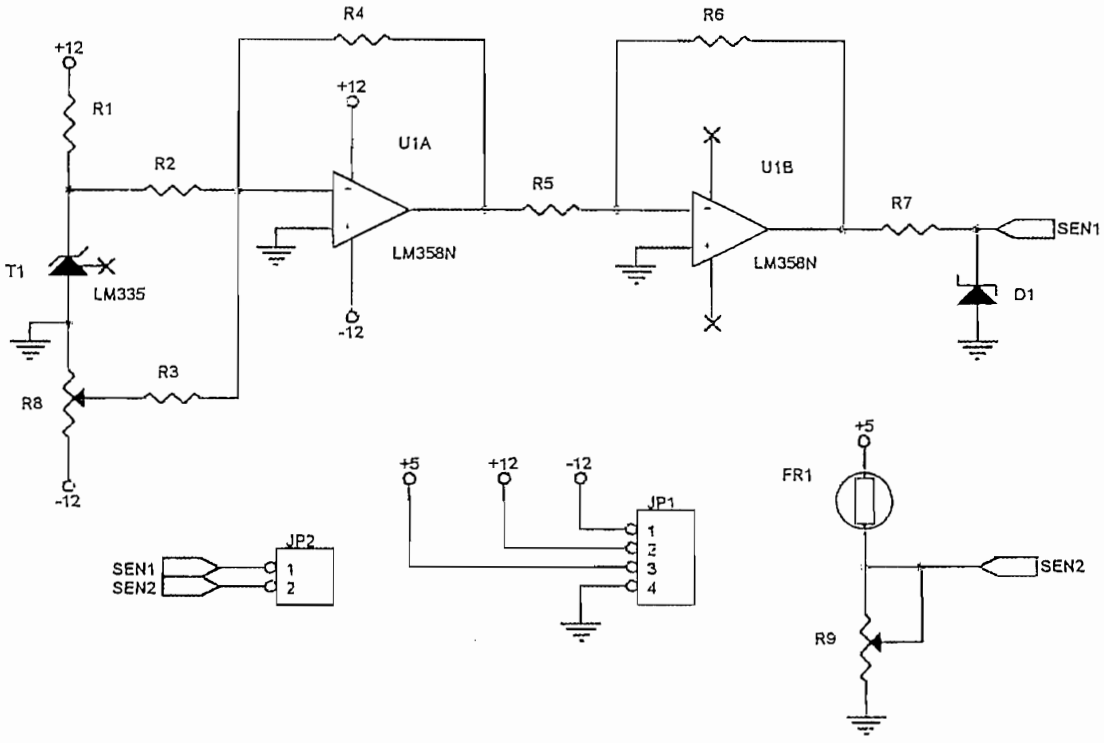


Figura D.1 DIAGRAMA #3 – Sensores de Luz y Temperatura

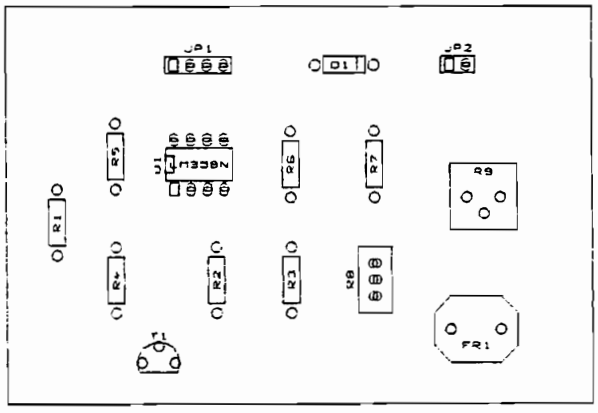


Figura D.2 PLACA #3 – Componentes

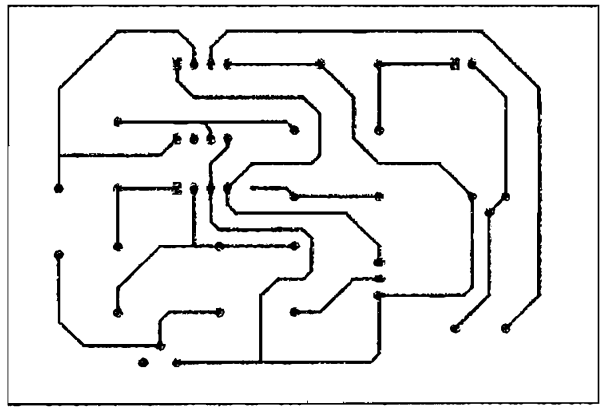


Figura D.3 PLACA #3 – Pista Inferior

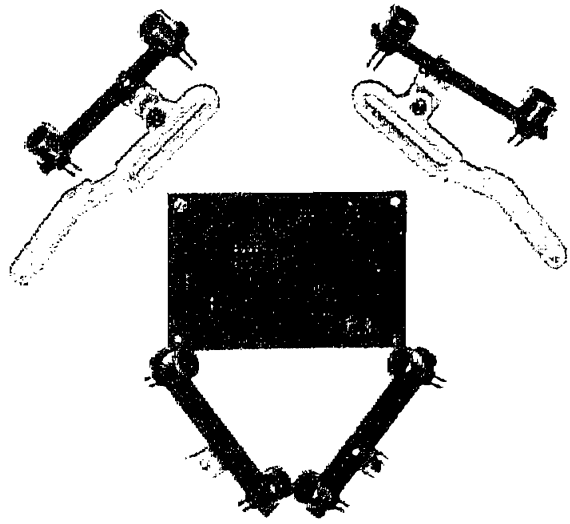


Figura D.4 Sensores de Ultrasonido, Temperatura, Luz

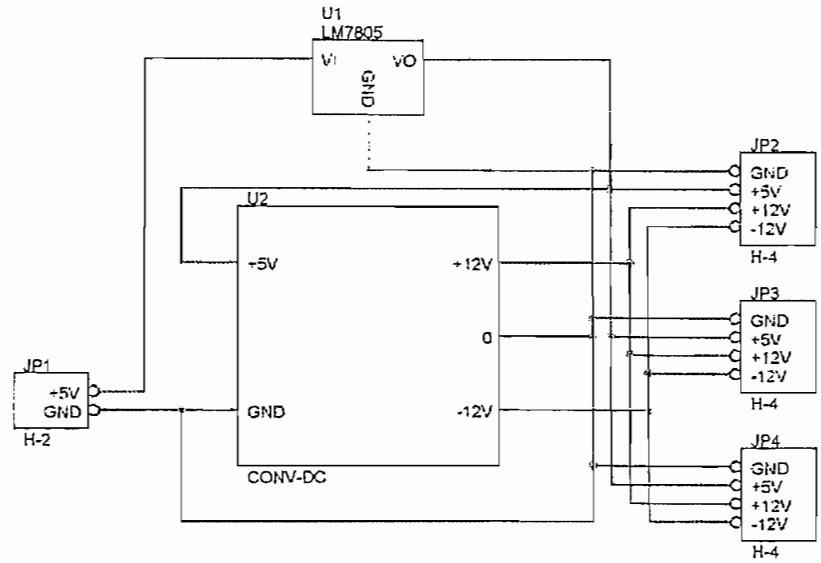


Figura D.5 DIAGRAMA #4 – Fuentes

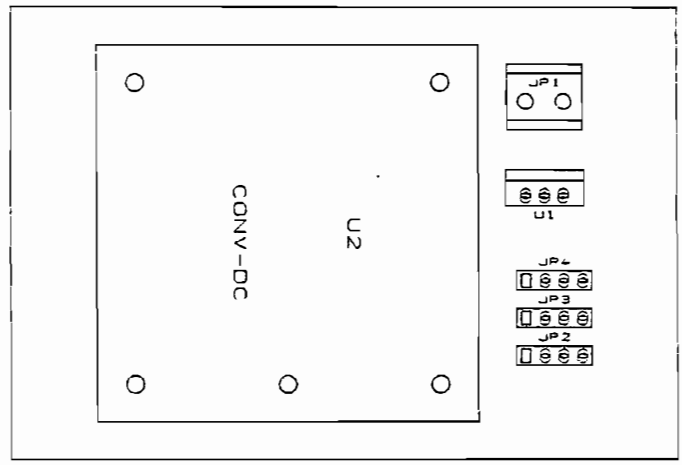


Figura D.6 PLACA #4 - Componentes

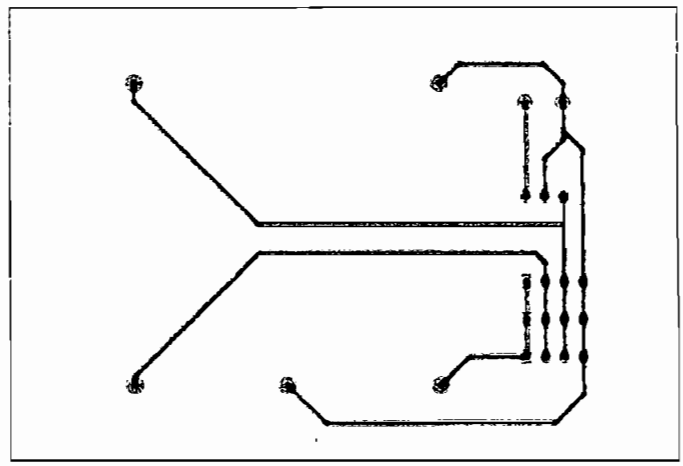


Figura D-7 PLACA #4 – Pista Inferior

Tabla D.1 Listado de Elementos Placa #3

ELEMENTO	VALOR	DESCRIPCION
R1, R2, R3, R5, R6, R7	10K	Resistencias de 1/4 W
R4	100K	Resistencias de 1/4 W
R8, R9	100K	Potenciometros
D1	5.1V	Diodo Zener
FR1		Fotocelda
T1		Sensor de Temperatura, LM335
U1		Operacional, LM358N

Tabla D.2 Listado de Elementos Placa #4

ELEMENTO	VALOR	DESCRIPCION
U1	LM7805	Regulador de Voltaje
U2		9 watt converter
JP1		Conector de Fuente
JP2 .. JP4		Conectores de Fuentes

Hoja del Fabricante

9 watt converter

-Input voltage: 5VDC @ 2.1A
-PI input and output filters
-Isolation voltage: 500VDC
-Isolation capacitance: 180pf
-Isolation resistance: 1.0Mohm
-Short circuit protection
-Switching frequency: 20kHz
-Output: +12VDC @ 375mA, -12VDC @ 375mA
-Size: 2.0"L x 2.0"W x 0.5"H
-Weight: 0.1 lbs.

Part No.	Product No.	Price (1+)	Price (10+)	Price (25+)	Quantity
116986	EM951S	7.95	7.25	6.49	
Add to Order					

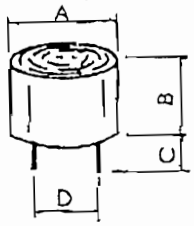
ULTRASONIC SENSOR (TRANSMITTER / RECEIVER USE)

MODEL NO.: 40TR16F (CASE:ALUMINUM/ FLOWER)

MODEL NO.: 40TR16P (CASE:PLASTIC/ BLACK)

136653

OUTLINE DIMENSION :



A=16.1 ±0.3
B=12.0 ±0.4
C=9.6 ±1.0
D=10.0 ±0.5

Unit:mm

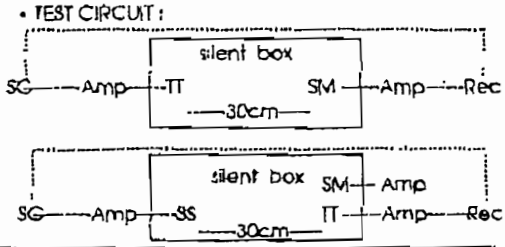
TRANSMITTER UNIT:

. Center Frequency (KHz)	: 40.0 ±1.0
. Sound Pressure Level (0 dB=0.0002 μbar)	: 119dB/40 ±1.0KH /10V SN/30cm /Min.
. Band Width (KHz) Min.	: 4.0 / 112 dB
. Capacitance (PF)	: 2000 ± 30%

RECEIVER UNIT:

. Center Frequency (KHz)	: 40.0 ±1.0
. Sensitivity Min. (0 dB vs 1V μbar)	: 65dB/40 ±1.0KH (R=3.9kΩ)
. Band Width (KHz) Min.	: 3.5 / (at -71 dB)
. Capacitance (PF)	: 2000 ± 30%

• TEST CIRCUIT :



TT : test transducer
Rec : recorder
SM : std. microphone
Amp : amplifier
SG : signal generator

TT : test transducer
Rec : recorder
SM : std. microphone
Amp: amplifier
SG : signal generator
SS : std. speaker
R : resistance

MODEL NO. 40TR16F (CASE:ALUMINUM/BLACK)

139491

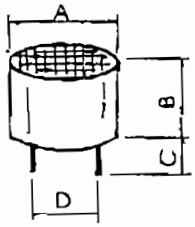
TRANSMITTER UNIT:

. Center Frequency (KHz)	: 40.0 ±1.0
. Sound Pressure Level (0 dB=0.0002 μbar)	: 112dB/40 ±1.0KH /10V SN/30cm /Min.
. Band Width (KHz) Min.	: 5.0 / 100 dB
. Capacitance (PF)	: 2000 ± 20%

RECEIVER UNIT:

. Center Frequency (KHz)	: 40.0 ±1.0
. Sensitivity Min. (0 dB vs 1V μbar)	: 67dB/40 ±1.0KH (R=3.9kΩ)
. Band Width (KHz) Min.	: 5.0 / (at -75 dB)
. Capacitance (PF)	: 2000 ± 20%

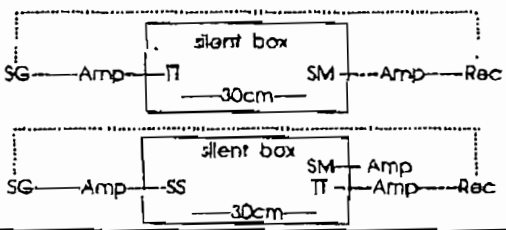
OUTLINE DIMENSION :



A=12.7 ±0.2
B= 9.4 ±0.4
C= 6.2 ±0.3
D= 8.5 ±0.2

Unit:mm

• TEST CIRCUIT :



TT : test transducer
Rec : recorder
SM : std. microphone
Amp : amplifier
SG : signal generator

TT : test transducer
Rec : recorder
SM : std. microphone
Amp: amplifier
SG : signal generator
SS : std. speaker
R : resistance

MODEL NO. 40TR16P (CASE:PLASTIC/BLACK)

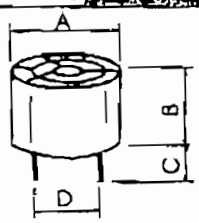
TRANSMITTER UNIT:

. Center Frequency (KHz)	: 40.0 ±1.0
. Sound Pressure Level (0 dB=0.0002 μbar)	: 110dB/40 ±1.0KH /10V SN/30cm /Min.
. Band Width (KHz) Min.	: 4.0 / 104dB
. Capacitance (PF)	: 2000 ± 30%

RECEIVER UNIT:

. Center Frequency (KHz)	: 40.0 ±1.0
. Sensitivity Min. (0 dB vs 1V μbar)	: -70dB/40 ±1.0KH (R=3.9KΩ)
. Band Width (KHz) Min.	: 3.5 / (at -76 dB)
. Capacitance (PF)	: 2000 ± 30%

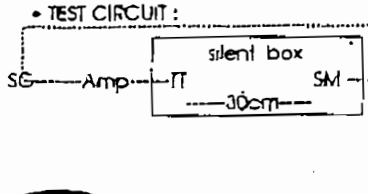
. OUTLINE DIMENSION :



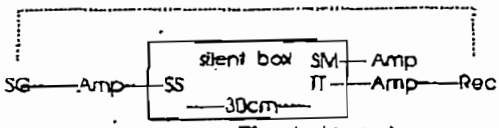
A= 9.8 ±0.2
B= 6.8 ±0.4
C= 11.5 ±0.8
D= 5.0 ±0.2

Unit:mm

• TEST CIRCUIT :



TT : test transducer
Rec : recorder
SM : std. microphone
Amp : amplifier
SG : signal generator



TT : test transducer
Rec : recorder
SM : std. microphone
Amp: amplifier
SG : signal generator
SS : std. speaker
R : resistance