UNIVERSIDAD POLITÉCNICA DE TULANCINGO

Tesis:

“Control PID para el seguimiento de trayectoria de vehículos autónomos mediante algoritmos bio-inspirados”

Como requerimiento parcial para obtener el título de:

Maestro en automatización y control

Presenta:

Ing. Luis Raúl Sosa Cortés

Asesores Académicos

Mtro. Marco Antonio Hernandez de Ita

Dr. Omar Aguilar Mejia

Tulancingo de Bravo, Hidalgo.

Septiembre 2017

ACTA DE COLOQUIO DE TESIS

En la **UNIVERSIDAD POLITÉCNICA DE TULANCINGO**, el día 30 de agosto de 2017, se reunieron los profesores: **DR. RAFAEL STANLEY NÚÑEZ CRUZ, DR. HIPÓLITO AGUILAR SIERRA, DR. OMAR AGUILAR MEJÍA Y MTRO. MARCO ANTONIO HERNÁNDEZ DE ITA** integrantes del jurado de **PRE-EXAMEN DE GRADO DE LA MAESTRÍA EN AUTOMATIZACIÓN Y CONTROL** del Ing.:

Luis Raúl Sosa Cortés

Que presentó el trabajo de Tesis **titulado: "** **Control PID para el seguimiento de trayectoria de un vehículo autónomo mediante algoritmos bio-inspirados"**

Después de escuchar la presentación y defensa del trabajo de tesis, el jurado examinó y resuelve: Aprobado por Unanimidad

DR. RAFAEL STANLEY NÚÑEZ CRUZ
Presidente

DR. HIPÓLITO AGUILAR SIERRA
Secretario

DR. OMAR AGUILAR MEJÍA
Vocal

MTRO. MARCO ANTONIO HERNÁNDEZ DE ITA
Vocal

Agradecimiento

UN trabajo de maestría puede considerarse como un reto que pocas personas se imponen, este requiere un arduo esfuerzo para su realización por parte del alumno, pero no por menos se requiere del esfuerzo de los catedráticos, que con su conocimiento y su don para enseñar permiten que los alumnos que deciden afrontar este desafío salgan victoriosos.

Por esto quiero agradecer al M. en M. Marco Antonio Hernández de Ita y al Dr. Omar Aguilar Mejía por su dedicación y apoyo en la enseñanza y su interés en la realización de mi trabajo de tesis, también quisiera agradecer al Dr. Rafael Stanley Núñez cruz y al Dr. Hipólito Aguilar Sierra por sus observaciones efectuadas en la elaboración del documento propio de la investigación.

También quiero agradecer a mi familia, por su apoyo incondicional y soporte durante la estancia en el posgrado, este logro es en parte gracias a ellos y por ellos he logrado concluir con éxito este proyecto, no podría sentirme más ameno con la confianza puesta en mi por mi madre la M. en E, Blanca Rosa Guadalupe Cortés Gallardo y la Ing. Mariana sosa cortés.

Por Ultimo, agradezco al Consejo nacional de Ciencia y Tecnología (CONACYT) y a la universidad Politécnica de Tulancingo por su apoyo y patrocinio para la realización de este proyecto de tesis, sin este apoyo, la realización de este trabajo no podría haber sido posible.

RESUMEN

El presente trabajo muestra el control de dos tipos de vehículos autónomos en su seguimiento de trayectorias mediante dos tipos distintos de esquemas de control, esto, aplicado a un robot móvil puede verse compuesto por distintas etapas, las cuales pueden ser el control de los actuadores y el control cinemático, cada uno con distintos desafíos dependiendo de las características de la planta, como puede ser no linealidad en el caso del control cinemático, el cálculo de las ganancias de control es asistido por algoritmos de optimización, debido a la complejidad que se puede presentar en el cálculo de los controles de un robot móvil, los algoritmos de optimización usados basan su búsqueda en el comportamiento de enjambres inteligentes, en este trabajo tres algoritmos son usados, los cuales corresponden a enjambres de murciélagos, de ballenas y el método de polinización de las flores, estos algoritmos tratan de minimizar una función objetivo, sujeta a ciertas restricciones que puede tener la búsqueda, en los problemas de control, pueden ser como el error en estado estacionario y el sobre impulso, tras la implementación de estos algoritmos de búsqueda se realizó la prueba en un prototipo real, para esto se tuvo que implementar un algoritmo de estimación de la posición conocido como odometria para poder obtener la posición del prototipo y así poder aplicar las leyes de control para el seguimiento de trayectorias, finalmente se formulan dos trayectorias, mediante curvas de Bézier y curvas paramétricas para ser usadas como trayectoria de referencia para su seguimiento en el prototipo y en la simulación.

1 ÍNDICE

ÍNDICE DE SÍMBOLOS

ω_r velocidad de la rueda derecha

ω_s velocidad de la rueda izquierda

ω_c velocidad promedio de los neumaticos de la traccion

δ_n aungulo con respecto del centro instantáneo de rotación.

$r(t), e(t), u(t)$ señales de referencia, error, y control.

$\mathbb{R}^n$ un conjunto de números reales

ζ_n número pseudo-aleatorio

Σ sumatoria

$f_i(x)$ Función objetivo (minimización)

$h_j(x)$ Función de restricción (igualdad)

$g_k(x)$ Función de restricción (menor que)

$L(s, \gamma, \mu)$ distribución de levy

x_{max} límite superior del rango de búsqueda

x_{min} límite inferior del rango de búsqueda

λ longitud de onda de la ráfaga de sonidos de murcielagos

v velocidad

f frecuencia de la ráfaga de sonidos de los murciélagos

r razón de las emisiones de pulsos

A_0, A_{min} volumen de la busqueda en el algoritmo del murciélago

f_{max}, f_{min} rango de frecuencias de la ráfaga de sonidos

$\lambda_{max}, \lambda_{min}$ rango de la longitud de onda de la ráfaga de sonidos

β Vector aleatorio arrojado de una distribución uniforme

x_* mejor solución en el algoritmo del murciélago

x_i solución en el algoritmo del murciélago

x_i^t Nueva solución en la iteración t

v_i^t Nueva velocidad en la iteración t

ε número aleatorio,

A^t promedio del volumen de todos los murciélagos en la iteración t

$\vec{D}$ vctor de distancia de los agentes de busqueda a la mejor solucion

$\vec{A}$ y $\vec{C}$ son vectores de coeficientes

$\overrightarrow{X^*}$ vector de posicion de la mejor solución

$\vec{X}$ posición de los agentes de busqueda

$\vec{a}$ Factor que decrece linealmente de 2 a 0

$\vec{r}$ vector de números aleatorion entre 0 y 1

$\vec{X}_{rand}$ una ballena (agente e busqueda) escogido aleatoriamente

b constante que define la forma de la espiral logarítmica

l número aleatorio entre el -1 y el 1 (en el algoritmo de la ballena)

g^* vector de la mejor solución en el algoritmo FPA

x_i^t solución de la partícula i en la iteración t

L distribución de levy, corresponde al paso de los polem

ε distribución aleatoria uniforme de números aleatorios entre 0 y 1

$\Gamma(\lambda)$ Función gamma estándar

x_j^t y x_k^t son polen de distintas flores de la misma especie de planta

x_{robot}, y_{robot} posición del robot con respecto del plano de referencia

ψ ángulo del eje longitudinal del robot con respecto del eje y del plano de referencia

a ángulo que las ruedas delanteras toman con respecto del eje longitudinal del robot

L distancia entre la direccion y la traccion del vehiculo

$\dot{\psi}$ razón de coambio del ángulo con respecto del eje de referencia

$v_{rueda\ trasera}$ velocidad de la traccion del robot

$\dot{x}, \dot{y}$ velocidad en los ejes del robot

ψ_{des} ángulo de referencia

x_{des}, y_{des} posiciones cartesianas de referencia

K constantes de control

V_{des} Velocidad deseada

V_{ref}, V_f Velocidad de referencia para el control de velocidad

R radio instantáneo de la curvatura de la trayectoria del robot

W distancia entre los neumáticos del vehículo

$v_{izquierda}$, $v_{derecha}$ velocidades de los neumáticos del robot diferencial

ω_d, ω_i velocidades de los neumáticos del robot diferencias

v_y velocidad del vehiculo

ω_{ref}, ω_f ángulo de referencia para el control de la direccion

φ ángulo de referencia en el control cinemático

C_i constante de penalización de las restricciones

δ_i variable boleana de insatisfacción de la restricción

i_a corriente de armadura de un motor PMDC

$R_a$resistencia de armadura

$L_a$es la inductancia de armadura

J es el coeficiente de inercia

B es el coeficiente de friccion viscosa

K_e es la constante de fuerza electromotriz

$\frac{K_e}{2\pi}$ constante de par del motor

n velocidad del motor en revoluciones sobre segundo.

$\boldsymbol{b_i}$ vector de puntod de control de la curva de bezier

p_i puntos de control para el eje x

q_i puntos de control para el eje y

$B_{i,n}$ polinomio de Bernstein

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS.

OBJETIVO GENERAL

Implementar diferentes algoritmos de optimización para la sintonización de los niveles de control para distintos tipos de robot móvil en su seguimiento de trayectorias generadas mediante curvas de Bézier.

OBJETIVOS ESPECÍFICOS

- Construcción de dos distintos tipos de robot móvil, siendo estos del tipo Ackerman y Diferencial.
- Implementación de tres algoritmos bio-inspirados como algoritmos de optimización o de búsqueda.
- Diseño de una ley de control para la etapa cinemática del vehículo autónomo.
- Diseño de las leyes de control para los actuadores del robot.
- Implementación de los algoritmos de optimización para el cálculo de las ganancias de control por cada nivel en la jerarquía de control del robot.
- Desarrollo de un algoritmo que permita la localización del robot basado en sus variables articulares.
- Implementación de un algoritmo basado en curvas de Bézier para la generación de trayectorias para el seguimiento des vehículo autónomo.

Capítulo 1

Estado del arte

1.1 Introducción

En este capítulo se definen los conceptos básicos para la realización del seguimiento de trayectorias de un robot móvil, así como una descripción del prototipo, los sistemas de locomoción principalmente usados en los robots móviles con ruedas o vehículos terrestres no tripulados, y una introducción al control moderno.

1.2 Antecedentes

Las aplicaciones de los robots móviles con ruedas son encontradas en áreas como las operaciones militares, la vigilancia, el entretenimiento, la transportación, la exploración interplanetaria y la minería [1], aquellas tareas en que se requiere trasladar algún objeto de un lugar a otro, operar en ambientes desconocidos y aquellas cuya finalidad es el servicio al ser humano, una clase importante de sistemas mecánicos no-holónomicos son los robots móviles con ruedas, por este motivo, se busca controlar a los robots móviles con ruedas de manera que se desplacen de un lugar a otro ejecutando movimientos predeterminados o deseados, para esto , existen varias tareas de control.

El Diseño de controladores para el seguimiento y la estabilización de estos sistemas es desafiante debido a las restricciones no holónomas [2]. A lo largo de varios problemas de control del movimiento de los robots móviles, varios investigadores se concentran en el seguimiento de trayectorias geométricas con un tiempo asociado, llamado seguimiento de trayectoria [3].

Varios trabajos han sido publicados como introducción para el seguimiento de trayectorias de los robots móviles [4-8]. Inicialmente los investigadores diseñaban sus controladores basados en el modelo cinemático [3-10], y posteriormente se ha introducido el seguimiento usando modelos dinámicos.

El control retroalimentado es importante en la práctica ya que la mayoría de los vehículos comerciales no cuentan con sensores de velocidad, el diseño de controles retroalimentados es un desafío por la presencia de incertidumbres paramétricas y no paramétricas en el modelo del sistema, tan bien como las restricciones en el movimiento del robot móvil. Un simple y efectivo controlador Difuso-PID para la cinemática del vehículo es presentado en [11], el controlador propuesto puede resolver el seguimiento de trayectorias para un robot Ackerman con un punto inicial arbitrario. También en [12] se diseñó un controlador para el seguimiento de trayectorias comparando varias soluciones de retroalimentación para la estabilización. Algunos investigadores han diseñado controladores usando el modelo cinemático que incluye deslizamiento y derrape de los neumáticos [13, 14].

En varias investigaciones respecto al seguimiento de trayectorias y sendas se utilizó control jerárquico, que es aquel que contempla o considera distintas etapas de control de un robot móvil,

diversos trabajos se han realizado considerando diversos subsistemas, los cuales pueden ser *control cinemático, control de la etapa de potencia, control de los actuadores.* Algunos de los trabajos que contemplan diversos subsistemas son:

- Considerando únicamente la cinemática de la estructura mecánica:

Las primeras investigaciones acerca de los robots móviles con ruedas contemplaban únicamente el modelo cinemático de la estructura mecánica, por ejemplo, Kanamaya propuso un control cinemático cuya estabilidad se demostró vía una función de Lyapunov, aplicable a diferentes topologías de robots móviles con ruedas no holónomos para el seguimiento de trayectorias [15]. Un control basado en linealización por retroalimentación dinámica para un robot móvil fue presentado por Oriolo en [16], Gu y Hu diseñaron un control RH (Receiding Horizon) que permite resolver los problemas de regulación y seguimiento en robots móviles [17], Defoort reporto controles por modos deslizantes para lograr maniobras de formación de un grupo de robots móviles independientes en [18]. Por otro lado, Wang, presentó un control visual, para el seguimiento de trayectorias en robots móviles diferenciales, sin medición directa de la posición vía un algoritmo adaptativo [19]. También, Wang en [20] desarrollo un control suave variante en el tiempo basado en el método de Lyapunov para la regulación y seguimiento de un robot móvil. Liang diseño un control adaptativo basado en un sistema de visión que no requiere el conocimiento exacto de la posición de una cámara en [21]. Un controlador para el seguimiento de una trayectoria basado en backstepping para un tractor remolque fue reportado por Yuan en [22]. Por último, Chen desarrollo un control cinemático basado en programación evolutiva, y un control dinámico difuso adaptativo por modos deslizantes para la tarea de seguimiento de trayectoria en [23].

- Considerando la cinemática de la estructura mecánica y los actuadores:

Estos trabajos consideran tanto el modelo cinemático de los robots móviles con ruedas, como la dinámica asociada a los actuadores. Espinoza propuso un control optimo adaptativo para los actuadores y un control optimo difuso para el seguimiento de trayectorias asociado a la estructura mecánica del robot móvil con ruedas en [24]. Silvia-Ortigoza presentó un control que no requiere de mediciones mecánicas, basado en la linealización por entrada-salida (para el modelo cinemático del robot móvil), que permite el seguimiento de trayectorias [25]. Un controlador basado en el control cinemático y en redes neuronales adaptativas fue desarrollado por Zuo en [26] [27], Silva-Ortigoza desarrollo un controlador jerárquico de dos niveles, en el nivel alto, propuso un controlador por linealización entrada-salida para la estructura mecánica, y en el nivel bajo un control PI para los motores de CD.

Otros trabajos interesantes que involucran el control jerárquico en los robots móviles con ruedas para solucionar tareas como el seguimiento de sendas, tele-operación, evasión de obstáculos, navegación son presentados en [28-32].

Las diversas etapas de control son logradas con un controlador, generalmente del tipo PID, el controlador proporcional, integral derivativo, ha sido usado por décadas debido a su simplicidad y eficiencia, cerca del 90% de los controladores industriales son construidos de forma similar al PID [33], este juega un rol importante en asegurar una salida óptima del sistema, El controlador tiene tres términos principales que consisten en las constantes K_p, K_i, K_d (ganancia proporcional, ganancia

integral, ganancia derivativa), que definen la estabilidad, el error en estado estable, el sobre impulso, el tiempo de asentamiento y el tiempo de ascenso [34].

Un sistema puede tener un comportamiento pobre o fallar si no es propiamente sintonizado [34], es importante que estos parámetros sean propiamente sintonizados para asegurar una salida óptima. Existen varios métodos analíticos que son utilizados en la selección de estos parámetros, entre los más conocidos se encuentran los métodos de asignación de polos, Ziegler-Nichols y Cohen Coon, los cuales toman un tiempo y costo considerable, también pueden producir un gran sobre impulso, el cual no es deseable [35], por lo tanto, nuevos algoritmos deben ser incorporados para seleccionar unas ganancias óptimas de control, estos algoritmos puedes estar inspirados en la naturaleza, procesos químicos o físicos entre otros, varios trabajos se han realizado donde se usan métodos de búsqueda heurística para la sintonización de estos parámetros, en [36] se utilizaron los algoritmos PSO (Particle Swarm Optimization)y BF (Bacterial Foraging) para la sintonización de un controlador PID para un motor sin escobillas, en [37] el control robusto de un robot manipulador bio-inspirado fue calculado utilizando el Algoritmo del Murciélago. En [38] distintas funciones objetivo fueron utilizadas con el algoritmo PSO para calcular las constantes PID para un regulador de voltaje automático, también para un regulador automático del voltaje, en [39] se compararon los controladores PID de orden fraccional y un PIDD^2 sintonizados con el algoritmo del enjambre caótico de hormigas (CAS), para un sistema interconectado de control de generación térmica se utilizó un controlador PI-PD sintonizado mediante el algoritmo de la polinización [40], en [41] se utilizaron diversas técnicas de inteligencia de enjambre, PSO, FFA (Firefly Algorithm) y CSA (Cucko Search Algorithm), para un sistema seguidor solar.

Estos algoritmos metaheurísticos tales como el algoritmo PSO y el recocido simulado se han vuelto métodos poderosos, para resolver varios problemas de optimización [42-46, 50]. Gran parte de la popularidad de algoritmos metaheurísticos se debe a que estos se basan en conceptos simples y son sencillos de implementar, no requieren información del gradiente, pueden evadir los óptimos locales, y ser utilizados en un rango amplio de problemas a través de distintas disciplinas [51]. La gran mayoría de los algoritmos heurísticos y metaheurísticos emulan el comportamiento de sistemas biológicas o físicos, por ejemplo, el PSO fue desarrollado basado en el comportamiento de enjambre de aves y peces [46, 47] mientras el recocido simulado se basado en el proceso de recocido de metales [48]. El algoritmo del Murciélago nació al combinar las características del algoritmo de las luciérnagas y el algoritmo “Harmony Search” [49].

Otro algoritmo metaheurístico basado en la naturaleza es el algoritmo de la polinización (FPA), desde un punto de vista evolutivo, el objetivo de la polinización es la supervivencia de los mejores y la reproducción óptima de las plantas en términos de números, así como de aptitud. Esto es un proceso de optimización en las plantas, varios factores y procesos de la polinización interactúan para lograr la reproducción óptima de las plantas, esto puede tomarse como inspiración para desarrollar algoritmos de optimización.

Generalmente hablando, los algoritmos basados en enjambres tienen ciertas ventajas sobre los algoritmos basados en la evolución, por ejemplo, los algoritmos basados en enjambres mantienen a los agentes de búsqueda quienes actualizan sus posiciones, mientras que los algoritmos basados en la evolución dan origen a nuevos individuos que heredan cierta información de la generación anterior,

los algoritmos de enjambres inteligentes usualmente incluyen menos operadores que los enfoques evolutivos (selección, cruza, mutación, elitismo) y por lo tanto son más sencillos de implementar.

El tercer algoritmo realizado en este trabajo describe el proceso de caza de la ballena jorobada, la principal diferencia entre este algoritmo y otros como el GWO (Grey Wolf Optimization) [52] es el comportamiento de caza simulado con un agente aleatorio que busca la presa y el uso de una espiral para simular el mecanismo de ataque y diversos operadores para simular la búsqueda de presas, así como un balance entre la exploración y la explotación que le permite escapar de los óptimos locales.

Retomando el seguimiento de caminos, algunas de las tareas más ampliamente utilizadas son [53]: *Regulación, seguimiento de trayectorias y evasión de obstáculos,* de estas formas, la tarea asociada al seguimiento de trayectorias es la más estudiada debido a su importancia práctica. La solución a dicha tarea conlleva generalmente dos aspectos importantes [54, 55]:

- Generación de la trayectoria deseada
- Aplicación del control para el seguimiento de la trayectoria.

De acuerdo con la literatura asociada a la generación de trayectorias, algunos métodos reportados son [56-58]: curvas paramétricas, técnicas de interpolación, polinomios cúbicos, funciones Spline, polinomios de Bézier, polinomios polares y concatenación de rectas con arcos y círculos. De estos el ultimo método es uno de los más simples y comúnmente empleados. Sin embargo, la trayectoria así generada no es continua y tampoco suave entre los segmentos concatenados; originando secuencias de arranque y frenado que ocasionan errores de la posición en el robot móvil.

Para dar la solución a esta problemática, diversos autores han implementado restricciones en la trayectoria deseada utilizando espirales cubicas [59], clotoides [60-62], funciones de costo [63], y curvas de Bézier [64], no obstante, las trayectorias así generadas resultan en formas no complejas, sino en formas simples, lo que ocasiona que la posible aplicación de un robot móvil con ruedas sea limitada; pues en la práctica la trayectoria que un móvil debe seguir es de formas elaboradas. Métodos ampliamente utilizados que permiten generar trayectorias deseadas de formas complejas, son:

- La unión de puntos mediante curvas de Bézier [65]
- La unión de puntos mediante curvas polinomiales [66]

Si bien, los polinomios de Bézier son uno de los métodos más empleados para la unión de puntos o segmentos, gráficos de computadora y animaciones [70], también lo son para la generación de trayectorias. En este contexto, algunos trabajos donde se utilizaron curvas de Bézier para la generación de trayectorias son los siguientes. La planeación de trayectorias en un sistema de robots multi-agente de fútbol se presenta en [67], en [68] se utilizó un polinomio de Bézier de cuarto orden para la generación de los perfiles de curvatura continuos y acotados para generar trayectorias, suaves.

Las curvas de Bézier tienen propiedades útiles para la generación de trayectorias [69], las cuales son:

- Propiedad del punto final:
  El punto inicial y el punto final de una curva de Bézier superponen las características relativas del polígono de control, es decir, superponen el punto inicial y el final con la trayectoria del objeto.

- Continuidad:
  La primera y segunda derivadas de la curva de Bézier son consecutivas, lo que significa que la curva es suave.
- Propiedad de la disminución de la variación:
  El número de intersecciones entre cada línea recta y la curva de Bézier es menor que el número de intersecciones entre esta y el polígono característico (polígono de control). Esta cualidad se conoce como la propiedad de la disminución de variación, la cual muestra que la curva de Bézier tiene menos fluctuaciones que su polígono característico.

Por todas sus propiedades y la posibilidad de generar formas elaboradas con curvas de Bézier, en este trabajo se toman para formular la generación de trayectorias que seguirá el robot móvil.

1.3 Robótica móvil

La robótica móvil es un área de investigación que lidia con el control de vehículos autónomos y semiautónomos, lo que aparta a la robótica móvil de otras áreas como la robótica de manipuladores es el énfasis en los problemas relacionados con el espacio [71].

El estudio de la robótica móvil es un área interdisciplinaria que involucra:

- Ingeniería mecánica: Diseño del vehículo y en particular mecanismos de locomoción.
- Ciencias de la computación: Representaciones, sensado y algoritmos de planeación.
- Ingeniería eléctrica: Integración de los sistemas, instrumentación y comunicaciones.

Aunque varias clases de robots móviles son fundamentalmente vehículos para la investigación y experimentales en naturaleza, un número substancial de robots móviles son desplegados en entornos industriales o domésticos [72].

Los robots móviles son especialmente útiles para tareas que exhiben las siguientes características:

- Tareas en las cuales no hay un operador humano
- Tareas donde en ambiente es inhóspito, donde enviar un humano seria costoso o complejo.
- Tareas donde el ambiente es remoto, tanto que enviar un humano es difícil o toma mucho tiempo, instancias extremas son los dominios que son completamente inaccesibles para los humanos, como las tuberías u otros ambientes pequeños.
- Tareas con ciclos de trabajo muy demandantes, o con un factor de fatiga alto
- Tareas que son altamente desagradables para los humanos

Ciertas aplicaciones involucran más de una de estas características, tomando como ejemplo el de los robots móviles en la minería subterránea, donde el ambiente es peligroso debido a la contaminación y al desprendimiento de la roca en las paredes de la mina, es remoto ya que las profundidades son considerables, otros ambientes con estas características pueden ser nucleares o submarinos [72].

Los Robots móviles, así como el nombre indica, tienen la habilidad de moverse alrededor, estos pueden viajar a través del suelo, en la superficie o debajo del agua y en el aire, en contraste con los manipuladores robóticos que son más comunes en operaciones de manufactura.

Mientras un robot móvil desarrolla sus tareas es importante para su supervisor mantener un conocimiento de su localización y orientación, solo entonces la información sensada puede ser reportada y usada, así, la navegación es esencial, esta también es requerida en el proceso de dirigir al robot móvil a un destino especificado, a lado de la navegación esta la necesidad de estrategias de control estables y eficientes, ambas deben de trabajar juntas, mano a mano.

Ejemplos de robots móviles en instalaciones manufactureras incluyen robots móviles con ruedas usados para transferir material de una estación de trabajo a otra, en las instalaciones hay una línea pintada en el suelo que designa la trayectoria para el robot móvil, ciertos sensores miden los límites de la línea y dan comandos al sistema de dirección que causa que el robot móvil siga la línea, esquemas como estos también pueden ser usados por los robots móviles cuya tarea es hacer revisiones de inventario o chequeos de seguridad en grandes complejos. Aquí la trayectoria del robot es especificada y los sensores adquieren y almacenan la información requerida mientras el robot hace sus rondas.

1.3.1 Mecanismos y computación

Los robots pueden ser considerador de distintas perspectivas, tales como el hardware o el nivel de los mecanismos, los robots pueden ser descompuestos en lo siguiente:

- Una fuente de poder, típicamente basada en baterías.
- Un mecanismo que permita al robot moverse en su entorno, los motores bandas y engranes que le permitan al sistema adquirir movimiento
- Una computadora que controle el robot.
- Un conjunto de sensores con los cuales el robot reúne información del entorno
- Un sistema de comunicación que le permita al robot comunicarse con el operador y otras computadoras.

Los subsistemas básicos en el software de un robot son los siguientes:

- Una unidad de control de movimiento
- Un subsistema de control de los sensores
- Un subsistema de interpretación de los sensores
- Un subsistema que controle la tarea

1.3.2 Principios de locomoción y direccionamiento

Existen dos tipos básicos de dirección usadas en los robots móviles operando en el suelo, para ambos tipos de dirección, el robot móvil puede tener uno o dos neumáticos delanteros, uno de los arquetipos es el de dirección delantera, similar a la de un automóvil, este tipo de dirección presenta desafíos interesantes para el control debido a que tiene un radio de giro no nulo limitado por el largo del robot y el ángulo máximo de la dirección. El otro tipo de dirección involucra control independiente de las ruedas de cada lado, al rotar la rueda izquierda y derecha en direcciones opuestas a la misma velocidad el robot puede girar en un mismo lugar [73].

1.3.3 Robot Ackerman

Cada robot necesita un método para controlar su posición, en el robot con locomoción Ackerman esto se logra con el control del sistema de dirección y del sistema de tracción, la mayoría de los vehículos en servicio actualmente usan dirección delantera, esto representa el tipo de locomoción y las capacidades que tiene el robot para desplazarse.

El problema fundamental en la dirección es permitir que el vehículo viaje a través de un arco de tal forma que todos los neumáticos viajen alrededor de un centro de rotación único, esto es logrado gracias al sistema de dirección Ackerman [74], la figura 1.3.1 muestra los componentes principales del sistema, el final de cada eje tiene un punto de rotación alrededor del kingpin (o perno principal), las juntas conectando los ejes forman un trapezoide, con la base del trapezoide formado por la flecha de actuación y las bielas. La distancia entre los finales de las bielas es menor que la distancia entre los pernos principales.

Las ruedas son paralelas entre si cuando están en posición recta, cuando estas son giradas, la rueda interna (respecto del centro instantáneo de rotación) gira un ángulo mayor que la rueda externa, la figura 1.3.1 también muestra que la disposición cambia con la curvatura de la trayectoria.

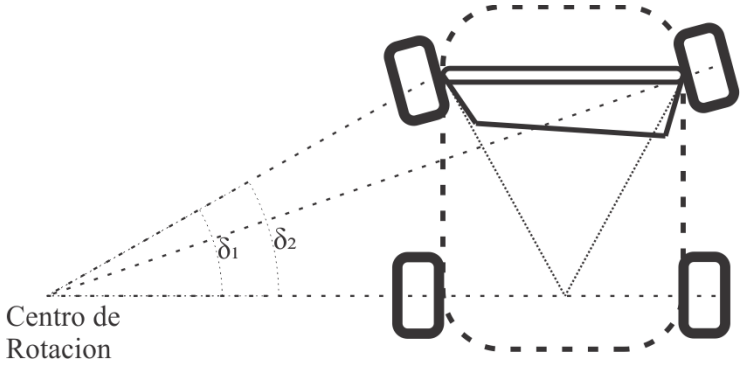


Figura 1.3.1 Arquetipo de un robot Ackerman.

La figura 1.3.2 muestra las uniones requeridas por el sistema de dirección de gusano (o sistema de 4 barras), el brazo de Pitman convierte la energía rotacional del actuador de la dirección en un movimiento de un lado a otro de la barra central. La barra central está unida a los brazos de la dirección por tirantes, y el movimiento angular causa que el eje pivoté alrededor de sus respectivos ángulos de dirección [75].

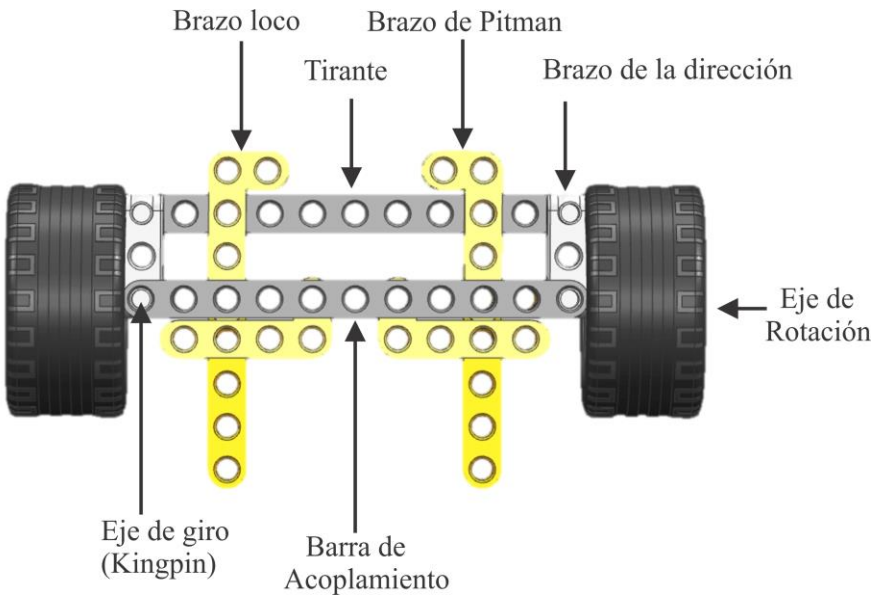


Figura 1.3.2 Sistema de direccionamiento de un robot Ackerman.

Para lograr la dirección Ackerman, la unión de 4 barras debe ser trapezoidal en vez de un paralelogramo.

1.3.3.1 Razón de giro

La razón de giro en los sistemas de dirección es la relación entre el volante o el mecanismo de rotación que comanda la dirección y la posición de los neumáticos del vehículo, esta relación está definida en grados y es lineal, se expresa como la relación entre 2 engranes, eg. 15:1 es una relación aceptada en los vehículos, esta razón provee de un giro de 1° en los neumáticos delanteros cuando el actuador de la dirección gira 15° [79]

1.3.3.2 Tracción

Para poder lograr la locomoción de un vehículo, este debe de disponer de un sistema de tracción, debido a que el vehículo debe ser capaz de realizar curvas se necesita un diferencial, la figura 1.3.1 demuestra la necesidad de un mecanismo diferencial.

1.3.3.3 Diferenciales

Cuando el vehículo en la figura 1.3.1 opera el giro, las 2 ruedas traseras muestran arcos de diferentes radios, así, la rueda al exterior del circulo de rotación tiende a girar más que la rueda interior, si el vehículo tuviese un eje rígido entre las ruedas traseras esto resultaría en un torsión del eje, un fenómeno conocido como “wind up” o “driveline binding”, que significa el desgaste de los componentes del tren motriz, también esto resultaría en un deslizamiento de los neumáticos al realizar un giro, la solución es permitir que cada una de las ruedas gire independientemente, cada una a su propia velocidad, esto es logrado por un diferencial.

Las ecuaciones que definen el comportamiento de un diferencial son [74]:

$$-\frac{N_S}{N_r}=\frac{\omega_r-\omega_c}{\omega_s-\omega_c} \tag{1}$$

En un diferencial el engrane “solar” y el engrane de “anillo” (los 2 unidos a las flechas de la dirección) tienen el mismo número de dientes [74], esto hace que la parte izquierda de la ecuación 1 siempre sea -1, para un diferencial, la ecuación 1 se transforma en:

$$\omega_c-\omega_s=\omega_r-\omega_c \tag{2}$$

Que se convierte en:

$$\omega_c=\frac{\omega_r-\omega_s}{2} \tag{3}$$

Así, la velocidad angular del diferencial es igual al promedio de las velocidades de los ejes, esto permite que el vehículo realice un giro sin “wind up” y sin deslizamiento de sus neumáticos, el mecanismo diferencial utilizado en el prototipo ackerman corresponde a la figura 1.3.3, donde cada uno de los neumáticos posee un actuador independiente.

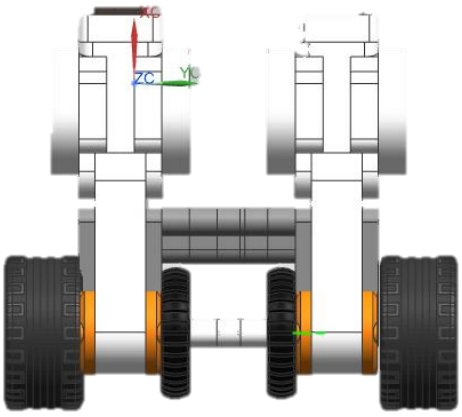


Figura 1.3.3 Mecanismo de tracción del robot móvil.

1.3.4 Robots diferenciales

Otro tipo común de dirección usada en los robots móviles es la locomoción diferencial, ilustrada en la figura 1.3.4 donde las ruedas en un extremo del robot son controladas independientemente de las ruedas en el otro extremo, coordinando las 2 velocidades uno puede inducir que el robot giré en su lugar, se mueva en línea recta, en una trayectoria circular o seguir cualquier otra trayectoria [73].

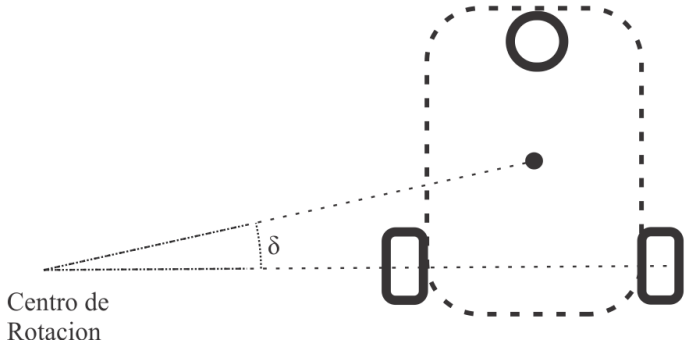


Figura 1.3.4 Arquetipo de un robot diferencial

1.4 Sistemas de control

Un sistema de control es diseñado para hacer que el sistema "haga lo que queremos que haga". Así el diseñador de sistema de control necesita saber el comportamiento deseado o el desempeño esperado del sistema. Las especificaciones del desempeño del sistema de control deben de cubrir ciertas características fundamentales, así como estabilidad, calidad de la respuesta, robustez, A pesar de la gran variedad y riqueza de la teoría de control, más del 90% de los controladores retroalimentados en la práctica son del tipo proporcional-integral-derivativo (PID) [75], debido a su amplio uso en la práctica, el control PID es considerado un tipo de controlador fundamental.

Las decisiones de control pueden ser tomadas por un circuito análogo, en el cual al controlador se le llama *controlador análogo*, o por una computadora digital, en cuyo caso al controlador se le llama *controlador digital*. En el control análogo, las reglas de decisión son diseñadas en la circuitería del controlador. En el control digital, las reglas de control son codificadas en un programa, este programa implementando las decisiones de control es llamado *algoritmo de control digital.* Algunas de las ventajas de un controlador digital sobre un controlador análogo son [76]:

- Incremento en la flexibilidad: los cambios en el algoritmo de control son cuestión de hacer un cambio de software, hacer cambios en el programa del controlador digital es mucho más fácil que cambiar el diseño de un circuito análogo en el control analógico
- Incremento en la capacidad de toma de decisiones: implementar funciones de control no lineal, funciones lógicas de decisión, acciones condicionales y el aprendizaje basado en la experiencia, pueden ser programados. Construir controladores análogos con estas capacidades puede suponer un reto complejo, si no imposible.

Es importante identificar el lugar del control de sistemas dinámicos en el marco de los sistemas de control, varios sistemas de control involucran varios eventos de control discretos usando secuencias y decisiones lógicas, otros sistemas, como el servo control son sistemas que pueden caer dentro de los sistemas de control en tiempo continuo o en tiempo discreto, el servo control en lazo cerrado puede ser un subsistema de un sistema de control lógico o de cualquier otro controlador organizado jerárquicamente.

1.4.1 Componentes de un sistema de control digital

La diferencia entre un controlador digital y uno análogo radica en que el análogo todas las señales son continuas, mientras que en el control digital las señales deben ser convertidas a señales digitales, y las decisiones del controlador deben ser convertidas según el sistema de actuación [77].

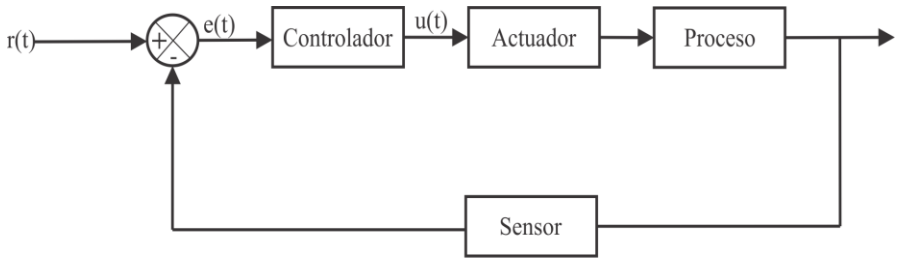


Figura 1.4.1 Diagrama a bloques de un control retroalimentado.

Los componentes básicos de un controlador digital son:

- Una Unidad Central de Procesamiento (CPU) para implementar los algoritmos de control y de lógica, esta es la etapa de toma de decisiones.
- Dispositivos de entrada y salida discretos.
- Convertidores análogos/digitales (A/D) para convertir las señales de los sensores de señales análogas a digitales.
- Convertidores digitales/análogos (D/A) para convertir las decisiones de control tomadas por un algoritmo de control en su CPU a una señal análoga para que pueda comandar a los sistemas de actuación.
- Un reloj que regule la velocidad de la computadora, esto también permitirá ajustar el periodo de muestreo en el cual la computadora obtendrá y procesara la información recabada por los sensores.

Una computadora digital es un dispositivo de eventos discretos, puede funcionar son muestras finitas de la señal, el periodo de muestro puede ser programado basado en la frecuencia del reloj, cada periodo de muestreo, las señales de los sensores son convertidas a una forma digital por el conversor A/D, durante el periodo de muestro, los cálculos del control son efectuados y los resultados deben ser enviados a través del convertidor análogo digital.

1.4.2 Control de lazo cerrado y Control de lazo abierto

Control en lazo abierto significa que las decisiones de control son hechas sin hacer uso de ninguna medida de la respuesta actual del sistema, un lazo abierto no necesita de ningún sensor. Control en lazo cerrado significa que las decisiones de control son hechas basado en las mediciones de la respuesta actual del sistema, este sistema es representado en la figura 1.4.1 La respuesta es retroalimentada al controlador y la decisión de control hecha basada en esta retroalimentación y la

señal deseada del sistema, se deben tomar distintas características del mundo real en consideración, como son [76]:

- Perturbaciones (w(s)): En cualquier sistema siempre hay perturbaciones, generalmente externas, que no están bajo nuestro control, existen y causan errores en la respuesta del sistema, como ejemplo, el viento actúa como perturbación en un aeroplano que intenta cambiar su dirección, así como otras condiciones ambientales a donde yace nuestro sistema de control.
- Las variaciones en la dinámica del proceso: la dinámica del proceso puede cambiar estructuralmente o paramétricamente, los cambios estructurales en la dinámica implican cambios drásticos significativos, tales como el cambio en la dinámica de un avión debido a la pérdida de un motor, mientras que los cambios paramétricos implican cambios menos significativos, más suaves y no tan dramáticos, tales como el cambio en el peso del avión mientras el combustible es consumido o el calentamiento del embobinado de un motor.
- Ruido de los sensores: los sistemas de lazo cerrado requieren las mediciones de la respuesta actual (la variable de control), las señales de los sensores siempre tienen un poco de ruido en la medición. El ruido es incluido en las decisiones de control y tiene efectos en el comportamiento global del sistema.

1.4.3 Tipos de control básicos retroalimentados

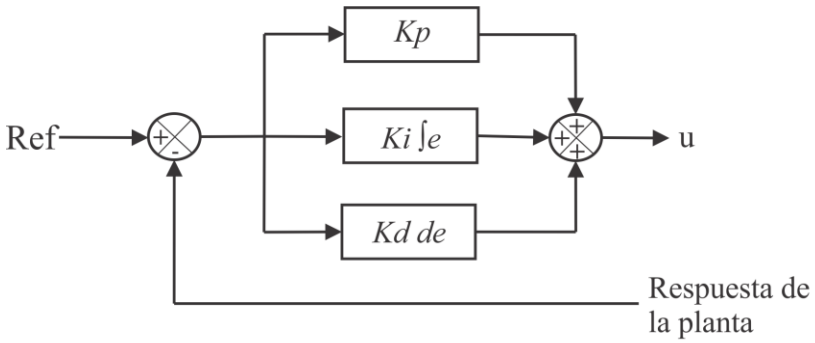


Figura 1.4.2 Acciones básicas de un controlador PID.

La figura 1.4.2 muestra las 3 acciones básicas de control retroalimentado: proporcional, integral y derivativa, la acción proporcional es generada basada en el error actual, la acción integral es generada basado en el error pasado, y la acción derivativa es generada basada en la anticipación del error futuro, la integral del error puede ser interpretada como como la información pasada del error, la derivada del error puede ser interpretada como una medida del error fututo [75].

Asumiendo que la señal del error entrando al bloque de control tiene forma trapezoidal, como se muestra en la figura 1.4.3, las acciones de control generadas por las acciones proporcional, integral y derivativa son mostradas en la figura 1.4.3, El control PID tiene bloques de decisión los cuales toman en cuenta el pasado, el presente, y el futuro, de cierta forma, cubre toda la historia del error, entonces, los controladores más prácticos son PID o alguna forma con las propiedades de un PID [76], el diagrama a bloques es mostrado en la figura 1.4.2 ,el controlador puede ser expresado en tiempo

continuo (análogo) o en tiempo discreto en un microcontrolador, en cualquier instante de tiempo dado *t* la señal de control *u(t)* es determinada por la función [75]:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \dot{e}(t) \tag{4}$$

La cual muestra que la señal de control es una función del error entre el comando y la señal medida $e(t)$ en el tiempo *t*, así como la derivada de la señal del error $\dot{e}(t)$ y la integral del error $\int_0^t e(t)dt$ La aproximación en tiempo discreto del algoritmo de control PID puede ser implementado por la aproximación de diferencias finitas para las funciones integral y derivativa, en la implementación digital, la señal de control puede ser actualizada en intervalos periódicos *T*, también llamados periodos de muestreo, el control es actualizado en múltiplos enteros del periodo de muestreo, el valor de la señal se mantiene constante entre cada periodo de actualización, en cada instante de actualización *k*, el tiempo es *t=kT*, y el instante de actualización anterior es *t-T=kT-T*, el siguiente instante es *t=kT+T*, y así, la señal de control puede ser expresada como *u(t)=u(kT)* [76]:

$$u(kT) = K_p \cdot e(kT) + K_i \cdot u_1(kT) + K_d \frac{(e(kT) - e(kT - T))}{T} \tag{5}$$

$$u_1(kT) = u_1(kT - T) + e(kT) \cdot T \tag{6}$$

$$u_1(0) = 0.0 \tag{7}$$

1.4.4 Traslación de tiempo continuo a tiempo discreto

Un controlador puede ser analizado completamente usando métodos en el tiempo continuo, sin embargo, puede aproximarse a un controlador digital para ser usado en una computadora, la herramienta fundamental es la aproximación de la diferenciación mediante diferencias finitas [76].

Las consideraciones finitas más comunes son:

- Aproximación mediante diferencias hacia adelante (Forward Euler)
- Aproximación mediante diferenciación hacia atrás (Backward Euler)
- Aproximación trapezoidal

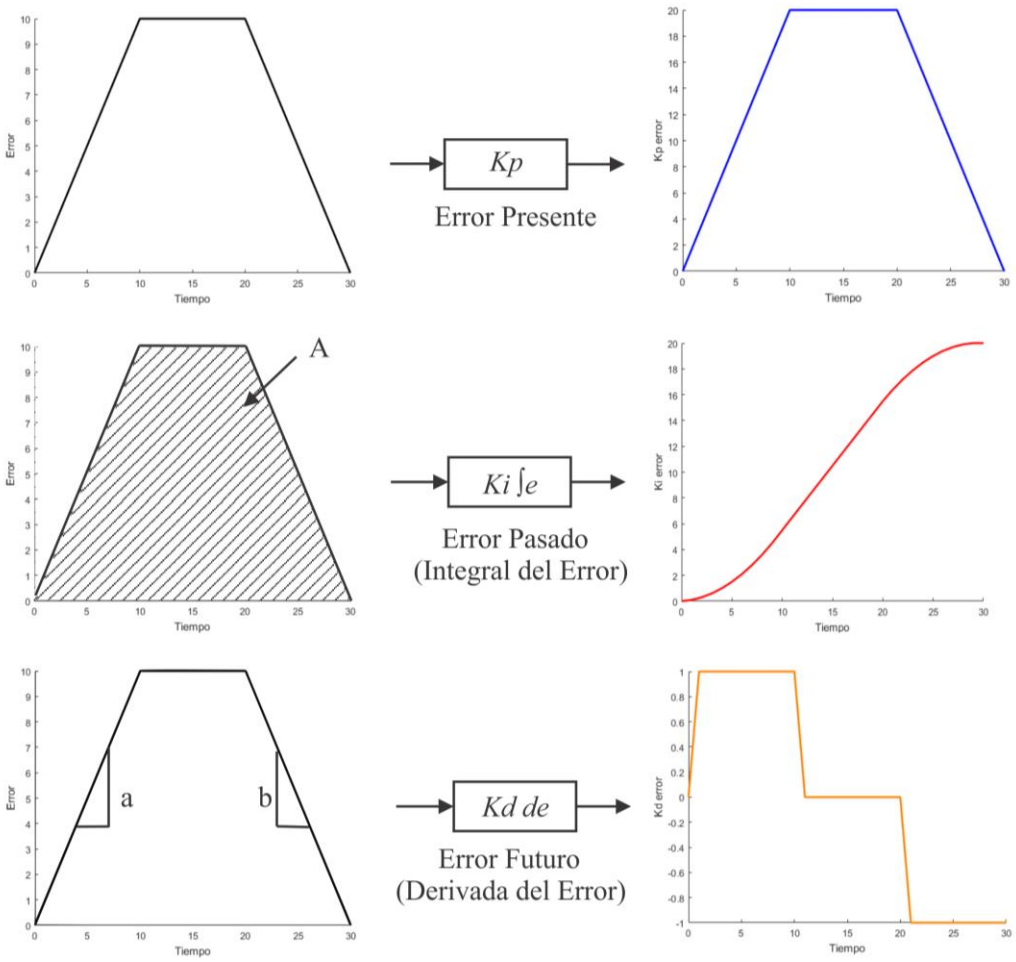


Figura 1.4.3 Respuestas de los componentes de un controlador PID.

1.4.5 Aproximación mediante diferencias finitas

El concepto básico para aproximar sistemas análogos (continuos) a sistemas digitales (discretos) es la aproximación mediante diferencias finitas de la diferenciación y la integración, Considerando una señal de error *e(t)* y su diferenciación e integración, y las muestras del error: $\{\ldots, e(kT-T), e(kT), e(kT+T), \ldots\}$

1.4.5.1 El método de Euler

El método de Euler es el más simple de los métodos numéricos para aproximar la solución de un problema que requiera integración [78] (como el problema del valor inicial) o la integración discreta de una señal de control muestreada por una microcomputadora, definiendo *h* como el paso de integración (o periodo de muestreo), como método para resolver un problema de valor inicial *h* está definida por:

$$h = \frac{x_f - x_0}{n} \tag{8}$$

Donde $x_f - x_0$ es el intervalo de integración y *n* es la cantidad de subintervalos de tamaño *h*, para la integración en un una micro-computadora se escoge este valor como el periodo de muestreo e.g. 0.001 segundos, en un conjunto discreto de puntos $x_0, x_1, x_2, ..., x_n$ del intervalo de interés $\lfloor x_0, x_f \rfloor$, para cualquiera de estos puntos se cumple que:

$$x_i = x_0 + ih \qquad 0 \le i \le n \tag{9}$$

Teniendo los valores de una función, en nuestro caso, una señal muestreada se resuelve que el método de Euler es el siguiente:

$$y_1 = y_0 + (x_1 - x_0) \cdot f(x_0, y_0) = y_0 + h(x_0, y_0) \tag{10}$$

Debido a que es una aproximación existe un pequeño error entre esta y la integral real de $F(x_1)$, este se denomina error de truncamiento, dicho error puede disminuirse tanto como se quiera reduciendo el valor de h, a cambio de un mayor número de cálculos y tiempo de máquina y un error de redondeo más alto, la repetición de este procedimiento a fin de generar una sucesión de aproximaciones es el siguiente

$$\begin{aligned} y_1 &= y_0 + h \cdot f(x_0, y_0) \\ y_2 &= y_1 + h \cdot f(x_1, y_1) \\ &\vdots \\ y_n &= y_{n-1} + h \cdot f(x_{n-1}, y_{n-1}) \end{aligned} \tag{11}$$

1.4.5.2 El metodo

Definimos $h = \dfrac{x_f - x_0}{n}$

Definir *h=periodo de muestreo (paso de integración)*

Hacer *I=1*

While (I <= N)

$$y_0 = y_0 + h \cdot f(x_0, y_0)$$

$$x_0 = x_0 + h$$

$$i = i + 1$$

End

Dicho esto, se puede definir la aproximación de diferencias hacia adelante como:

$$x(t) = x(t) + \dot{x}(t)\Delta t$$

Donde $x(t + \Delta t)$ es el valor de nuestra integral en el tiempo $t + \Delta t$, $x(t)$ es el valor de la integral antes del periodo $t+\Delta t$, y $\dot{x}(t)\Delta t$ es el valor de nuestra variable o función a integrar, en el tiempo t

1.4.6 La plataforma lego

La plataforma lego Mindstorms NXT es un kit de robótica programable lanzado al mercado por Lego en julio del 2006. Este reemplazo la primera generación de kits lego llamada "Robotics Invention System". El kit incluye el software NXT-G de programación u opcionalmente Labview para Lego Mindsotrms. Una variedad de lenguajes no oficiales se ha desarrollado para la plataforma, tales como el NXC, el NBC, lejos, NXJ y RobotC.

1.4.6.1 El ladrillo inteligente

El componente principal del kit es una computadora en forma de ladrillo llamada "Ladrillo inteligente NXT", la cual puede manejar hasta 4 sensores y 3 motores de DC, mediante versiones modificadas de cables RJ12. El ladrillo tiene una pantalla LCD monocromática de 100x60 pixeles y 4 botones que pueden ser usados para navegar en la interfaz del usuario usando menús jerárquicos. Tiene un procesador Atmel de 32 bits AT91SAM7S256 con núcleo ARM7TDMI con 256 Kb de RAM y un microcontrolador Atmel de 8 bits ATmega 48, y soporte para bluethoot. La energía es suministrada por 6 Baterías AA o una Batería de Ion de Litio recargable

1.4.6.2 Programación

Programas muy sencillos pueden ser creados usando el menú en el ladrillo, programas más complicados pueden ser descargados usando un puerto USB o inalámbricamente por bluethoot, estos archivos pueden ser copiados entre ladrillos, también estos pueden ser programados para comunicarse con otros ladrillos mediante bluethoot.

1.4.6.3 Programación utilizando Simulink

Simulink es un ambiente para modelar y simular sistemas dinámicos basado en bloques. Usando este software, el usuario puede diseñar y simular algoritmos de control, y posteriormente descargarlos en el ladrillo inteligente, la paquetería para programar el sistema Lego solo requiere de Simulink.

1.4.6.4 Desarrollo del prototipo

Utilizando la plataforma CAD NX Unigraphics se desarrolló el diseño del prototipo, el cual permitió el análisis de los accionamientos y el posterior ensamblaje de la plataforma de pruebas.

1.5 Referencias:

[1] W. E. Dixon, M. S. de Queiroz, D. M. Dawson, T. J. Flynn, "Adaptive Tracking and Regulation of a Wheeled Mobile Robot with Controller/Update Law Modularity," IEEE Transactions of Control Systems Technology, vol. 12, no. I, pp. 138-147,2004

[2] R. W. Brockett, "Asymptotic stability and feedback stabilization," in Differential Geometric Control Theory, R. W. Brockett, R.S. Milman, H. 1. Sussman Eds., pp. 181-191, Boston, 1983.

[3] A. P. Aguiar, I. P. Hespanha, "Trajectory-Tracking and Path Following of Underactuated Autonomous Vehicles with Parametric Modeling Uncertainty," IEEE Transactions on Automatic Control, vol. 52, no. 8, pp. 1362-1379,2007.

[4] G. Campion, B. d'Andrea-Novel, G. Bastin, "Modeling and State Feedback Control of Nonholonomic Mechanical Systems," Proceedings of the 30th Conference on Decision and Control IEEE, pp. 1184- 1189, England 1991.

[5] C. Samson, K. Ait-Abderrahim, "Feedback control of a nonholonomic wheeled cart in Cartesian space," Proc. IEEE Int. Conf. Robotics and Automation, pp. 1136-1141, 1991.

[6] N. Sarkar, X. Yun, V. Kumar, "Control of mechanical systems with rolling constraint: Application to dynamic control of mobile robots,". International Journal of Robotic Research, vol. 13, no.l, pp. 55-69, 1994.

[7] C. de Wit, H. Khennouf: "Quasi-Continuous Stabilizing Controllers for Nonholonomic Systems: Design and Robustness Considerations," Proceedings of the European Control Conference, pp.2630-2635, 1995.

[8] Y. Kanayama, Y. Kimura, F. Miyazaki, T. Nogushi, "A stable tracking control method for an autonomous mobile robot," Proceedings of IEEE International conference on robotics and automation, pp. 384-389, 1990.

[9] Z. Jiang, H. Nijmeijer, " Tracking Control of Mobile Robots: A case study in Backstepping," Automatica, vol. 33, no. 7,pp. 1393-1399.

[10] R. Fierro, F. L. Lewis "Control of a nonholonomic mobile robot: backstepping kinematics into dynamics," Proc. 34th Conf. Decision and Control, pp. 3805-3810, LA, USA, December 1995.

[11] H. Talebi ABATari, A. Dehghani Tafti ," Using a Fuzzy PID Controller for the Path Following of a Car-like Mobile Robot,"Robotics and Mechatronics, pp. 189 - 193, 2013.

[12] A. D. Luca, G. Oriolo, C. Samso, "Feedback control of a nonholonomic car-like robot," in Robot Motion Planning and Control, ser. Lecture Notes in Control and Information Sciences, I. Laumond, Ed. Springer Berlin, vol. 229, pp. 171-253,1998.

[13] C. B. Low, D. Wang, "GPS-Based Path Following Control for a Car Like Wheeled Mobile Robot With Skidding and Slipping," IEEEControl Systems Technology, vol. 16, pp. 1063-6536, March 2008

[14] F. Hamerlain," Trajectory tracking control of a car-like mobile robot in presence of sliding,., IEEE Conf. on Control, pp. 502-507,2012.

[15]Y.Kanayama, Y.Kimura, F.Miyazaki y T.Noguchi " A stable tracking control method for an autonomous mobile robot" in *Proceedings IEEE International Conference on Robotics and Automation 1990*, Cincinnati, OH, USA, May 12-18 1990, vol.1 pp. 384-389.

[16] G.Oriolo, A. De Luca, and M. Vendittellu "WMR control via feedback linearization: Design, Implementation and experimental validation" *IEEE Transactions on Control Systems Technology*, vol.10, no. 6, pp. 835-852, 2002.

[17]D.Gu and H.Hu "Receding horizon tracking control of wheeled mobile robots" *IEEE Transactions on Control Systems Technology*, vol. 14, no. 4, pp. 743-749, 2006

[18] M.Defoort, T.Floquet, A.Kökösy, and W. Perruquetti "Sliding mode formation control for cooperative autonomous mobile robots" *IEEE Transactions on Industrial Electronics*, vol. 55, no 11, pp 3922-2953, 2008

[19] K.Wang, Y.Liu and L.li "visual servoing trajectory tracking of nonholonomic mobile robots without direct position measurement" *IEEE Transactions on Robotics*, vol. 30, no 4, pp.1026-1035,2014.

[20] Y.Wang, Z.Miao, H. Zhong and Q.Pan, "simultaneous stabilization and tracking of nonholonomic mobile robots: A Lyapunov-based approach" *IEEE Transactions on Control Systems Technology*, vol. 23, no.4, pp. 1440-1450,2015.

[21] X.liang, H.Wang, W.chen, D.Guo and T.liu "adaptative image-based trajectory tracking control of wheeled mobile robots with an uncalibrated fixed camera" *IEEE transactions on Control Systems Technology*, vol. 23, no. 6, pp. 2266-2282, 2015.

[22] J.Yuan, F.Sun, and Y. Huang "Trajectory generation and tracking control for double-steering tractor-trailer mobile robots with on-axle hitching" *IEEE Transactions on industrial Electronics*, vol.62, no.12, pp.7665-7677, 2015.

[23] C.Y.Chen, T.H.S.Li and Y.C.Yeh "EP-based kinematic control and adaptative fuzzy sliding mode synamic control for wheeled mobile robots" *information Sciences*, vol.179, no 1-2,pp.180-195,2009.

[24] F.Espinosa, E.lópez, R.Mateos, M.Mazo, and R.Garcia "Advanced and Intelligent control techniques applied to the drive control and path tracking systems on a robotic wheelchair", *Autonomous Robots*, vol. 11, no.2, pp.137-148,2001.

[25] R. Silva-Ortigoza, G. Silva-Ortigoza, V.M. Hernández-Guzmán, V.R.Barrientos-Sotelo, J.M. Albarrán-Jimenez, and V.M. Silva-García, "Trajectory tracking in mobile robot without using velocity measurements for control of wheels" *IEEE Latin America Transactions*, vol.6, no.7, pp.589-607, 2008. [Online]. Available: http://dx.doi.org/10.1109/TLA.2008.4917431

[26] Y.Zuo, Y.Wang, X.Liu, S.Yang, L.Huang, X.Wu, and Z.Wang "Neural network robust control for nonholonomic mobile robot including actuator dynamics" *International Journal of Innovative Computing Information and Control,* vol. 6, no. 8, pp. 3437-3449,2010.

[27]R.Silva-Ortigoza, C.Márquez-Sánchez, M.Marcelino-Aranda, M.Marciano-Melchor, G.Silva-Ortigoza, R.Bautista-Quintero, E.R. Ramos-Silvestre, J.C. Rivera-Diaz and D.Muñoz-Carrillo, "Construction of a WMR for trajectory tracking control: Experimental results" *The scientific World Journal,* vol 2013, pp. 1-17, 2013, [Online] Available: http://dx.doi.org/10.1155/2013/723645

[28] F. Gonzales, J.G. Guarnizo, and G.Benavides, "Emulation system for a distribution center using mobile robot, controlled by artificial vision and fuzzy logic" *IEEE Latin American Transactions,* vol.12, no. 4, pp.557-563,2014.

[29] F. Penizzoto, E.Slawiñske and V.mut, "Analysis and expectation of Mobile robot teleoperation system over internet" *IEEE Latin American Transactions,* vol 12, no. 7, pp. 1191-1198, 2014 [Online].Available: http://www.ieee.org/reg/9/etrans/esp/publicaciones.php

[30] D.L.A.Ojeda, Y.G.Vera, and M.A.I. Manzano, "obstacle detection and avoidance by a mobile robot using probabilistic models" *IEEE Latin American Transactions,* vol.13, no. 1, pp. 69-75, 2015 [Online] Available: http:// www.ewh.ieee.org/reg/9/etrans/esp/publicaciones.php

[31] F. Penizzoto, E.Slawiñske and V.mut, "laser radar based autonomous mobile robot guidance system for olive groves navigation" *IEEE Latin American Transactions, no*. 13, no.5, pp.1303-1312, 2015 [Online] Available: http://www.ewh.ieee.org/reg/9/etrans/esp/publicaciones.php

[32] D. Reyes, H. Millán, R.Osorio, and G.Lafrane, "Mobile robot navigation assisted by GPS" *IEEE Latin American Transactions,* vol 13, no. 6, pp.1915-1920.2015 [Online] Available: http://www.ewh.ieee.org/reg/9/etrans/esp/publicaciones.php

[33] Dashti, M., Shojaee, K., Seyedkashi, S. M. H., & Behnam, M. (n.d.)." Tuning of digital PID controller using particle swarm optimization", 3383–3389. Retrieved from http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=5572133

[34] Ang, K. H., Chong, G., & Li, Y. (2005). PID control system analysis, design, and technology. IEEE Transactions on Control Systems Technology, 13(4), 559–576. http://doi.org/10.1109/TCST.2005.847331

[35] De Almeida, G. M., e Silva, V. V. R., Nepomuceno, E. G., & Yokoyama, R. (2005). Application of Genetic Programming for Fine Tuning {PID} Controller Parameters Designed Through {Ziegler-Nichols} Technique. In Advances in Natural Computation, First International Conference, ICNC 2005, Proceedings, Part III (Vol. 3612, pp. 313–322). http://doi.org/doi:10.1007/11539902_37

[36] H.E.A. Ibrahim, F.N. Hassan, Anas O. Shomer. "Optimal PID control of a brushless DC motor using PSO and BF techniques", Ain Shams Engineering Journal, 2014.

[37] M. Rahmani, A.Ghanbari, M.M Ettefagh. "Robust adaptive control of a bio-inspired robot manipulator using BAT algorithm", Expert Systems With Applications 2016

[38] M. A. Sahib, B. S. Ahmed, “A new multiobjective performance criterion used in PID tuning optimization algorithms”, Journal of Advanced Research (2016)7, 125–134.

[39] Mouayad A. Sahib, “A novel optimal PID plus second order derivative controller for AVR system”, Engineering Science and Technology, an International Journal 18 (2015) 194e206.

[40] P.Dash, L.C. Saikia, N. Sinha, “Flower Pollination Algorithm Optimized PI-PD Cascade Controller in Automatic Generation Control of a Multi-area Power System”, Electrical Power and Energy Systems 82 (2016) 19–28.

[41] M. M. Sabir, T. Ali “Optimal PID controller design through swarm intelligence algorithms for sun tracking system”, Applied Mathematics and Computation 274 (2016) 690–699.

[42] Deep, K., Bansal, J. C.: Mean particle swarm optimisation for function optimisation. Int. J. Comput. Intel. Studies, 1, 72-92 (2009).

[43] Geem, Z.W., Kim, J. H., Loganathan, G. V.: A new heuristic optimization algorithm: Harmony search. Simulation, 76, 60-68 (2001).

[44] Goldberg, D.: Genetic Algorithms in Search, Optimiztion and Machine Learning. Addison-Wesley, Reading, MA, (1989).

[45] Holland, J. H.: Adapation in Natural and Artificial Systems. University of MichiganPress, Ann Arbor, (1975).

[46] Kennedy, J. and Eberhart, R.: Particle swarm optimization, Proc. IEEE Int. Conf.Neural Networks. Perth, Australia, 1942-1945 (1995).

[47] Kennedy, J. and Eberhart, R., Swarm Intelligence. Academic Press, (2001).

[48] Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P.: Optimization by simulated annealing. Science, 220, 671-680 (1983).

[49] X.-S. Yang, A New Metaheuristic BAT-Inspired Algorithm, in: Nature Inspired Cooperative Strategies for Optimization (NISCO 2010) (Eds. J. R. Gonzalez et al.), Studies in Computational Intelligence, Springer Berlin, 284, Springer, 65-74 (2010).

[50] Mitchell, M.: An Introduction to Genetic Algorithms. MIT Press, (1998).

[51] S. Mirjalilia, A. Lewis “The Whale Optimization Algorithm”, Advances in Engineering Software 95 (2016) 51–67

[52] S. Mirjalili , S.M. Mirjalili, A. Lewis, “ Grey wolf optimizer”. Advances in Engineering Software, 2014;69:46–61

[53] R. S. Ortigoza, M. M. Aranda, G. S. Ortigoza, V. M. H. Guzmán, M. A. M. Vilchis, G. S. González, J. C. H. Lozada, and M. O. Carbajal, “Wheeled mobile robots: A review,” IEEE Latin Am. Trans., vol. 10, no. 6, pp. 2209–2217, 2012.

[55] J. S. Kim and B. K. Kim, “Minimum-time trajectory generation algorithm along curved paths for mobile robots with a motor control input constraint,” in Proc. 2012 IEEE/SICE Int. Symp. Syst. Integr., Fukuoka, Japan, Dec. 2012, pp. 224–229.

[56] A. Ollero BATurone, Robótica: Manipuladores y Robots Móviles. Barcelona, Spain: arcombo-Boixareu Editores, 2001.

[57] A. Segovia, M. Rombaut, A. Preciado, and D. Meizel, “Comparative study of the different methods of path generation for a mobile robot in a free environment,” in Proc. 5th Int. Conf. Adv. Robot. 1991. Robots in Unstructured Environments, Pisa, Italy, Jun. 1991, pp. 1667–1670.

[58] M. Nakamura and S. Yuta, “Trajectory control of trailer type mobile robots,” in Proc. 1993 IEEE/RSJ Int. Conf. Intell. Robot. Syst., Yokohama, Japan, Jul. 1993, pp. 2257–2263.

[59] M. Tounsi and J. F. Le Corre, “Trajectory generation for mobile robots,” Math. Comput. Simul., vol. 41, no. 3–4, pp. 367–376, 1996.

[60] A. Scheuer and Th. Fraichard, “Planning continuous-curvature paths for car-like robots,” in Proc. 1996 IEEE/RSJ Int. Conf. Intell. Robot. Syst., Osaka, Japan, Nov. 1996, pp. 1304–1311.

[61] A. Scheuer and Th. Fraichard, “Continuous-curvature path planning for car-like vehicles,” in Proc. 1997 IEEE/RSJ Int. Conf. Intell. Robot. Syst., Grenoble, France, Sep. 1997, pp. 997–1003.

[62] D. H. Shin and S. Singh, “Path generation for robot vehicles using composite clothoid segments,” The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA, Tech. Rep. CMU-RI-TR-90-31, Dec. 1990.

[63] Y. Kanayama and B. I. Hartman, “Smooth local path planning for autonomous vehicles,” in Proc. 1989 IEEE Int. Conf. Robot. Autom., Scottsdale, AZ, USA, May 1989, pp. 1265–1270.

[64] J.-H. Hwang, R. C. Arkin, and D.-S. Kwon, “Mobile robots at your fingertip: Bezier curve on-line trajectory generation for supervisory control,” in Proc. 2003 IEEE/RSJ Int. Conf. Intell. Robot. Syst., Las Vegas, NV, USA, Oct. 2003, vol. 2, pp. 1444–1449.

[65] J.-W. Choi, R. E. Curry, and G. H. Elkaim, “Continuous curvature path generation based on Bézier curves for autonomous vehicles,” IAENG Int. J. Appl. Math., vol. 40, no. 2, pp. 91–101, 2010.

[66] K. KawaBATa, L. Ma, J. Xue, S. Yokota, Y. Mitsukura, and N. Zheng, “Iterative trajectory generation for mobile robot to pass through the waypoints,” in Proc. 10th Asian Control Conf., Kota Kinabalu, Malaysia, Jun. 2015, pp. 1–6.

[67] K.G. Jolly, R. S. Kumar, and R. Vijayakumar, “A Bezier curve based path planning in a multi-agent robot soccer system without violating the acceleration limits,” Robot. Auton. Syst., vol. 57, no. 1, pp. 23–33, 2009.

[68] C. Chen, Y. He, C. Bu, J. Han, and X. Zhang, “Quartic Bézier curve based trajectory generation for autonomous vehicles with curvature and velocity constraints,” in Proc. 2014 IEEE Int. Conf. Robot. Autom., Hong Kong, China, May 2014, pp. 6108–6113.

[69] Y. Linquan, L. Zhongwen, T.Zhonghua, L.Weixian, “Path Planning Algorithm for Mobile Robot Obstacle Avoidance adopting Bezier Curve Based on Genetic Algorithm”, *Chinese Control and Decision Conference*, CCDC 2008.

[70] Sederber, T.W., “Computer aided geometric design,” CAGD Course Notes, Brigham Young University, Provo, UT, 2007

[71] A. S. Matveev, A.V.Savkin, M. Hoy, C. Wang. “Safe Robot Navigation Among Moving and Steady Obstacles”, 2016 Elsevier Inc. ISBN:978-0-12-803730-0.

[72] G. Dudek, M. Jenkin “Computational principles of mobile robotics 2nd ed.”, Cambridge University Press 2010, ISBN 978-0-521-87157-0.

[73] G. Cook “Mobile robots: navigation, control and remote sensing”, Institute of Electrical and Electronics Engineers, 2011, ISBN 978-0-470-64021-1.

[74] R. Stone, J. K. Ball, “Automotive engineering fundamentals” ISBN 0-7680-0987-1, TL240.S853 2004

[75] K. Ogata. “Ingeniería de Control Moderna”, Pearson Educación, S.A.Madrid, 2010, ISBN: 978-84-8322-660-5

[76] Cetinkunt, Sabri.” Mechatronics with experiments, Second edition” ISBN 987-1-118-80246-5, TJ163.12.C43 2015

[77] K. Ogata. “Sistemas de Control en Tiempo Discreto”, Prentice Hall, Hispanoamerica S.A., 21996, ISBN: 968-880-539-4.

[78] A. Nieves, F. C. Dominguez, “Métodos Numéricos, aplicados a la ingeniería “, ISBN 970-24-0258-1 (primera edición)

[79] D. Knowles, “Classroom Manual for Automotive Suspension and Steering Systems, Fifth edition”, Delmar, Cengage Learning, 2011, ISBN-13: 978-1-4354-8111-2.

Capítulo 2

Optimización

2.1 Introducción

En este capítulo se mencionan tres métodos heurísticos que actualmente son utilizados en problemas de optimización, su inspiración, las ecuaciones y la secuencia que da origen a los algoritmos, así como una comparación entre ellos al final del capítulo.

2.2 Optimización

La optimización es una ciencia con múltiples campos de aplicación, desde aplicaciones de ingeniería y ciencias de la computación hasta las finanzas y la toma de decisiones, la optimización es un paradigma importante en si misma con un gran rango de aplicaciones, en casi todas las aplicaciones de ingeniería e industria siempre se trata de optimizar algo, ya sea minimizar el costo y el consumo de energía o maximizar los ingresos, las salidas, o la eficiencia [1]

La optimización matemática es el estudio de tales problemas de planeación y diseño usando herramientas matemáticas [2], es posible escribir la mayoría de los problemas de optimización utilizando las ecuaciones 1,2 y 3:

$$\min_{x \in \Re^n} f_i(x), (i = 1,2,...,M) \tag{1}$$

Sujeto a:

$$h_j(x) = 0 \quad (j = 1, 2, ..., J) \tag{2}$$

$$g_k(x) \leq 0 \quad (k = 1, 2, ..., K) \tag{3}$$

donde M es el número de funciones objetivo a ser optimizadas $f_i(x)$ es la función objetivo representando el objetivo *i* y *x* es la variable de diseño a buscar por el algoritmo de optimización y su tamaño es n, $\Re^n$ es el espacio abarcado por las variables de decisión y el llamado espacio de diseño o espacio de búsqueda, las igualdades $h_j(x)$ y las desigualdades $g_k(x)$ son llamadas restricciones

Si *M=1* se llama optimización de un solo objetivo, si *M > 1* se le denomina optimización multi-objetivo, los problemas de optimización pueden ser clasificados según su linealidad.

Si la función y las restricciones son no lineales al problema se le llama "problema de optimización no lineal" cuando solo las restricciones son no lineales se le conoce como un problema de restricciones no lineales, y cuando las restricciones son lineales se le conoce como un problema linealmente restringido, en algunos problemas de optimización, algunas variables están linealmente relacionadas con la función objetivo mientras que otras no lo están.

La clasificación de un algoritmo de optimización puede ser tomada de varias maneras, un camino simple es ver la naturaleza de los algoritmos, esto los divide en 2 categorías, algoritmos deterministas y algoritmos estocásticos [2], los algoritmos deterministas siguen procedimientos rigurosos y el camino y los valores de las variables de diseño y la función es repetible.

Los algoritmos estocásticos tienen cierta aleatoriedad, así las soluciones intermedias y las soluciones finales difieren entre simulaciones, La Optimización mediante enjambre de partículas (PSO) es un ejemplo de un algoritmo sin repeticiones, basado en el número de agentes usado en la búsqueda pueden ser clasificados en agente simple o multi-agente, el recocido simulado es un algoritmo se agente simple inspirado en procesos químicos [3], el PSO es multi-agente [1].

Dependiendo de las fuentes de inspiración, pueden ser clasificados como bio-inspirados, inspirados en la naturaleza y meta-heurísticas en general.

Heurística significa por ensayo y error, y meta-heurística puede ser considerado un método de alto nivel por el uso de mecanismos de selección e intercambio de información[1], Los algoritmos meta-heurísticos son métodos poderosos para resolver varios problemas de optimización que son complejos de resolver con métodos convencionales, como el método de descenso en gradiente, problemas por ejemplo que tienen funciones objetivo o restricciones no lineales, o que incluyen demasiadas variables, los problemas de control caen dentro de este rango. Una clase especial de algoritmos ha sido desarrollada dentro de los algoritmos bio-inspirados, esta llamada "inteligencia de enjambre" (Swarm Intelligence), algunos algoritmos bio-inspirados pueden ser llamados de inteligencia de enjambre, como ejemplo están Cuckoo Search[1], El algoritmo del murciélago [1], y la Colonia artificial de abejas[4]

2.2.1 Algoritmos basados en enjambres inteligentes

La inteligencia de enjambre compete el comportamiento colectivo de múltiples agentes siguiendo reglas simples[5], cada agente puede ser considerado como un agente no-inteligente mientras todo el sistema de múltiples agentes muestran cierto comportamiento organizado, y así se pueden comportar de cierta forma como una inteligencia colectiva, varios algoritmos han sido descritos tomando inspiración de la inteligencia de enjambre de la naturaleza[5], las principales propiedades para un algoritmo basado en enjambres inteligentes pueden ser resumidas en:

- Los múltiples agentes intercambian información entre si
- Los agentes se auto-organizan y co-evolucionan
- El enjambre es altamente eficiente en su co-aprendizaje
- Puede ser paralelizado de forma sencilla para problemas prácticos y en tiempo real

2.2.2 Variables aleatorias

Una variable aleatoria puede ser considerada como una expresión cuyo valor es la realización o salida de eventos asociados a un proceso aleatorio [2] una variable aleatoria es una función la cual mapea eventos a números reales, el dominio de este mapeo es llamado espacio muestra, cada variable aleatoria es representada por una función de probabilidad para expresar la distribución de la probabilidad, las variables aleatorias son las piedras angulares de todos los algoritmos estocásticos de optimización. A continuación, se muestran las distribuciones empleadas en este trabajo.

2.2.3 Distribución uniforme

La distribución uniforme [6] es un caso muy simple representado con la ecuación 4:

$$f(x;a,b) = \frac{1}{b-a}\ para: a \le x \le b \tag{4}$$

Donde *a,b* son los límites del rango de la distribución, y es fijo entre [0, 1], la distribución uniforme tiene un valor de expectación de $E(x) = (a+b)/2$ y una varianza de $\frac{(b-a)^a}{12}$

2.2.4 Distribución de Lévy

La distribución de Lévy [2] es la distribución de la suma de N variables de la distribución aleatoria, una forma simple de una distribución de lévy puede ser definida con:

$$L(s,\gamma,\mu) = \begin{cases} \sqrt{\frac{\gamma}{2\pi}}\, exp\left(-\frac{\gamma}{2(s-\mu)}\right)\left(\frac{1}{(s-\mu)^{1.5}}\right) & si\ 0 < \mu < s < \infty \\ 0 & de\ otra\ forma \end{cases} \tag{5}$$

Donde $\mu > 0$ es el mínimo paso y γ es el parámetro de la escala y si $s \to \infty$ tenemos la siguiente forma

$$L(s,\gamma,\mu) \approx \sqrt{\frac{\gamma}{2\pi}}\,\frac{1}{s^{1.5}} \tag{6}$$

2.2.5 Generación de números aleatorios

Las simulaciones que requieren números aleatorios son críticas en campos como las comunicaciones, los modelos financieros y la optimización, un amplio rango de generadores de números aleatorios han sido descritos en la literatura [7], utilizan principios matemáticos básicos, generalmente involucrando transformaciones de números aleatorios uniformes.

2.2.6 Técnica Aceptación/Rechazo

Una técnica útil es la técnica de aceptación y rechazo, donde escogemos $f_{\max}$ para ser más grande o igual que $f(x)$ en el intervalo entre $x_{\max}$ y, $x_{\min}$ se procede así:

- Genera un par de números pseudo-aleatorios ζ_1, ζ_2
- Calcula *x* como $x = x_{\min} + \zeta_1(x_{\max} - x_{\min})$.
- Determina *y* como $y = f_{\max} \cdot \zeta_2$
- Si $y - f(x) > 0$ rechaza y ve a 1, de otra forma acepta x como un número pseudo aleatorio para $f(x)$

La eficiencia de este método depende del valor promedio entre $\frac{f(x)}{f_{max}}$ en el intervalo, si este valor está cerca de 1, el método es eficiente.

2.2.7 Caminata aleatoria

La caminata aleatoria es un proceso aleatorio que consiste en tomar una serie de pasos aleatorios consecutivos [2] como ocurre en la optimización. Permitiendo que S_N denote la suma de cada paso consecutivo X_i, entonces S_N forma una caminata aleatoria.

$$S_N = \sum_{i=1}^{N} X_i = X_1 + \ldots X_N \tag{6}$$

donde X_i es un paso aleatorio tomado de una distribución aleatoria, esta relación puede ser escrita también como una formula recursiva de la siguiente forma.

$$S_N = S_{N-1} + X_N \tag{7}$$

Lo que significa que el siguiente estado de S_N dependerá únicamente del estado actual S_{N-1} y el movimiento o transición X_N del estado actual al siguiente estado, cuando el largo del paso obedece a la distribución de Lévy, tal caminata aleatoria es llamada vuelo de Lévy o caminata de Lévy

Un estudio por Reynolds y Frye [8] muestra que las moscas de fruta exploran usando una serie de vuelos rectos puntuados por un giro súbito de 90°, llevando a un estilo de vuelo de Lévy en su búsqueda

2.3 El algoritmo del murciélago

2.3.1 Inspiración

La gran mayoría de los métodos heurísticos y meta heurísticos se han convertido en métodos poderosos para resolver varios problemas de optimización complejos [9-14]. han sido derivados del comportamiento de sistemas biológicos o de sistemas físicos en la naturaleza, algoritmos como el PSO ha sido desarrollado basado en el comportamiento de enjambre de las aves y peces[13,15], el recocido simulado en el proceso de temple de los metales [16], el más popular , el algoritmo genético, inspirado en la evolución darwiniana [20], otros algoritmos populares basados en la evolución son el algoritmo basado en la estrategia evolutiva [22], la programación genética [21], y el algoritmo de optimización basado en biogeografía [23]

Los distintos algoritmos ofrecen ciertas ventajas y desventajas, por ejemplo, el recocido simulado puede garantizar encontrar la solución óptima si el proceso de enfriamiento es lo suficientemente lento y la simulación corre durante suficiente tiempo [19], el algoritmo del murciélago combina algunas ventajas del Harmony Search y el FFA [19].

Los micro murciélagos, animales en los que se inspira esta heurística usan un tipo de sonar llamado eco-locación para detectar a sus presas, evadir obstáculos y localizar a sus pares en la obscuridad [17,18], estos murciélagos emiten un pulso de sonido muy alto y esperan al rebote del eco de los objetos que los rodean sus pulsos varían en sus propiedades y pueden estar correlacionadas con sus estrategias de cacería, la mayoría de los murciélagos usan señales pequeñas, que duran entre 8 y 10 milisegundos, los cuales se emiten entre 10 y 20 veces cada segundo hasta 200 veces por segundo mientras vuelan cerca de una presa, con frecuencias constantes de 25kHz a 150kHz dependiendo de la especie [19]. Los micro murciélagos usan la diferencia en el tiempo entre la emisión y la detección del eco, las diferencias junto con las variaciones del eco le permiten construir una representación tridimensional de su entorno, así pueden detectar la distancia y la orientación de su presa, y el tipo de presa, este comportamiento puede ser formulado de una forma tal que, se puede asociar la presa con una función a ser optimizada, y la localización del micro murciélago como las variables de diseño del problema.

2.3.2 Modelado

La velocidad del sonido viaja en el aire a $v = 340 m/s$, la longitud de onda λ de la ráfaga de sonidos ultrasónicos con una frecuencia constante f está dada por la ecuación 9:

$$\lambda = \frac{v}{f} \tag{9}$$

Si idealizamos algunas de las características de la eco-localización de los murciélagos, podemos desarrollas varios algoritmos inspirados en ellos, para esto se usan las siguientes ideas:

- Todos los murciélagos usan la eco-localización para medir la distancia, también, conocen la diferencia entre comida, presas y las barreras del fondo.

- Los murciélagos vuelan aleatoriamente con una velocidad v_i a una posición x_i con una frecuencia ajustada f_{min} variando la longitud de onda λ y el volumen A_0 en búsqueda de sus presas, ello puede automáticamente ajustar la longitud de onda de sus pulsos y ajustar la razón de las emisiones de los pulsos $r \in [0, 1]$, dependiendo de la proximidad a su objetivo.
- Aunque el volumen puede variar en varias formas, asumimos que el volumen varía de un gran positivo A_0 a un valor constante mínimo A_{min}

En adición, se usan las siguientes aproximaciones por su simplicidad, en general, la frecuencia f en el rango $[f_{min}, f_{max}]$, corresponde a un rango de longitudes de onda $[\lambda_{min}, \lambda_{max}]$, por ejemplo, un rango de frecuencia de $[20\ kHz, 500\ kHz]$ corresponden a un rango de longitudes de onda entre 0.7 a 17 mm [19].

Para un problema dado, podemos usar cualquier longitud de onda para la sencilla implementación, se ajusta el rango ajustando las longitudes de onda (o frecuencias) y el rango detectable (o la mayor longitud de onda) deben ser escogidas tal forma que sea comparable al tamaño del domino de interés, y después rotar a los valores más pequeños.

Por simplicidad, podemos asumir que, $f \in [0, f_{max}]$, sabemos que las frecuencias más altas tienen longitudes de onda cortas y viajan pequeñas distancias, para los murciélagos, los valores típicos son de unos cuantos metros, la razón de los pulsos se puede simplificar en el rango de $[0, 1]$, donde 0 significa que no hay pulsos y 1 la razón máxima de la emisión de pulsos.

En el pseudo-código de la sección 2.3.4 se definen las reglas de como las posiciones x_i y las velocidades v_i en un espacio de búsqueda d-dimensional son actualizadas, las nuevas soluciones x_i^t y velocidades v_i^t en un paso t están dadas por las ecuaciones (11), (12) y (13) donde $\beta \in [0,1]$ es un vector aleatorio arrojado de una distribución uniforme, aquí x_* es la mejor solución global actual, la cual es obtenida al comparar todas las soluciones entre los n murciélagos, como el producto λ_i y f_i es el incremento de la velocidad, podemos usar λ_i o f_i para ajustar el cambio de velocidad mientras se fija el otro factor λ_i o f_i, dependiendo del problema de interés.

En este estudio se propondrán las frecuencias $f_{min} = 0$ y $f_{max} = 50$ tras la experimentación se encontró que entre menor sea la frecuencia máxima menos tiempo de computo se requiere, probando valores de $f_{max} = [10, 20, 50, 100, 200, 500]$, se encontró que los mejores tiempos de computo se encuentran con valores de 10 a 100, inicialmente a cada murciélago se le asigna una frecuencia extraída uniformemente del rango $[f_{min}, f_{max}]$, para la búsqueda local, una vez que la solución es seleccionada entre las mejores soluciones, una nueva solución para cada murciélago es generada localmente utilizando caminata aleatoria

$$x_{nuevo} = x_{anterior} + \varepsilon A^t \tag{10}$$

Donde $\varepsilon \in [0,1]$ es un número aleatorio, mientras que $A^t = < A_i^t >$ es el promedio del volumen de todos los murciélagos en ese momento.

La actualización de las velocidades y posiciones de los murciélagos tienen cierta similitud al procedimiento del PSO estándar [13] ya que f_i controla el rango del movimiento del enjambre de partículas.

Las razones por las cuales los algoritmos basados en murciélagos son eficientes son:

Sintonización de la frecuencia: El algoritmo del murciélago usa la eco-localización y la sintonización de la frecuencia para resolver problemas, aunque la eco-localización no es usada directamente para imitar la verdadera función, en realidad variaciones de la frecuencia son usados, esta habilidad puede proveer alguna funcionalidad que puede ser similar a la función clave usada en el PSO o Harmony Search, así el algoritmo del murciélago posee las ventajas de otros algoritmos de inteligencia de enjambre ya que el comportamiento del enjambre está definido por un parámetro no estático.

Aproximación automática: El Algoritmo del murciélago tiene una ventaja distintiva sobre otros métodos heurísticos, esta es la capacidad de aproximarse automáticamente dentro de una región si se han encontrado soluciones prometedoras, otros algoritmos no son capaces de reducir su razón de búsqueda para concentrarse en un área específica, la aproximación es acompañada por el cambio automático de los movimientos exploratorios a explotación local intensiva, como resultado, el algoritmo del murciélago tiene una razón de convergencia rápida, al menos en etapas tempranas de las iteraciones, comparado con otros algoritmos

Algunas de las desventajas que posee el algoritmo del murciélago son que toda la población se mueve alrededor de una posición (x_*), esto guía a que todos los murciélagos arrojen soluciones similares, otra desventaja que se puede encontrar es que no hay operador o comportamiento que le permita al algoritmo escapar de óptimos locales, o explorar nuevas regiones aleatorias en el espacio de búsqueda, lo que puede arrojar soluciones subóptimas, la tercera desventaja es que el criterio de aceptación tiene ciertas limitaciones, la formulación básica del algoritmo del murciélago acepta las nuevas soluciones solo si son mejores que la mejor solución global, esto puede limitar el número de movimientos que el murciélago puede efectuar.

2.3.3 Control de parámetros

Varios métodos metaheurísticos usan parámetros fijados usando parámetros pre sintonizados, en contraste el algoritmo del murciélago usa el control de los parámetros, que puede variar los valores de (r_i y A_i) acorde a las ecuaciones 15 y 16, según proceden las iteraciones, esto provee un método que automáticamente intercambia la exploración a la explotación cuando se aproxima a las soluciones óptimas, esto le otorga otra ventaja al algoritmo del murciélago sobre otros métodos metaheurísticos

2.3.4 Pseudo-código

Entradas: función objetivo $f(x), \quad x = (x_1, \dots, x_d)$

Limites Superior e Inferior de la búsqueda [Lower bound,Upper bound]

Salidas: Posición Óptima de los murciélagos

Valor de la función objetivo optimizada

Parámetros: Rango de frecuencia de los pulsos $[f_{min}, f_{max}]$, Volumen A_0, Razón de emisión de pulsos r_i, Parámetro de control de la razón de pulsos γ, Máximo número de iteraciones. Número de murciélagos en la población.

Inicia la población de murciélagos $x_i = (1,2,\dots,n)$ y $\boldsymbol{v}_i$ de forma aleatoria

While (t < Máximo número de iteraciones)

For cada murciélago:

Genera nuevas soluciones ajustando la frecuencia y actualiza las velocidades y localizaciones (soluciones) con las ecuaciones

$$f_i = f_{\min} + \left(f_{\max} - f_{\min}\right)\beta \qquad \text{(Ajuste de frecuencia)} \qquad (11)$$

$$v_i^t = v_i^{t-1} + \left(x_i^t - x_*\right)f_i \qquad \text{(Actualización de la velocidad)} \qquad (12)$$

$$x_i^t = x_i^{t-1} + v_i^t \qquad \text{(Actualización de las soluciones)} \qquad (13)$$

IF (rand > r_i)

Genera una solución local alrededor de la mejor solución, ecuación 10.

End IF

IF (rand < A_i **&&** $f(x_1) < f(x_*)$)

$$f(x_*) = f(x_i) \qquad \text{Acepta las nuevas soluciones} \qquad (14)$$

$$r_i^{t+1} = r_i^0[1 - exp(-\gamma t)] \qquad \text{(Incrementa } r_i\text{)} \qquad (15)$$

$$A_i^{t+1} = aA^t \qquad \text{(Reduce } A\text{)} \qquad (16)$$

End IF

Clasifica a los murciélagos y encuentra al mejor x_*

End FOR

End While

Muestra los resultados y la visualización

2.4 El algoritmo de la ballena (Whale optimization Algorithm)

2.4.1 Inspiración

Las ballenas son criaturas elegantes, son consideradas los mamíferos más grandes del mundo, una ballena adulta puede crecer hasta 30 metros de largo y pesar 180 toneladas, hay 7 especies diferentes de ballena, tales como "Asesina", "minke", "Sei" "jorobada" "derecha" "finback" y "azul" las ballenas son consideradas principalmente predadores, nunca duermen porque tiene que respirar en la superficie del océano, de hecho, solo la mitad de su cerebro duerme. Un aspecto interesante de las ballenas es que son consideradas como animales inteligentes con emociones [26].

De acuerdo con Hof y Van Der Gutch [24] las ballenas tienen células "células fusiformes" en ciertas áreas de sus cerebros similares a las del humano, estas células son responsables del juicio, las emociones y el comportamiento social en humanos, en otras palabras, las células fusiformes nos distinguen de otras criaturas. Las ballenas tienen el doble de estas celular que las encontradas en un humano adulto las cuales son la causa principal de su inteligencia, ha sido probado que las ballenas pueden aprender, juzgar, comunicarse e incluso ser emocionales, así como los humanos, obviamente con un nivel menor de inteligencia, ha sido observado que las ballenas (en especial las ballenas asesinas) son capaces de desarrollar su propio dialecto

Otro aspecto interesante es el comportamiento social de las ballenas, viven solas o en grupos, aunque mayormente en grupos, algunas de sus especies (por ejemplo, las ballenas asesinas)

Pueden vivir en una familia por toda su vida, una de las ballenas "barbadas" más grandes es la ballena jorobada (Megapteranovae angliae), una ballena jorobada adulto es casi del tamaño de un autobús, sus presas favoritas son el "krill" (camarón antártico) y pequeños bancos de peces.

Lo más interesante acerca de la ballena jorobada es su método especial de cacería, este comportamiento predador es llamado "método de alimentación mediante una red de burbujas" [25] , Las ballenas jorobadas prefieren cazar bancos de krill o de peces cerca de la superficie, ha sido observado que ese comportamiento sucede creando burbujas distintivas a lo largo de un circulo o un patrón en forma de "9", antes del 2011, este comportamiento solo había sido investigado basado en la observación desde la superficie. Sin embargo, Goldber en [26], investigo este comportamiento usando sensores de etiqueta, ellos capturaron 300 eventos sensados de las redes de burbujas de 9 ballenas jorobadas individualmente, encontraron también dos maniobras asociadas con las burbujas las cuales nombraron "Espirales hacia arriba" y "bucles dobles". En la primera maniobra, las ballenas bucean aproximadamente 12 metros debajo y empiezan a crear burbujas en forma de espiral alrededor de su presa y nadan hacia arriba de la superficie, la maniobra siguiente incluye 3 estados diferentes:

- El bucle coral
- El golpe con la cola
- El bucle de Captura

Información detallada de estos comportamientos puede ser encontrada en [26].

Es importante mencionar que la alimentación con la red de burbujas es un comportamiento únicamente observado en las ballenas jorobadas, el cual se modela matemáticamente para realizar

optimización, el método de cacería de las ballenas combina dos comportamientos que rodean a la presa reduciendo el área de rodeo hasta que la presa es capturada, mientras que otras ballenas exploran de forma aleatoria con cierta distancia con respecto de las otras ballenas, esto se puede modelar como un algoritmo de optimización en el cual los agentes de búsqueda exploran el espacio de búsqueda d-dimensional de forma global y local así como las ballenas exploran el espacio tri-dimensional y eventualmente capturan a sus presas.

2.4.2 Modelado matemático

2.4.2.1 Rodeo de las presas

Las ballenas jorobadas pueden reconocer la localización de las presas y rodearlas, ya que la posición óptima en el espacio de búsqueda no es conocida con anterioridad, el algoritmo de la ballena asume que el mejor candidato actual para una solución es la presa objetivo o está cerca del optimo [27], después de que el mejor agente es definido, otros agentes trataran de actualizar sus posiciones alrededor del mejor agente de búsqueda, este comportamiento está representado por las siguientes ecuaciones:

$$\vec{D} = \left|\vec{C} \cdot \overrightarrow{X^*}(t) - \vec{X}(t)\right| \tag{17}$$

$$\vec{X}(t+1) = \overrightarrow{X^*}(t) - \vec{A} - \vec{D} \tag{18}$$

Donde *t* es la iteración actual, $\vec{A}$ y $\vec{C}$ son vectores de coeficientes, X^*es el vector de posiciones de la mejor solución obtenida hasta el momento, $|\,|$ es el valor absoluto, $\cdot$ es la multiplicación de elemento por elemento, cabe mencionar que X^* debe ser actualizada en cada iteración si hay una mejor solución.

Los vectores $\vec{A}$ y $\vec{C}$ se calculan como:

$$\vec{A} = 2\vec{a} \cdot \vec{r} - \vec{a} \tag{19}$$

$$\vec{C} = 2 * \vec{r} \tag{20}$$

Donde $\vec{a}$ decrece linealmente entre 2 y 0 a través de las iteraciones (en las fases de exploración y explotación) y $\vec{r}$ es un vector aleatorio entre $[0,1]$

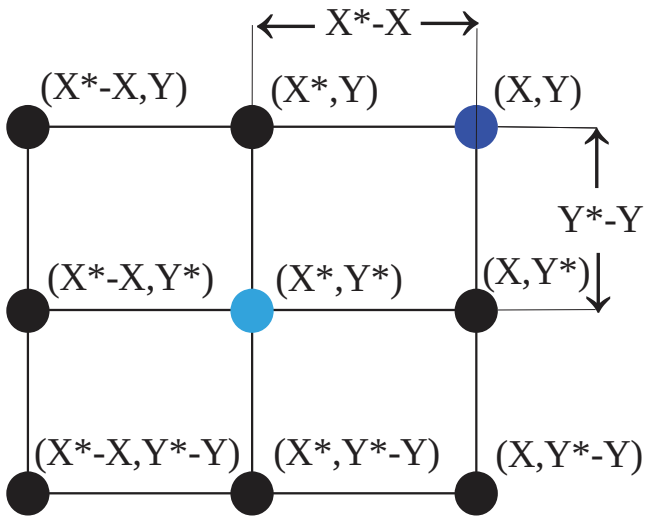


Figura 2.4.1 Vector de posiciones bidimensional y sus próximas nuevas localizaciones (X* es la mejor solución hasta ahora).

En la figura 2.4.1 se ilustra la razón fundamental detrás de la ecuación 18 para un problema de 2 dimensiones, la posición (X, Y) del agente de búsqueda puede ser actualizada de acuerdo con la mejor posición hasta el momento (X*, Y*). Distintos lugares alrededor del mejor agente pueden ser alcanzados ajustando el valor de $\vec{A}$ y $\vec{C}$.

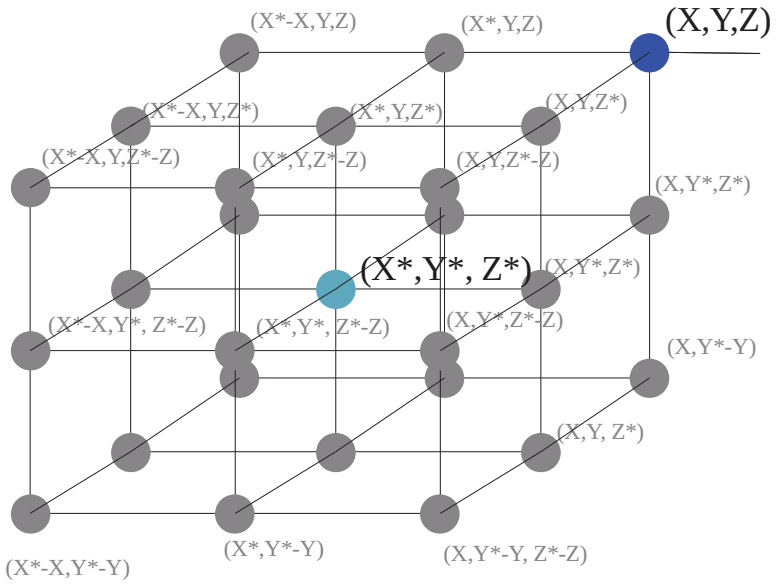


Figura 2.4.2 Vector de posiciones tridimensional y sus próximas nuevas localizaciones (X* es la mejor solución hasta ahora).

La posible actualización de la posición en un espacio tridimensional es mostrada en la figura 2.4.2, debe notarse que con la definición de un vector aleatorio $\vec{r}$ es posible alcanzar cualquier posición en el espacio de búsqueda localizado entre los puntos mostrados en las figuras 2.4.1 y 2.4.2, por lo tanto $\vec{X}(t+1) = \overrightarrow{X^*}(t) - \vec{A} - \vec{D}$ nos permite que cualquier agente de búsqueda actualice su posición en el vecindario de la mejor solución actual y simule rodear la presa.

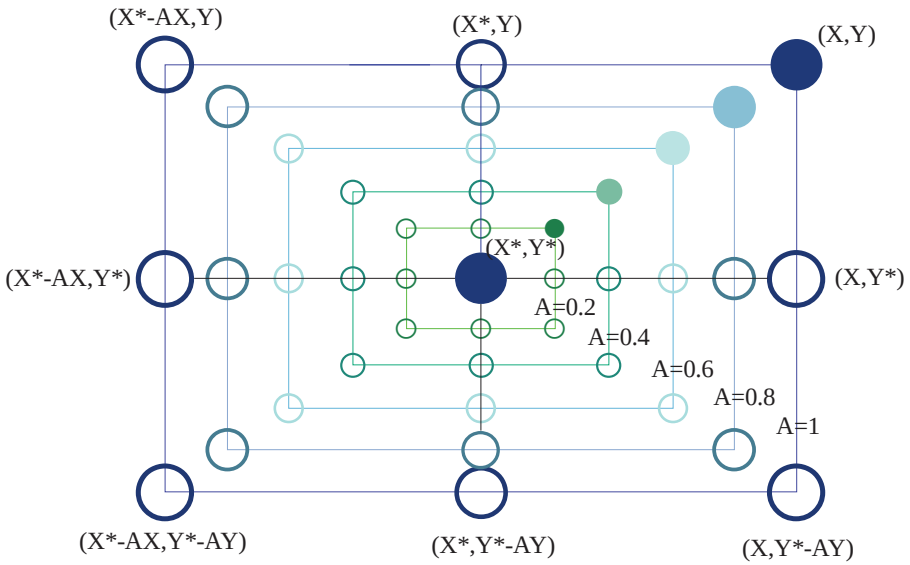


Figura 2.4.3 Mecanismo de búsqueda de la red de burbujas implementado en el WOA (X* es la mejor solución obtenida hasta ahora), mecanismo de encogimiento circular.

El mismo concepto puede ser extendido para buscar en un espacio de n dimensiones, y los agentes de búsqueda se moverán en hipercubos alrededor de la mejor solución obtenida hasta el momento, como se mencionó anteriormente, la ballena jorobada también ataca a sus presas con la estrategia de la "red de burbujas", este método es matemáticamente formulado de la siguiente manera:

2.4.2.2 Método de ataque de la red de burbujas (fase de explotación, búsqueda local)

Para poder modelar matemáticamente este comportamiento de la ballena jorobada, dos enfoques son diseñados como sigue:

- Mecanismo del encogimiento del rodeo: este comportamiento es logrado haciendo decrecer el valor de $\vec{a}$ en la ecuación 19, nótese que las fluctuaciones en el rango de fluctuación de $\vec{A}$ también decrece con $\vec{a}$, en otras palabras $\vec{A}$ es un valor aleatorio en el intervalo de $[-a, a]$ donde a decrece de 2 a 0 a través del curso de las iteraciones, asignando valores aleatorios para $\vec{A}$ entre $[-1,1]$, la nueva posición del agente de búsqueda puede ser definido en cualquier lugar entre la posición original del agente y la posición del mejor agente actual, la figura 2.4.3 muestra las posibles posiciones de (X,Y) hacia (X*,Y*) que pueden ser logradas por $0 \leq A \leq 1$ en un espacio bidimensional

- Actualización de las posiciones en espiral: Como se puede ver en la figura 2.4.4 este enfoque primero calcula la distancia entre la ballena localizada en (X, Y) y la presa localizada entre (X*, Y*), una ecuación espiral es creada entre la posición de la ballena y la presa para emular el movimiento en hélice de la ballena jorobada:

$$\vec{X}(t+1) = \vec{D}' \cdot e^{bl} \cdot \cos(2\pi l) + \vec{X}^*(t) \tag{20}$$

Donde $\overrightarrow{D'} = \left|\overrightarrow{X^*}(t) - \vec{X}(t)\right|$que indica la distancia de a i-esima ballena a la presa (la mejor solución obtenida hasta el momento), b es una constante para definir la forma de la espiral logarítmica, L es un número aleatorio entre $[-1,1]$, y $\cdot$ es una multiplicación elemento por elemento.

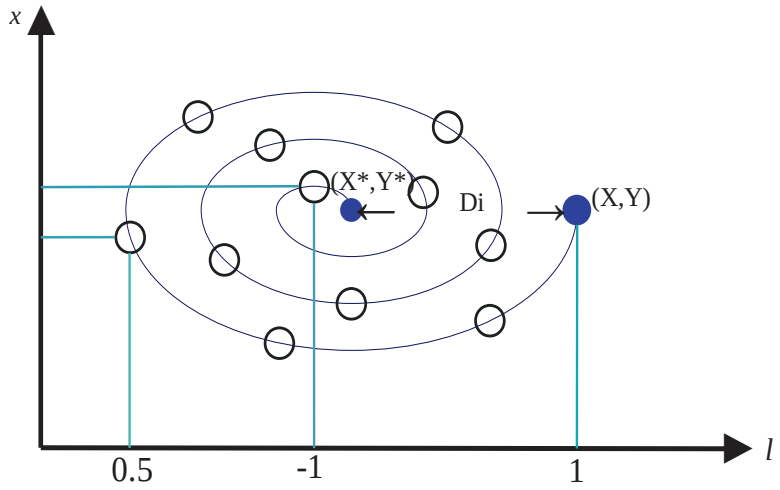


Figura 2.4.4 Mecanismo de búsqueda de la red de burbujas implementado en el WOA (X* es la mejor solución obtenida hasta ahora), actualización de las posiciones en espiral.

Se debe notar que las ballenas jorobadas nadan alrededor de sus presas dentro de un circulo que se encoje y a lo largo de la trayectoria en espiral [27], para modelar este comportamiento simultaneo, se asume que hay una probabilidad del 50% de escoger entre el circulo que se encoje o el modelo en espiral para actualizar la posición de las ballenas durante la optimización, el modelo matemático es el siguiente:

$$\vec{X}^*(t+1) = \begin{cases} \vec{X}^*(t) - \vec{A} \cdot \vec{D} & si,\ p < 0.5 \\ \vec{D}' \cdot e^{bl} \cdot \cos(2\pi l) + \vec{X}^*(t) & si,\ p \geq 0.5 \end{cases} \tag{21}$$

Donde p es un número aleatorio entre $[0,1]$, en adición al método de la red de burbujas, las ballenas buscan sus presas de forma aleatoria, el modelo matemático es el siguiente.

2.4.2.3 Búsqueda de las presas (fase de exploración, búsqueda global)

El mismo enfoque basado en la variación del vector $\vec{A}$ puede ser utilizado para buscar a sus presas(exploración), de hecho, la ballena jorobada busca de forma aleatoria de acuerdo a la posición

de cada una, por lo tanto, usamos $\vec{A}$ con valores aleatorios mayores que 1 o menores que -1 para forzar a los agentes de búsqueda a moverse lejos de la ballena de referencia, en contraste a la fase de explotación, se actualizan las posiciones de los agentes de búsqueda en la fase de exploración de acuerdo a un agente escogido de forma aleatoria en vez del mejor agente encontrado hasta ahora, este mecanismo y $|\vec{A}| > 1$ enfatiza la exploración y permite al algoritmo de la ballena realizar una búsqueda local, el modelo matemático es el siguiente

$$\vec{D} = |\vec{C} \cdot \vec{X}_{rand} - \vec{X}| \tag{22}$$

$$\vec{X}^{*}(t+1) = \vec{X}_{rand} - \vec{A} \cdot \vec{D} \tag{23}$$

Dónde $\vec{X}_{rand}$ es una vector de posiciones aleatorias (una ballena aleatoria) escogida de la población actual, algunas de las posiciones aleatorias alrededor de una solución particular con $|\vec{A}| > 1$, son mostradas en la figura 2.4.5.

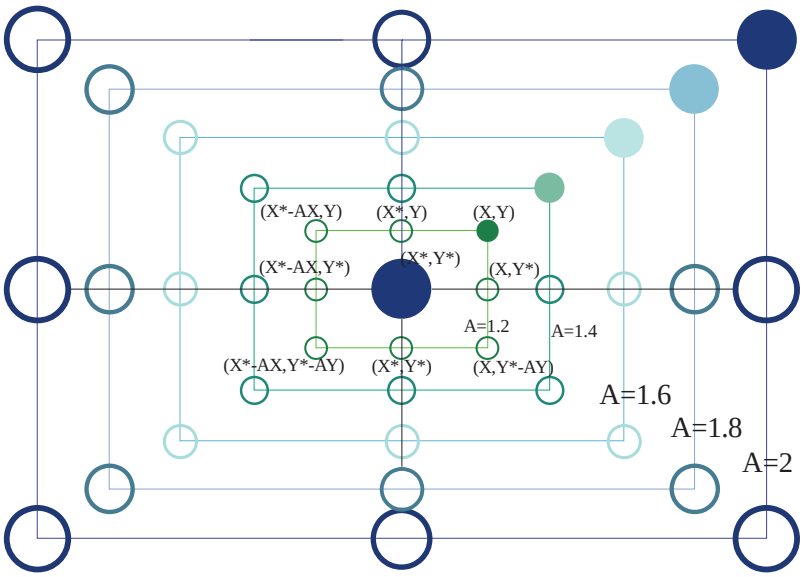


Figura 2.4.5 Mecanismo de exploración implementado en WOA (X* es un agente de búsqueda seleccionado de forma aleatoria).

El algoritmo de la ballena inicia con un set de soluciones aleatorias, en cada iteración , los agentes de búsqueda actualizan sus posiciones con respecto a un agente seleccionado de forma aleatoria, o a la mejor solución obtenida hasta ahora [27], el parámetro a decrece de 2 a 0 para proveer la exploración y la explotación respectivamente, un agente aleatorio es seleccionado cuando $|\vec{A}| > 1$, mientras que la mejor solución es seleccionada cuando $|\vec{A}| < 1$, para actualizar la posición de cada agente de búsqueda dependiendo del valor de *p*, WOA es capaz de cambiar ente un movimiento circular o un movimiento espiral, finalmente el WOA termina cuando se cumplen los criterios de paro.

Desde un punto de vista teórico, WOA puede ser considerado un optimizador global, porque incluye la habilidad en la exploración y la explotación, además, el mecanismo propuesto del hipercubo define un espacio de búsqueda en la vecindad de la mejor solución y permite a otros agentes de búsqueda explotar la mejor solución en ese domino, la variación adaptativa de vector de búsqueda A le permite al WOA transitar suavemente entre la exploración y la explotación, al decrecer A , algunas iteraciones se dedican a la exploración $(|\vec{A}| > 1)$ y el resto a la explotación $(|\vec{A}| < 1)$, notablemente WOA incluye solo dos parámetros internos principales a ser ajustados (A y C).

2.4.3 Pseudo código

Entradas: función objetivo $f(x), \quad x = (x_1, \dots, x_d)$

Limites Superior e Inferior de la búsqueda [Lower bound,Upper bound]

Salidas: Posición Óptima de los agentes de búsqueda (ballenas)

Valor de la función objetivo-optimizada

Parámetros: Máximo número de iteraciones, Cantidad de agentes de búsqueda en la población.

Inicializa la población de ballenas $X_i(i = 1,2, \dots, n)$ de forma aleatoria

Calcula la aptitud de cada agente de búsqueda

X^* = el mejor agente de búsqueda

While (t < Máximo número de iteraciones)

For Cada agente de búsqueda

Actualiza $a = 2 - t \cdot \left(\frac{2}{Numero\ Max.de\ Iteraciones}\right)$

Actualiza A mediante la ecuación 19

Actualiza C mediante la ecuación 20

p se calcula de forma aleatoria con un rango entre [0,1]

L se calcula aleatoriamente en un rango ente [-1,1]

If 1 (p<0.5)

If 2 ($|A<1|$)

Actualiza la posición del agente de búsqueda actual con la ecuación 17 y 18

Else If 2 ($|A \geq 1|$)

Selecciona un agente aleatorio con la ecuación 22 y 23

End if 2

Esle if 1 ($p \geq 0.5$)

Actualice la posición del mejor agente de búsqueda con la ecuación 20

End if 1

End for

Revisa si algún agente de búsqueda sobrepasa el espacio de búsqueda y lo satura de acuerdo a sus límites.

Calcula la aptitud de cada agente de búsqueda

Actualiza X*si hay una solución mejor

t=t+1

End While

Regresa **X***

2.5 El algoritmo FPA (Flower Pollination Algorithm)

2.5.1 Inspiración

Esta estimado que existen más de 250,000 tipos de plantas florecen en la naturaleza, y que aproximadamente el 80% de todas las especies de plantas florecen, sigue siendo un misterio parcial como las plantas llegaron a dominar el terreno durante el periodo cretáceo [28, 34]. Estas plantas han estado evolucionando por más de 125 millones de años y las flores han influenciado tanto la evolución, que no podemos imaginar como la familia de las plantas seria sin flores. El propósito principal de las flores es la reproducción mediante la polinización. La polinización es típicamente asociada con la transferencia de polen, tal transferencia es comúnmente relacionada con los polinizadores tales como insectos, aves, murciélagos y otros animales, De hecho, algunas flores e insectos han evolucionado en una relación especializada, por ejemplo, algunas flores pueden solo atraer y pueden solo depender en una especie específica de insectos para una polinización efectiva [35].

La polinización puede tomar 2 formas principales, abiótica y biótica, aproximadamente el 90% de las flores perteneces a la categoría de polinización biótica, esto es que el polen es transferido por un

polinizador tal como los insectos y los animales, aproximadamente el 10% de la polinización es abiótica, lo que significa que no se requieren polinizadores. El viento y la difusión por el agua ayuda a la polinización de tales plantas, el pasto es un buen ejemplo [35]. Los polinizadores (también llamados vectores de polen) pueden ser muy diversos, se estima que hay al menos 200,000 variedades de polinizadores como insectos, murciélagos y aves.

Las abejas son un buen ejemplo de polinizador, y demuestran algo llamado "consistencia de flores" [36], que significa que estos polinizadores tienden a visitar exclusivamente cierto tipo de flores mientras evitan otras flores. Tal consistencia tiene ventajas evolutivas porque maximizan la transferencia de polen a las mismas plantas, esto también es ventajoso para los polinizadores porque pueden estar seguros que el suministro de néctar está disponible con su memoria limitada y costo mínimo de aprender y explorar [39].

Más que concentrarse en la incertidumbre o el mejor beneficio de nuevas especies de flores, la consistencia requiere un costo mínimo y un mayor aseguramiento de la toma del néctar [29]

La polinización puede ser alcanzada por la autopolinización o la polinización cruzada, la polinización cruzada o allogamia significa que la polinización puede ocurrir del polen de una flor de una planta diferente, mientras que la auto polinización es la fertilización de una flor (como las flores de durazno) del polen de la misma flor o flores diferentes de la misma planta, lo cual sucede frecuentemente cuando no hay polinizadores disponibles.

La polinización biótica cruzada puede ocurrir a una larga distancia, y los polinizadores tales como las abejas, los murciélagos y las aves pueden volar largas distancias, ellos pueden considerarse como polinizadores globales, en adición, las abejar y las aves pueden describir comportamientos como vuelos de Lévy [37], con saltos o vuelos con saltos distanciados obedeciendo a la distribución de Lévy, además, la consistencia de las flores puede ser usada con pasos incrementales usando la similitud o diferencia de 2 flores.

2.5.2 El Algoritmo

Idealizando las características del proceso de polinización, la consistencia de las flores, y el comportamiento del polinizador, se siguen las condiciones del apartado 2.5.2.1:

2.5.2.1 Reglas de polinización

- La polinización biótica y la polinización cruzada es considerada como un proceso de polinización global con los polinizadores ejecutando vuelos de Lévy
- La autopolinización y la polinización abiótica son considerados como polinización local
- La consistencia puede ser considerada como la probabilidad de reproducción y es proporcional a la similitud entre 2 flores involucradas
- La polinización global y la local es controlada por una probabilidad $p \in [0,1].$ debido a la proximidad física y a otros factores tales como el viento, la polinización local puede tener una fracción significativa p en las actividades de polinización

Obviamente, en la realidad, cada planta tiene múltiples flores, y cada flor libera millones e incluso billones de gametos de polen, por simplicidad, asumimos que cada planta tiene una flor y que cada

flor produce solo un gameto, así no hay necesidad de distinguir entre gametos, flores, plantas y soluciones a un problema, esta simplicidad significa que la solución x_i es equivalente a un gameto o una flor, así se puede extender a múltiples gametos de polen o múltiples flores para problemas de optimización multi-objetivo.

Basado en esto se puede diseñar un algoritmo basado en las flores, llamado "Algoritmo de polinización" (FPA, por sus siglas en inglés), hay 2 pasos claves en este algoritmo y son la polinización global y local.

En el paso de la polinización global, el polen es llevado por los polinizadores (como los insectos) y el polen puede viajar por una gran distancia porque los insectos pueden volar en rangos muy grandes, esto asegura que la polinización y la reproducción de los más aptos, los cuales se pueden representar como " g^* ", La primera regla más la consistencia de las flores puede ser representada matemáticamente como [39]:

$$x_i^{t+1} = x_i^t + L(x_i^t - g^*) \tag{24}$$

Donde x_i^t es el polen *i* o el vector solución x_i en la iteración *t* y g^* es la mejor solución actual encontrada entre todas las soluciones de la generación en curso. El parámetro *L* es la fuerza de la polinización la cual es esencialmente el paso, Como los insectos pueden moverse a través de la distancia con diferente longitud de paso, se puede utilizar el vuelo de Lévy para simular esta característica eficientemente [37, 38], designamos *L>0* de una distribución de Lévy, la cual define el siguiente paso del polen actual con respecto de la mejor solución encontrada en las iteraciones.

$$L \approx \frac{\lambda \Gamma(\lambda) \sin(\pi\lambda/2)}{\pi} \frac{1}{s^{1+\lambda}} (s > s_0 > 0) \tag{25}$$

2.5.3 Pseudo Código

Entradas: función objetivo $f(x), \quad x = (x_1, \ldots, x_d)$

Limites Superior e Inferior de la búsqueda [Lower bound,Upper bound]

Salidas: Posición Óptima de los agentes de búsqueda (polinizadores)

Valor de la función objetivo optimizada

Parámetros: Máximo número de iteraciones, probabilidad de conmutación $p \in [0,1].$, Número de individuos en la población "n".

Inicializa la población de *n* flores/gametos con soluciones aleatorias x_i^{t0}

Evalúa la población inicial y asigna a g^* la mejor solución

While (t < Iteraciones Máximas)

For i=1 hasta n (Tamaño de la población)

If rand < p

Calcula un vector L que responda a la distribución de Lévy, ecuación 25

Poliniza globalmente con $x_i^{t+1} = x_i^t + L(x_i^t - g^*)$

Else

Genera ε de una distribución uniforme entre [0,1]

Escoge aleatoriamente j y k entre todas las soluciones

Ejecuta una polinización local con $x_i^{t+1} = x_i^t + \varepsilon(x_j^t - x_k^t)$

End If

If $f(x_i^{t+1}) < f(g^*)$ Evalúa las nuevas soluciones

$f(g^*) = f(x_i^{t+1})$ Si la nueva solución es mejor, actualiza la población

End IF

End For

Encuentra la mejor solución en g^*

End While

Aquí $\Gamma(\lambda)$ es la función gamma estándar, y esta distribución es válida para grandes pasos $s>0$, en las simulaciones, se utilizo $\lambda = 1.5$

La polinización local, puede ser representada por:

$$x_i^{t+1} = x_i^t + \varepsilon(x_j^t - x_k^t) \tag{26}$$

Donde x_j^t y x_k^t son polen de distintas flores de la misma especie de planta, esto esencialmente replica la consistencia de las flores en un vecindario limitado, matemáticamente, si x_j^t y x_k^t vienen se la misma especie o son seleccionadas de la misma población, esto se convierte en una caminata aleatoria si dibujamos ε de una distribución uniforme entre [0,1]

La mayoría de las actividades de las flores pueden ocurrir en ambas escalas (local y global), en la práctica, las flores adyacentes o las flores en un vecindario no tan alejado son más propensas a ser

polinizadas que las flores por el polen local que por el más alejado, para esto, se usa una conmutación de probabilidad (regla 4 del apartado 2.5.2.1) o la probabilidad aproximada *p* para conmutar entre la exploración global y la exploración local , para iniciar se puede utilizar una *p=0.5* como valor inicial entones, hacer un estudio paramétrico para encontrar el valor más apropiado, las pruebas experimentales encuentran que un valor de *p=0.8* que da preferencia a la búsqueda local, funciona bien para la mayoría de las aplicaciones [39].

2.6 Resultados Numéricos

Para asegurar la valides de los algoritmos deben de ser extensivamente comparados con otros, existen varias funciones de prueba o funciones de referencia para la validación de los algoritmos, varias funciones de prueba son recopiladas en [40-42].

Las funciones objetivo son clasificadas como funciones multimodales y unimodales, algunas funciones son de dimensión fija, mientras otras son más genéricas. Aquí, las funciones (27) y (28) son funciones unimodales, las funciones (29) y (30) son funciones multimodales de dimensiones genéricas, la función (31) es una función multimodal de dimensión fija.

Todos los algoritmos son probados con una población de 50 individuo, para el algoritmo del murciélago se escogió un rango de frecuencias de [0,10], un volumen inicial A de 0.5 y una razón de pulsos r de 0.5, el parámetro de control de la razón de pulsos se escogió de 0.1, en el algoritmo FPA se asigna una probabilidad de conmutación de 0.8, en el algoritmo WOA el único parámetro necesario de definir es la cantidad de iteraciones máximas permitidas al algoritmo, las cuales varían ente cada función de prueba en orden de aparición como [1000, 1500, 1500, 1000, 100].

TABLA 2.6-1. Funciones "benchmark" de prueba para las distintas heurísticas.

Función	Rango	Nombre	No.
$f(x)=\sum_{i=1}^{n} x_i^2$	$-100 \le x_i \le 100$	“Sphere”	(27)
$f(x)=\max_i\{\lvert x_i\rvert, 1\le i\le n\}$	$-100 \le x_i \le 100$	“Schwefel 2.21”	(28)
$f(x)=\sum_{i=1}^{n}\left[x_i^2-10\cos(2\pi x_i)+10\right]$	$-5.12 \le x_i \le 5.12$	“Rastringin”	(29)
$f(x)=\frac{1}{4000}\sum_{i=1}^{n} x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	$-600 \le x_i \le 600$	“Grienwangk”	(30)
$f(x)=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2+4x_2^2+4x_2^4$	$-5 \le x_i \le 5$	“Three hump camel”	(31)

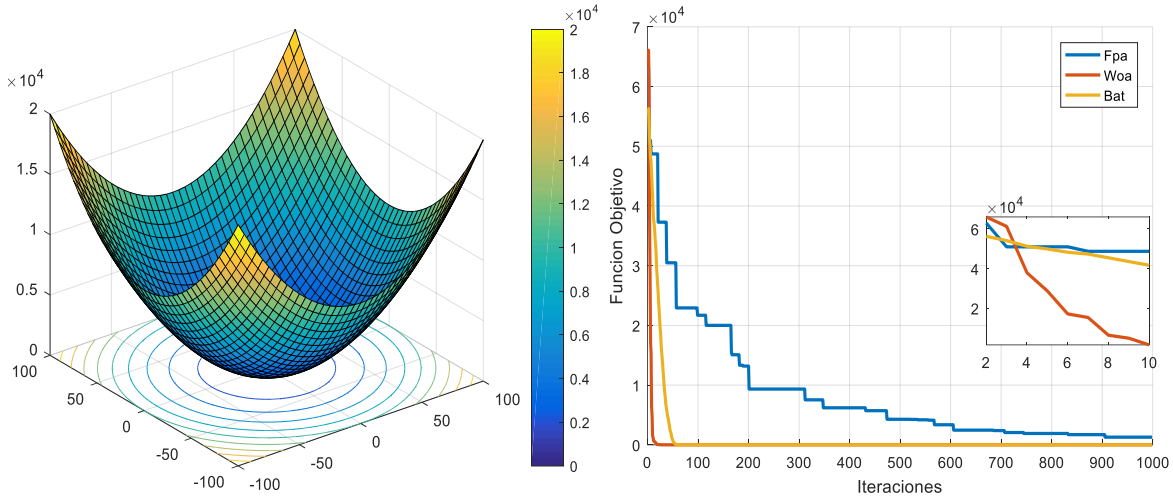


Figura 2.6.1 Representación bidimensional de la función Sphere(a), Curva de convergencia (b).

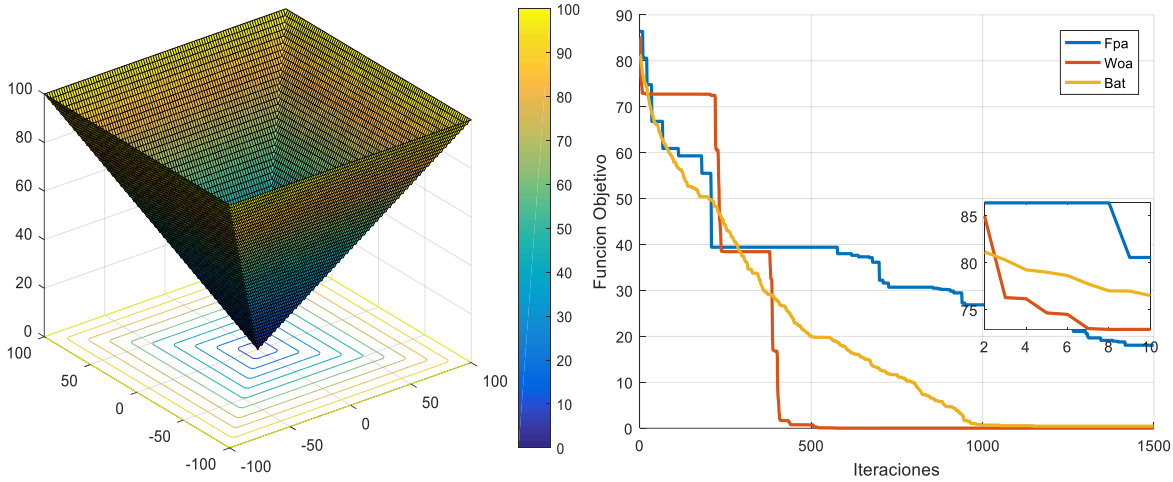


Figura 2.6.2 Representación bidimensional de la función Schwefel (a), Curva de convergencia (b).

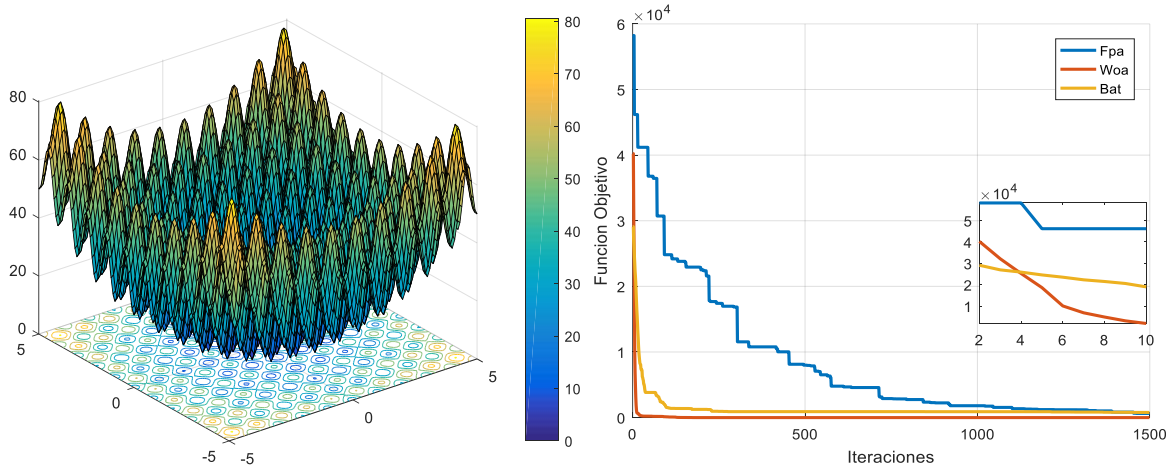


Figura 2.6.3 Representación bidimensional de la función Rastringin (a), Curva de convergencia (b).

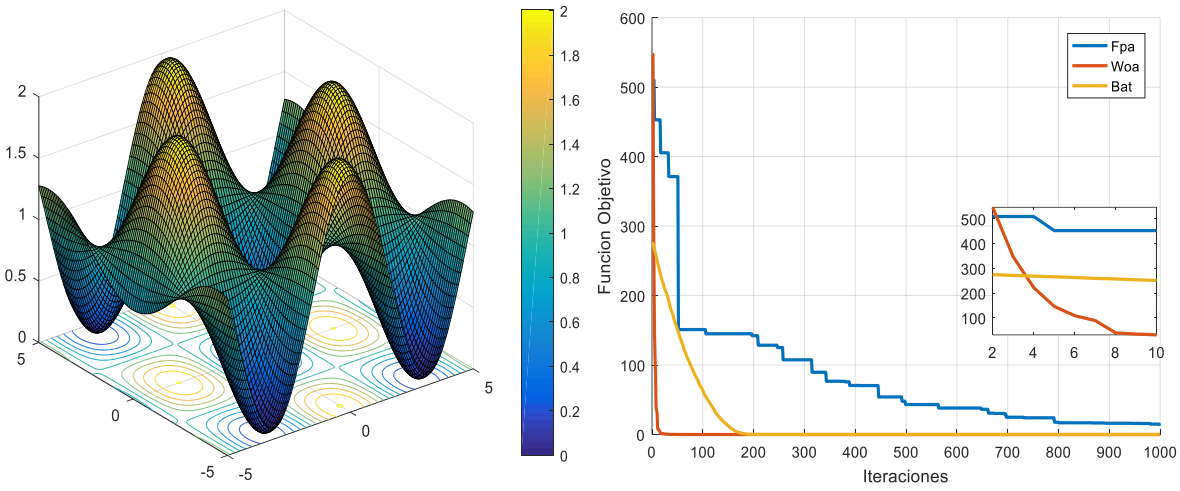


Figura 2.6.4 Representación bidimensional de la función Grienwangk (a), Curva de convergencia (b).

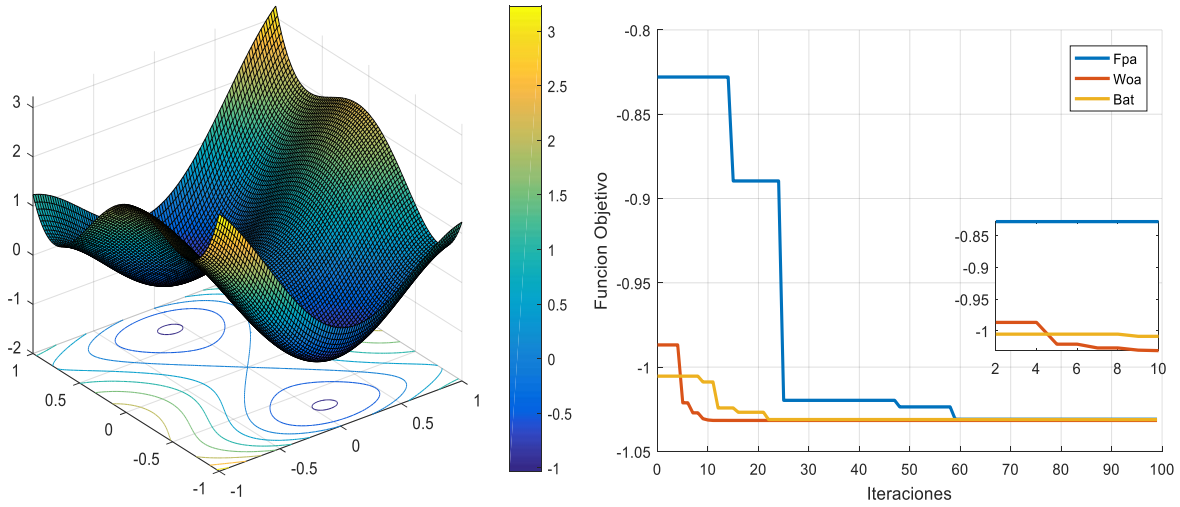


Figura 2.6.5 Representación bidimensional de la función Three hump camel (a), Curva de convergencia (b).

2.6.1 Análisis de Resultados

Entre las curvas de convergencia de los distintos algoritmos para los cinco problemas de prueba se puede notar una tendencia en el comportamiento de estos, donde el algoritmo WOA muestra la mayor velocidad de convergencia de los tres, seguido de algortimo BAT y del FPA, a su vez, en las figuras 2.6.3 y 2.6.4 se aprecia una mayor velocidad de convergencia hacia el optimo con respecto de las otras funciones objetivo, siendo estas funiones multimodales se puede apreciar como los algoritmos son capaces de evadir optimos locales, dirigiéndose en dirección al global, las giguras 2.6.1 y 2.6.2 corresponen a funciones unimodales, se puede apreciar, en la función correspondiente a la figura 2.6.1 que el decenso es muy aelerado para WOA y para BAT, mientras que FPA desciende con lentitud debido a que da preferencia a busquedas locales, en la figura 2.6.2 se observa que es mas complicado

mejorar la solución debido a la naturaleza de la función Schwefel, por ultimo, en la figura 2.6.5 se aprecia como los tres algorimtos convergen en pocas iteraciones al optimo, desto debido a que el problema es fijado para dos dimesiones, cuando los otros, en la prueba experimental se fijaron en 30 dimensiones pudiendo ser este número variable

2.7 Conclusiones

En este capítulo se trataron tres diferentes algoritmos de optimización, dos de los cuales son basados en el comportamiento de enjambre inteligentes, mientras que el tercero es basado en el proceso natural de polinización de las flores, tras un análisis de su composición, se hicieron pruebas numéricas comparativas de la actuación de estos tres métodos heurísticos frente a 5 problemas de optimización de distintas características, todos del tipo "Benchmark" o funciones de prueba, estos resultados arrojas que la convergencia de un algoritmo a una solución global está definido por la forma en la que formula su búsqueda tanto global como local, y los parámetros propios del algoritmo, como la probabilidad de transición entre la búsqueda global y la explotación intensiva en la búsqueda local, el algoritmo FPA tuvo una peor actuación que los otros dos algoritmos al dar preferencia a la explotación de mínimos locales, el mejor algoritmo, que mostro la convergencia más rápida al mínimo global fue el algoritmo WOA, seguido en su actuación por el algoritmo BAT, la actuación del algoritmo WOA se debe a sus mecanismos de acción en la explotación y la exploración y a su habilidad de escapar de los óptimos locales extraída de la estrategia de adaptación utilizada para actualizar el vector *A.* La naturaleza de la inspiración de los tres algoritmos puede proveer características diferentes, una similitud entre estos es que son algoritmos basados en población, y entre el algoritmo del murciélago y el algoritmo FPA la relación entre la convergencia y la selección de parámetros internos del algoritmo, la forma en la que evolucionan apunta a que cada uno de estos podría tener una actuación mejor que los otros dependiendo de la naturaleza del problema de optimización, y aunque el algoritmo WOA tuvo la mejor actuación con las funciones de prueba en este estudio, no significa que sería el que tiene la mejor actuación para todos los problemas.

2.8 Referencias:

[1] Slawomir Koziel and Xin-She Yang (Eds.), Computational Optimization, Methods, and Algorithms, Studies in Computational Intelligence, Springer-Verlag Berlin Heidelberg, Vol. 356, 2011.

[2] Xin-She Yang, Nature-Inspired, Metaheuristic Algorithms, Luniver Press, 2010.

[3] Scott Kirkpatrick, D. Gelatt Jr., and Mario P Vecchi, “Optimization by simulated annealing". Science, Vol. 220(4598), pp. 671-680, 1983

[4] Iztok Fister Jr., Xin-She Yang, Iztok Fister, Janez Brest, and Dusan Fister, “A Brief Review of Nature-Inspired Algorithms for Optimization", Elektrotrhniski, Vestnik, Vol. 80(3), pp. 116-122, 2013.

[5] Aboul Ella Hassanien, Eid Emary, “Swarm Intelligence, Principles, Advances, and Aplications”, ISBN 978-1-4987-4107-1.

[6] Christian Walck, “Handbook on statistical distributions for experimentalists". Internal Report SUFPFY/9601 Stockholm, 11 December 1996, 2007.

[7] David B. Thomas, Wayne Luk, Philip H.W, Leong, and John D. Villasenor, \Gaussian random number generators". ACM Computing Surveys, Vol. 39(4), Article 11,October 2007.

[8] A. M. Reynolds and M. A. “Frye, Free-ight odor tracking in Drosophila is consistent with an optimal intermittent scale-free search", PLoS One, 2, e354 (2007).

[9] Deep, K., Bansal, J. C.: Mean particle swarm optimisation for function optimisation. Int. J. Comput. Intel. Studies, 1, 72-92 (2009).

[10] Geem, Z.W., Kim, J. H., Loganathan, G. V.: A new heuristic optimization algorithm: Harmony search. Simulation, 76, 60-68 (2001).

[11] Goldberg, D.: Genetic Algorithms in Search, Optimiztion and Machine Learning. Addison-Wesley, Reading, MA, (1989).

[12] Holland, J. H.: Adapation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, (1975)

[13] Kennedy, J. and Eberhart, R.: Particle swarm optimization, Proc. IEEE Int. Conf.Neural Networks. Perth, Australia, 1942-1945 (1995)

[14] Mitchell, M.: An Introduction to Genetic Algorithms. MIT Press, (1998).

[15] Kennedy, J. and Eberhart, R., Swarm Intelligence. Academic Press, (2001).

[16] Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P.: Optimization by simulated annealing. Science, 220, 671-680 (1983).

[17] Richardson, P.: BATs. Natural History Museum, London, (2008).

[18] Richardson, P.: The secrete life of BATs. http://www.nhm.ac.uk

[19] X.-S. Yang, “A New Metaheuristic BAT-Inspired Algorithm", in: Nature Inspired Cooperative Strategies for Optimization (NISCO 2010) (Eds. J. R. Gonzalez et al.), Studies in Computational Intelligence, Springer Berlin, 284, pp. 65-74, 2010

[20] HollandJH. Genetic algorithms. SciAm1992; 267:66–72.

[21] J.R. Koza,“Genetic programming,” 1992.

[22] Rechenberg I. “Evolution strategies”. Springer Berlin Heidelberg;1978. p.83–114.

[23] Simon D. Biogeography-based optimization. IEEE Transactions on Evolutive Computing 2008; 12:702–13.

[24] Hof P R, Van Der Gucht E.” Structure of the cerebral cortex of the humpback whale”, Megaptera novaeangliae (Cetacea, Mysticeti, Balaenopteridae).AnatRec 2007;290:1–31

[25] WatkinsWA, Schevill WE. “Aerial observation of feeding behavior in four baleen whales: Eubalaenaglacialis, Balaenopteraborealis, Megapteranovae angliae, and Balaenopteraphysalus”. J Mammal1979:155–63.

[26] Goldbogen J.A., Friedlaender A.S, Calambokidis J, Mckenna M.F., Simon M, Nowacek D.P., “Integrative approaches to the study of baleen whale diving behavior, feeding performance, and foraging ecology”. BioScience2013;63:90–100.

[27] Seyedali Mirjalili, Andrew Lewis, “The Whale Optimization Algorithm”, Advances in Engineering Software 95 (2016) 51-67, Online http://dx.doi.org/10.1016/j.advengsoft.2016.01.008

[28] Walker, M., “How flowers conquered the world”, BBC Earth News, 10 July 2009.http://news.bbc.co.uk/earth/hi/earth news/newsid 8143000/8143095.stm

[29] Waser, N.M., “Flower constancy: definition, cause and measurement”. The American Naturalist, 127(5), 596-603 (1986)

[30] Yang, X. S., “Nature-Inspired Metaheuristic Algorithms”, Luniver Press, (2008).

[31] Yang, X. S., “Firefly algorithm, stochastic test functions and design optimization”, Int. J. Bio-Inspired Computation, 2(2), 78-84 (2010).

[32] Yang, X. S., Engineering Optimization: “An Introduction with Metaheuristic Applications”, Wiley (2010).

[33] Yang, X. S., (2010c). “A new metaheuristic BAT-inspired algorithm”, in: Nature-Inspired Cooperative Strategies for Optimization (NICSO 2010) (Eds. Gonzalez J. R. et al.), Springer, SCI 284, pp. 65–74.

[34] “Oily Fossils provide clues to the evolution of flowers”, Science Daily, 5 April 2001. http://www.sciencedaily.com/releases/2001/04/010403071438.htm

[35] Glover, B. J., “Understanding Flowers and Flowering: An Integrated Approach, Oxford University Press”, (2007).

[36] Chittka, L., Thomson, J. D., and Waser, N. M., “Flower constancy, insect psychology, and plant evolution”, Naturwissenschaften, 86, 361-177 (1999).

[37] Pavlyukevich I., “Lévy flights, non-local search and simulated annealing”, J. Computational Physics, 226, 1830-1844 (2007).

[38] Reynolds A. M. and Frye M. A., “Free-flight odor tracking in Drosophila is consistent with an optimal intermittent scale-free search”, PLoS One, 2, e354 (2007)

[39] Xin-She Yang, “Flower pollination algorithm for global optimization", in: Unconventional Computation and Natural Computation 2012, Lecture Notes in Computer Science, Vol. 7445, pp. 240-249, 2012

[40] M.M. Ali C. Khompatraporn, Z.B. Zabinsky “A Numerical evaluation of several stochastic algorithms on selected contious global optimization test problems”, Journal of global Optimization, Vol.31, No. 4, pp. 635-672 (2005)

[41] B.M. Averick, R.G. Carter, J.J.More “The MINIPACK 2 Test Problem Collection”, Mathematics and Computer Science Division, Agronne National Laboratory, Technical Memorandum No.150 (1991)

[42] J. Liang, P. Suganthan, and K. Deb, "Novel composition test functions for numerical global optimization, " in Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE, 2005, pp. 68-75.

Capítulo 3

Control óptimo

3.1 Introducción

En este capítulo se habla de las etapas del control jerárquico de los dos arquetipos de robot móvil, así como del uso de las heurísticas en el cálculo de las ganancias de los controladores en el nivel alto (estructura mecánica) y nivel bajo (sistemas de actuación).

3.2 Control de nivel alto, estructura mecánica

3.2.1 Robot Ackerman

Los vehículos con ruedas, o los robots móviles con ruedas son una clase muy efectiva de vehículo, así como el arquetipo para la mayoría de los robots móviles siendo uno de estos el tipo automóvil, este tipo de robot móvil actúa con el direccionamiento frontal, y es usualmente conducido por las ruedas traseras y el direccionamiento es logrado por un actuador que gira las ruedas delanteras [1].

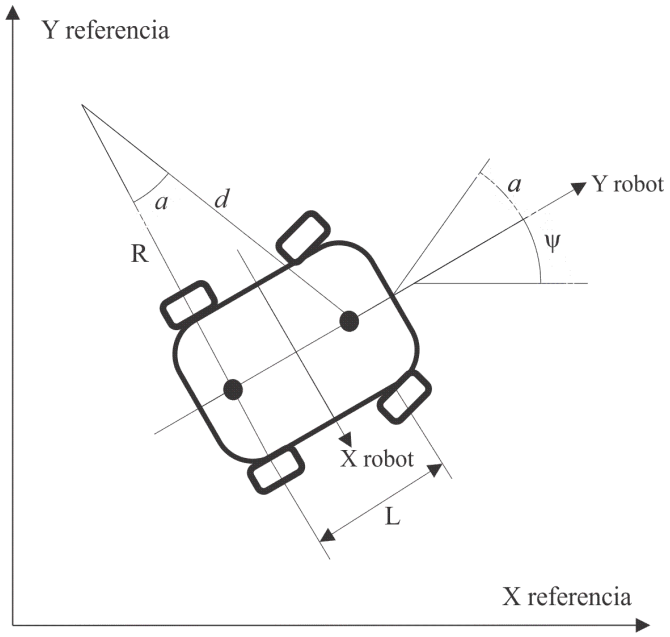


Figura 3.2.1 Diagrama esquemático del robot Ackerman con dirección delantera

El ángulo que las ruedas delanteras toman con respecto del eje longitudinal del robot y_{robot}, es definido como *a* medido en la dirección contraria a las manecillas del reloj, el ángulo que el eje longitudinal y_{robot} toma con respecto del plano de referencia $y_{p\ referencia}$ es definido por ψ, también es medido en dirección contraria a las manecillas del reloj, El centro instantáneo de rotación en el cual el robot gira el punto de intersección de 2 líneas pasando a través de los ejes de las ruedas.

De la geometría del vehículo se puede obtener que:

$$\frac{L}{R} = \tan a \tag{1}$$

Lo cual puede ser resuelto para dar el radio instantáneo de curvatura para la trayectoria del punto medio del eje trasero del robot.

$$R = \frac{L}{\tan a} \tag{2}$$

De la geometría también tenemos que:

$$v_{rueda\,trasera} = r\frac{d}{dt}(\psi) = R\dot{\psi} \tag{3}$$

Despejando se puede obtener que:

$$\dot{\psi} = \frac{v_{rueda\,trasera}}{R} \tag{4}$$

Lo que puede ser reescrito como:

$$\dot{\psi} = \frac{v_{rueda\,trasera}}{L/\tan a} = \frac{v_{rueda\,trasera}}{L}\tan a \tag{5}$$

Si uno mantiene el ángulo *a* constante, la trayectoria resultante es un círculo cuyo radio es dictado por el largo del robot y el ángulo de la dirección usado en la ecuación 2. El conjunto completo de ecuaciones cinemáticas para el robot es:

$$\dot{x} = v_{rueda\,trasera}\sin\psi \tag{6}$$

$$\dot{y} = v_{rueda\,trasera}\cos\psi \tag{7}$$

$$\dot{\psi} = \frac{v_{rueda\,trasera}}{L}\tan a \tag{8}$$

La complejidad de estas ecuaciones yace en que son no lineales, las ecuaciones proveen una relación cinemática correcta acerca entre las variables para el movimiento y la rotación en el plano *xy* pero no incluyen la complejidad de la suspensión o el motor [2], para la simulación cinemática se toma el largo entre ejes L el cual corresponde a 0.177 metros, en el caso del robot diferencial, la distancia entre neumáticos es de 0.1 m.

Por simplicidad de la notación tomamos:

$$v_{rueda\,trasera} = V \tag{9}$$

3.2.1.1 Control de la dirección

La dirección deseada puede ser dada directamente como un comando:

$$\psi_{des} = direccion\ especificada \tag{10}$$

Esta dirección puede surgir de una trayectoria predeterminada, o puede ser designada por un sensor que detecte algo de interés, la dirección deseada puede ser computada en términos de la localización actual y las coordenadas del destino, esto si no hay presencia de obstáculos, la dirección de la posición actual del robot al destino puede ser expresada como:

$$\psi_{des} = \tan^{-1}\left(\frac{y_{des} - y}{x_{des} - x}\right) \tag{11}$$

Inicialmente puede ser asumido que la dirección puede ser comandada directamente, así se puede escoger una expresión de la forma

$$a = K(\psi_{des} - \psi) \tag{12}$$

El Ángulo de la dirección es proporcional al error en la dirección.

3.2.1.2 Control de la velocidad

Una forma de seleccionar la velocidad deseada del robot seria en términos de la distancia al destino [3] de la forma:

$$V_{des} = \sqrt{(x_{des} - x)^2 + (y_{des} - y)^2} \tag{13}$$

La cual es tomada como consigna utilizando un esquema de control del tipo PI, entonces, la señal de control para el motor de la tracción es:

$$u = K_{p\,vel} V_{des} + K_{i\,vel} \int_0^t V_{des}\, dt \tag{14}$$

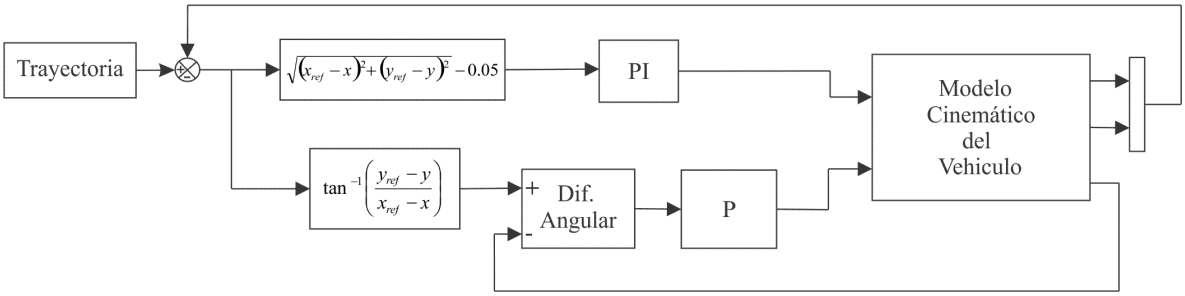


Figura 3.2.2 Diagrama a bloques del control del robot.

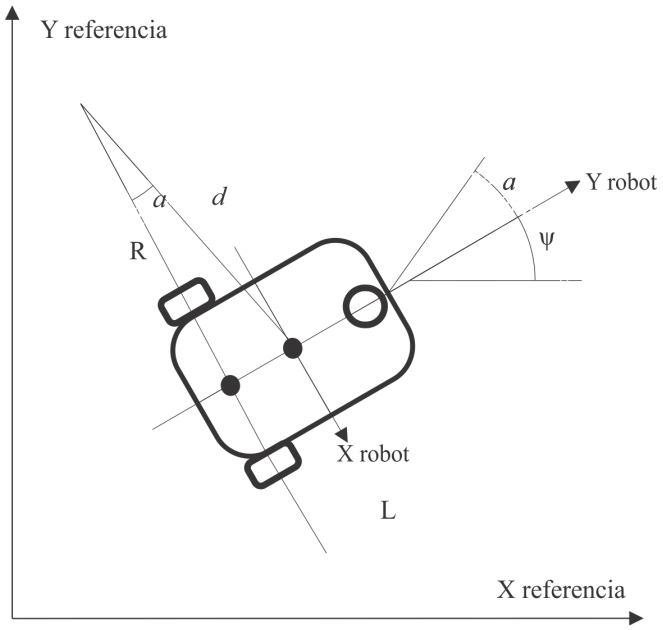


Figura 3.2.3 Diagrama Esquemático del robot diferencial

3.2.2 Robot Diferencial

El segundo tipo más común de dirección utilizada en robots móviles es el direccionamiento diferencial [1] ilustrado en la figura 3.2.3, donde las ruedas laterales del robot son controladas independientemente, las ecuaciones de movimiento de direccionamiento diferencial son ahora mostradas.

Permitamos que R represente el radio instantáneo de curvatura de la trayectoria del robot, el grosor del vehículo (el espacio entre las ruedas) se le designa W, de las consideraciones geométricas tenemos que:

$$v_{izquierda} = \dot{\psi}\left(R - W/2\right) \tag{15}$$

$$v_{derecha} = \dot{\psi}\left(R + {}^{W}/_{2}\right) \tag{15.1}$$

Substrayendo de ambas ecuaciones lleva a

$$v_{derecha} - v_{izquierda} = \dot{\psi}W \tag{16}$$

Mediante un despeje se puede obtener la razón de desplazamiento angular del robot

$$\dot{\psi} = \frac{v_{derecha} - v_{izquierda}}{W} \tag{17}$$

Resolviendo para el radio instantáneo de curvatura tenemos que:

$$R = \frac{v_{izquierda}}{\dot{\psi}} + \frac{W}{2} \tag{18}$$

$$R = \frac{v_{izquierda}}{\frac{v_{derecha} - v_{izquierda}}{W}} + \frac{W}{2} \tag{19}$$

O finalmente

$$R = \frac{v_{derecha} + v_{izquierda}}{v_{derecha} - v_{izquierda}} \frac{W}{2} \tag{20}$$

Esto resulta e la expresión para la velocidad a través del eje longitudinal del robot:

$$v_y = \dot{\psi}R = \frac{v_{derecha} - v_{izquierda}}{W} \frac{W}{2} \frac{v_{derecha} + v_{izquierda}}{v_{derecha} - v_{izquierda}} = \frac{v_{derecha} + v_{izquierda}}{2} \tag{21}$$

En resumen, las ecuaciones de movimiento en las coordenadas del robot son:

$$v_x = 0$$

$$v_y = \frac{v_{derecha} + v_{izquierda}}{2} \tag{22}$$

$$\dot{\psi} = \frac{v_{derecha} - v_{izquierda}}{W} \tag{23}$$

En coordenadas terrestres, estas se convierten en:

$$\dot{x} = \frac{v_{derecha} + v_{izquierda}}{2} \sin\psi \tag{24}$$

$$\dot{y} = \frac{v_{derecha} + v_{izquierda}}{2} \cos\psi \tag{25}$$

$$\dot{\psi} = \frac{v_{derecha} - v_{izquierda}}{W} \tag{26}$$

Para este sistema de ecuaciones también se puede utilizar el método de Euler para obtener un modelo en tiempo discreto para el sistema de ecuaciones no lineales

$$x\big((k+1)T\big) = x(kT) + T\frac{v_{derecha}(kT) + v_{izquierda}(kT)}{2}\sin\psi\,(kT) \tag{27}$$

$$y\big((k+1)T\big) = y(kT) + T\frac{v_{derecha}x(kT) + v_{izquierda}x(kT)}{2}\cos\psi\,x(kT) \tag{28}$$

$$\psi\big((k+1)T\big) = \psi(kT) + T\frac{v_{derecha}(kT) - v_{izquierda}(kT)}{W} \tag{29}$$

3.2.2.1 Control del robot diferencial

Para controlar la trayectoria del robot diferencial se debe controlar la velocidad angular de cada uno de los motores, esto en términos de la distancia al punto del destino como:

$$v_{des} = \sqrt{(x_{des} - x)^2 + (y_{des} - y)^2} \tag{30}$$

El robot con control diferencial de los neumáticos gira usando velocidades diferentes en los neumáticos, un método para girar en torno a una dirección particular seria girar el robot en su centro

Para obtener el ángulo deseado de giro y el error angular, se usa las ecuaciones (11) y (12), las cual se repiten aquí por conveniencia:

$$\psi_{des} = \tan^{-1}\left(\frac{y_{des} - y}{x_{des} - x}\right) \tag{31}$$

$$\omega_{ref} = K\left(\psi_{des} - \psi\right) \tag{32}$$

$$v_{ref} = K_{p\,vel}V_{des} + K_{i\,vel}\int_0^t V_{des}\,dt \tag{33}$$

Buscando combinar el giro del robot con el movimiento longitudinal, la estrategia debe considerar el giro deseado y la velocidad deseada, una vez que esto es obtenido las ecuaciones que definen la velocidad deseada y la razón de giro se puede combinar para llegar a la solución:

$$\omega_d = v_{ref} + \omega_{ref} \tag{34}$$

$$\omega_i = v_{ref} - \omega_{ref} \tag{35}$$

es sencillo ver que esta combinación satisface el giro deseado y la velocidad longitudinal deseada.

3.2.3 Extensión de la ley de control no lineal

El control cinemático presentado a continuación fue reportado en [4], en este, se definen las señales de error como:

$$\begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix} = \begin{pmatrix} K_2\cos\varphi & K_2\sin\varphi & 0 \\ -K_2\sin\varphi & K_2\cos\varphi & a \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_{ref} - x \\ y_{ref} - y \\ \psi_{ref} - \psi \end{pmatrix} \tag{36}$$

$$v_f = v_{des}\cos e_3 + K_1 e_1 \tag{37}$$

$$\omega_f = \omega_{ref} + v_{des} K_2 e_2 + K_3 \sin e_3 \tag{38}$$

Donde ψ_{ref} corresponde a la ecuación (31), ω_{ref} corresponde a la ecuación (32), y v_{des} corresponde a la ecuación (33). De acuerdo con [5], los controles que logran que $(x, y, \varphi) \to (x_{ref}, y_{ref}, \varphi_{ref})$ corresponden a las ecuaciones 37 y 38.

Para la implementación experimental de (37,38) tras utilizar la transformación en (36) se encuentra que ω_d y ω_i son las referencias de velocidad angular deseados que deberán seguir las flechas de los motores CD

$$\omega_d = \frac{1}{r} v_f + \frac{l}{r} \omega_f \tag{39}$$

$$\omega_i = \frac{1}{r} v_f - \frac{l}{r} \omega_f \tag{40}$$

3.2.4 Sintonización de los controladores cinemáticos (de alto nivel) para el Robot

Debido a que las ecuaciones que definen el comportamiento del sistema son no lineales, es complicado realizar el cálculo de las constantes de control del sistema, por esto, se decide utilizar los algoritmos bio-inspirados (B.I.) revisados en el capítulo 2 (BAT, WOA, FPA) para realizar el computo de las ganancias de control del sistema, el diagrama representado en la figura 3.2.4 es aplicable a ambas topologías de robot. La sintonización de dichos controladores se hace fuera de línea mediante MATLAB, esto debido al tiempo de computo necesario para llegar a la solución óptima, así como a la naturaleza no determinista de los algoritmos de búsqueda y las limitaciones de hardware que puede llegar a tener la unidad central de procesamiento del robot.

En la simulación, las variables de diseño (posición del individuo en el algoritmo) es probada como las ganancias de control cinemático del robot, el error entre el resultado del desplazamiento del robot y la trayectoria deseada son la función objetivo a optimizar, entre más pequeño sea este error, el robot se aproxima más a la trayectoria de referencia.

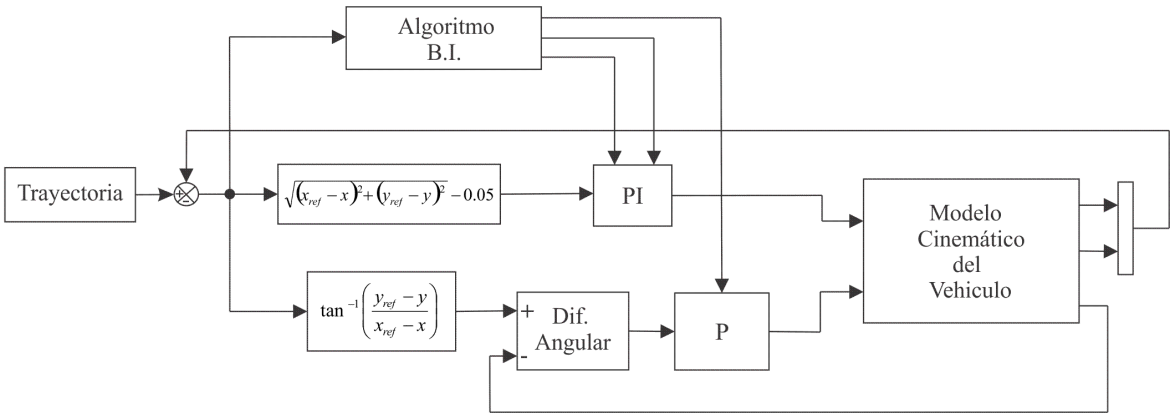


Figura 3.2.4 Sintonización de las leyes de control para el robot móvil.

Sabemos que los controladores son estables mientras $K_{p\ direccion} > 0, K_{p\ velocidad} > 0$, $K_{i\ velocidad} > 0$, se procede a diseñar una función objetivo de la siguiente forma:

$$\min f(x) = \sum_{i=1}^{n} \left| x_i^{des} - x_i^{rob} \right| + \sum_{i=0}^{n} \left| y_i^{des} - y_i^{rob} \right| \tag{41}$$

Sujeto a:

$$K_{p\ direccion} > 0 \quad K_{p\ velocidad} > 0 \quad K_{i\ velocidad} > 0 \tag{42}$$

Para una trayectoria previamente diseñada con el arquetipo Ackerman se tiene que:

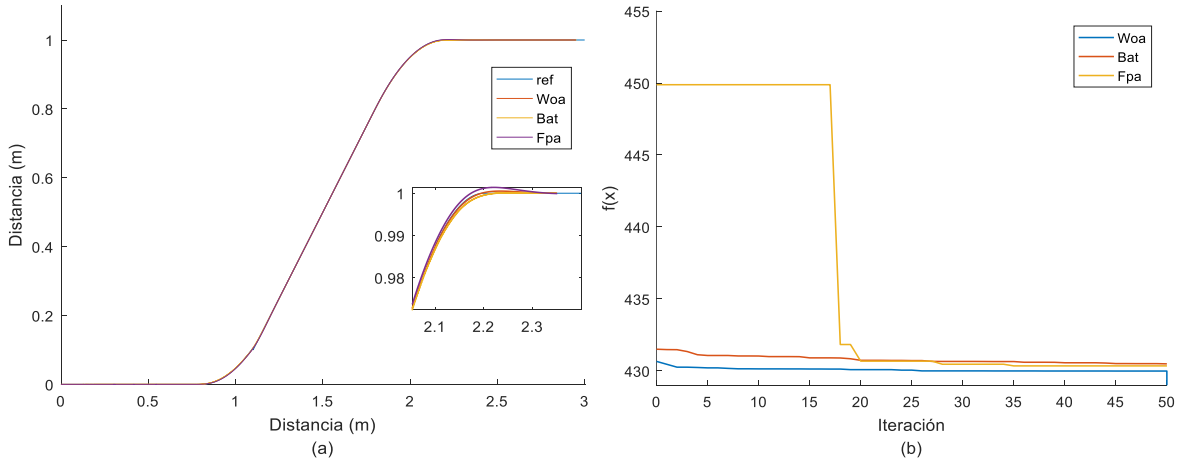


Figura 3.2.5 Respuesta del control cinemático (a), curva de convergencia del proceso de optimización (b).

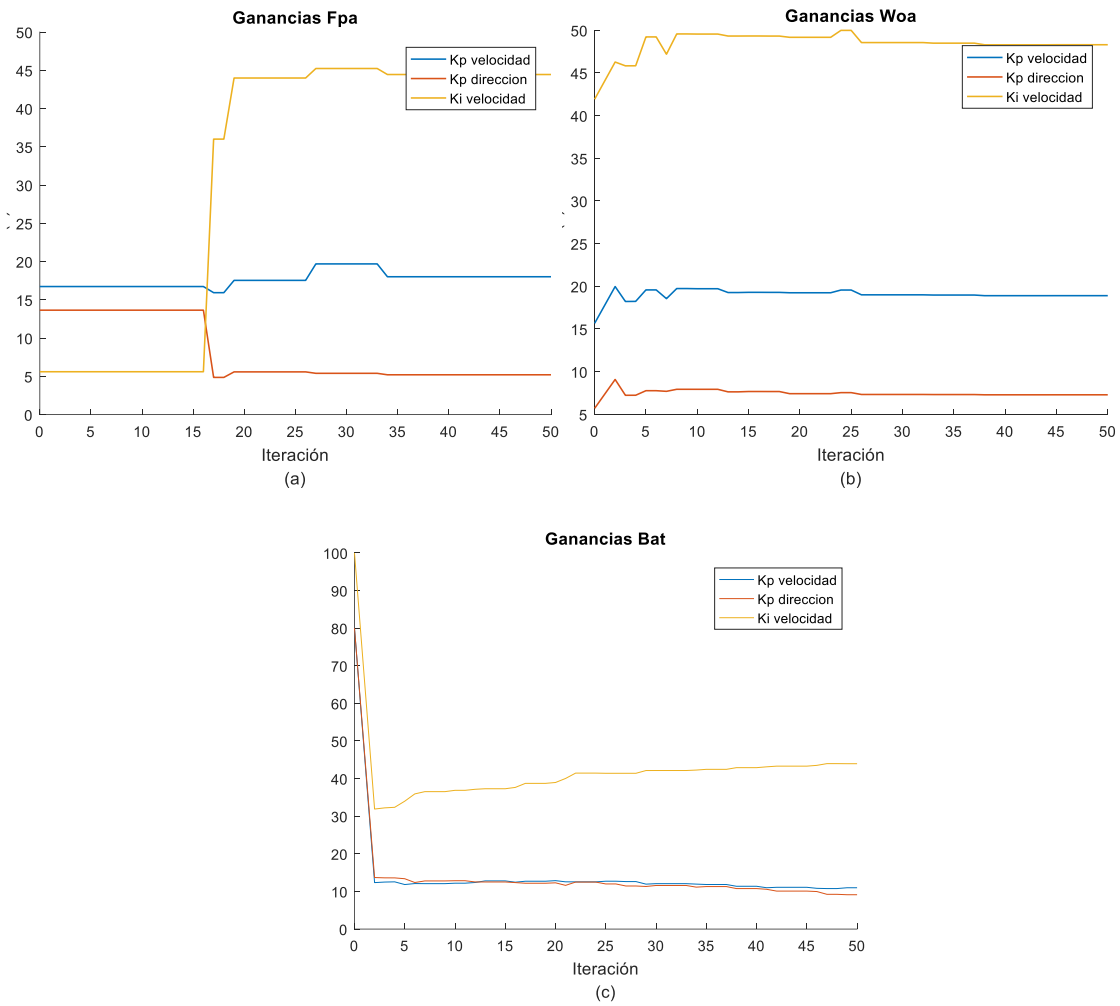


Figura 3.2.6 Evolución de las ganancias para el control cinemático del robot Ackerman con el algoritmo FPA (a), Algoritmo WOA (b), Algoritmo BAT (c).

TABLA 3.2-1. Ganancias óptimas y valor de la función objetivo calculado para el control cinemático del robot ackerman.

	$K_p\ vel$	$K_p\ dir$	$K_i\ vel$	$Fun.obj.$
FPA	17.117	4.876	44.774	430.256
WOA	18.464	7.729	46.321	429.986
BAT	10.127	8.761	42.368	430.312

Los resultados del control cinemático del robot diferencial son los siguientes:

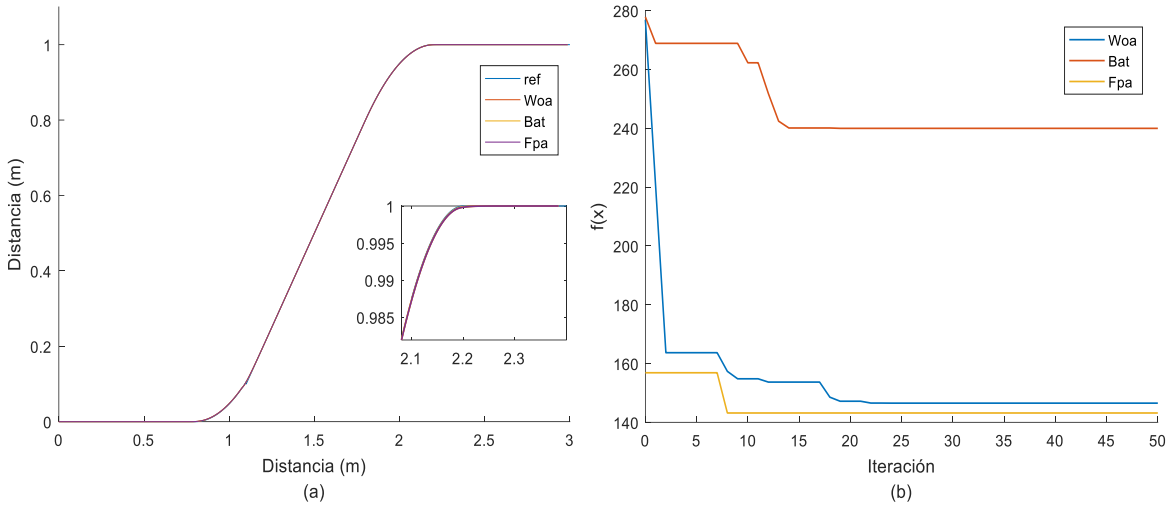


Figura 3.2.7 Respuesta del control cinemático (a), curva de convergencia de la optimización (b).

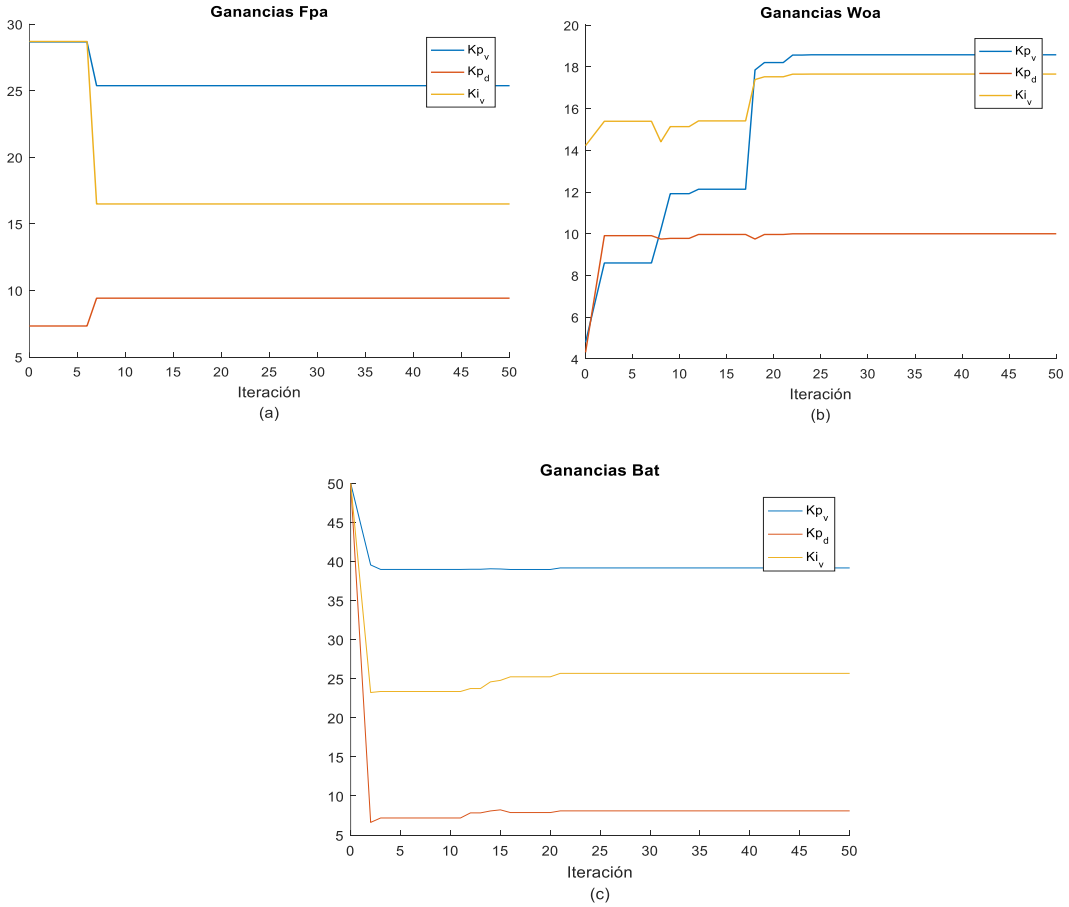


Figura 3.2.8 evolución de las ganancias para el control cinemático del robot diferencial con el algoritmo FPA (a), Algoritmo WOA (b), Algoritmo BAT (c).

TABLA 3.2-2. Ganancias óptimas y valor de la función objetivo calculado para el control cinemático del robot diferencial

	$K_p\ vel$	$K_p\ dir$	$K_i\ vel$	$Fun.obj.$
FPA	25.091	9.465	16.841	141.671
WOA	18.269	9.814	17.381	144.267
BAT	39.234	7.466	25.018	239.874

En cuanto a la extensión del control no lineal para la locomoción Ackerman con una trayectoria circular se tiene que:

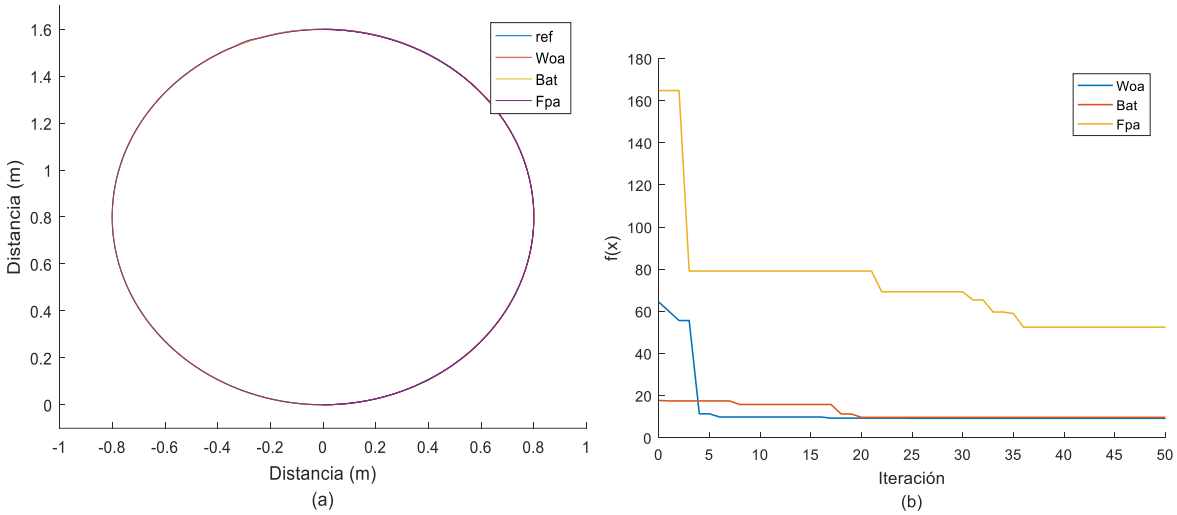
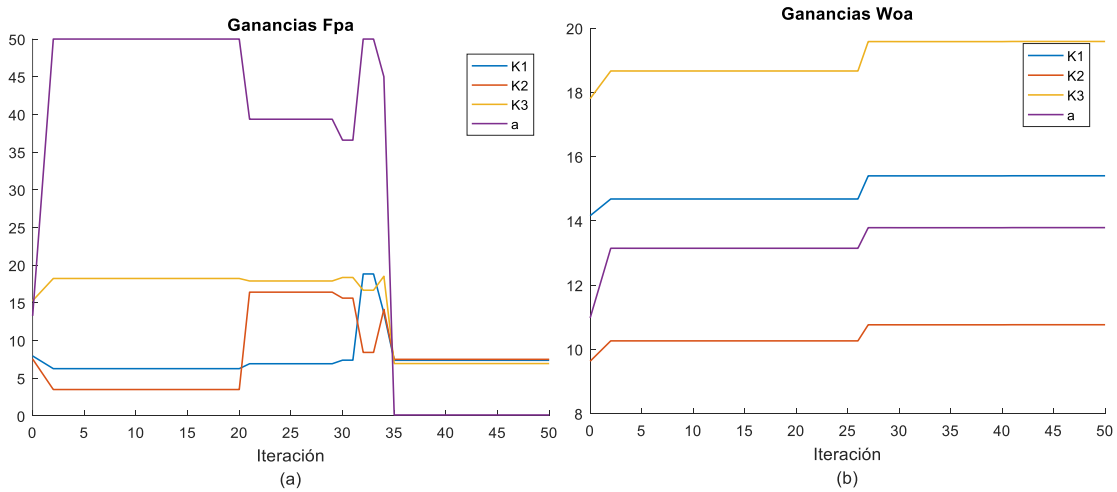


Figura 3.2.9 Respuesta del control cinemático (a), curva de convergencia de la optimización (b).



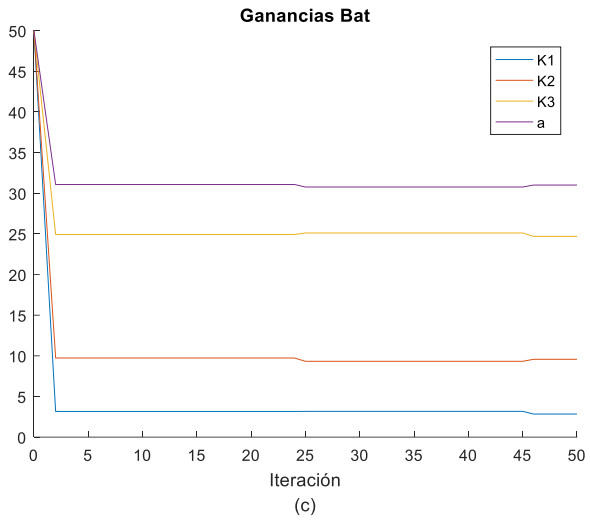


Figura 3.2.10 Evolución de las ganancias para la extensión del control cinemático del robot Ackerman con el algoritmo FPA (a), Algoritmo WOA (b), Algoritmo BAT (c).

TABLA 3.2-3. Ganancias óptimas y valor de la función objetivo calculado la extensión del control cinemático del robot ackerman

	K_1	K_2	K_3	a	$Fun.obj.$
FPA	7.768	7.716	7.641	0.149	52.341
WOA	15.269	10.278	19.127	13.667	9.825
BAT	3.143	9.385	24.493	31.424	9.841

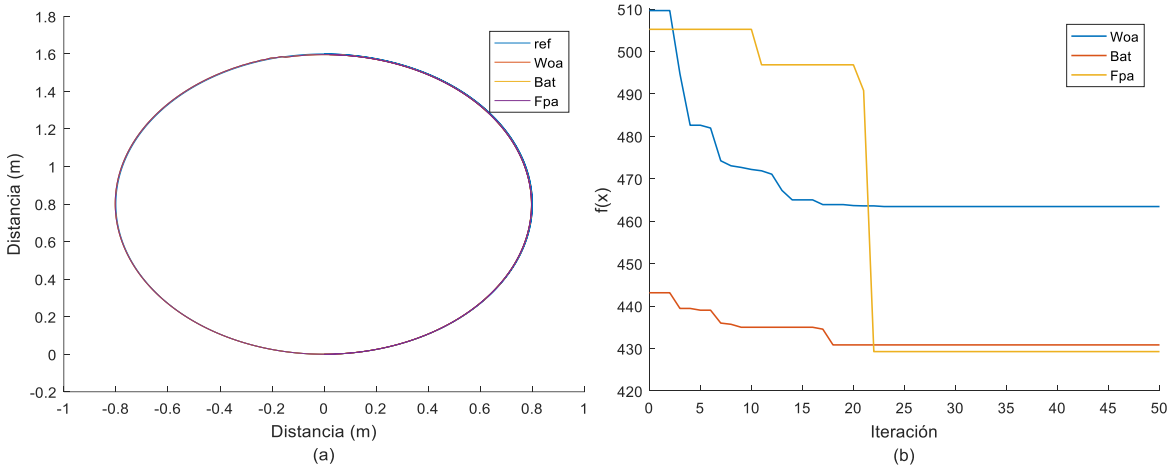


Figura 3.2.11 Respuesta del control cinemático (a), curva de convergencia de la optimización (b).

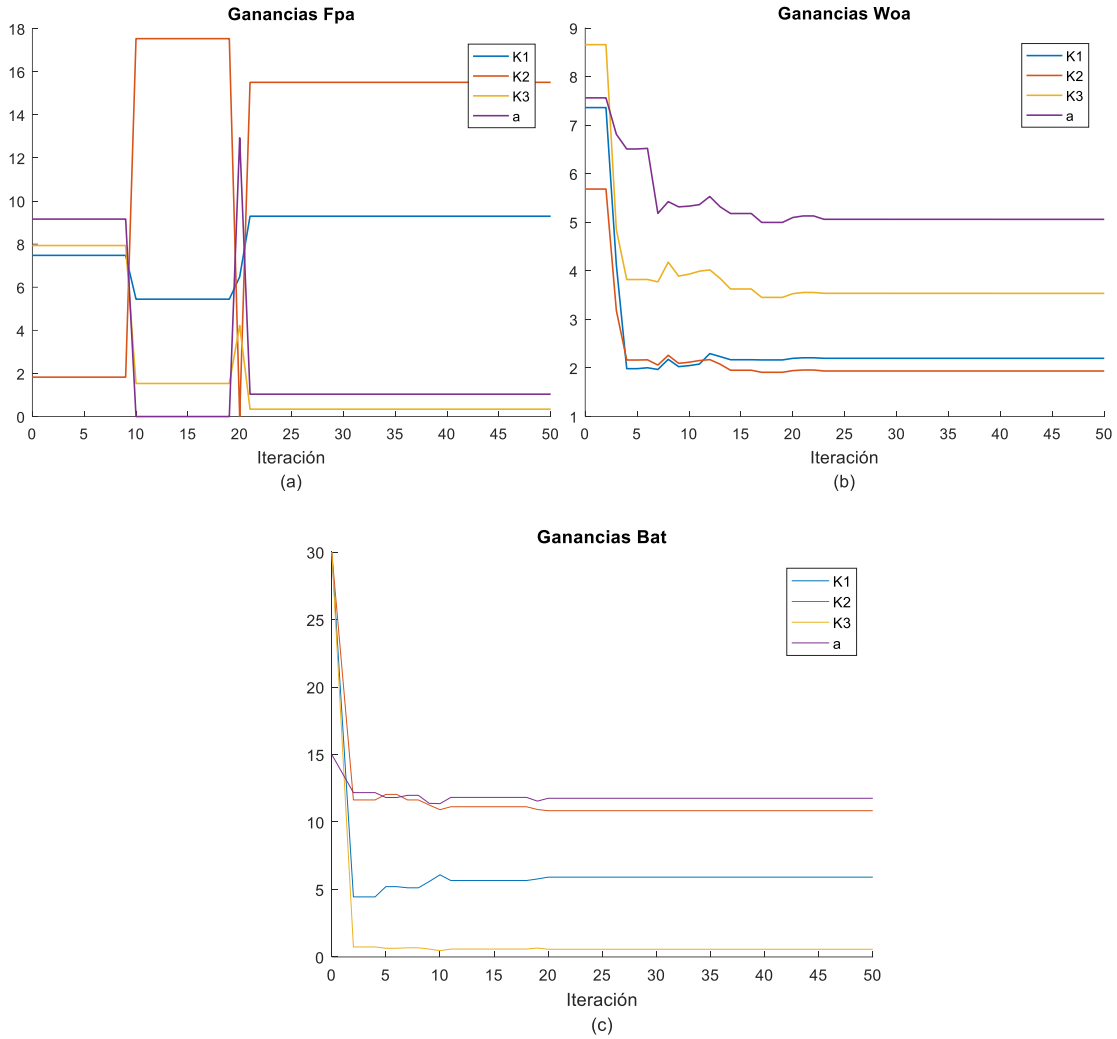


Figura 3.2.12 Evolución de las ganancias para la extensión del control cinemático del robot diferencial con el algoritmo FPA (a), Algoritmo WOA (b), Algoritmo BAT (c).

Tabla 3.2-4. Ganancias óptimas y valor de la función objetivo calculado para la extensión del control cinemático del robot diferencial.

	K_1	K_2	K_3	a	$Fun.obj.$
FPA	9.354	15.438	0.178	1.141	429.766
WOA	2.257	1.857	3.564	5.019	464.147
BAT	5.281	10.773	0.785	11.651	430.185

3.3 Optimización con restricciones

Para la sintonización mediante técnicas de optimización para los sistemas de actuación se tienen que considerar ciertas restricciones, tales como el sobre impulso máximo, el tiempo de asentamiento, el error en estado estacionario entre otras [6].

Para poder convertir un problema de optimización sin restricciones a uno con restricciones se usan funciones de penalización, dos tipos básicos de función de penalización existen, funciones de penalización exterior, las cuales penalizan las soluciones infactibles, y las funciones de penalización interior las cuales penalizan soluciones factibles [7], las cuales requieren que la restricción sea activa mientras la solución óptima yace entre la frontera de la factibilidad y la infactibilidad, las funciones de penalización exterior son tratadas a continuación.

3.3.1 Penalización de las restricciones

Un método simple para penalizar soluciones infactibles es aplicar una penalización constante a las soluciones que violan la factibilidad [8]. La función objetivo-penalizada puede ser una función objetivo no penalizada más una penalización (para un problema de minimización). Una variación es construir esta simple función de penalización como una función del número de restricciones violadas donde estas son múltiples restricciones. La función de penalización para un problema de m restricciones puede ser la siguiente (para un problema de minimización):

$$f_p(x) = f(x) + \sum_{i=1}^{m} C_i \delta_i \qquad donde \begin{cases} \delta_i = 1 & Si\ la\ restriccion\ i\ es\ violada \\ \delta_i = 0 & Si\ la\ restriccion\ i\ es\ satisfecha \end{cases}$$

$f_p(x)$ es la función objetivo penalizada, $f(x)$ es la funcion objetivo no penalizada, y C_i es una constante impuesta para la violación de la restricción i. Esta función de penalización está basada únicamente en el número de restricciones violadas.

Puede ser complicado encontrar una función de penalización que sea efectiva y eficiente, una sintonización pobre de una función de penalización para un problema dado puede negar la calidad de una solución dada, en siedlecki y sklansky [9] se menciona que mucha de la dificultad se debe a que las soluciones óptimas frecuentemente yacerán en la frontera de la región factible. Imponer penalizaciones severas puede hacer complicado encontrar el esquema en el cual la población de individuos se dirija al optimo, de forma contraria, si la función de penalización no es lo suficientemente severa, con una región de exploración lo suficientemente amplia mucho del tiempo de búsqueda será usado para explorar regiones lejos de la solución factible, así la búsqueda tendera a permanecer fuera de la solución factible

3.4 Control de los actuadores

Las velocidades y las posiciones angulares requeridas por los robots móviles son suministradas por motores de corriente directa de imán permanente (PMDC) que se controlan de manera independiente. El control de estos motores permite que a las variables de control converger en los perfiles de posición y velocidad deseados, impuestos por el control cinemático. En esta subsección se tratará un control PI en cascada para la velocidad y un PI en cascada para la posición.

El modelo matemático asociado a un motor PMDC en términos de la velocidad está dado por [10]:

$$\frac{di_a}{dt} = \frac{-R_a \cdot i_a - K_e \cdot n + U}{L_a}$$
$$\frac{dn}{dt} = \frac{-\frac{K_e}{2\pi} \cdot i_a - T_{carga} - B \cdot n}{2\pi J} \quad (43)$$
$$\frac{d\theta}{dt} = 2\pi n$$

Donde U es el voltaje aplicado a las terminales del motor, i_a es la corriente de armadura, $R_a$es la resistencia de armadura, $L_a$es la inductancia de armadura, J es el coeficiente de inercia, B es el coeficiente de friccion viscosa, K_e es la constante de fuerza electromotriz, y debido a que el par eléctrico en estado estable es igual al par mecánico, la constante de par del motor se expresa como $\frac{K_e}{2\pi}$ y n es la velocidad en revoluciones sobre segundo.

3.4.1 Control en cascada

En la práctica existen grandes cantidades de procesos con lazos simples de control retroalimentado, entre los que se produce interacción, para eliminarla, es necesario medirla y controlarla por medio de un sistema de control denominado cascada, los sistemas en cascada son aquellos sistemas en los que la salida de cada ley de control sirve como referencia para la siguiente, creando varios bucles de control internos y uno externo.

En un sistema de control en cascada, la dinámica del lazo secundario debe ser siempre más rápida que la del primario, en caso contrario, es posible que no funcione correctamente,

Las características de un controlador en cascada son:

- Si la perturbación ocurre en el lazo de control interior, el controlador secundario inicia la acción correctiva antes de que se traslade al lazo de control exterior
- Si la perturbación ocurre en el lazo exterior, el comportamiento de la cascada hace que se modifique el punto de referencia del lazo interior, en este caso el conjunto se comporta prácticamente como si fuera un solo control retroalimentado

Para nuestro problema de control será necesaria la implementación de un controlador en cascada, basado en lo dicho por [11] un controlador en cascada usa uno o más controladores retroalimentados, solamente uno de ellos, denominado esclavo o secundario, tiene salida al proceso, en nuestra aplicación, el controlador de velocidad o de posición, también conocido como maestro, o primario, se utiliza para fijar la referencia del secundario, la variable a controlar es la velocidad o posición mientras que la medida del torque es una variable intermedia, basado en el diagrama para el control de la velocidad y el torque presentado por [10], partiendo hacia atrás, también podemos observar que los controladores en cascada a su vez son comandados por los controladores cinemáticos, en [12, 13] se le denomina a estos controladores como controladores jerárquicos.

Diagrama a bloques del controlador en cascada:

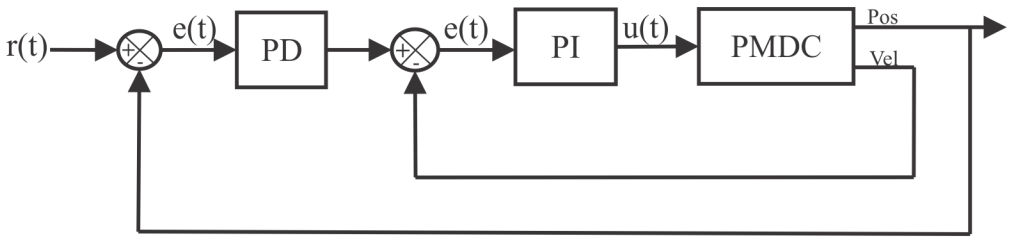


Figura 3.4.1 Control en cascada para el control de posición-velocidad del motor

Para un motor lego cuyas constantes son: $R_a = 4.5$, $L_a = 0.00517$, $J = 0.000217$, $B = 0$, $K_e = 3.2086$, con una carga constante de $T = 0$. En la figura 3.4.2 se le aplica una entrada de escalón unitario *U=10* en lazo abierto para observar la respuesta dinámica del sistema.

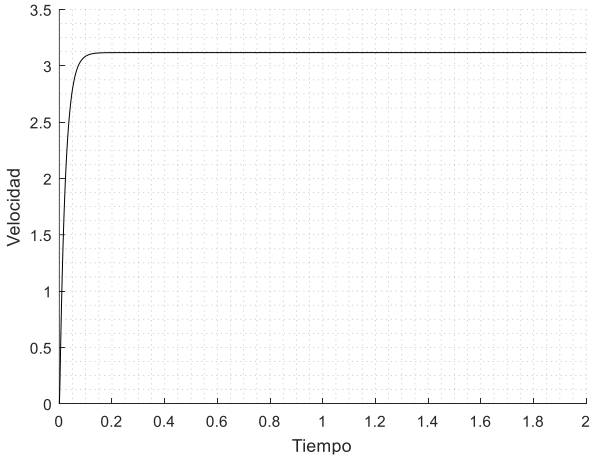


Figura 3.4.2 Respuesta del sistema ante una entrada de U=10 v.

3.4.2 Controlador PD-PI y PI

Los controles que le permiten al robot Ackerman realizar el seguimiento de trayectorias son controles de posición para el actuador de la dirección y velocidad para el actuador de la tracción, en el caso del robot diferencial, solo son necesarios los controladores de la velocidad, debido a que el hardware del robot no permite la medición de corriente del motor, se propone un controlador PID para la tracción, en el caso del control de la posición, se puede aplicar la derivación mediante diferencias finitas para obtener la velocidad del actuador mientras se mide la posición.
Para realizar el control de velocidad se propone un regulador PID con un bucle de control simple, basado en lo dicho por [14, 15], un controlador PID tiene la forma:

$$U = K_p e(t) + K_i \int_0^t e(t)dt + K_d \dot{e}(t) \tag{44}$$

Donde el error para el control de la velocidad es calculado como:

$$e_{vel} = Ref_{vel} - Vel_{medida} \tag{45}$$

Así *U* es la señal que se usara para el control de nuestra planta, en el caso del control de la posición, el controlador de esta precede al control de velocidad, dejando que las ecuaciones para el control de velocidad sean:

$$e_{pos} = Ref_{pos} - Pos_{medida} \tag{46}$$

$$Ref_{vel} = \left(e_{pos} * Kp_{pod}\right) + Kd_{pos}\dot{e}_{pos}(t) \tag{47}$$

$$e_{vel} = Ref_{vel} - Vel_{medida} \tag{48}$$

$$U = (e_{vel} * Kp_{vel}) + Ki_{vel}\int_{0}^{t} e_{vel} \tag{49}$$

Para diseñar los controladores PID y PD-PI en cascada del sistema se utilizaron distintos métodos heurísticos para encontrar las ganancias del controlador maestro, y en el caso del controlador de la posición, el esclavo, las ganancias respectivas de ambos controladores son optimizadas simultáneamente, las ecuaciones dinámicas son resueltas mediante Runge-Kutta en cada paso en el que el algoritmo de optimización hace una evaluación de la función objetivo.

Las respuestas de los controladores optimizados se muestran en las figuras 3.4.6 y 3.4.9, mientras que la evolución de las ganancias se encuentra en las figuras 3.4.7 y 3.4.10, tomando como función objetivo la suma del error absoluto, los algoritmos bio-inspirados buscan las variables de diseño óptimas que minimicen el error respetando las restricciones [16] donde t_r, M_p, E_{ss} son el tiempo de asentamiento, el sobre-impulso máximo, y el error en estado estable, respectivamente.

El cálculo de las constantes de control con una respuesta óptima puede ser complicado por métodos analíticos, tomando un controlador PI convencional como un problema de optimización de 2 dimensiones se formularon las superficies de las figuras 3.4.3 y 3.4.4, estas fueron obtenidas como respuesta a la función objetivo:

$$f(x) = \sum_{i=1}^{m} \left|Ref_{v_i} - Vel_i\right| + C\delta \tag{50}$$

$$C = 30\begin{cases}\delta = 1 & si\ M_{si} > Ref_v + Ref_v * 0.01 \\ \delta = 0 & si\ M_{si} \leq Ref_v + Ref_v * 0.01\end{cases} \tag{51}$$

También, en las figuras 3.4.3 y 3.4.4 se aprecia que el problema de optimización es un problema multi-modal debido a que contiene varios óptimos globales y que, dependiendo de las constantes del motor, así como de las restricciones, el área de búsqueda resulta ser diferente incluso usando el mismo tipo de controlador para un miso tipo de motor, en este caso un controlador PI y un motor PMDC

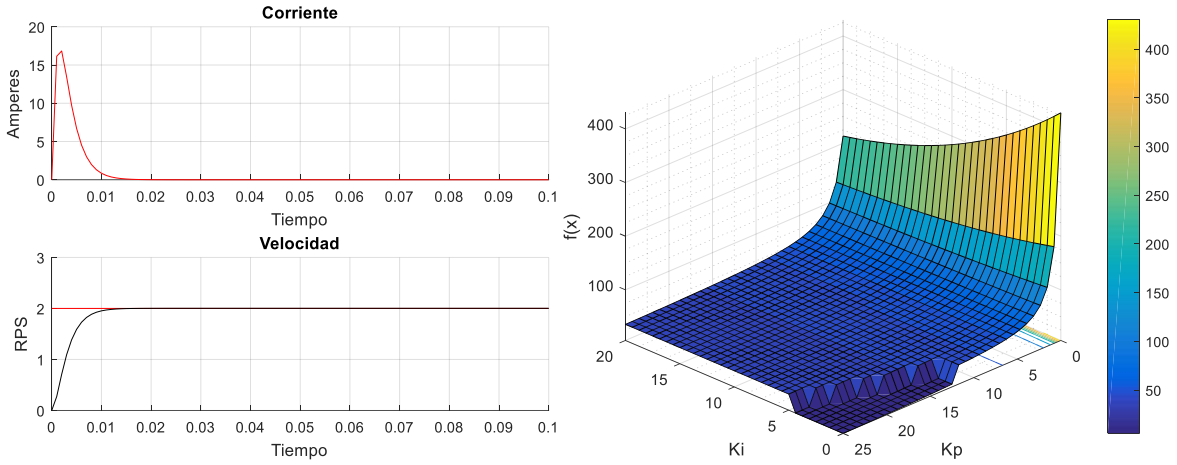


Figura 3.4.3 Comportamiento de un motor PMDC con las ganancias calculadas mediante la ecuación (50) (a), representación del control PI en función de la ecuación (50) (b).

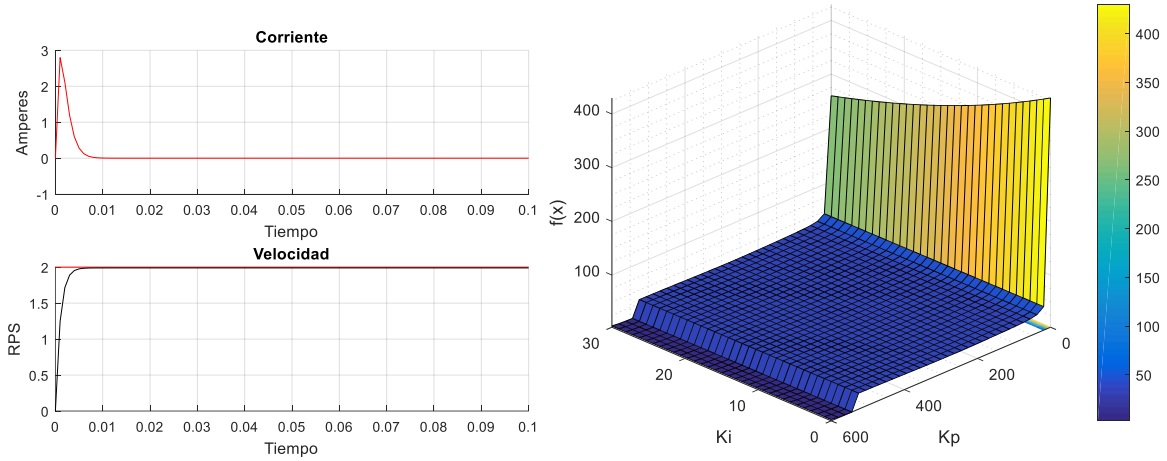


Figura 3.4.4 Comportamiento del motor lego con las ganancias calculadas mediante la ecuación (50) (a), representación del control PI en función de la ecuación (50) (b).

3.4.3 Control de la velocidad

Debido a las limitaciones de hardware en la plataforma de desarrollo, para el control de la velocidad de los motores lego, se optó por un controlador PID sin bucles de retroalimentación internos, dejando únicamente el bucle de retroalimentación de la velocidad, representado mediante la figura 3.4.5.

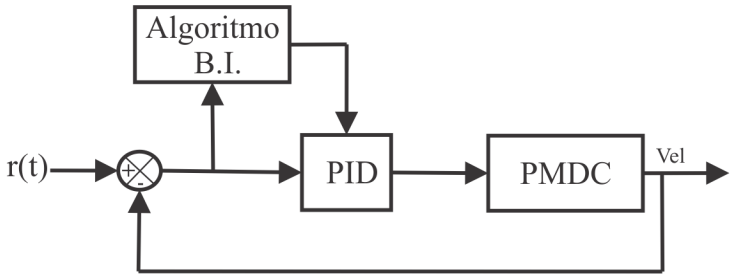


Figura 3.4.5 Diagrama a bloques de la sintonización del controlador PID mediante algoritmos bio-inspirados.

Función objetivo:

$$f(x) = \sum_{i=1}^{m} \left| Ref_{v_i} - Vel_i \right| + C_1\delta_1 + C_2\delta_2 \tag{52}$$

$$C = inf \begin{cases} \delta = 1 & si\ M_{si} > Ref_v + Ref_v * 0.01 \\ \delta = 0 & si\ M_{si} \leq Ref_v + Ref_v * 0.01 \end{cases} \tag{53}$$

$$C = inf \begin{cases} \delta = 1 & si\ E_{ss} > |Ref_v - Ref_v * 0.01| \\ \delta = 0 & si\ E_{ss} \leq |Ref_v - Ref_v * 0.01| \end{cases} \tag{54}$$

$$0 > Kp_{pos} \leq 300 \quad 0 > Kd_{pos} \leq 300 \quad 0 > Kp_{vel} \leq 300 \tag{55}$$

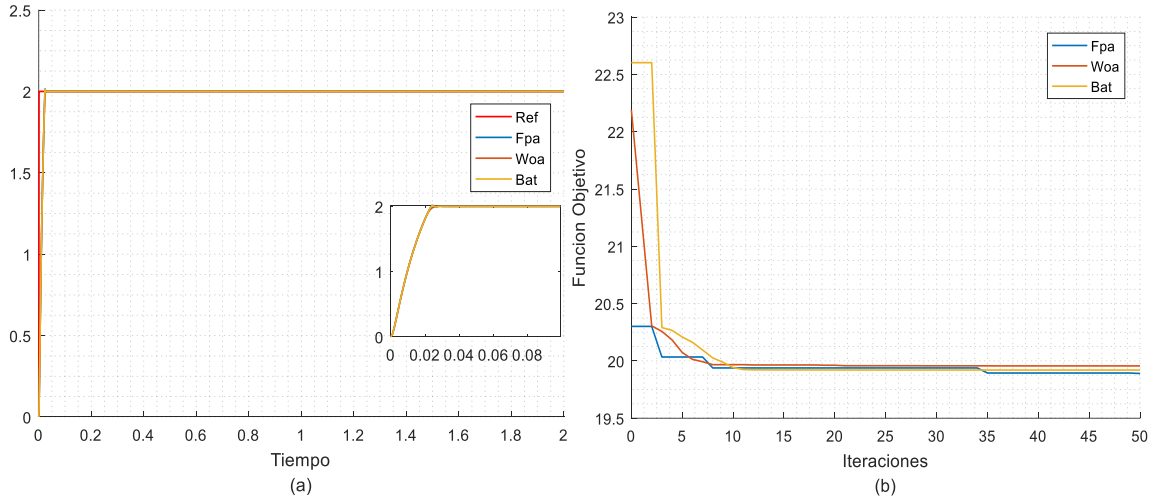


Figura 3.4.6 Respuesta para el control de la velocidad con las ganancias óptimas (a), curva de convergencia de los tres métodos heurísticos (b)

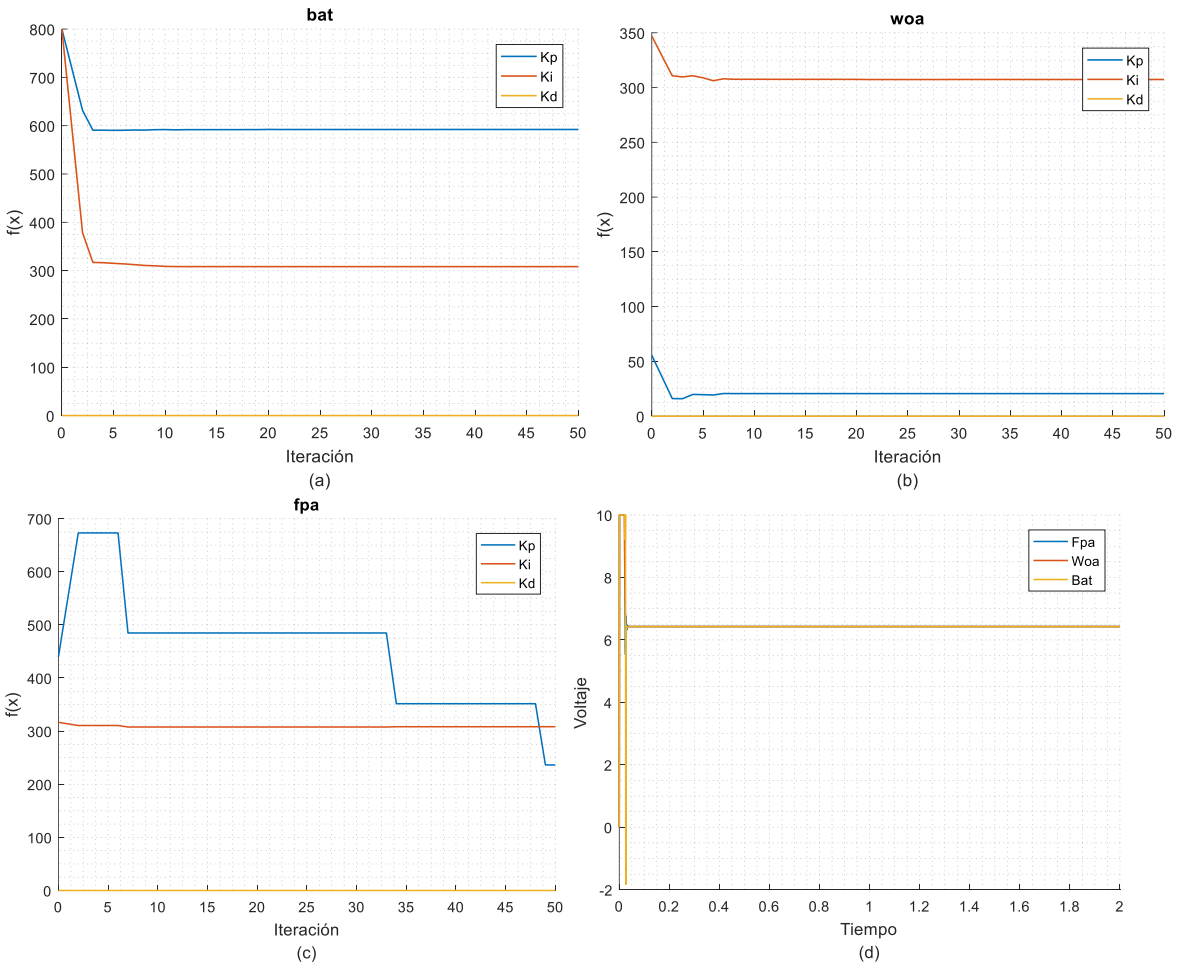


Figura 3.4.7 Evolución en las ganancias con el algoritmo BAT(a), con WOA(b), con FPA(c), respuesta del controlador, con una saturación en U=10 (d).

TABLA 3.4-1. Ganancias óptimas y valor de la función objetivo para el control de velocidad del motor PMDC.

	K_p	K_i	K_d	$Fun.obj.$
FPA	236.321	308.303	0.0	19.89
WOA	20.604	307.362	0.0	19.956
BAT	592.161	308.297	0.0	19.920

3.4.4 Control de la posición

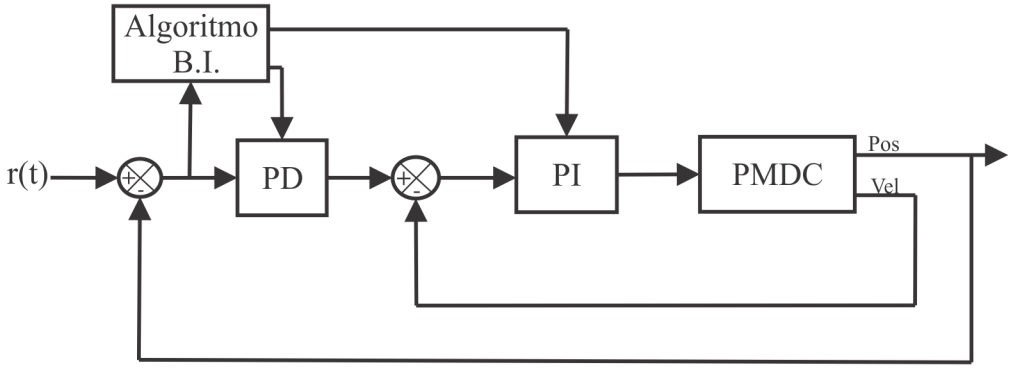


Figura 3.4.8 Diagrama a bloques de la sintonización del controlador de posición-velocidad mediante algoritmos bio-inspirados.

Función objetivo:
La función objetivo empleada en el cálculo de las ganancias del control de velocidad se presenta en la ecuación (56):

$$f(x) = \sum_{i=1}^{m} \left| Ref_{pos_i} - pos_i \right| + C_1\delta_1 + C_2\delta_2 \tag{56}$$

$$C = inf \begin{cases} \delta = 1 & si\ M_{si} > Ref_{pos} + Ref_{pos} * 0.01 \\ \delta = 0 & si\ M_{si} \leq Ref_{pos} + Ref_{pos} * 0.01 \end{cases} \tag{57}$$

$$C = inf \begin{cases} \delta = 1 & si\ E_{ss} > \left| Ref_{pos} - Ref_{pos} * 0.01 \right| \\ \delta = 0 & si\ E_{ss} \leq \left| Ref_{pos} - Ref_{pos} * 0.01 \right| \end{cases} \tag{58}$$

$$0 > Kp_{pos} \leq 300 \quad 0 > Kd_{pos} \leq 300 \quad 0 > Kp_{vel} \leq 300 \quad 0 > Ki_{vel} \leq 300 \tag{59}$$

Resultados de la Optimización del controlador para el motor lego:

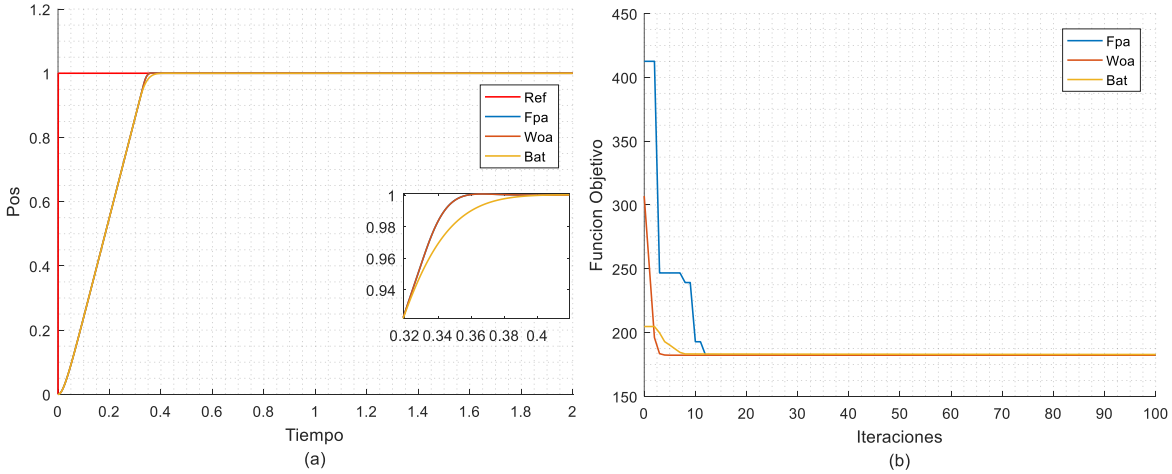


Figura 3.4.9 Respuesta del controlador posición-velocidad con las ganancias encontradas por las distintas heurísticas (a), curva de convergencia de los algoritmos (b).

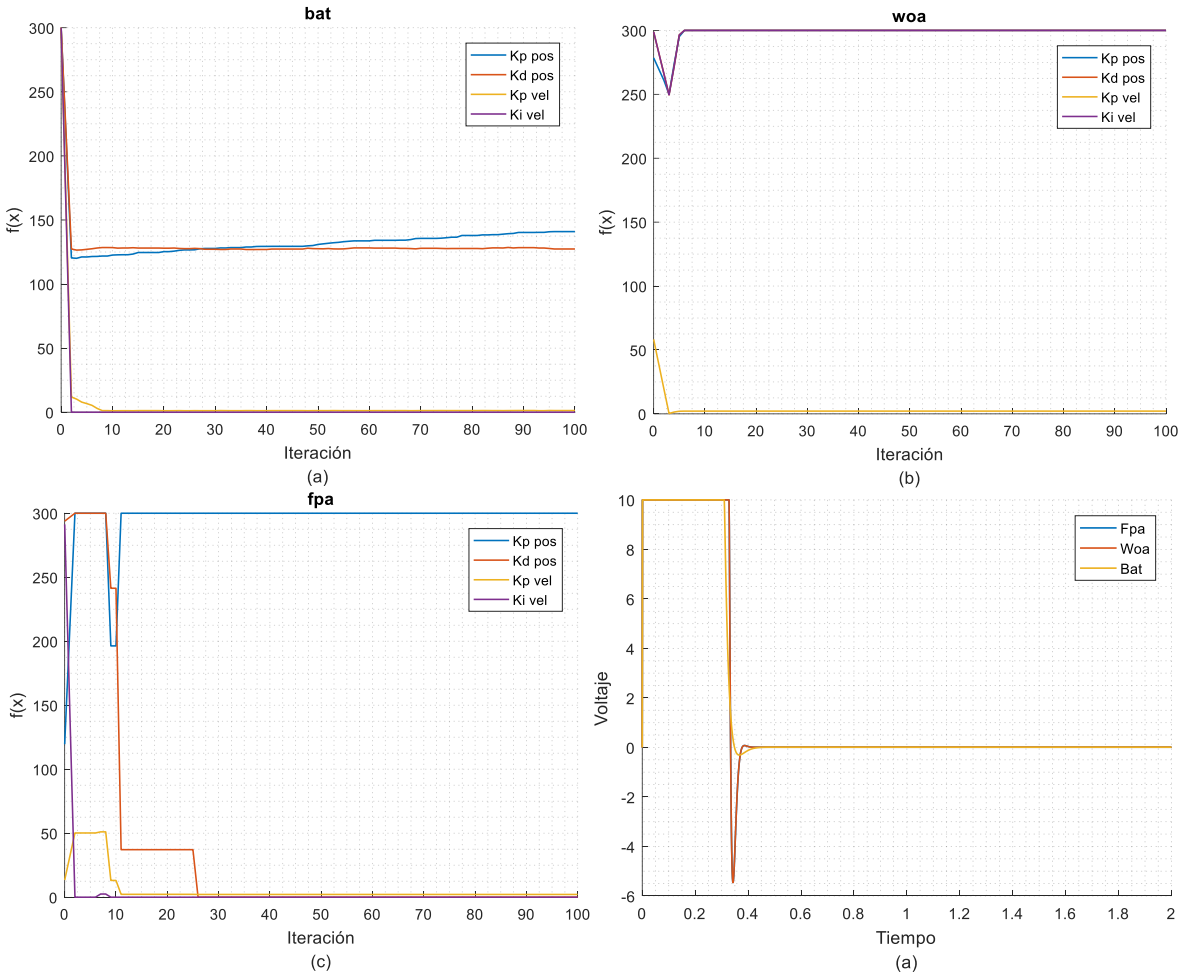


Figura 3.4.10 Evolución en las ganancias con el algoritmo BAT(a), con WOA(b), con FPA(c), respuesta del controlador, con una saturación en U=10 (d).

TABLA 3.4-2. Ganancias óptimas y valor de la función objetivo para el control de posición del motor PMDC.

	$K_{p\ pos}$	$K_{d\ pos}$	$K_{p\ vel}$	$K_{i\ vel}$	$Fun.obj.$
FPA	300.000	0.000	2.143	0.000	182.2
WOA	300.000	300.000	2.148	0.000	182.2
BAT	141.202	127.099	1.337	1.337	182.3

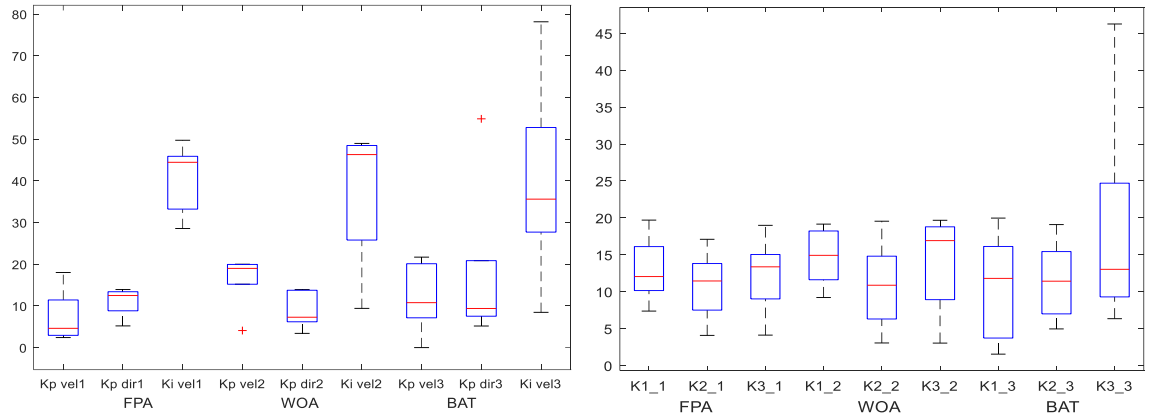


Figura 3.4.11 Distribución de las variables de diseño encontradas por los algoritmos de optimización correspondientes al control PID y al control no lineal del vehículo ackerman

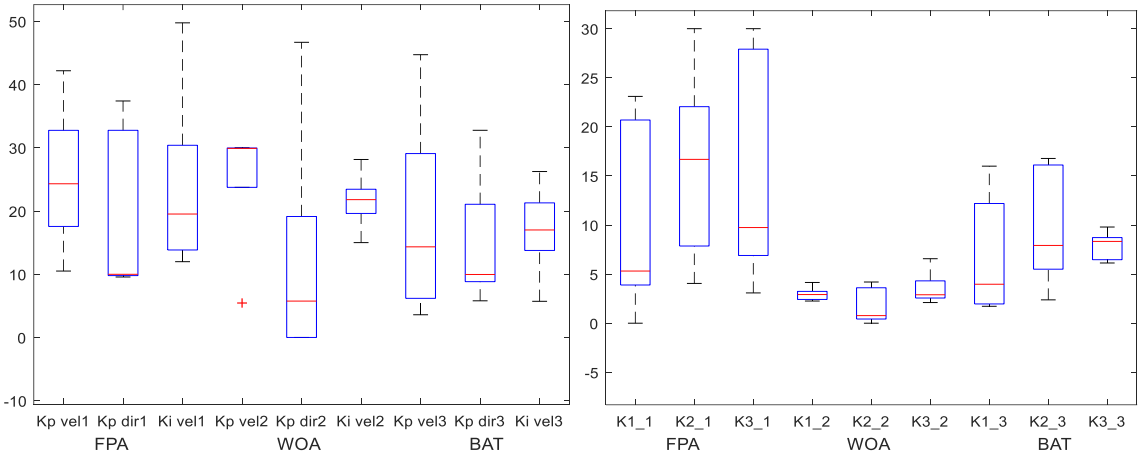


Figura 3.4.12 Distribución de las variables de diseño encontradas por los algoritmos de optimización correspondientes al control PID y al control no lineal del vehículo diferencial.

Una característica importante es que, un sistema de control visto desde la perspectiva de la optimización posee características que convierten a la función objetivo en una función multimodal , en las figuras 3.4.11 y 3.4.12, correspondientes a los controles del vehículo ackerman y al vehículo diferencial muestran que los controles cinemáticos para este robot son multimodales, debido a que un rango de valores en las variables de diseño tienen representan un mismo valor en la función objetivo, esto se puede apreciar en el rango de las cajas en las gráficas, donde ambos extremos de los valores corresponden al 25% y al 75% de los valores de todas las pruebas de los algoritmos, los extremos con líneas punteadas corresponden a los valores mínimos y máximos encontrados en cada corrida de los algoritmos de optimización, y la línea en el medio corresponde a la media de cada una variable de diseño entre cada corrida de los algoritmos.

3.5 Conclusiones

El cálculo de controladores mediante métodos convencionales puede ser una tarea compleja, particularmente si se consideran varias salidas de una planta o la planta tiene características no lineales, en este capítulo se realizó la experimentación del cálculo de las constantes de control para las distintas etapas de control de los robots móviles con ruedas, donde la primera etapa incluye las relaciones geométricas del robot, la segunda hace referencia al control de los actuadores como un

sistema mimo en el caso de la dirección y un sistema siso en el caso del control de velocidad del motor, debido a las características de la plataforma no se pudo incluir un lazo de control de par en los motores, aun así, el cálculo de las ganancias en las simulaciones tanto como en la experimentación resulto ser optimo, dando como resultado el menor tiempo transitorio y ningún sobre-impulso, así como un error nulo en el estado estacionario, el uso de algoritmos heurísticos resulta ser una herramienta eficaz en el cálculo de diversas leyes de control, así como en distintos niveles de control de robos, abriendo camino para experimentación en jerarquías más altas de control y en toma de decisiones autónomas dentro de los robots. Las figuras 3.2.5, 3.2.7, 3.2.9, 3.2.11 muestran que los algoritmos probados tienen una distinta actuación dependiendo del tipo de control al que se apliquen, esto, particularmente no significa que un algoritmo sea mejor que otro algoritmo, sino, que, respecto al problema de optimización que se plantee, un algoritmo puede mostrar una mejor actuación que otros en ese problema específico.

3.6 Referencias

[1] G.Cook “Mobile robots: navigation, control and remote sensing”, Institute of Electrical and Electronics Engineers, 2011, ISBN 978-0-470-64021-1.

[2] G. Dudek, M. Jenkin “Computational principles of mobile robotics 2nd ed.”, Cambridge University Press 2010, ISBN 978-0-521-87157-0.

[3] P.I. Corke, “Robotics, Vision & Control”**,** Springer 2011, ISBN 978-3-642-20143-1.

[4] T. Fukao, H. Nakagawa, and N. Adachi, “Adaptive tracking control of a nonholonomic mobile robot,” IEEE Trans. Robot. Autom., vol. 16, no. 5, pp. 609–615, 2000.

[5] J. L. Avendaño-Juárez,V. M. Hernández-Guzmán, and R. Silva Ortigoza, “Velocity and current inner loops in a wheeled mobile robot,” Adv. Robot., vol. 24, no. 8–9, pp. 1385–1404, 2010.

[6] Neenu Thomas, Dr. P. Poongodi “Position Control of DC Moto Using Genetic Algorithm Based PID Controller”, Proceedings of th world Congress on Engineering 2099 Vol II, WCE 2009, Juy 1 – 3, 2009, London U.K. ISBN:978-988-18210-1-0

[7] Lee, Kunwoo “Principles of CAD/CAM/CAE systems” Addison Wesley Longman, Inc. (1999), ISBN 0-201-38036-6

[8] T.Back, D.B.Fogel, Z. Michalewicz “Handbook of Evolutionary Computation”, IOP Publishing Ltd. Bristol, UK, 1997, ISBN: 0750303921

[9] Siedlecki W and Sklansky J 1989 Constrained genetic optimization via dynamic reward–penalty balancing and its use in pattern recognition Proc. 3rd Int. Conf. on Genetic Algorithms (Fairfax, VA, June 1989)ed J D Schaffer (San Mateo, CA: Morgan Kaufmann) pp 141–50

[10] Ion Boldea, S.A Nasar: Electric drives.CRC Press(1999) ISBN 0-8293-2521-8

[11] José Acedo Sánchez: Control avanzado de procesos (teoría y práctica). ISBN: 84-7978-545-4.

[12] J. R. G. Sánchez, S.T.Mosqueda, R. S. Ortigoza, M.A. Cruz, G S. Ortigoza and J. de J Rubio, “Assessment of an Avarage Trackin Controlles that Considers all Subsystems Involved in a WMR Implementation via PWM or Sigma-Delta Modulation”, IEEE Latin American Transactions, Vol 14, No 2, March 2016.

[13] C. M. Sánchez, J. R. G. Sánchez, C. Y. S. Cervantes, R. S. Ortigoza, V. M. H. Guzmán, J. N. A. Juárez, and M. M. Aranda “Trajectory Generation for Wheeled Mobile Robots Via Bézier Polynomials, IEEE Latin american Transactions.

[14] Katsuhiko Ogata:” Ingenieria de control moderna”. PEARSON EDUCACIÓN, S.A., 2010

[15] Cetinkunt, Sabri.”Mechatronics with experiments, Second edition” ISBN 987-1-118-80246-5, TJ163.12.C43 2015

[16]D. Puangdownreong, A.Nawikavatan, C. Thammarat: Optimal Design of I-PD Controller for a DC Motor Speed Control System by Cuckoo Search, 2016 International Electrical Engineering Congress, iEECON2016, 2-4 March 2016, Chiang Mai, Thailand

Capítulo 4

Instrumentación y resultados

4.1 Introducción

En este capítulo se abordan el método usado para la navegación del vehículo autónomo, así como los sensores utilizados que permiten que el robot conozca su posición respecto del plano de referencia a su vez se tratan dos métodos para generar trayectorias, siendo estos las curvas paramétricas y las curvas de Bézier, también se mencionan los resultados de implementación y trabajos futuros.

4.2 Navegación

La navegación es una de las competencias más desafiantes requeridas en la robótica móvil, el éxito en la navegación requiere éxito en cuatro bloques básicos [1]:

- Percepción: el robot debe interpretar sus sensores y extraer los valores significativos.
- Localización: El robot debe determinar su posición en el ambiente.
- Cognición: el robot debe decidir cómo actuar para llegar a su meta.
- Control del movimiento: El robot debe modular las salidas de sus motores para lograr la trayectoria deseada.

De estos cuatro componentes, la localización ha recibido gran atención por parte de investigadores en la década pasada y, como resultado, avances significativos han sido hechos en esta área, el método utilizado en este trabajo, conocido como "Navegación por estima" es uno de los métodos más usados para la localización, en el apartado 4.4.1 se explica más a detalle este método.

4.2.1 Sensado

El sensado es un requisito fundamental para cualquier comportamiento del robot, para que el robot pueda actuar este debe ser capaz de sensar y razonar acerca de las respuestas de sus sensores, El sensado es un componente crítico de la tarea fundamental de la estimación de la posición (determinar donde el robot está en el área de trabajo), mantenimiento de la posición (mantener el estimado de la posición del robot), y la construcción de mapas, construye una representación del ambiente del robot.

Esta necesidad del sensado es ejemplificada cuando un robot ejecuta un movimiento continuamente, cuando la rueda se desliza la verdadera posición se desvía de su posición comandada, sin mediciones externas del mundo exterior, esta desviación rápidamente resulta en un robot "perdido" [2].

4.2.2 Conceptos básicos

La clasificación más fundamental de sensores es dentro de los Sensores de "estados internos" (sensores propioceptivos) y los sensores de "estados externos" (sensores extereoceptivos). Los sensores de estados internos proveen retroalimentación de los parámetros internos del sistema robótico, tales como nivel de la Batería, la posición de los neumáticos, o los ángulos de las juntas, los sensores de estados externos tratan con la observación de aspectos del mundo fuera del mismo robot, como la humedad y el color de los objetos. También, son conocidos como sensores activos los que emiten energía al realizar sus observaciones, y como sensores pasivos, se les conoce a los sensores que reciben energía para hacer sus observaciones [3].

Hay observaciones amplias, pero críticas que pueden ser hechas con respecto a los sensores en el contexto de la robótica móvil:

- Los sensores son sensibles a las perturbaciones.
- Los sensores retornan una descripción incompleta del ambiente
- Los sensores no pueden ser (regularmente) sensados completamente

Subestimar la medida que cada una de estas 3 generalizaciones tiene puede llevar a dificultades considerables cuando algoritmos "ideales" son aplicados en el mundo real.

Una variedad amplia de tecnologías en los sensores está disponible, esta plétora de opciones puede ser ampliamente descompuesta en 4 clases principales basados en el tipo de datos regresados [2]:

- Sensores de rango: retornan mediciones de la distancia entre el sensor y objetos en el ambiente.
- Sensores de la posición absoluta: retornan la posición del robot (o algunos elementos del vector posición) en términos absolutos (ejemplo. latitud y longitud)
- Sensores ambientales: regresan las propiedades del ambiente estas pueden ser propiedades tales como temperatura, o con propiedades puntuales tales como el color en un punto frente al robot
- Sensores de inercia: retornan las propiedades diferenciales de la posición del robot, por ejemplo, la aceleración.

Una variedad de cuestiones adicionales puede ser usada para clasificar diferentes tecnologías de los sensores, algunos ejemplos las propiedades importantes de los sensores [2]:

- Velocidad de operación: es la razón a la cual las mediciones son regresadas cuando el sensor opera continuamente, o el retraso de la medición cuando esta es pedida intermitentemente.
- Costo: un sensor infrarrojo puede costar poco en comparación con un giroscopio preciso
- Razón del error: varios factores pueden ser relevantes entre ellos el promedio del error, el número de mediciones incorrectas, y el número de mediciones perdidas
- Robustez: esto se refiere al grado en el cual el sensor tolera varias desviaciones ambientales de las condiciones de operación ideales, los factores relevantes pueden ser perturbaciones físicas, ruido ambiental, en términos del estímulo de interés, también del ruido eléctrico

- Requisitos computacionales: un sensor de contacto puede no requerir esfuerzo computacional para interpretar su resultado, sin embargo, ciertos sensores requieren algoritmos con cierta complejidad computacional e incluso procesadores de matrices de propósito especial para obtener un resultado en tiempo.
- Potencia, peso, y requisitos del tamaño: algunos sistemas requieren potencia continua para que las lecturas de los sensores puedan ser mientras que otros sensores pueden no estar alimentados hasta que las mediciones sean requeridas

Los principales sensores usados en la robótica se pueden clasificar como

Sensores de contacto: son usados para crear un "sentido del contacto", a pesar de que algunos aspectos del contacto pueden ser capturados por sensores sin contacto que pasar cerca de la superficie de un objeto y explotan fenómenos tales como la capacitancia.

Sensores inerciales: la clase de sensores externos que hacen la última referencia al mundo exterior son conocidos como sensores de inercia, estos sensores miden las derivadas de las variables de posición del robot, en particular, el termino sensor inercial es comúnmente usado para referirse a acelerómetros y giroscopios, los cuales miden la segunda derivada de la posición que es la aceleración, y la aceleración angular del vehículo, estos son:

- Acelerómetros: los acelerómetros son esencialmente masas montadas en resortes cuyo desplazamiento bajo aceleración puede ser medido explotando la ley de newton $F=ma$ y la (ideal) relación masa resorte $F = kx^2$, resolviendo para a resulta en

$$a = \frac{kx^2}{m} \tag{1}$$

  Donde a es la aceleración, m es la masa y k es la constante del resorte, en la práctica, alternativas para el sistema masa resorte son usadas, cada medición acelerómetro es en una sola dirección, montando 3 acelerómetros ortogonales uno a otro, uno puede fabricar un sistema de aceleración omnidireccional, tales dispositivos están disponibles en encapsulados. Los acelerómetros son vistos como sensores que no necesitan hacer referencia al mundo externo, y mientras esto es casi cierto, el vector de gravedad local debe ser conocido en dirección y magnitud, para que pueda ser descartado de las mediciones del acelerómetro, con esta excepción, estos sensores tienen buenas características para ser usados donde sea.

- Giroscopios: Los giroscopios funcionan a partir de la medición de la variación de ángulos que se basa en el efecto de la aceleración de Coriolis [4], consisten esencialmente en 2 masas oscilando radialmente en los extremos de fibras similares a cartílagos formando un diapasón. Cuando se produce un cambio en la orientación, las fuerzas de Coriolis generan pares proporcionales a las velocidades angulares de giro. La principal dificultad con la estimación de la orientación de giroscopio son los pequeños errores en la medición sobre el tiempo [2]. La pérdida gradual de precisión debido a la acumulación del error referida como "drift", en los sistemas mecánicos el drift es resultado de la fricción de las inevitables imperfecciones de los rodamientos del giroscopio, la razón del crecimiento de este error puede variar de pocos a decenas de grados por segundo dependiendo de la precisión del maquinado.

- Giroscopios electrónicos: son normalmente sensores de velocidad angular que emplean el efecto de coriolis mencionado, para ello se realizan micro mecanizados del silicio configurando un anillo que se hace vibrar a una frecuencia de resonancia, el movimiento de rotación produce fuerzas de coriolis que dependen de la velocidad de giro. La medida de la velocidad se obtiene determinando la diferencia de las vibraciones a diferentes ángulos, un sensor típico puede tener dimensiones de 2 milímetros y permite medir hasta 100 grados por segundo.

4.3 Sistemas de navegación inercial

Un sistema de navegación inercial emplea acelerómetros para determinar la aceleración en cada uno de los tres ejes de movimiento, las velocidades y posiciones se obtienen mediante doble integración. Combinando esta información con la posición original da la posición actual del vehículo en el espacio de inercia, si hay algún sesgo en los acelerómetros, el proceso de la doble integración crea un error que crece como el cuadrado del tiempo, eventualmente tornándose inaceptable, algunos medios auxiliares de determinar la posición son requeridos para la re-calibración de la unidad inercial, el tiempo entre re-calibraciones es dictado por los requerimientos de precisión y la medida del sesgo del acelerómetro. Los sesgos pueden variar desde cientos de mili-g's (milésima de la aceleración producida por la gravedad) a unos cuantos milli-g's. En adición a estos errores de la posición causados por los sesgos del acelerómetro, existen los errores de orientación causados por el "drift", el sistema cree que esta rotando cuando no es así, la razón del drift puede ser tan grande como pocos grados por hora, y tan pequeña como pocos mili-grados por hora [2].

Los errores de orientación no crecen tan rápido como los errores de posición ya que las estimaciones no sufren de la doble integración de los sesgos como los estimados de la posición lo hacen

En vehículos robóticos terrestres es frecuente que las aceleraciones sean reducidas, por lo que la relación señal-ruido es baja, dificultando la medida correcta. Asimismo, las posiciones estimadas acumulan derivas, necesitándose sensores de gran calidad para obtener valores aceptables del error de posición, es también importante que la plataforma este perfectamente estabilizada por lo cual es necesario emplear giroscopios de calidad para la medida de las orientaciones [4].

Por todo ello, los sistemas de navegación inercial, para obtener precisiones razonables, son muy costosos. Sin embargo, el empleo de giroscopios ópticos y acelerómetros de estado sólido tiende a mejorar la situación.

4.3.1 Orientación del robot

La orientación es importante en el estudio del movimiento de cualquier tipo de vehículo como aviones, barcos y en este caso robots móviles, en la figura 4.4.2, el robot móvil tiene un marco propio de referencia, este cuadro se mueve con el robot y es llamado marco del robot, el eje X está alineado con el eje longitudinal del robot, mientras que el eje Y apunta a los lados del robot, el eje Z apunta hacia arriba en este sistema. Este tipo de marco es comúnmente usado en la robótica.

Según el robot se mueve, experimenta translación o cambio en la posición, en adición a esto, puede experimentar rotación o cambio en la orientación, las rotaciones son definidas como:

- Yaw (viraje): este es la rotación a través del eje Z en dirección a las manecillas del reloj, visto desde el eje Z.
- Pitch (inclinación): Rotación a través del eje Y, también en el sentido a las manecillas del reloj
- Roll (balanceo): Rotación a través del eje X, también medido en el sentido de las manecillas del reloj

4.4 Localización

A partir del modelo cinemático del robot Ackerman se puede formular un modelo discreto para su uso en la navegación, localización y odometria, claramente, las ecuaciones que definen el sistema son no lineales, así, los métodos utilizados para convertir los sistemas lineales en tiempo continuo no son aplicables[5], un método posible es utilizar el método de integración de Euler, este método en es una aproximación de primer orden a las series de Taylor para integrar, y dice que la derivada puede ser aproximada por la diferencia finita:

$$\dot{x}(t) \approx \frac{x(t+\Delta t) - x(t)}{\Delta t} \tag{2}$$

Que pueden ser reacomodados de la siguiente forma:

$$x(t+\Delta t) \approx x(t) + \dot{x}(t)\Delta t \tag{3}$$

Haciendo un cambio de variables de $t=kT$ y el periodo de muestreo $\Delta t = T$ y aplicando esto a las ecuaciones cinemáticas se obtiene:

$$x(k+1) = x(k) + V(k) * \sin\psi\,(k) \tag{4}$$

$$y(k+1) = y(k) + V(k) * \cos\psi\,(k) \tag{5}$$

$$\psi(k+1) = \psi(k) + \frac{V(k)}{L} * \tan a\,(k) \tag{6}$$

4.4.1 Dead reckoning (Navegación por estima)

La navegación por estima típicamente usa encoders o dispositivos similares para medir la rotación angular de las ruedas, la siguiente formula es usada para convertir estas mediciones en distancia viajada

$$AS = r\Delta\theta \tag{7}$$

Donde r es el radio de la rueda, esto produce el largo de la trayectoria viajada por el neumático, pero esto no resulta en una nueva posición ya que no contiene información de la curvatura de la trayectoria viajada

En la odometria (sensado de los neumáticos) y el dead reckoning (navegación por estima) la actualización de la posición está basada en sensores "propioceptivos", el movimiento del robot, sensado con encoders en los neumáticos, sensores de dirección o ambos, es integrado para computar la posición, debido a que los errores de los sensores son integrados, el error en la posición se acumula con el tiempo, así la posición tiene que ser actualizada cada cierto tiempo por otros mecanismos de localización, de otra forma, el robot no es capaz de mantener una posición significativa en largas distancias, el uso de sensores de dirección (como el giroscopio) pueden ayudar a reducir los errores acumulativos, pero los problemas principales permanecen intactos.

Existen varias fuentes para los errores odometricos, desde factores ambientales hasta la resolución [2]:

- Resolución limitada durante la integración (incrementos del tiempo, resolución de la medición);
- Desalineación de las ruedas (error determinístico)
- Incertidumbre en el diámetro de las ruedas y/o una diferencia en el diámetro de las ruedas (error determinista)
- Variaciones en el punto de contacto de las ruedas.
- Contacto No-igual en el suelo (deslizamiento, superficies no planares, etc.)

Algunos errores pueden ser deterministas, estos pueden ser eliminados mediante una calibración propia del sistema, aun así, siguen existiendo errores no determinísticos (o aleatorios) que permanecen, conduciendo a incertidumbres en la estimación de la posición con el tiempo. Estos errores se pueden clasificar en 3 tipos:

- Error de rango: la integración del largo del recorrido del robot no es la correcta.
- Error de giro: similar al error de rango, para las vueltas, puede ser ocasionado por las diferencias en los movimientos de las ruedas
- Error por deslizamiento: la diferencia en el error de las ruedas conduce a un error en la orientación angular del robot.

Aquí el periodo de muestreo T debe ser escogido lo suficientemente pequeño dependiendo de la dinámica del sistema de ecuaciones diferenciales original. El comportamiento del modelo discreto debe empatar con el del sistema original. Para un sistema lineal este corresponde a seleccionar el periodo de muestro para ser un quinto de la constante de tiempo más pequeña, o más pequeño dependiendo del grado de precisión que se requiera, para sistemas no lineales, puede ser necesario determinar este tamaño de forma empírica.

Debido a las características del vehículo, al efectuar la navegación por estima se utiliza un giroscopio, este al otorgar como medida la velocidad angular mostrada en la figura 4.4.3 (yaw rate, ángulo de la dirección) tiene que ser integrado para obtener la orientación y la posición en coordenadas del robot,

en lugar de obtener la posición angular del motor que conduce la dirección lo cual es mostrado en la figura 4.4.1, el modelo en tiempo discreto para ambos arquetipos de robot de la siguiente forma:

$$x(k+1) = x(k) + V(k) * \sin\psi\,(k) \tag{7}$$

$$y(k+1) = y(k) + V(k) * \cos\psi\,(k) \tag{8}$$

$$\psi(k+1) = \psi(k) + G_{iroscopio}(k) \tag{9}$$

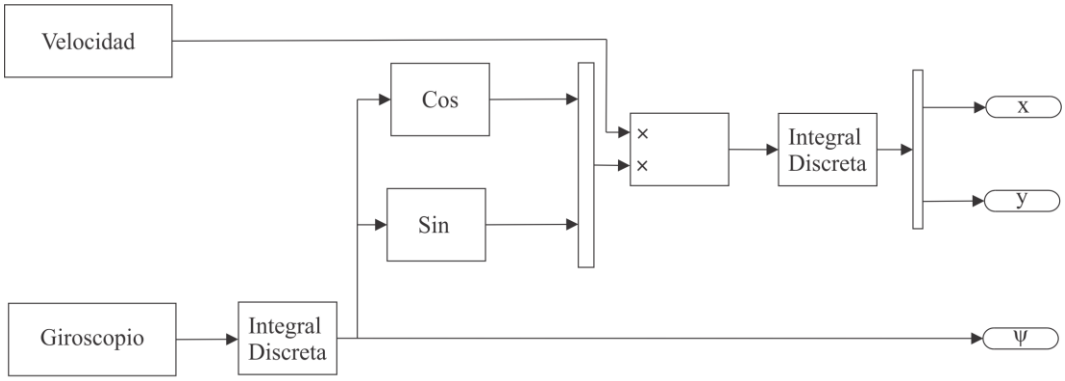


Figura 4.4.1 Diagrama a bloques para la estimación de la posición del robot.

La posición puede ser estimada si se inicia desde una posición conocida mediante la integración del movimiento. Con las condiciones iniciales en los bloques de integración el robot puede asumir cualquier posición y orientación deseada, dejando las condiciones iniciales de los integradores en 0, estos valores colocan al robot en el centro del plano de coordenadas de referencia como se muestra en la figura 4.4.2, con una orientación inicial paralela al eje X del plano de coordenadas.

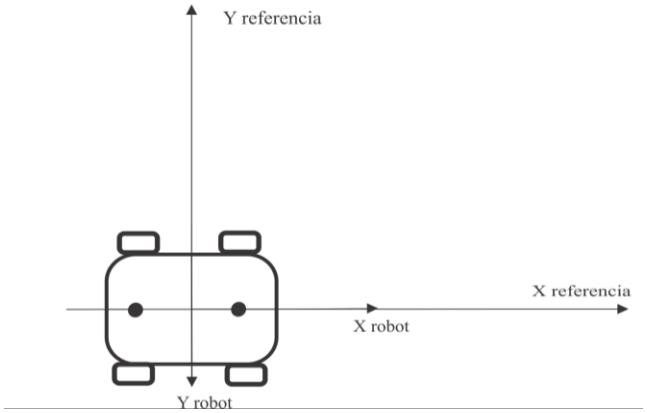


Figura 4.4.2 Marco de referencia del robot.

Los instrumentos de medición del robot nos otorgan la posición angular de la flecha del motor mediante un encoder incremental, lo cual resulta ser particularmente útil en el control de la dirección,

para poder extraer la velocidad de la tracción con respecto de la posición del eje de los neumáticos se hace uso de una derivada discreta, la salida de esta alimenta la variable de velocidad del modelo utilizado para obtener la localización del robot. El uso del giroscopio en la estimación de la orientación del robot es importante debido a que este no es susceptible a fallos mecánicos, como el juego entre las juntas del sistema de dirección, como lo sería la estimación basada en el encoder que mide la posición de la dirección, también, el giroscopio mide la orientación de forma externa al robot, lo que significa que cualquier perturbación externa es captada por este, lo que no sucedería con la tecnología de los encoders.

Una vez obtenida una localización correcta mediante la estima, los valores de posición y orientación del robot se comparan con los de la trayectoria, conduciendo a las leyes de control para el vehículo.

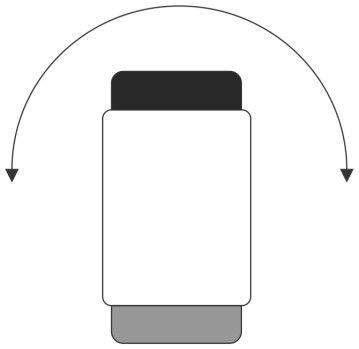


Figura 4.4.3 Ángulo de viraje con respecto del giroscopio.

4.5 Calibración de los instrumentos.

El algoritmo de calibración de los instrumentos (giroscopio) consiste en obtener el promedio de la señal del instrumento durante cierto periodo de tiempo, posteriormente este promedio se le resta a la señal del instrumento como se muestra en la figura 4.4.4, así se elimina el drift del giroscopio y se calcula cualquier velocidad angular a partir de la velocidad que detecta el aparato estando estático. Este proceso se repite cada vez que el robot se pone en funcionamiento debido a las variaciones que pueda tener el offset entre cada puesta en marcha

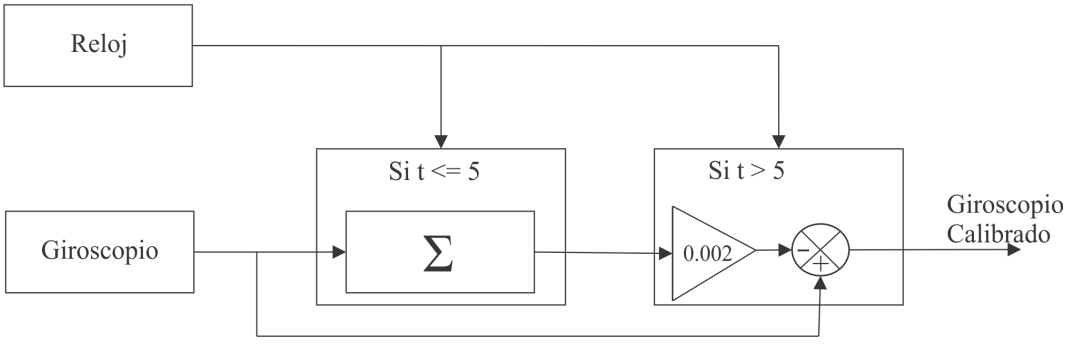


Figura 4.5.1 Diagrama a bloques del algoritmo de calibración.

4.6 Obtención de los parámetros de los motores de la Dirección

Para poder realizar la simulación de los motores, y posteriormente diseñar una ley de control se toman los parámetros del motor mínimos necesarios, estos son:

- K_e Constante de fuerza electromotriz (Webers)
- B Constante de Fricción
- J_m Constante de Inercia
- R_a Resistencia del estator (ohms)
- L_a Inductancia del motor (henrios)

La resistencia y la inductancia del motor se obtienen de forma sencilla haciendo mediciones con un multímetro y un multímetro LCR (Inductancia, Capacitancia, Resistencia, por su simbología) [6], eso da los siguientes resultados:

$$R_a = 4.5\ \Omega \tag{10}$$

$$L_a = 5.10 * 10^{-3}\ L \tag{11}$$

La fricción viscosa se asume que es 0 y se procede a calcular el resto de las constantes, en [7] se menciona que, el cálculo de la constante de fuerza electromotriz es la relación entre el voltaje aplicado y las revoluciones por minuto:

$$K_e = \frac{V}{RPS} = \frac{5}{\frac{93.5}{60}} = 3.208\ Wb \tag{12}$$

que representa la constante de fuerza electromotriz del motor de corriente directa., A su vez, al torque electromagnético se obtiene mediante:

$$K_t = \frac{K_e}{2\pi} = \frac{3.208}{2\pi} = 0.5106 \tag{14}$$

Bajo estado estable el par de carga es igual al par motor [7], entones en nuestro caso

$$T_{carga} = T_{elec} = K_t * I_a = 0.5105 * 0.0011 = 0.0005973 \tag{15}$$

El cálculo de la constante de inercia se puede realizar de forma paramétrica con las constantes conocidas y reportadas con antelación mediante la ecuación.

$$J_m = \frac{tm * K_t * K_e}{R_a} = \frac{0.0005973 * 0.5106 * 3.208}{4.5} = 0.0002176 \tag{16}$$

4.7 Generación de trayectorias

4.7.1 Curvas paramétricas

Las curvas paramétricas surgen en varias aplicaciones, tales como diseño industrial, matemáticas discretas, arquitectura y en este caso son usadas para la generación de trayectorias de robots, para

permitir la modificación el análisis y la visualización de las curvas en sus aplicaciones, se deben de ejecutar esas operaciones se requiere de la representación matemática de las curvas.

Los puntos de coordenadas de una curva paramétrica están expresados como función de una variable o parámetro como *t* [8]. Una curva en el plano tiene la forma $\boldsymbol{C}(t) = \big(x(t), y(t)\big)$, y una curva en el espacio tiene la forma $\boldsymbol{C}(t) = (x(t), y(t), z(t))$. Las funciones de $x(t), y(t), z(t)$ son llamadas *funciones de coordenadas.* La imagen de $\boldsymbol{C}(t)$ es llamada *trazo* de $\boldsymbol{C}$, y $\boldsymbol{C}(t)$ el cual es denominado como parametrización de $\boldsymbol{C}$. Un subsegmento de la curva $\boldsymbol{C}$ el cual también es una curva es conocido como *segmento de curva*. Una curva paramétrica definida por una función de coordenadas polinomial es denominada como *curva polinomial*.

Las curvas paramétricas comprenden las parábolas, los círculos y las secciones de círculos, las espirales, y las hélices, ejemplos de estas curvas pueden ser observados en la figura 4.7.1.

Un circulo o una porción de este con un radio *R* y con centro (X_c, Y_c) pueden ser representados por las ecuaciones:

$$x = Rcos(\theta) + X_c \tag{17}$$

$$y = Rsin(\theta) + Y_c \tag{18}$$

$$0 \leq \theta \leq 2\pi \tag{19}$$

Las curvas pueden ser evaluadas substituyendo los valores de θ con un incremento de $\Delta\theta$ que inicia desde 0 hasta 2π para completar el circulo, o hasta cierto valor del arco circular, el valor de $\Delta\theta$ tiene que ser escogido apropiadamente debido a que valores muy pequeños pueden causar ineficiencias computacionales, mientras que valores muy grandes puedes convertir el circulo en un polígono [8].

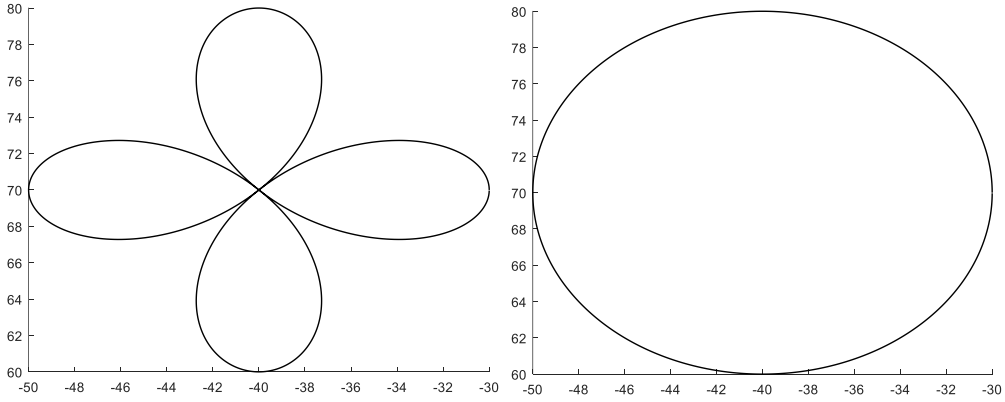


Figura 4.7.1 Ejemplo de curvas paramétricas, rosa polar (a), circulo (b).

4.7.2 Curvas de Bézier

Dos de las representaciones matemáticas de las curvas y superficies usadas en gráficos de computadora y diseño asistido por computadora son las curvas de Bézier y las formas b-spline. El desarrollo original de las curvas de Bézier tomo lugar en la industria automotriz durante el periodo

de 1958-60 por los franceses Pierre Bézier en Renault y Paul de Casteljau en Citröen, el desarrollo de las B-splines siguió la publicación en 1946 por el documento [9] en splines.

Las curvas de Bézier son curvas polinomiales las cuales tienen representaciones matemáticas particulares, su popularidad se debe al hecho de que poseen un número de propiedades matemáticas que facilitan su manipulación y análisis sin un profundo conocimiento matemático para usar las curvas [10]. Una curva de grado *n* es especificada por una secuencia de *n+1* puntos los cuales son llamados *puntos de control*. El polígono obtenido uniendo los puntos de control con segmentos lineales en el orden prescrito es llamado *polígono de control.* [11-13].

El control de la forma de la Curva de Bézier es facilitado por el hecho de que el polígono de control refleja la forma básica de la curva.

4.7.2.1 Curvas de Bézier lineales

Una curva de Bézier lineal es un segmento de línea formado por 2 puntos de control $\boldsymbol{b_0}(p_0, q_0)$ y $\boldsymbol{b_1}(p_1, q_1)$ parametrizadas por:

$$\big(x(t), y(t)\big) = (1-t)(p_0, q_0) + (t)(p_1, q_1) \qquad para \quad t \in [0,1] \tag{20}$$

Así que $x(t) = (1-t)(p_0) + (t)(p_1)$, así como $y(t) = (1-t)(q_0) + (t)(q_1)$. Permitiendo que $\boldsymbol{B}(t) = \big(x(t), y(t)\big)$la curva puede ser escrita en la forma vectorial como

$$\boldsymbol{B}(t) = (1-t)(\boldsymbol{b_0}) + (t)(\boldsymbol{b_1}) \tag{21}$$

La curva está definida en el intervalo de [0,1], así que el punto inicial de la curva él $\boldsymbol{B}(0) = (\boldsymbol{b_0})$ y el punto final es $\boldsymbol{B}(1) = (\boldsymbol{b_1})$, esto es, que la curva de Bézier interpola el primer y el último punto de control [8].

4.7.2.2 Curvas de Bézier cuadráticas

Supóngase tres puntos de control $\boldsymbol{b_0}(p_0, q_0)$, $\boldsymbol{b_1}(p_1, q_1)$ y $\boldsymbol{b_2}(p_2, q_2)$**,** son especificados. Entonces la curva cuadrática de Bézier está definida por:

$$\boldsymbol{B}(t) = (1-t)^2(p_0, q_0) + 2(1-t)(p_1, q_1) + (t)^2(p_2, q_2) \qquad para \quad t \in [0,1] \tag{22}$$

La curva también está definida en el intervalo de [0,1], así que el punto inicial de la curva él $\boldsymbol{B}(0) = (\boldsymbol{b_0})$ y el punto final es $\boldsymbol{B}(1) = (\boldsymbol{b_1})$, esta puede ser expresada de forma paramétrica $\big(x(t), y(t)\big)$ donde:

$$x(t) = (1-t)^2(p_0) + 2(1-t)(p_1) + (t)^2(p_2) \qquad para \quad t \in [0,1] \tag{23}$$

$$y(t) = (1-t)^2(q_0) + 2(1-t)(q_1) + (t)^2(q_2) \qquad para \quad t \in [0,1] \tag{24}$$

4.7.2.3 Curva de Bézier cubica

Supóngase cuatro puntos de control $\boldsymbol{b}_0, \boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3$ son especificados, la curva de Bézier está definida por:

$$\boldsymbol{B}(t) = (1-t)^3\boldsymbol{b}_0 + 3(1-t)^2(t)\boldsymbol{b}_1 + 3(1-t)(t^2)\boldsymbol{b}_2 + (t^3)\boldsymbol{b}_3 \qquad para \quad t \in [0,1] \qquad (25)$$

Así como en el caso cuadrático, el polígono obtenido de unir los puntos de control en el orden especificado es llamado “polígono de control”.

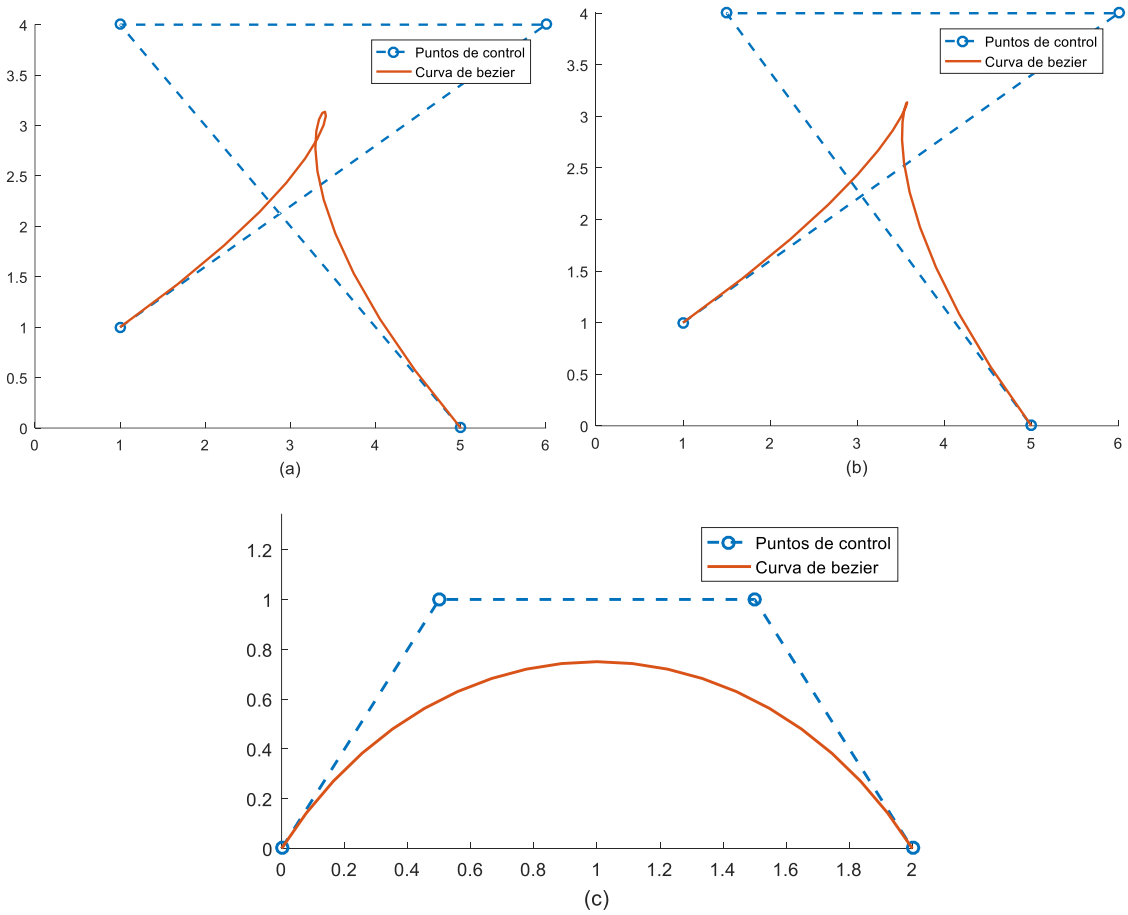


Figura 4.7.2 Representaciones de curvas de Bézier cubicas.

Las curvas de Bézier cubicas proveen un mayor rango de formas que las curvas cuadráticas, ya que estas pueden exhibir bucles como el mostrado en la figura 4.7.2 (a), curvas agudas (llamadas cúspides) como en la figura 4.7.2 (b), e inflexiones [10].

4.7.2.4 La curva general de Bézier

Dados *n+1* puntos de control $\boldsymbol{b}_0, \boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ la curva de Bézier de grado *n* está definida como:

$$\mathbf{B}(t) = \sum_{i=0}^{n} \mathbf{b}_i B_{i,n}(t) \tag{26}$$

Donde:

$$B_{i,n}(t) = \begin{cases} \dfrac{n!}{(n-i)!i!}(1-t)^{n-i}t^i, & si\ 0 \le i \le n \\ 0 & de\ otra\ \ forma \end{cases} \tag{27}$$

Es llamado polinomio de *Bernstein* o *Funciones bases de Bernstein de grado n*. Para distinguir las curvas de Bézier de las curvas racionales de Bézier, nos referimos a estas como curvas integrales de Bézier. Los casos donde *n=1*, *n=2*, *n=3*, corresponden a las curvas lineales, cuadráticas y cubicas de Bézier.

Las cantidades $\dfrac{n!}{(n-i)!i!}$ son llamadas *coeficientes binomiales* y son denotados por $\binom{n}{i}$ o ${}^nC_i$.

¡Recordemos la convención de que 0! =1, entonces $\binom{n}{i} = \dfrac{n!}{n!0!} = 1$ y que $\binom{n}{n} = \dfrac{n!}{0!n!} = 1$

4.8 Resultados de la implementación en el robot

Para la implementación del robot se usan los algoritmos de control y navegación mencionados en los capítulos anteriores para lograr el seguimiento de trayectorias generadas mediante curvas paramétricas y curvas de Bézier, esto fue implementado en un par de robots construidos con el Kit de robótica Mindstorms NXT de Lego, los cuales fueron programados usando Simulink, los resultados de la implementación se muestran en este apartado, varias pruebas fueron realizadas con las distintas ganancias de control calculadas por los algoritmos.

El seguimiento de trayectorias con las ganancias calculadas mediante en algoritmo WOA son mostradas en las figuras 4.8.3 y 4.8.4, la razón por la que solo se muestran estos resultados es por la similitud entre pruebas con las distintas ganancias, pruebas que incluían la perturbación del robot en su recorrido fueron hechas para demostrar la correcta inclusión de un giroscopio en la navegación por estima del vehículo, tales resultados no son mostrados por haber sido parte en las pruebas predecesoras a aquellas que incluían las curvas de Bezier, donde la trayectoria era seleccionada como una trayectoria punto a punto. Los prototipos cad de ambos arquetipos de robot se presentan en las figuras 4.8.1 y 4.8.2

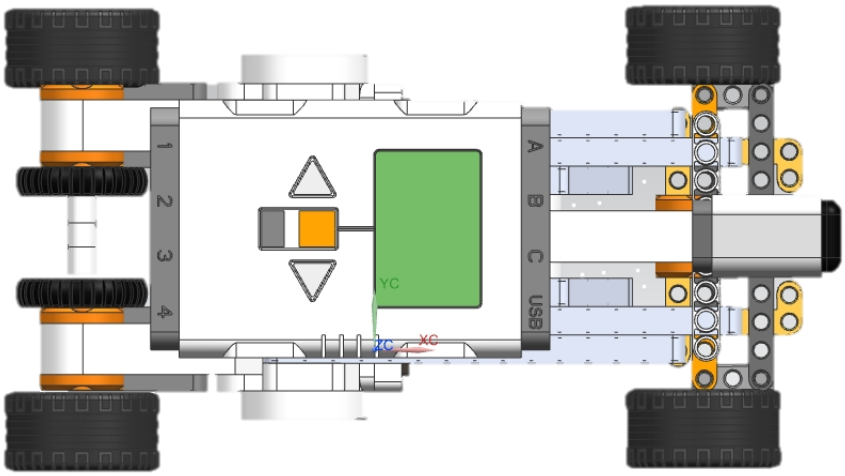


Figura 4.8.1 Modelo CAD del Robot Ackerman.

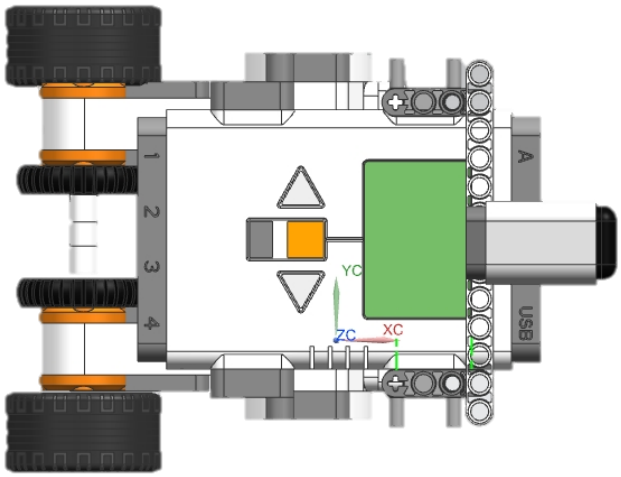


Figura 4.8.2 Modelo CAD del robot Diferencial.

De las figuras 4.8.1 y 4.8.2 se puede apreciar la locomoción y la arquitectura de ambos robots, donde la propulsión es dada por los dos motores de dc en la parte trasera estos cuentan con un encoder incremental integrado, en la locomoción diferencial solo estos son necesarios para dar direccionamiento, en el caso ackerman es necesario un sistema de dirección. También se muestra la posición en la cual es montado el giroscopio, ya que este solo detecta la rotación en un sentido, este debe ser acorde al sentido de rotación de los robots móviles, una descripción mas detallada de la plataforma puede encontrarse en el capítulo 1.

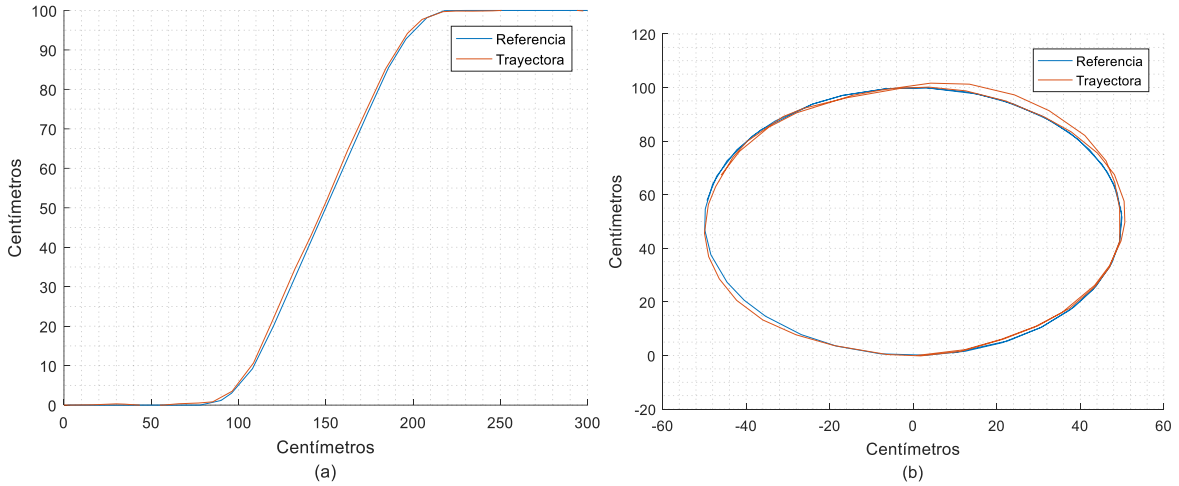


Figura 4.8.3 Resultados de implementación de la ley PI, P del robot Ackerman.

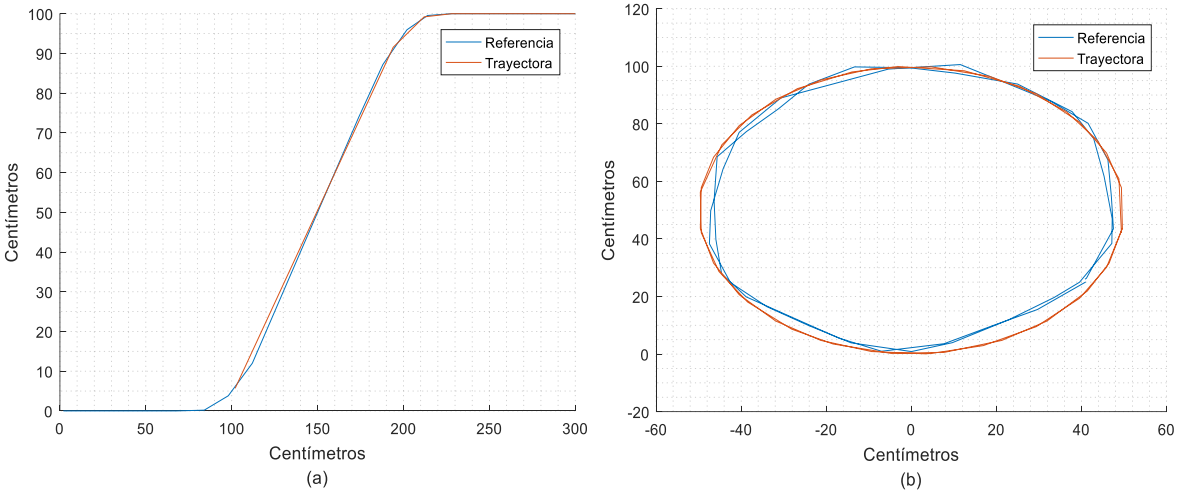


Figura 4.8.4 Resultados de implementación de la ley PI, P del robot diferencial.

4.9 Conclusiones

En este trabajo se presentó el problema de seguimiento de trayectorias para robots móviles, los cuales muestran distintos desafíos como la selección de ganancias que regulen de forma correcta a los actuadores de los robots en sus lazos de control, así como el control cinemático que permite que el robot siga una trayectoria trazada en un plano cartesiano.

Diversos algoritmos fueron planteados para resolver este problema, basándose principalmente en la selección de dos esquemas de control cinemático y en los esquemas de control de los actuadores, estos algoritmos de optimización fueron puestos en marcha para calcular unos parámetros tales, que permitan el seguimiento de las trayectorias con un error mínimo entre la posición del robot y la trayectoria deseada, basado en los resultados de las simulaciones se esperaba que el comportamiento real del robot fuese muy similar a reportado por estos algoritmos, tal implementación cumplió con lo esperado, revelando que el uso de algoritmos de optimización aplicados a la robótica da buenos

resultados en la selección de ganancias de control para las etapas necesarias que necesite un robot para cumplir su tarea.

4.10 Trabajos Futuros

En las investigaciones posteriores se planea realizar una ley de control que contemple la reversa y la orientación del vehículo, esto debido a que, en aplicaciones más reales, el área de trabajo puede ser reducida o estar restringida en cuanto a la orientación del vehículo, una tarea que puede ejemplificar este problema es el estacionar un vehículo en paralelo.

También se planea extender el problema de la generación de trayectorias, esto, haciendo uso de algoritmos inteligentes que busquen trayectorias con ciertas características, como la evasión de obstáculos, o la máxima cobertura de un área, así como el diseño basado en las restricciones cinemáticas del sistema.

Un tercer punto que se planea abordar en el futuro es el del mapeo (ya sea continuo, o discreto) del ambiente, como parte importante de la planeación inteligente de trayectorias, utilizando técnicas de "SLAM", con esto, el cuarto desafío posterior es la introducción de mecanismos más avanzados de localización, tales como el filtro de partículas, o el filtro kalman extendido, los cuales permiten la fusión de sensores, y con esto obtener una estimación de la posición más robusta que la navegación por estima.

Actualmente se está trabajando en un segundo prototipo cuya función es la poda de las áreas verdes en la institución, y métodos para la evasión de obstáculos basados en la percepción.

4.11 Referencias

[1] R. Siegwart, I. Noubakhsh, “Introduction to autonomous mobile robots”, Massachusetts institute of Technology, 2004, ISBN 0-262-19502

[2] G. Dudek, M. Jenkin “Computational principles of mobile robotics 2nd ed.”, Cambridge University Press 2010, ISBN 978-0-521-87157-0.

[3] Cetinkunt, Sabri. ”Mechatronics with experiments, Second edition” ISBN 987-1-118-80246-5, TJ163.12.C43 2015

[4] A. Ollero BATurone, Robótica: Manipuladores y Robots Móviles. Barcelona, Spain: arcombo-Boixareu Editores, 2001.

[5] G.Cook “Mobile robots: navigation, control and remote sensing”, Institute of Electrical and Electronics Engineers, 2011, ISBN 978-0-470-64021-1.

[6] Chapman, S. J. (2012). Máquinas Eléctricas. México D.F.: McGraw Hill.

[7] Ion Boldea, S.A Nasar: Electric drives.CRC Press(1999) ISBN 0-8293-2521-8

[8] K. Lee. “Principles of CAD/CAM/CAE Systems”, ISBN 0-201-38036-6

[9] Schoenberg, I, ‘Contributions to the problem of approximation of equidistant data by analytic functions’,Quart. Appl. Math.Vol. 4, pp45–99, 1946.

[10] D. Marsh, “Applied Geometry for Computer Graphics and Cad 2nd edition”. ISBN 1-85233-801-6

[11] Bezier, P, “Style, mathematics and NC”. Computer-Aided Design Vol. 22, No. 9, pp. 523–526, 1990.

[12] Davis, P, “B-splines and geometric design”, SIAM NewsVol. 29 No. 5,1996.

[13] Forrest, A R, “Interactive interpolation and approximation by Bezier polynomials”. Computer-Aided Design Vol. 22 No. 9, pp527–537, 1990. Originally published in The Computer Journal Vol. 15 No. 1, pp71–79, 1972.