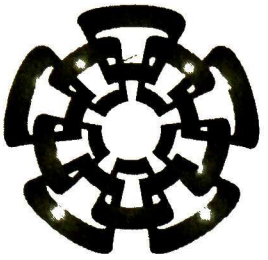
BNL-16171



CINVESTAV - IPN

Centro de Investigación y de Estudios Avanzados del IPN
Unidad Guadalajara

Unidad Guadalajara

CONTROL JERARQUICO DIFUSO:

Aplicación a la Navegación de Robots Móviles

CINVESTAV I. P. N.
SECCION DE INFORMACION
Y DOCUMENTACION

TESIS QUE PRESENTA

THAMAR ELENA MORA MONROY

PARA OBTENER EL GRADO DE

MAESTRO EN CIENCIAS

EN LA ESPECIALIDAD DE

INGENIERIA ELECTRICA

Guadalajara, Jal., Noviembre de 1998.

CLASIF.:
ADQUIS.: TESIS-1999
FECHA: 19-11-99
PROCED.: Depto Sec
$
Bibl.

CONTROL JERARQUICO DIFUSO:
Aplicación a la navegación de Robots Móviles

Tesis de Maestría en Ciencias
Ingeniería Eléctrica

Por:

Thamar Elena Mora Monroy

Ingeniero Industrial
Instituto Tecnológico de Hermosillo, 1992-1996

Becario del CONACyT, expediente No. 112937

Director de Tesis:

Dr. Edgar Nelson Sánchez Camperos

CINVESTAV del IPN Unidad Guadalajara, Noviembre de 1998.

CONTENIDO

Página

Capítulo 3. PLANEACION DE TRAYECTORIAS

Capítulo 4. CONTROL JERARQUICO DIFUSO

Capítulo 5. CONTROL DE NAVEGACION

Capítulo 6. CONCLUSIONES

Capítulo 1
INTRODUCCION

1.1 Motivación

El mundo real incluye situaciones que pueden ser impredecibles, dinámicas, ruidosas e incluso desconocidas. Esto representa un inmenso nivel de incertidumbre que incrementa el esfuerzo necesario para desarrollar control de sistemas de robots móviles.

El presente trabajo de tesis versa sobre el desarrollo de dos algoritmos de control para robots móviles: el primero de ellos aplicado a planeación de trayectorias y el segundo aplicado a navegación. Ambos controladores están basados en Lógica Difusa, y cada uno de ellos fue creado especialmente para un diseño específico de robots móviles. Se presentan resultados de simulación obtenidos por el algoritmo de planeación de trayectorias; y se anexan los resultados de implementación en el sistema físico al aplicar el algoritmo de navegación.

Una de las ventajas de los sistemas basados en Lógica Difusa es que permite que el conocimiento intuitivo de navegación basada en sonares sea fácilmente descrita por medio de términos lingüísticos. La carga computacional que implica un sistema típico de inferencia difusa no es compleja. Como resultado, los sistemas de control difusos permiten tomar decisiones en tiempo real.

1.2 Objetivos del trabajo de tesis

Desarrollar herramientas para el control de un robot móvil libres de modelo, basadas en Lógica Difusa.

Desarrollar un algoritmo de control de navegación para robots móviles compuestos por dos carros capaces de interactuar en tiempo real con un medio ambiente ya sea estático o dinámico.

Resolver tareas de planeación de trayectorias y navegación para diseños específicos de robots móviles.

1.3 Aportación del trabajo de tesis

Una aportación es la aplicación satisfactoria de curvas *"B-Splines"* cúbicas como método de planeación de trayectorias. Además se muestra que el algoritmo funciona para el caso de trayectorias cerradas.

En cuanto al problema de navegación se trata, el algoritmo de control está diseñado bajo el principio de utilización de un subsistema sensorial muy reducido (solo tres sensores de ultrasonido) y de bajo costo. Este diseño permite que el robot móvil capture suficiente información del medio ambiente de tal forma que garantice que la navegación será segura, libre de colisiones.

El desarrollo del presente trabajo de tesis impulsó el rediseño y reconstrucción de un prototipo de robot móvil en CINVESTAV Unidad Guadalajara. Se complementó el subsistema sensorial y se mejoró el procesamiento de información, entre otras modificaciones. El rediseño del sistema convierte al robot móvil en un elemento más del Laboratorio de Control Automático de nuestra Unidad Académica sobre el cual se podrán implementar otro tipo de controladores y por lo tanto será soporte como material didáctico.

1.4 Estructura de la tesis

El presente trabajo de tesis fue organizado de tal forma que conduzca al lector a través de varios capítulos especializados, de tal forma que transmitan una idea completa de un sistema robot móvil desde los elementos del diseño físico, métodos de planeación de trayectorias, estructuras de controladores jerárquicos y control de navegación.

El Capítulo 2 ofrece una descripción detallada sobre la arquitectura del diseño de un robot móvil específico compuesto por dos carros conectados por una unión giratoria. Este Capítulo cubre tanto la configuración física ("*hardware*" y "*software*") como las ecuaciones del modelo del robot móvil que se utilizan en el proceso de simulación para planeación de trayectorias.

El Capítulo 3 trata el problema de planeación de trayectorias. Por lo tanto, describe brevemente varias técnicas de representación de trayectorias y finalmente fundamenta la elección de las curvas "*B-Splines*" cúbicas como método de planeación de trayectorias. Se muestra que el algoritmo basado en "*B-Splines*" es capaz de generar las velocidades de referencia necesarias para que el robot móvil describa una trayectoria predefinida. Se anexan los resultados obtenidos en la simulación de tal algoritmo.

En el Capítulo 4 se presentan varias estructuras de control jerárquicas, así también se incluyen conceptos básicos de Lógica Difusa. Este Capítulo lleva por objetivo primeramente, dar un panorama de algunas estructuras de control jerárquico y presentar las ventajas que ofrece un controlador difuso. Las páginas de este Capítulo sirven de base para el diseño de una estructura de control jerárquica difusa útil para el problema de planeación de trayectorias.

El Capítulo 5 hace una clara descripción del problema de navegación de robots móviles; se describen varias estructuras de control de navegación y se obtiene un algoritmo de control de navegación para aplicarse sobre robots móviles de dos carros, diseño correspondiente al descrito en el Capítulo 2. Se incluyen resultados de la implementación

del algoritmo sobre el sistema físico. Los resultados obtenidos en estos experimentos de laboratorio quedaron grabados en cinta de video VHS.

Finalmente, en el Capítulo 6 se presentan las conclusiones que se obtuvieron una vez terminado el presente trabajo de tesis. Así mismo, se presentan las directivas para un trabajo futuro que permita extender y mejorar los logros de nuestra tesis. En el Anexo A se incluye un artículo presentado en un congreso internacional.

Capítulo 2
ARQUITECTURA DEL ROBOT MOVIL

2.1 Introducción

El objetivo del presente Capítulo es describir los elementos más importantes de un sistema Robot Móvil. Para llevar a cabo esta descripción se seleccionaron dos diseños de robots móviles.

Primeramente se explica la parte de obtención del modelo, y para ello se selecciona un diseño sencillo de robot móvil [Fig. 2.1]. Se muestra el desarrollo matemático que conduce a las ecuaciones dinámicas que representan el sistema del robot móvil. Este modelo matemático se utiliza en el Capítulo 5 para soportar la simulación de un algoritmo de control para seguimiento de trayectoria.

Por otra parte, para describir los principales componentes de *"hardware"* y *"software"* se selecciona el diseño de un robot móvil compuesto por dos carros conectados por una unión giratoria [Fig. 2.4]. Este diseño corresponde al robot móvil en el cual se implementa el algoritmo de navegación propuesto en el Capítulo 5 del presente trabajo de tesis.

2.2 Modelo de un Robot Móvil.

Normalmente el control de un robot móvil se aplica sobre los motores que van acoplados a las ruedas, de tal forma que transmiten el movimiento necesario para describir una trayectoria deseada.

A continuación se explica el modelo de un sistema robot móvil sencillo [Fig. 2.1], donde el subsistema mecánico del robot trabaja en base a una configuración diferencial [3],

[20] como muestra la Figura 2.8. El robot móvil tiene dos ruedas independientes colocadas paralelamente una con respecto a otra. Las velocidades de las ruedas son controladas por separado por motores eléctricos de corriente continua. De esta forma, con un control apropiado sobre cada rueda, el robot móvil puede moverse tanto hacia delante como hacia atrás; así como girar o describir un arco específico al existir una diferencia entre las velocidades de cada motor.

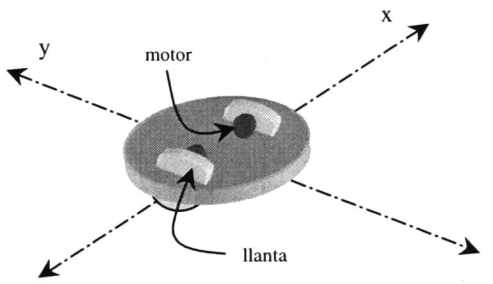


Fig. 2.1 Diseño de robot móvil sencillo.

Un motor de corriente continua (CD) trabaja básicamente bajo el principio de que la corriente que pasa a través de un campo magnético experimenta una fuerza $F=\phi \cdot i$, donde ϕ es el flujo magnético e i es la corriente que pasa por el conductor. Cada motor está formado por dos partes principales: el estator (parte fija) y el rotor que gira dentro del estator. El estator crea un flujo magnético radial ϕ y la corriente en el rotor, como consecuencia existirá un par en el rotor que provoque su giro [33]. La magnitud del par será:

$$\tau_m = K_I \phi \, i_a \qquad (2.1)$$

τ_m	Torque del motor (N-m)
ϕ	Flujo magnético (webers)
i_a	Corriente de armadura (Amps)
K_I	Constante física

Siempre que un conductor se mueve dentro de un campo magnético se genera un voltaje V_b proporcional a la velocidad del conductor en el campo. Este voltaje llamado

fuerza contraelectromotriz (FCE) se opone al flujo de corriente en el conductor. La FCE está dada por:

$$V_b = K_2 \phi \omega_m \qquad (2.2)$$

V_b	FCE (Volts)
ω_m	Vel. angular del rotor(rad/seg)
K_2	Constante de proporcionalidad

En los motores CD de magneto permanente se considera que ϕ es constante. El par del motor se controla por medio de la corriente de armadura i_a.

Considere el siguiente diagrama esquemático:

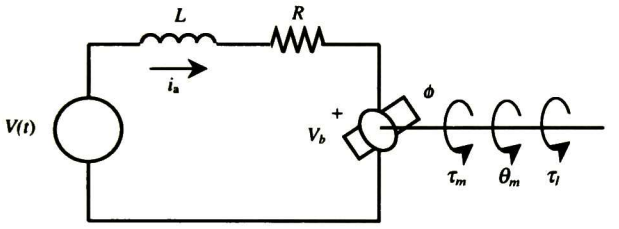


Fig. 2.2 Diagrama eléctrico de un motor CD de magneto permanente.

$V(t)$	Voltaje de armadura
L	Inductancia de armadura
R	Resistencia de armadura
V_b	Fuerza contraelectromotriz
i_a	Corriente de armadura
θ_m	Posición del rotor (radianes)
τ_m	Par generado
τ_l	Par aplicado
ϕ	Flujo magnético inducido por el estator.

La ecuación diferencial para la corriente de armadura es:

$$L\frac{di_a}{dt} + Ri_a = V - V_b \qquad (2.3)$$

Tomando en cuenta que el flujo ϕ es constante, el par desarrollado por el motor es:

$$\tau_m = K_1 \phi i_a = K_m i_a \qquad (2.4)$$

donde K_m es la constante de par en N-m/amp. A partir de la Ecuación 2.2 se tiene que:

$$V_b = K_2 \phi \omega_m = K_b \omega_m = K_b \frac{d\theta_m}{dt} \qquad (2.5)$$

donde K_b es la constante de fuerza contraelectromotriz y ω_m es la velocidad angular del rotor en rad/seg.

Es posible determinar la constante de par de un motor CD utilizando un conjunto de curvas par-velocidad para varios valores de voltaje de alimentación. Para obtener una definición completa de las condiciones de movimiento es necesario considerar los parámetros mostrados en la siguiente figura.

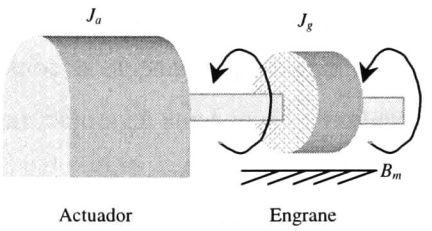


Fig. 2.3 Esquema de los parámetros mecánicos del motor.

La inercia total es igual a la suma de la inercia del actuador y la inercia del engrane o acoplador mecánico: $J_m = J_a + J_g$. El coeficiente de fricción total está dado por B_m. La descripción del sistema es:

$$J_m \frac{d^2\theta_m}{dt^2} + B_m \frac{d\theta_m}{dt} = \tau_m \tag{2.6}$$

Al aplicarle una carga resulta:

$$J_m \ddot{\theta}_m + B_m \dot{\theta} = K_m i_a - \tau_l \tag{2.7}$$

donde τ_l es el par aplicado.

En términos de velocidad angular se tiene que:

$$J_m \omega_m + B_m \omega_m = K_m i_a - \tau_l \tag{2.8}$$

Despejando τ_l resulta:

$$\tau_l = K_m i_a - J_m \omega_m - B_m \omega_m \tag{2.9}$$

Si se escoge el voltaje como la variable de control se tiene que, a partir de la derivada de:

$$i_a = \frac{\tau_l}{K_m} + J_m \frac{\dot{\omega}_m}{K_m} + B_m \frac{\omega_m}{K_m} \tag{2.10}$$

se llega a:

$$L\left(\frac{\dot{\tau}_l + J_m \ddot{\omega}_m + B_m \dot{\omega}_m}{K_m} \right) + R_e \left(\frac{\tau_l + J_m \dot{\omega}_m + B_m \omega_m}{K_m} \right) = \vec{V} - \vec{V}_b \tag{2.11}$$

Es sabido que en el caso de motores de CD de bajo voltaje con magneto permanente, los efectos de L son despreciables comparados con los de R_e y con los valores de operación de V [41]. Por esto y por la equivalencia $V_b = K_b \omega_m$, la ecuación (2.11) se reduce a:

$$\frac{R_e J_m}{K_m} \dot{\omega}_m + \frac{R_e B_m}{K_m} \omega_m + \frac{R_e}{K_m} \tau_l = \vec{V} - K_b \omega_m \tag{2.12}$$

Tomando en cuenta que $\tau_l = R^2 \gamma m \omega_m$, donde γ es una constante de distribución de masa y R es el radio de cada rueda se tiene que:

$$\begin{aligned} \omega_{m1} \left(\frac{R_e J_m}{K_m} + \frac{R_e R^2 \gamma\, m}{K_m} \right) + \omega_{m1} \left(\frac{R_e B_m}{K_m} + K_b \right) &= V_1 \\ \omega_{m2} \left(\frac{R_e J_m}{K_m} + \frac{R_e R^2 \gamma\, m}{K_m} \right) + \omega_{m2} \left(\frac{R_e B_m}{K_m} + K_b \right) &= V_2 \end{aligned} \tag{2.13}$$

Las ecuaciones (2.13) corresponden al modelo del robot móvil de la Figura 2.1. Este modelo se utiliza en el Capítulo 3 para representar a un robot móvil en la simulación de un algoritmo de planeación de trayectorias.

2.2.1 Ecuaciones de movimiento

Para el robot móvil correspondiente a la Figura 2.1, la velocidad instantánea de cada rueda está dada por:

$$\begin{aligned} v_d &= \omega_d R \\ v_i &= \omega_i R \end{aligned} \tag{2.14}$$

donde ω_d y ω_i representan la velocidad angular de la rueda derecha y rueda izquierda respectivamente. R representa el radio de la rueda.

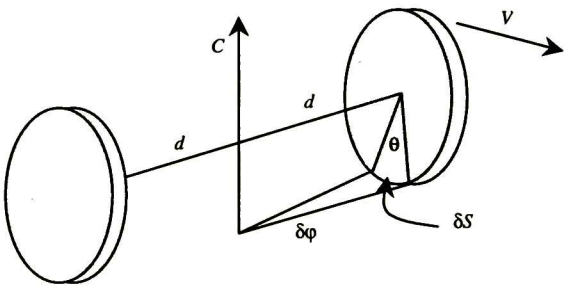


Fig. 2.4 Elementos diferenciales de la cinemática de las ruedas.

Los elementos de arco alrededor del punto central C [Fig. 2.4] son generados por los desplazamientos de la rueda:

$$\begin{aligned} \delta S_d &= d\delta\varphi \\ \delta S_i &= -d\delta\varphi \end{aligned} \tag{2.15}$$

La distancia entre las ruedas esta dada por $2d$ De estos desplazamientos se obtienen las velocidades alrededor del punto C:

$$\begin{aligned} \frac{\delta S_d}{\delta t} &= d\frac{\delta\varphi}{\delta t} \\ \frac{\delta S_i}{\delta t} &= -d\frac{\delta\varphi}{\delta t} \end{aligned} \tag{2.16}$$

que equivalen a

$$\begin{aligned} v_d &= d\varphi \\ v_i &= -d\varphi \end{aligned} \tag{2.17}$$

Igualando las ecuaciones (2.14) y (2.17) resulta:

$$\begin{aligned} \omega_d R &= d\varphi \\ \omega_i R &= -d\varphi \end{aligned} \tag{2.18}$$

Restando las ecuaciones (2.18) y despejando φ se tiene que:

$$\varphi = \frac{(\omega_d - \omega_i)R}{2d} \tag{2.19}$$

La velocidad resultante es:

$$v = \frac{v_d + v_i}{2} \tag{2.20}$$

En términos de velocidad angular:

$$v = \frac{\omega_d R + \omega_i R}{2} = \frac{(\omega_d + \omega_i)R}{2} \tag{2.21}$$

Las componentes de la velocidad son:

$$\begin{aligned} \dot{x} &= v\cos\varphi \\ \dot{y} &= v\,\mathrm{sen}\,\varphi \end{aligned} \tag{2.22}$$

o bien

$$\begin{aligned} \dot{x} &= \frac{(\omega_d + \omega_i)R}{2}\cos\varphi \\ \dot{y} &= \frac{(\omega_d + \omega_i)R}{2}\,\mathrm{sen}\,\varphi \end{aligned} \tag{2.23}$$

Debido a que las acciones de control sobre el robot móvil se llevan a cabo sobre dos motores de corriente continua independientes entre sí y acoplados a las ruedas, el modelo del sistema robot móvil de la Figura 2.1 está representado por las ecuaciones de cada motor independiente. Las características físicas de los motores se explican en la sección 2.3.1.1.1 de este Capítulo.

2.3 Configuración de un robot móvil

En esta sección se presentan los componentes más importantes del robot móvil que se utiliza para llevar a cabo la implementación del algoritmo de control de navegación propuesto en el Capítulo 5. Este robot móvil cuenta con un subsistema mecánico, un subsistema sensorial, un subsistema de cómputo y una interfaz electrónica que comunica los diferentes elementos. A diferencia del diseño de robot de la sección anterior, este diseño consta de dos carros: en el primer carro se encuentra la fase electrónica que administra la información del "*hardware*", y el segundo carro puede ser utilizado como remolque.

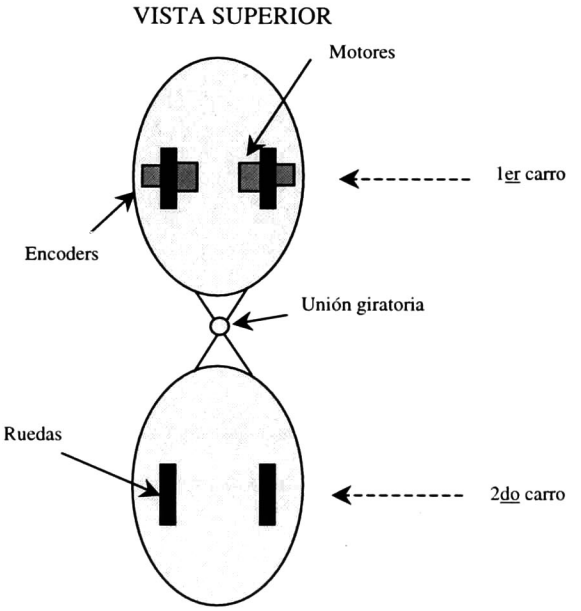
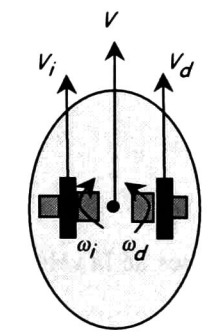


Fig. 2.5 Configuración del subsistema mecánico del robot móvil.



V Velocidad lineal
V_d Velocidad rueda derecha
V_i Velocidad rueda izquierda

ω Velocidad angular
ω_d Vel. angular rueda derecha
ωi Vel. angular rueda izquierda

Fig. 2.6 Cinemática del sistema

El sistema tiene cuatro grados de libertad, basados en las siguientes condiciones:

I. Cada rueda es libre de deslizarse únicamente sobre el plano sin resbalar.

II. Cada rueda es libre de girar solamente sobre su eje vertical.

Con relación a un sistema de coordenadas global *W*, el primer carro puede estar ubicado en cualquier posición representada por dos coordenadas, *x* y *y*, y con cualquier orientación representada por una tercer coordenada: el ángulo φ. Para poder describir el sistema completo, incluyendo el segundo carro (remolque), es necesario incluir el ángulo β que representa la posición relativa entre ambos carros [Fig. 2.10].

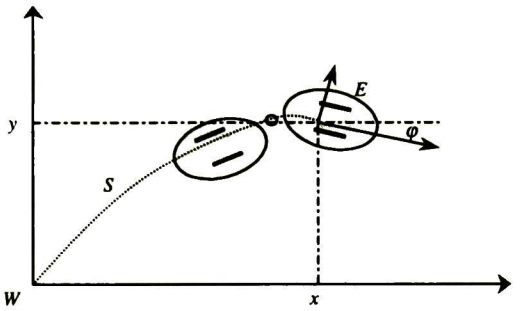


Fig. 2.7 Grados de libertad del sistema.

Estos cuatro grados de libertad (x, y, φ, β) indican la posición y orientación entre el sistema de coordenadas global *W* y el sistema de coordenadas local *E*. Con estos cuatro datos, la descripción del sistema es única.

2.3.1 Descripción del "*Hardware*"

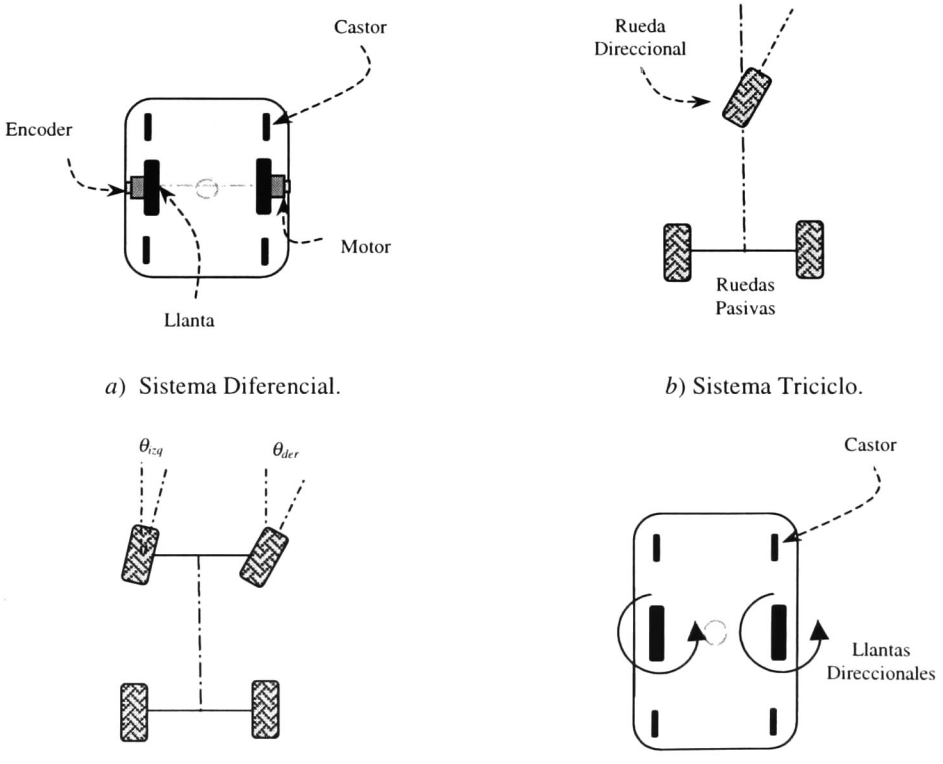
El subsistema mecánico del robot móvil descrito en esta sección está compuesto por dos motores iguales que transmiten su movimiento hacia las ruedas por medio de un acoplamiento mecánico en su flecha. Cada rueda es libre de moverse tanto hacia adelante como hacia atrás.

El subsistema sensorial esta compuesto por varios tipos de sensores: de ultrasonido, de desplazamiento rotacional, de velocidad y de nivel de voltaje; cuya tarea es capturar información del medio ambiente.

A continuación se presenta información sobre cada uno de los componentes, tanto del subsistema mecánico como del subsistema sensorial. Después se describe la interfaz electrónica que permite la comunicación entre el "*software*" y el "*hardware*" del sistema robot móvil.

2.3.1.1 Subsistema Mecánico

La precisión de las mediciones odométricas (odometría es una herramienta para medir el desplazamiento de un vehículo a lo largo de una trayectoria) es una función directa del diseño cinemático del vehículo [3]. A continuación se muestran varias configuraciones de subsistemas mecánicos comenzando por la configuración diferencial de nuestro robot móvil.



a) Sistema Diferencial.

b) Sistema Triciclo.

c) Direccionamiento Ackerman ($\theta_{izq} \neq \theta_{der}$)

d) Vehículo de Multi-grados de libertad.

Fig. 2.8 Configuraciones típicas de subsistemas mecánicos.

Además existen otros tipos de configuraciones, por ejemplo la configuración de direccionamiento omnidireccional, la configuración de tracción (oruga), la configuración "*synchro drive*", etc. Para una explicación ilustrativa y detallada consultar [3].

2.3.1.1.1 Motores

El robot móvil cuenta con dos motores de magneto permanente BARBER-COLMAN de engranaje interno, acoplados por la flecha a las ruedas del primer carro [Fig. 2.1]. Los motores tienen movimiento en ambos sentidos y trabajan como actuadores regulando las velocidades de las ruedas. Las especificaciones de los motores son las siguientes:

Voltaje de alimentación	12 Volts DC
Velocidad (sin carga)	57.22 RPM
Velocidad (máxima carga)	42.91 RPM
Consumo de corriente (sin carga)	0.158 Amps.
Consumo de corriente (máx. carga)	0.450 Amps.
Par	53.63 oz-in.

Tabla 2.1 Especificaciones de los motores.

Para que el sistema lleve a cabo un movimiento en línea recta, teóricamente debe enviarse el mismo voltaje a ambos motores puesto que éstos son iguales. Sin embargo, a pesar de que los dos motores tienen las mismas características de fabricación, su comportamiento puede variar debido a diferencias en los componentes. Otras causas pueden ser que las ruedas no se encuentren sobre la misma superficie, que alguna de ellas resbale, que las ruedas no tengan el mismo diámetro o que estén desalineadas.

La sencillez que se obtiene en el sistema del robot móvil al utilizar un sistema diferencial se ve disminuida por la complejidad que implica controlarlo. Sin embargo, normalmente es más conveniente disminuir la complejidad mecánica e incrementar la

complejidad electrónica y de *"software"* [20]. En el Capítulo 5 se expone el control de la velocidad de los motores con el fin de describir una trayectoria deseada.

2.3.1.2. Subsistema Sensorial

El robot móvil obtiene información del medio ambiente a través de sus sensores y posteriormente, por medio del *"software"* y de la fase electrónica se hace llegar la información capturada hasta el controlador. Estos sensores, también llamados transductores, son simplemente convertidores de algún fenómeno físico en señales eléctricas. En nuestro caso el subsistema sensorial está formado por cuatro tipos de sensores: ultrasonido, de velocidad, de desplazamiento rotacional y de nivel bajo de batería.

2.3.1.2.1 Sensores de velocidad

Para verificar que los motores estén girando a las velocidades de referencia (ω_i y ω_d) calculadas por los algoritmos de control, es necesario sensar la velocidad real de cada motor por separado.

Un *"encoder"* es un sensor que mide la posición o velocidad angular de una flecha. Normalmente se monta sobre la flecha de un motor, o bien, se acopla a un eje. La señal de salida de un *"encoder"* puede ser: un código que corresponde a una orientación particular de la flecha (*"encoder"* absoluto) o bien, puede ser un tren de pulsos (*"encoder"* incremental). En otras palabras, los *"encoders"* absolutos miden directamente la posición angular (en grados o radianes) e infieren la velocidad; los *"encoders"* incrementales miden la velocidad rotacional e infieren la posición [3]. A medida que gira flecha, la salida cambia de un estado alto a un estado bajo y viceversa. La proporción en que se generan los pulsos es igual a la proporción de giro de la flecha [20]. También es posible utilizar un potenciómetro como un *"encoder"* absoluto. Cada posición de la flecha produce un único

valor de resistencia. Los "*encoders*" absolutos son utilizados normalmente para determinar la posición de los brazos de robots.

Los sensores de velocidad del robot móvil son tamaño miniatura de la SERIE E11 de Dynapar. Sus características son las siguientes:

Código	Incremental
Pulsos por revolución	1024
Señal de salida	Señal cuadrada
Respuesta de frecuencia[†]	hasta 100 kHz
Cuadratura	90° ± 36°

Tabla 2.2 Características eléctricas de los sensores de velocidad.

Peso	3 oz. max.
Par inicial a 25° C	0.1 oz-in max.
Par normal a 25° C	0.06 oz-in max.
Momento de inercia	4.5×10^{-6} oz-in seg.^{2} max.
Rotación de la flecha	Continua y reversible

Tabla 2.3 Características mecánicas de los sensores de velocidad.

2.3.1.2.2 Sensores de ultrasonido

La tarea de los sensores de ultrasonido es obtener información que permita calcular la distancia a la que se encuentran los objetos, siempre y cuando estén dentro de su rango de alcance. En esencia, este tipo de sensores son transductores que convierten las señales acústicas en impulsos eléctricos. El procesamiento de información es el siguiente: cada sensor emite una señal ultrasónica, y espera su rebote (eco) al chocar con algún objeto

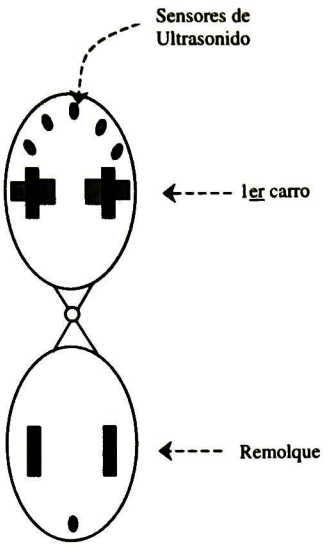
[†] *En inglés "Frequency Response"*

localizado dentro de su alcance. Se calcula el tiempo que la señal de sonido estuvo en el aire, y finalmente se calcula la distancia por medio de la siguiente fórmula:

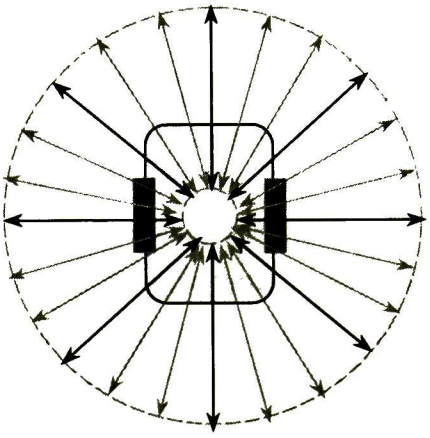
$$\text{Distancia del objeto} = \frac{\text{Velocidad del sonido} \times \text{Tiempo de vuelo de la señal}}{2}$$

Esta técnica empleada para calcular la distancia a la que se encuentran los objetos se llama Técnica por Tiempo de Vuelo (en inglés "*Time of Flight*" TOF). Debe tomarse en cuenta que la velocidad del sonido puede variar dependiendo de la temperatura ambiental presente al momento de la medición. La velocidad del sonido es proporcional a la raíz cuadrada de la temperatura expresada en grados Rankine [3]. Por lo tanto, a veces es necesario aplicar un ajuste en las lecturas sensoriales de acuerdo a la temperatura ambiental al momento de las pruebas con el robot. Esto no implica problema alguno debido a que, después de trabajar numerosas ocasiones con este tipo de sensores de ultrasonido, llegamos a la conclusión de que el comportamiento de los sensores se puede considerar lineal, aún en condiciones variantes de temperatura. En forma muy semejante los sensores tipo láser se ven afectados en sus mediciones ante variaciones de la humedad ambiental.

La distribución física de los sensores de ultrasonido es la siguiente: cinco sensores montados en el primer carro y uno montado en el remolque [Fig. 2.9]. A diferencia de otros robots móviles, nuestro prototipo con remolque tiene un número reducido de sensores, cantidad considerada suficiente para capturar la información necesaria para que el robot móvil navegue en un medio ambiente estático o dinámico. En el Capítulo 5 se muestra el resultado de un algoritmo de control de navegación propuesto basado en la lectura de sólo tres sensores, y que garantiza una trayectoria segura debido a que el subsistema sensorial toma suficiente información del medio ambiente de tal forma que evita colisiones. Otros diseños [25] utilizan al menos 24 sensores de ultrasonido.



a) Seis sensores de ultrasonido.



b) 24 sensores de ultrasonido.

Fig. 2.9 Distribución física de los sensores de ultrasonido:
a) Nuestro robot móvil, *b*) Otros robots móviles [25]

El conjunto de sensores de ultrasonido está formado por 6 sensores Herian Proffer con las siguientes características:

Voltaje de Alimentación	5-15 Volts DC
Consumo de Corriente	25 mA
Angulo de operación	$\pm$ 12°
Frecuencia	40 $\pm$ 1 Khz
Alcance	0.1-20 metros

Tabla 2.4 Especificaciones de los sensores de ultrasonido.

El cálculo de distancia con sonares es útil para desempeñar tareas de localización, detección de obstáculos, navegación por pasillos y construcción de mapas.

Aún cuando las especificaciones del fabricante indican que el alcance de cada sensor es aproximadamente hasta 20 metros, para la implementación de nuestros algoritmos de control se truncó el alcance hasta 3 metros. El acotamiento se calculó después de realizar varias pruebas. Basados en el conocimiento que se tiene sobre el comportamiento del sistema sensorial, se llegó a la conclusión de que la información capturada sobre objetos detectados en este segmento de distancia es suficiente para lograr una navegación satisfactoria. Cabe mencionar que el sistema físico es flexible para modificar este alcance según los requerimientos de información sobre el medio ambiente. Una modificación del alcance de los sensores se hace por medio de *"software"*, editando el código del microcontrolador.

La distancia mínima a la que un sensor detecta la presencia de un objeto (llamada Distancia Ciega) depende directamente del tiempo requerido para emitir la señal ultrasónica. Esto se debe a que el receptor captura la emisión de la señal como si fuera el rebote de la misma; por consecuencia durante el período de tiempo de emisión no es posible detectar objeto alguno. Para nuestra aplicación, la distancia de seguridad se determinó en 40 centímetros ya que la Distancia Ciega es de 0.371 metros.

2.3.1.2.3 Sensor de desplazamiento rotacional

Para medir la posición relativa del primer carro con respecto al segundo carro (remolque), el robot móvil tiene colocado un potenciómetro justo en la unión giratoria entre ambos carros [Fig. 2.10]. De esta forma, el potenciómetro envía un voltaje (único) proporcional al ángulo β comprendido entre ambos carros. La lectura del potenciómetro es importante para el caso en que el algoritmo de control envíe como referencia un movimiento en reversa. Para este caso en particular es necesario conocer la posición relativa, o desplazamiento rotacional, entre ambos carros.

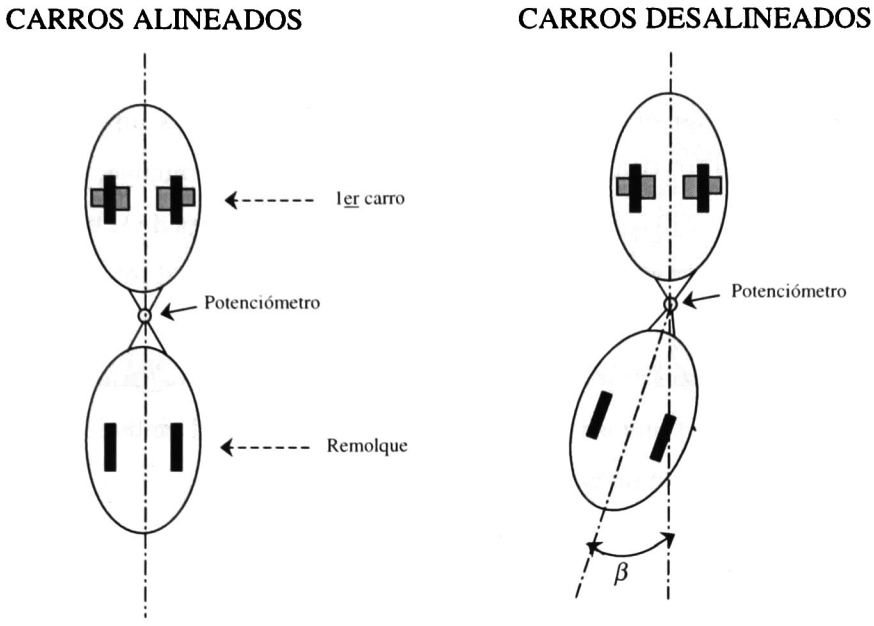


Fig 2.10 Potenciómetro: sensor de posición relativa de un carro con respecto a otro.

El potenciómetro no tiene límite de vueltas; este tipo de potenciómetro también se conoce como potenciómetro sinfin. En la etapa de Tratamiento de Señales de la sección 2.3.1.4, se ajusta la señal de salida del potenciómetro para que varíe dentro de un rango de 0 a +5 volts. Por restricciones físicas del sistema el máximo ángulo de giro del remolque es de $0 \leq \beta \leq 65°$ hacia la derecha y otro tanto hacia la izquierda (un de total 130°). Mediante un Convertidor Analógico Digital (ADC) de 8 bits se logra una resolución de 0.50° por escalón. Esta resolución es suficiente para nuestra aplicación, ya que, como se verá en el Capítulo 5, el controlador jerárquico difuso no requiere más precisión para el movimiento en reversa.

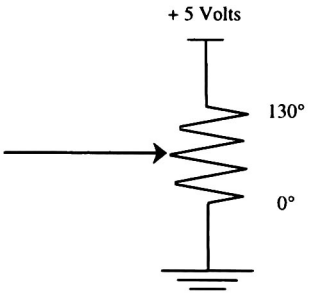


Fig. 2.11 Lectura del potenciómetro.

2.3.1.3 Alimentación del sistema

La fuente de alimentación del sistema del robot móvil es una batería recargable YUASA de +12 volts, 7.0 Amp.Hr.. Si el nivel de batería disminuye, el desempeño del sistema se ve alterado y los resultados son engañosos. La descarga de la batería no se puede evitar, pero es posible detectar un nivel bajo de voltaje. Para ello, el robot móvil cuenta con un circuito integrado que emite una señal en cuanto el nivel de voltaje es inferior a un umbral predeterminado. La señal de salida de este circuito integrado puede verificarse desde la PC en cualquier momento; además es posible cambiar el umbral que define cuándo la batería tiene insuficiente nivel de voltaje.

Los +12 volts de la batería alimentan directamente a los dos motores y a dos reguladores de los cuales se obtienen +7 volts y +5 volts respectivamente. Los sensores de ultrasonido son los únicos que reciben alimentación de +7 volts. El resto de la circuitería se alimenta con +5 volts.

Haciendo una sumatoria del consumo de corriente total del sistema físico, se obtuvo que, considerando máxima carga en los motores, el consumo del sistema no sobrepasa la cantidad de 1.5 Amperes, de los cuales 0.9 Amp. corresponden a los motores. De cualquier forma, se adaptaron dos fusibles como protección del sistema (de 1 y 0.5 Amperes).

2.3.1.4 Fase de Tratamiento de Señales

Una parte esencial del sistema de control del robot móvil es el flujo de información entre la computadora y el robot. En la computadora se encuentran codificados, en Lenguaje C, varios algoritmos de control que calculan las velocidades de referencia para cada uno de los motores. El resultado de este cálculo depende directamente de la información que sobre el medio ambiente (dinámico o estático) captura el sistema sensorial.

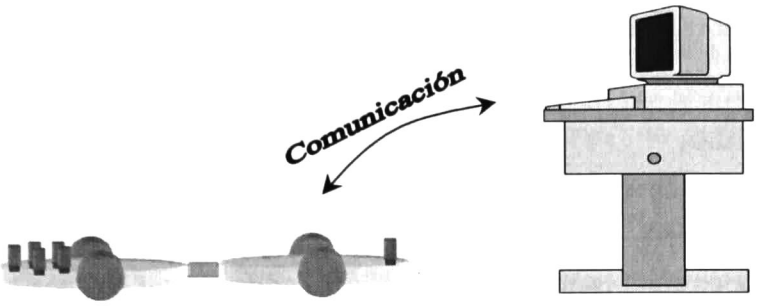


Fig. 2.12 Sistema robot móvil.

Para administrar la información del *"hardware"* y establecer la comunicación con la computadora se utiliza un microcontrolador, con su respectiva fase electrónica, montado en el robot móvil. La tarea del microcontrolador es traducir las velocidades de referencia calculadas por los algoritmos de control en la PC y hacerlas llegar a los motores. El microcontrolador debe traducir también la información capturada por los sensores (de ultrasonido, de velocidad, de desplazamiento rotacional y de nivel de voltaje) y hacerla llegar a la PC. El hecho de que el microcontrolador se haga cargo de estas tareas, ahorra trabajo y por lo tanto tiempo de procesamiento a la PC. Por esta razón se pueden obtener con mayor rapidez tanto el cálculo como la ejecución de las leyes de control del robot móvil. El microcontrolador fue programado para interpretar las lecturas de los sensores. Estos trozos de código normalmente son escritos en lenguaje ensamblador y son conocidos como "*software drivers*" Estas líneas de código representan una interfaz bien definida entre los dispositivos de *"hardware"* y la computadora.

Ahora bien, el término traducir utilizado en el párrafo anterior, implica que el microcontrolador reciba información desde una fase electrónica de tratamiento de señales, tanto digitales como analógicas y las convierta a información utilizable por la computadora. El flujo de información se muestra en la siguiente figura.

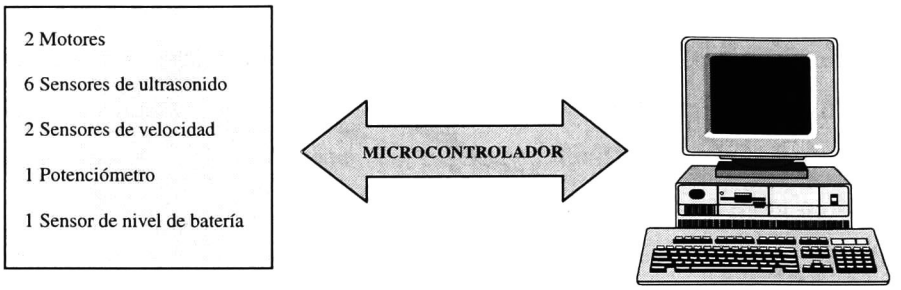


Fig. 2.13 Función del microcontrolador.

En la Figura 2.14 se presenta un diagrama a nivel macro sobre la administración de información del *"hardware"* del robot móvil.

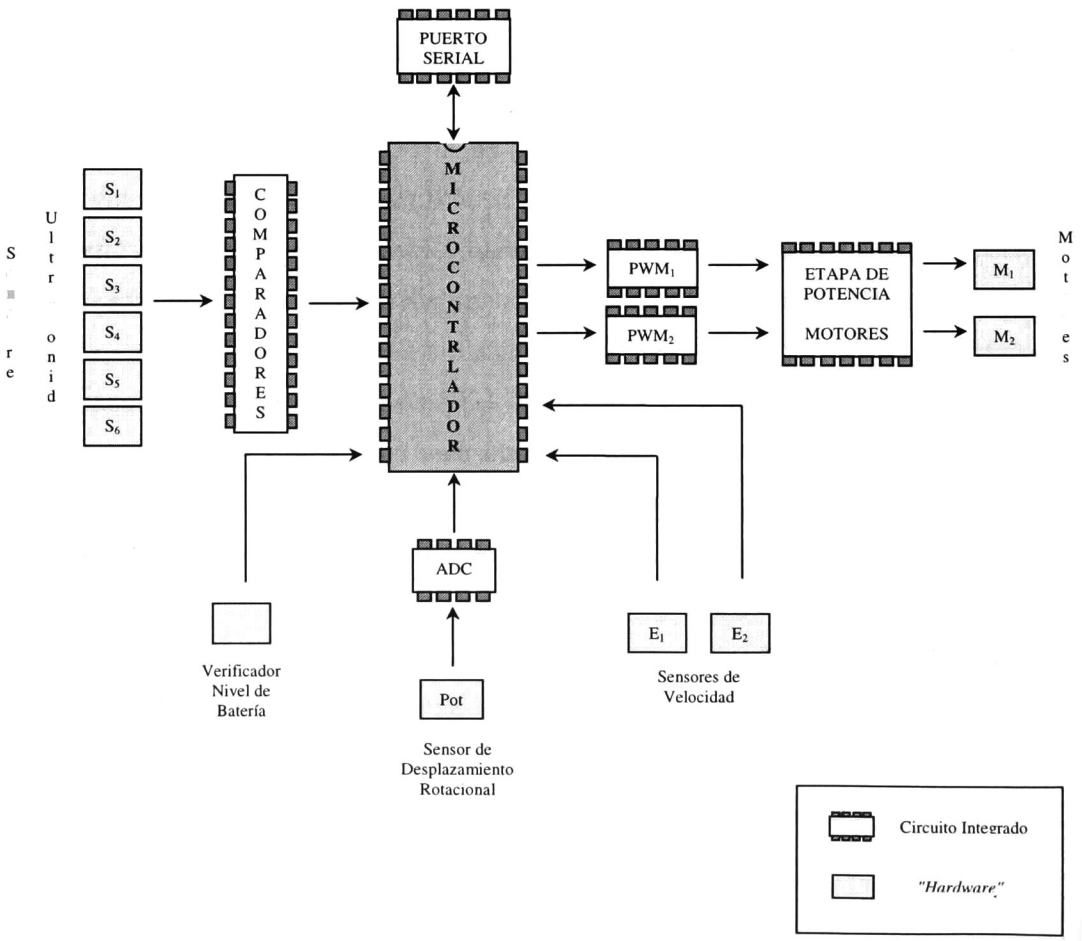


Fig. 2.14 Diagrama macro de la administración de información de "*hardware*"

2.3.2 Descripción del "*Software*"

Hablar de *"software"* en el sistema robot móvil es hablar de varios tipos de código, y por lo tanto, de varios lenguajes de programación. Existen tres partes esenciales de *"software"*: *1*) el controlador del robot móvil instalado en la computadora, 2) el administrador del *"hardware"* instalado en el sistema físico, y *3*) el código de simulación.

En los Capítulos 3 y 5 se utiliza MATLAB como lenguaje base para llevar a cabo la etapa de simulación de un algoritmo de planeación de trayectorias, así como también la herramienta MATLAB/Simulink, cuya máxima ventaja es la representación gráfica de programación, que por medio de iconos predeterminados evita la necesidad de teclear gran cantidad de líneas de código. Además permite personalizar iconos.

Para llevar a cabo la implementación de estos algoritmos de control en la computadora se hace uso de Lenguaje C. Con esta herramienta se codifican varios algoritmos de control permitiendo activar cualquiera de ellos según la elección del usuario del robot móvil. La comunicación entre la computadora y el microcontrolador instalado en el sistema físico se lleva a cabo por puerto serial a una velocidad de 19,200 Bd.

También como parte del *"software"*, el robot móvil incluye dos programas editados en lenguaje ensamblador. Uno de ellos tiene por objetivo administrar la información de los sensores de velocidad y los *"encoders"* en una primera etapa. El segundo programa corresponde al microcontrolador, responsable de la información completa del *"hardware"*.

En esta sección se describen las estructuras de *"software"* aplicadas sobre el sistema robot móvil.

2.3.2.1 Protocolo de Comunicación Computadora ↔ Robot Móvil

Como se mencionó en la Sección 2.3.4, la tarea del microcontrolador es interpretar la información obtenida del *"hardware"*, traducirla y hacerla llegar a la PC por medio de un protocolo de comunicación que ésta reconozca. De igual forma debe recibir, traducir y hacer llegar a cada motor la velocidad de referencia generada por el algoritmo de control que esté activo en la computadora.

En esencia, el microcontrolador deposita la información del *"hardware"* en registros específicos de memoria en la computadora. De forma similar, recoge las velocidades de referencia desde otros registros específicos de memoria. De esta forma, la información contenida en dichos registros es interpretable tanto por la computadora como por el microcontrolador. Parte del protocolo de comunicación que representa la información del "*hardware*" expresados en términos de Lenguaje C es el siguiente:

Motores.MotorDerecho(*X*)
Motores.MotorIzquierdo(*Y*)
Motores.Sentido(*Z*)
Datos.Sensor1
Datos.Sensor2
Datos.Sensor3
Datos.Sensor4
Datos.Sensor5
Datos.Sensor6
Datos.EncoderDerecho
Datos.EncoderIzquierdo
Datos.Potenciometro
Datos.Bateria

Los valores de *X* y de *Y* pueden variar entre [0,254] e indican el rango de operación de cada motor; 0 representa la velocidad máxima y 254 la velocidad mínima. El valor de *Z* está restringido por diseño a ser el mismo para ambos motores, ya sea hacia "*ADELANTE*"

o *"ATRÁS"*, según el movimiento deseado. Estas primeras tres instrucciones de protocolo de comunicación son actualizadas por la computadora de acuerdo a los resultados obtenidos por el algoritmo de control; posteriormente el microcontrolador envía las señales de control correspondientes a los actuadores (motores). La información capturada por las instrucciones restantes es depositada por el microcontrolador y utilizada por la computadora. El sistema inicializa los registros de memoria con valor igual a cero, con excepción del registro correspondiente al sentido de los motores, que siempre es hacia *"ADELANTE"*

El valor correspondiente a las variables restantes (Datos.Sensor1, Datos.Sensor2, ..., Datos.Sensor6, Datos.EncoderDerecho, Datos.EncoderIzquierdo, Datos.Potenciometro, Datos.Batería) es extraído del medio ambiente y depositado en registros de memoria específicos. A partir de ese momento, la información está disponible para ser consultada por el algoritmo de control desde la computadora.

La información contenida en los registros de memoria está expresada en las siguientes unidades:

Dispositivo	***Unidades***
Sensores de ultrasonido	milímetros
Sensores de velocidad	pulsos / 41.6346 ms
Sensor de desplazamiento	grados
Detector de nivel bajo de voltaje	1 = Nivel bajo
Motores	[0,254]

Tabla 2.5 Unidades en que se expresa la información de *"hardware"*

La rutina o programa del microcontrolador es secuencial; siempre está interpretando la información obtenida del *"hardware"* independientemente de que el algoritmo de control lo requiera o no. De igual forma el microcontrolador siempre verifica si la información de los registros de memoria correspondientes a las velocidades de referencia han cambiado de estado o no, para actualizarlas en los motores. El microcontrolador logra actualizar diez

veces por segundo el paquete completo de información: 6 sensores de ultrasonido, 2 *"encoders"*, un sensor de desplazamiento rotacional, un detector de nivel bajo de batería y 2 motores. El muestreo pudiera ser mucho mayor; para esta implementación en particular el proceso más lento de actualización corresponde a la lectura de los seis sensores de ultrasonido, cuya duración depende directamente del alcance (o truncamiento) que se le dé a los sonares. Es decir, si en lugar de 3 metros de alcance se definieran sólo 2 metros el muestreo sería mayor que el actual; esta modificación se lleva a cabo en el código del microcontrolador. La actualización del resto del subsistema sensorial y de los motores es mucho más rápida comparada con la de los sonares. Actualizar la información diez veces por segundo es suficiente para ejecutar las tareas que se describen en los capítulos subsecuentes; es por ello que se justifica el término Tiempo Real en la ejecución de los algoritmos del presente trabajo de tesis.

Una de las características de los sistemas en tiempo real es que su eficiencia no está solamente en función del resultado lógico de su ejecución, sino que además depende del tiempo que tarda en llegar a tales resultados. En [45] se define un sistema en tiempo real como cualquier sistema (o actividad) procesador de información el cual tiene que responder a estímulos externamente generados en un periodo de tiempo finito y especificado.

2.3.2.2 Plataforma de Control en la PC

Inicialmente, el *"software"* de control presenta el MENU PRINCIPAL:

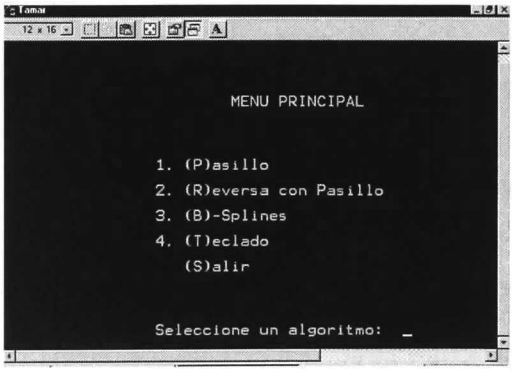


Fig. 2.15 Menú principal del *"software"* de control.

El usuario puede seleccionar cualquiera de las siguientes opciones:

- Seleccionar un algoritmo de control para implementarlo en el robot móvil, oprimiendo cualquiera de las siguientes teclas:

  "P", "*p*" ó "*1*" para activar el algoritmo de control Pasillo.
  "*R*", "*r*" ó "*2*" para activar el algoritmo de control Reversa con Pasillo.
  "*B*", "*b*" ó "*3*" para activar el algoritmo de control *"B-Splines"* *

- Desplegar la información actual del *"hardware"*, oprimiendo la tecla "*T*", "*t*" o "*4*" Además, esta opción permite controlar la velocidad de cada motor por separado en forma manual. La pantalla que aparecerá será la siguiente:

Fig. 2.16 Opción CONTROL MANUAL en el *"software"* de control.

Una vez que aparece esta pantalla, el usuario podrá controlar manualmente el robot móvil por medio del teclado.

"*o*" incrementa la velocidad del motor izquierdo.
"*i*" decrementa la velocidad del motor izquierdo.
"*w*" incrementa la velocidad del motor derecho.
"*q*" decrementa la velocidad del motor derecho.

- Salir de la plataforma de control tecleando "*S*" ó "*s*".

Las funciones básicas del *"software"* de control en Lenguaje C son funciones tales como Detener_Motores( ), Setup( ), Calibración( ), etc., incluyendo las funciones mencionadas en la sección 2.3.2.1 sobre Protocolo de Comunicación. A partir de este conjunto de funciones especializadas es posible obtener el movimiento calculado por los algoritmos de control.

A continuación se describen algunas características del *"software"* de control.

Paro de Emergencia

Todos los algoritmos de control desarrollados en esta tesis cuentan con un paro de emergencia desde el teclado. Al oprimir la tecla *"S"* o *"s"* se desactiva inmediatamente el sistema, enviando por seguridad a ambos motores velocidades de referencia iguales a cero y sentido de los motores hacia adelante.

Otra forma de detener el movimiento del robot móvil de forma inmediata es utilizar el interruptor general del sistema físico ubicado detrás de la batería, cuya tarea es impedir el paso de alimentación de la batería hacia el sistema. El inconveniente de esta segunda opción es que el microcontrolador guardará en memoria las últimas velocidades de referencia que recibió; es decir, si se restaura la alimentación del sistema el robot tenderá de forma inmediata hacia la última referencia recibida. Una forma de evitar este problema es que una vez desalimentado el sistema, se detenga la ejecución del algoritmo de control desde la computadora y se reinicialice con velocidades iguales a cero.

Exportabilidad

Es posible controlar el robot móvil desde cualquier computadora que soporte Lenguaje C y que tenga disponible un puerto serial. No es necesario llevar a cabo

---

[*] *Al momento de la impresión de la tesis, este algoritmo aún no ha sido implementado en el sistema físico por*

configuración alguna en la máquina (p.e. definición de puertos, velocidad de comunicación, etc.), puesto que esto es tarea del programa del microcontrolador y de cuatro librerías personalizadas para esta aplicación (ibmcom.h, timer.h, typedef.h, tamar.h).

Reutilización del *"software"* de control:

La plataforma de control construida en Lenguaje C puede ser reutilizada por cualquier persona cuyo objetivo sea implementar un nuevo algoritmo de control de navegación para el robot móvil. Lo único a tener en cuenta son las instrucciones del protocolo de comunicación que le permitan trasmitir las acciones de control a las ruedas del robot móvil basándose en la información capturada por el subsistema sensorial. Aún más, cualquier usuario puede crear su propio proyecto en C (project.ide) incluyendo nuevos controladores, funciones y variables; el único requisito es incluir las librerías responsables del protocolo de comunicación y respetar las instrucciones que depositan y retiran datos de los registros de memoria de la computadora.

Los algoritmos de control propuestos en el presente trabajo de tesis no son indispensables para el buen funcionamiento de un nuevo algoritmo de control.

---

problemas mecánicos, solamente se muestran los resultados del proceso de simulación en el Capítulo 3.

Capítulo 3
PLANEACION DE TRAYECTORIAS

3.1 Introducción

En este Capítulo se introducen varios métodos existentes de obtención de una trayectoria suave y continua a partir de una secuencia de puntos en un plano. Después de comparar los diferentes métodos se fundamenta la elección de uno de ellos con el fin de que sea el generador de velocidades de referencia para un robot móvil de configuración sencilla correspondiente a la Figura 2.1. Se anexan los resultados obtenidos en simulación para la suavización de curvas por este método.

En el Capítulo 5 se propone un algoritmo de control de velocidad de motores cuyo objetivo es lograr que el robot móvil ejecute las velocidades de referencia que genere el algoritmo de planeación de trayectorias que se selecciona en este Capítulo.

3.2 Métodos de obtención de una trayectoria a partir de una secuencia de puntos en un plano.

Existen muchas formas de guiar el movimiento de un robot móvil. La trayectoria a seguir puede determinarse por medio de marcas físicas ("*landmarks*"), que el robot móvil debe ubicar y seguir sin necesidad de tomar decisiones; lo cual resta autonomía. Otra opción es definir una trayectoria mediante una secuencia de movimientos en términos de posición, velocidad y/o aceleración. Otra posibilidad es que el robot móvil navegue sin un plan previo y quizá añadir tareas como evadir obstáculos, seguir una pared como referencia, etc.

Un problema básico en robótica es la planeación de movimientos con el fin de resolver una tarea específica, por ejemplo definir una trayectoria, y después controlar al

robot para que ejecute los movimientos necesarios para realizar dicha tarea. El proceso de convertir la descripción de una trayectoria en una secuencia de posiciones, velocidades y aceleraciones se denomina planeación de trayectorias. El generador de la trayectoria debe tomar en cuenta que las posiciones, velocidades y/o aceleraciones calculadas deben estar dentro del rango físicamente realizable por el sistema.

En la práctica es frecuente no lograr la trayectoria deseada aún cuando los motores ejecuten los valores de referencia. Esto se debe a efectos acumulativos de perturbaciones impredecibles que surgen por imprecisiones en el modelado, por limitaciones en la exactitud computacional, o bien, por efectos mecánicos como fricción o vibración. Una solución es medir el movimiento actual, calcular el error con respecto al movimiento deseado y modificar las acciones de control con el fin de reducir estos errores. A esto se le llama control por retroalimentación.

Generalmente los mapas utilizados para navegación son de dos tipos: mapas geométricos y mapas topológicos. Los mapas geométricos representan el medio ambiente en un sistema de coordenadas global, mientras que los mapas topológicos lo representan como una red de nodos y arcos [3].

La trayectoria deseada se puede especificar por medio de una serie de puntos, situados en la superficie de movimiento del robot, como se muestra en la Figura 3.1.

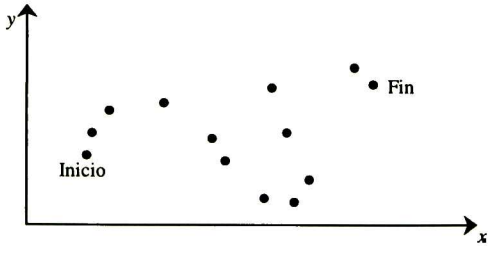


Fig3.1 Trayectoria expresada por puntos en un plano.

Cuando la trayectoria se obtiene a partir de una secuencia de puntos en un plano, deben tomarse en cuenta dos aspectos:

❶ Es necesario estimar puntos intermedios para lograr un movimiento ininterrumpido (continuidad).

❷ Es probable que la trayectoria incluya cambios abruptos de dirección que no sean físicamente realizables por el robot móvil; por lo que es necesario suavizar la trayectoria (suavidad).

Existen varios métodos para la representación continua de una secuencia de puntos, por ejemplo: los métodos de ajuste de curvas, las curvas "*Bézier Splines*", las curvas "*B-Spline*" Racionales No Uniformes (NURBS: "*Non-Uniform Rational B-Splines*"), las curvas "*B-Splines*", las curvas "*C-Splines*", la Interpolación de Hermite, etc.. A continuación se presenta una breve explicación de algunas de estas técnicas. Posteriormente se justifica la elección de una de ellas para la generación de una trayectoria suave e ininterrumpida (propiedades ❶ ,❷) a partir de un conjunto de puntos en el plano de movimiento de un robot móvil.

3.2.1 Ajuste de curvas

Entre los métodos de ajuste de curvas figuran dos esquemas generales: la regresión por mínimos cuadrados y la interpolación [8].

Cuando los datos muestran un grado significativo de error o ruido, la estrategia es ajustar una curva simple que represente el comportamiento general de los datos. Debido a que cada punto individualmente puede ser incorrecto, no es necesario ajustar una curva que pase exactamente por cada uno de ellos. En vez de esto, la curva se diseña de tal manera que siga un patrón sobre los puntos tomados como un todo. Es necesario tomar en cuenta un criterio que cuantifique la calidad del ajuste de una curva al conjunto de puntos, como por ejemplo, obtener una curva que minimice la diferencia entre los datos y la curva. Existen diferentes técnicas de Regresión por Mínimos Cuadrados: regresión lineal (ajuste de la mejor recta), regresión polinomial (ajuste del mejor polinomio), y regresión lineal

múltiple. Como el método utilizado en este trabajo de tesis no se basa en el procedimiento de Regresión por Mínimos Cuadrados, no se dan más detalles de esta técnica. Este método se encuentra bien documentado en [22].

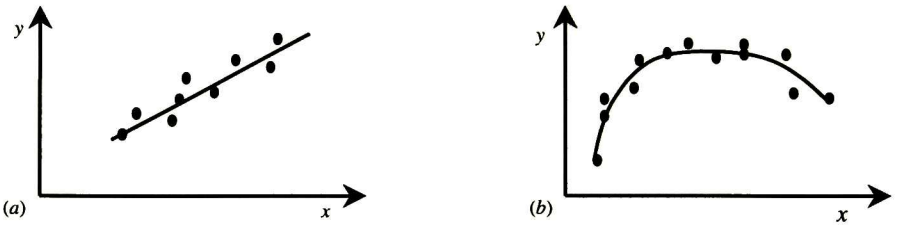


Fig. 3.2 Ejemplos de Regresión: a) lineal, b) polinomial.

A la estimación de valores entre los puntos dados, se le conoce con el nombre de Interpolación. El polinomio de interpolación consiste en determinar el único polinomio de n-ésimo orden que se ajusta a los n+1 puntos conocidos. Este polinomio proporciona una fórmula para calcular los valores intermedios. Aunque existe uno y sólo un polinomio de n-ésimo orden que se ajusta a los n+1 puntos, puede existir una gran variedad de fórmulas matemáticas para expresar este polinomio.

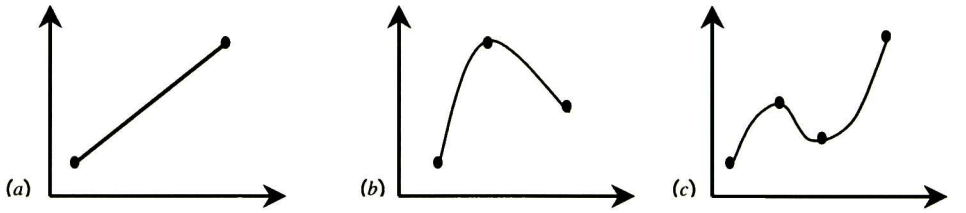


Fig. 3.3 Interpolación polinomial: *a*) primer orden (lineal),
b) segundo orden (cuadrática o parabólica) y *c*) tercer orden (cúbico).

El polinomino de interpolación de Newton (o polinomio de interpolación con diferencias divididas de Newton) es preferible cuando se desconoce el orden correcto del polinomio. Suponiendo que el polinomio de n-ésimo orden sea:

$$f_n(x) = b_0 + b_1(x - x_0) + \ldots + b_n(x - x_0)(x - x_1)\ldots(x - x_{n-1}) \qquad (3.1)$$

El polinomio de interpolación será:

$$f_n(x) = f(x_0) + (x - x_0) f[x_1 - x_0] + (x - x_0)[x - x_1] f[x_2, x_1, x_0] + \ldots + (x - x_0)(x - x_1)..(x - x_{n-1}) f[x_n, x_{n-1}, \ldots, x_0] \quad (3.2)$$

donde $f[x_i, \ldots, x_0]$ es la i-ésima diferencia dividida finita, x_i representa uno de los puntos, b_i son coeficientes en función de los valores de x_i, con $i = 0, \ldots, n$. Debe notarse que la estructura de esta ecuación es similar a la expansión de la serie de Taylor en el sentido de que los términos agregados secuencialmente consideran el comportamiento de orden superior de la función representada. Estos términos son diferencias divididas finitas y, por lo tanto, representan aproximaciones a las derivadas de orden superior. Al igual que en la series de Taylor, se puede obtener una formulación del error de truncamiento.

El polinomio de interpolación de Lagrange tiene algunas ventajas cuando el orden del polinomio se conoce de antemano. Es una reformulación del polinomio de Newton que evita los cálculos de las diferencias divididas. El polinomio de Lagrange se puede expresar como:

$$f_n(x) = \sum_{i=0}^{n} L_i(x) f(x_i) \quad (3.3)$$

donde

$$L_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j} \quad (3.4)$$

Así como para el caso del polinomio de Newton, para la versión de Lagrange se puede obtener una formulación del error de truncamiento.

Otra técnica es la Interpolación Segmentaria (en inglés "*spline*") que ajusta los datos a polinomios en intervalos. A diferencia de las técnicas anteriores, se aplican polinomios de orden inferior a los subconjuntos de datos. Estos polinomios conectados se llaman funciones de interpolación segmentaria (en inglés, *"spline functions"*). Estas funciones

tienen la propiedad adicional de que las conexiones entre las ecuaciones adyacentes son suaves. De ahí que sea particularmente útil cuando se ajustan datos que muestran cambios locales abruptos.

Por ejemplo, el objetivo de la interpolación cúbica segmentaria es obtener polinomios de tercer orden para cada uno de los intervalos entre cada pareja de puntos, de la forma:

$$f_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i \tag{3.5}$$

Las ecuaciones cúbicas para cada intervalo son:

$$f_i(x) = \frac{f''(x_{i-1})}{6(x_i - x_{i-1})}(x_i - x)^3 + \frac{f''(x_i)}{6(x_i - x_{i-1})}(x - x_{i-1})^3$$

$$+\left[\frac{f(x_{i-1})}{x_i - x_{i-1}} - \frac{f''(x_{i-1})(x_i - x_{i-1})}{6}\right](x_i - x) \tag{3.6}$$

$$+\left[\frac{f(x_i)}{x_i - x_{i-1}} - \frac{f''(x_{i-1})(x_i - x_{i-1})}{6}\right](x - x_{i-1})$$

donde $f''(x_i)$ es el valor de la segunda derivada en el nodo x dentro del i-ésimo intervalo.

Esta ecuación contiene dos incógnitas que son las segundas derivadas evaluadas al final de cada intervalo. Para determinar estas incógnitas, se debe utilizar la siguiente ecuación:

$$(x_i x_{i-1})f''(x_{i-1}) + 2(x_{i+1} - x_{i-1})f''(x_i) + (x_{i+1} - x_i)f''(x_{i+1})$$
$$= \frac{6}{(x_{i+1} - x_i)}[f(x_{i+1}) - f(x_i)] + \frac{6}{(x_i - x_{i-1})}[f(x_{i-1}) - f(x_i)] \tag{3.7}$$

El sistema de ecuaciones a resolver queda en forma tridiagonal, por lo que resta aplicar un algoritmo adicional para solucionar las n-1 ecuaciones simultáneas.

3.2.2 Curvas *"B-Splines"*

Para hacer una descripción del método de curvas *"B-Spline"*, se tomó como principal referencia a [41], [44].

Se asigna el término *"spline"* a un curvígrafo que se utiliza para generar curvas suaves que pasen exactamente por un conjunto determinado de puntos. La suavidad deseada de la curva puede lograrse a través de los pesos que se le asignen [Fig. 3.4].

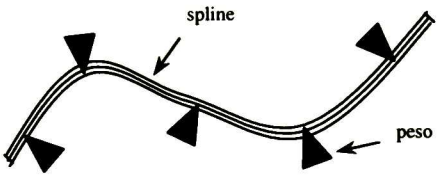


Fig. 3.4 Un spline y sus pesos.

Definición 3.1: Una función *"spline"* (polinomial) de orden M y de grado m = M-1, para una secuencia creciente de números reales

$$x_{-1} < x_0 < x_1 < \ldots < x_{n-1} < x_n < x_{n+1}$$

como nudos, es una función S(x) que satisface las siguientes dos condiciones [44]:

❶ S(x) es un polinomio de grado m o menor, en cada intervalo (x_{i-1}, x_i) $(i = 0,1,2,\ldots,n+1; x_{-1} = -\infty, x_{n+1} = +\infty)$.

❷ S (x) y sus primera, segunda, ..., (m-1)-ésima derivadas son continuas en (x_{-1}, x_{n+1}).

De esta forma, una función *"spline"* está compuesta por polinomios definidos por separado en intervalos pequeños unidos tan suavemente como sea posible. La (m-1)-ésima derivada de una función *"spline"* de grado m y todas aquellas derivadas de orden menor son continuas; la m-ésima derivada es una función escalón. En esta tesis, a las funciones

"*spline*" se les llama simplemente "*splines*". La derivada de un "*spline*" es también un "*spline*", de un grado menor, con los mismos pesos.

Una curva "*B-Spline*" ("*Spline*" Básico), o "*spline*" fundamental de grado 2, es una función definida por las siguientes ecuaciones:

$x_j \le x \le x_{j+1}$

$$S(x) = \frac{1}{(x_{j+1} - x_j)(x_{j+2} - x_j)}(x - x_j)^2 \tag{3.8}$$

$x_{j+1} \le x \le x_{j+2}$

$$S(x) = \frac{1}{(x_{j+1} - x_j)(x_{j+2} - x_j)}\left[(x - x_j)^2 - \frac{(x_{j+2} - x_j)(x_{j+3} - x_j)}{(x_{j+2} - x_{j+1})(x_{j+3} - x_{j+1})}(x - x_{j+1})^2\right] \tag{3.9}$$

$x_{j+2} \le x \le x_{j+3}$

$$S(x) = \frac{1}{(x_{j+3} - x_{j+1})(x_{j+3} - x_{j+2})}(x - x_{j+3})^2 \tag{3.10}$$

donde $j = i\text{-}1, \ldots, i\text{+}2$.

Curvas *"B-Splines"* Cúbicas

A partir de los puntos $Q_0, Q_1, \ldots, Q_n$ que definen una trayectoria en un plano [Fig. 3.5], se consideran las siguientes (n-2) combinaciones lineales formadas por cuatro puntos sucesivos:

$$\begin{aligned} &P_i(t) = X_0(t)Q_{i-1} + X_1(t)Q_i + X_2(t)Q_{i+1} + X_3(t)Q_{i+2} \\ &(i = 1,2,\ldots,n-2) \end{aligned} \tag{3.11}$$

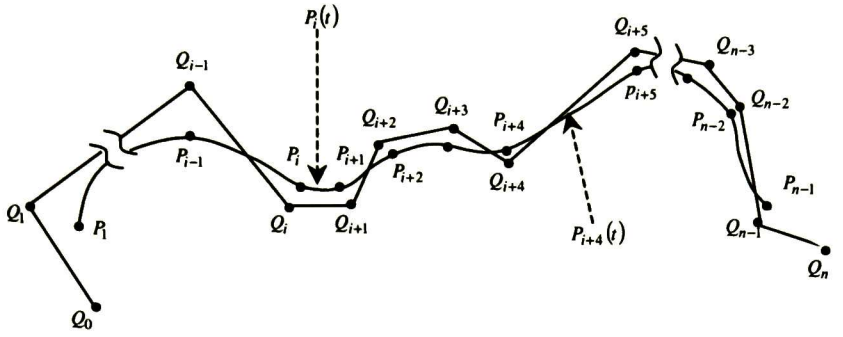


Fig. 3.5 Generación de un "*B-Spline*" Cúbico.

donde $X_0(t), X_1(t), X_2(t)$ y $X_3(t)$ son polinomios con parámetro $t(0 \le t \le 1)$. $P_i(t)$ es un segmento de curva generado al variar el parámetro *t*. La condición de continuidad entre dos segmentos contiguos $P_i(t)$ y $P_{i+1}(t)$ en *t*=0 para el segmento de curva *i*, y en *t*=1 para el segmento *i+1*, está dada por:

$$\begin{aligned} X_0(1) &= X_3(0) = 0 \\ X_1(1) &= X_0(0) \\ X_2(1) &= X_1(0) \\ X_3(1) &= X_2(0) \end{aligned} \tag{3.12}$$

Las condiciones para que la primera y segunda derivada sean continuas en todo Q_j (*j*= *i*-1, *i*, ..., *i*+2), es decir, que se cumpla que $\dot{P}_i(1) = \dot{P}_{i+1}(0)$ y que $\ddot{P}_i(1) = \ddot{P}_{i+1}(0)$, son:

$$\begin{aligned} \dot{X}_0(1) &= \dot{X}_3(0) = 0 \\ \dot{X}_1(1) &= \dot{X}_0(0) \\ \dot{X}_2(1) &= \dot{X}_1(0) \\ \dot{X}_3(1) &= \dot{X}_2(0) \end{aligned} \tag{3.13}$$

$$\begin{aligned} \dot{X}_0(1) &= \dot{X}_3(0) = 0 \\ \dot{X}_1(1) &= \dot{X}_0(0) \\ \dot{X}_2(1) &= \dot{X}_1(0) \\ \dot{X}_3(1) &= \dot{X}_2(0) \end{aligned} \tag{3.14}$$

Además, la condición de invariancia bajo una transformación de coordenadas (Condición de Cauchy) se define como:

$$X_0(t) + X_1(t) + X_2(t) + X_3(t) \equiv 1 \tag{3.15}$$

Si se definen $X_0(t), X_1(t), X_2(t)$ y $X_3(t)$ como polinomios cúbicos y si se considera la simetría de las ecuaciones (3.12), (3.13) y (3.14), las funciones $X_0(t),\ldots, X_3(t)$ pueden definirse como:

$$\begin{aligned} X_0(t) &= X_3(1-t) \\ X_1(t) &= X_2(1-t) \\ X_2(t) &= a_1 t^3 + b_1 t^2 + c_1 t + d_1 \\ X_3(t) &= a_2 t^3 + b_2 t^2 + c_2 t + d_2 \end{aligned} \tag{3.16}$$

A partir de la primera condición de (3.12), (3.13) y (3.14) se tiene que:

$$b_2 = c_2 = d_2 = 0$$

De la cuarta condición de (3.12), (3.13) y (3.14) resulta:

$$b_1 = 3a_2, c_1 = 3a_2, d_1 = a_2$$

De los polinomios $X_0(t)$ y $X_1(t)$, y de la condición de Cauchy, se obtiene:

$$a_1 = -3a_2$$

$$a_2 = \frac{1}{6}$$

A partir de estos resultados se llega a que los cuatro polinomios $X_0(t), X_1(t), X_2(t)$ y $X_3(t)$ de la Ecuación 3.11 son:

$$\begin{aligned} X_0(t) &= \tfrac{1}{6}(1-t)^3 \\ X_1(t) &= \frac{t^3}{2} - t^2 + \frac{2}{3} \\ X_2(t) &= -\frac{t^3}{2} + \frac{t^2}{2} + \frac{t}{2} + \frac{1}{6} \\ X_3(t) &= \frac{t^3}{6} \end{aligned} \tag{3.17}$$

3.2.3 Curvas "*Bézier Splines*"

Las curvas "*Bézier Splines*" [41] son curvas paramétricas generadas por un polígono de control (un polígono de control es el conjunto de puntos que describen una trayectoria en un plano como se muestra en la Figura 3.1). La trayectoria es generada en su totalidad al variar el parámetro t de los polinomios entre 0 y 1. En otras palabras, el grado del polinomio de la curva *"Bézier Spline"* depende del número de puntos de control del polígono. Si por ejemplo el polígono de control tiene n lados (un lado es la recta que une dos puntos de control consecutivos), entonces el polinomio de *Bézier* correspondiente será de grado n. Las curvas *"Bézier Splines"* tienen la propiedad de coincidir con los puntos extremos del polígono de control, por lo tanto la curva tendrá la misma pendiente que el polígono en esos puntos. Analíticamente una *"Bézier Spline"* puede expresarse de la siguiente forma:

$$P(t) = \sum_{i=0}^{n} \binom{n}{i} (1-t)^{n-i} t^i Q \tag{3.18}$$

que contiene la expresión binomial de $[(1-t)+t]^n$ La desventaja de este método es que el grado del polinomio depende directamente del grado del polígono generador.

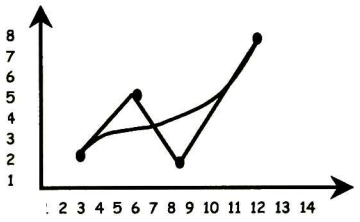


Fig. 3.6 Curva Bézier generada a partir de un polígono de orden 4.

Finalmente, según [8], [31] y [44], se llegó a la conclusión de que el método de representación de una secuencia de puntos por Curvas *"B-Splines"* cúbicas se logra una aproximación de la secuencia de puntos siempre suave, continua y de fácil manejo. Aún cuando algunos de los métodos aquí expuestos tienen un algoritmo de solución sencillo, las Curvas *"B-Splines"* cúbicas tienen un algoritmo de complejidad computacional mucho menor comparada con los demás.

Por otra parte, entre las curvas *"B-Splines"*, se realizaron pruebas con polinomios cuadráticos y polinomios cúbicos para seleccionar aquellos polinomios que arrojaran el menor error con respecto a la trayectoria propuesta. Las pruebas consistieron en proponer trayectorias abiertas y cerradas que incluyeran cambios abruptos y suaves. Los resultados señalaron que por lo general los polinomios cúbicos eran los que minimizaban mejor el error.

A continuación se explica el algoritmo de planeación de trayectorias que genera velocidades de referencia con el fin de guiar a un robot móvil a través de una trayectoria deseada. Esta trayectoria, inicialmente es expresada por medio de una secuencia de puntos en un plano, quizá óptima en algún sentido. Así pues, las Curvas *"B-Splines"* con polinomios cúbicos es la herramienta a utilizar debido a su sencillez y ventajas computacionales.

3.3 Algoritmo para generar una trayectoria suave y continua por medio de curvas *"B-Spline"*.

Como se menciona en el Capítulo 2, el control del sistema del robot móvil se ejerce sobre dos motores acoplados a las ruedas del primer carro. Esto implica que el algoritmo de planeación de trayectorias debe generar velocidades angulares de referencia, tanto para la rueda derecha, como para la rueda izquierda (ω_d y ω_i respectivamente). A continuación se presenta el algoritmo de planeación de trayectorias por Curvas *"B-Splines"*, y posteriormente se muestran los resultados obtenidos del proceso de simulación efectuado en MATLAB.

3.3.1 Planeación de trayectorias para un robot móvil por medio de "*B-Splines*".

$\dot{X}_0, \dot{X}_1, \dot{X}_2$ y $\dot{X}_3$ son la primera derivada de los polinomios $X_0(t), X_1(t), X_2(t)$ y $X_3(t)$ [Ec. (3.17)].

$\dot{x}$ y $\dot{y}$ son la representación escalar de $P_i(t)$ [Ec. 3.11].

El ciclo *i* se itera con paso de 1.

El ciclo *t* se itera con paso de 0.01.

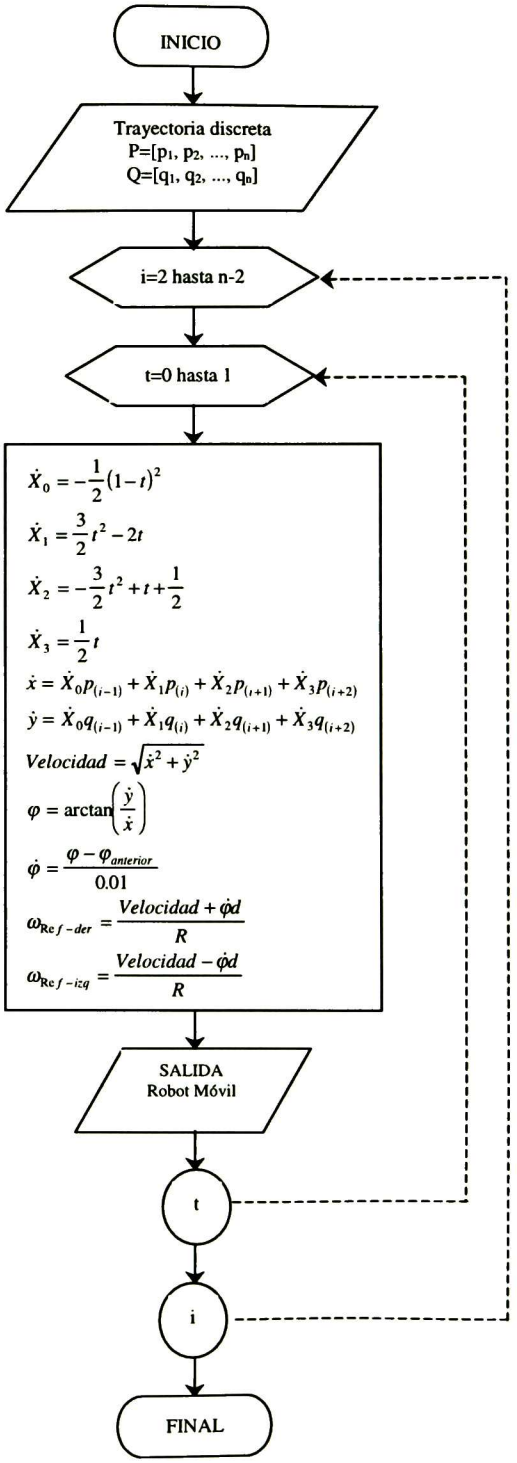


Fig. 3.7 Algoritmo de planeación de trayectorias en términos de velocidad angular.

3.3.2 Resultados de Simulación

El algoritmo para generar una trayectoria suave e ininterrumpida a partir de una secuencia de puntos en un plano que se propone en esta tesis fue simulado en MATLAB. A continuación se presentan dos resultados ilustrativos.

Ejemplo 3.1: Trayectoria Abierta

La figura 3.8 muestra la distribución del lugar donde se encuentra el robot móvil. La línea punteada indica la trayectoria deseada: iniciar el recorrido a partir de ★ y finalizar en ✸.

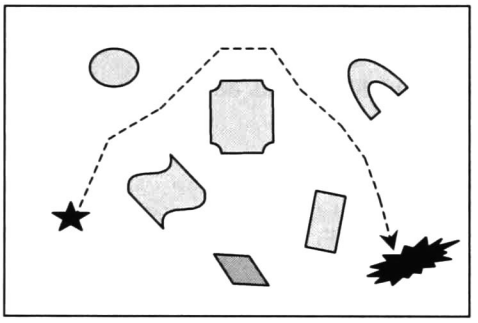


Fig. 3.8 Distribución física y descripción del movimiento deseado.

La descripción del movimiento deseado, expresado por medio de puntos en un plano, está contenida en la siguiente gráfica:

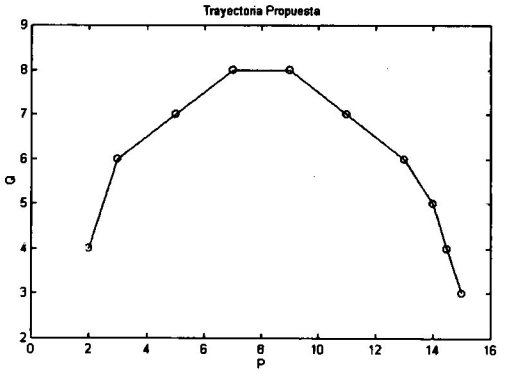


Fig. 3.9 Representación de la trayectoria

Al aplicar el algoritmo de la sección 3.3.1, se obtiene una curva suave que cumple las condiciones de continuidad (3.12), (3.13) y (3.14).

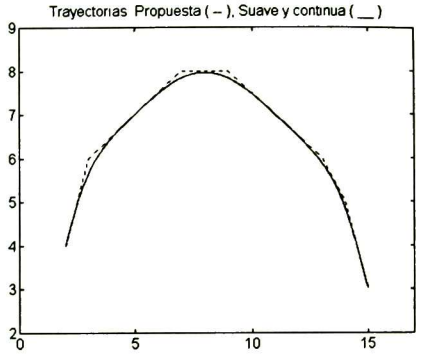
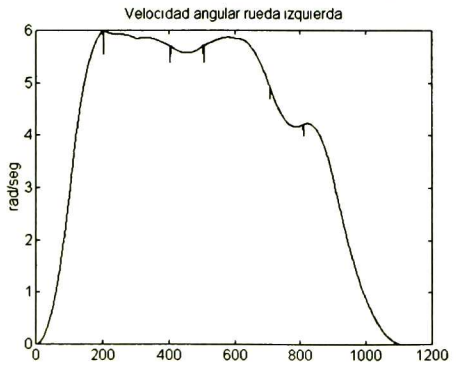


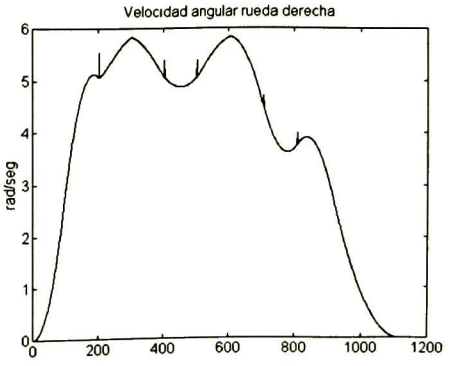
Fig. 3.10 Trayectoria suavizada por "*B-Splines*"

El método de *"B-Splines"* aplicado en el presente trabajo de tesis está basado en polinomios de orden 4 (combinación lineal de cuatro puntos consecutivos) y de grado 3 (polinomios cúbicos).

Debe tomarse en cuenta que, de acuerdo a la ecuación (3.11), solamente se generarán n-3 segmentos de curva suave (n es el número de puntos que describen la trayectoria inicial). Si se desea que la curva generada incluya punto inicial y el punto final de la trayectoria, sólo es necesario repetir los puntos en ambos extremos.

Es importante que las velocidades angulares que genera el algoritmo sean compatibles con el rango que en realidad pueden ejecutar los motores. Sintetizando, la esencia del algoritmo radica en respetar la proporción entre las velocidades generadas. Siempre y cuando se respete esta proporción, será posible escalar las velocidades de referencia de tal forma que sean compatibles con los rangos de los motores (para mayor detalle de "*hardware*" consultar el Capítulo 2).





El número de velocidades de referencia que se obtienen una vez efectuado el proceso de simulación depende directamente del número de puntos n que definen la trayectoria inicial, así como también del paso que se utilice al iterar el parámetro t entre $0<t<1$ [Fig. 3.7].

De acuerdo a las velocidades angulares de referencia que se calcularon, la velocidad lineal del robot móvil será:

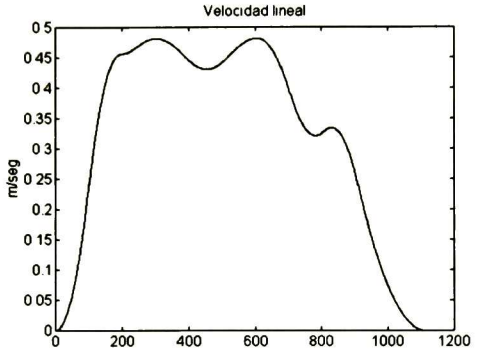


Fig. 3.12 Velocidad Lineal del robot móvil.

Para comprobar si el algoritmo genera velocidades de referencia que conduzcan a la trayectoria deseada, se procede a reconstruir la trayectoria a partir de las velocidades angulares calculadas. La fórmula de reconstrucción de trayectoria es:

$$x = \int \frac{(\omega_d + \omega_i)}{2} R\cos\varphi$$
$$y = \int \frac{(\omega_d + \omega_i)}{2} R\,\mathrm{sen}\,\varphi \qquad (3.19)$$

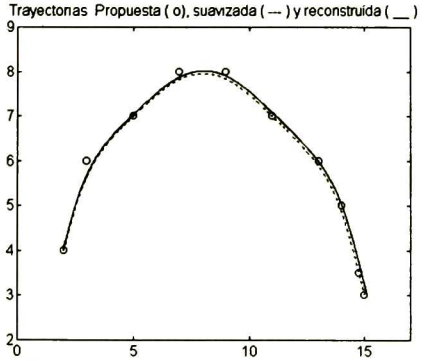


Fig. 3.13 Comparación de trayectorias.

Como puede observarse, las pruebas de simulación muestran que el algoritmo desarrollado en este Capítulo es adecuado para la planeación de trayectorias de un robot móvil de configuración sencilla [Fig. 2.1]. Durante las pruebas a las que se sometió el algoritmo, se incluyeron patrones de referencia para casos en los que la abscisa de los puntos que definen la trayectoria deseada no tiene un comportamiento necesariamente creciente. El siguiente ejemplo muestra lo anterior.

Ejemplo 3.2 Trayectoria Cerrada

La figura 3.14 muestra la distribución de un lugar en el que se encuentra el robot móvil (o vehículo autoguiado). La línea punteada indica la trayectoria deseada: iniciar en el punto donde se encuentra el vehículo, hacer un recorrido por diferentes puntos de la ciudad, y finalizar exactamente en el punto inicial.

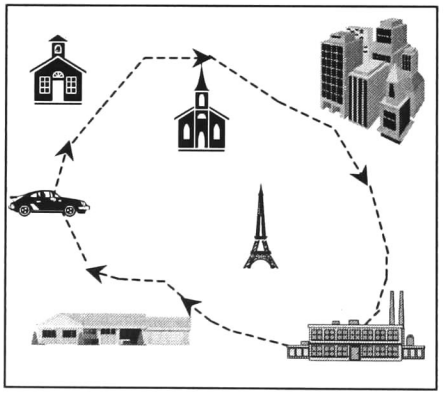


Fig. 3.14 Distribución física y descripción del movimiento deseado.

La descripción del movimiento deseado, expresada por puntos en el plano, está contenida en la siguiente gráfica:

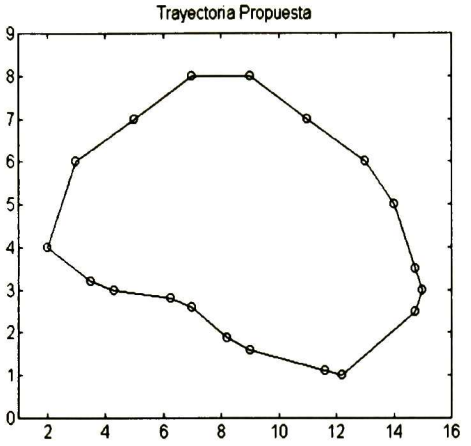


Fig. 3.15 Representación de la trayectoria

Al aplicar el algoritmo de la sección 3.3.1 se obtiene una curva suave que cumple las condiciones de continuidad (3.12), (3.13) y (3.14).

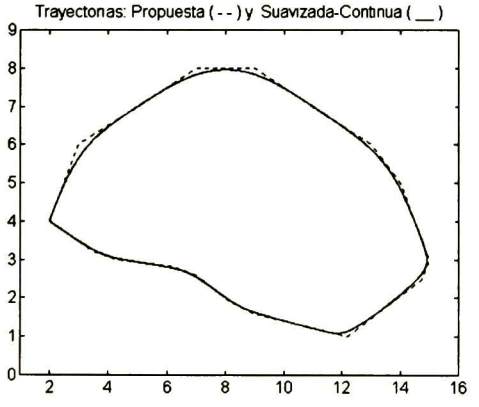


Fig. 3.16 Trayectoria suavizada por *"B-Splines"*

Las velocidades angulares de referencia calculadas para cada uno de los motores son:

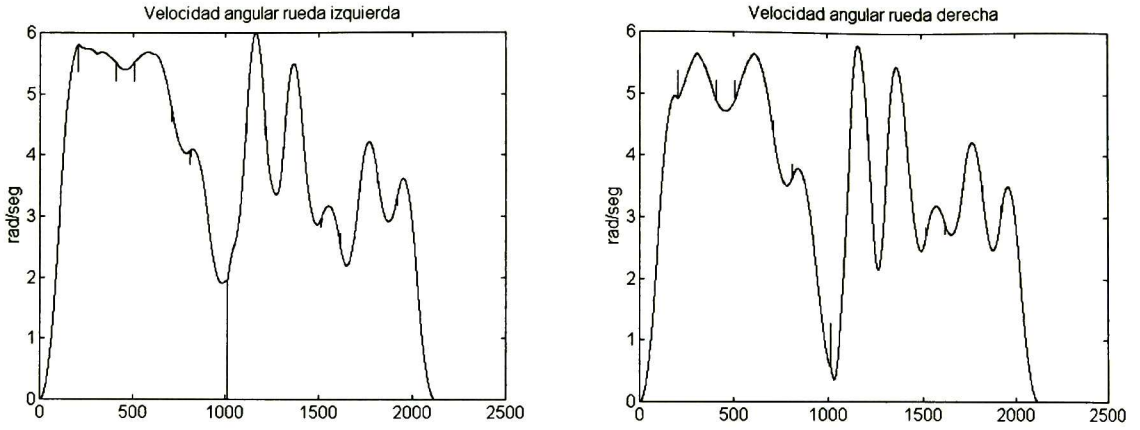


Fig. 3.17 Velocidades de referencia para cada motor.

De acuerdo a las velocidades angulares de referencia que se calcularon, la velocidad lineal del robot móvil será:

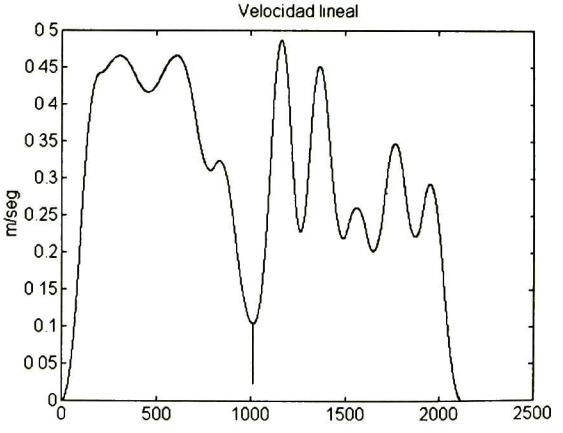


Fig. 3.18 Velocidad lineal del robot móvil.

La reconstrucción de la trayectoria a partir de las velocidades angulares calculadas es:

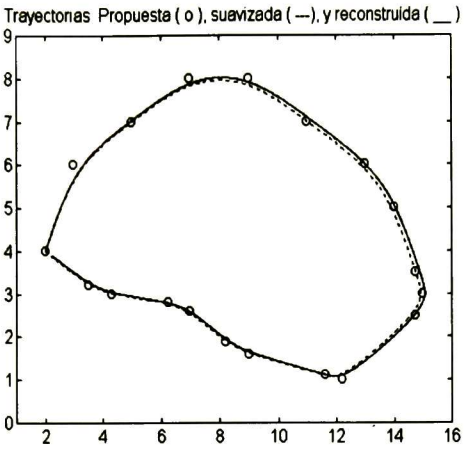


Fig. 3.19 Comparación de trayectorias.

Como puede observarse en las pruebas de simulación, el algoritmo desarrollado en este capítulo es adecuado para la planeación de trayectorias de un robot móvil correspondiente a la configuración sencilla de la Fig. 2.1. Se mostró que el algoritmo funciona inclusive en trayectorias que describen una curva cerrada (por una curva cerrada se entiende aquella curva cuyo punto inicial coincide con el punto final). Para una demostración formal de que el algoritmo funciona adecuadamente para este tipo de trayectorias ver [31].

En el próximo Capítulo, las velocidades de referencia generadas por este algoritmo de planeación de trayectorias se tomarán como referencia a seguir en un algoritmo de control de navegación de un robot móvil.

Capítulo 4
CONTROL JERARQUICO DIFUSO

4.1 Introducción

Este Capítulo inicia con la presentación de conceptos fundamentales de lógica difusa, motiva su aplicación al problema de control de sistemas y posteriormente introduce varios esquemas de control jerárquico.

A partir de los esquemas jerárquicos aquí presentados, se diseña un esquema de control jerárquico para el seguimiento de trayectorias y navegación expuestos en el Capítulo 5.

4.2 Lógica Difusa

El Control Difuso es una de las áreas más activas de aplicación de la Lógica Difusa y de la Teoría de Conjuntos Difusos introducida por Zadeh [46]. La Teoría de Conjuntos Difusos aporta las bases de la llamada síntesis lingüística, estableciendo un camino para, que a partir de planteamientos lógicos no precisos, obtener conclusiones. La Lógica Difusa fue aplicada al control de procesos industriales por primera vez por E. Mamdani y su grupo de trabajo del Queen Mary College de Londres, Inglaterra. A partir de los años 70, un numeroso grupo de científicos de varias nacionalidades europeas fueron pioneros de la aplicación de la Lógica Difusa a sistemas automáticos [16]. La principal característica de la Lógica Difusa es la robustez de su mecanismo de razonamiento interpolativo [36].

La gran ventaja de la Lógica Difusa es representar información lingüística gradual ("Está lloviendo", "Juan es joven", "El clima está templado", etc.), subjetiva (confortable,

peligroso, color, aroma, etc.), imprecisa (La temperatura está entre 20 y 30° C), e incierta (El clima de mañana será agradable).

Determinar una acción de control para un sistema es una tarea de toma de decisiones. La Lógica Difusa resulta una herramienta útil en situaciones de imprecisión en el modelado, de imprecisión en la información, o cuando se tienen restricciones en las acciones de control [11]. Los controladores basados en Lógica Difusa son sistemas de control inteligente que interpolan suavemente entre reglas de control; es decir, las reglas que se disparan dan lugar a diferentes acciones de control, las cuales se combinan para llegar a un resultado interpolado [39].

Los sistemas difusos pueden ser vistos como: *a*) sistemas basados en reglas (reglas SI-ENTONCES difusas), *b*) mapeos no lineales, con propiedades matemáticas como la aproximación universal [43]. Los sistemas difusos son ampliamente utilizados para formular operaciones de supervisión y planeación debido a que en muchos sistemas prácticos estas tareas son ejecutadas por humanos, quienes expresan el conocimiento usualmente en forma lingüística. En estos casos, las reglas SI-ENTONCES son una estructura apropiada para representar el conocimiento humano.

A continuación se presentan algunos conceptos básicos de Lógica Difusa.

Definición 4.1: Sea U un conjunto cuyos elementos toman valores discretos o continuos. U es llamado universo de discurso; y u representa un elemento genérico de U ($u \in U$).

Definición 4.2: Un conjunto difuso F, definido en un universo de discurso U, se caracteriza por una función de pertenencia μ_F, la cual toma valores en el intervalo [0, 1], $\mu_F : U \rightarrow [0, 1]$. Un conjunto difuso puede ser visto como una generalización del concepto de un conjunto ordinario, para el cual las funciones de pertenencia (función característica) toman solamente dos valores {0, 1}. Un conjunto difuso F definido en U se representa como el

conjunto de pares ordenados formados por un elemento genérico u y su grado de función de pertenencia:

$$F = \{(u, \mu_F(u)) \mid u \in U\}$$

Cuando U es continuo, un conjunto difuso F puede ser escrito como:

$$F = \int_U \mu_F(u)/u$$

donde $\int$ no significa integral sino unión de los pares ordenados $(u, \mu_F(u))$.

Para U discreto, un conjunto difuso F, puede ser escrito como:

$$F = \sum_{i=1}^{n} \mu_F(u_i)/u_i$$

donde $\sum$ no significa sumatoria sino unión de los pares ordenados $(u, \mu_F(u))$.

Un valor de cero en una función de membresía significa que definitivamente ese elemento no es miembro de dicho conjunto difuso; mientras que un valor de 1 en una función de membresía significa que definitivamente se trata de un miembro de ese conjunto difuso [17]. Las funciones de pertenencia más utilizadas en control difuso son de forma triangular y trapezoidal.

$$\mu_{tri}(x) = \begin{cases} 0 : x < a \\ \dfrac{(x-a)}{(b-a)} : a \le x \le b \\ \dfrac{(c-x)}{(b-a)} : b < x < c \\ 0 : c \le x \end{cases}$$

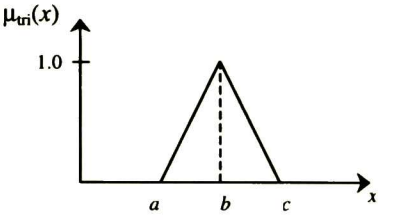


Fig 4.1 Función de membresía triangular.

$$\mu_{trap}(x)=\begin{cases}0: x<a \\ \dfrac{(x-a)}{(b-a)}: a\le x\le b \\ 1: b\le x\le c \\ \dfrac{(d-x)}{(d-c)}: c<x<d \\ 0: d\le x\end{cases}$$

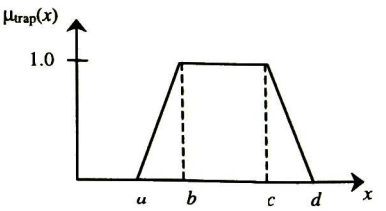


Fig 4.2 Función de membresía trapezoidal.

donde $a, b, c, d \in \Re$.

La capacidad de los conjuntos difusos para representar información imprecisa e incierta [36], se muestra en la siguiente figura con un ejemplo sobre temperatura.

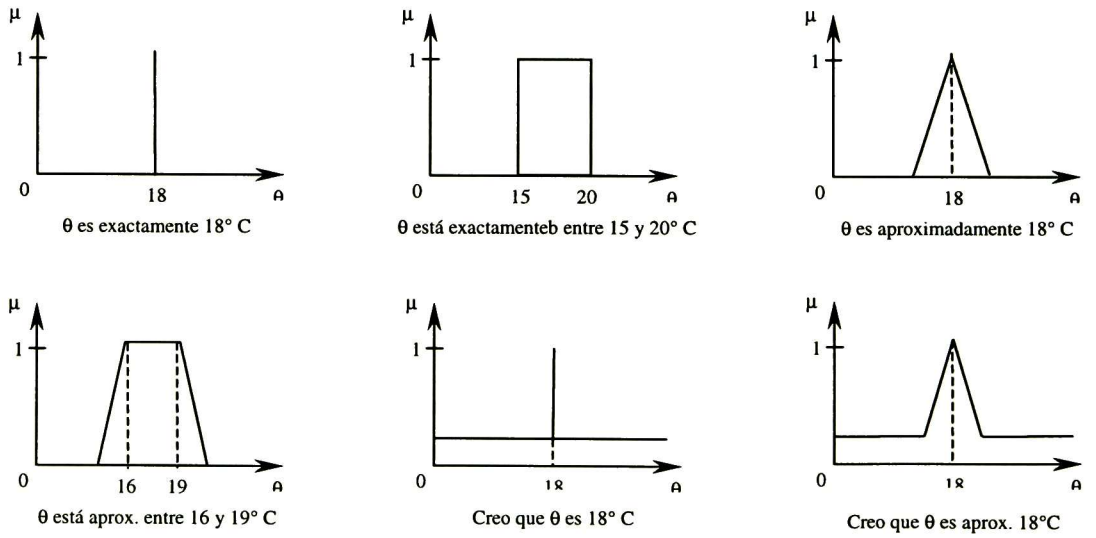


Fig. 4.3 Representación de información.

Definición 4.3: Una variable lingüística puede ser caracterizada por el trío ordenado (x, $T(x)$, U), donde x es el nombre de la variable, $T(x)$ es el conjunto calificativo de x, y U es el universo de discurso.

Ejemplo 4.1: Representación difusa de la variable *Velocidad.*

x = Velocidad

T(x) = T(Velocidad) = {lenta, media, rápida}

U = [0, 100] Km/hr

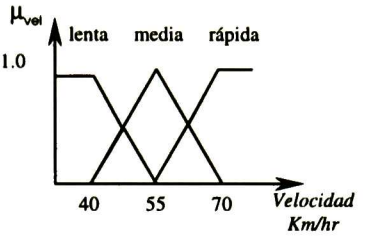


Fig. 4.4 Variable lingüística *Velocidad.*

Una forma canónica de una regla difusa es:

$$\text{SI } (X \text{ es } A) \text{ ENTONCES } (Y \text{ es } B) \quad \equiv \quad \text{SI } A \text{ ENTONCES } B$$

Es posible formar reglas más complejas:

- Con múltiples antecedentes conjuntivos.

$$\text{SI } \{(X_1 \text{ es } A_1) \text{ AND } (X_2 \text{ es } A_2)\ldots\} \text{ ENTONCES } (Y \text{ es } B)$$

que puede ser reescrita como:

$$\text{SI } A \text{ ENTONCES } B \qquad \text{donde } A = A_1 \cap A_2 \cap \ldots$$

- Con múltiples antecedentes disyuntivos:

$$\text{SI } \{(X_1 \text{ es } A_1) \text{ OR } (X_2 \text{ es } A_2)\ldots\} \text{ ENTONCES } (Y \text{ es } B)$$

que puede ser reescrita como:

$$\text{SI } A \text{ ENTONCES } B \qquad \text{donde } A = A_1 \cup A_2 \cup \ldots$$

donde A, A_1, A_2, B son conjuntos difusos; X_1, X_2 y Y son variables lingüísticas.

4.3 Control Difuso

La implementación de un controlador difuso requiere la asignación de funciones de pertenencia tanto a las variables lingüísticas de entrada como a las variables lingüísticas de salida.

Las entradas de un controlador difuso son normalmente variables de medición, asociadas con estados de la planta a controlar. Las entradas son difusificadas (proceso que asigna valores de pertenencia) antes de ser procesadas por una máquina de inferencia. Esta máquina está conformada por un conjunto de reglas SI-ENTONCES (*"if-then"*) cuyos antecedentes y consecuentes son variables lingüísticas. Los consecuentes del conjunto de reglas son combinados por la operación de unión y después son desdifusificados para obtener así una salida numérica (*"crisp value"*) como acción de control sobre el sistema. La Figura 4.5 muestra la estructura de un controlador difuso.

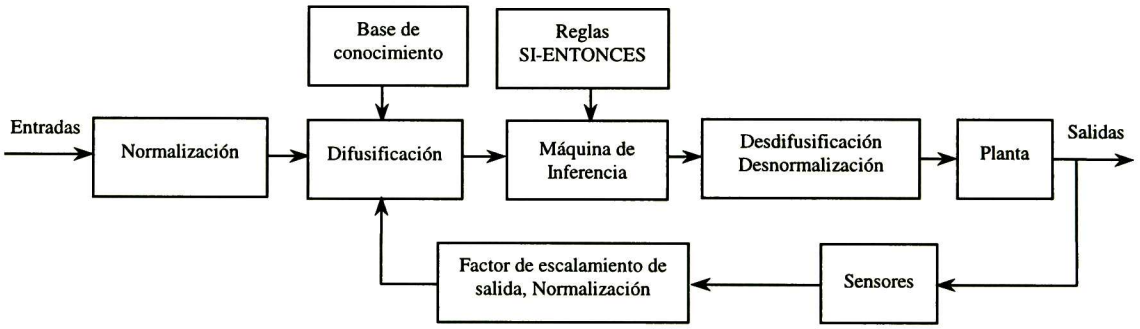


Fig. 4.5 Estructura de un Controlador Difuso [Titli].

¿Por qué aplicar Lógica Difusa en Control?

- Por su capacidad de trabajar con sistemas complejos.

- Porque a pesar de los avances en la Teoría de Sistemas No lineales, existe limitación en el modelado matemático; inconveniente que no presenta la Lógica Difusa.

- Porque toma ventaja de la experiencia humana, y la representa por medio de términos lingüísticos. Este tipo de información normalmente no se toma en cuenta en el desarrollo de modelos matemáticos..

- Por su capacidad de trabajar con información cualitativa, cuantitativa, subjetiva e imprecisa.

Debido a que un Controlador Difuso está conectado con el proceso a controlar, necesita una interfaz que convierta valores numéricos a simbólicos (proceso de difusificación), además de una interfaz que convierta valores simbólicos a numéricos (proceso de dedifusificación).

Las funciones del proceso de Difusificacción (o "*fuzzification*") [11] son:

1. Convertir el valor numérico de la variable en un conjunto difuso (función de pertenencia), para hacerlo compatible con la representación de conjuntos en el antecedente de las reglas SI-ENTONCES.

2. Determinar el proceso de difusificación, dependiendo del tipo de inferencia que se escogió previamente.

La Base de Conocimiento incluye el conocimiento específico de la aplicación. Está formada por una Base de Datos y una Base de Reglas de Control lingüísticas [23]. Sus funciones son:

1. Base de Datos: Proveer las definiciones necesarias para activar las reglas de control lingüísticas y para manipular los datos.

2. Base de Reglas: Caracterizar las tácticas de control heurísticas extraídas de los expertos codificadas como reglas lingüísticas.

La Máquina de Inferencia es la parte más importante de un Controlador Difuso; infiere acciones de control difusas empleando implicaciones difusas por medio de reglas de inferencia de Lógica Difusa.

La función del proceso de Desdifusificación (o "*difuzzification*") [11] es:

1. Determinar una acción de control numérica a partir de una acción de control difusa previamente calculada.

Un Controlador Difuso puede verse como un mapeo nolineal del espacio de entradas al espacio de salidas del sistema. Un Controlador Difuso es un sistema estático nolineal [36]. La no linearidad de los Controladores Difusos se debe a:

I. El uso de operadores no lineales
II. La no linearidad introducida, de forma voluntaria, en las reglas difusas

Un controlador difuso puede utilizar reglas de la forma:

$$\text{SI } \{(X_1 \text{ es } A_1) \text{ AND } (X_2 \text{ es } A_2)\ldots\} \text{ ENTONCES } (Y \text{ es } B)$$

para un controlador tipo Mamdami; o bien de la forma:

$$\text{SI } \{(X_1 \text{ es } A_1) \text{ AND } (X_2 \text{ es } A_2)\ldots\} \text{ ENTONCES } (Y = f(X_1, X_2, \ldots))$$

para un controlador tipo Sugeno.

Para reducir la complejidad de los sistemas de control difuso, se puede acudir a dos soluciones [36]:

❶ Descomposición del problema, estableciendo
   Jerarquía
   - Descentralización

❷ Fusión
   - Fusión Simbólica
   Fusión Matemática

En el contexto de control de robots móviles, un sistema basado en Lógica Difusa tiene la ventaja de representar fácilmente por medio de términos lingüísticos, la naturaleza intuitiva de la navegación basada en sensores. Además representa menor complejidad computacional y conduce la toma de decisiones en tiempo real permitiendo al robot un movimiento suave e ininterrumpido.

Para una introducción más detallada de Control Difuso, de operaciones con conjuntos difusos, y de conceptos como difusificación, inferencia, desdifusificación, etc., consultar [11], [17], [23].

4.4 Control Jerárquico

Los controladores difusos no jerárquicos son estructuras de procesamiento de información de un solo nivel. El razonamiento se ejecuta en un solo paso basándose en la información de entrada.

Existen diferentes clases de estructuras jerárquicas a las cuales se agregan niveles conceptuales o bien, el controlador mismo es reestructurado en una red de jerarquías formadas por subcontroladores.

La idea básica de control jerárquico consiste en descomponer la tarea de control en subtareas que se distribuyan en diferentes niveles de jerarquía. El diseñador puede crear un modelo del sistema compuesto de varios submodelos que aíslen diferentes segmentos para poder estudiarlos en detalle sin alterar la estructura completa del sistema. Se logra así un enfoque modular.

Los conceptos formales de multinivel y sistemas jerárquicos se definen en [27]. El concepto de *estrato* fue introducido en [7] para el modelado de arquitecturas organizacionales a ser analizadas con diferentes niveles de detalle. La definición formal de

una organización estratificada de toma de decisiones es: Una organización de toma de decisiones en la cual una unidad (o sistema) en un estrato dado es a su vez un componente (o subsistema) del estrato superior.

Por otro lado, los sistemas de control inteligente pueden definirse como sistemas que incluyen retroalimentación de control en su nivel más bajo, y supervisión y planeación en sus niveles más altos [42]. En sentido más amplio, una teoría de control inteligente es una teoría enfocada al análisis y diseño de sistemas de control organizados jerárquicamente.

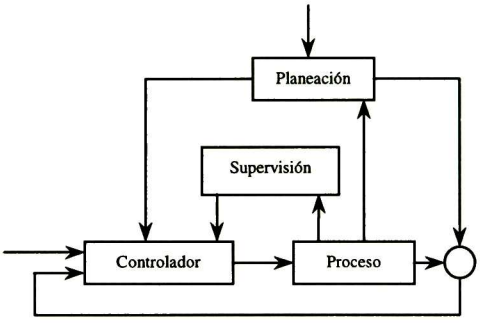


Fig. 4.6 Arquitectura general de un sistema de control inteligente.

En la literatura se encuentra que los sistemas jerárquicos pueden ser clasificados bajo dos enfoques: el heurístico y el matemático. El enfoque heurístico se caracteriza por sus diseños *ad hoc*, basados en la experiencia. El enfoque matemático establece los diferentes niveles jerárquicos dentro de un marco formal, con el que es posible diseñar y analizar los sistemas en forma rigurosa.

Ambos enfoques tienen ventajas y desventajas. Por ejemplo, la simplicidad y facilidad de implementar los sistemas jerárquicos con enfoque heurístico hacen de la intuición su principal ventaja. El inconveniente más importantes surge ante la necesidad de hacer análisis, de optimizar el diseño o de garantizar la estabilidad del sistema completo [42].

Los sistemas jerárquicos de enfoque matemático, por ejemplo los sistemas híbridos que hacen uso de diferentes herramientas de modelado, permiten un desarrollo matemático muy riguroso que da lugar a un mejor diseño y a un análisis más profundo. La principal desventaja es su complejidad.

Debido a que la capacidad de modelado de las teorías existentes no es lo suficientemente poderosa para tomar en cuenta todos los elementos del sistema de control real, se debe considerar también el enfoque heurístico. Esto tiene un inconveniente, la complejidad combinatoria. Debido a su naturaleza *ad hoc*, el enfoque heurístico considera todos los casos posibles. A veces la cantidad de líneas de código de las formulaciones heurísticas termina superando la cantidad que implica la programación matemática. Por lo que la alternativa matemática vuelve a ser atractiva para obtener un código más compacto y eficiente [42].

Hay al menos dos formas de realizar la etapa de modelado de un sistema conservando un enfoque matemático. La primera opción es combinar diferentes herramientas matemáticas de modelado para los diferentes elementos del sistema, dando lugar a un modelo híbrido. Por ejemplo, se utilizan ecuaciones diferenciales para modelar el nivel inferior, añadiendo sistemas de eventos discretos en los niveles de supervisión y planeación.

La segunda opción es llevar las heurísticas directamente a su forma matemática, y después combinarlas con herramientas formales de modelado ya existentes. La idea es aprovechar la dualidad que caracteriza a los sistemas difusos; por una parte son sistemas basados en reglas ("*IF-THEN*" difusas) y por otra, son mapeos no lineales con propiedades matemáticas [30].

Haciendo una comparación entre el modelado difuso y el modelado heurístico de sistemas de control jerárquico, la ventaja del enfoque difuso es la de ser una teoría matemáticamente más rigurosa, que permite un mejor análisis y diseño de sistemas

jerárquicos. Ahora bien, comparando el modelado difuso con los sistemas que utilizan ecuaciones diferenciales combinadas con sistemas de eventos discretos, la ventaja del enfoque difuso es que su modelo completo se encuentra en el dominio continuo de las ecuaciones diferenciales, por lo tanto, el análisis es más fácil y el sistema resultante es sencillo de implementar.

Según [42], los altos niveles de jerarquía son fuertemente influenciados por la toma de decisiones humana; los niveles más bajos reciben influencia de las leyes físicas. Bajo esta suposición utilizaremos sistemas difusos para formular las operaciones de supervisión y planeación, así mismo para controlar el nivel inferior: en nuestro caso el robot móvil.

4.5 Estructuras Jerárquicas de Controladores Difusos

4.5.1. Jerarquía de Reglas de Control. Las reglas de control de un controlador difuso pueden ser estructuradas de tal forma que los conjuntos de reglas específicas sean invocadas por reglas más generales. La idea de poder hacer una ampliación (*"zoom"*) en las reglas de control surge como solución a la necesidad de tener diferentes niveles de granulidad en los conjuntos difusos. El efecto de ampliación se origina en el conjunto de reglas típicas, donde algunas de ellas activan una colección de reglas que describen una estructura de control más detallada [34].

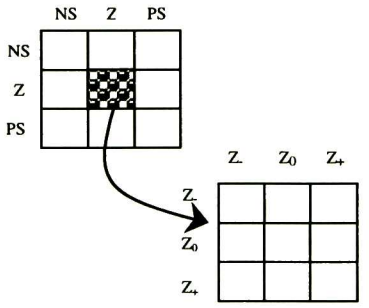


Fig 4.7 Efecto de ampliación en reglas de control

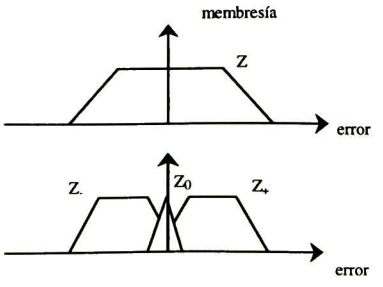


Fig. 4.8 Granulidad de conjuntos difusos

Este tipo de jerarquía se puede desarrollar con el fin de evitar la explosión combinatoria del número de reglas al incrementar el número de subcondiciones en el antecedente de cada regla.

4.5.2 Controlador Difuso Híbrido. El controlador difuso híbrido toma ventaja de las características no lineales de un controlador difuso (asegura las propiedades dinámicas del sistema bajo control) y la precisión alrededor de un punto garantizada por un controlador PID típico [30].

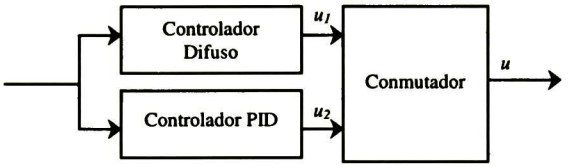


Fig. 4.9 Arquitectura de un controlador difuso híbrido.

La parte esencial de esta arquitectura radica en la conmutación. El objetivo es lograr una señal de control u, compuesta por la señal producida por el controlador difuso (u_1) y la señal proveniente del controlador PID (u_2). Para ello es necesario definir dos modos de operación del sistema: Cerca y Lejos, describiendo el grado de satisfacción/cercanía hacia los objetivos del sistema, basándose en el error y su derivada.

$$\text{Cerca}(e,de) = \text{Cerca\_cero}(e) \text{ AND } \text{Cerca\_cero}(de)$$
$$\text{Lejos}(e,de) = \text{Lejos}(e) \text{ AND } \text{Lejos}(de)$$

El valor de control u se calcula mediante una suma ponderada, donde los efectos de u_1 y u_2 son afectados por los grados de pertenencia anteriormente descritos.

$$u = \frac{\text{Cerca}(e,de)\, u_1 + \text{Lejos}(e,de)\, u_2}{\text{Cerca}(e,de) + \text{Lejos}(e,de)}$$

Para condiciones donde la función de membresía de Cerca se aproxime a 1 (la función de membresía de Lejos se aproximará a 0), el valor de control será casi dominado por la señal proveniente del controlador PID. En una situación contraria, la estructura exhibirá características dominantes del controlador difuso.

Esta estructura también puede ser aplicada a esquemas donde cada controlador es difuso [32].

4.5.3 Estructura Supervisora del Controlador Difuso. A diferencia de la estructura jerárquica anterior, donde el controlador difuso y el PID pertenecían al mismo nivel de operación, ahora el controlador difuso tiene un papel supervisorio con respecto a una serie de controladores PID [30]. La idea básica es descomponer una política de control compleja en un grupo de algoritmos de control locales. El algoritmo de control local operará en una región muy pequeña de valores de entrada (usualmente alrededor de un punto). Aún para sistemas no lineales y ligeramente no estacionarios, las políticas de control serán lineales dentro de una región limitada.

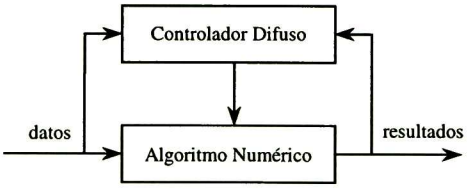


Fig. 4.10 Estructura supervisora de un controlador difuso

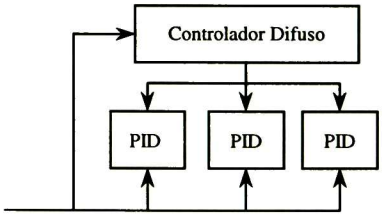


Fig. 4.11 Estructura supervisora utilizando controladores PID

Las reglas son más abstractas (menos precisas), y no están relacionadas con variables de control sino que invocan algoritmos de control locales (controladores PID especializados).

La estructura supervisora es común en diferentes áreas. El propósito general es que el controlador difuso represente conocimiento sobre el uso de algoritmos locales y menos generales. El objetivo es encapsular las heurísticas sobre la solución de problemas y delegar responsabilidad computacional de cómo llevar a cabo las acciones en un nivel más bajo. La estructura de control coordinada por el controlador difuso puede incluir un solo PID (o cualquier otro controlador) con parámetros ajustables.

En [1] se reportó una tarea interesante para mejorar el aprendizaje en redes neuronales. El aprendizaje se lleva a cabo por medio del método de Retropropagación. El controlador difuso se utiliza para ajustar los parámetros de aprendizaje del algoritmo. La regla de control relaciona el valor del criterio de desempeño (mapeo del error) con los parámetros del algoritmo.

4.5.4 Jerarquía de Comportamientos Difusos. En el nivel inferior de este tipo de estructuras reside un conjunto de comportamientos primitivos sencillos. Se trata de controladores difusos autocontenidos que buscan un mismo objetivo, operando ya sea en forma reactiva (no deliberada) o reflexiva (sin memoria). Por medio de inferencia difusa se llevan a cabo mapeos no lineales desde conjuntos difusos, por ejemplo de sensores, hacia actuadores comunes.

En [38] se utilizó esta estructura jerárquica para el problema de navegación de un robot móvil. De acuerdo a la estructura, también expuesta en [18], un solo comportamiento sería insuficiente para ejecutar tareas de navegación autónoma. Un robot móvil debe ser capaz de navegar evitando colisiones tanto en medio ambiente dinámico como estático. Es necesario además, que tenga capacidad de buscar el punto final de una trayectoria dada, navegar a través de habitaciones y pasillos, tomar como referencia una o varias paredes, etc. La jerarquía del ejemplo citado es la siguiente:

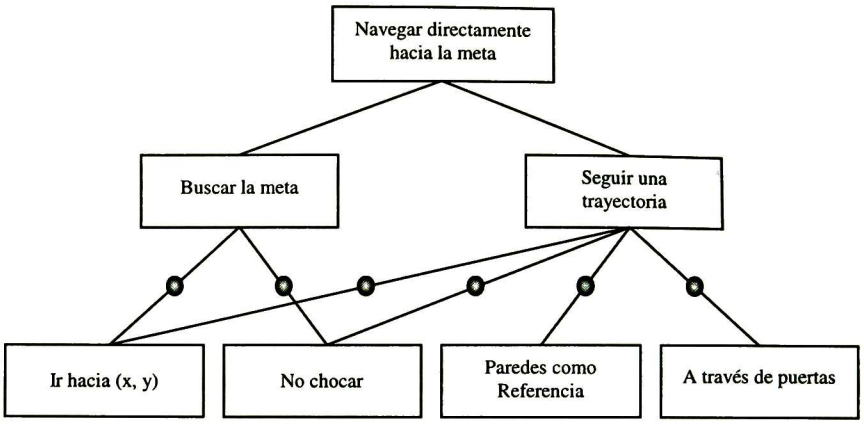


Fig. 4.12 Estructura jerárquica de las conductas de un robot móvil.

Los círculos interconectados representan pesos y umbrales de activación del comportamiento primitivo asociado. Como puede observarse, la descomposición de cada comportamiento en un robot móvil no es única.

4.5.5 Descomposición en Módulos Funcionales. La descomposición en Módulos Funcionales es el enfoque tradicional de sistemas de control de robots autónomos. El problema se descompone en una serie de unidades funcionales de la forma:

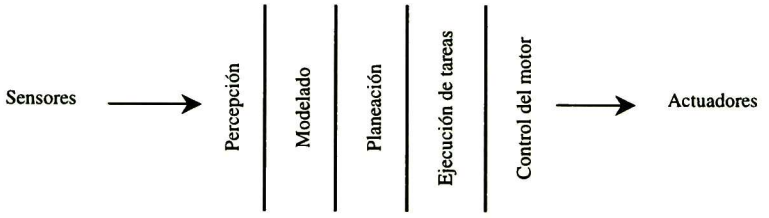


Fig. 4.13 Descomposición tradicional de un sistema de control de un robot móvil en módulos funcionales

Con frecuencia, los diseñadores de robots móviles [10], [15], [21], [28], [29], [37] desglosan el problema de la siguiente forma:

- captura de información del medio ambiente por medio de un subsistema sensorial

- creación de un mapa del mundo real de acuerdo a la información sensorial
- planeación
- ejecución de tareas
- control de motores

Los módulos funcionales forman una cadena por la que fluye la información del sistema de control; se inicia con la captura de información del medio ambiente por medio del subsistema sensorial y regresa al medio ambiente a través de acciones de control sobre los actuadores, cerrando así el lazo de retroalimentación. Es claro que algunas de las particiones por sí mismas tienen retroalimentación interna. Para efectuar modificaciones en la estructura de alguno de los módulos funcionales, ya sea para mejorar o simplemente para extender su funcionalidad, debe tenerse especial cuidado en las interfaces que unen cada pareja de módulos para continuar con un buen funcionamiento en la comunicación, o bien, las modificaciones también deben propagarse.

4.5.6 Sistemas de Control por Capas. En [6] en lugar de dividir el problema basándose en la solución de tareas internas, se divide tomando en cuenta las manifestaciones externas del sistema de control.

Con este fin, se definen los *niveles de competencia* para un robot móvil autónomo. Un nivel de competencia es una especificación informal de conductas de un robot en los diferentes medios en que se pueda encontrar. Un nivel de competencia alto, implica comportamientos más específicos.

En [5], se presentan los siguientes niveles de competencia:

- Evitar el contacto con objetos (ambiente dinámico y/o estático).
- Navegar sin meta fija, evitando el contacto con objetos.
- Explorar el mundo buscando lugares hacia donde se pueda dirigir, y dirigirse a ellos.

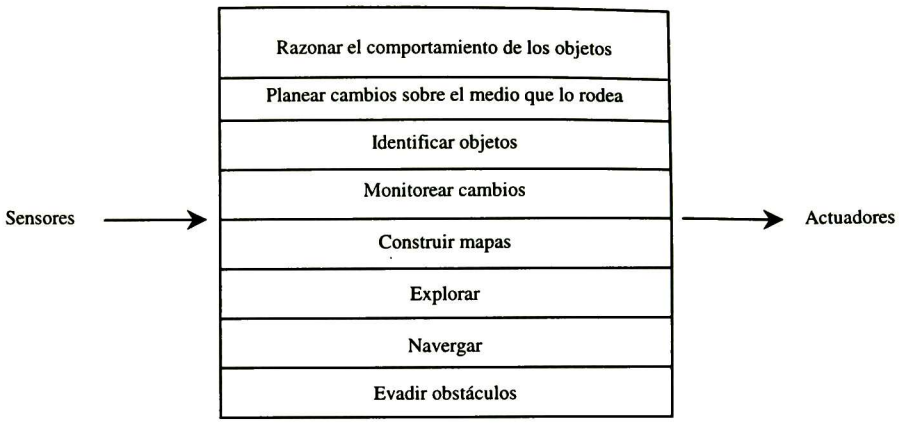


Fig. 4.14 Descomposición de un sistema de control por capas de un robot móvil.

- Construir un mapa del medio ambiente y planear trayectorias de un lugar a otro.
- Notificar algún cambio en el medio ambiente estático.
- Razonar sobre el medio ambiente en base a objetos identificables y ejecutar tareas relacionadas con ciertos objetos.
- Formular y ejecutar tareas que contemplen cambios en el estado del medio ambiente.
- Razonar sobre el comportamiento de los objetos del medio ambiente y hacer modificaciones en las tareas de acuerdo a estos comportamientos.

Cada nivel de competencia incluye como subconjunto al nivel anterior. Debido a que un nivel de competencia define un tipo de comportamiento válido, puede verse que los niveles más altos implican mayores restricciones que los niveles inferiores.

La idea principal de los niveles de competencia es construir un sistema de control por capas, donde cada capa corresponda a un nivel de competencia específico. Cuando sea necesario añadir un nuevo nivel de competencia, simplemente se agrega una nueva capa de control.

Basándose en las estructuras de control presentadas en este Capítulo, en el Capítulo 5 se presenta una estructura de control jerárquico difuso para robots móviles.

Capítulo 5
CONTROL DE NAVEGACION

5.1 Introducción

En la primer parte del este Capítulo se resume el problema de navegación de robots móviles, así como las herramientas más utilizadas para su solución.

En la segunda parte se presenta la principal aportación al trabajo de tesis. Se proponen dos algoritmos de control:

- ✓ Control de Navegación
- ✓ Control de Seguimiento de Trayectoria

El primer algoritmo se llevó a la implementación en tiempo real sobre el sistema físico en una configuración de robot móvil correspondiente a dos carros conectados por una unión giratoria.

El segundo algoritmo se llevó a simulación, tomando como modelo matemático las ecuaciones diferenciales correspondientes a una configuración simple de robot móvil, compuesta por un solo carro.

Para ambos algoritmos se presentan los resultados correspondientes.

5.2 El problema de Navegación

Los robots móviles han sido utilizados en diferentes tareas especializadas, como mensajeros dentro de oficinas y hospitales, como medio de transporte de materia prima en

plantas industriales, para desempeñar tareas en sistemas marítimos o subterráneos, así como también para exploraciones espaciales.

El problema de navegación de robots móviles se torna complejo cuando se trata de navegar en tiempo real en un medio desconocido con información ambigua e imprecisa [13], [35]. El principal objetivo en este campo es desarrollar algoritmos de navegación para afrontar situaciones donde la información capturada por el sistema sensorial es imprecisa. La mayor parte de estos algoritmos de control necesitan mapas del medio ambiente y consideran que siempre es posible obtener información precisa a través de sus sensores. Algunos de estos algoritmos son capaces de lograr una navegación satisfactoria en un medio ambiente desconocido; es decir, sin ayuda de mapas. La desventaja de tales algoritmos radica en que, con el fin de obtener suficiente información del medio ambiente, requieren de un sofisticado sistema sensorial con alto costo. En [2] se desarrolla un controlador difuso que utiliza información obtenida por unos cuantos sensores de ultrasonido para lograr la navegación de un robot móvil a través de un medio ambiente dinámico. La desventaja de este enfoque es que no controla la velocidad del robot móvil y por consecuencia, no es posible optimizar el tiempo de recorrido entre un punto y otro. Para satisfacer esta necesidad es necesario reducir el tiempo y la distancia recorrida por medio de un mecanismo que controle la velocidad del robot móvil, así como también resolver el problema de trabajar con información imprecisa e incierta. En un medio ambiente desconocido la distancia óptima de navegación también es desconocida, por lo tanto no es posible optimizarla formalmente. Sin embargo, el tiempo de navegación puede reducirse controlando la velocidad del robot móvil [4].

De acuerdo a [24], el problema de navegación de robots móviles puede ser resumido en tres preguntas:

- ¿Dónde estoy?
- ¿Hacia dónde voy?
- ¿Cómo llegaré allá?

A pesar de que el problema de navegación ha sido muy estudiado, aún no existe una verdadera solución del problema.

Con respecto a la primer pregunta, [3] plantea que las soluciones parciales más importantes para determinar la posición de un sistema (barco, avión, robot, etc.) durante su proceso de navegación se pueden dividir en dos categorías: mediciones de posición relativa y mediciones de posición absoluta. Debido a la carencia de una solución única, los diseñadores de vehículos guiados automáticamente (AGV's por sus siglas en inglés) y de robots móviles combinan métodos de ambas categorías. Cada categoría de medición puede ser dividida a su vez en varios subgrupos [3]. A continuación se presentan las dos categorías de medición y las subcategorías correspondientes.

Mediciones de posición relativa.

- **Odometría.** El procedimiento matemático que sirve para determinar la posición de un vehículo en base a posiciones anteriores y a información sobre su velocidad durante un período de tiempo a lo largo de una trayectoria, se llama "*dead reckoning*" La implementación más sencilla de "*dead reckoning*" para medir el desplazamiento de un vehículo a lo largo de una trayectoria se conoce como odometría [12]. Este método utiliza "*encoders*" para medir la velocidad de rotación de cada rueda así como su dirección. La principal ventaja es que, además de ser un sistema totalmente autocontenido, siempre es capaz de estimar la posición del vehículo. La desventaja es que el error de posición puede crecer sin límite a menos que se utilice periódicamente una referencia independiente [9]. Un ejemplo de instrumentos de odometría es el encoder óptico que va directamente acoplado a la flecha del motor o al eje de la rueda. Otros tipos de sensores de desplazamiento rotacional y de velocidad son:

  - Potenciómetros
  - Sincronizadores
  - "*Encoders*" de brocha

- "*Encoders*" magnéticos
- "*Encoders*" inductivos
- "*Encoders*" capacitivos

❷ **Navegación inercial.** Este método utiliza giroscopios y algunas veces acelerómetros para medir los cambios de rotación y de aceleración. Estas mediciones son integradas una o dos veces, según sea el caso, para obtener la posición. Estos sistemas también tienen la ventaja de ser autocontenidos, pero tienen como inconveniente la necesidad de integración lo que implica que cualquier pequeño error constante crecerá de forma ilimitada. En conclusión, los sensores inerciales no son recomendables para aplicaciones que requieran mediciones de posición por un período largo de tiempo. Además, los sensores inerciales son equipo de muy alto costo.

Mediciones de posición absoluta.

❶ **Faro activo (*"Active Beacon"*).** Los faros activos han sido utilizados durante varios siglos; las constelaciones estelares son un buen ejemplo de este tipo de guía de navegación. El método calcula la posición absoluta del robot móvil por medio de la recepción de señales de luz o radio frecuencia que se emiten desde tres o más transmisores colocados en lugares predeterminados del medio ambiente y un receptor que capta la señal en el robot móvil (o viceversa). Los faros activos pueden detectarse confiablemente y ofrecen información precisa con un mínimo de procesamiento. Como consecuencia, permite una frecuencia de muestreo mucho más grande. Existen dos tipos de sistemas de navegación de faro activo: trilateral y triangular (para mayor información consultar [3]). Un ejemplo es el reciente Sistema de Posicionamiento Global Navstar (*"Global Positioning System"*, GPS por sus siglas en inglés) desarrollado por el Departamento de Defensa de los Estados Unidos. Este sistema utiliza una constelación de 24 satélites que orbitan la Tierra cada 12 horas a una altura aproximada de 10,900 millas náuticas [14]. Conociendo la distancia exacta (por técnica de Tiempo de Vuelo mencionada en el Capítulo 2 del presente trabajo de tesis) entre un

receptor ubicado en la Tierra y tres satélites es posible calcular la latitud, longitud y altitud del receptor.

❷ **Reconocimiento de marcas (*"Landmarks"*) artificiales.** Este método consiste en distinguir marcas artificiales (*"landmarks"*) que se colocan en lugares predeterminados del medio ambiente. Un "*landmark*" es un distintivo que un robot puede reconocer por medio de su sistema sensorial. Puede tener forma geométrica (rectángulo, círculo, línea recta, etc.) así como también puede contener información adicional (p. e. en forma de código de barras). La ventaja del método es que es posible diseñar una distribución de "*landmarks*" que sea óptimamente detectable aún bajo condiciones adversas del medio ambiente. Al menos tres "*landmarks*" deben estar a la vista del robot para poder estimar su posición. El error de medición permanece acotado; pero no siempre es posible llevar a cabo la detección de "*landmarks*" y el cálculo de la posición en tiempo real.

❸ **Reconocimiento de marcas (*"Landmarks"*) naturales.** En este caso las marcas son características distintivas que pertenecen al medio ambiente. No hay necesidad de preparar el medio, pero sí es necesario conocerlo previamente. La confiabilidad de este método no es tan alta como la del reconocimiento de marcas artificiales.

❹ **Igualación de modelos.** Este método compara la información adquirida por medio del sistema sensorial del robot móvil y la información contenida en un mapa o modelo del medio ambiente. Si la información capturada por los sensores coincide con la información contenida en el mapa, entonces es posible calcular la posición absoluta del robot móvil. Este método normalmente incluye el mejoramiento de mapas globales basado en las nuevas observaciones sensoriales en un medio ambiente dinámico. Generalmente los mapas utilizados para navegación son de dos tipos: mapas geométricos y mapas topológicos. Los mapas geométricos representan al medio ambiente en un sistema de coordenadas global, mientras que los mapas topológicos representan al mundo como una red de nodos y arcos.

Para calcular la posición actual del robot móvil, debe elegirse cualesquiera de las técnicas de medición antes mencionadas (absoluta o relativa), o bien, seleccionar una combinación de ellas. Así se da respuesta a la primer pregunta, *¿Dónde estoy?*, sobre el problema de navegación.

Ahora bien, para responder a la pregunta *¿Hacia dónde voy?*, es necesario definir una condición que represente la meta o dicho en otras palabras, el fin del proceso de navegación; por ejemplo, alcanzar un punto predefinido (x, y), llegar a un cierto estado, terminar una secuencia de pasos que guíen al punto final de la navegación, etc. Para el caso en el que la meta sea un punto (x, y) predeterminado puede utilizarse cualquiera de los métodos de cálculo de posición explicados anteriormente para reconocer su ubicación.

Una vez que se conoce la posición actual del robot y la meta hacia la que se debe dirigir la navegación, surge la pregunta: *¿Cómo llegar hasta allá?*. Es probable que existan ciertas restricciones de navegación sobre el cómo llegar a la meta. Por ejemplo, las restricciones pueden ser que el robot móvil:

- ✓ Simplemente llegue a la meta
- ✓ Llegue óptimamente (según algún criterio o función de costo: tiempo, distancia, recursos, etc.)
- ✓ Alcance la meta evadiendo los obstáculos que se encuentren en su camino
- ✓ Ejecute una trayectoria predeterminada
- ✓ Combine algunas de las tareas anteriores

Obviamente para ejecutar estas tareas es necesario que el robot móvil cuente con los elementos de "*hardware*" (sensores y actuadores) necesarios para obtener información del medio ambiente que lo rodea y ejecutar acciones basadas en esa información.

Según [40], un robot móvil debe ser capaz de navegar sin chocar tanto en un medio ambiente estático como en un medio ambiente dinámico. Para ello es necesario que el robot

sea capaz de evadir obstáculos, localizar una meta, navegar en espacios complicados, etc. En [38] se expone una jerarquía de conductas que forman parte del proceso de navegación de un robot móvil [Fig. 5.1].

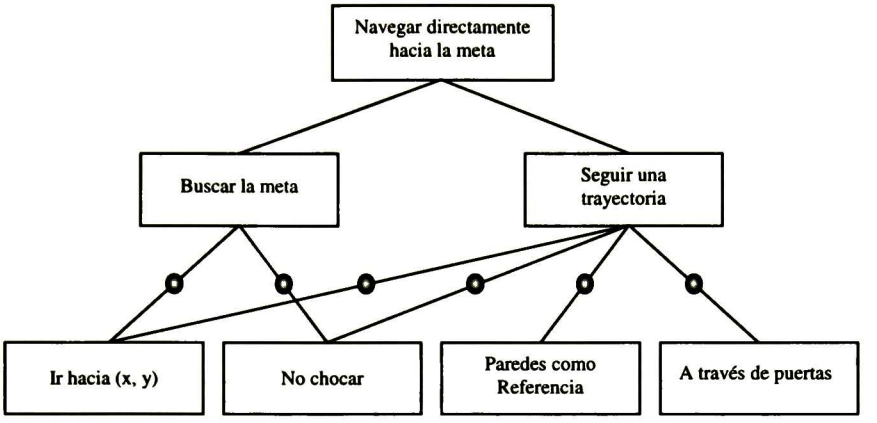


Fig. 5.1 Estructura jerárquica de las conductas de navegación de un robot móvil.

Esta figura implica que la navegación dirigida hacia una meta (*"goal-directed navigation"*) puede ser descompuesta en funciones conductuales tales como *"Buscar la meta"* (navegar hacia una posición determinada sin chocar) y *"Seguir una trayectoria"* (suponiendo que la dirección está dada por una secuencia de puntos o un mapa de trayectorias). A su vez, estas conductas pueden ser subdivididas en conductas primitivas que se visualizan en el nivel inferior de la Figura 5.1, con las dependencias indicadas por las líneas. Los círculos indican pesos o niveles de activación asociados a cada una de las conductas primitivas. La conducta *"Ir hacia (x, y)"* hace que el robot móvil navegue a lo largo de una trayectoria recta hacia una posición determinada. El objetivo de las primitivas *"No chocar"* y *"Paredes como Referencia"* va implícito en sus nombres. *"A través de puertas"* significa que el robot móvil debe ser capaz de navegar a través de pasadizos y puertas. Es importante ver que la descomposición de la conducta de un robot móvil no es única.

En resumen, la tercer pregunta *"¿Cómo llegar a la meta?"* debe ser contestada por medio de un algoritmo que permita satisfacer alguna (s) restricción (es) de navegación. En

la práctica es frecuente que un robot móvil no logre la trayectoria deseada aún cuando los motores de las ruedas ejecuten los valores de referencia. Esto puede ser causado por efectos acumulativos de perturbaciones impredecibles que surgen por imprecisiones en el modelado, por limitaciones en la exactitud computacional, o bien, por efectos mecánicos como fricción o vibración.

En la siguiente sección se describe un algoritmo para control de navegación de un robot móvil compuesto por dos carros, y además un algoritmo de control de seguimiento de trayectorias para un robot móvil de un solo carro.

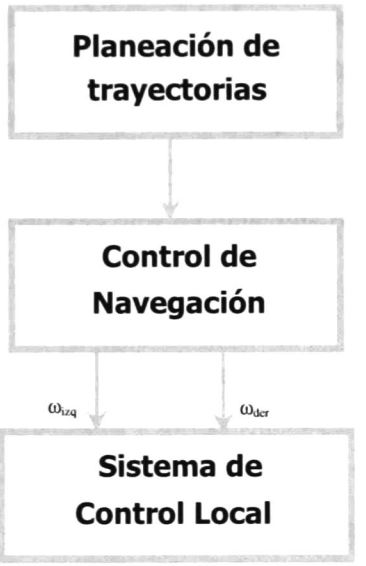
5.3 Algoritmos de Control de Navegación

El objetivo de los trabajos de investigación sobre control de robots móviles es lograr que estos sistemas sean capaces de interactuar con un medio ambiente dinámico. La noción de inteligencia en tales sistemas implica un sistema de control basado en conocimiento. El hecho de que un sistema sea autónomo implica que debe ser autocontenido, es decir, debe ser capaz de tomar información del medio ambiente por medio de sensores, razonar sobre la información capturada y actuar de acuerdo a esa información [39].

El mundo real incluye situaciones que pueden ser impredecibles, dinámicas, ruidosas e incluso desconocidas. Esto representa un inmenso nivel de incertidumbre que incrementa el esfuerzo necesario para desarrollar control en sistemas de robots móviles. En ocasiones algunos sistemas resultan tan complicados que es necesario suponer ciertas condiciones para que el control sea efectivo. Esto reduce el éxito de la implementación a una cantidad limitada de sistemas que cumplen con las características supuestas. Las investigaciones y aplicaciones más recientes que utilizan métodos no analíticos de inteligencia computacional como Lógica Difusa, Computación Evolutiva y Redes Neuronales, han demostrado el potencial y la utilidad de estas técnicas para el control inteligente de sistemas complejos [19].

Los dos algoritmos propuestos difieren uno a otro principalmente en su objetivo: uno va dirigido al control de navegación de un robot móvil y otro al seguimiento de trayectorias.

La estructura de control jerárquico diseñada para llevar a cabo los algoritmos propuestos es la siguiente:



5.3.2 Control de Navegación Difuso: Evasión de obstáculos.

En esta sección se presenta un algoritmo de control de navegación para robots móviles en tiempo real basado en Lógica Difusa. El acoplamiento que se logra entre la información sensorial y las acciones de control proveen al robot móvil de la adaptabilidad necesaria para interactuar con un medio ambiente dinámico. El robot es capaz de reaccionar inmediatamente ante la información percibida por los sensores.

5.3.2.1 Algoritmo

Objetivos del algoritmo de control:

- Evasión de obstáculos
- Pasar a través de puertas
- Reaccionar en tiempo real
- Describir una trayectoria suave, continua y segura

El robot móvil inicia el proceso de navegación con movimiento hacia adelante. Captura información del medio ambiente a través del subsistema sensorial para calcular la distancia a la que se encuentran los objetos más cercanos a su trayectoria. El robot móvil está restringido a no chocar. Con este algoritmo el robot móvil tiene facultad de navegar a lo largo de un pasillo, en un medio ambiente no solo desconocido sino dinámico, evitando colisiones.

El "*hardware*" necesario para llevar a cabo este algoritmo de control es:

- 3 sensores de ultrasonido
- 2 motores que trasmitan movimiento a las ruedas.

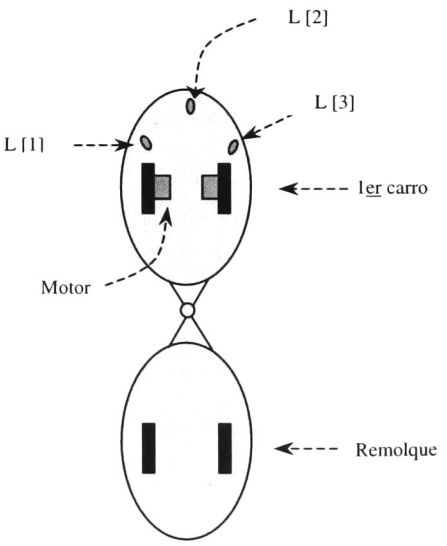


Fig. 5.5 Componentes del robot móvil necesarios para este algoritmo.

Las decisiones que el robot móvil toma durante este tipo de navegación están basadas en información sensorial, es decir, en la distancia a la que se detectan los objetos en el camino del robot móvil. La arquitectura de "*hardware*" necesaria para este algoritmo de control se muestra en la Fig. 5.5. En caso de que el robot móvil no tenga opciones de movimiento hacia adelante, se detendrá y esperará a que el objeto deje de interponerse en su trayectoria (en caso de ser un objeto dinámico). El resultado de este algoritmo de control son las velocidades angulares (ω_i, ω_d) a ejecutar por cada uno de los motores del robot móvil, lo cual se traduce en una trayectoria suave y segura.

Las reglas de control típicas de este algoritmo se aplican bajo las siguientes condiciones:

Sea:

R_r Límite máximo de acercamiento que puede existir con respecto a un objeto.
$L[i]$ Lectura del sensor i (i=1, 2, 3)

El vehículo comienza su movimiento en línea recta. El algoritmo de navegación consiste en lo siguiente:

1) Movimiento en línea recta hacia adelante.
2) Si $L[2] \leq R_r$, entonces detener el movimiento del robot enviando velocidades de referencia iguales a cero. Cuando $L[2] > R_r$, entonces ir al Paso 1.
3) Si $L[2] > R_r$, y
   3.1) Si $L[1] > L[3]$, entonces girar a la izquierda.
   3.2) Si $L[1] < L[3]$, entonces girar a la derecha.
4) Regresar al Paso 1.

Las reglas de control se escogieron de tal forma que por seguridad el máximo valor de velocidad del robot móvil se alcance únicamente al describir una línea recta. Al

momento en que el algoritmo de control indique un cambio en la dirección del robot, se disminuye la velocidad de la rueda correspondiente al lado que se quiera virar. Es decir, si el robot móvil debe girar a la derecha, el motor de la rueda derecha debe disminuir su velocidad en la proporción que indique la variable lingüística asociada a la acción de control. Para virar hacia la izquierda el procedimiento es similar.

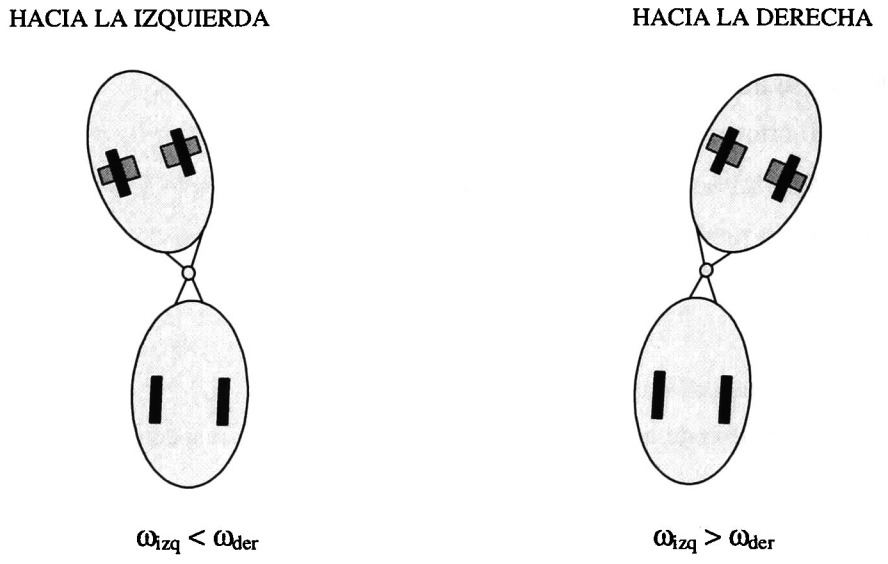


Fig. 5.6 Control de giro.

El primer paso del algoritmo indica movimiento del robot móvil en línea recta hacia adelante.

El segundo paso se lleva a cabo cuando el robot móvil detecta un obstáculo dentro de su límite de seguridad[†], entonces envía velocidades de referencia a los motores iguales a cero. Una vez que el objeto detectado deja de interponerse en la trayectoria se reanuda el movimiento en línea recta hacia adelante.

El tercer paso del algoritmo se lleva a cabo cuando la trayectoria del robot móvil está libre de obstáculos. Se determina la acción de control a llevar a cabo en cada rueda dependiendo de la variable lingüística asociada a la diferencia entre la lectura del sensor del

[†] *El límite de seguridad depende directamente de las restricciones físicas de los sensores de ultrasonido*

lado izquierdo y la lectura del sensor del lado derecho del robot. Esta diferencia se denomina delta: $\Delta = L[1] - L[3]$. En caso de que Δ sea positiva, la acción de control se lleva a cabo sobre la rueda derecha; en caso de que Δ sea negativa, la acción de control actúa sobre la rueda izquierda. Para un Δ igual a cero la acción de control hará que el vehículo siga navegando en línea recta. Las variables lingüísticas asociadas a Δ son: NB, NM, NS, ZE, PS, PM y PB; donde NB significa Grande Negativo (*"Negative Big"*), NM significa Mediano Negativo (*"Negative Medium"*), NS significa Pequeño Negativo (*"Negative Small"*), ZE significa Cero (*"Zero"*), PS significa Pequeño Positivo (*"Positive Small"*), PM significa Mediano Positivo (*"Positive Medium"*), y PB significa Grande Positivo (*"Positive Big"*).

El control difuso MIMO tiene la siguiente estructura:

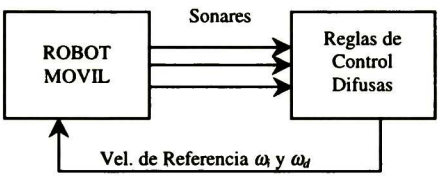


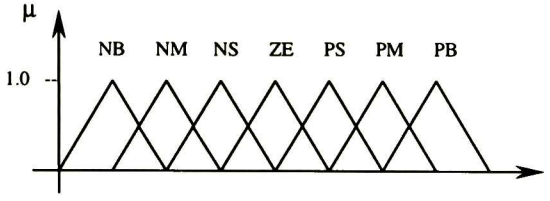
Fig. 5. 7 Estructura de control difuso MIMO.

Las salidas del controlador difuso son: Velocidad Rueda Izquierda (ω_i) y Velocidad Rueda Derecha (ω_d), cuyas variables lingüísticas son NB, NM, NS, ZE, PS, PM, PB.

Las funciones de pertenencia de las variables de entrada y de salida son:

ENTRADA

$\Delta = L[_1] - L[_3]$

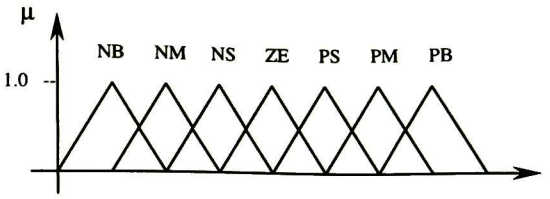


(Capítulo 2)

SALIDAS

Velocidad Rueda Izquierda (ω_i)

Velocidad Rueda Derecha (ω_d)



Las reglas de control difusas son:

R_1. Si Δes NB entonces *Velocidad* es NB

R_2. Si Δes NM entonces *Velocidad* es NM

R_3. Si Δes NS entonces *Velocidad* es NS

R_4. Si Δes ZE entonces *Velocidad* es ZE

R_5. Si Δes PS entonces *Velocidad* es PS

R_6. Si Δes PM entonces *Velocidad* es PM

R_7. Si Δes PB entonces *Velocidad* es PB

Este controlador difuso mapea en tiempo real un espacio de entrada (información capturada por los sensores) hacia un espacio de salida (una trayectoria libre de colisiones). Esto se logra por medio de una máquina de inferencia basada en reglas SI-ENTONCES tomadas de una base de conocimiento. El algoritmo genera las velocidades (y la dirección) necesarias para que el robot móvil alcance la meta en forma segura.

Definición 5.1. Sea X una variable lingüística y sea *LX* el significado de su valor lingüístico representado por la función de pertenencia $\mu_{LX} : X \rightarrow [0, 1]$ donde *X* es el dominio de X. El Valor Pico de μ_{LX} es el valor del dominio de *X* cuyo valor de pertenencia es igual a 1, i.e., $\mu_{LX}(x_{pico}) = 1$. La siguiente figura muestra el Valor Pico de una función de pertenencia triangular. En el caso de funciones de pertenencia trapezoidales el Valor Pico es el valor medio del intervalo donde se cumple la condición anterior.

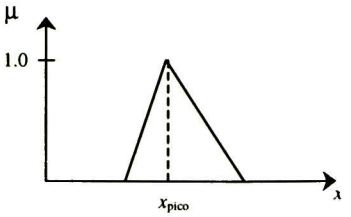


Fig 5.8 Valor Pico de una función de pertenencia.

El diseño del Controlador Difuso activa como máximo dos reglas al mismo tiempo. Se utiliza *Producto* para la inferencia y el método de *Altura* para el proceso de desdifusificación [11]. Este método utiliza salidas de control escaladas (*"individual clipped"*); toma el Valor Pico de cada salida y calcula la suma ponderada de estos valores. Ni el soporte ni la forma de las funciones de pertenencia de las variables lingüísticas de salida juegan un papel importante en el cálculo de u^* (salida pico). Este método de desdifusificación es simple y rápido. Sea c_k el valor del universo de discurso correspondiente al pico de salida, y f_k la altura de dicha salida (función de pertenencia). El método de desdifusificación por Valor Pico para m reglas está formalmente definido por:

$$u^* = \frac{\sum_{k=1}^{m} c_k f_k}{\sum_{k=1}^{m} f_k}$$

Para nuestra aplicación esta fórmula es evaluada de la siguiente manera:

$$u^* = \frac{c_1 f_1 + c_2 f_2}{f_1 + f_2}$$

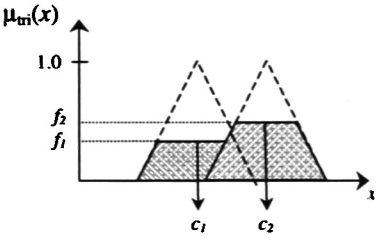


Fig. 5.9 Método de desdifusificación por Valor Pico.

donde los índices 1 y 2 identifican las dos reglas activadas.

Con el fin de controlar en tiempo real al robot móvil durante el proceso de implementación se simplifican las reglas reduciendo su cantidad. Para ello se tomó como principal idea la estructura jerárquica de la sección 4.5.1. Los valores lingüísticos de Δ (entrada) y de las Velocidades de Referencia Izquierda y Derecha (salida) fueron reducidos de siete a cuatro, con lo cual también se redujo el número de reglas SI-ENTONCES y por lo tanto, el tiempo de procesamiento de la ley de control.

La ventaja de este algoritmo sobre otros semejantes [25], [39] es que hace posible la navegación con una cantidad y variedad de sensores reducida: sólo 3 sensores de ultrasonido. En [25] se desarrolla un algoritmo de control semejante; su robot móvil cuenta con 24 sensores de ultrasonido y el sistema físico está compuesto por un solo carro.

La nomenclatura de la Fig. 5.12 representa los límites utilizados en el diagrama de flujo navegación con evasión de obstáculos de la Fig. 5.11. Puede observarse la reducción en el número de variables lingüísticas tanto de entrada como de salida.

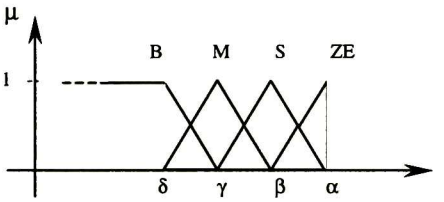


Fig. 5.12 Categorización de $|\Delta|$.

La referencia $\Delta = 0$ significa que el robot móvil siempre tenderá a ubicarse en una posición tal que el objeto detectado a su lado derecho sea equidistante al objeto detectado a su lado izquierdo.

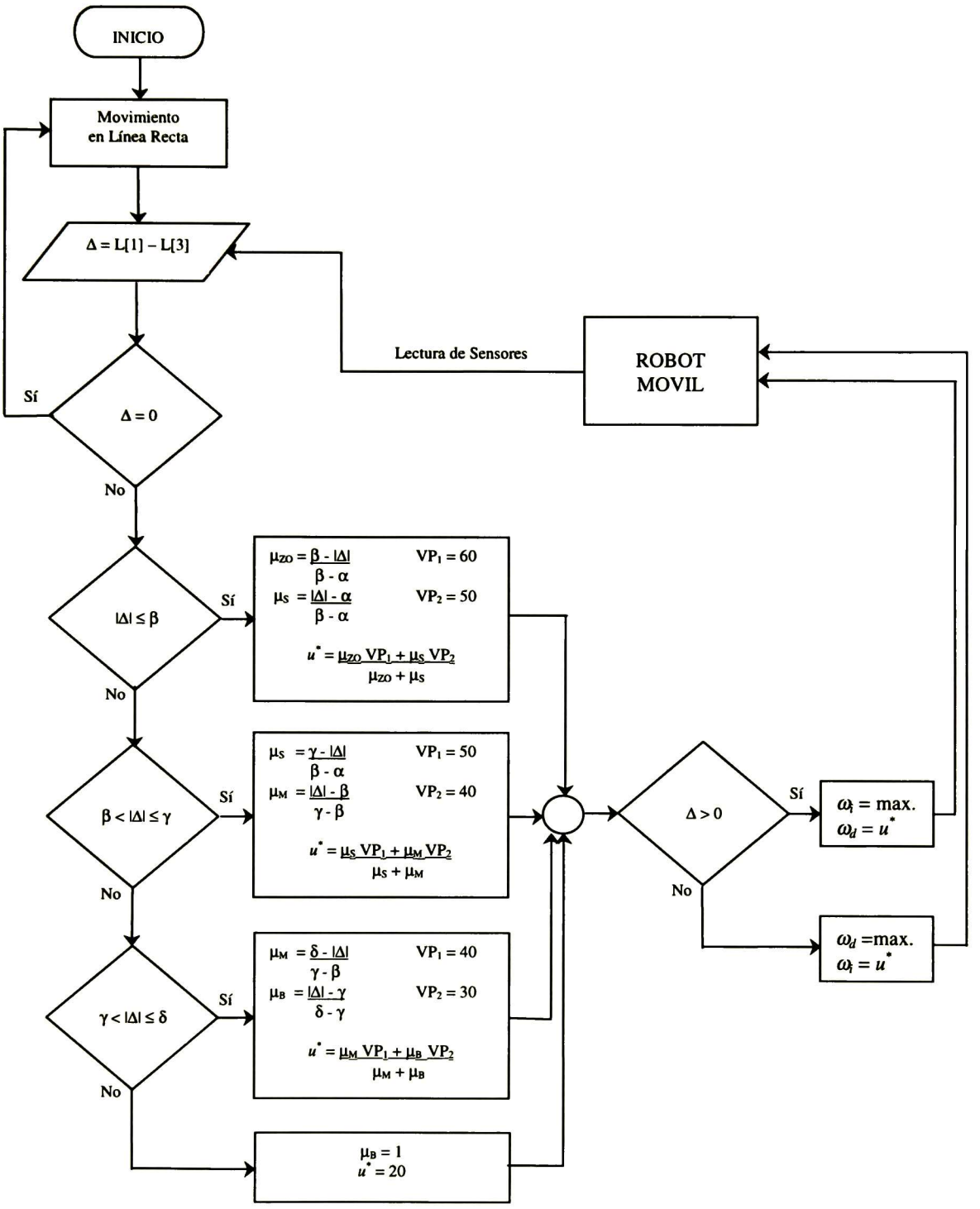


Fig. 5.11 Algoritmo de control de navegación para evasión de obstáculos.

5.3.2.1 Resultados Experimentales

Los resultados experimentales fueron satisfactorios, se alcanzaron todos los objetivos:

- ✓ Navegación en interiores
- ✓ Evasión de obstáculos
- ✓ Cruce de puertas
- ✓ Ejecución de una trayectoria suave, continua y segura.

El acoplamiento que se logró entre la información sensorial y las acciones de control proveen al robot móvil de la adaptabilidad necesaria para interactuar con un medio ambiente dinámico. El robot fue capaz de reaccionar inmediatamente ante la información percibida por los sensores. Se dejó evidencia de la implementación en una grabación en VHS.

5.3.1 Seguimiento de Trayectoria

En esta sección se presenta un algoritmo que controla la velocidad de las ruedas del robot móvil con el fin de que ejecute velocidades de referencia. Cuando ésto se logra, el robot móvil describe la trayectoria propuesta. Este algoritmo está diseñado para un robot móvil de estructura sencilla de un solo carro (Fig. [2.1]). Se anexan los resultados obtenidos a nivel simulación.

La trayectoria deseada se genera a partir de un conjunto de puntos en un plano, como se explicó en el Capítulo 3 del presente trabajo de tesis. Se utiliza el algoritmo de generación de trayectorias por *"B-Splines"*, del cual se obtienen las velocidades de referencia para la rueda derecha y para la rueda izquierda (ω_d, ω_i) del robot móvil.

5.3.1.1 Algoritmo

Objetivos del algoritmo de control:

- Seguimiento de trayectoria
- Control de velocidad de los motores con el fin de mantener el error acotado

Las velocidades de referencia generadas por el algoritmo de *"B-Splines"* se normalizan a un rango de [0, 6] radianes por segundo de acuerdo a la capacidad de los motores (Capítulo 2, sección 2.3.1.1).

5.3.1.2 Resultados de Simulación

El proceso de simulación se llevó a cabo en MATLAB Simulink. Se creó una estructura de control de tal forma que aceptara las velocidades de referencia generadas por el algoritmo de *"B-Splines"*, independientemente de la forma de la trayectoria deseada.

Los elementos más importantes de esta estructura de control son los siguientes:

- Modelo del robot móvil
- Controlador Difuso
- Velocidades de referencia

La estructura del controlador en MATLAB Simulink es:

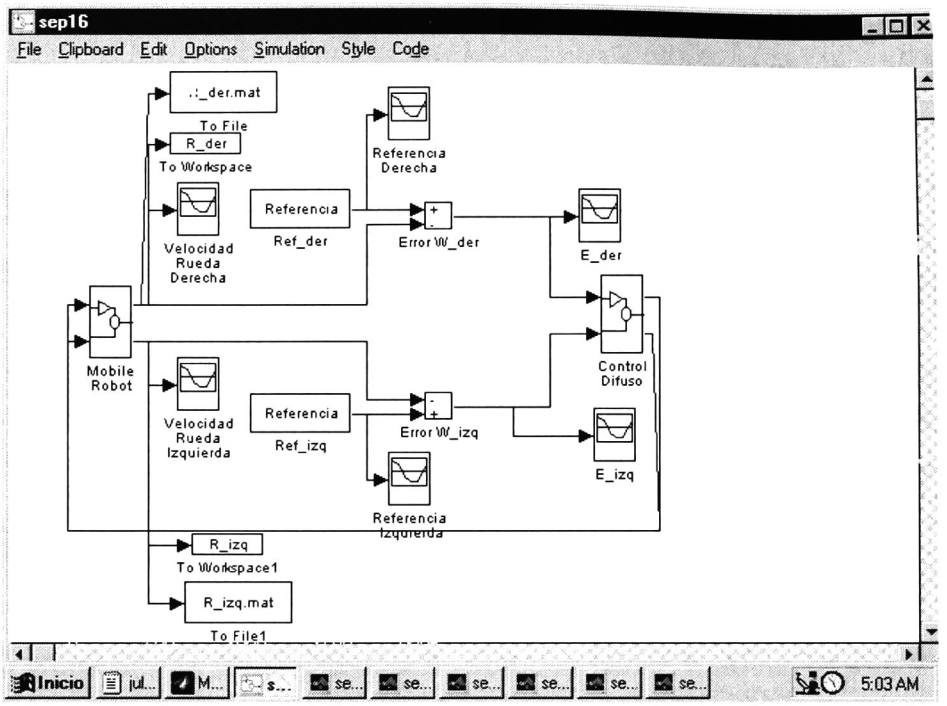


Fig. 5.2 Estructura del controlador PD-Difuso en MATLAB Simulink

El icono con título *"Mobile Robot"* contiene las ecuaciones dinámicas correspondientes al modelo del robot móvil de un solo carro. Estas ecuaciones fueron expuestas en el Capítulo 2 (sección 2.2). Su estructura es la siguiente:

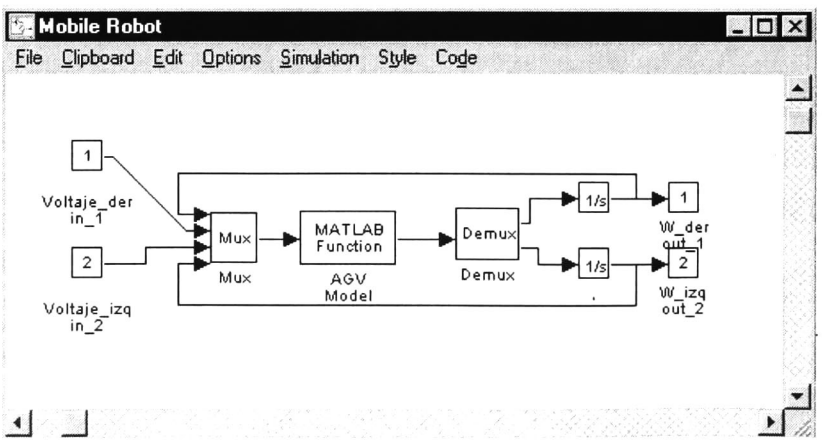


Fig. 5.3 Estructura del modelo del robot móvil en MATLAB Simulink.

El Controlador Difuso tiene la siguiente estructura interna:

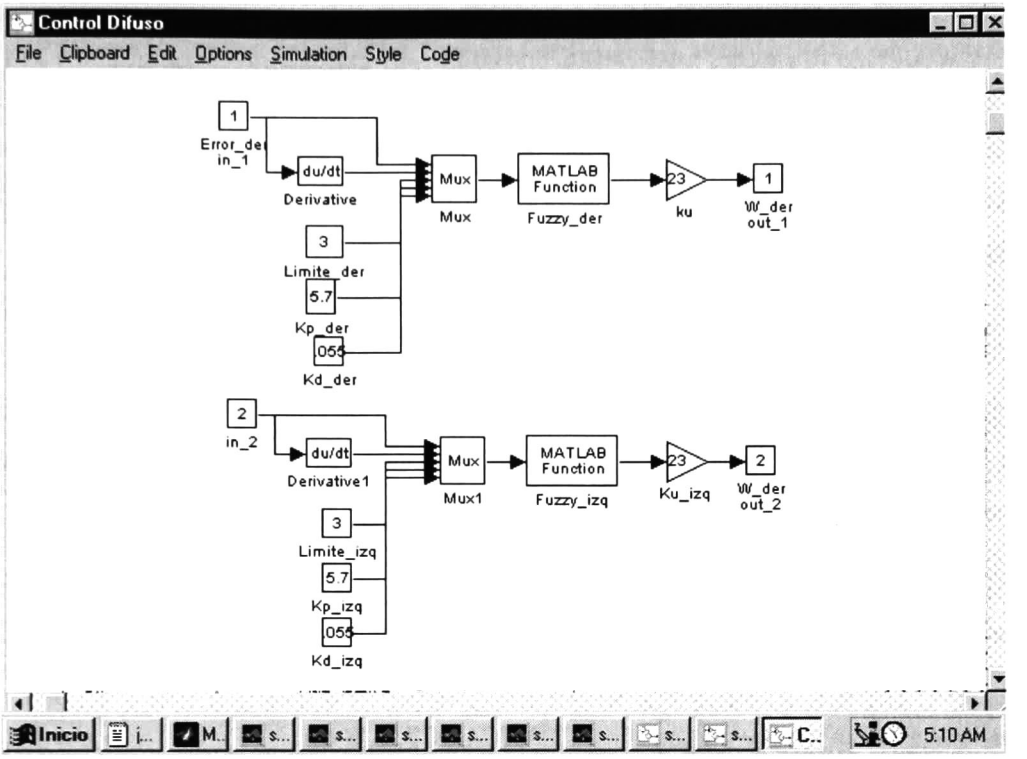


Fig. 5.4 Estructura del Controlador Difuso en MATLAB Simulink.

Como puede verse, se trata de dos estructuras completamente idénticas. La estructura superior corresponde a la rueda derecha y la inferior a la rueda izquierda. Esta duplicidad se debe a que la velocidad de cada rueda es calculada por medio del mismo proceso, pero de forma independiente.

Las entradas del controlador difuso son: el error de la rueda derecha y el error de la rueda izquierda, así como sus derivadas. El error se calcula de la siguiente forma:

$$\text{Error}_{\text{Rueda Derecha}} = \text{Referencia}_{\text{Rueda Derecha}} - \text{Velocidad Real}_{\text{Rueda Derecha}}$$

$$\text{Error}_{\text{Rueda Izquierda}} = \text{Referencia}_{\text{Rueda Izquierda}} - \text{Velocidad Real}_{\text{Rueda Izquierda}}$$

Las reglas de control utilizadas corresponden a una derivación de las siguientes reglas básicas:

Si e es P AND $\dot{e}$ es P entonces $u^{\blacksquare}$ es N
Si e es P AND $\dot{e}$ es N entonces $u^{\blacksquare}$ es ZO
Si e es N AND $\dot{e}$ es P entonces $u^{\blacksquare}$ es ZO
Si e es N AND $\dot{e}$ es N entonces $u^{\blacksquare}$ es P

e \ $\dot{e}$	P	N
P	N	ZO
N	ZO	P

Apartir de este conjunto de reglas básicas [26] obtiene una descomposición de veinte regiones mostrada en la siguiente figura, donde cada región corresponde a una ley de control:

$u^{\blacksquare} = \frac{L}{2(2L - k_p|e|)}[k_p e + k_d \dot{e}]$ IC1, IC2, IC5, IC6

$u^{\blacksquare} = \frac{L}{2(2L - k_d|\dot{e}|)}[k_p e + k_d \dot{e}]$ IC3, IC4, IC7, IC8

$u^{\blacksquare} = 0.5(L + k_d \dot{e})$ IC9, IC19

$u^{\blacksquare} = 0.5(L + k_p e)$ IC11, IC12

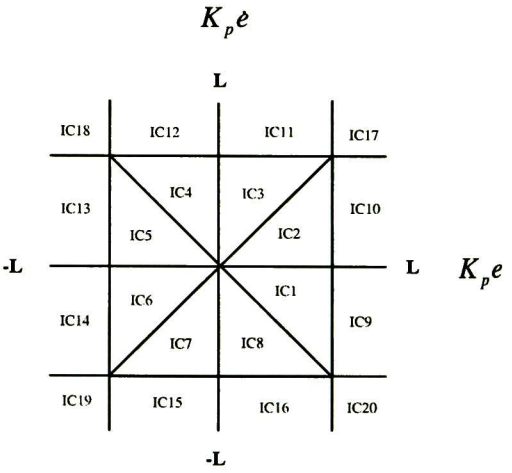
$u^{\blacksquare} = 0.5(-L + k_d \dot{e})$ IC13, IC14

$u^{\blacksquare} = 0.5(-L + K_P e)$ IC15, IC16

$u^{\blacksquare} = L$ IC17

$u^{\blacksquare} = -L$ IC19

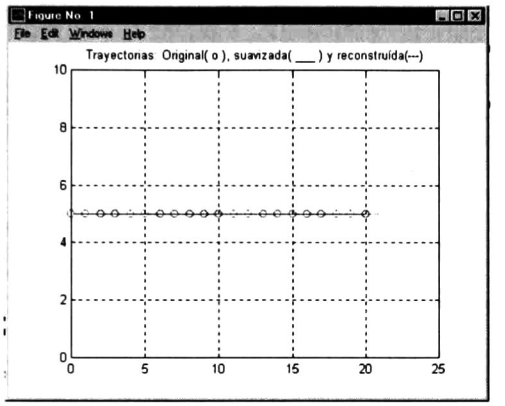
$u^{\blacksquare} = 0$ IC18, IC20



Además de observar el comportamiento del error (y de su derivada) para cada rueda, se hizo la reconstrucción de la trayectoria a partir de las velocidades de referencia generadas. A continuación se presentan los resultados obtenidos para dos trayectorias propuestas.

Simulación 5.1: Trayectoria propuesta: en línea recta.

La siguiente gráfica muestra la trayectoria propuesta representada con pequeños círculos. En la misma gráfica se muestra la trayectoria suavizada por *"B-Splines"* y la trayectoria reconstruida. Para una trayectoria en línea recta es difícil notar la efectividad de los algoritmos, pero los ejemplos subsecuentes dan una mejor prueba.



Las gráficas que se muestran a continuación corresponden a:

Vel. de Referencia Rueda Izquierda

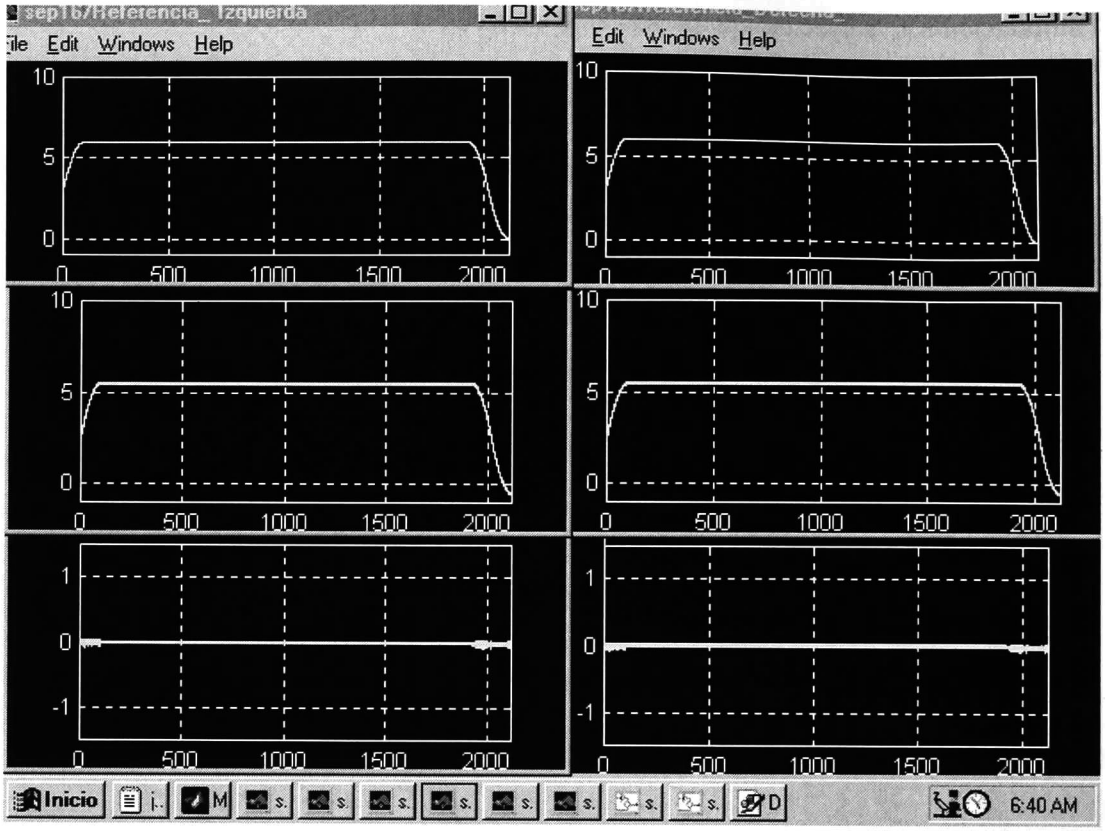
Vel. Real Rueda Izquierda

Error Rueda Izquierda

Vel. de Referencia Rueda Derecha

Vel. Real Rueda Derecha

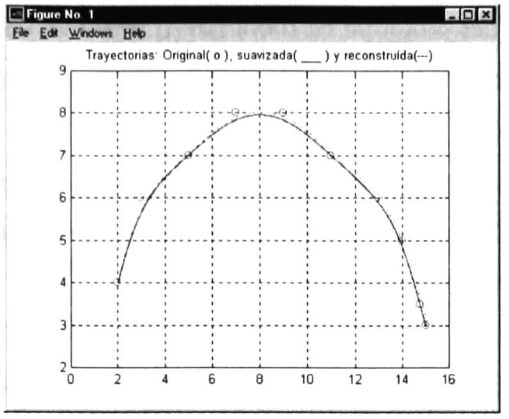
Error Rueda Derecha



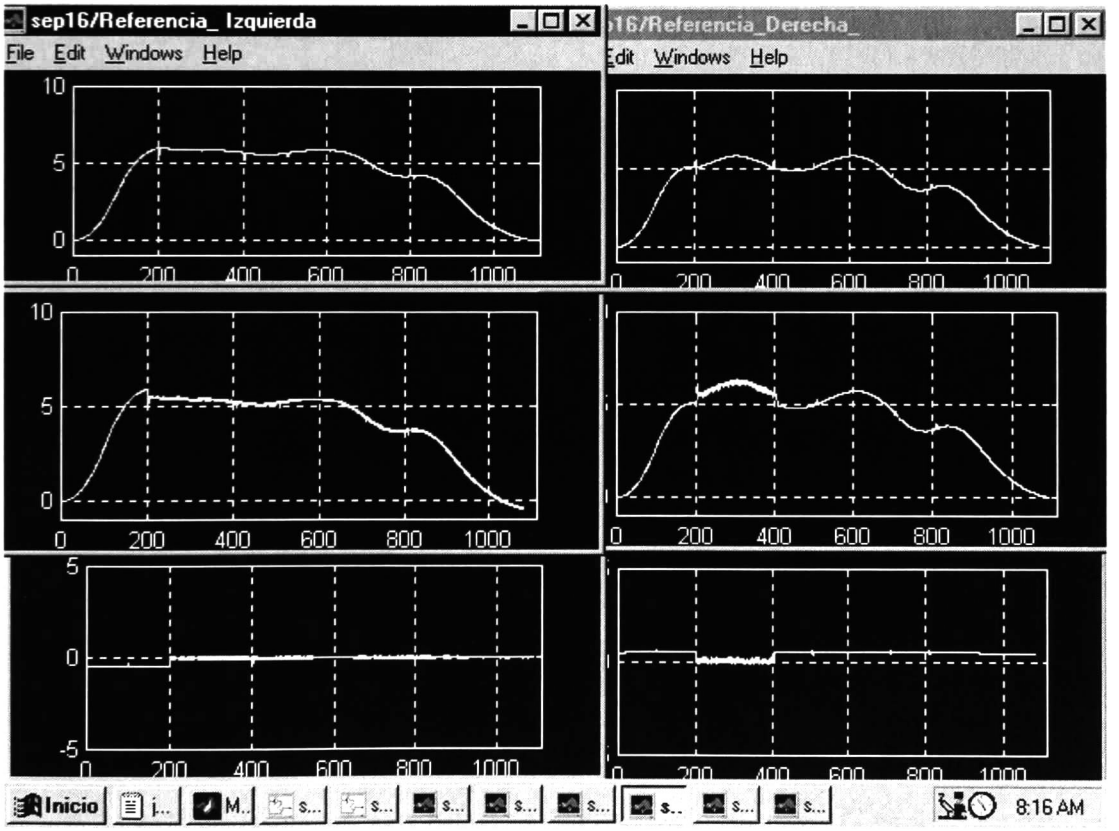
La mejor forma de evaluar el desempeño de los algoritmos es observando los resultados en las gráficas, en especial las dos gráficas ubicadas en el nivel inferior correspondientes al error de cada rueda.

Simulación 5.2: Trayectoria propuesta: curva abierta suave.

Con el mismo formato que la simulación anterior, aquí se presenta la trayectoria original, suavizada y reconstruida.



En las siguientes gráficas se muestran las velocidades de referencia, las velocidades reales y el error entre las mismas para este proceso de simulación.



Aún cuando el modelo que se utilizó para llevar a cabo esta simulación no corresponde al modelo del prototipo de robot móvil de CINVESTAV Unidad Gdl, llevar a cabo la experimentación del algoritmo hubiera sido una tarea interesante.

El principal inconveniente fue que se presentaron fuertes errores mecánicos en el sistema, por lo cual, la información capturada del medio ambiente sería errónea. Razón por la cual decidió dejarse en simulación.

Los resultados que aquí se presentan son bastante satisfactorios, ya que indican que es posible obtener velocidades de referencia acordes a la capacidad física de los motores; y lo mejor, que el algoritmo de control tiene un mínimo error con respecto a la referencia.

Capítulo 6
CONCLUSIONES

6.1 Conclusiones

En el presente trabajo de tesis:

- Se participó en el diseño, construcción y control del prototipo de robot móvil de CINVESTAV-Gdl.
  - Se comprendieron los principios del funcionamiento del prototipo de robot móvil de CINVESTAV-Gdl.
  - Se programó el *software"* que permite controlar al prototipo de robot móvil desde la PC. El *"software"* se caracteriza por ser transportable y el protocolo de comunicación permite que la información de *"hardware"* sea reutilizable por cualquier persona que quiera implementar algún algoritmo de control.
- Se asentaron las bases de Lógica Difusa, así como también de Control Difuso.
- Se propuso un algoritmo de control de velocidad PD-Difuso de las ruedas de un robot móvil compuesto por un solo carro.
  - Se resolvió el problema de planeación de trayectorias para casos en los que la trayectoria es una curva cerrada y para aquellos casos en los que el valor de la abscisa no siempre se comporta en forma ascendente (o descendente).
  - Se presentaron resultados de simulación, los cuales son favorables.
- Se propuso un algoritmo para control de navegación basado en Lógica Difusa para un robot móvil compuesto de dos carros conectados por una unión giratoria.

- El algoritmo superó el efecto del segundo carro sobre el primero.
- Se aplicó simplificación de reglas de tal forma que el cálculo de la ley de control fue mucho más rápido, y por lo tanto el robot siempre reaccionó a tiempo.
- El algoritmo se elaboró de tal forma que el subsistema sensorial necesario para lograr una navegación suave y segura fue mínimo, y por lo tanto económico, y capaz de trabajar con información incierta y ruidosa.
- Se llevó a cabo la implementación en tiempo real de este algoritmo, obteniendo resultados favorables.
- Para el éxito del algoritmo no fue necesario el conocimiento del modelo del robot móvil compuesto por dos carros.
- A raíz de este algoritmo de control de navegación se presentó un artículo en el *1998 IEEE/RSJ International Conference Intelligent Robots and Systems, Victoria, B.C. Canada.*

6.2 Trabajo Futuro

Las directivas que permitirán en un futuro extender y mejorar los logros de la presente tesis son:

- El desarrollo de una interfaz gráfica amigable para la plataforma de control.

- El logro de la autonomía física del robot móvil (RF)

- Añadir tracción (2 motores) al remolque del prototipo de robot móvil de CINVESTAV-Gdl para probar algoritmos considerando conmutación de control entre ambos carros.

- Probar un proceso adaptativo para el cálculo de las constantes *Kp* y *Kd* del controlador PD-Difuso.

- Llevar a implementación el algoritmo de control de velocidad de las ruedas del robot móvil cuyo objetivo es el seguimiento de trayectorias

- Analizar detalladamente la conveniencia de continuar invirtiendo recursos (humanos, monetarios y tiempo) en el prototipo de robot móvil.

- Concretar el algoritmo de control que resuelve movimiento en reversa.

6.3 Referencias

1. Arabshahi P,, Choi J.J., Marks II R.J., Caudell T.P., *Fuzzy Control of backpropagation, IEEE Int. Conf. on Fuzzy Systems*, San Diego, pp 967-972. 1992.
2. Beaufrere B., Durrant-Whyte H. F., *Inertial navigation system for mobile robots*, IEEE Transactions on Robotics and Manufacturing, 11(3), 328-342, 1995.
3. Borenstein J., Heverett H.R., Feng L., *Navigating Mobile Robots, Systems and Techniques*, A K Peters, Wellesley MA, 1996.
4. Boumedine Montaner M., Ramírez-Serrano A., *Fuzzy knowledge-based controller design for autonomous robot navigation*, Expert Systems with Applications, 14 pp. 179-186, 1998.
5. Brooks Rodney A., *Aspects of mobile robot visual map making,* Robotics Research 2, Hanafusa and Inoue, Eds. Cambridge, MA, M.I.T., 1984, pp. 369-375.
6. Brooks Rodney A., *A Robust Layered Control System for a Mobile Robot*, IEEE Journal of Robotics and Automation, Vol. RA-2, No.1, March 1986.
7. Cassandras C.G., *Discrete Event Systems*, Boston: Richard D. Irwin, 1993.
8. Chapra S.C., Canale R.P., *Métodos Numéricos para Ingenieros, con aplicaciones en PC.*, Mc Graw Hill, 1987.

9. Cox I. J., *Blanche – An Experiment in Guidance and Navigationof an Autonomous Mobile Robot*, IEEE Transactions Robotics and Automations, 7(3), pp. 193-204, 1991.
10. Crowley James L., *Navigation for an intelligent mobile robot*, IEEE J. Robotics Autom., Vol. RA-1 No.1, March 1995, pp 31-41.
11. Driankov D., Hellendoorn H. and Reinfrank M., *An Introduction to Fuzzy Control*, Springer, 1996.
12. Dunlap G. D. and Shufeldt H.H., *Dutton's Navigation and Piloting*, Naval Institute Press, pp. 557-579.
13. Freund E. and Hoyer H. *Real-time pathfinding in multirobot systems including obstacle avoidance*, International Journal of Robotics Research, 7(1), 42-70, 1988.
14. Getting, I.A., *The Global Positioning System, IEEE Spectrum*, Pp 36-47, December 1993.
15. Giralt G., Cahtila R., Vaisset M., *An Integrated navigation and motion control system for autonomous multisensory mobile robots*, Robotics Research 1, Brady and Paul, Eds., Cambridge, MA: M.I.T., 1983, 191-214.
16. Haber R., *Control Borroso*, Instituto Superior Politécnico "Julio Antonio Mella", Santiago de Cuba, 1991.
17. Jamshidi M., Vadiee N. and T. Ross (Eds.) *Fuzzy Logic and Control: Software and hardware applications*. Prentice-Hall, Englewood Cliffs, NJ, 1993.
18. Jamshidi M., Titli André, Zadeh Lofti and Boverie Serge, *Applications of Fuzzy Logic*, Prentice-Hall, 1997.
19. Jarvis R. A., *A Perspective on Range Finding Techniques for Computer Vision*, IEEE Trans. Pattern Anal. Machine Intell. Vol. PAMI-5, 1983.
20. Jones Joseph L., Flynn Anita M., *Mobile Robots, Inspiration to Implementation*, AK Peters, Wellesley Massachusetts,1993.
21. Kanayama Y. *Concurrent Programming of intelligent robots*, Proc. IJCAI, 1983, pp. 834-838.
22. Lancaster P.and Salkauskas K., *Curve and Surface Fitting, An Introduction*, Academic Press, 1987.

23. Lee C. C., *Fuzzy Logic in control systems: Fuzzy Logic Controller Part 1, and Part II*, IEEE Transactions on Systems, Man and Cybernetics, Vol. 20, No. 2, pp. 404-418, 1990.

24. Leonard J. and Durrant-Whyte H. F., *Application of Multi-Target Tracking to Sonar Based Mobile Robot Navigation*, International Conference on Decision and Control, 1990.

25. Liu K. and Lewis F. L., *Fuzzy Logic-based Navigatin Controller for an Autonomous Mobile Robot*, IEEE International Conference on Systems, Man, and Cybernetics, San Antonio Texas, 1994.

26. Malki, H. Li, and G. Chen, *New Design and stability analysis of fuzzy proportional-derivative control systems*, IEEE Trans. On Fuzzy Systems, 1994.

27. Mesarovic M.D., D. Macko, Y. Takahara, *Theory of Hierarchical, Multilevel Systems*, New York: Academic Press, 1970.

28. Moravec H.P., *The Stanford cart and the CMU rover*, Proc. IEEE, Vol. 71, pp. 872-884, July 1983.

29. Nilsson N.J., *Shakey the robot*, SRI AI Center, tech. note 323, Apr. 1984.

30. Pedrycz, Witold, *Fuzzy Control and Fuzzy Systems*, Second Extended Edition,Research Studies Press Ltd., England TA1 1HD. 1993.

31. Rogers David F., Adams J. Alan, *Mathematical Elements for Computer Graphics*, Second Edition, Mc Graw Hill, 1990.

32. Sánchez C. Edgar, Nuño Luis A., Ya-Chen Ilsu and Guanrong Chen, *Fuzzy PD Scheme for Underactued Robot Swing-up Control*, IEEE World Congress on Computational Intelligence 1998.

33. Spong Mark W, and Vidyasagar M.*Robots Dynamics and Control*, John Wiley & Sons, 19989.

34. Sugeno, M. (Ed.), *Industrial Applications of Fuzzy Control.* North Holland, Amsterdam, 1985.

35. Surman H., Huser J. and Peters L., *Fuzzy system for indoor robot navigation*, In Proceedings of the Fourth IEEE International Conference on Fuzzy Systems, Yokohoma, pp. 26-31, 1995.

36. Titli André, LAAS-CNRS and INSA, Touluse, *Memorias del Curso de Lógica Difusa* en CINVESTAV-Guadalajara, 25 de Junio de 1997.

37. Tsuji S., *Monitoring of a building environmente by a mobile robot*, Robotics research 2, Hanafusa and Inoue, Eds. Cambridge, MA: M.I.T., 1985, pp. 349-356.

38. Tunstel E. W., Danny H., Lippicott T. & Jamshidi Mo, *Adaptive Fuzzy-Behavior Hierarchy for Autonomous Navigation,* Int. Conf. on Robotics and Automation, Alburqueque, New Mexico, April 1997.

39. Tunstel Jr. E. W. and Jamshidi M., *Applications of Fuzzy Logic, Intelligent Control and Evolution of Mobile Robot Behavior,* NASA Center for Autonomous Control Engineering, University of New Mexico, USA, 1997.

40. Tunstel E. and Jamshidi Mo, *Embebedded Fuzzy Logic-Based Wall-Following Behavior for Mobile Robot Navigation.,* CAD Laboratory for Robotics and Intelligent Systems Department of Electrical and Computer Engineering University of New Mexico, Alburqueque, NM USA.

41. Vázquez G. B., *Uso de los B-Splines en el modelado de curvas: Aplicaciones al control de un robot móvil*, Tesis para obtener M.S. CINVESTAV-México, 1994.

42. Wang Li-Xin, *Modeling and Control of Hierarchical Systems with Fuzzy Systems. Automatica, Vol. 33 No 6 pp 1041-1053, Great Britain, 1997.*

43. Wang Li-Xin, *Adaptive Fuzzy Systems and Control Design and Stability Analysis*, Prentice-Hall,Englewood Cliffs, NJ, 1994.

44. Yamaguchi F., *Curves and Surfaces in Computer Aided Geometric Design*, Springer-Verlag, 1988.

45. Young S.J., *Real Time Languages: Design and Development.* Chichester: Hellis Horwood, 1982.

46. Zadeh, L.A., *Fuzzy Sets*, Information and Control Vol. 12 pp. 338-353, 1965.

6.4 Bibliografía

- Benveniste A. and Aström K. J. *Meeting the challenge of computer science in the industrial applications of control*, IEEE Trans. Autom. Control AC-38/Automatica 29. 1993.
- Boumedine Montaner M., Ramírez-Serrano A., *Fuzzy knowledge-based controller design for autonomous robot navigation*, Expert Systems with Applications, 14 pp. 179-186, 1998.
- Gupta M.M. and Shina N.K., *Intelligent Control Systems, Theory and Applications*, IEEE Press, New York, 1996.
- Vaneck T. W., *Fuzzy Gidance Controller for an Autonomous Boat*, IEEE Control Systems, April 1997.

6.5 Anexo A. Publicación Internacional

Proceedings of the 1998 IEEE/RSJ
Intl. Conference on Intelligent Robots and Systems
Victoria, B.C., Canada • October 1998

FUZZY LOGIC-BASED REAL-TIME NAVIGATION CONTROLLER FOR A MOBILE ROBOT

Thamar E. Mora and Edgar N. Sánchez

CINVESTAV, Unidad Guadalajara, A.P. 31-438 Guadalajara, Jalisco 44550 Mexico
E-mail: thamar@gdl.cinvestav.mx

Abstract

In this paper a fuzzy logic-based real-time navigation controller for a mobile robot is presented; reactive control provides the mobile robot with the adaptability necessary for coping with a dynamically changing world. The robot makes intelligent decisions based only on current sensory input as it navigates through the world. The algorithm was implemented in C. Real time implementation constitute our main contribution.

Key words: Fuzzy control, navigation, mobile robot and sonar.

I. Introduction

Mobile robots (MR) play a very important role in flexible manufacturing systems. In addition they are applied in construction automations, military missions, harmful-materials handling, hazardous environments, interplanetary explorations, and so on.

The main issues in MR research include mission planning, navigation, maneuvering, and manipulation.

A research aim of intelligent robot control is to develop autonomous mobile robot systems that can dynamically interact with the real world. The notion of intelligence in such systems dictates the need for cognitive or knowledge-based control. System autonomy implies that the mobile robot is to be self-contained. That is, it must be capable of sensing the real world, reasoning about the real world, and physically influencing the real world. In this context, real world is meant to refer to physical environments that may be unstructured, unpredictable, dynamic, noisy, and/or unknown. As such, the real world represents an immense level of uncertainty that complicates the endeavor of developing and controlling autonomous mobile robot systems [6].

Researches are frequently faced with the necessity of considering tradeoffs between developing complex cognitive systems that are difficult to control, or adopting a host of assumptions that lead to simplified models which are not sufficiently representative of the system. Recent research and applications employing non-analytical methods of soft computing such as fuzzy logic, evolutionary computation, and neural networks have demonstrated the utility and potential of these paradigms for intelligent control of complex systems [2].

In the context of mobile robot control, a fuzzy logic-based system has the advantage that it allows the intuitive nature of sensor-based navigation to be easily modeled using linguistic terminology. The computational loads of typical fuzzy inference systems are relatively light. As a result, reactive fuzzy control systems permit intelligent decisions to be made in real time, thus allowing for smooth and uninterrupted motion.

Autonomous navigation of a mobile robot has been studied by many researches [1, 3, 8]. In [10] a sonar-based navigation system for an autonomous mobile robot working in unknown and unstructured environments was developed. The workspace is classified as "probably empty region", "some where occupied region" and "unknown region" based on the interpretation of the data obtained from sonar system. In [8], a stereo-vision system is used to control a mobile robot autonomously operating in a complex, dynamical and previously unknown environment. In [1], a navigational path-planning scheme consisting of a multilevel representation and architecture to support multi-sensor navigation is proposed. This navigational planner provides the mobile base with a path guaranteed to be free of collisions with modeled obstacles.

Although the schemes described above work well in specific environments, the path planning and navigation control, are always treated as two isolated issues. The static path planning strategy does not provide the essential adaptability necessary for coping with unexpected events. The success of the navigation control depends mostly on

0-7803-4465-0/98 $10.00 © 1998 IEEE

the accuracy of absolute measurements of position, velocity, orientation and its rate of change. All of them must be measured in (or can be transformed to) cartesian space [9].

In this paper a real time fuzzy logic-based navigation controller is implemented. It consists of achieving sensor based motion control of a mobile robot among obstacles in structured and/or unstructured environments with collision-free motion as the priority. A fuzzy logic based intelligent control strategy has been developed here to computationally implement the approximate reasoning necessary for handling the uncertainty inherent in the collision avoidance problem.

In following sections a description of the mobile robot architecture, as well as its main characteristics will be presented. Then an explanation of the navigation algorithm is presented. Finally the related conclusions are established.

II. Mobile Robot configuration

Our mobile robot is a didactic prototype developed in CINVESTAV Guadalajara, Mexico. The shape and motion of snakes inspired the design named Active Cord Mechanism in [4]. It is a device, which incorporates the basic elements of the snake as a biological machine; we define it as a functional body which connects in series joint units, which can bend in an animated manner, and which forms a cord.

The mobile robot studied in this research consists of a driven subsystem and a sensing subsystem.

The driving configuration of robot is differential-drive type; has two independent driving wheels arranged parallel to each other (Fig.1). Their speed can be controlled separately. Thus, by appropriately controlling the speed of each driving wheel, this mechanism is able to drive the vehicle forward and backward, as well as to steer its heading angle by differentiating their speed. Even through this configuration requires a somewhat more complex control strategy than the steer-drive configuration, its capability of making small-radius turning, even making a turning on the spot, makes it the first-choice in most researches. The prototype is based in the biological snake shape; control is on the motors coupled to the wheels of the first car. Second car has not motors. Snake-like form provides mechanical stability.

The MIMO control system has six inputs and two outputs. Three inputs correspond to the sonar system, representing the distance of nearest objects. The fourth input is the signal sensed by a potentiometer mounted on the cars joint, which helps to determine the relational position of first car with respect of the second one. Finally, the fifth and sixth input represent encoder lectures. The two system outputs are voltage applied to each motor.

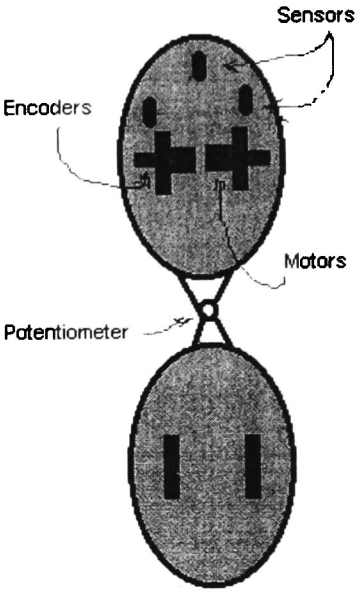


Fig. 1 Mobile Robot Top View

The robot parameters are:

Radius of each wheel:	0.0825 m
Vehicle mass:	0.8 Kg
Dist. between wheels:	0.4 m
Motors resistance:	54 Ohms
Supply Voltage:	12 VDC
Speed:	0-60 RPM
Torque (Full Load):	53.63 oz/in

While we would like our robot to understand and be aware of its environment, in actuality, a robot is limited by the sensors we give it and the software we write for it. Sensing is not perceiving. Sensors are merely transducers that convert some physical phenomena into electrical signals that the microprocessor can read [7].

There exist a variety of sensors for mobile robots, such as near-infrared proximity detectors, sonar

rangefinders, microwave sensors, pyroelectric sensors, earthquake and flood sensors, force sensors, potentiometers, photosensors, bump switches, microphones, bend sensors, gyroscopes, accelerometers, compasses, cameras, etc.

Ultrasonic transducers help the robot detect and avoid obstacles. While a near-infrared detector only delivers proximity information (something is or is not there), a sonar transducer can actually provide distance information because it is possible to measure the time of flight between the initiation of a ping and the return of its echo. By measuring the time of flight and knowing the speed of sound in air, it is possible to calculate distance covered by the round trip of the ping [7].

Two kinds of range-finding devices are available; they are laser range-findings and ultrasonic range-transducers. The structure and configuration of laser range-finding systems are very complicated which makes the system itself very expensive. On the other hand, sonar systems are simple and low-cost, probably multiple orders of magnitude less expensive than laser-based systems.

The advantages of time-of-flight (TOF) systems arise from the direct nature of their straight-line active sensing. The returned signal follows essentially the same path back to a receiver with or in close proximity to the transmitter. In fact, it is possible in some cases for the transmitting and the receiving transducers to be the same device. The absolute range to an observed point is directly available as an output with no complicated analysis required. Furthermore, TOF sensors maintain range accuracy in a linear fashion as long as reliable echo detection is sustained, while triangulation schemes suffer diminishing accuracy as distance to the target increases.

Typical problems associated with ultrasonic sensors are variations in the speed of propagation, uncertainties in determining the exact time of arrival of the reflected pulse, interaction of the incident wave with the target surface, the equipment accuracy/resolution limitations, external interference from nearby sources, and the poor directionality characteristics of sonic waves.

The sensing subsystem mounted in our mobile robot consists of three Herian Proffer HEUF-23 ultrasonic transceivers. If in a reasonable time period there is no reflected signal detected implies that no objects are in the area of interest. Otherwise, the time for round-trip is determined and the distance to the objects is calculated.

In other researches at least 24-30 transducers are needed to detect the whole area surrounding the mobile robot. In this research, three Herian Proffer ultrasonic transceivers are used to detect the environment surrounding the mobile robot. Transducers have 5-15 VDC, 25 mA quiescent current consumption, beam spread of ± 12 degrees, 40 ± 1 Khz frequency and bandwidth, 3.6 x 1.2 x 1.1 inches size and 0.4 to 66 feet of range. The sonar system acquires the data for the distances between the transducers and any possible objects surrounding it. Let $S=\{S[_1], S[_2], S[_3]\}$ be the distance vector, where $S[_1]$ and $S[_3]$ represent the distance measured to the nearest object in mobile robot left and right side respectively, and $S[_2]$ represents the distance to the nearest object in mobile robot front.

Due to the measurement noise, the distance vector S cannot provide us with exact locations of the objects surrounding the mobile robot. However a great deal of information about whether or not there are possible objects in certain directions can be obtained from S, and this information is good enough for the proposed real-time navigation controller.

Implementation of a fuzzy logic-based navigation controller on a mobile robot with two cars constitutes our original contribution. The controller is robust enough to support influence of the second car over the first one. Joint between cars has a potentiometer used to know second car position with respect to the first car; it is useful to activate safe backward movements.

The navigation control tasks can be resumed as follow [9]:

- If there is no obstacle between the car and the goal, drive to the goal by appropriately steering the wheels.

  Adjust the heading angle to be consistent with the docking angle before the car approaching the goal.

  If there is an obstacle on the road, plan and execute a maneuver to go around it and the drive to the goal by appropriately steering the wheels.

The advantages of this strategy are evident. It unites navigation and maneuvering into a single set of algorithms. The knowledge of the system dynamics is not required. The problem consists of deriving a fuzzy rule base that considers robot sensory data as antecedents and infers consequents that specify appropriate motion control actions [11].

III. Real-Time Fuzzy Logic-based Navigation.

Constantly the mobile robot needs moving along a physically constrained path, for instance, a curbed road or a hall in a building. These physical constraints may limit the motion of mobile robot from one side, or from both sides.

Our implementation includes an algorithm using just three sensor data (left, right and front side lectures). Depending on this sensor data, independent reference speeds are sent to both motors, such that the mobile robot moves through a passageway with out hitting obstacles.

The mobile robot starts moving forward describing a straight line. The navigation algorithm works under the following condition [9]:

1) If $S[_2] > Rs$, then continue moving straight.

2) If $S[_2] < Rr$, then brake, take potentiometer lecture and activate moving backward according to potentiometer data. Step 1.

3) If $R_r < L[_2] < R_s$, and
   3.1) If $S[_1] > S[_3]$, then turn left.
   3.2) If $S[_1] < S[_3]$, then turn right.

4) Repeat step 3.

Where:

Rs Safe distance in the front of the mobile robot.
Rr Restricted distance.
$S[_i]$ Sensor i lecture (i=1,2,3).

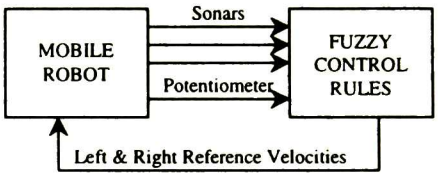
Rules were formed such that maximum speed gets on straight movement. When a steer is needed, the related wheel reduces its angular velocity. It means that, if mobile robot has to turn to right side, right motor has to reduce its velocity a specific amount corresponding to the crisp output of the fuzzy control. The procedure to turn left is similar.

The first step of the algorithm is focused in allowing robot movement when no object is in its trajectory.

The second step becomes when the algorithm detects an object in the safe distance, sends reference velocities equal to zero to the motors, and then takes the last potentiometer lecture. Data obtained by the potentiometer helps to make a decision about the direction of backward movement. Linguistic variables associated to potentiometer lectures are LEFT, RIGHT, and CENTERED, interpreting the second car orientation with respect to the first car. When an object is avoided, straight movement continues.

The third algorithm step works when mobile robot is collisions free. It determines control actions for each wheel depending on the sensing system. We have defined as Δ the difference between left side sensor lecture and right side sensor lecture, i.e. $\Delta = S[_1] \quad S[_3]$. If Δ is positive, the control action acts over left motor; if Δ is negative the control action is send to the right motor. If Δ is zero the controller sends reference velocities such that mobile robot motors have the same velocity (straight movement). Linguistic variables for Δ are the typical NB, NM, NS, ZE, PS, PM and PB; where NB means "Negative Big" and PB means "Positive Big", etc.

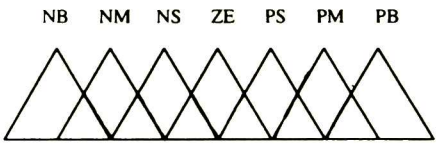
The MIMO fuzzy control has the following structure:



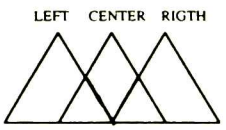
Outputs are the Left Reference Velocity and Right Reference Velocity with linguistic variables NB, NM, NS, ZE, PS, PM, PB.

Membership functions of input and output variables are:

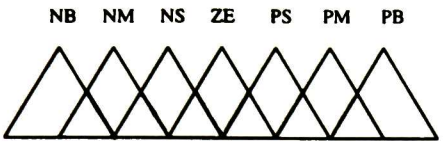
INPUT: $\Delta = S[_1] - S[_3]$



INPUT: Potentiometer Lecture



OUTPUTS: Reference Velocities



The fuzzy control rules are established as:

R1. If Δ is NB then *Velocity* is NB
R2. If Δ is NM then *Velocity* is NM
R3. If Δ is NS then *Velocity* is NS
R4. If Δ is ZE then *Velocity* is ZE
R5. If Δ is PS then *Velocity* is PS
R6. If Δ is PM then *Velocity* is PM
R7. If Δ is PB then *Velocity* is PB

For our fuzzy logic-based controller design at most two rules are fired at the same time. We use the product for the inference and the Peak Value (or Height) Method for the defuzzification [2]. This method uses the individual clipped or scaled control outputs; takes the peak value of each clipped output and builds the weighted sum of these peak values. Thus neither the support nor shape of clipped outputs plays a role in the computation of u^* (crisp output). The Peak Value method is both a very simple and a very quick method. Let c_k be the value on the universe of discourse corresponding to the peak of the output, and f_k the clipped height of the output, then the Peak Value defuzzification method for m rules is formally given by

$$u^* = \frac{\sum_{k=1}^{m} c_k \ f_k}{\sum_{k=1}^{n} f_k}$$

For our application, this formula is evaluated as:

$$u^* = \frac{c_1 f_1 + c_2 f_2}{f_1 + f_2}$$

where indices 1 and 2 identifies the two fired rules.

In order to obtain real-time response of the mobile robot system during implementation process, control rules were simplified. Linguistic values of Δ (input) and reference velocities (outputs) were reduced from seven to four. Algorithms were programmed in language C.

IV Conclusion

In this paper a fuzzy logic-based real-time navigation controller is developed and implemented on a mobile robot with two cars. The fuzzy controller is robust enough to support influence of the second car over the first one. The controller was tested on a mobile robot system in an indoor environment and found to perform satisfactorily despite having minimal sensory feedback, (only three ultrasonic transceivers). The knowledge of the system dynamics is not required. Tight coupling between sensor data and control actions provides the adaptability necessary for copying with a dynamically changing world. The driving mechanism reacts immediately to perceived sensor data as the mobile robot navigates through the world.

V References

1. Arkin, R.C., "Navigational Path Planning for a Vision-based Mobile Robot", Robotica, Vol.7, pp. 49-63, 1989.
2. Driankov D., Hellendoorn H., Reinfrank M., "An introduction to Fuzzy Control", Springer, 1996.
3. Elfes, A., "Sonar-Based Real-World Mapping and Navigation", IEEE Journal of Robotics and Automation, Vol.RA-3, No.3, pp. 249-265, 1987.
4. Hirose S., "Biologically Inspired Robots; Snake-Like Locomotors and Manipulators", 1993.
5. Jamshidi M., Titli A., Zadeh L., Boverie S., "Applications of Fuzzy Logic", pp. 2-21, 1997.
6. Jarvis, R.A. "A Perspective on Range Finding Techniques for Computer Vision", IEEE Trans. Pattern Anal. Machine Intell. Vol. PAMI-5, 1983.
7. Jones J.L., Flynn A. M., "Mobile Robots: Inspiration to Implementation", A.K. Peters, Wellesley. Massachusetts, 1993.
8. Kriegman, D.J., et al., "Stereo Vision and Navigation in Buildings for Mobile Robots", IEEE Trans. Robotics and Automation, Vol.5, No.6, pp. 792-803, 1989.
9. Liu, K. and Lewis F.L., "Fuzzy Logic-Based Navigation Controller for an Autonomous Mobile Robot", IEEE International Conference on Systems, Man and Cybernetics, 1994.
10. Moravec, H.P. and Elfes, A., "High Resolution Maps from Wide Angle Sonar", Proc. IEEE Int. Conf. On Robotics and Automation, 1985.

11. Tunstel, E. and Jamshidi, M. "Embedded Fuzzy Logic-Based Wall-Following Behavior for Mobile Robot Navigation", NAFIPS/IFIS/NASA '94, San Antonio, TX, Dec. 18-20, 1994.

12. Vaneck T.W., "Fuzzy Guidance Controller for an Autonomous Boat", IEEE Control Systems, Volume 7#2,1997.

CENTRO DE INVESTIGACION Y DE ESTUDIOS AVANZADOS DEL IPN
UNIDAD GUADALAJARA

El jurado designado por la Unidad Guadalajara del Centro de Investigación y Estudios Avanzados del Instituto Politécnico Nacional, aprobó la tesis titulada "Control Jerárquico Difuso: Aplicación a la navegación de robots móviles" de la C. **Thamar Elena Mora Monroy** el día 14 de Diciembre de 1998.

Dr. Edgar Nelson Sánchez Camperos
Investigador CINVESTAV 3B
CINVESTAV DEL IPN
Unidad Guadalajara

Dr. Manuel [illegible]ardo Guzmán Rentería
Investigador CINVESTAV 3A
Jefe CINVESTAV DEL IPN
Unidad Guadalajara

Dra. Ofelia Begovich Mendoza
Investigador CINVESTAV 3A
CINVESTAV DEL IPN
Unidad Guadalajara

Dr. Antonio Ramírez Treviño
Investigador CINVESTAV 2A
CINVESTAV DEL IPN
Unidad Guadalajara

Dr. Fernando Lara Rojo
Profesor Investigador
Departamento de Electrónica, Sistemas e Informática
Instituto Tecnologico y de Estudios Superiores
Guadalajara.

CINVESTAV
BIBLIOTECA CENTRAL
SSIT000003824