TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE ENSENADA
DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

MAESTRÍA EN CIENCIAS EN INGENIERÍA MECATRÓNICA

Síntesis de comportamientos analíticos aplicados al control de sistemas multirobot.

Trabajo de tesis presentado por Rodrigo Alexandro Villalvazo Covián
para obtener el grado de Maestro en Ciencias.

Bajo la dirección de Dr. Eddie Helbert Clemente Torres
y la co-dirección de Dra. Iliana Marlen Meza Sánchez.

ensenada, baja california, méxico, junio 2022

EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico de Ensenada
División de Estudios de Posgrado e Investigación

Ensenada, Baja California, 24/Junio/2022
No. Oficio: DEPI/012/2022
Asunto: Autorización de impresión de Trabajo de Tesis

C. RODRIGO A. VILLALVAZO COVIÁN
ESTUDIANTE DE LA MAESTRÍA EN CIENCIAS
EN INGENIERÍA MECATRÓNICA
PRESENTE

Con base en el dictamen de aprobación emitido por el Comité Tutorial de la Tesis con título: SÍNTESIS DE COMPORTAMIENTOS ANALÍTICOS APLICADOS AL CONTROL DE SISTEMAS MULTIROBOT, entregado por Usted para su análisis, le informamos atentamente que se AUTORIZA la impresión de dicho trabajo de tesis; en virtud de que satisface los requisitos señalados por las disposiciones reglamentarias.

ATENTAMENTE
Excelencia en Educación Tecnológica®

EDUCACIÓN | TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE ENSENADA
DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

EUSEBIO BUGARIN CARLOS
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

ccp. Archivo

EBC

MAESTRÍA EN CIENCIAS EN INGENIERÍA MECATRÓNICA
DICTAMEN DEL COMITÉ TUTORIAL

Ensenada, B.C. a 24 de junio de 2022

DATOS DEL ESTUDIANTE			
Nombre:	**Rodrigo Alexandro Villalvazo Covián**	**No. de Control:**	**M13760303**

DATOS DEL PROYECTO DE TESIS	
Título:	Síntesis de comportamientos analíticos aplicados al control de sistemas multirobot.
Línea de Investigación:	Sistemas Cognitivos

COMITÉ DE TESIS		
Miembro	**Nombre**	**Firma**
DIRECTOR DE TESIS (Director del comité)	Eddie Helbert Clemente Torres	
CODIRECTORA DE TESIS	Iliana Marlen Meza Sánchez	
MIEMBRO DEL COMITÉ	Gustavo Olague Caballero	
MIEMBRO DEL COMITÉ	Ismael Hernández Capuchin	
Dictamen:	Aprobado por unanimidad	

Josefina Campos García
Coordinadora Académica

Ana Yaveni Aguilar Bustos
Presidente del Consejo de Posgrado

STA_10_dictamen_tesis_comite_tutorial_V1.1

CARTA DE CESIÓN DE DERECHOS

Ensenada, Baja California, a 23 de junio de 2022

Bajo protesta de decir verdad, quien suscribe **RODRIGO ALEXANDRO VILLALVAZO COVIÁN**, con número de control **M13760303**, alumno de la **Maestría en Ciencias en Ingeniería Mecatrónica** y con la dirección de Eddie Helbert Clemente Torres, declaro que soy autor del trabajo original de Tesis

SÍNTESIS DE COMPORTAMIENTOS ANALÍTICOS APLICADOS AL CONTROL DE SISTEMAS MULTIROBOT.

Y otorgo mi conformidad, sin límite de temporabilidad, para ceder los derechos de reproducir y distribuir copias en su totalidad o en partes al Tecnológico Nacional de México campus Instituto Tecnológico de Ensenada (TecNM-ITE), mencionando la fuente. Lo anterior incluye que la tesis pueda ser publicada y difundida por el TecNM-ITE en los medios que considere convenientes. Asimismo, manifiesto que el TecNM-ITE queda liberado de cualquier conflicto surgido a raíz de la publicación. De igual manera, es de mi conocimiento que la publicación de la tesis no es con finalidad lucrativa, sino académica.

ATENTAMENTE

RODRIGO ALEXANDRO VILLALVAZO COVIÁN
ALUMNO DE LA MAESTRÍA EN CIENCIAS
EN INGENIERÍA MECATRÓNICA

STA_11_carta_cesion_derechos_V1.1

Resumen

El presente trabajo aborda el diseño de una metodología para la síntesis automatizada de un conjunto de controladores para resolver el problema de navegación colaborativa de tipo *rendezvous* en sistemas multirobot. Esta propuesta hace uso del paradigma de los comportamientos analíticos. Específicamente, se aborda el analísis con dos robots móviles omnidireccionales con tres objetivos generales de control para dar solución a la tarea colaborativa de *rendezvous*. La formulación de los controladores se define como la combinación de tres controladores parciales. Los dos primeros controladores parciales se obtienen de la literatura y resuelven el problema de regulación con evasión de obstáculos. El tercer controlador parcial se deriva de una formulación basada en comportamientos analíticos para mantener una convergencia a la posición deseada donde se especifica un tiempo de llegada predefinido. El paradigma de los comportamientos analíticos establece que la salida de cada robot (esto es, el comportamiento "natural") es la suma de tres comportamientos base definidos como *no forzado* (*unforced*), *forzado* (*forced*) y *aprendido* (*learned*). Al aplicar el control por comportamientos analíticos, el comportamiento *no forzado* se define como el comportamiento del robot donde sólo se consideran condiciones iniciales, el comportamiento *forzado* corresponde al comportamiento del robot cuando se le aplica la suma de los controladores parciales para lograr llegar a las posiciones deseadas evadiendo obstáculos, y finalmente, el comportamiento *aprendido* es el comportamiento de los robot al tercer controlador parcial que busca modificar el comportamiento imponiendo un tiempo de llegada. En este trabajo, se extiende esta formulación para el sistema de dos robots móviles omnidireccionales donde los tres objetivos de control deben ser alcanzados. El paradigma de comportamientos analíticos hace uso de la Programación Genética (GP, del inglés Genetic Programming) para construir de forma automatizada el tercer controlador parcial que da origen a los comportamientos *aprendidos* en los robots. Mediante la construcción de una función de optimización para guiar la búsqueda del GP, se genera un conjunto de controladores no lineales que resuelven de forma óptima los tres problemas planteados. El sistema multirobot resuelve el problema donde el *rendezvous* confina las posiciones finales a un área designada que ha sido desarrollada usando polígonos inscritos. Los controladores sintetizados son clasificados con respecto a su aptitud para resolver el problema planteado, y los mejores han sido validados numéricamente para evaluar su desempeño. Se presentan simulaciones que

pretenden mostrar la contribución de las soluciones para un escenario en donde se incrementa el número de robots que forman parte del sistema multirobot.

Palabras clave: Control por comportamientos analíticos, Navegación autónoma, Campos de velocidad, Redezvous, Multiples robots.

Abstract

In this work, the design of a methodology that automates the synthesis of nonlinear controllers to solve a "rendezvous" cooperative task in multi-robot systems is outlined. This proposal takes advantage of the analytical behaviors framework. Specifically, the "rendezvous" problem applied to two omnidirectional mobile robots with three control objectives is addressed. The formulation of the nonlinear controllers is defined as the combination of three partial controllers. The first two partial controllers correspond to state-of-the-art controllers that solve the "rendezvous" task with obstacle avoidance. The third controller is derived from the analytic behaviors paradigm to maintain the convergence of the robots to the desired position while the arrival time is predefined. The analytic behaviors framework defines that the output of each robot (this is, their "natural" behavior) is composed of three basis behaviors denoted as the "unforced", "forced", and "learned" behaviors. The "unforced" is given as the behavior of the robot denoted by its dynamics subject to initial conditions, the "forced" behavior is the response of the robots to the application of the sum of the first two partial controllers from the state-of-the-art, and finally, the "learned" behaviors is given by the response of the robots to the addition of the third partial controller which modifies the behavior to fulfill the specific arrival time policy. The formulation for the "rendezvous" cooperative task in a system of two omnidirectional mobile robots is addressed in this work where the three defined control objectives must be achieved. The analytic behaviors framework applies the Genetic Programming paradigm for the automated construction of the third partial controller that gives rise to the *learned* behaviors in the robots. The construction of an optimization function that guides the search of the GP, a set of nonlinear controllers that optimally solve the three control objectives, is built. The multi-robot system solves the "rendezvous" cooperative task where the desired positions are set in a designated area built by means of inscribed polygons. The synthesized controllers are classified with respect to their aptitude (*fitness*) to solve the formulated task, and the best are selected and tested numerically. Simulations increasing the number of robots in the multi-robot system are provided to highlight the virtues of the proposed framework.

Keywords: Analytic Behavior Control, Autonomous Navigation, Velocity Fields, Rendezvous, Multi-Robot.

Dedicatoria

Para todos mis seres queridos y aquellas personas que creen en mí.

Agredezco con cariño:

A mis padres por su enorme apoyo y sacrificio que hicieron posible la oportunidad de cursar esta carrera tan maravillosa, además de haberme apoyado para lograr este logro profesional, porque sin ellos no sería nada.

A mis hermanos por hacer más alegre mi vida, por su sacrificio durante mi formación, por estar ahí a mi lado siempre.

A la Dra. Iliana Marlen Meza Sánchez, por apoyarme antes y durante, por instruirme y darme consejos durante mi formación, además de brindarme su confianza en muchas formas.

Al Dr. Eddie Helbert Clemente Torres por apoyarme cuando lo necesité, por brindarme consejos y estar ahí durante el desarrollo de este proyecto y la guía brindada.

Al Dr. Gustavo Olague, el Dr. Ismael Capuchin, la Dra. Maria del Carmen y a todos aquellos profesores e investigadores que formaron parte de mi formación en este arduó proceso para crecer como profesionista y como persona.

Al Instituto Tecnológico de Ensenada y al personal que ahí labora, por haberme dado la oportunidad de realizar en la institución una de mis muchas metas por cumplir.

Y a todas aquellas personas que estuvieron ahí desde el principio hasta este momento.

1

Introducción

El concepto de robótica se introduce en 1921 en el libro "Rossum´s Universal Robots" escrito por Karel Capek, donde se define la palabra "robota" como la fuerza del trabajo, dando inicio a los primeros estudios relacionados con los robots. Sin embargo, fue hasta la década de los 50s que comenzaron a desarrollarse los primeros autómatas, que tan solo eran capaces de ejecutar tareas básicas y con un consumo energético descomunal. Fue hasta la década de los 70s que la robótica tomó importancia a tal grado que los robots fueron accesibles para las universidades alrededor del mundo, esto permitió desarrollar proyectos para solucionar algún problema como caminar o mover objetos pesados entre otros, Ichikawa and Ozaki (1985).

Con el paso del tiempo la robótica se volvio un campo de estudio bastante complejo, el cual se tuvo que dividir en múltiples clasificaciones según su función, las cuales son:

- **Industrial:** Son aquellos robots dentro de las industrias que sirven para automatizar procesos.
- **De servicios:** Son aquellos robots que tienen como proposito facilitar tareas cotidianas como la limpieza o la investigación.
- **Militares:** Son aquellos robots que tienen como finalidad asistir las labores militares o proporcionar alguna ventaja respecto a los enemigos.
- **Médicos:** Son aquellos robots que sirven como apoyo para las tareas de los médicos como cirujias o analísis complejos de los pacientes, entre otras tareas.
- **Educativos:** Son aquellos robots que sirven como introducción a la robótica para personas que tienen poco conocimiento o nulo, permitiendo crear proyectos con facilidad.

Particularmente, dentro de la robótica de servicios podemos encontrar tres subcategorías, **robots domésticos**, **robots de investigación** y **robots de exploración**. Los robots de investigación se enfocan en dos áreas principales: robótica humanoide y robótica móvil. La robótica humanoide tiene como enfoque el imitar los movimientos y comportamientos de los seres humanos, mientras que la robótica móvil esta enfocada en imitar el comportamiento de animales e insectos, Ichikawa and Ozaki (1985).

La robótica móvil tiene tres enfoques según las características del robot: robótica con ruedas, con orugas y con patas. Analizar los robots móviles con ruedas representa una ventaja respecto a los robots con patas o con orugas, esto debido a que el analísis matemático del modelo del robot es sencillo de representar respecto a los otros. Específicamente los robots móviles omnidireccionales, los cuales pueden representarse con sus componentes de velocidad o posición en el eje coordenado, donde las velocidades del robot estan dadas directamente por la entrada de la ley de control. Por esta razón, comúnmente se pueden encontrar trabajos usando este tipo de robots en diversos campos de investigación de la robótica móvil, por ejemplo, la robótica colaborativa la cual tiene como objetivo principal abordar problemas complejos y difíciles de resover usando métodos clásicos de un solo robot, Siciliano and Khatib (2007).

De acuerdo a Bayindir (2016), una tarea de navegación colaborativa puede definirse como aquél escenario en el que un robot con las capacidades de detección y localización pueden alcanzar un objetivo en una ubicación desconocida con la ayuda de otros robots. Las tareas colaborativas se clasifican en 8 principales problemas, cada uno con enfoques y metodologías propias para cumplir con sus objetivos:

- Agregación autoorganizada (*Aggregation*), es la tarea de reunir un número de individuos autónomos en un lugar común, es un comportamiento básico ampliamente observado en la naturaleza con muchas especies animales. El trabajo de Gasparri et al. (2012), se propone un algoritmo de agregación de enjambre para sistemas multirobots basado en la interacción local, los robots mostraron un comportamiento cohesivo a través de las interacciones locales utilizando un sistema de control remoto basado en Kinect.

- Congregación (*Flocking*), es un comportamiento observado en la naturaleza en muchas especies de aves o mamíferos, que forman grandes grupos de

individuos que se mueven juntos hacia un ubicación objetivo común. En el trabajo de Kumar et al. (2012), se diseña un sistema basado en *flocking* utilizando múltiples robots para la exploración, el mapeo y la búsqueda adecuada para grandes grupos, que funcionan en un entorno desconocido y que carecen de un marco de comunicaciones de apoyo existente y utilizan reglas de dirección para que el sistema multirobot avance en forma de parvada.

- Búsqueda organizada (*Foraging*), es la tarea de búsqueda colectiva, inspirada en el comportamiento de colonias de hormigas, otro escenario comúnmente estudiado en robótica de enjambre. En el trabajo de Otte et al. (2013), se propone estudiar el problema de la navegación con búsqueda organizada, en el que un agente con una gama limitada de sensores debe simultáneamente: navegar hacia un objetivo global y cambiar de ruta a medida que se detectan oportunidades de búsqueda. Cada intento provoca una desviación de la ruta más corta a la ruta más larga, con consecuencias para la longitud de la ruta, la duración de la misión y el uso de combustible.

- Agrupación y clasificación de objetos (*Object clustering and sortering*), se refiere a una tarea donde los objetos dispersos en el entorno deben estar agrupados. En la tarea de agrupamiento de objetos no hay un destino predefinido ni lugar para objetos recolectados, el objetivo es colocar los objetos cerca el uno al otro mediante diversas clasificaciones. En el trabajo de Vardy et al. (2014), se presenta un nuevo método que permite a un enjambre de robots clasificar objetos arbitrariamente ordenados en *clusters* homogéneos. Cada robot mantiene un punto de memoria para cada tipo de objeto, al recoger un objeto vuelve para añadirlo al grupo que rodea el punto de almacenamiento. Otro ejemplo es el trabajo de Smirnova et al. (2015), el cuál describe una técnica de detección y descripción de puntos de referencia visuales naturales para la navegación cognitiva de robots móviles. Los puntos de referencia propuestos se construyen mediante el filtrado y la agrupación simultánea de un mapa de saliencia y un mapa de puntos que puede lograrse mediante un método visual SLAM(Simultaneous Localization And Mapping) como PTAM(Parallel Tracking And Mapping).

- Formación de patrones (*Path formation*), se refiere a un proceso donde los robots pueden construir colectivamente un patrón entre múltiples ubicaciones dentro del entorno, de modo que el tiempo necesario para llegar a una ubicación se minimiza. En el trabajo de Bae et al. (2019), se propo-

ne un algoritmo suave de planificación de trayectorias para múltiples robots que utiliza el aprendizaje profundo que combinado con el algoritmo CNN (Convolution Neural Network). Por otro lado, en el trabajo de Sperati et al. (2011) se estudia el problema de la exploración y la navegación en un entorno desconocido desde la perspectiva de la robótica evolutiva de enjambre. La estrategia colectiva se sintetiza mediante técnicas de robótica evolutiva y se basa en la aparición de una estructura dinámica formada por los robots que van y vienen entre las dos zonas objetivo. Gracias a esta estructura, cada robot es capaz de mantener el rumbo correcto y navegar eficazmente entre las dos zonas.

- Despliegue (*Deployment*), los robots deben desplegarse en un ambiente sin coordinación central. Esta tarea es empleada en el mapeo de entornos desconocidos para sistemas de vigilancia autónomos. En el trabajo de Alitappeh et al. (2017), se propone encontrar una distribución de un grupo de robots en un entorno, mejor conocido como despliegue, abordando un método de optimización multiobjetivo para desplegar/redistribuir robots en el entorno usando algoritmos evolutivos, teniendo en cuenta dos objetivos. El primer objetivo representa una buena estimación de las posiciones finales en las que se ubicarán los robots. El segundo objetivo es encontrar el camino más corto desde la ubicación inicial de los robots hasta estas posiciones.

- Manipulación colaborativa (*Collaborative manipulation*), los sistemas de enjambre permiten a los agentes ejecutar tareas más eficientemente que requiere cooperación entre múltiples individuos. En el trabajo de Bechlioulis and Kyriakopoulos (2018), se aborda el problema del transporte cooperativo de objetos en un espacio de trabajo restringido con obstáculos estáticos, en el que la coordinación se basa en la comunicación implícita establecida a través del objeto comúnmente agarrado. El trabajo de Alonso-Mora et al. (2015), explota la deformabilidad durante la manipulación de objetos blandos por parte de robots. Un enfoque híbrido centralizado/distribuido restringe la planificación centralizada a la guía global de alto nivel del objeto para el consenso.

- Cita organizada (*rendezvous*), se considera un grupo de robots móviles con ruedas, mediante sensado y restricciones de energía, tiene la tarea de reunirse en un destino común, Francis Manfredi Maggiore (2016).

Dentro de las tareas colaborativas existen dos enfoques centralizado y descentralizado, el centralizado se refiere a establecer todo el control dentro de una computadora central, la cual, le proporciona instrucciones a todos los robots involucrados. El enfoque descentralizado define a cada robot como independiente, y cuenta con su lazo de control interno, donde solo conoce las variables de los demas robots como su posición o velocidad. Una tarea colaborativa que que se puede analizar mediantes métodos descentralizados es la tarea de *rendezvous*, algunos trabajos que abordan este concepto son Friudenberg and Koziol (2018); Kunwar and Benhabib (2006); Luo et al. (2019) y en el libro de Francis Manfredi Maggiore (2016) *Flocking and Rendezvous in Distributed Robotics*. Al momento de analizar el problema *rendezvous*, resulta que conforme el número de robots incrementa, generar una ley de control para forzar a todos los robots a llegar en el tiempo deseado de forma individual se vuelve complicado. Esto genera que al momento de plantear la mejor estrategia para resolver la tarea de navegación *rendezvous* se comience por acotar el tiempo en el que tienen que llegar los robots a la meta. Esto permite que se planteen nuevos métodos para analizar, y generar una ley de control que abarque la mayor cantidad de escenarios posibles y así generar una solución que no sea única a ese sistema.

Para analizar la problemática de *rendezvous* y cumplir con el objetivo de tiempo definido para este proyecto, se toman en cuenta los trabajos de navegación autónoma en robots móviles de los autores; Anggraeni et al. (2019); Colunga et al. (2020); Becerra et al. (2016); Chang et al. (2021); Shang and Huang (2020); Li et al. (2021); Zhang et al. (2020); Zhao et al. (2021). Estos artículos destacan en la resolución de problemas tiempo específicos en tareas de navegación autónoma para resolver problemas específicos para sistemas de un solo robot como múltiples de ellos.

Flocking y *Rendezvous* son métodos distribuidos dentro de la robótica, cuentan con un número idéntico de robots móviles con únicamente sensores a bordo para moverse a un área en común usando control distribuido, también es un problema llamado **reunión** o **encuentro**.

La metodología de comportamientos analíticos es aplicada en Clemente et al. (2018); Peñaloza-Mejía et al. (2019); Meza-Sánchez et al. (2019), para la síntesis automatizada de leyes de control como expresiones matemáticas explícitas. Se toma la idea de construir comportamientos en orden de modelar y analizar el

comportamiento completo de un sistema dinámico, como en los sistemas basados en comportamientos presente en Matarić and Michaud (2008).

De cualquier manera, en lugar de considerar un conjunto de comportamientos básicos como instrucciones que construyen los comporamientos del robot, se usan funciones matemáticas y variables de la formulación del problema de control para construir leyes de control no lineales. Los comportamientos analíticos integran la Teoría de Control(TC) con un proceso de optimización basado en el paradigma de Programación Genética(GP).

La TC es usada para formular el problema de control y establecer la función de aptitud que guiará la búsqueda optimizada de controladores. La aproximación mediante GP es usada para automatizar la búsqueda de controladores no lineales usando la diámica del sistema y el diseño de la función de aptitud.

En este proyecto de tesis, se propone una metodología para la síntesis de controladores no lineales que resuelva el problema de *rendezvous* con evasión de obstáculos integrando un objetivo de llegada en tiempo. La propuesta de comportamientos analíticos es teóricamente extendida para desarrollar un marco general, para la síntesis automatizada de leyes de control óptimas, que resuelva el problema descrito en un esquema distribuido.

Las virtudes de la metodología propuesta son ejemplificadas en la configuración de dos robots móviles omnidireccionales. El problema *rendezvous* es configurado con la construcción de un área designada mediante poligonos inscritos que definen las posiciones deseadas. Un conjunto de n controladores óptimos ha sido encontrados que complé con los criterios de este proyecto. Los resultados de simulación son presentados para validar el desempeño de cinco de los mejores controladores donde incrementar el número de robots es también considerado.

1.1 Planteamiento del problema

En este trabajo se propone una metodología de diseño automático de controladores para un sistema de robots omnidireccionales que resuelven la tarea de *rendezvous* considerando las siguientes restricciones y características:

- Se consideran al menos dos robots en un escenario libre de obstáculos.

- Se diseña para cada robot un controlador independiente, pero se conoce las variables de posición y velocidad del otro robot.
- Cada robot independientemente de su posición inicial debe llegar a una meta definada en un tiempo dado.
- Los robots deben de evitar una colisión entre sí.
- Cada robot tiene dimensión definida por un radio constante.
- Se define un procedimiento que define la posición organizada dentro de una zona en la que los robots deben reunirse.
- Se implementa la metodología de control por comportamientos analíticos para la construcción controladores de navegación que resuelvan el problema definido.
- Se establece una tolerencia de tiempo para la navegación de los dos robots.
- Se validan los resultados a partir de simulaciones numéricas con escenarios diferentes a los utilizados para la construcción de los controladores.

1.2 Objetivos

1.2.1 *Objetivo general*

Desarrollar una metodología para la síntesis automatizada de controladores analíticos que ayude resolver el problema colaborativo de *rendezvous* en un sistema multirobot que contemple una restricción de reunión con un tiempo deseado.

1.2.2 *Objetivos específicos*

- Diseño de una metodología para la síntesis de controladores que resuelva tareas de navegación colaborativa en sistemas multi-robot, haciendo uso del paradigma de comportamientos analíticos.

- Diseño de una función de desempeño que pérmita satisfacer los objetivos de navegación autonóma, evasión de obstáculos y tiempo de navegación.
- Validación númerica del cumplimiento de los objetivos de control para los controladores obtenidos mediante la metodología de comportamiento analíticos.
- Simulación del comportamiento del sistema multi-robot, haciendo uso del controlador con mejor desempeño, arrojado por el paradigma de comportamientos analíticos.

1.3 Antecedentes

Siendo la metodología seleccionada el control por comportamientos analíticos, la cual es la fusión de teoría de control y computo evolutivo. Además el control por comportamientos analíticos parte de su analísis se basa en campos de velocidad, lo cual hace uso de funciones atractivas y repulsivas.

Un ejemplo de la aplicación de la metodología de comportamientos analíticos que incorpora campos de velocidad para una tarea de navegación es el trabajo de Villalvazo-Covián et al. (2021), donde se usan los campos de velocidad en sistemas de evasión de obstáculos dinámicos. Cabe resaltar que los campos de velocidad en sistemas dinámicos son difíciles de representar, debido a que el campo de velocidad cambia conforme el obstáculo se mueve a la vez que el robot. El trabajo de Villalvazo-Covián et al. (2021), parte como continuación de los trabajos de Clemente et al. (2018); Meza-Sánchez et al. (2019), los cuales presentan un enfoque para la sintesis automatizada de un conjunto de comportamientos que cumplan con ciertos objetivos de control.

Esta tesis tiene como base el trabajo propuesto por Clemente et al. (2018), con el cual se busca ampliar el alcance de la metodología de control por comportamientos analíticos aplicado hacia la resolución de tareas de multirobot. Entonces se vuelve indispensable generar una metodología que permita la búsqueda y optimización autónoma de controladores de navegación enfocada a conjuntos mayores a 2 robots con evasión de obstáculos y con restricciones de movimiento. Abordando la problematica de *rendezvous* dentro de las tareas de navegación colaborativa para automatizar la busqueda de comportamientos que brinden una solución hacia el problema de navegación de tipo *rendezvous* con capacidad de

evadir obstáculos dinámicos dentro de una restricción de tiempo deseado para un sistema multirobot.

El enfoque analítico basado en comportamientos analíticos fue propuesto y definido por los trabajos de Meza-Sánchez et al. (2019); Meza-Sánchez et al. (2019); Clemente et al. (2018); Villalvazo-Covián et al. (2021), en donde se han abordado diversos problemas de control, los cuales sientan las bases para el desarrollo de este proyecto. La metodología de comportamientos analíticos, se describe en Meza-Sánchez et al. (2019), donde se establecen tres características principales que lo distinguen de las metodologías clásicas de Control por comportamientos (*Behavior-based control*). Primero, define los comportamientos base como funciones analíticas, que también pueden ser no lineales; en segundo lugar, utiliza un modelo del sistema para incluir la dinámica interna; y tercero, integra un controlador basado en Teoría de control con un proceso de aprendizaje que aplica Programación Genética (GP, por sus siglas en inglés) para generar comportamientos aprendidos que resuelven el problema a abordar. Un aspecto importante de los controladores construidos con esta metodología es que pueden analizarse en términos del marco de la teoría de control, ya que son expresiones matemáticas. De esta forma se puede garantizar el rendimiento del sistema a través del concepto de la Teoría de Control denominado *estabilidad en los puntos de equilibrio*. Un punto de equilibrio es una coordenada ubicada en el espacio de estados, tal que, cada vez que el robot móvil comienza en él, permanecerá en ese punto durante todo el tiempo futuro. La estabilidad de un sistema se demuestra a través de un analísis matemático de la convergencia del sistema hacia la condición deseada. El criterio de estabilidad más importante dentro del enfoque de Teoría de Control es el teorema de estabilidad de Lyapunov.

En el trabajo propuesto por Clemente et al. (2018), se propone una metodología analítica para desarrollar controladores que sean capaces de evadir obstáculos con velocidad limitada para el problema de control de posición en robots móviles omnidireccionales. La metodología propuesta aprovecha el enfoque de campos de velocidad para derivar un comportamiento forzado, y una búsqueda evolutiva de comportamientos adaptativos como funciones de atracción y repulsión. En Peñaloza-Mejía et al. (2017), esta metodología se extendió para resolver el problema de seguimiento de trayectoria en un sistema de doble integrador; y en Peñaloza-Mejía et al. (2019) para sistemas dinámicos de segundo orden. Se propone un controlador Proporcional-Diferencial tradicional con dinámica inversa para generar el comportamiento *forzado*, mientras que los

comportamientos *aprendidos* se buscan mediante un enfoque evolutivo para una variable de flujo acotado.

La integración del método teoría de control con el enfoque de GP permite la implementación de una etapa de aprendizaje que busca el cumplimiento de características adicionales en el comportamiento del robot. La técnica de GP permite la construcción de un árbol sintáctico para representar una solución dada en forma de controladores no lineales; tales soluciones están compuestas de operadores matemáticos y funciones analíticas. Siendo la programación genética una técnica para resolver problemas autómaticamente, sin la necesidad de tener que explicar cómo hacerlo, es un conjunto amplio de técnicas de la computación evolutiva. La programación genética generalmente se representa mediante árboles sintácticos, lo cual facilita la interpretación de los resultados, siendo el modelo más usado, (Langdon and Poli (2002)).

En este trabajo, se busca brindar una nueva metodología para la búsqueda automatizada de controladores para sistemas multi-robot, como una extensión de los trabajos de control por comportamientos analíticos. Partiendo de un analísis basado en campos de velocidad y en un controlador parcial predefinido, obtenido de los trabajos de Clemente et al. (2018); Meza-Sánchez et al. (2019). Para así, poder sintetizar controladores, con la capacidad de cumplir con objetivos predefinidos y además conservar las características originales del sistema.

1.4 Organización de la tesis

Esta tesis se organiza de la siguiente forma: el capítulo(2) se describen conceptos preliminares para desarrollar la metodología de control por comportamientos analíticos aplicados a sistemas multirobot. También se realiza una revisión del estado del arte sobre tareas de navegación de tipo rendezvous y el control por comportamientos analíticos.

En el capítulo(3) se describe la metodología usada para resolver la tarea de navegación de tipo rendezvous con un enfoque multirobot. Además, se analiza el trabajo que dio apertura a la realización de esta tesis, resaltando los aspectos mas relevantes.

En el capítulo(4)se analizan los resultados obtenidos para las características propuestas en los capitulos anteriores, así como las estadísticas obtenidas del proceso de simulación.

Finalmente, en el capítulo(5) se exponen las conclusiones y se define el trabajo futuro que se genera con la terminación de este proyecto.

2

Preliminares

En este capítulo se establece el marco teórico que sustenta el problema abordado por este proyecto de tesis. Se detallan los temas de tareas de navegación autonóma en robótica colaborativa, particularmente el problema *rendezvous*. Se establecen los aspectos de diseño de controladores aplicando la metodología de los comportamientos analíticos, y se diferencia la extensión propuesta en este trabajo para abordar sistemas multi-robot. También se definen los campos de velocidad que describen las trayectorias de los robots dentro de un marco referencial y la teoría que fundamenta los procesos evolutivos implementados con Programación Genética.

2.1 Tareas de navegación autonóma en sistemas multirobot

La robótica colaborativa ha sido estudiada durante años, tratando de aportar soluciones para convertir una tarea que puede ser difícil de realizar para un solo robot en algo sencillo usando múltiples de ellos. En particular, un objetivo de la robótica móvil es plantear un sistema en el cual, un robot móvil es encargado de navegar y realizar una tarea de forma autónoma. Por otro lado, la robótica colaborativa permite extender este objetivo a sistemas multirobot.

En Nedjah and Silva (2019), se presenta una definición generalizada de lo que es un sistema multirobot, abordando características de la robótica colaborativa. En su trabajo titulado *Review of methodologies and tasks in swarm robotics towards standardization*, menciona que los sistemas multirobot (MRS) surgieron como una extensión de la investigación sobre robots móviles. Donde, la idea principal de los sistemas multirobot es que muchos robots pueden cooperar para completar una tarea determinada más rápidamente que un solo robot móvil,

actuando en diferentes lugares al mismo tiempo. Estos sistemas no sólo explotan un paralelismo intrínseco, sino que también pueden ser tolerantes a los fallos, gracias a la redundancia de los robots. Algunos ejemplos son los siguientes:

En Kim et al. (2013), se explica el problema de planificación de movimiento en robots móviles, el cual, consiste en determinar una trayectoria para un robot móvil desde su posición inicial hasta su posición final a través de un espacio de trabajo con diversos obstáculos. Entonces, considerando obstáculos dinámicos, estáticos o ambos, se define una trayectoria óptima sin colisiones entre el o los robots y los obstáculos. Si sólo existe un robot en un entorno con obstáculos estacionarios, el problema de planificación del movimiento se reduce a encontrar una trayectoria óptima sin colisiones. Sin embargo, si el entorno es complejo, es necesario considerar la tarea de forma colaborativa y cooperativa utilizando un equipo de robots.

Algunos trabajos donde proponen sistemas autónomos capaces de transportar cargas a traves de un zona con multiples objetos con la mínima interacción humana, como se muestra en la Figura 2.1a. En Siciliano and Khatib (2007), se habla acerca de los trabajos más destacados en la industria, donde se abordan las principales tareas que cada robot, ya sea manipulador o robot móvil, debe desempeñar. Algunas de estas tareas son soldadura, ensamble de automoviles, pintura, automatización de transferencias de materiales, mecanizado, solo por mencionar algunos ejemplos de aplicaciones. Sin embargo, algunas otras tareas requieren mayor organización, por ejemplo, organizar y distribuir un conjunto de objetos, según su color o su forma, como se menciona en Sahare and Sahare (2010).

La tarea de mover objetos haciendo uso de sistemas multi-robot, es abordada en Hawley and Suleiman (2019). Ahí, se considera una metodología para optimizar el transporte de objetos de diversos tamaños usando dos diferentes métodos para la manipulación del objeto; empuje(“push only”) y arrastre(“grasping”), las cuales son usadas para abordar diferentes escenarios. Cabe destacar que en estos trabajos hay una pieza clave que los relaciona, y es, la navegación autónoma. Entonces, teniendo en cuenta que el problema que se quiere abordar es de navegación autónoma, se comienza a analizar la posibilidad de ejercer un enfoque a sistemas multirobot, sin embargo, antes que nada se debe definir que es un sistema multirobot.

Retomando el trabajo Nedjah and Silva (2019), se redefine el concepto de sistemas multirobot como robótica de enjambre, debido a la necesidad diseñar un

sistema con gran número de agentes relativamente sencillos y físicamente centralizados de manera que surja un comportamiento colectivo deseado a partir de las interacciones locales entre los agentes y el entorno (véase Fig. 2.1b). Además, Nedjah and Silva (2019) define cinco características principales de los sistemas de robótica de enjambre:

1. Un enjambre está compuesto por robots autónomos que pueden moverse físicamente en un entorno e interactuar con otros objetos físicos.

2. El control, que es distribuido en la mayoría de las aplicaciones, debe soportar la existencia de un gran número de robots.

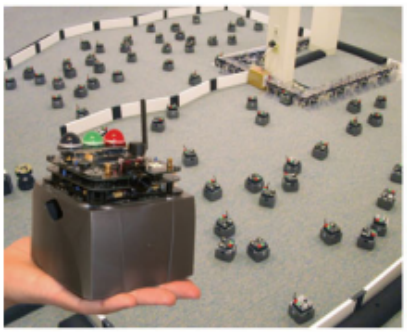
3. El enjambre es homogéneo o heterogéneo, es decir, está compuesto por varios grupos homogéneos.

4. Los robots individuales deben cooperar para llevar a cabo una tarea determinada.

5. Las capacidades de detección y comunicación son locales y, por tanto, los robots no tienen acceso a información global.

De las cinco características de los sistemas de enjambre, se pueden usar tres de ellas en el enfoque multirobot, siendo la primera, la cuarta y la quinta.

(a) Robot autónomo, configurado como montacargas, capaz de mover objetos pesados de una zona a otra, Sahni et al. (2019).



(b) Representación de un sistema multi-robot, Sahni et al. (2019).

Figura 2.1: Ejemplos de robots autónomos y un sistema multirobot

2.2 La tarea de navegación *Rendezvous* y su importancia en la robótica móvil

A menudo es necesario que un robot móvil intercepte y se reúna en un lugar específico, evadiendo obstáculos estáticos o en movimiento, con el fin de satisfacer objetivos de navegación. La tarea que se adaptá a dicha premisa es mediante *rendezvous* tambien llamado **encuentro**. La tarea de **encuentro** se define como la acción de coincidir la posición de uno o varios objetos en movimiento dentro de una zona delimitada, en un tiempo definido y con una velocidad determinada. Entiéndase, que dichos objetos puede llegar a ser cualquier cosa, incluido robots móviles.

El trabajo de Friudenberg and Koziol (2018), aborda el analísis del problema de **encuentro** dentro de las aplicaciones de robótica industrial, militar, topográfica y de reparto han sentado las bases para la investigación de máquinas autónomas, como los vehículos aéreos no tripulados (UAV), los vehículos terrestres autónomos (AGV) y los vehículos submarinos autónomos (AUV). Su trabajo aborda un método de guiado que combina los campos potenciales, típicamente utilizados para evitar obstáculos, y la navegación paralela, un popular método de guiado de misiles. Más aún, propone un nuevo algoritmo unificado que permite a un robot móvil interceptor se guíe hacia un objetivo en movimiento y se encuentre con él, evitando al mismo tiempo los obstáculos en su camino. Las simulaciones y el análisis muestran que el algoritmo combinado aumenta el rendimiento reduciendo el tiempo de contacto en la mayoría de los casos. En concreto, se ha comprobado que proporciona una mejora del 14.4-21.3 % en aproximadamente el 96-98 % de los casos de simulación.

Por otra parte, Luo et al. (2019) aborda el problema de control de múltiples robots sin perder la conectividad con la red, la cual incluye implicaciones como el control de formación, la asignación de tareas coordinadas y las misiones robóticas cooperativas. Sobre todo, presenta un algoritmo basado en la navegación sin coordenadas para permitir el encuentro de robots móviles distribuidos en cualquier nodo hacia el robot líder designado mediante el seguimiento jerárquico de la topología de la red inalámbrica. Se asume que el robot sólo puede detectar y comunicarse con sus vecinos (es decir, detección local).

El enfoque propuesto por Luo preserva la conectividad durante la tarea de **encuentro**, se adapta a los cambios dinámicos en la topología de la red (por ejemplo, la pérdida o recuperación de un enlace de comunicación) y es tolerante

a los fallos de movilidad de los robots. Se analiza teóricamente el algoritmo propuesto y se demostró experimentalmente el enfoque mediante simulaciones y amplios experimentos de campo. Los resultados indican que el método es eficaz en diversos escenarios realistas en los que los robots están distribuidos en un entorno desordenado.

Mientras que en el trabajo de Kunwar and Benhabib (2006), presenta un novedoso método de planificación de trayectorias para la intercepción autónoma de objetivos en movimiento en presencia de obstáculos dinámicos, es decir, la coincidencia de la posición y velocidad. El método de intercepción óptimo en el tiempo que se propone, es un algoritmo híbrido que aumenta una novedosa técnica de seguimiento en el problema de **encuentro** con el enfoque de velocidad-obstáculo, para la evasion de obstáculos. El propio algoritmo de evasión de obstáculos no podía utilizarse en su forma original y tuvo que ser modificado para garantizar que la trayectoria planificada se desviara mínimamente de la generada por el algoritmo. Se demostró que la eficacia temporal es tangible mediante el método de intercepción propuesto.

Existen muchas técnicas que permiten abordar las tareas de navegación colaborativa, siendo la más importante para este proyecto la metodología de comportamientos analíticos, la cual tiene un principio basado en algoritmos evolutivos, teoría de control y campos de velocidad, la cual se aborda a continuación.

Figura 2.2: Ejemplo de un sistema multirobot cumpliendo una tarea colaborativa de **encuentro**, Luo et al. (2019)

2.3 Comportamientos analíticos.

Un comportamiento puede definirse como una acción independiente que resulta de la interacción directa del sistema con su entorno. Este concepto se introdujo para representar la inteligencia en sistemas artificiales, Brooks (1991). Además, un sistema puede mostrar comportamientos complejos cuando dichos productores de actividad se entrelazan y ejecutan en paralelo.

Los trabajos de Mataric (1994); Matarić (1995) y los desarrollos propuestos en Arkin (1998), llevaron a la evolución del concepto de conductas hacia un enfoque llamado control basado en el comportamiento. Este enfoque apunta a resolver problemas de control dentro del campo de la robótica; propone el desarrollo de un proceso donde se combinan un conjunto de acciones o módulos, denominados comportamientos base, para lograr características deseadas en el sistema.

Este método fue desarrollado originalmente para robots que necesitan adaptarse a la dinámica de los entornos del mundo real sin tener en cuenta dos cosas, (a) la dinámica interna del sistema, (b) ni las representaciones abstractas del mundo que lo rodea, Matarić and Michaud (2008).

El diseño basado en comportamientos analíticos fue desarrollado por primera vez en Clemente et al. (2018). En esta metodología, el cual nombra como **Sistema de Comportamiento Natural** denotando todo el sistema, esto incluye las dinámicas no modeladas, los parámetros desconocidos del sistema, tres comportamientos *base* de la metodología y la influencia en el entorno. El control por comportamientos analíticos propuesto por Clemente et al. (2018) se diferencia del enfoque clásico de Matarić and Michaud (2008) mediante tres puntos:

- Primero, se definen los comportamientos base como funciones analíticas, que pueden ser no lineales.
- Segundo, utiliza un modelo del sistema para incluir la dinámica interna.
- Tercero, integra un controlador basado en teoría de control con un proceso de aprendizaje que aplica Programación Genética (GP por sus siglas en inglés) para generar los comportamientos aprendidos.

En particular, la definición de la metodología de control por comportamientos analíticos parte del desarrollo de problemas de control y relaciona el comportamiento de sistemas artificiales que gobiernan dentro del aprendizaje en humanos. Esto apunta a automatizar y sintetizar el proceso de adquirir nuevos comportamientos y/o modificar los ya existentes. De esta manera, el comportamiento natural del sistema se puede definir mediante tres comportamientos base:

- **No Forzado**: Es el sistema en sí, sin ninguna modificación, ni lazos de control.
- **Forzado**: Es el sistema con un lazo de control para el cumplimiento de algún objetivo predefinido.
- **Aprendido**: Es el comportamiento proporcionado mediante un enfoque evolutivo para cumplir con objetivos de navegación.

Además, dentro del comportamiento natural se pueden añadir nuevos comportamientos no considerados por ejemplo, perturbaciones externas. Por otro lado, si se quiere aplicar los conceptos propuestos por Clemente et al. (2018) al problema de encuentro, se debe tomar encuentra los parámetros y modelos del sistema para dicho problema. De esta manera, obtenemos el diagrama representativo del sistema natural como se aprecia en la figura(2.3).

Dentro del control por comportamientos analíticos, la teoría de control juega un papel importante para validar que el comportamiento aprendido cumpla con la tarea deseada, para esto se deben cumplir tres tareas, las cuales son las siguientes:

- El análisis de los controladores a través del concepto de estabilidad de los puntos de equilibrio.
- La aplicación del criterio de estabilidad basado en el teorema de estabilidad de Lyapunov para la mayoría de controladores encontrados.
- Asegurar que los robots involucrados siempre convergen a la posición deseada, debido a los criterios de estabilidad.

Por último, la optimización por medio de Programación Genética debe de considerar las restricciones del sistema así como los objetivos de control para

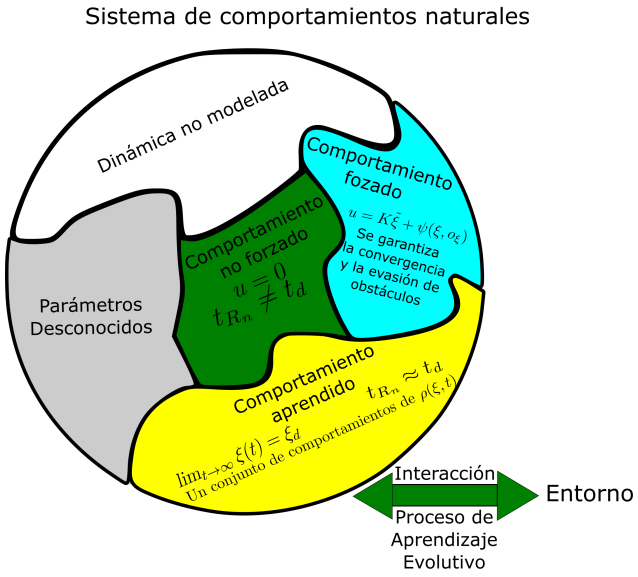


Figura 2.3: Diagrama del sistema natural para el problema de **encuentro**.

obtener funciones que puedan cumplir con la tarea deseada. A continuación se darán algunos conceptos básicos de la Programación Genética.

2.4 Cómputo Evolutivo y Programación Genética

El cómputo evolutivo es una rama de la inteligencia artificial, y es aplicado en diferentes áreas de la ciencia y la ingeniería. Conformado por tres tipos de algoritmos: Estrategias Evolutivas (ES por sus siglas en inglés), Programación Evolutiva (EP por sus siglas en inglés) y Algoritmos Genéticos (GA por sus siglas en inglés), son consistentes con el objetivo de utilizar el mecanismo de la evolución biológica, para mejorar la capacidad de usar computadoras para resolver problemas.

Se enfatiza el cambio de comportamiento en el nivel individual, EP enfatiza el cambio de comportamiento en el nivel de la población, y GA enfatiza la operación del cromosoma. Además, el trabajo de Ding et al. (2013) menciona la representación del individuo puede variar según los diferentes algoritmos existentes siguientes; Programación Genética, Algoritmo Memético, Programación Genética Cartesiana, entre otros. En particular, en el control por comportamien-

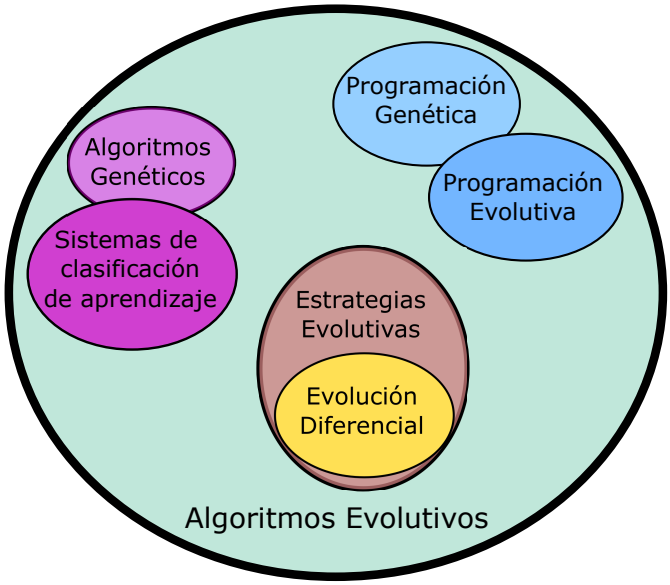


Figura 2.4: Representación de los algoritmos evolutivos

tos analíticos se utiliza una especialidad del cómputo evolutivo, que es la Programación Genética.

2.4.1 *Programación Genética*

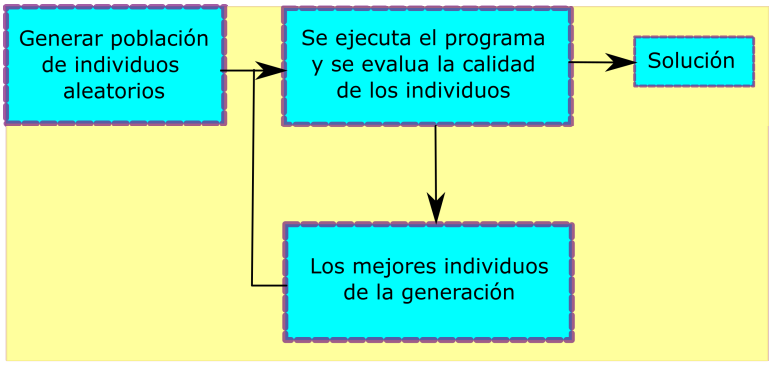


Figura 2.5: Ciclo evolutivo básico para la Programación Genética

La Programación Genética originalmente fue propuesta por Koza (1992). En este paradigma se busca una metodología en la cual la computadora sea capaz

de programarse así misma, siguiendo los procesos de la evolución artificial, ver Figura 2.5. En este sentido, la evolución se realiza sobre programas de computadoras o en su defecto, sobre abstracciones de los mismos, como lo son las expresiones matemáticas.

2.4.1.1 *Bases de la programación genética*

1	Aleatoriamente se crea una población inicial de individuos primitivos.
2	**Repetir**
3	Ejecuta cada programa y se evalua su desempeño.
4	Selecciona uno o dos individuos de la población con una probabilidad basada en si desempeño para participar en los operadores genéticos.
5	Crea nuevos individuos para aplicar operaciones genéticas con probabilidades específicas.
6	**Hasta** que encuentre una solución aceptable o cumpla alguna condición de paro.
7	**Regresa** el mejor individuo de toda las poblaciones.

Tabla 2.1: Algoritmo de Programación Genética

En la naturaleza, los individuos de una población compiten constantemente con otros por recursos tales como comida, agua y refugio. Los individuos que tienen más éxito en la lucha por los recursos tienen mayores probabilidades de sobrevivir y generalmente una descendencia mayor. Al contrario, los individuos peor adaptados tienen un menor número de descendientes, o incluso ninguno. Esto implica que los genes de los individuos mejor adaptados se propagarán a un número cada vez mayor de individuos de las sucesivas generaciones. De igual manera, en Programación Genética, se realiza una analogía a este proceso. Los individuos de la población que compite, son representados por programas, que son evaluados para medir su ajuste a la solución de un problema dado. El objetivo es conservar las características principales de la mejor solución para que al final de varias generaciones, se cumplan con los objetivos planteados, este proceso se ilustra en la tabla 2.1.

Como en la evolución natural de las especies, la programación genética podría considerarse un proceso en el cuál se desarrolla la solución de un problema que puede ser difícil de resolver por métodos tradicionales, este proceso

evolutivo artificial consiste en el aprendizaje y adaptación al medio en el que se encuentra, optimizando la aptitud de la especie. En este sentido, es posible imitar el punto de vista de la genética, es decir, "la supervivencia del más apto".

Yu and Gen (2010) define el diseño de la optimización o aprendizaje de los algoritmos evolutivos que realizan tareas con la capacidad de adaptación mediante tres características principales:

- *Población:* los algoritmos evolutivos mantienen un grupo de soluciones, llamada población, para optimizar o aprender el problema de una manera paralela. La población es un principio básico del proceso evolutivo.
- *Aptitud:* Cada solución en una población se llama individuo. El individuo tiene su representación génica, llamado código, y la evaluación del desempeño, llamado valor de la aptitud. Los algoritmos evolutivos prefieren individuos más aptos, que es la fundación de la optimización y convergencia de este tipo algoritmos.
- *Operadores:* los individuos se someterán a una serie de operaciones de imitar los cambios genéticos, lo que es fundamental para buscar la solución en espacio.

2.4.1.2 *Representación*

Dentro de la programación genética la manera en que se representa una solución encontrada por la evolución, es mediante árboles sintácticos, sin embargo, existen diferentes formas de representar el arbol resultante e incluso sus partes internas mediante algunos conceptos, los cuales son:

- *Cromosoma:* El cromosoma es una parte de la solución que proporciona algúna característica al sistema general, está representado como un árbol sintáctico, veasé figura(2.6).
- *Terminal:* Es aquella sección externa de un arbol sintáctico que puede ser cualquier variable o literal que forma parte del problema, veasé figura(2.7).
- *Fenotipo:* Se denomina de esta manera a la solución general, despues de incorporar el o los cromosomas encontrados en el algoritmo evolutivo, suele representarse como arbol sintáctico.

- *Función:* Es aquella sección interna de un cromosoma que proporciona una caracteristica unica y esta compuesta por funciones aritmeticas, trigonometricas, entre otras, veasé figura(2.7).
- *Conjunto Primitivo:* La configuración de funciones y terminales en una forma conjunta, representa el primer conjunto de resultados que es denominado primitivo.
- *Descendencia:* Son los hijos (resultados), encontrados por el algoritmo evolutivo que tienen las mejores características, y darán paso a la nueva población, hasta que se cumpla la condición de paro.

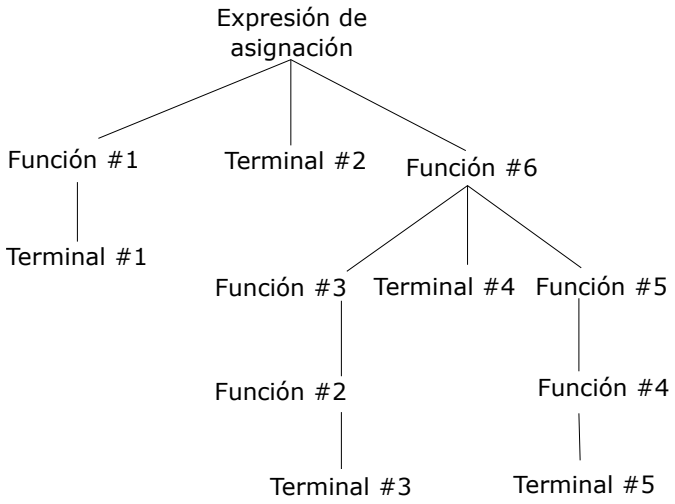


Figura 2.6: Representación de un cromosoma en Programación Genética

2.4.1.3 *Inicialización*

En la Programación Genética se realiza un método de selección probabilistico para generar los primeros individuos de una población, los métodos principales son; *Ramped Half and half*, *Grown* y *Full* por su traducción en ingles. En particular, cada método se refiere a como se genera cada árbol sintáctico (individuo) dentro de la población inicial. Los tres métodos para generar una población inicial mejor conocidos son los siguientes:

- Crecimiento (Grow): Profundidad variable para las ramas.

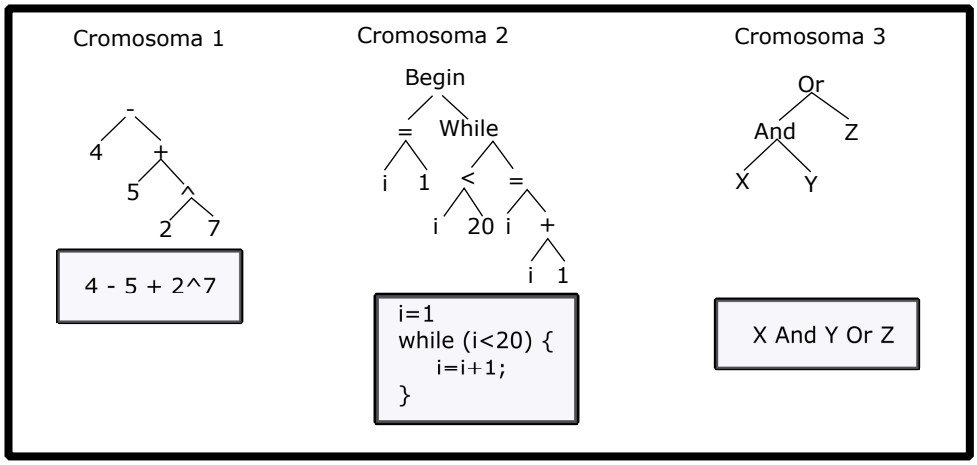


Figura 2.7: Representación de algunas funciones(operaciones aritmeticas, logicas, etc.) y terminales(números, variables, letras, etc.) en programación genética.

- Completo (Full): Misma profundidad para todas las ramas del árbol.
- Mitad y Mitad en Rampa (Ramped Half and Half):
  - Se generan árboles para cada posible profundidad.
  - El 50% será full y el 50% sera grow.

En la definición que propone Yu and Gen (2010), cada método de generación para la población incial opera con un determinado número de cromosomas o individuos que en conjunto conforman una población. En ocasiones es posible saber qué parte del dominio del problema contiene las mejores soluciones, pero en la mayoría de las situaciones, se inicializa con una búsqueda ciega sobre el espacio de la solución, mediante una población aleatoria. En particular, la razón para una inicialización aleatoria es que los algoritmos evolutivos son capaces de hacer una búsqueda global. Sin embargo, lo más simple es generar cada individuo con una distribución uniforme, aplicando operadores genéticos.

Finalmente, dentro de la Programación Genética existe un concepto denominado profundidad, se refiere al tamaño que puede alcanzar un árbol sintáctico, una profundidad menor indica un árbol pequeño, mientras que una profundidad mayor quiere decir que el árbol sintáctico sera grande.

2.4.1.4 *Evaluación*

La función de desempeño es una función de evaluación que devuelve una medida absoluta del rendimiento del individuo sobre el problema. Esto quiere decir, que es una función que mide el valor del individuo en relación con el resto de la población. La función de desempeño se puede elegir dependiendo de la tarea a resolver. Los individuos pueden enumerarse en orden de los valores más bajos a los más altos de la función de evaluación, y se puede aplicar una clasificación ordinal. La clave es que el desempeño de un individuo debe representar su valor en relación con el resto de la población, de modo que el mejor individuo representa la mejor solución, Thede (2004).

2.4.1.5 *Selección*

El objetivo de la selección es elegir dentro de la población los individuos que van a crear descendencian para la próxima generación, comúnmente conocido como piscina de apareamiento. El grupo de parejas seleccionadas participa en operaciones, haciendo avanzar a la población hacia la próxima generación acercando los valores de los genes a la solución óptima, Kumar (2012).

Cada individuo i cuenta con una aptitud f_i obtenida de la evaluación realizada. De acuerdo con la selección natural los individuos con mejores aptitudes tienen mayor oportunidad de ser seleccionados, por lo tanto definimos el valor relativo de la aptitud de cada individuo i dado por:

$$p_i = \frac{f_i}{\sum_{i=1}^{popsize} f_i} \tag{2.1}$$

Donde popsize es el tamaño de la población, es fácil comprender que $\sum_{i=1}^{popsize} p_i = 1$, así podemos establecer que el individuo con p_i más alto tiene mayor probabilidad de ser elegido. Un método común a utilizar en la selección es el método de la ruleta, en donde cada modo matemático de cada individuo significa un espacio en una ruleta, mientras mas p_i, contará con más espacio en la ruleta el individuo y por lo tanto más probabilidad de ser elegido.

Una vez establecido el espacio de cada individuo en la ruleta, se procede a generar un número aleatorio $u(0,1)$, el cual representa la flecha como es utilizada

en el juego de la ruleta. Esta flecha terminará en uno de los espacios, y hay más probabilidad de que termine en el más grande. Una vez que la flecha ha caído en un espacio se selecciona el individuo correspondiente $\sum_{i=1}^{k} p_i < rand < \sum_{i=1}^{k+1} p_i$ El proceso se repite hasta obtener el número requerido de parejas para los nuevos individuos, Yu and Gen (2010). En particular, existen diferentes métodos de selección cada uno con diferentes ventajas y desventajas entre cada uno, siendo los métodos de selección los siguientes:

- Ruleta o reproducción proporcional: El método de la ruleta supone que los genes son elegidos de acuerdo a sus posibilidades cuando son iguales a su valor de desempeño. Este proceso es un principio de elección similar a la rueda de la ruleta. En la rueda de la ruleta la posibilidad de elegir un sector es igual a la magnitud del angulo central de dicho sector. Similarmente, en los Algoritmos Evolutivos la población total se divide dentro de la rueda y cada parte indica un hijo. La proporción del desempeño de los hijos de todo los valores de desempeño de la población total determina la probabilidad de selección para los genes de la siguiente generación. La principal ventaja de usar la estrategia de la ruleta es que nunca desecha los genes dentro de la población y proporciona una oportunidad para todos los genes de ser elegidos. Una de las principales desventajas de usar el método de la ruleta consiste en que no se puede aplicar a problemas de minimización, debido a que en estos casos el algoritmo puede arrojar uno o mas genes apropiados, pero saturar la población completa evitando que se exploren otros genes mejor adaptados. Ahvanooey et al. (2019).

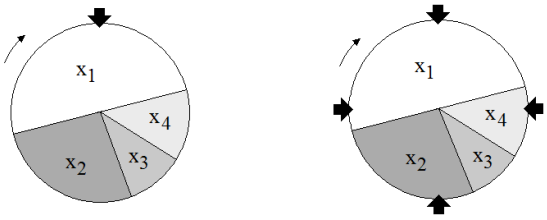


Figura 2.8: Método de selección de la ruleta, imagen obtenida de Méndez (2008)

- De torneo: es uno de los métodos de selección más significantes dentro de los algoritmos evolutivos debido a su gran efectividad y la facilidad de implementación que presenta en todas las plataformas existentes. La estrategia consiste en n genes existentes que son elegidos aleatoriamente dentro de una gran población, y los genes elegidos compiten uno contra

otro dependiendo del tamaño de torneo. El gen con mayor valor de desempeño es asignado como uno de la siguiente generación. La estrategia puede controlar la presión de selección facilmente alternando el tamaño del torneo si el tamaño es mas grande que los genes mas debiles. También prové una oportunidad para todos los genes de ser elegidos y retener la diversidad, sin embardo, al preservar la diversidad se reduce la velocidad de convergencia, Ahvanooey et al. (2019).

Figura 2.9: Método de torneo, imagen obtenida de Ayoub et al. (2020)

- Basada en clasificaciones: Dentro de esta estrategia, los genes son primero ordenados en base a sus valores de desempeño y despues la posición donde seran puestos. Los mejores genes se desplazan dentro del rango N y el peor obtiene el rango 1, de esta forma la selección de probabilidad es distribuida linealmente a los genes de acuerdo a su rango, Ahvanooey et al. (2019).

- Truncada: se usa para elegir soluciones potenciales por la recombinación de los genes despues del método de reproducción. En esta selección, los genes candidatos son ordenados por su valor de desempeño y con cierta proporción de los genes que son elegidos que mejor se ajustan y son reproducidos $\frac{1}{t}$ veces. La principal desventaja de este método de selección es que es menos sofisticado que los otros métodos de selección y es muy poco usado en la practica, Ahvanooey et al. (2019).

- Exponencial: Esta estrategia, es basada en el la estrategia truncada, donde se usa la estrategia de rango en una manera donde la probabilidad es calculada exponencialmete, la base del exponente es C, donde $0 < C < 1$, Ahvanooey et al. (2019).

- **Elitismo:** Es comunmente usado en la Programación Genética para asegurar que los mejores individuos encontrados en una población no se pierdan en el proceso y hace posible que esten disponibles para futuras mejoras a la nueva población. La técnica de elitismo se usa donde uno o mas individuos con el mas alto valor de desempeño son copiados, sin cambios, de

una generación a la siguiente. Una de las principales ventajas del elitismo es que forma un tipo de control de crecimiento en el árbol, evitando que se presente el problema de *Bloat* que resulta ser muy común dentro de los algoritmos evolutivos, Poli et al. (2008b).

Por otro lado, el *Bloat* es un problema de descontrol en el tamaño de un árbol sintáctico donde crece infinitamente, hasta que alcanza el tamaño maximo de memoria disponible por la variable asignada. Mediante el elitismo, este problema de crecimiento desmesurado tiene una posibilidad menor de aparición, debido a que se forza a tener un árbol de menor tamaño con mejor desempeño como individuo mejor adaptado para la siguiente generación. En el trabajo de Whigham and Dick (2010), se aborda el concepto de *Bloat* y diversas técnicas de elitismo usadas para el control de esta problematica. En particular, se abordan diversas maneras de control para *Bloat*, partiendo desde poner un limite a la profundidad del árbol, hasta forzar a los individuos de mayor tamaño a migrar a una población exclusiva para mantener el control de la población principal.

2.4.1.6 *Operaciones Genéticas: Cruzamiento y Mutación*

Los algoritmos de programación genética involucran dos tipos de operadores principales: cruce y mutación, según Melanie (1999); Poli et al. (2008a).

- *Cruce.* Este operador combina los genes de los individuos previamente seleccionados para crear dos individuos descendientes para la siguiente generación. El operador de cruce imita la recombinación biológica entre dos organismos, tal y como se muestra en la figura(2.10).

  Se seleccionan dos resultados con el mejor valor y se cruzan entre ellos generando un nuevo resultado (hijo), el cual es evaluado en el problema y si tiene una buena aptitud es utilizado en una nueva poblacion, de lo contrario es descartado.

- *Mutación.* Este operador cambia aleatoriamente algunos de los bits en un cromosoma. La mutación puede ocurrir en cada posición de bit en una cadena con alguna probabilidad, generalmente muy pequeña.

  Dado que la mutacion siempre tiene una probabilidad menor al operador cruce, se cambia una sección del resultado (alelo), modificando el resulta-

do y generando un nuevo individuo con una aptitud que podrá ser mejor o peor.

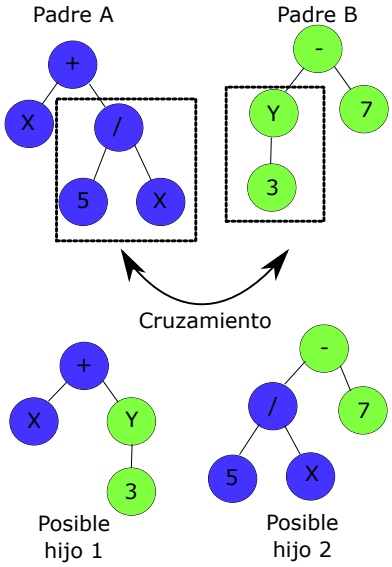


Figura 2.10: Cruza

Si el algoritmo de programación se ha implementado correctamente, la población evolucionará a lo largo de sucesivas generaciones. La convergencia es la progresión hacia la uniformidad creciente. Se dice que la población ha convergido cuando todos los genes han convergido al mejor resultado posible, Beasley et al. (1993).

2.4.1.7 *Criterio de paro.*

Es la condición para terminar la evolución que se debe cumplir cuando alguno de los siguientes requisistos se presenten:

- El algoritmo evolutivo encontró el individuo mejor adaptado de toda la población.
- El algoritmo evolutivo alcanzó el limite de generaciones definidas.
- El algoritmo evolutivo presentó el problema de “bloat” y la evolución se detuvo.

2.5 Campos de velocidad

Para navegación autonóma de robots móviles, el utilizar campos potenciales, es ampliamente usado debido a su sencillo procedimiento para el diseño de control.

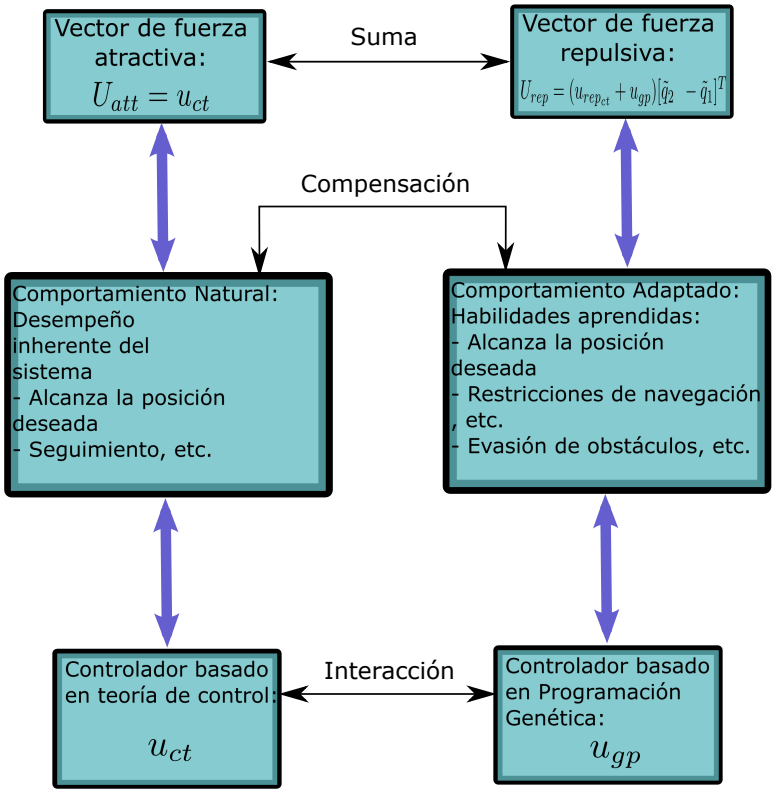


Figura 2.11: Integración conceptual de los campos vectoriales con control basado en comportamientos para desarrollar una familia de controladores.

Este acercamiento, concive la idea de combinar las fortalezas de dos campos de fuerza artificiales; uno es el campo de fuerza diseñado para cumplir condiciones de estabilidad para el sistema de contro, y el otro es la fuerza para crear una barrera potencial en cada punto de cada obstáculo. Matématicamnte, el acercamiento mediante campos potenciales propone la construcción de controladores de la forma:

$$U = U_{att} + U_{rep} \tag{2.2}$$

Donde la ley de control:

$$U \in \mathfrak{Re}^n \tag{2.3}$$

Es la suma del campo potencial atractivo $U_{att} \in \mathfrak{Re}^n$ para un comportamiento deseado con un campo potencial repulsivo $U_{rep} \in \mathfrak{Re}^n$ forzando un movimiento de repulsión de los obstáculos. Esta estrategia se puede expandir como la suma de campos potenciales repulsivos m acuerdo al número de obstáculos, por ejemplo:

$$U_{rep} = \sum_{i=1}^{m} U_{rep_i} \tag{2.4}$$

La principal desventaja de este acercamiento, es comunmente causado por la combinación de fuerzas repulsivas ejercidas, es la posibilidad de ser atrapado en un minimo local del campo propuesto, pueden ocurrir oscilaciones, y el robot no puede navegar a travez de obstáculos que esten muy cerca. Un analisis exahustivo de esas y otras limitaciones de la estrategía ha sido abordado en Clemente et al. (2018).

Dentro de los campos potenciales se modela al robot como una partícula, la cual es influida por dos fuerzas, una fuerza de atracción y una fuerza de repulsión. Estas fuerzas se pueden representar mediante un valle y una montaña, el valle ocasiona que el robot avance hacia el punto más bajo, si no existe ningún obstáculo este se deslizará en linea recta pues no existe fuerza que repela al robot. Sin embargo, si existiera uno o más obstáculos el robot se veria afectado por estos y la trayectoria ya no seria recta, sino que se veria influenciada por los obstáculos dentro del camino que recorreria el robot. En la representación de campos potenciales, el comportamiento de la trayectoria del robot va a estar dado por la suma de las fuerzas de repulsión y atracción, y el resultado final sera la fuerza resultante de dicha suma. Por otra parte, una forma de representar este método conservando sus características es mediante velocidades en lugar de fuerzas, a esto se le llama campos de velocidad. Al igual que con los campos potenciales, hacen uso de fuerzas de atracción y repulsión, la diferencia es que la resultante no se mide en newtons, sino que es la velocidad del robot. Para un robot móvil, se construye una función de potencial artificial de acuerdo con los obstáculos y un estado deseado del robot, y el robot se mueve a lo largo del vector gradiente, Urakubo (2018). Algunos de los trabajos que abordan los campos potenciales y los campos de velocidad aplicados a la robótica móvil son los siguientes:

En Urakubo (2018), se presenta un analísis teórico sobre la estabilidad de los sistemas no holonómicos cuando se les aplica un método de campo potencial, y muestra un ejemplo numérico para un vehículo submarino no holonómico entre obstáculos. Los sistemas no holonómicos con una función potencial tienen un número infinito de puntos de equilibrio, porque el movimiento de los sistemas no siempre puede generarse exactamente a lo largo del vector gradiente de la función potencial.

Por otro lado, Masoud (2013) proporciona una herramienta para realizar la planificación de la trayectoria a nivel de servo de un robot móvil. La capacidad de llevar a cabo, de una manera probadamente correcta, una tarea tan compleja a nivel de servo puede conducir a un gran aumento en la velocidad de operación, bajo consumo de energía y alta calidad de respuesta. La señal de velocidad de guiado de esta etapa suele convertirse en una señal de control mediante lo que se conoce como controlador electrónico de velocidad (ESC). Demuestra la capacidad del enfoque del campo potencial armónico (HPF) para generar una trayectoria y una señal de control probadamente correctas, restringidas y que se comportan bien para un robot rígido y no nómico en un entorno estacionario y desordenado.

Mientras que Mohammad et al. (2019), habla de como los temas relacionados con la robótica se han convertido en uno de los campos de investigación. Entretanto, los robots móviles inteligentes tienen una gran aceptación, pero el control y la navegación de estos dispositivos son muy difíciles, y la falta de trato con los obstáculos fijos y la evitación de los mismos, debido a un enrutamiento seguro, es el requisito básico de estos sistemas. En este trabajo se propone el método del campo de potencial artificial (APF) modificado para que el robot evite la colisión con los obstáculos fijos y alcance el objetivo en una trayectoria óptima; utilizando este algoritmo, el robot puede correr hacia el objetivo en entornos óptimos sin ningún problema al evitar los obstáculos, y también utilizando este algoritmo, a diferencia del algoritmo APF, el robot no se queda atascado en el mínimo local.

Como parte de la propuesta de Meza-Sánchez et al. (2019), podemos definir que la evasión dinámica de obstáculos es una de las tareas más difíciles en la navegación autónoma de los robots móviles. Puede definirse como una restricción de movimiento para el robot móvil mientras busca el cumplimiento de una posición o trayectoria deseada dentro de un plano coordinado. En particular, el desarrollo de controladores para robots móviles utilizando el enfoque de cam-

pos de potencial artificial es muy atractivo debido a su sencillo procedimiento de diseño. Este enfoque, se basa en la suma de dos controladores parciales. Una parte del controlador (llamada fuerza de atracción) tiene como objetivo que el robot móvil sea conducido a la posición o trayectoria deseada, la segunda parte del controlador (denominada fuerza de repulsión) intenta formar una barrera alrededor de los obstáculos para proporcionar una navegación segura. Las fuerzas repulsivas se implementan en cada celda ocupada para empujar al robot lejos de ella. El uso de funciones armónicas permitie evitar el problema de los mínimos locales; la integración del método del panel permite considerar obstáculos con forma arbitraria.

Finalmente, Muñoz-Vázquez et al. (2021) presenta un novedoso esquema de seguimiento de contornos basado en una representación cinemática bien planteada de robots móviles no holonómicos de conducción diferencial. En primer lugar, una agregación difusa de conjuntos espaciales en entornos desordenados permite diseñar un campo de velocidad para codificar el vector de velocidad deseado que apunta al objetivo (el contorno). Así, la trayectoria resultante evita los obstáculos mediante la combinación de campos de velocidad espacialmente distribuidos que permiten la navegación del robot de forma suave.

3

Metodología

En este capítulo se desarrolla la metodología utilizada en este proyecto, se describe el trabajo propuesto por Villalvazo-Covián et al. (2021), en donde se plantea la base del control por comportamientos analíticos aplicado a sistemas de un robot y uno obstáculo dinámico. Enseguida, se propone la extensión de dicho trabajo a sistemas multirobot enfocado al problema de **encuentro**. Para lograr los puntos anteriores, el problema de **encuentro** se restringe a un conjunto de escenarios y se define una métrica de rendimiento para esta propuesta. Finalmente se describe el uso de la herramienta computacional para el desarrollo y simulación de este trabajo.

3.1 Control de posición con evasión de obstáculos dinámicos en robots móviles omnidireccionales

La evasión de obstáculos dinámicos es una de las tareas más desafiantes en la navegación autónoma de robots móviles. Puede ser definida como una restricción de movimiento para el robot móvil mientras búsca cumplir con una posición deseada o trayectoria dentro de un plano coordenado. En particular, el desarrollo de controladores para robots móviles usando campos potenciales es llamativo, debido a su procedimiento sencillo de diseño. Este acercamiento, inicialmente fue propuesto por Khatib (1980) y Khatib (1985), consiste en la sumatoria de dos controladores parciales. La primer parte del controlador parcial (llamada fuerza atractiva) tiene el objetivo de conducir al robot móvil a una posición deseada. Mientras, que la segunda parte (se denota como fuerza repulsiva) intenta formar una barrera alrededor del obstáculo para proveer una navegación segura.

La idea fue aplicada por Borenstein and Koren (1989),usando una cuadrícula de celdas. Así, las fuerzas repulsivas son implementadas en cada celda ocupada por un obstáculo, y de esta forma forzar al robot a alejarse de él. La construcción de un campo potencial artificial usando funciones armónicas fue propuesto por Kim and Khosla (1992). Usar esta clase de funciones permitió evitar el problema del mínimo local, donde, se encuentra un punto de equilibrio suboptimo y el error del sistema no es cero. La integración del método de panel permite considerar obstáculos con figuras arbitrarias. Debido a que el método de panel, es básicamente una aproximación numérica que se basa en el uso de elementos discretos en la superficie de un objeto y luego prescribe un elemento de flujo (como un vórtice, un doblete, una fuente o un sumidero) en cada elemento que cumpla ciertas condiciones de contorno.

La aplicación de la técnica de modos deslizantes para el diseño de campos potenciales en robots móviles omnidireccionales fue presentado en Guldner and Utkin (1995). El uso de un método de campos potenciales, donde el objetivo y el obstáculo son dinámicos, fue propuesto en Ge and Cui (2002). Inspirado por el método de campos potenciales, el uso de módulos interconectados jerárquicamente que comandan la dirección del movimiento en un robot móvil fue presentado en Zavlangas and Tzafestas (2003). Como los módulos son acciones para el robot, donde dos controladores fuzzy dictan la dirección y velocidad del robot móvil. La regulación de una fuerza virtual relacionada con la distancia entre el robot y el obstáculo es hecha en Eun Soo Jang et al. (2005) usando un algoritmo de control de fuerza de impedancia. En Osorio-Comparán et al. (2016), fue presentada la combinación del campo potencial artificial con una evasión de mínimo local (LMA).

En Clemente et al. (2018), se introduce el paradigma de comportamientos analíticos para representar campos atractivos y de repulsión de velocidad en el espacio de trabajo de un robot móvil. La idea es diseñar un controlador que evite colisiones entre el robot y un obstáculo en movimiento. Basado en el trabajo de Clemente et al. (2018), un controlador proporcional tradicional ha sido usado como la fuerza atractiva hacia la posición deseada; y, el paradigma de programación genética ha sido implementado en la búsqueda de controladores parciales que evaden colisiones con obstáculos estáticos. Así, el objetivo es proponer un nuevo controlador parcial que permite la construcción de campos repulsivos de velocidad alrededor de un obstáculo circular en movimiento. De forma que la estabilidad asintótica global del robot móvil se conserve. Es decir, asegurar que el robot móvil siempre converge a la posición deseada.

El análisis de la dinámica en lazo cerrado del robot móvil controlado, y las propiedades de la función repulsiva, permite el cumplimiento simultáneo de los desafios de la evasión de osbtáculos. La función repulsiva propuesta aprovecha las propiedades del logaritmo natural, el cual, al ser la inversa de la función exponencial, exhibe un crecimiento al infinito conforme su argumento tiende a cero. Esta función es usada para relaciónar las distancias del robot móvil al obstáculo y la posición deseada. El controlador propuesto cuenta con un análisis de estabilidad mediante la teoría de Lyapunov, de esta forma, se garantiza el cumplimiento de los objetivos de control, además, el controlador esta conformado por dos partes principales. La primera parte, un controlador parcial de atracción tipo Proporcional-Diferencial, obtenido de la literatura. La segunda parte, un controlador parcial de repulsión, el cuál fue obtenido aplicando el paradigma de comportamientos analíticos. Así, este controlador de evasión, es aplicado en un escenario donde existe siempre un robot y al menos un obstáculo.

3.1.1 *Síntesis de control*

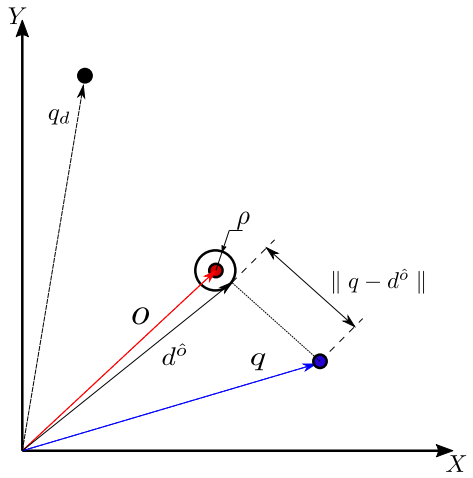


Figura 3.1: Representación del sistema de navegación de un robot móvil con evasión de obstáculos dinámicos

Para el diseño del controlador se utiliza la representación que se muestra en la figura(3.1), en un robot omnidireccional la dinámica es directamente la entrada de control, ver la ecuación (3.1). Donde, las variabes que conforman el controlador son: u es la entrada de control, $\dot{q}$ es la dinámica del sistema, $q(t)$ es

la posición del robot, q_d es la posición deseada, ρ es el radio del obstáculo y O es la posición del obstáculo.

El desarrollo de esta propuesta constá de dos objetivos de control, el primer objetivo de control esta dado por la ecuación(3.2), cuyo cumplimiento garantiza que el robot va a llegar a la posición deseada en tiempo infinito. En otras palabras la función atractiva debe ser asintótica y que garantice la convergencia a la meta.

$$\dot{q} = u. \tag{3.1}$$

$$\lim_{t\to\infty} q(t) = q_d \tag{3.2}$$

Mientras, que el segundo objetivo de control es la evasión de un obstáculo dinámico circular. Donde la evasión se define en la ecuación(3.3) cómo la distancia mayor al radio del obstáculo.

$$\| q - d^{\hat{o}} \| > \rho, \tag{3.3}$$

En terminos generales, la dinámica del robot móvil es la suma del controlador de atracción más el controlador de repulsión, formando parte de un controlador general, el cuál garantiza que el robot evadé siempre los obstáculos y siempre llega a la meta, salvo cuando el obstáculo esta en la misma posición que la meta.

3.1.1.1 *Controlador propuesto*

Se define el controlador de navegación cómo u, siendo $\tilde{q}$ el error de posición entre el robot y la posición deseada. El controlador de atracción esta compuesto como $K\tilde{q}$, donde K es una ganancia, de forma que $K > 0 \in \mathbf{R}^{2\times 2}$ es una matriz diagonal definida positiva.

$$u = K\tilde{q} + \psi(q, d^{\hat{o}}) \begin{bmatrix} \tilde{x}_2 \\ -\tilde{x}_1 \end{bmatrix} \tag{3.4}$$

El controlador dado en la ecuación(3.5) constá de dos variables q y $d^{\hat{o}}$, la variable q es la posición actual del robot, mientras que la variable $d^{\hat{o}}$ es posición de la frontera del obstáculo dinámico. En particular, la distancia entre el robot y la frontera del osbtáculo es de la forma $\| q - d^{\hat{o}} \|$. Por otro lado, el controlador de repulsión esta definido como se aprecia en la ecuación(3.5).

$$\psi(q, d^{i\hat{o}}) = \mathrm{Re}\left(\ln\left(\| q - d^{\hat{o}} \|\right)\right) \tag{3.5}$$

De este modo, la frontera del obstáculo esta difinida por la ecuación(3.6), con el obstáculo de radio ρ. Esta ecuación arroja un vector distancia entre el robot y la frontera del obstáculo el cuál depende de la posición actual del robot.

$$d^{\hat{o}} = \hat{o} + \frac{\rho\,(q - \hat{o})}{\|q - \hat{o}\|} \tag{3.6}$$

Finalmente, el vector de errores de posición está dado por la ecuación(3.7), que es el controlador de atracción sin el vector de ganancia K.

$$\tilde{q} = [\tilde{x}_1\ \tilde{x}_2]^T = [x_d - x \quad y_d - y]^T \tag{3.7}$$

3.1.1.2 *Análisis de convergencia*

Ahora, conviene analizar la convergencia del controlador parcial y así verificar que cumpla con los objetivos de control de evasión y navegación.

Teorema 1: Dada la dinámica en lazo cerrado definida por la ecuación(3.1) para la cinemática del robot móvil omnidireccional, basado en las ecuaciones (3.4) y (3.5), es global y asintóticamente estable.

Prueba: Aplicando la función clásica de Lyapunov, siendo una función cuadrática candidata $V(\tilde{q}) = \frac{1}{2}\tilde{q}^T\tilde{q} > 0$ al aplicar su derivada queda cómo; $\dot{V}(\tilde{q}) = -\tilde{q}^T K\tilde{q} < 0$. Donde $K > 0$ y es una matriz diagonal definida positiva.

La dinámica del sistema se representa mediante las ecuaciones(3.8-3.9), de esta manera, se puede analizar la estabilidad del sistema al aplicar el análisis de estabilidad de lyapunov y verificar si le función candidata es estable o no.

Dentro de estas expresiones matemáticas, k_{11} y k_{22} son ganancias constantes pertenecientes a la matriz diagonal definida positiva K

$$\dot{\tilde{x}}_1 = -k_{11}\tilde{x}_1 - \mathrm{Re}\left(\ln\left(\| q - d^{\hat{o}} \|\right)\right)\tilde{x}_2 \tag{3.8}$$

$$\dot{\tilde{x}}_2 = -k_{22}\tilde{x}_2 + \mathrm{Re}\left(\ln\left(\| q - d^{\hat{o}} \|\right)\right)\tilde{x}_1 \tag{3.9}$$

A partir de la función cuadrática candidata de Lyapunov presente en la ecuación(3.10), se procede a analizar la estabilidad del sistema.

$$V(\tilde{q}) = \frac{1}{2}\tilde{q}^T\tilde{q} \tag{3.10}$$

Al aplicar la derivada a la función $V(\tilde{q})$, nos queda la expresión de la forma:

$$\dot{V}(\tilde{q}) = \frac{1}{2}\left(\tilde{q}^T\dot{\tilde{q}} + \dot{\tilde{q}}^T\tilde{q}\right) \tag{3.11}$$

Tomando en cuenta que $\tilde{q}$ esta representada mediante la ecuación(3.12) y $\dot{\tilde{q}}$ es la dinámica del sistema representado en las ecuaciones(3.8 y 3.9), podemos sustituir las ecuaciones(3.8-3.9-3.12) en la expresión 3.11, obteniendo la ecuación(3.13).

$$\tilde{q} = [\tilde{x}_1\ \tilde{x}_2] \tag{3.12}$$

$$\dot{V}(\tilde{q}) = \frac{1}{2}\left([\tilde{x}_1\ \tilde{x}_2]\begin{bmatrix}\dot{\tilde{x}}_1\\ \dot{\tilde{x}}_2\end{bmatrix} + [\dot{\tilde{x}}_1\ \dot{\tilde{x}}_2]\begin{bmatrix}\tilde{x}_1\\ \tilde{x}_2\end{bmatrix}\right) \tag{3.13}$$

Resolvemos la ecuación(3.13) para obtener la expresión 3.14.

$$\dot{V}(\tilde{q}) = \frac{1}{2}\left(\tilde{x}_1\dot{\tilde{x}}_1 + \tilde{x}_2\dot{\tilde{x}}_2 + \tilde{x}_1\dot{\tilde{x}}_1 + \tilde{x}_2\dot{\tilde{x}}_2\right) \tag{3.14}$$

Simplíficamos la ecuación(3.14), para dar pasó a la ecuación(3.15).

$$\dot{V}(\tilde{q}) = \frac{1}{2}\left(2\left(\tilde{x}_1\dot{\tilde{x}}_1\right) + 2\left(\tilde{x}_2\dot{\tilde{x}}_2\right)\right) \tag{3.15}$$

Reducimos la expresión 3.15 para obtener la siguiente ecuación:

$$\dot{V}(\tilde{q}) = \tilde{x}_1\dot{\tilde{x}}_1 + \tilde{x}_2\dot{\tilde{x}}_2 \tag{3.16}$$

Ahora procedemos a sustituir la dinámica del sistema en la ecuación(3.16).

$$\dot{V}(\tilde{q}) = \tilde{x}_1\left(-k_{11}\tilde{x}_1 - \mathrm{Re}\left(\ln\left(\| q - d^{\hat{o}} \|\right)\right)\tilde{x}_2\right) + \tilde{x}_2\left(-k_{22}\tilde{x}_2 + \mathrm{Re}\left(\ln\left(\| q - d^{\hat{o}} \|\right)\right)\tilde{x}_1\right) \tag{3.17}$$

Resolvemos la ecuación(3.17):

$$\dot{V}(\tilde{q}) = -k_{11}\tilde{x}_1^2 - \mathrm{Re}\left(\ln\left(\| q - d^{\hat{o}} \|\right)\right)\tilde{x}_1\tilde{x}_2 - k_{22}\tilde{x}_2^2 + \mathrm{Re}\left(\ln\left(\| q - d^{\hat{o}} \|\right)\right)\tilde{x}_1\tilde{x}_2 \tag{3.18}$$

Simplíficamos la ecuación(3.18) y obtenemos la ecuación(3.19).

$$\dot{V}(\tilde{q}) = -k_{11}\tilde{x}_1^2 - k_{22}\tilde{x}_2^2 \tag{3.19}$$

La ecuación(3.18) es equivalente a tener la forma de la expresión 3.20

$$\dot{V}(\tilde{q}) = -\begin{bmatrix}\tilde{x}_1 & \tilde{x}_2\end{bmatrix}\left[\begin{array}{c|c} \tilde{k}_{11} & 0 \\ 0 & \tilde{k}_{22} \end{array}\right]\begin{bmatrix}\tilde{x}_1 \\ \tilde{x}_2\end{bmatrix} \tag{3.20}$$

Lo que se puede representar como:

$$\dot{V}(\tilde{q}) = -\tilde{q}^T K \tilde{q} \tag{3.21}$$

Demostrando así, que la función candidata de lyapunov cumple con las condiciones de la **prueba** para el Teorema 1. Como se puede denotar en la expresión(3.18), sin importar la función que componga el controlador de repulsión $\psi(q, d^{i\hat{o}})$, mediante lyapunov se asegura que el robot móvil siempre va a llegar a la meta, dicho de otra forma, el controlador de repulsión no afecta la convergencia del sistema.

3.1.1.3 *Análisis de repulsión*

Teorema 2: En relación con el controlador propuesto para el robot móvil omnidireccional abordado en las ecuaciones (3.4) y (3.5). Se evita la colisión con un obstáculo circular solido de radio ρ con la restricción que siempre la distancia entre el robot y la frontera del obstáculo es mayor al radio del obstáculo $\| q - d^{\hat{o}} \| > \rho$.

Prueba: De esta manera, como se establece en el teorema 1, la función de repulsión (3.5) no afecta la estabilidad del controlador 3.4. Es importante destacar que el controlador de repulsión se divide en tres casos de análisis, como se muestra en la figura(3.2).

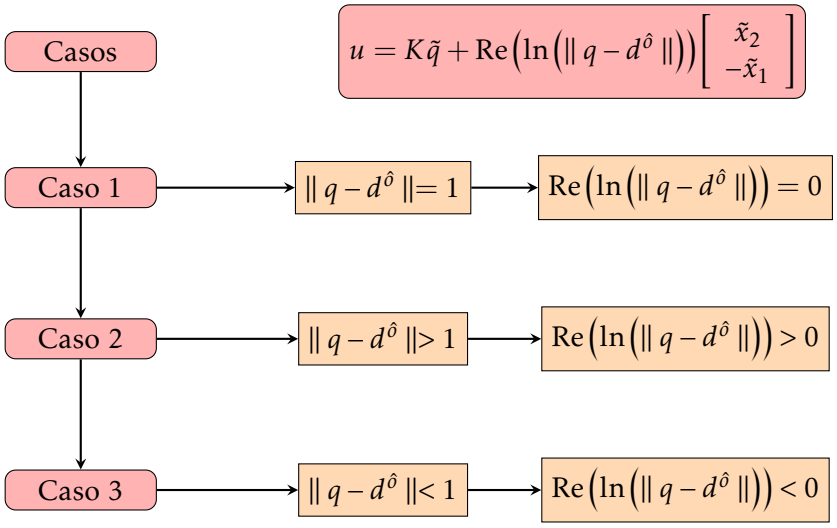
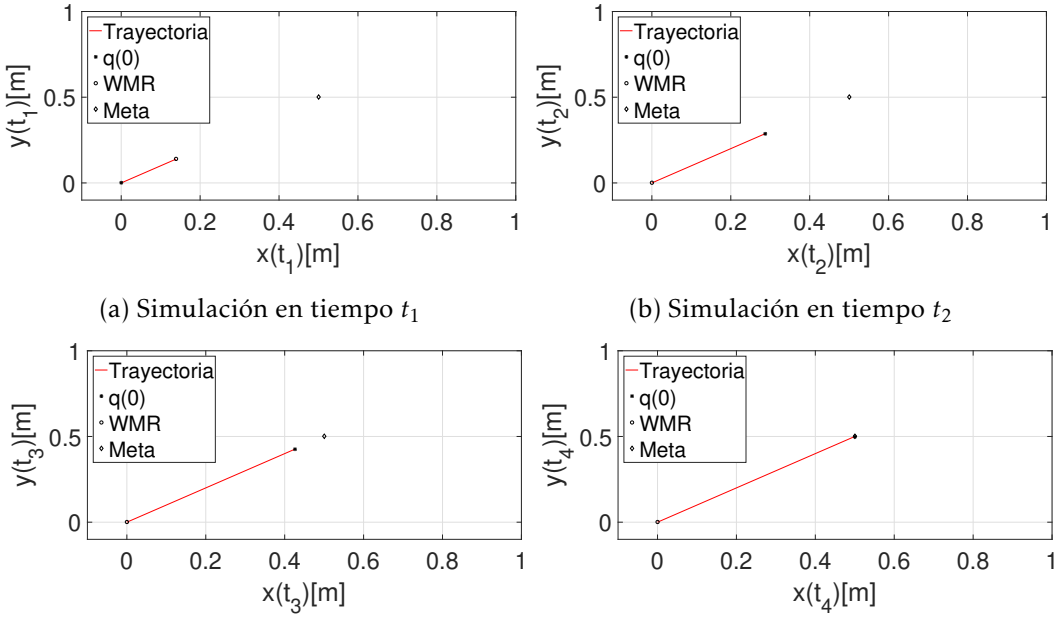


Figura 3.2: Casos de análisis del comportamiento general del sistema de evasión de obstáculos dinámicos.

- El primer caso, se presenta cuando la función de repulsión es nula, $\psi(q, d^{\hat{o}}) = 0$. Esto significa que la distancia entre el robot y el obstáculo es igual a 1 unidad, $\| q - d^{\hat{o}} \| = 1$, por lo tanto, la única fuerza que actua es la de atracción. Esto quiere decir que la función de repulsión es cero y significa que no hay colisión, por lo que la única fuerza es la del controlador de atracción. En la figura(3.3) se puede apreciar el comportamiento del robot cuando el objeto esta alejado.

- El segundo caso, existe cuando la fuerza de atracción es $\psi(q, d^{\hat{o}}) > 0$. Esto significa que la distancia entre el robot y el obstáculo es mayor a 1 unidad, $\| q - d^{\hat{o}} \| > 1$. Significa, que el obstáculo está bastante alejado del robot y no hay necesidad de evadir algo. En este caso, la estabilidad asintótica permanece y el robot llegá a la posición deseada. El comportamiento que presenta es el mostrado en la figura(3.4).



(a) Simulación en tiempo t_1

(b) Simulación en tiempo t_2

(c) Simulación en tiempo t_3

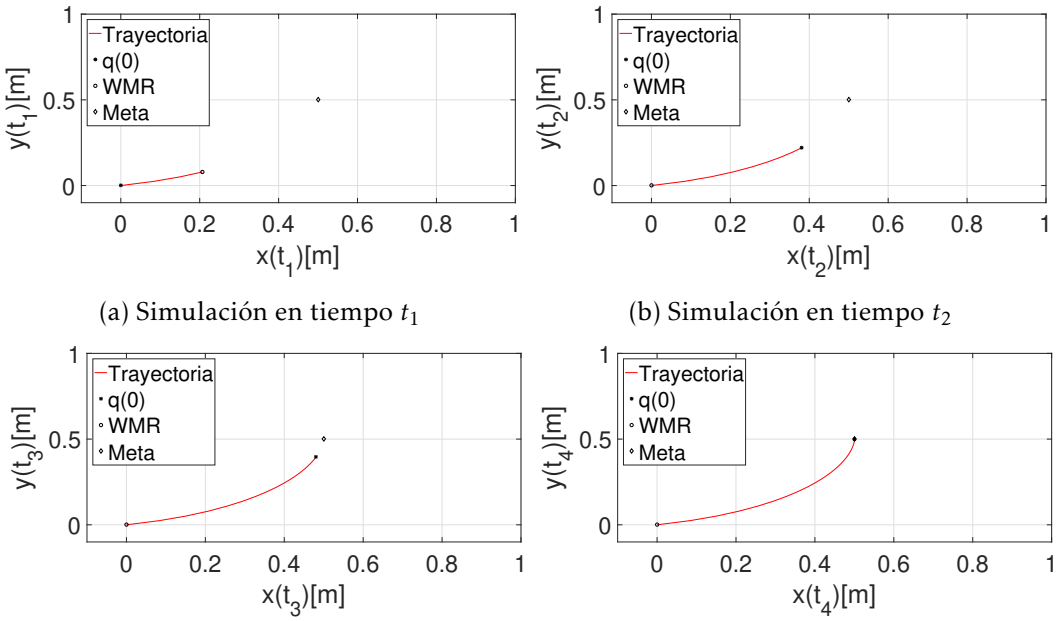
(d) Simulación en tiempo t_4

Figura 3.3: Trayectoria del robot al suceder el caso 1 de la figura(3.2)

- El tercer caso se presenta cuando el robot móvil se mueve cerca del obstáculo, lo que significa que la distancia entre el robot y el obstáculo es

menor a 1 unidad, $\| q - d^{\hat{o}} \| < 1$. En este sentido, se debe realizar un análisis profundo debido a que se generan tres casos especiales. En este punto, cuando el controlador de repulsión incrementa su fuerza, el de atracción la disminuye y cuando el controlador de atracción la aumenta el de repulsión es el que se hace más débil, esto conforme el robot se acerca o aleja del obstáculo. Este análisis, considera la dinámica en lazo cerrado del robot móvil, que se aprecia en la ecuación(3.4).

Cuando sucede el tercer caso, donde $\| q - d^{\hat{o}} \| < 1$ pueden pasar tres escenarios, al analizar la ecuación(3.4) y separarla en sus componentes, se puede entender la decisión que debera tomar el robot, basada en las situaciones de la figura(3.5).



(a) Simulación en tiempo t_1

(b) Simulación en tiempo t_2

(c) Simulación en tiempo t_3

(d) Simulación en tiempo t_4

Figura 3.4: Trayectoria del robot al suceder el caso 2 de la figura(3.2)

Nota: Se recuerda que la posición deseada no se encuentra dentro del radio del obstáculo, esto porque no hay solución que pueda satisfacer ambos objetivos de control de manera simultanea y la condición del segundo caso de la figura(3.5) no puede ser cumplida.

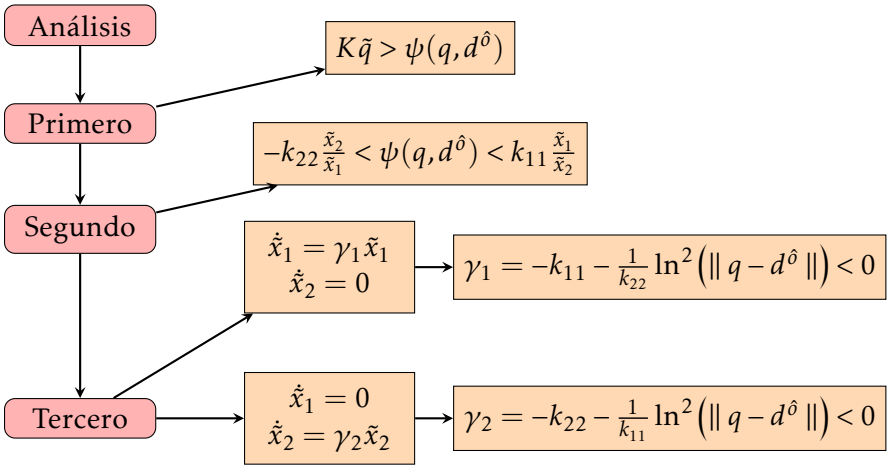


Figura 3.5: Análisis del tercer caso de la figura(3.2), dividido en tres posibles resultados.

El primer caso de análisis se dá cuando $K\tilde{q} > \psi(q, d^{\hat{o}})$, entonces la función atractiva es más grande que la repulsiva, lo que ocasiona que a medida que el robot se acerca al obstáculo su velocidad disminuya y la función de repulsión incremente su valor.

El segundo caso de análisis, ocurre cuando el umbral entre ambas funciones se cruza, $-k_{22}\frac{\tilde{x}_2}{\tilde{x}_1} < \psi(q, d^{\hat{o}}) < k_{11}\frac{\tilde{x}_1}{\tilde{x}_2}$. Entonces el cumplimiento de la condición genera un campo de velocidad opuesta al obstáculo, causando que la evasión de colisión se cumpla.

El tercer caso de análisis se divide en dos partes.

- La primera parte sucede cuando la dinámica de $\dot{\tilde{x}}_1$ persiste y $\dot{\tilde{x}}_2$ es cero.
- La segunda parte sucede cuando $\dot{\tilde{x}}_1$ es igual a cero y $\dot{\tilde{x}}_2$ persiste.

Esto quiere decir que cuando el error ($\dot{\tilde{x}}$) en alguno de los ejes sea cero, el movimiento hacia la posición deseada continua en el eje que no es cero. De esta forma se garantiza que la evasión del obstáculo siempre va a existir, mientras que el obstáculo no este posicionado en el mismo lugar que la meta. Cabe señalar que ambas condiciones de evasión y repulsión no se pueden cumplir simultaneamente, esto implica cuando el obstáculo y la meta ocupan el mismo espacio en un mismo tiempo.

En el momento que se presenta el caso 3 de la figura diagrama(3.5), los campos de velocidad forman una superficie como la de la figura(3.6), compuesta de la función de atracción siendo la zona en forma de valle y la repulsión en forma de cresta. A partir, de la forma de esta superficie se puede concluir que el robot seguirá la trayectoria óptima, hacia la posición deseada evadiendo el obstáculo.

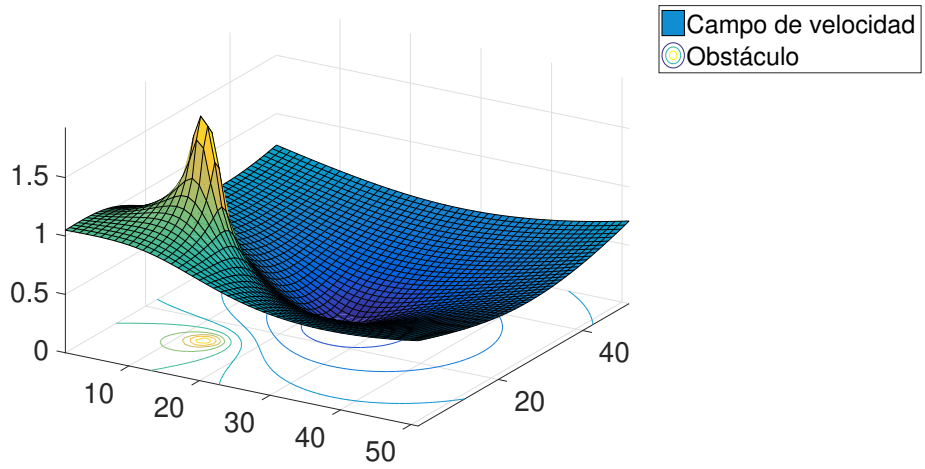


Figura 3.6: Superficie generada mediante los campos de velocidad para el caso 3

La superficie de la figura(3.6) es generada por el campo de velocidad que se crea para un robot con las condiciones; posición deseada en (0.5,0.5), radio del obstáculo $\rho = 0{,}02$, obstáculo en la posición (0.25,0.25). En este caso el obstáculo permanece fijo, para poder representar el campo de velocidad. En el escenario donde el obstáculo se encuentre en movimiento es difícil de generar una representación gráfica, esto, a causa de que cada vez que el robot y el obstáculo se mueven, se genera un campo de velocidad totalmete diferente al del tiempo anterior. De esta manera con el obstáculo fijo se puede generar una idea de como se comportará el campo de velocidad instantaneamente al agregar un obstáculo en movimiento.

3.1.2 *Resultados de simulación*

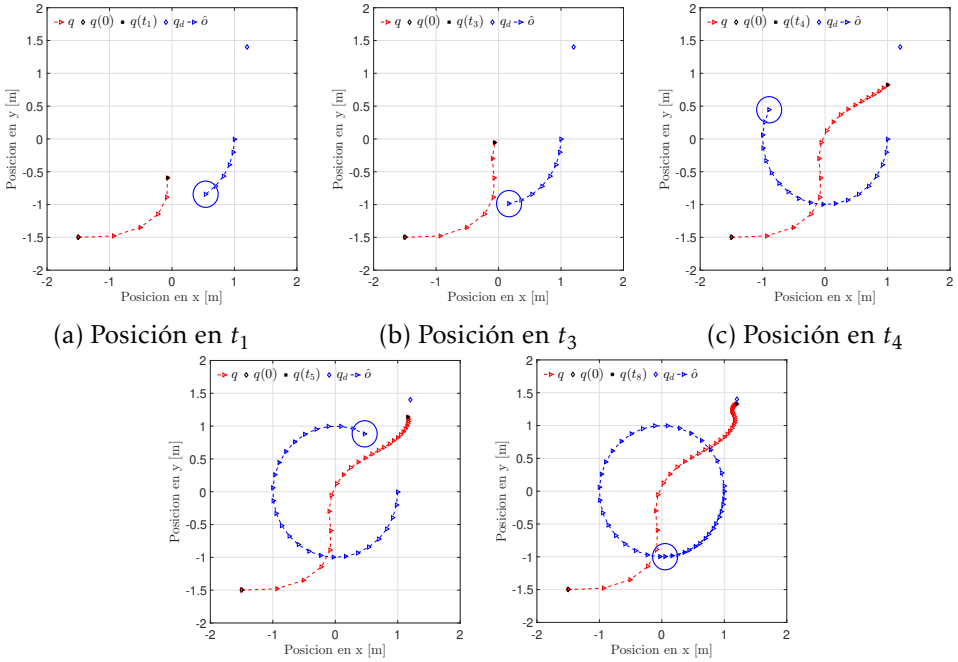
El controlador de navegación propuesto en (3.4)-(3.5) es evaluado a traves de simulaciones numéricas. Finalmente, el obstáculo describe un movimiento

circular respecto al tiempo, con un radio $\rho = 0{,}02$. El comportamiento natural del sistema causa que el robot evite con premeritación el obstáculo, antes de que este pueda colisionar, aún estando ambos en movimiento, como se muestra en la figura(3.7).

Se usarón las condiciones iniciales de la tabla(3.1) para generar la figura(3.7). Cuando el robot esta a punto de colisionar con el obstáculo que presenta una trayectoria circular y que interfiere en la trayectoria del robot, el robot puede evitar la colisión con premeritación.

Simbolo	Condición inicial	Valor numerico
$q(0)$	$[x(0), y(0)]^T$	$[-1{,}5, -1{,}5]^T$
q_d	$[x_d, y_d]^T$	$[1{,}2, 1{,}4]^T$
o	$[o_x, o_y]^T$	$[\cos 0, -\sin 0]^T$

Tabla 3.1: Condiciones iniciales usadas para la simulación de 3.7



(a) Posición en t_1 (b) Posición en t_3 (c) Posición en t_4

(d) Posición en t_5 (e) Posición en t_f

Figura 3.7: Visualización secuencial de la navegación del robot móvil usando el controlador u con evasión de obstáculos dinámicos de la ecuación(3.4).

3.2 Sistema multirobot con evasión de obstáculos dinámicos para la navegación autonoma enfocada en el problema de **encuentro** aplicando control por comportamientos analíticos

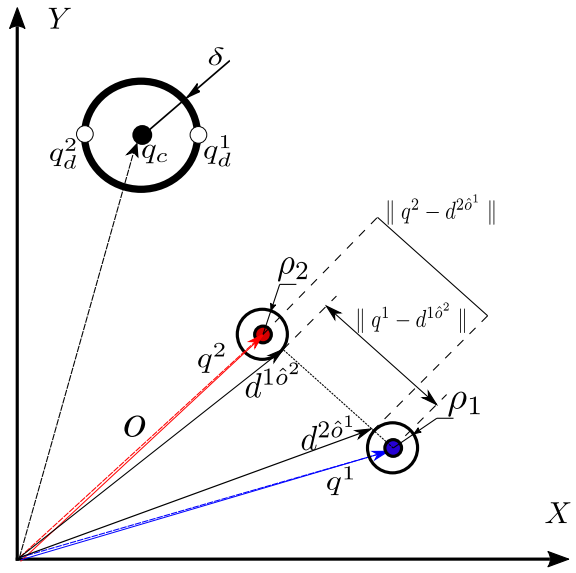


Figura 3.8: Nuevo escenario con 2 robots que deben llegar a un área específica en un tiempo dado.

Cuando se usa la metodología de comportamientos analíticos se le "enseña" al robot a cumplir objetivos predefinidos, en este caso a evadir el obstáculo dinámico. Principalmente, se propone que un robot puede ser interpretado como obstáculo visto desde la perspectiva de otro robot. Esto debido a que los controladores de evasión propuestos por Villalvazo-Covián et al. (2021) y Meza-Sánchez et al. (2019) pueden ser ligeramente modificados, para ser aplicados en sistemas multirobot, solo basta replantear los obstáculos dinámicos como robots.

Una de las ventajas de la metodología abordada, es que se puede sumar objetivos de control a los controladores sin afectar su convergencia. Permitiendo ampliar la metodología para ser resolver tareas colaborativas en sistemas multirobot y a su vez cumplir con nuevos objetivos de navegación. A continuación, se diseña una versión modificada de la metodología de control por comportamientos analíticos aplicado a sistemas con obstáculos dinámicos, para analizar y resolver la tarea de **encuentro**.

Se propone el escenario de dos robos que se deben evadir mutuamente y llegar a un punto de encuentro en un mismo tiempo. Se considera este escenario, debido a que se pretende generar una forma general para ser aplicada hasta n robots. Se redefine el controlador propuesto en la ecuación(3.4), este nuevo controlador servira como la regla de aprendizaje para ser usada en la metodología de comportamientos analíticos.

A partir de ahora, el **comportamiento no forzado** del sistema es la dinámica del robot sin ninguna entrada de control, siendo $q = u$. Mientras que el **comportamiento forzado** sera el controlador de la ecuación(3.4), donde la función de repulsión($\psi(q, d^{i\hat{o}})$) propuesta en la ecuación(3.4) sera modificada por el controlador obtenido del trabajo de Meza-Sánchez et al. (2019), quedando nuestra u como se muestra en la ecuación(3.25).

Por otra parte, el **comportamiento aprendido** sera encontrado mediante la metodología de control por comportamientos analíticos y se representará de la forma, u_{l_k}.

Finalmente, los objetivos de control son los siguientes:

- El primer objetivo de control es el de navegación, los dos robots deben llegar siempre a la meta.
- El segundo es el de evasión, los robots siempre se deben evadir entre ellos.
- El tercer objetivo de control es el de tiempo de navegación, ambos robots deben llegar a la meta en el tiempo especificado sin importar su posición inicial.

3.2.1 *Formulación de las posiciones finales*

Se propone, diseñar un sistema de navegación que considere las restricciones del problema de **encuentro**, el cual consiste en dos robots móviles posicionados alrededor del cuadrante coordenado I, con el objetivo de llegar al mismo tiempo a la meta. La meta se diseña como un área circular, donde cada robot tendrá una posición final. Esta posición, se diseña a partir de poligonos inscritos, y así generar las posiciones deseadas de n.

La figura(3.8) muestra como se ubicarán los robots en posiciones alrededor de la zona de meta, donde, mediante el controlador porpuesto en la ecuación(3.24), ambos robots deberán llegar como se muestra en la figura(3.8) al mismo tiempo y a su vez evadirse entre ellos.

Partiendo de las coordenadas de un área de reunión, se generan los puntos donde los robots debe llegar. Basado en la ecuación(3.23), se define un vector de rango $\frac{2\cdot\pi}{n}$ hasta $2\cdot\pi$, donde n es la cantidad de robots involucrados en el sistema. Haciendo uso de la geometria involucrada en un polígono inscrito, donde las ecuaciónes (3.22) y (3.23) nos proporcionará las posiciónes deseadas en el eje X y el eje Y respectivamente. Dicho polígono tiene rango que va desde 0 hasta $2\cdot\pi$, con incrementos dependientes del número de robots involucrados, como se muestra en la ecuación(3.22).

$$\theta_i = \frac{2\pi}{n} i, \quad i = 1, ..., n. \tag{3.22}$$

De esta manera, se utiliza la variable θ para plantear las condiciones iniciales para cada robot móvil, usando poligonos inscritos para diseñar un área donde los robots deben llegar. Mediante esta premisa se pueden calcular pas posiciones deseadas con la ecuación(3.23) para i robots.

$$q_d^i = [x_c + r\cdot\cos(\theta_i), y_c + r\cdot\sin(\theta_i)]^T, \tag{3.23}$$

Donde x_c y y_c, son las coordenadas en cada eje respectivamente para la posición deseada del área, i es el indice de cada robot, r es el radio del área de convergencia, proponiendo el circulo como área deseada y donde n es el número de robots involucrados y $r = 0{,}1$.

3.2.1.1 *Poligonos inscritos*

La principal razón de diseñar las posiciones deseadas mediante poligonos inscritos consiste en la sencilles para formar figuras haciendo uso de puntos distribuidos alrededor de un circulo. La característica de aplicar esta metodología en sistemas multirobot es que habre la posiblidad de organizar grupos de robots en diversas formas basadas en figuras geometricas. En el caso de aplicación de dos robots móviles, la figura formada no es en sí una figura sino una linea como

se muestra en la figura(3.9a), si se incrementa el número de robots equivale a aumentar el numero de puntos dentro del circulo, por ejemplo, en el caso de tres robots la figura formada seria un triangulo como se ve en la figura(3.9b), mientras que en el caso de cuatro robots seria un cuadrado como se aprecia en la figura(3.9c) y así sucesivamente. Sin embargo, esto no se límita a una sola figura, mediante poligonos inscritos se puede incorporar multiples figuras en un mismo escenario como se muestra en la figura(3.9d).

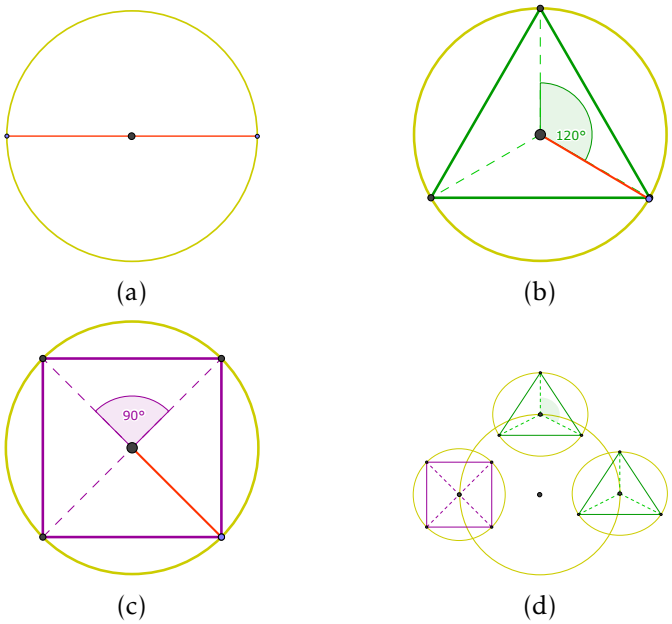


Figura 3.9: Representación de los poligonos inscritos en una circunferencia aplicados en el sistema multirobot

3.2.1.2 *Diseño del controlador general del sistema de **encuentro***

Para cumplir los objetivos de nuestra propuesta, redefinimos el controlador propuesto por Villalvazo-Covián et al. (2021) y agregando un tercer controlador parcial y así obtener la ecuación(3.24). Dicho controlador es aplicado a cada

robot y el algoritmo evolutivo debe encontrar una solución para que el sistema multi-robot cumpla los objetivos proupestos.

$$u_f^i = K^i\left(q_d^i - q^i\right) + \psi(q^i, d^{i\hat{o}^\beta})\begin{bmatrix} y_d^i - y^i \\ -\left(x_d^i - x^i\right) \end{bmatrix} \tag{3.24}$$

Por otro lado, el controlador general para el control de posición del robot móvil, esta dado por la ecuación(3.25), donde se incorpora el controlador parcial de la ecuación(3.24) y el controlador encontrado por comportamientos analíticos u_{l_k}

$$u = u_f^i + u_{l_k} \tag{3.25}$$

Donde u_{l_k} es el controlador que se debe encontrar mediante el paradigma de comportamientos analíticos.

Mientras que el controlador parcial de evasión originalmente propuesto en la ecuación(3.5) representado como $\psi(q^i, d^{i\hat{o}^\beta})$ ha sido reemplazado por el controlador de evasión u_{gp6} del trabajo propuestos por Meza-Sánchez et al. (2019) que se aprecia en la ecuación(3.26), con unas lijeras modificaciones para manejar solamente valores reales.

$$\psi(q^i, d^{i\hat{o}^\beta}) = \mathrm{Re}\Big(\ln(\mathrm{abs}(\ln(\cos((\ln\left(\|q^i - d^{i\hat{o}^\beta}\|\right)))))))\times\Big(\ln\left(\| q^i - d^{i\hat{o}^\beta} \|\right)\Big), \beta \in \{1,2\} \tag{3.26}$$

Con $\hat{o} = [o_x, o_y]^T$ siendo la distancia del robot acual a la frontera robot anterior y $\| q - d^{i\hat{o}^\beta} \|$ como la distancia del robot móvil hacia la frontera del robot anterior.

Para calcular la frontera de los robots $d^{i\hat{o}^\beta}$ partimos de la ecuación(3.27), así obtener la distancia entre ambos robots y evitar que colisionen.

$$d^{i\hat{o}} = \hat{o} + \frac{\rho\left(q^i - \hat{o}\right)}{\|q^i - \hat{o}\|}. \tag{3.27}$$

En el caso de caso de 2 robots el análisis para obtener cada distancia sería dada como se muestra en las ecuaciones (3.29).

$$d^{1\hat{o}^2} = \hat{o}^2 + \frac{\rho\left(q^i - \hat{o}^2\right)}{\|q^i - \hat{o}^2\|} \tag{3.28}$$

$$d^{2\hat{o}^1} = \hat{o}^1 + \frac{\rho\left(q^i - \hat{o}^1\right)}{\|q^i - \hat{o}^1\|} \tag{3.29}$$

3.3 Consideraciones para el diseño del comportamiento aprendido

El comportamiento aprendido es propuesto mediante un proceso de Programación Genética. Para esto es necesario, definir la función de aptitud, las funciones y terminales que se utilizarán para generar los árboles sintácticos, así como los escenarios de entrenamiento

3.3.1 *Función de aptitud*

En esta sección se propone la función de aptitud, la cual sera apliacada en el algoritmo evolutivo y arrojará como resultado el puntaje de desempeño de cada una de las soluciones encontradas por al algoritmo evolutivo. La función de aptitud se define como:

$$F_v^1 = \frac{1}{\gamma}\sum_{\alpha=1}^{\gamma}\left(\|q_d^i - q^i\|\right)^2 + (t_a - t_c^i)^2, \quad i = 1,...n, \tag{3.30}$$

Donde, son 10 las condiciones iniciales evaluadas correspondientes para cada robot, $q_d^i - q^i$ es el error de posición entre cada robot y su posición deseada, mientras que $t_a - t_c^i$ corresponde al error en el tiempo de llegada de cada robot respecto al tiempo deseado. La función de aptitud es la media de la suma del desempeño para cada condición inicial, siendo el desempeño la suma de los errores de cada robot involucrado en el sistema, quedando como se muestra en la ecuación(3.31).

$$F_v^1 = \frac{1}{10}\sum_{\alpha=1}^{10}\left(\|q_d^1 - q^1\|\right)^2 + \left(\|q_d^2 - q^2\|\right)^2 + (t_a - t_c^1)^2 + (t_a - t_c^2)^2, \tag{3.31}$$

La principal característica de la función de desempeño es que cuenta con penalizaciones a la evolución que se definen asignando valores constantes $P_1 > 0$ y $P_2 > 0$ cuando existe la situación de colisión y no se cumple la tarea de encuentro. Por otro lado, $P_3 > 0$ se penaliza cuando existe algun error número o se indefine la solución, de lo contrario el desempeño esta dado por F_v^1. De esta manera, se puede establecer la tarea de optimización como:

$$\begin{aligned} \min_{F_v} \quad & F_v = \begin{cases} P_1 & \text{Colisión existe} \\ P_2 & \text{Encuentro no cumplido} \\ P_3 & \text{Error número o indefinido} \\ F_v^1 & \text{De otra forma} \end{cases}, \\ s.t. \quad & \dot{q}^i = u^i, \\ & u^i = u_f^i + u_{l_v}, \\ & S^\alpha, \alpha \in \{1, ..., \gamma\}. \end{aligned} \tag{3.32}$$

- Se penaliza P_1 si sucede, con $P_1 = 3\times10^6$.
- Se penaliza P_2 si sucede, con $P_2 = 4\times10^6$
- Se penaliza P_3 se sucede, con $P_3 = 5\times10^6$.
- Si no sucede P_1, ni P_2, ni P_3 entonces se aplica la ecuación(3.31).

3.3.2 *Funciones y Terminales*

Las funciones y terminales son una pieza importante para el proceso evolutivo, debido a que son requeridos para la formación del invidividuo. Es decir, definen el espacio de busqueda de las posibles soluciones, por lo que se deben tener ciertas consideraciones al momento de seleccionarlas:

- Las funciones y terminales se eligieron a partir de los estados del sistema, posición, velocidad, y parámetros propios del sistema como las distancias entre los robots y las distancias entre los objetivos.

Funciones Trascendentes	Significado	Funciones Algebraicas	Significado
$s_h(\cdot)$	Seno Hiperbolico	$\sqrt{(\cdot)}$	Raiz cuadrada
$c_h(\cdot)$	Coseno Hiperbolico	$(\cdot)^2$	Exponente al cuadrado
$t_h(\cdot)$	Tangente Hiperbolica	$norm(\cdot)$	Norma Euclidiana
$cs_h(\cdot)$	Cosecante Hiperbolica	$abs(\cdot)$	Valor absoluto
$sc_h(\cdot)$	Secante Hiperbolica	$real(\cdot)$	Valor real
$ct_h(\cdot)$	Cotangente Hiperbolica	$+$	Suma
$c(\cdot)$	Coseno	$-$	Resta
$s(\cdot)$	Seno	$\backslash$	División
$t(\cdot)$	Tangente	$\times$	Multiplicación
$cs(\cdot)$	Cosecante	$power(\cdot,\cdot)$	Exponenciación
$sc(\cdot)$	Secante	$max(\cdot,\cdot)$	Máximo de dos valores
$ct(\cdot)$	Cotangente	$min(\cdot,\cdot)$	Mínimo de dos valores
$s_i(\cdot)$	ArcoSeno		
$c_i(\cdot)$	ArcoCoseno		
atr$(\cdot)$	ArcoTangente Real(Unaria)		
at2r$(\cdot,\cdot)$	Arcotangente Real(Binaria)		
ln$(\cdot)$	Logaritmo Natural		
$e^{(\cdot)}$	Exponencial		

Tabla 3.2: Funciones usadas en el aprendizaje.

Las consideraciones ayudaron a definir de manera correcta las funciones y terminales como las variables usadas para la simulación, en este caso la simulación de dos robots móviles omnidireccionales las cuales se pueden apreciar en la tabla(3.3), estas incluyen las velocidades de ambos robots, la distancia entre ambos, las posiciones de los robots, el tiempo deseado y su desplazamiento.

En cuanto a las funciones, se pueden clasificar en dos tipos, las funciones **unarias** y funciones **binarias**. Las funciones unarias son aquellas que solo reciben un parámetro de entrada, mientras que las funciones binarias son las que reciben dos parámetros de entrada. Dentro del grupo de funciones unarias y

Terminales	Significado
x_d^1	Posición deseada del WMR1 en el eje x
y_d^1	Posición deseada del WMR1 en el eje y
V_x^1	Velocidad de desplazamiento en el eje x del WMR1
V_y^1	Velocidad de desplazamiento en el eje y del WMR1
x^1	Posición actual en el eje x del WMR1
y^1	Posición actual en el eje y del WMR1
t_c^1	Tiempo actual transcurrido del WMR1
x_d^2	Posición deseada del WMR2 en el eje x
y_d^2	Posición deseada del WMR2 en el eje y
V_x^2	Velocidad de desplazamiento en el eje x del WMR2
V_y^2	Velocidad de desplazamiento en el eje y del WMR2
x^2	Posición actual en el eje x del WMR2
y^2	Posición actual en el eje y del WMR2
t_c^2	Tiempo actual transcurrido del WMR2
t_a	Tiempo de llegada deseado
$d_x^{1\hat{o}^2}$	Distancia entre el WMR1 a la frontera del WMR2 en el eje x
$d_y^{1\hat{o}^2}$	Distancia entre el WMR1 a la frontera del WMR2 en el eje y
$d_x^{2\hat{o}^1}$	Distancia entre el WMR2 a la frontera del WMR1 en el eje x
$d_y^{2\hat{o}^1}$	Distancia entre el WMR2 a la frontera del WMR1 en el eje y

Tabla 3.3: Terminales usadas en el aprendizaje.

binarias, existen dos tipos de funciones, las aritméticas y las trascendentes, las cuales se muestrán en la tabla(3.2).

Se considera la tabla(3.2) con el objetivo de abordar la mayor cantidad de funciones usables para el proceso evolutivo. Permitiendo obtener un espacio de búsqueda amplio y asi tener un algoritmo evolutivo con una mayor flexibilidad en la búsqueda de soluciones.

3.4 Diseño del algoritmo evolutivo

El Algoritmo Evolutivo originalmente diseñado en python es un programa que hace uso de la librería “DEAP”, la cuál es un programa para el diseño de algoritmos evolutivos distribuidos y tiene una gran variedad de herramientas que

le permiten ejecutar y diseñar algoritmos complejos, tanto, algoritmos genéticos como programación genética. El Algoritmo Evolutivo diseñado para este proyecto se basó en la sintesis de comportamientos analíticos, el cual hace uso principalmente de programación genética, mas no se limita a ella. Entonces, usando programación genética se diseñó un programa capaz de automatizar la sintesis de controladores para cumplir con los objetivos de la tarea de navegación basada en el problema de **encuentro**, la que se definio en los primeros capitulos. Dicho algoritmo esta diseñado mediante modulos con funciones especificas, que permiten ampliar el uso de la metodología a diversas aplicaciones como el uso de sistemas CUDA por ejemplo. El diseño en general se basa en la versión de DEAP 1.15 para el lenguaje de programación python 3.7 y haciendo uso de multiprocesos. De esta manera, se pueden simular multiples condiciones iniciales en multiples procesos de forma paralela y agilizar la evolución.

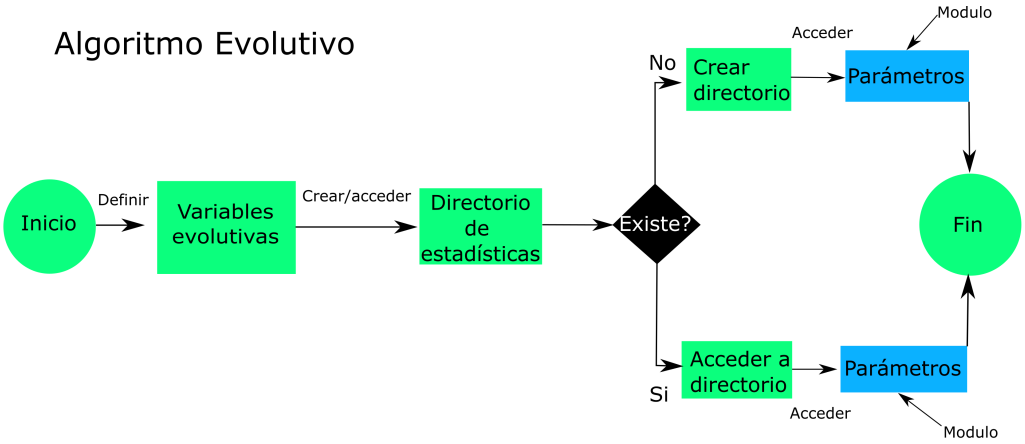


Figura 3.10: Modulo número 1 del Algoritmo Evolutivo, creación de directorios e inicio del proceso evolutivo.

El primer modulo se presenta en la figura(3.10), que consiste en definir el tiempo límite para llegar al encuentro, la cantidad de generaciones y población para introducir al algoritmo evolutivo, asi como la cantidad de condiciones iniciales a simular. Al iniciar el algoritmo, el modulo analizara si existe el directorio de las estadísticas, en caso de no existir creara uno. En el directorio creado se almacenará la información de las estadística y las soluciones arrojadas por el programa y en caso de existir, el programa se pasa directamentee al siguiente módulo, donde se definen los parámetros evolutivos y esta representado en la figura(3.11).

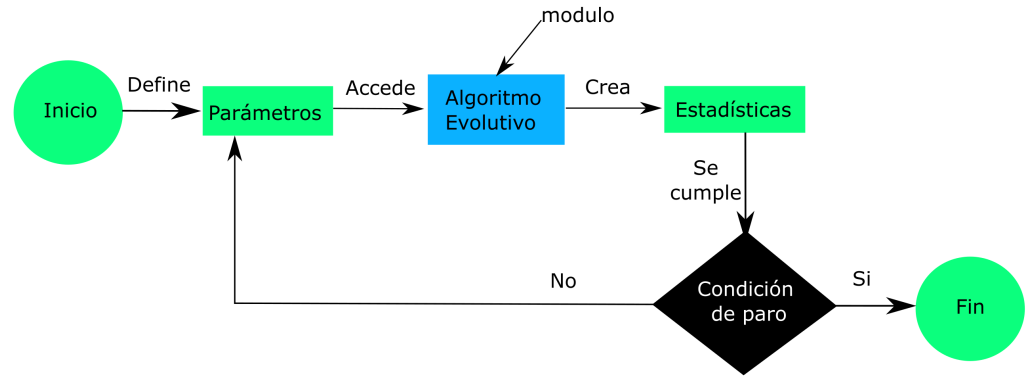


Figura 3.11: Modulo número 2 del Algoritmo Evolutivos, parámatros de la evolución y estadísticas.

En el modulo número 2 se definen los parámetros evolutivos, los cuales incluyen las funciones de la tabla(3.2) y terminales de la tabla(3.3), la creación de las herramientas que llaman al algoritmo evolutivo y las variables que permiten al algoritmo evolutivo el acceder al modulo de la función de desempeño y evaluarla mediante las características de la Programación Genética mediante árboles sintácticos. Además, en este modulo es donde se presenta el programa principal que inicia y termina la evolución despues de cumplir con los criterios de paro y arroja las estadísticas del proceso evolutivo. El modulo 2 esta representado en la figura(3.11), despues de introducir los parámetros requeridos para el método de programación genética, se inicializa una primera población y se comienza con el proceso de la función de desempeño. Una vez que se obtiene la evaluación de cada individuo, se generan las estadísticas y se pasa a la siguiente generación. Se repite el proceso hasta que se cumplan dos condiciones principales:

- El proceso evolutivo alcanzo el máximo de generaciones predefinidas.
- El individuo fue tan grande que consumió la memoria disponible.

Nota: La segunda condición esta diseñada para terminar el proceso evolutivo solo en el caso extremo que el individuo generado utilize todos los recursos de la computadora utilizada para la evolución. Mediante esta condición se forza a que el proceso evolutivo termine y asi evitar un desbordamiento por exceso de dimensiones en el arbol generado.

Por otro lado, en el tercer modulo representado por la figura(3.12) se define el proceso de cálculo de desempeño de manera general, el cual recibe como parámetros de entrada la cantidad de condiciones inciales usadas, el individuo generado por el algoritmo evolutivo y el tiempo deseado. En este modulo se implementa la ecuación(3.31) y las restricciones de 3.32, donde, mediante procesamiento en paralelo se evalua cada condición inicial en un proceso independiente, permitiendo asi, evaluar una condición inicial por proceso, esto quiere decir que habria n procesos por n condiciones iniciales. Se hace el promedio del desempeño de las simulaciones en paralelo y se pasa el resultado al algoritmo evolutivo, basado en las penalizaciones de 3.32, el proceso se repite hasta que se cumple alguna de las condiciones de paro definidas anteriormente.

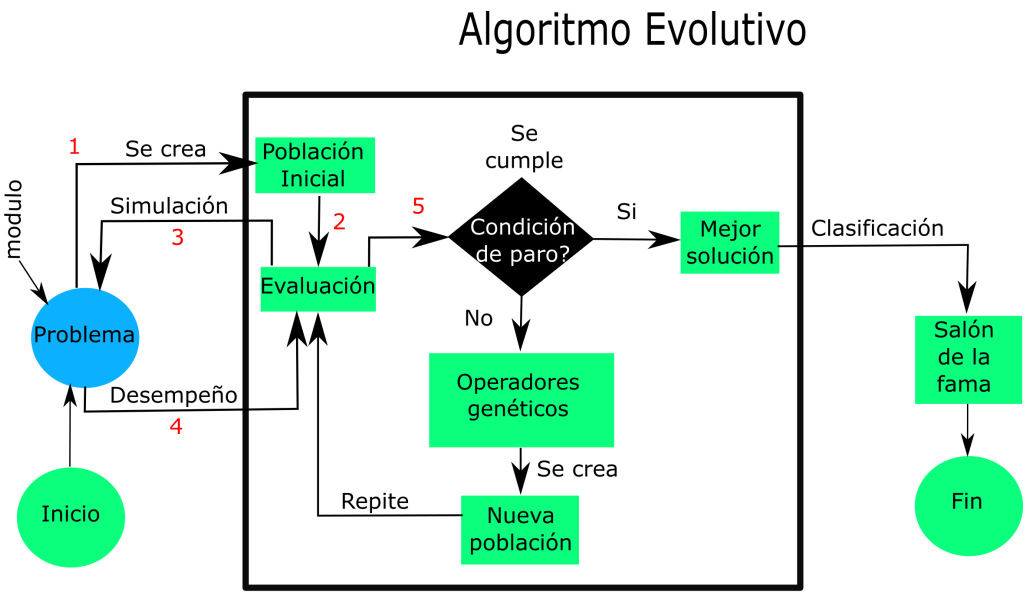


Figura 3.12: Modulo número 3, comportamiento generalizado del proceso evolutivo.

Finalmente, en el modulo 4 se realiza la simulación del problema de encuentro, donde se definen los modelos cinematicos para los dos robots moviles, se calculan las distancias entre cada uno se implementan las ecuaciónes 3.22 3.29. Dentro de la simulación ya se encuentro predefinido el controlador propuesto en la ecuación(3.24), el cual incorpora la solución propuesta por el Algoritmo Evolutivo. Se realiza la simulación para generar las variables de salida necesarias para el cálculo del desempeño.

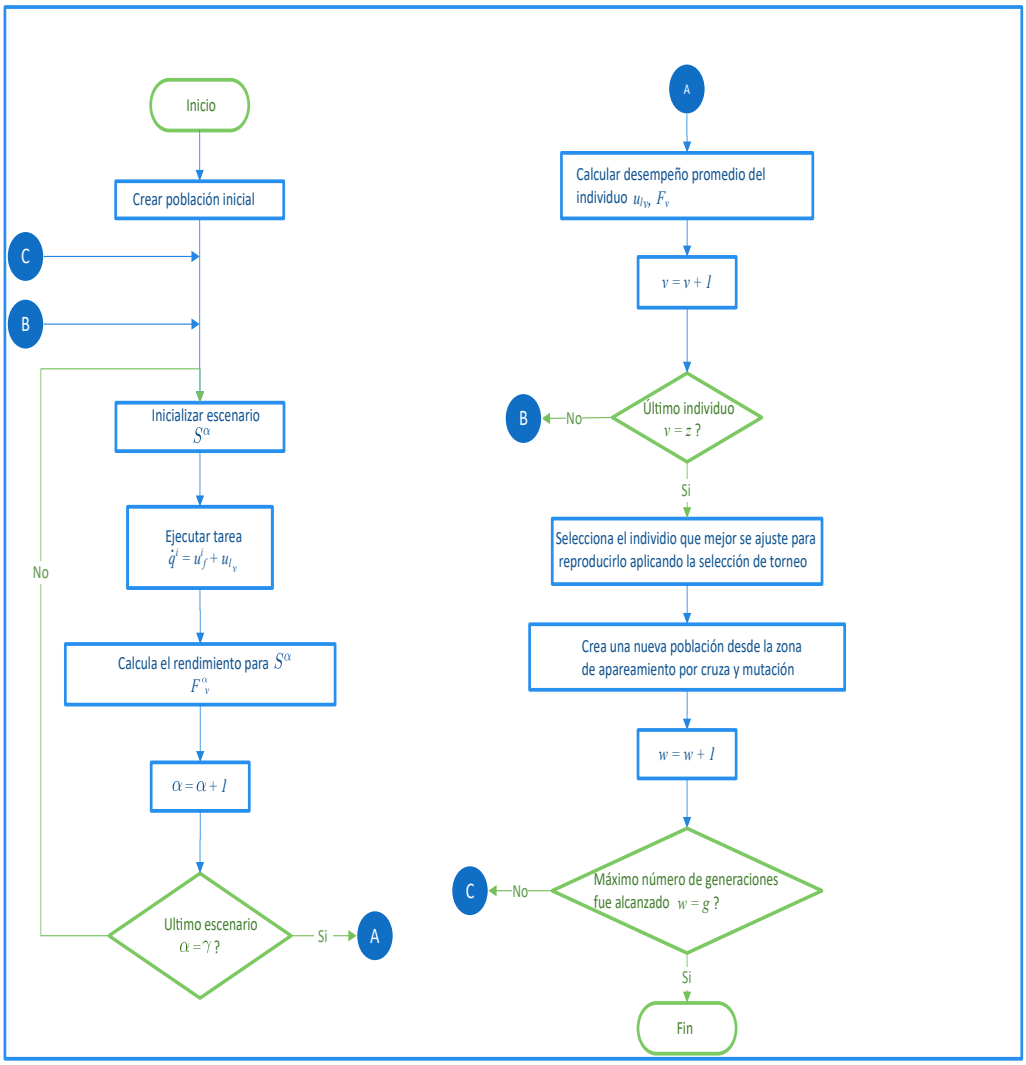


Figura 3.13: Descripción computacional del proceso evolutivo para buscar los controladores optimizados que resuelvan la tarea de **encuentro** con evasión de colisión y política de tiempo de llegada.

3.4.1 *Condiciones iniciales usadas en el aprendizaje*

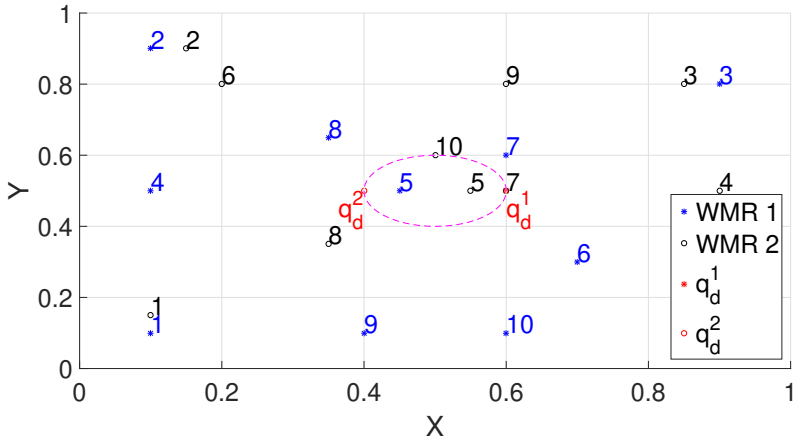


Figura 3.14: Distribución de las condiciones iniciales a lo largo del eje coordenado.

Para la evolución del sistema multirobot se definierón 10 escenarios, los cuales son 10 condiciones inciales distribuidas alrededor del eje coordenado en el cuadrante I y acotadas entre 0 y 1 en el eje x y el eje y. Como se muestra en la tabla(3.4), se definierón las condiciones iniciales para dos robots móviles omnidireccionales, definidas como las posiciones de cada robot en el sistema coordenado, las posiciones deseadas y el radio de los robots. Se considera que ambos robots cuentan con el mismo radio y que las posiciones deseadas estan dadas por la ecuación(3.23).

Condiciones iniciales					
α	WMR 1	WMR 2	α	WMR 1	WMR 2
#	$q^1(0)$	$q^2(0)$	#	$q^1(0)$	$q^2(0)$
1	$[0.1, 0.1]^T$	$[0.1, 0.15]^T$	2	$[0.1, 0.9]^T$	$[0.15, 0.9]^T$
3	$[0.9, 0.8]^T$	$[0.85, 0.8]^T$	4	$[0.1, 0.5]^T$	$[0.9, 0.5]^T$
5	$[0.45, 0.5]^T$	$[0.55, 0.5]^T$	6	$[0.7, 0.3]^T$	$[0.2, 0.8]^T$
7	$[0.6, 0.6]^T$	$[0.6, 0.5]^T$	8	$[0.35, 0.65]^T$	$[0.35, 0.35]^T$
9	$[0.4, 0.1]^T$	$[0.6, 0.8]^T$	10	$[0.6, 0.1]^T$	$[0.5, 0.6]^T$

Tabla 3.4: Tabla de condiciones iniciales usadas para la evolución

3.4.2 *Parámetros*

En primera instancia, se hablará acerca de los parámetros utilizados para realizar el proceso evolutivo abordado en el capitulo(3). Primero se definen parámetros generales de la evolución, los cuales incluyen la cantidad de corridas, de generaciones, el tamaño de la población, la probabilidad de cruza y mutación, el máximo tamaño de profundidad del arbol generado por el algoritmo evolutivo, el tipo de selección, asi como el método de elitismo.

De forma general, se definió para todas las ejecuciones el tipo de selección como el metodo de torneo, el cual consiste en elegir un número predefinido de participantes de manera aleatoria, los cuales "compiten entre si" para ver cual es el mejor y mediante elitismo conservarlo y pasar a la siguiente generación. El método de cruza se selecciona mediante el proceso mitad y mitad, el cual consiste en implementar dos metodos de generación de arboles sintácticos, el método "grow" y el metodo "full". El método mitad y mitad, genera arboles sintácticos de diferente forma y tamaño, los cuales una mitad se genera mediante el metodo "full" y la otra mitad se genera por el metodo"grow". Además, para establecer un limite en el tamaño de los arboles sintácticos generados se definio un número estático de profundidad, y asi evitar que el arbol crezca infinitamente. En cuanto a la forma en que se selecciona el mejor individuo es mediante elitismo, que es basicamente tomar al mejor individuo de toda la población actual y pasarlo a la siguiente generación.

Durante la primera ejecución, se realizaron corridas con los parámetros definidos en la tabla(3.5), como se muestra en la tabla, primero se realizaron 4 corridas con una población de 200 individuos y 200 generaciones, esto nos dio como resultado un total de **126175** soluciones que son clasificadas mediante su desempeño. Cabe mencionar que los comportamientos aprendidos cumplen con los objetivos de control propuesto en el capitulo(2), de manera que el comportamiento con menor desempeño sera el que mejor resultado que cumpla los objetivos.

Mientras que en el segundo conjunto de soluciones, se incremento el tamaño de la población a 300 y se modifico la probabilidad de cruza y mutación, se mantuvo el número de participantes en el torneo igual que el primer conjunto. En el segundo conjunto de ejecuciones se obtuvieron **189455** soluciones que al igual que el primer conjunto son clasificadas por su desempeño. Se espera que, conforme se incremente el número de población y el número de generaciones el

algoritmo evolutivo tenga un mayor espacio de búsqueda, permitiendo encontrar soluciones cada vez mejores.

Seleccionar los parámetros para el algoritmo evolutivo, es una tarea que comienzao por planear el tiempo de ejecución, debido a que este aumenta de manera exponencial conforme el espacio de búsqueda se vuelve más grande. Considerando el tiempo de ejecución se comienza con proponer el primer conjunto de corridas, el cual se muestra en la tabla(3.5) dichos parámetros tambien incluyen las funciones y terminales de las tablas(3.3)-(3.2). Se realizan las primeras ejecuciones y al obtener un resultado, se analiza la posibilidad de incrementar la cantidad de generaciones y población, asi como definir si se cambia la probabilidad de cruza y mutación. En la tabla(3.5) se aprecian los parámetros definidos, asi como el tiempo de ejecución promedio que tomo terminar el proceso evolutivo.

Tabla 3.5: Parámetros para tres configuraciones del proceso evolutivo empleado por Programación Genética en la busqueda de controladores no lineales u_{l_v}, $v \in \{1,...,rgz\}$ que inducen los comportamientos aprendidos.

Parámetros	**Ev 1**	**Ev 2**	**Ev 3**
No. Generaciones	200	200	300
Tamaño de población	200	300	400
Probabilidad de cruza	60%	70%	60%
Probabilidad de mutación	40%	30%	30%
Participantes del torneo	7	7	15% de pob
No. individuos aproximados	126,175	189,455	222,483
Tiempo promedio de ejecución (dias)	4	4	7

Por otro lado, para el tercer conjunto de soluciones se decidio incrementar el número de generaciones a 300 y el tamaño de población a 400, asi como la cantidad de participantes en el proceso de torneo al 15% de la población total, obteniendo un aproximado de **222 483** soluciones. Los parámetros mostrados en la tabla(3.5) muestra como el tiempo de ejecución incremento bastante en comparación con el primer conjunto y el segundo. Esto se debe, a que al imcrementar la cantidad de generaciones y población se aumenta el tiempo de computo necesario para el algoritmo evolutivo. El conjunto de 3 ejecuciones representan tan solo las ejecuciones completas o con generaciones avanzadas, debido a que otras ejecuciones no tomadas en cuenta fueron terminadas de manera abrupta por motivos ajenos al proyecto.

Por último, en la tabla(3.6) se presentan las características de la computadora con la cual se realizaron las evoluciónes del capitulo(4) y con la cual se obtuvieron los resultados analizados.

Tabla 3.6: Características de la computadora usada para el proceso evolutivo.

Características	Descripción
Modelo	DELL Precision Tower 7910
Procesadores	2 (20 threads x CPU)
RAM	64 GB (8x8GB) 2133MHz DDR4 RDIM M ECC
Procesador	Dual Intel Xeon Processor E5-2 687W v3(10C HT, 25MB Cache, 3.1GHz Turbo).

Posiciones deseadas				
Descripción	WMR 1	WMR 2	WMR 3	WMR 4
q_d^i	q_d^1	q_d^2	q_d^3	q_d^4
Aprendizaje	$[0.600, 0.500]^T$	$[0.400, 0.500]^T$		
Validación	$[0.600, 0.500]^T$	$[0.400, 0.500]^T$		
3 WMRs	$[1.000, 0.500]^T$	$[0.250, 0.933]^T$	$[0.250, 0.067]^T$	
4 WMRs	$[1.000, 0.500]^T$	$[0.500, 1.000]^T$	$[0.000, 0.500]^T$	$[0.5, 0.0]^T$

Tabla 3.7: Posiciones deseadas para las diferentes fases de simulación.

Consideraciones				
Escenarios	t_s	$\#S_\alpha$	ρ^i	δ
Aprendizaje	4.5seg	10	0.02[m]	0.1[m]
Validación	6.0seg	4	0.02[m]	0.1[m]
3 WMRs	6.0seg	4	0.02[m]	0.5[m]
4 WMRs	6.0seg	4	0.02[m]	0.5[m]

Tabla 3.8: Consideraciones especiales para las simulaciones; tiempo de simulación, cantidad de condiciones iniciales, radio de cada robot y el radio del área del poligono circunscrito.

4

Análisis de resultados

En este capítulo se abordan los resultados obtenidos para el problema de navegación autónoma de tipo "encuentro" en un sistema multirobot. Los resultados consisten en el análisis de 3 conjuntos de ejecuciones, donde, cada una cuenta con 4 corridas y un cierto número de individuos y generaciones. Para cada conjunto se analizarón aquellas soluciones que sean únicas, y se seleccionaran solo las que cumplan con el objetivo de tiempo de llegada en la mayoría de las condiciones iniciales. De estas soluciones se eligen las que tengan el mejor desempeño y se realiza la simulación para estás. Después se selecciona la mejor solución sobre las demás. Finalmente se realiza una comparación entre el sistema cuando cuenta con un comportamiento aprendido y cuando no. Además,se analizán las estadísticas del conjunto de ejecuciones y las posibles extensiones de la metodología de comportamientos analíticos aplicados a sistemas multirobot.

4.1 Resultados del proceso evolutivo

	Valor
Soluciones encontradas aproximadas	538,113
Soluciones no repetidas y con un buen desempeño	1,835
Soluciones seleccionadas con el mejor desempeño de todas las corridas analizadas	13
Tiempo máximo de ejecución con las características de la Workstation de la tabla(3.6)	9 dias

Tabla 4.1: Resultados generales de las evoluciones.

En general se obtuvieron un aproximado de 538,113 soluciones, de las cuales 1,835 son soluciones únicas que presentarón un buen desempeño, seleccionando a las 13 mejores de todo el conjunto. Como se menciona en la tabla(4.1) el tiempo máximo de ejecución aproximado fue de nueve dias. Las 13 soluciones seleccionadas se presentan en la tabla(4.2), siendo estas soluciones los mejores comportamientos encontrados por el proceso evolutivo, o al menos las que cumplen con la tarea de **encuentro** para la mayoría de las condiciones iniciales. Dentro de la tabla, la solución u_{l_1} representa la mejor de las mejores, u_f^i representa el comportamiento forzado y su desempeño para la tarea de **encuentro** en las condiciones inciales usadas en el aprendizaje.

Como resultado del proceso de simulación se diseño una tabla comparativa entre el tiempo en que los robots llegan a la meta, para el sistema cuando se usa comportamiento aprendido y el comportamiento forzado. En la tabla(4.3) se aprecia la diferencia entre la navegación con el comportamiento aprendido y el comportamiento forzado. La simulación esta dada para las diez condiciones iniciales y permite estipular que; "el sistema si aprendio a cumplir con la tarea dada".

En la tabla(4.3), se puede notar que cuando no se utiliza el comportamiento aprendido, en algunas condiciones iniciales el tiempo de llegada esta lejos del tiempo deseado, esto se debe a que el comportamiento forzado solo toma en cuenta la nevegación y la evasión del otro robot, por lo que su dinámica no se ve forzada a cumplir con el objetivo de tiempo.

Por otro lado, cuando se incorpora el comportamiento aprendido se nota que ambos robots se ajustan al tiempo deseado o al menos estan dentro de la tolerancia permitida. Comparando ambas situaciones, se puede apreciar en el tiempo promedio que el ajuste de la solución si forza a los robots a cumplir con el **encuentro** para el tiempo entrenado de 4s, esto quiere decir, que independientemente del tiempo de simulación los robots siempre van a llegar en el tiempo que se entrenó el sistema.

Como complemento de la validación númerica del objetivo de control se pueden analizar la figura (4.1) hasta la figura(4.61), las cuales representan los resultados de la simulación comparando los sistemas cuando cuentan con comportamiento aprendido y cuando no cuentan con dicho comportamiento. Al analizar las trayectorias, que van de la figura(4.1) hasta la figura(4.10) para las diez condiciones iniciales usadas en el aprendizaje, claramente se puede apreciar como

Tabla 4.2: Soluciones óptimas u_{l_k}, $k \in \{1,...,13\}$ encontradas por el proceso evolutivo.

Sol	Expresión	Desempeño
u_{l_1}	$\big(\big(\mathrm{at2r}\big(c\big(\ln\big(\big(\mathrm{at2r}\big(s_i\big(s_i\big(\|t\big(t_c^1\big)\|\big)\big),\mathrm{at2r}\big(\dot{x}^1,\ln\big(y^1\big)\big)\big)+\mathrm{at2r}\big(sc_h(t_c^1),\mathrm{at2r}\big(\dot{x}^1,\big(\big(t\big(t_c^1\big)t\big(\dot{x}^1\big)\big)\dot{x}^1\big)\big)\big)\big)\big),$ $\mathrm{at2r}\big(\dot{x}^1,\big(\big(\dot{y}^1\|t_c^1\|\big)\|\big(\big(\big(\dot{x}^1\,t(t_c^1)\big)+\big(\max\big(\big(sc_h\big(\dot{x}^1\big)+t(t_c^1)\big),\big(\dot{x}^i\,t(t_c^1)\big)\big)+s_i\big(\big(\dot{x}^1+t\big(t_c^1\big)\big)\big)\big)\big)\|\big)\big)\big)$ $+\mathrm{at2r}\big(\dot{x}^1,\ln\big(c\big(\big(\max\big(s_i\big(s_i\big(\|y^1\|\big)\big),\max\big(\big(t\big(t_c^1\big)t\big(\min\big(y^1,\dot{x}^1\big)\big)\big),\dot{x}^1\big)\big)+s_i\big(\big(sc_h(t_c^1)+t\big(t_c^1\big)\big)\big)\big)\big)+t_c^1\big)$	0.07088
u_{l_2}	$\mathrm{at2r}\big(c\big(\ln\big(\mathrm{at2r}\big(s_i\big(s_i\big(\|t\big(t_c^1\big)\|\big)\big),\mathrm{at2r}\big(\dot{x}^1,\ln\big(y^1\big)\big)\big)+\mathrm{at2r}\big(sc_h\big(t_c^1\big),\mathrm{at2r}\big(\dot{x}^1,\big(\dot{x}^1t\big(t_c^1\big)t(\dot{x}^1\big)\big)\big)\big)\big),$ $\mathrm{at2r}\big(\dot{x}^1,\big(\dot{x}^1\|\|t_c^1\|\|\|\big(\big(\dot{x}^1\,t\big(t_c^1\big)\big)+\big(\max\big(\big(sc_h\big(\dot{x}^1\big)+t\big(t_c^1\big)\big),\dot{x}^1t\big(t_c^1\big)\big)+s_i\big(\dot{x}^1+t\big(t_c^1\big)\big)\big)\big)\|\big)\big)$ $+\mathrm{at2r}\big(\dot{x}^1,\ln\big(c\big(\max\big(s_i\big(s_i\big(\|y^1\|\big)\big),\max\big(t\big(t_c^1\big)t\big(\min\big(y^1,\dot{x}^1\big)\big),\dot{x}^1\big)\big)+s_i\big(sc_h\big(t_c^1\big)+t\big(t_c^1\big)\big)\big)\big)+t_c^1$	0.07088
u_{l_3}	$\mathrm{at2r}\big(c\big(\ln\big(\mathrm{at2r}\big(s_i\big(s_i\big(\|t\big(t_c^1\big)\|\big)\big),\mathrm{at2r}\big(\dot{x}^1,\ln\big(t_c^1\big)\big)\big)+\mathrm{at2r}\big(sc_h\big(t_c^1\big),\mathrm{at2r}\big(\dot{x}^1,\dot{x}^1t\big(t_c^1\big)t\big(\dot{x}^1\big)\big)\big)\big),$ $\mathrm{at2r}\big(\dot{x}^1,\dot{y}^1\|t_c^1\|\|\dot{x}^1t\big(t_c^1\big)+\max\big(sc_h\big(\dot{x}^1\big)+t\big(t_c^1\big),\dot{x}^1t\big(t_c^1\big)\big)+s_i\big(\dot{x}^1+t\big(t_c^1\big)\big)\|\big)\big)$ $+\mathrm{at2r}\big(\dot{x}^1,\ln\big(c\big(\max\big(s_i\big(s_i\big(\|y^1\|\big)\big),\max\big(t\big(t_c^1\big)t\big(\min\big(y^1,\dot{x}^1\big)\big),\mathrm{abs}\big(t\big(\dot{x}^1\big)\big)\big)+s_i\big(sc_h\big(t_c^1\big)+t\big(t_c^1\big)\big)\big)\big)+$ t_c^1	0.07091
u_{l_4}	$\mathrm{at2r}\big(c\big(\ln\big(\mathrm{at2r}\big(s_i\big(s_i\big(\|t\big(t_c^1\big)\|\big)\big),at2r\big(\dot{x}^1,\ln\big(y^1\big)\big)\big)+\mathrm{at2r}\big(sc_h\big(t_c^1\big),\mathrm{at2r}\big(\dot{x}^1,\dot{x}^1t\big(t_c^1\big)t\big(\dot{x}^1\big)\big)\big)\big),$ $\mathrm{at2r}\big(\dot{x}^1,\dot{y}^1\|t_c^1\|\|\dot{x}^1t\big(t_c^1\big)+\big(\max\big(sc_h\big(\dot{x}^1\big)+t\big(t_c^1\big),\mathrm{abs}\big(\dot{x}^1\big)\big)+s_i\big(\dot{x}^1+t\big(t_c^1\big)\big)\big)\|\big)\big)$ $+\mathrm{at2r}\big(\dot{x}^1,\ln\big(c\big(\max\big(s_i\big(s_i\big(\|y^1\|\big)\big),\max\big(t\big(t_c^1\big)t\big(\min\big(x_d^2,\dot{x}^1\big)\big),\mathrm{abs}\big(t\big(\dot{x}^1\big)\big)\big)+s_i\big(sc_h\big(t_c^1\big)+t\big(t_c^1\big)\big)\big)\big)+$ t_c^1	0.07102
u_{l_5}	$\mathrm{at2r}\big(c\big(\ln\big(\mathrm{at2r}\big(s_i\big(s_i\big(\|t\big(t_c^1\big)\|\big)\big),\mathrm{at2r}\big(\min\big(\dot{x}^1,\dot{x}^1\big),\ln\big(y^1\big)\big)\big)+\mathrm{at2r}\big(sc_h\big(t_c^1\big),\mathrm{at2r}\big(\dot{x}^1,\dot{x}^1t\big(t_c^1\big)t\big(\dot{x}^1\big)\big)\big)\big),$ $\mathrm{at2r}\big(\dot{x}^1,\dot{y}^1\|t_c^1\|\|\dot{x}^1t\big(t_c^1\big)+\max\big(sc_h\big(\dot{x}^1\big)+t\big(t_c^1\big),\mathrm{abs}\big(\dot{x}^1\big)\big)+s_i\big(\dot{x}^1+t\big(t_c^1\big)\big)\|\big)\big)$ $+\mathrm{at2r}\big(\dot{x}^1,\ln\big(c\big(\max\big(s_i\big(s_i\big(\|y^1\|\big)\big),\max\big(t\big(t_c^1\big)t\big(\min\big(x_d^2,\dot{x}^1\big)\big),\mathrm{abs}\big(t\big(\dot{x}^1\big)\big)\big)+s_i\big(sc_h\big(t_c^1\big)+t\big(t_c^1\big)\big)\big)\big)+$ t_c^1	0.07102
u_{l_6}	$\mathrm{at2r}\big(c\big(\ln\big(\mathrm{at2r}\big(s_i\big(s_i\big(\|t\big(t_c^1\big)\|\big)\big),\mathrm{at2r}\big(\min\big(x_d^2,\dot{x}^1\big),\ln\big(y^1\big)\big)\big)+\mathrm{at2r}\big(sc_h\big(t_c^1\big),\mathrm{at2r}\big(\dot{x}^1,\dot{x}^1t\big(t_c^1\big)t\big(\dot{x}^1\big)\big)\big)\big),$ $\mathrm{at2r}\big(\dot{x}^1,\dot{y}^1\|t_c^1\|\|\dot{x}^1t\big(t_c^1\big)+\max\big(sc_h\big(\dot{x}^1\big)+t\big(t_c^1\big),\mathrm{abs}\big(\dot{x}^1\big)\big)+s_i\big(\dot{x}^1+t\big(t_c^1\big)\big)\|\big)\big)$ $+\mathrm{at2r}\big(\dot{x}^1,\ln\big(c\big(\max\big(s_i\big(s_i\big(\|y^1\|\big)\big),\max\big(t\big(t_c^1\big)t\big(\dot{x}^1\big),\mathrm{abs}\big(t\big(\dot{x}^1\big)\big)\big)+s_i\big(sc_h\big(t_c^1\big)+t\big(t_c^1\big)\big)\big)\big)+t_c^1$	0.07111
u_{l_7}	$\mathrm{at2r}\big(c\big(\ln\big(\mathrm{at2r}\big(s_i\big(s_i\big(\|t\big(t_c^1\big)\|\big)\big),\mathrm{at2r}\big(\dot{x}^1,\ln\big(y^1\big)\big)\big)+\mathrm{at2r}\big(sc_h\big(t_c^1\big),\mathrm{at2r}\big(\dot{x}^1,\dot{x}^1t\big(t_c^1\big)t\big(\dot{x}^1\big)\big)\big)\big),$ $\mathrm{at2r}\big(\dot{x}^1,\dot{y}^1\|t_c^1\|\|\dot{x}^1t\big(t_c^1\big)+\max\big(sc_h\big(\dot{x}^1\big)+t\big(t_c^1\big),\mathrm{abs}\big(\dot{x}^1\big)\big)+s_i\big(\dot{x}^1+t\big(t_c^1\big)\big)\|\big)\big)$ $+\mathrm{at2r}\big(\dot{x}^1,\ln\big(c\big(\max\big(s_i\big(s_i\big(\|y^1\|\big)\big),\max\big(t\big(t_c^1\big)t\big(\dot{x}^1\big),\mathrm{abs}\big(t\big(\dot{x}^1\big)\big)\big)+s_i\big(sc_h\big(t_c^1\big)+t\big(t_c^1\big)\big)\big)\big)+t_c^1$	0.07115
u_{l_8}	$\mathrm{at2r}\big(c\big(\ln\big(\mathrm{at2r}\big(s_i\big(s_i\big(\|t\big(t_c^1\big)\|\big)\big),\mathrm{at2r}\big(\dot{x}^1,\ln\big(y^1\big)\big)\big)+\mathrm{at2r}\big(sc_h\big(t_c^1\big),\mathrm{at2r}\big(\dot{x}^1,\dot{x}^1t\big(t_c^1\big)t\big(\dot{x}^1\big)\big)\big)\big),$ $\mathrm{at2r}\big(\dot{x}^1,\dot{y}^1\|t_c^1\|\|\dot{x}^1t\big(t_c^1\big)+\max\big(sc_h\big(\dot{x}^1\big)+t\big(t_c^1\big),\mathrm{abs}\big(\dot{x}^1\big)\big)+s_i\big(sc_h\big(t_c^1\big)+t\big(t_c^1\big)\big)\|\big)\big)$ $+\mathrm{at2r}\big(\dot{x}^1,\ln\big(c\big(\max\big(s_i\big(\|y^1\|\big),\max\big(t\big(t_c^1\big)t\big(\dot{x}^1\big)t\big(\dot{x}^1\big),\mathrm{abs}\big(t\big(\dot{x}^1\big)\big)\big)+s_i\big(sc_h\big(t_c^1\big)+t\big(t_c^1\big)\big)\big)\big)+t_c^1$	0.07188
u_{l_9}	$\mathrm{at2r}\big(c\big(\ln\big(\mathrm{at2r}\big(s_i\big(s_i\big(\|t\big(t_c^1\big)\|\big)\big),\mathrm{at2r}\big(\dot{x}^1,\ln\big(y^1\big)\big)\big)+\mathrm{at2r}\big(sc_h\big(t_c^1\big),\mathrm{at2r}\big(\dot{x}^1,\dot{x}^1t\big(t_c^1\big)t\big(\dot{x}^1\big)\big)\big)\big),$ $\mathrm{at2r}\big(\dot{x}^1,\dot{y}^1\|t_c^1\|\|t\big(t_c^1\big)t\big(\dot{x}^1\big)+\max\big(sc_h\big(\dot{x}^1\big)+t\big(t_c^1\big),\mathrm{abs}\big(\dot{x}^1\big)\big)+s_i\big(sc_h\big(t_c^1\big)+t\big(t_c^1\big)\big)\|\big)\big)$ $+\mathrm{at2r}\big(\dot{x}^1,\ln\big(c\big(\max\big(s_i\big(\|y^1\|\big),\max\big(t\big(t_c^1\big)t\big(\dot{x}^1\big)t\big(\dot{x}^1\big),\mathrm{abs}\big(y^1\big)\big)\big)+s_i\big(sc_h\big(t_c^1\big)+t\big(t_c^1\big)\big)\big)\big)+t_c^1$	0.07199
$u_{l_{10}}$	$\mathrm{at2r}\big(c\big(\ln\big(\mathrm{at2r}\big(s_i\big(s_i\big(\|t\big(t_c^1\big)\|\big)\big),\mathrm{at2r}\big(\dot{x}^1,\ln\big(y^1\big)\big)\big)+\mathrm{at2r}\big(sc_h\big(t_c^1\big),\mathrm{at2r}\big(\dot{x}^1,\dot{x}^1t\big(t_c^1\big)t\big(\dot{x}^1\big)\big)\big)\big),$ $\mathrm{at2r}\big(\dot{x}^1,\dot{y}^1\|t_c^1\|\|t\big(t_c^1\big)t\big(\dot{x}^1\big)+\max\big(sc_h\big(\dot{x}^1\big)+t\big(t_c^1\big),\mathrm{abs}\big(atr\big(y^1\big)\big)\big)+s_i\big(sc_h\big(t_c^1\big)+t\big(t_c^1\big)\big)\|\big)\big)$ $+\mathrm{at2r}\big(\dot{x}^1,\ln\big(c\big(\max\big(s_i\big(\|y^1\|\big),\max\big(\dot{x}^1t\big(t_c^1\big)t\big(\dot{x}^1\big),\mathrm{abs}\big(y^1\big)\big)\big)+s_i\big(sc_h\big(t_c^1\big)+t\big(t_c^1\big)\big)\big)\big)+t_c^1$	0.07264
$u_{l_{11}}$	$\mathrm{at2r}\big(c\big(\ln\big(\mathrm{at2r}\big(s_i\big(s_i\big(\|t\big(t_c^1\big)\|\big)\big),\mathrm{at2r}\big(\dot{x}^1,\ln\big(s_i\big(\|y^1\|\big)\big)\big)\big)+\mathrm{at2r}\big(sc_h\big(t_c^1\big),\mathrm{at2r}\big(\dot{x}^1,\dot{x}^1t\big(t_c^1\big)t\big(\dot{x}^1\big)\big)\big)\big),$ $\mathrm{at2r}\big(\dot{x}^1,\dot{y}^1\|t_c^1\|\|t\big(t_c^1\big)t\big(\dot{x}^1\big)+\max\big(sc_h\big(t_c^1\big)+t\big(t_c^1\big),\mathrm{abs}\big(\mathrm{atr}\big(y^1\big)\big)\big)+s_i\big(sc_h\big(t_c^1\big)+t\big(t_c^1\big)\big)\|\big)\big)$ $+\mathrm{at2r}\big(\dot{x}^1,\ln\big(c\big(\max\big(\dot{x}^1,\max\big(t\big(t_c^1\big)t\big(\dot{x}^1\big)sc_h\big(t_c^1\big),\mathrm{abs}\big(y^1\big)\big)\big)+s_i\big(sc_h\big(t_c^1\big)+t\big(t_c^1\big)\big)\big)\big)+t_c^1$	0.07402
$u_{l_{12}}$	$\mathrm{at2r}\big(c\big(\ln\big(\mathrm{at2r}\big(s_i\big(s_i\big(\|t\big(t_c^1\big)\|\big)\big),\mathrm{at2r}\big(\dot{x}^1,\ln\big(y^1\big)\big)\big)+\mathrm{at2r}\big(sc_h\big(t_c^1\big),\mathrm{at2r}\big(\dot{x}^1,\dot{x}^1t\big(t_c^1\big)t\big(\dot{x}^1\big)\big)\big)\big),$ $\mathrm{at2r}\big(\dot{x}^1,\dot{y}^1\|t_c^1\|\|\dot{x}^1t\big(t_c^1\big)+y^1+s_i\big(sc_h\big(t_c^1\big)+t\big(t_c^1\big)\big)\|\big)\big)$ $+\mathrm{at2r}\big(\dot{x}^1,\ln\big(c\big(s_i\big(\max\big(\dot{x}^1,\max\big(\dot{x}^1,\mathrm{abs}\big(y^1\big)\big)\big)+s_i\big(sc_h\big(t_c^1\big)+t\big(t_c^1\big)\big)\big)\big)\big)+t_c^1$	0.07843
$u_{l_{13}}$	$\mathrm{at2r}\big(c\big(\ln\big(\mathrm{at2r}\big(s_i\big(s_i\big(\|t\big(t_c^1\big)\|\big)\big),\mathrm{at2r}\big(\dot{x}^1,\ln\big(s_i\big(\|y^1\|\big)\big)\big)\big)+\mathrm{at2r}\big(sc_h\big(t_c^1\big),\mathrm{at2r}\big(\dot{x}^1,\dot{x}^1t\big(t_c^1\big)t\big(\dot{x}^1\big)\big)\big)\big),$ $\mathrm{at2r}\big(\dot{x}^1,\dot{y}^1\|t_c^1\|\|\dot{x}^1t\big(t_c^1\big)+\max\big(\dot{x}^1,\mathrm{abs}\big(\mathrm{atr}\big(y^1\big)\big)\big)+s_i\big(sc_h\big(t_c^1\big)+t\big(t_c^1\big)\big)\|\big)\big)$ $+\mathrm{at2r}\big(\dot{x}^1,\ln\big(c\big(s_i\big(\max\big(\dot{x}^1,\max\big(\dot{x}^1,\mathrm{abs}\big(y^1\big)\big)\big)+s_i\big(sc_h\big(t_c^1\big)+t\big(t_c^1\big)\big)\big)\big)\big)+t_c^1$	0.07863
u_f^i	$K^i\big(q_d^i-q^i\big)+\mathrm{Re}\Big(\ln(\mathrm{abs}(\ln(\cos(\big(\ln\big(\|q^i-d^{i\hat{o}^\beta}\|\big)\big)))))\Big)\times\Big(\ln\Big(\|\,q^i-d^{i\hat{o}^\beta}\,\|\Big)\Big)$	2.4255

afecta a la trayectoria el incluir un comportamiento nuevo (**comportamiento aprendido**) al sistema.

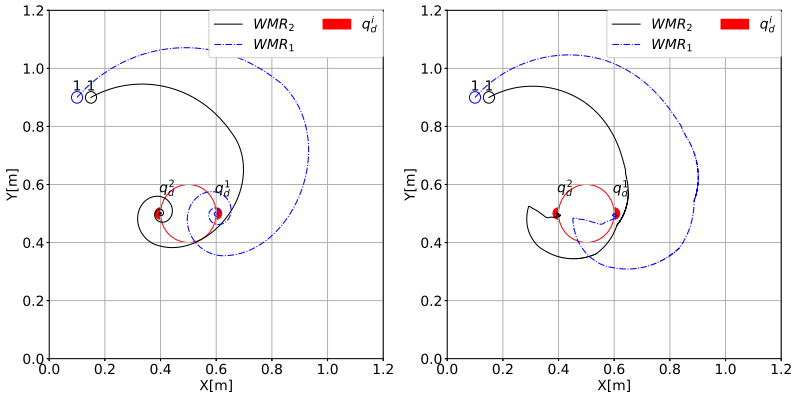
Tabla 4.3: Entrenamiento: Tiempo de llegada para los diez escenarios selectos del ejemplo de aplicación.

Entrenamiento: Tiempo de llegada para la tarea de **encuentro**, t_a= 4 segundos.				
	$u^i = u^i_f$		$u^i = u^i_f + u_{l_1}$	
α	WMR_1	WMR_2	WMR_1	WMR_2
1	3.8619	3.7869	3.9770	3.9770
2	3.0829	3.3279	3.9350	3.9350
3	3.2459	3.9269	4.3060	3.8760
4	2.7959	2.1199	4.1780	3.6730
5	3.3919	2.8999	4.2820	3.7890
6	2.2629	3.3069	3.8550	4.3110
7	3.0749	2.1059	4.0140	3.9420
8	3.6929	2.9929	4.2050	3.8090
9	3.0089	2.0569	4.1380	3.9250
10	3.3389	2.7679	4.2660	3.8210
Avg	3.1757	2.9292	4.1156	3.9058

Por otro lado, en las figura(4.11), hasta la figura(4.20) se aprecia el comportamiendo de los robots a lo largo del eje X, comparando el desplazamiento respecto al tiempo de las situaciones cuando cuenta con comportamiento aprendido y cuando no cuenta con el. A su vez en las figuras(4.21) hasta (4.30) se hace la misma comparación, solo que ahora respecto al eje Y. Para el análisis de las figutas(4.31) hasta (4.50) representan las velocidades en los ejes X y Y para ambos robots, como se puede apreciar en dichas figuras la dinámica de velocidad cambia para ajustarse al objetivo de tiempo, resultado más que evidente cuando cuenta con comportamiento aprendido y cuando no cuenta con él. Ahora analizando las figuras(4.51) hasta (4.60) se puede apreciar como ambos robots siempre llegan a la meta, debido a que el error tiende a cero conforme estos se acercan, siendo parte esta caracteristica del comportamiento forzado. Por ultimo analizamos las figura(4.61), donde claramente se puede ver la diferencia entre los tiempos de cada condición inicial y como se establecio en la tablas(4.3), los tiempos de llegada son muy diferentes para cada condición inicial en la comparación entre el sistema multirobot cuando cuenta con el comportamiento aprendido y cuando no cuenta con él.

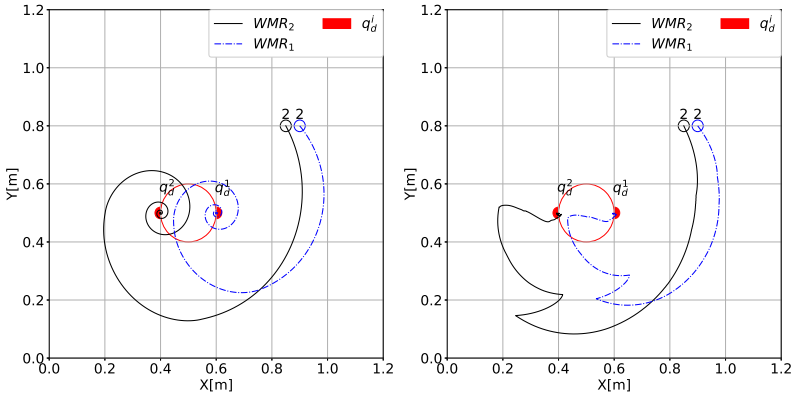
4.1.0.1 *Trayectorias de los WMR's*

En esta sección se representan las trayectorias para las diez condiciones iniciales usadas en el aprendizaje para los $WMR1$ y $WMR2$.



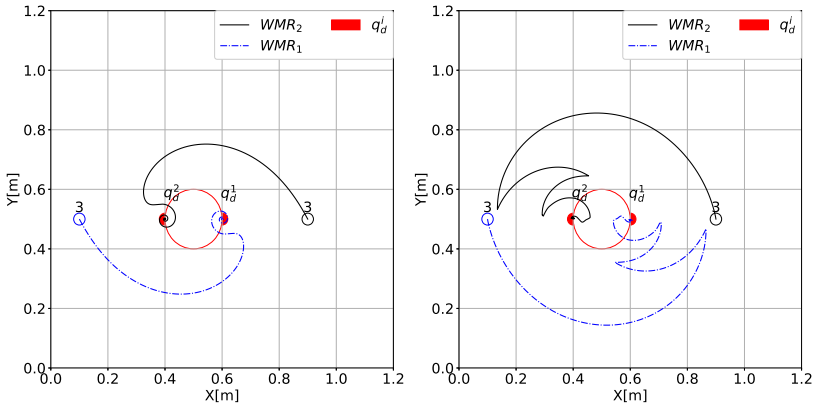
(a) (b)

Figura 4.1: En el escenario (a) se muestra la trayectoria sin controlador, mientras que el escenario (b) es la trayectoria con el controlador u_{l_1}, en estos casos el desempeño para (a) fue de 0,0645 y para (b) de 0,0011 en la condición inicial #1.



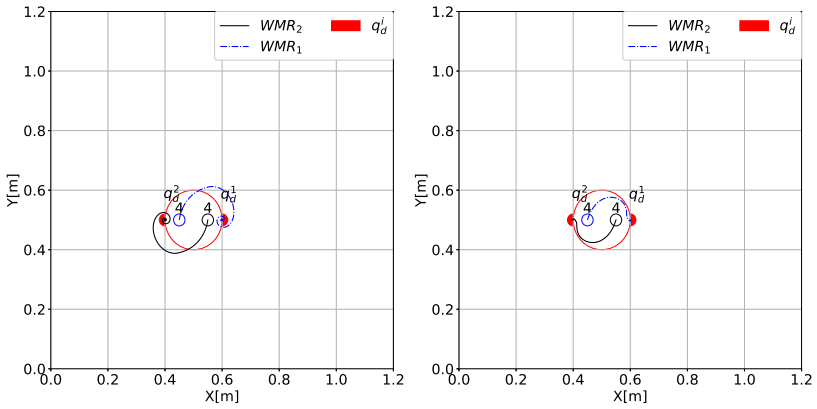
(a) (b)

Figura 4.2: En el escenario (a) se muestra la trayectoria sin controlador, mientras que el escenario (b) es la trayectoria con el controlador u_{l_1}, en estos casos el desempeño para (a) fue de 1,2925 y para (b) de 0,0085 en la condición inicial #2.



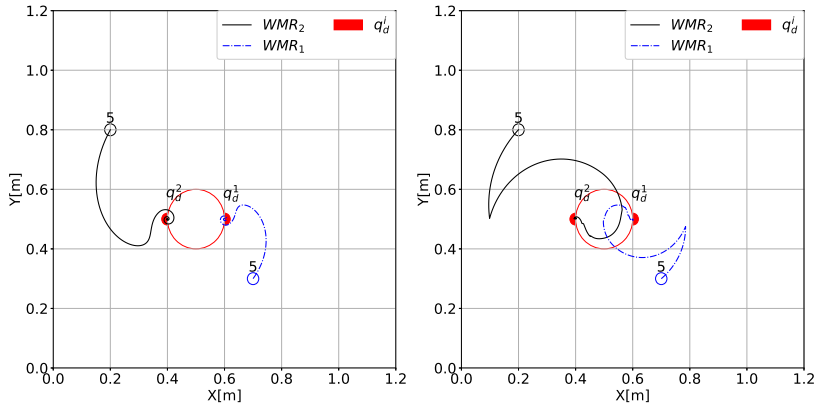
(a) (b)

Figura 4.3: En el escenario (a) se muestra la trayectoria sin controlador, mientras que el escenario (b) es la trayectoria con el controlador u_{l_1}, en estos casos el desempeño para (a) fue de 0,5739 y para (b) de 0,1091 en la condición inicial #3.



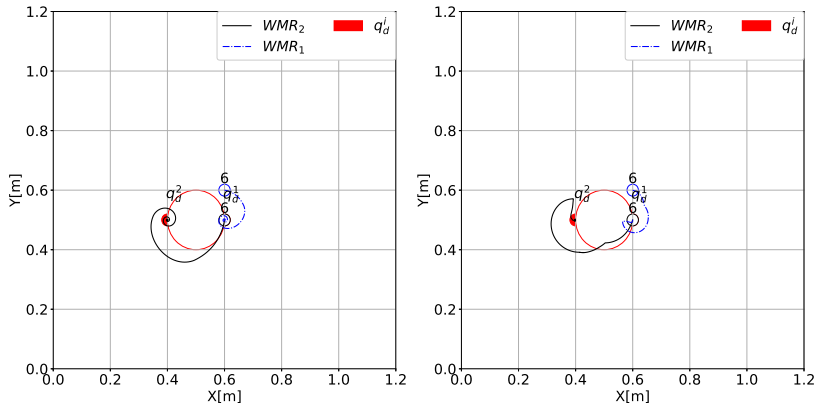
(a) (b)

Figura 4.4: En el escenario (a) se muestra la trayectoria sin controlador, mientras que el escenario (b) es la trayectoria con el controlador u_{l_1}, en estos casos el desempeño para (a) fue de 4,9840 y para (b) de 0,1386 en la condición inicial #4.



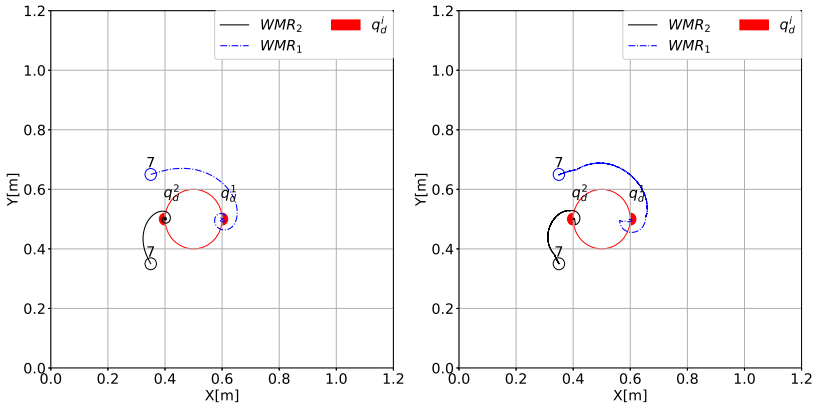
(a) (b)

Figura 4.5: En el escenario (a) se muestra la trayectoria sin controlador, mientras que el escenario (b) es la trayectoria con el controlador u_{l_1}, en estos casos el desempeño para (a) fue de 1,5797 y para (b) de 0,1241 en la condición inicial #5.



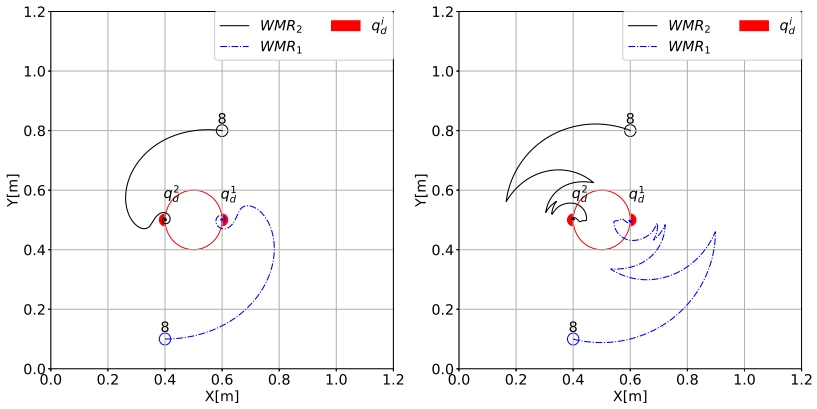
(a) (b)

Figura 4.6: En el escenario (a) se muestra la trayectoria sin controlador, mientras que el escenario (b) es la trayectoria con el controlador u_{l_1}, en estos casos el desempeño para (a) fue de 3,4974 y para (b) de 0,1178 en la condición inicial #6.



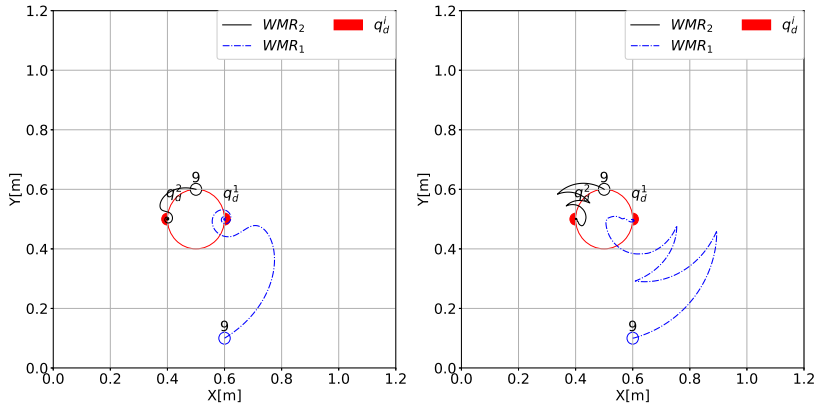
(a) (b)

Figura 4.7: En el escenario (a) se muestra la trayectoria sin controlador, mientras que el escenario (b) es la trayectoria con el controlador u_{l_1}, en estos casos el desempeño para (a) fue de 4,4429 y para (b) de 0,0036 en la condición inicial #7.



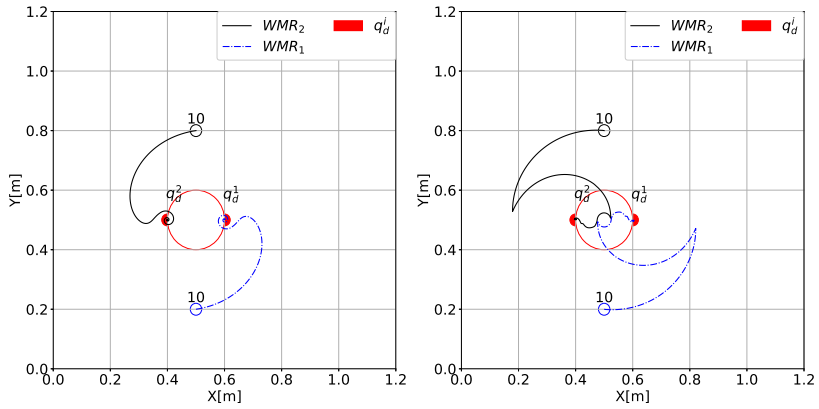
(a) (b)

Figura 4.8: En el escenario (a) se muestra la trayectoria sin controlador, mientras que el escenario (b) es la trayectoria con el controlador u_{l_1}, en estos casos el desempeño para (a) fue de 1,1083 y para (b) de 0,0785 en la condición inicial #8.



(a) (b)

Figura 4.9: En el escenario (a) se muestra la trayectoria sin controlador, mientras que el escenario (b) es la trayectoria con el controlador u_{l_1}, en estos casos el desempeño para (a) fue de 4,7574 y para (b) de 0,0247 en la condición inicial #9.

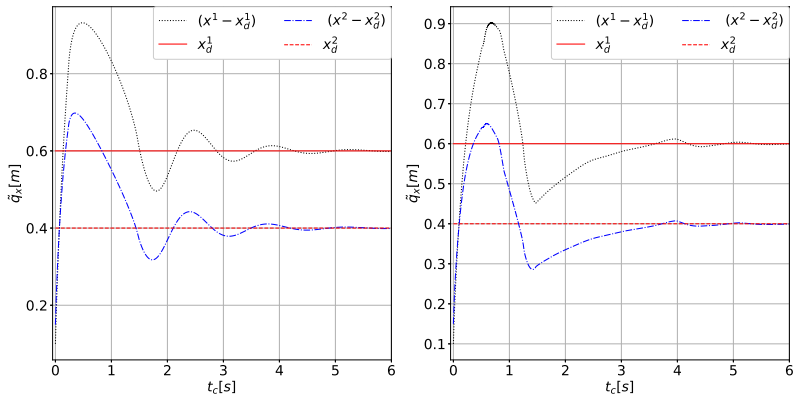


(a) (b)

Figura 4.10: En el escenario (a) se muestra la trayectoria sin controlador, mientras que el escenario (b) es la trayectoria con el controlador u_{l_1}, en estos casos el desempeño para (a) fue de 1,9548 y para (b) de 0,1028 en la condición inicial #10.

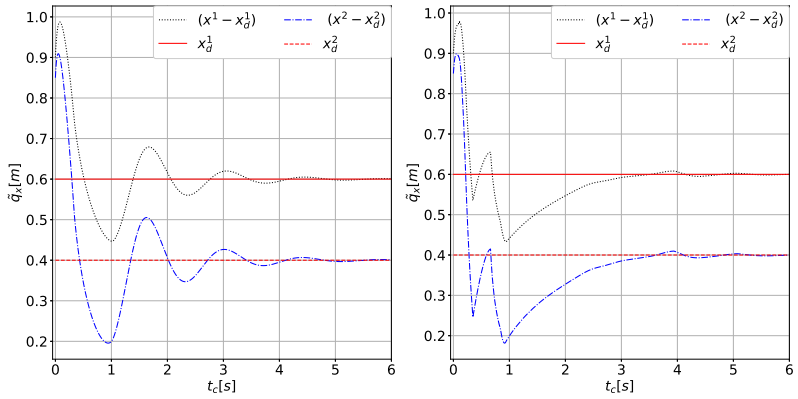
4.1.0.2 *Errores de posición en el eje coordenado X*

En esta sección se representá el error de posición para el eje x de las diez condiciones iniciales usadas en el aprendizaje y tomadas de la tabla(3.4) para los $WMR1$ y $WMR2$.



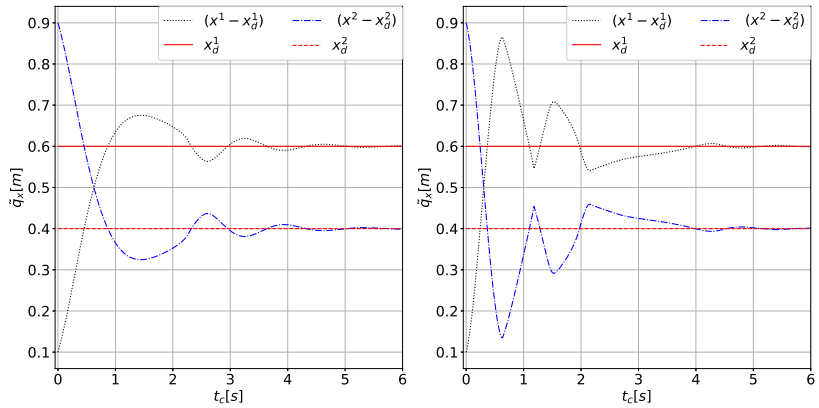
(a) (b)

Figura 4.11: En el escenario (a) se muestra el error sin controlador para el $WMR1$ y $WMR2$, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #1.



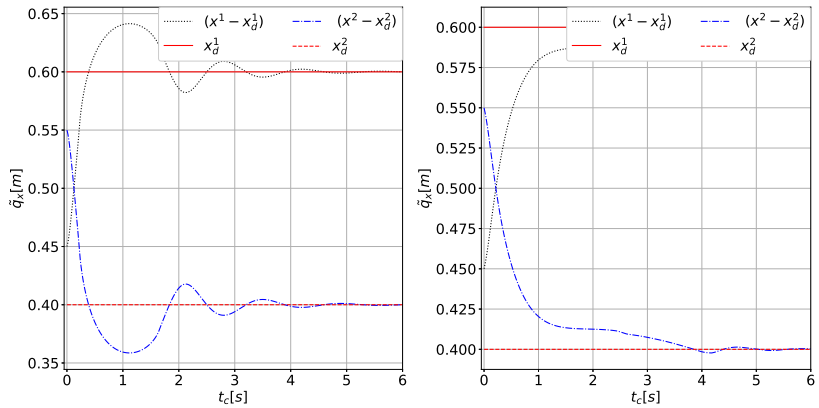
(a) (b)

Figura 4.12: En el escenario (a) se muestra el error sin controlador para el $WMR1$ y $WMR2$, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #2.



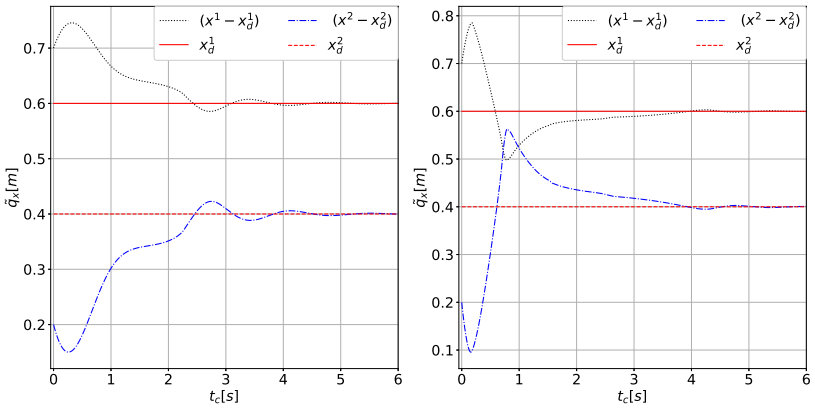
(a) (b)

Figura 4.13: En el escenario (a) se muestra el error sin controlador para el $WMR1$ y $WMR2$, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #3.



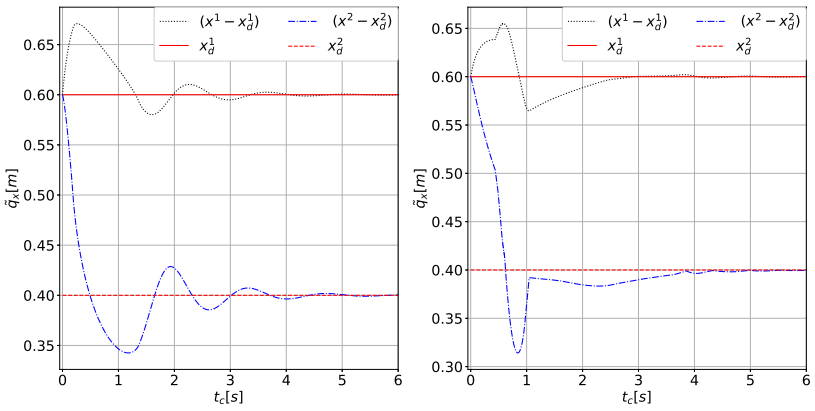
(a) (b)

Figura 4.14: En el escenario (a) se muestra el error sin controlador para el $WMR1$ y $WMR2$, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #4.



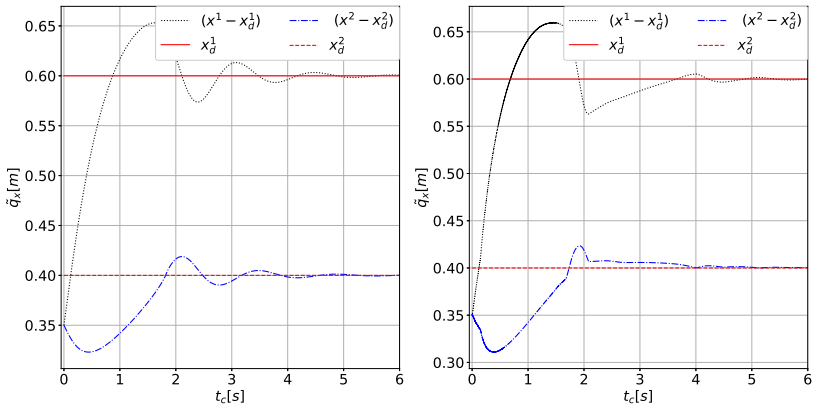
(a) (b)

Figura 4.15: En el escenario (a) se muestra el error sin controlador para el $WMR1$ y $WMR2$, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #5.



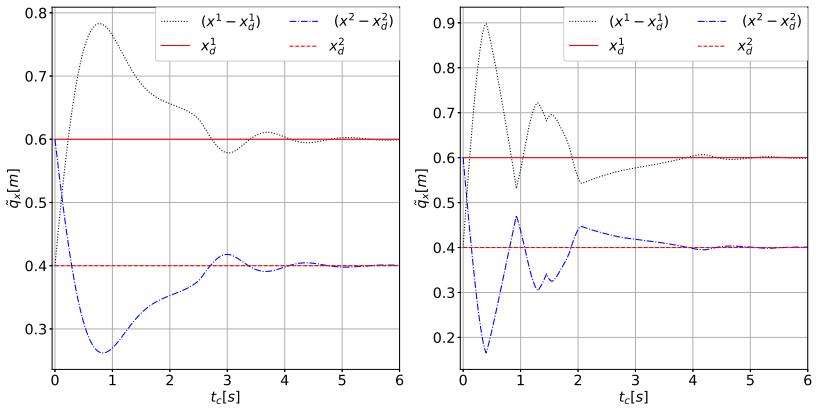
(a) (b)

Figura 4.16: En el escenario (a) se muestra el error sin controlador para el $WMR1$ y $WMR2$, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #6.



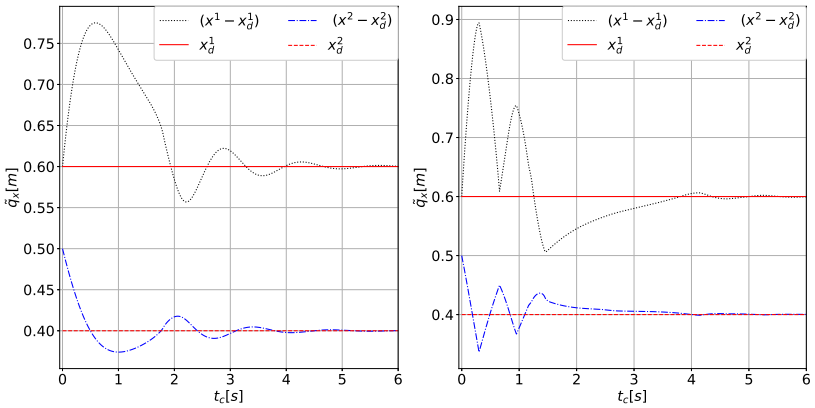
(a) (b)

Figura 4.17: En el escenario (a) se muestra el error sin controlador para el $WMR1$ y $WMR2$, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #7.



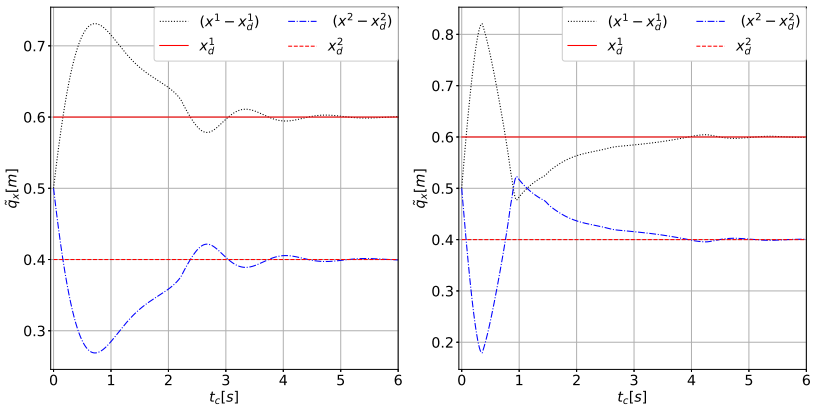
(a) (b)

Figura 4.18: En el escenario (a) se muestra el error sin controlador para el $WMR1$ y $WMR2$, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #8.



(a) (b)

Figura 4.19: En el escenario (a) se muestra el error sin controlador para el $WMR1$ y $WMR2$, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #9.

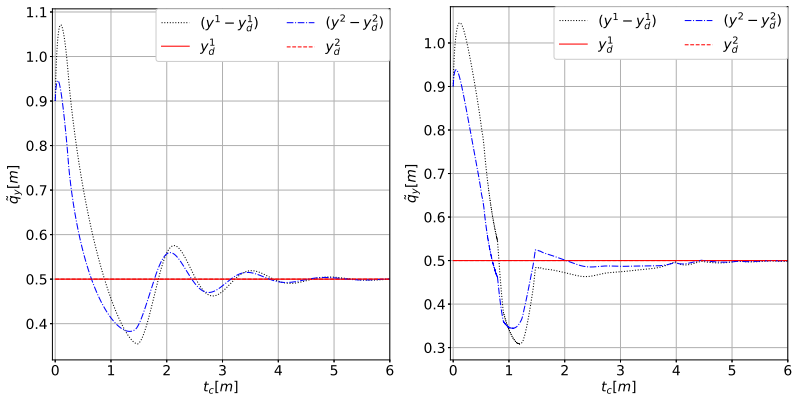


(a) (b)

Figura 4.20: En el escenario (a) se muestra el error sin controlador para el $WMR1$ y $WMR2$, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #10.

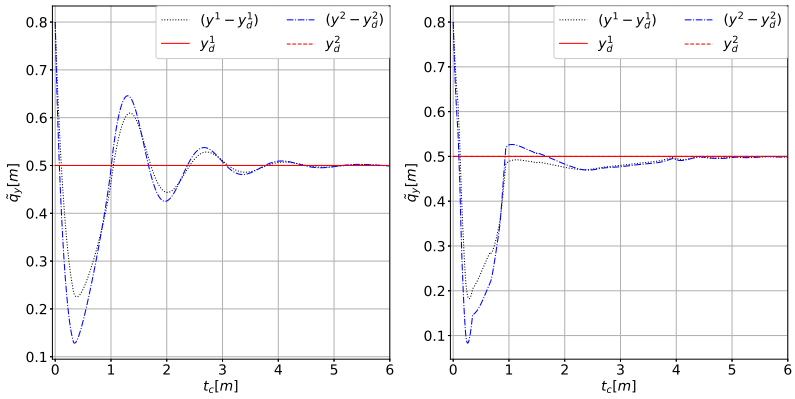
4.1.0.3 *Errores de posición en el eje coordenado Y*

En esta sección se representa el error de posición en el eje y, paraa las condiciones iniciales de la tabla(3.4), se consideran dos robots móviles nombrados $WMR1$ y $WMR2$.



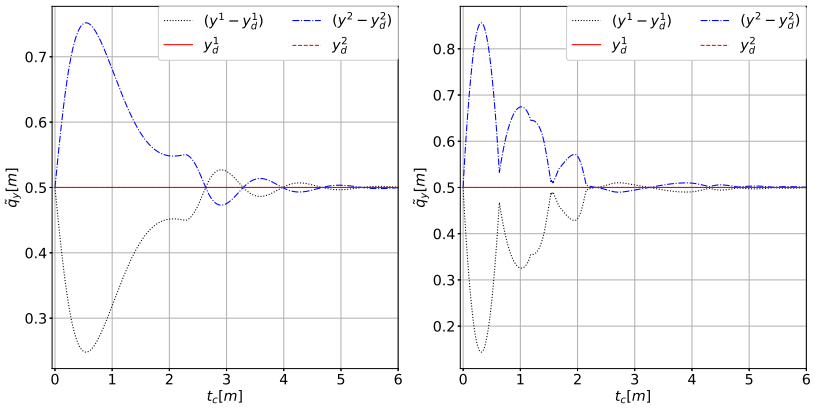
(a) (b)

Figura 4.21: En el escenario (a) se muestra el error sin controlador, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #1.



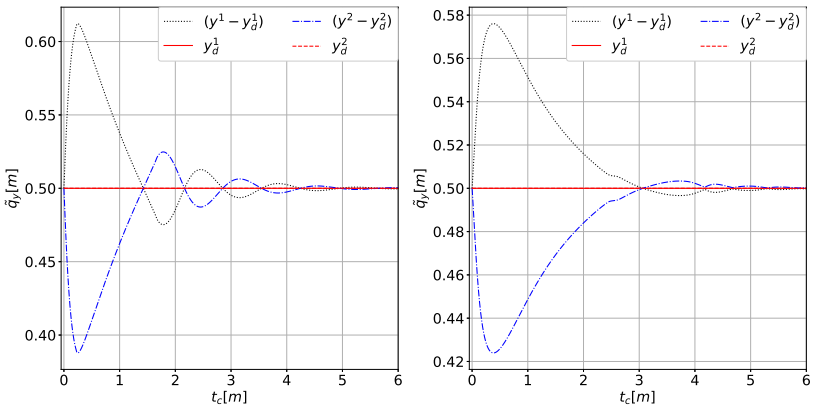
(a) (b)

Figura 4.22: En el escenario (a) se muestra el error sin controlador, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #2.



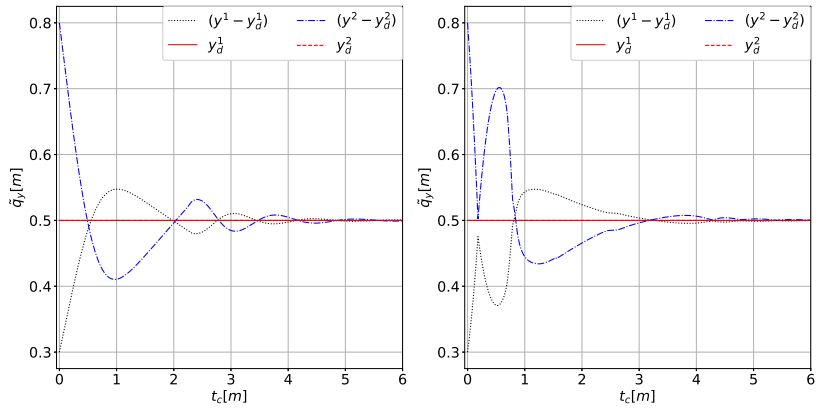
(a) (b)

Figura 4.23: En el escenario (a) se muestra el error sin controlador, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #3.



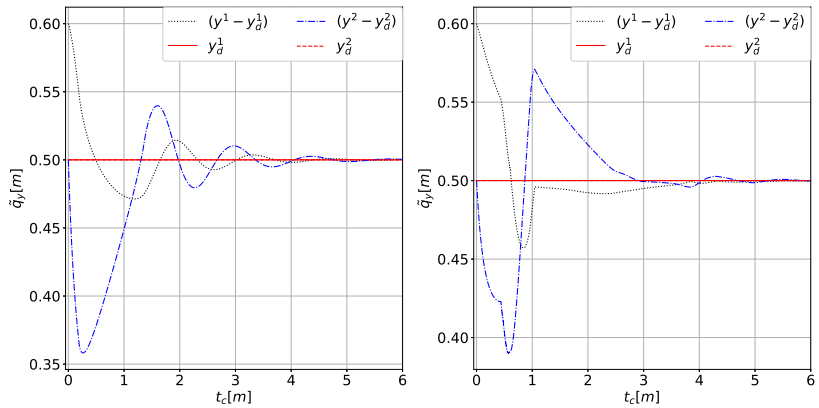
(a) (b)

Figura 4.24: En el escenario (a) se muestra el error sin controlador, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #4.



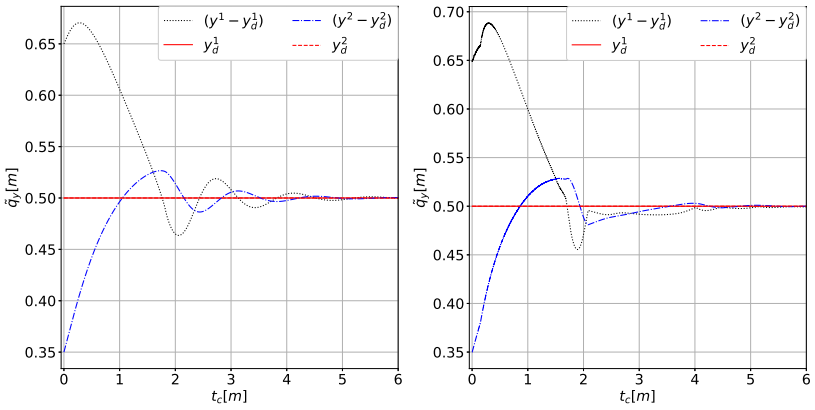
(a) (b)

Figura 4.25: En el escenario (a) se muestra el error sin controlador, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #5.



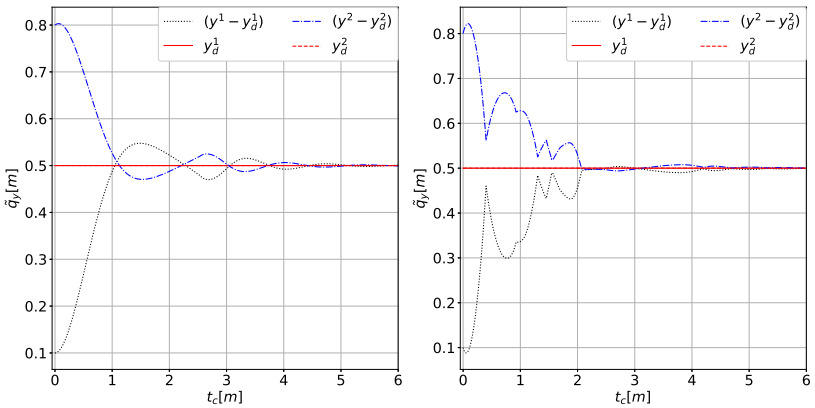
(a) (b)

Figura 4.26: En el escenario (a) se muestra el error sin controlador, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #6.



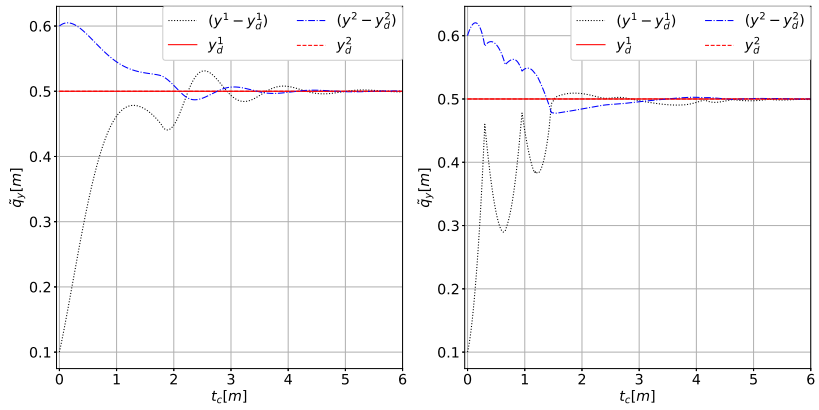
(a) (b)

Figura 4.27: En el escenario (a) se muestra el error sin controlador, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #7.



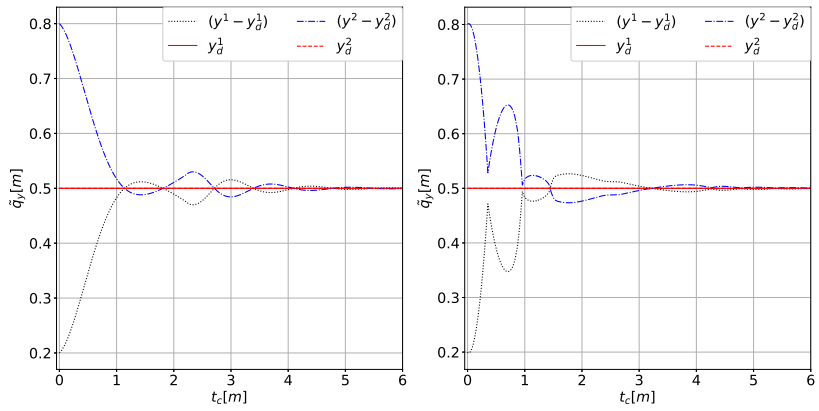
(a) (b)

Figura 4.28: En el escenario (a) se muestra el error sin controlador, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #8.



(a) (b)

Figura 4.29: En el escenario (a) se muestra el error sin controlador, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #9.

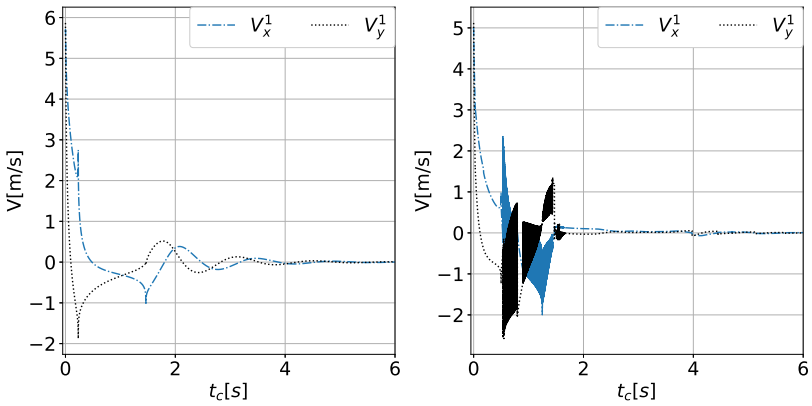


(a) (b)

Figura 4.30: En el escenario (a) se muestra el error sin controlador, mientras que en el escenario (b) se hace uso del controlador u_{l_1}, ambos escenarios son simulados para la condición inicial #10.

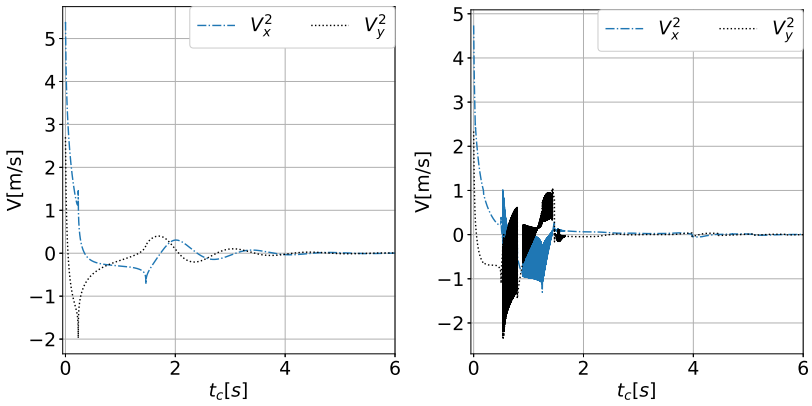
4.1.0.4 *Velocidades de los WMR's*

En esta sección se aborda el comportamiento de las velocidades para los robots móviles $WMR1$ y $WMR2$, en los ejes x y y.



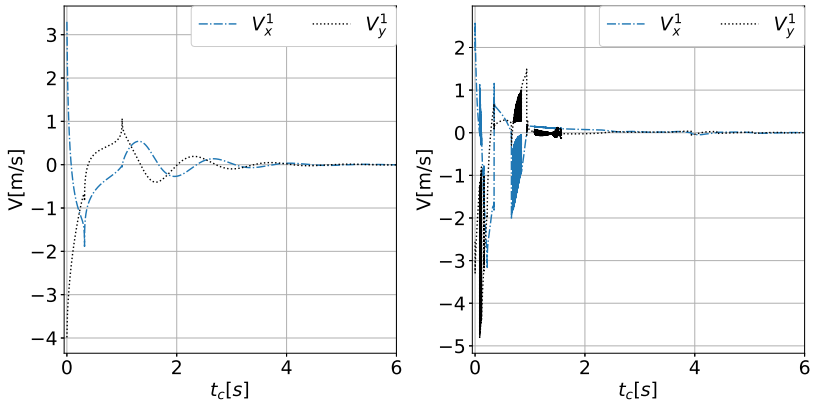
(a) (b)

Figura 4.31: En el escenario (a) se muestran las velocidades del $WMR1$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #1.



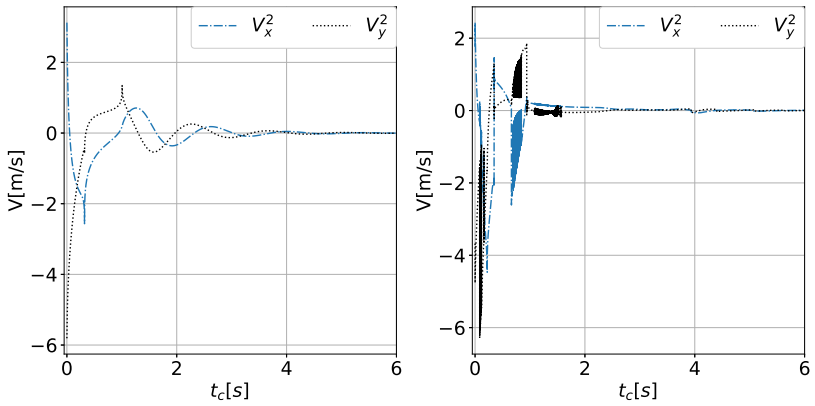
(a) (b)

Figura 4.32: En el escenario (a) se muestran las velocidades del $WMR2$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #1.



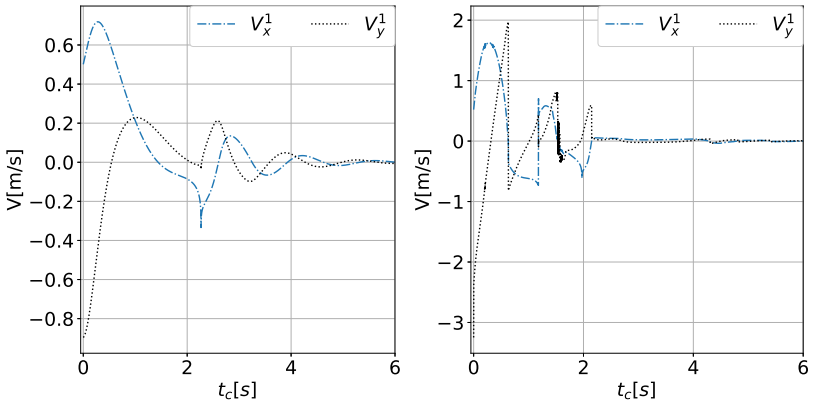
(a) (b)

Figura 4.33: En el escenario (a) se muestran las velocidades del $WMR1$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #2.



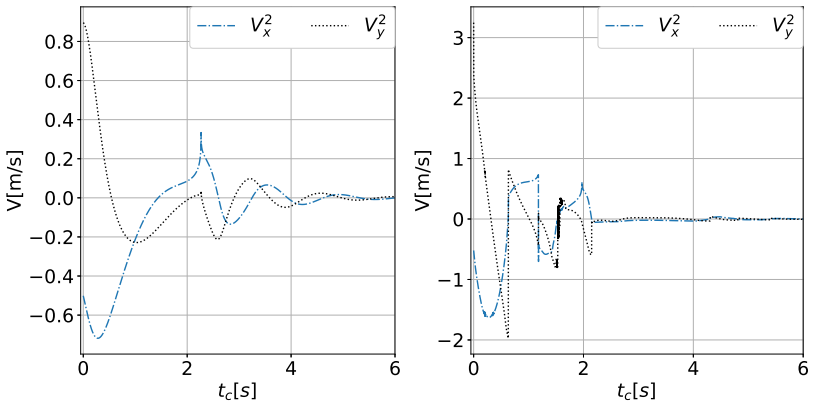
(a) (b)

Figura 4.34: En el escenario (a) se muestran las velocidades del $WMR2$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #2.



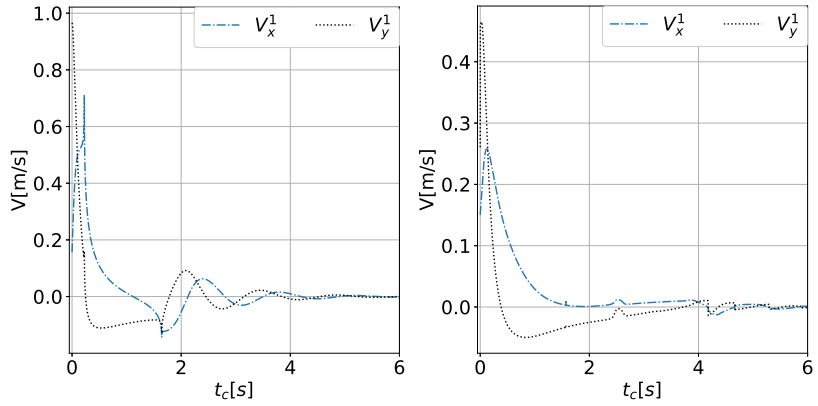
(a) (b)

Figura 4.35: En el escenario (a) se muestran las velocidades del $WMR1$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #3.



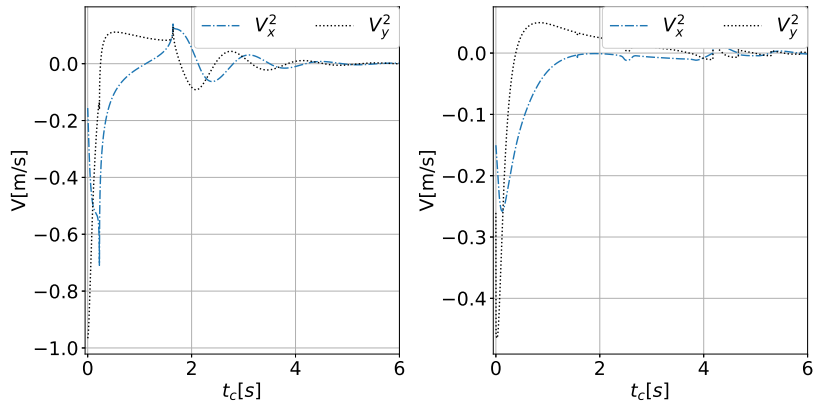
(a) (b)

Figura 4.36: En el escenario (a) se muestran las velocidades del $WMR2$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #3.



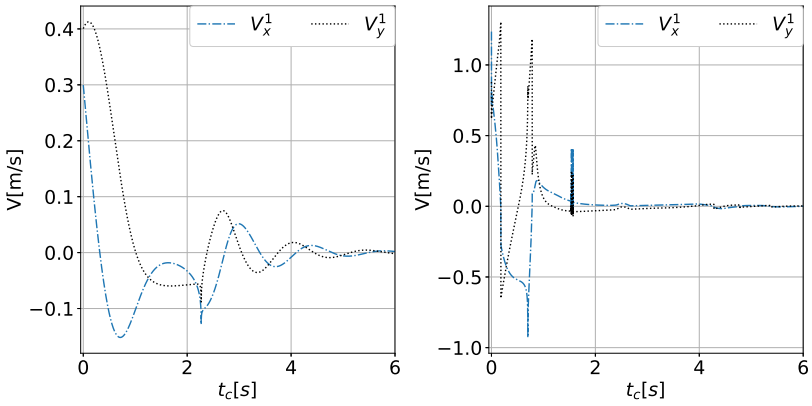
(a) (b)

Figura 4.37: En el escenario (a) se muestran las velocidades del $WMR1$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #4.



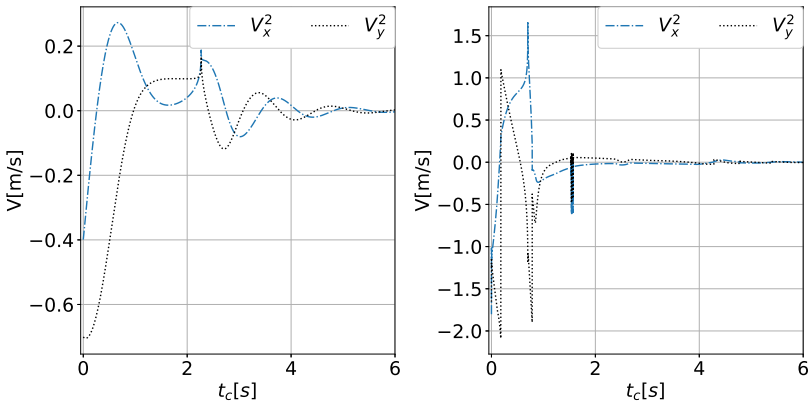
(a) (b)

Figura 4.38: En el escenario (a) se muestran las velocidades del $WMR2$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #4.



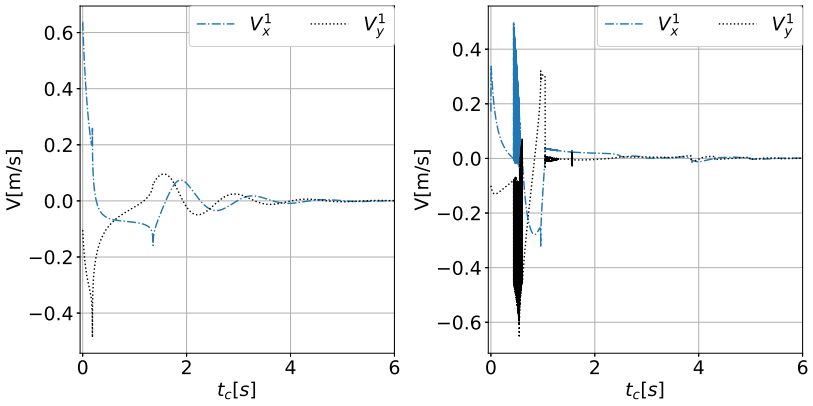
(a) (b)

Figura 4.39: En el escenario (a) se muestran las velocidades del $WMR1$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #5.



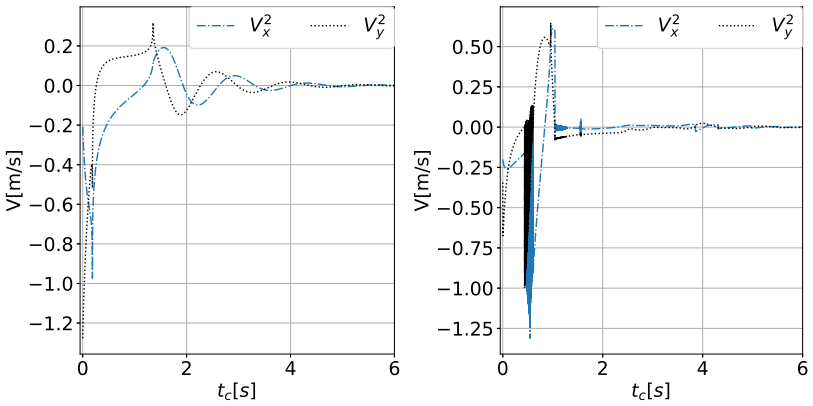
(a) (b)

Figura 4.40: En el escenario (a) se muestran las velocidades del $WMR2$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #5.



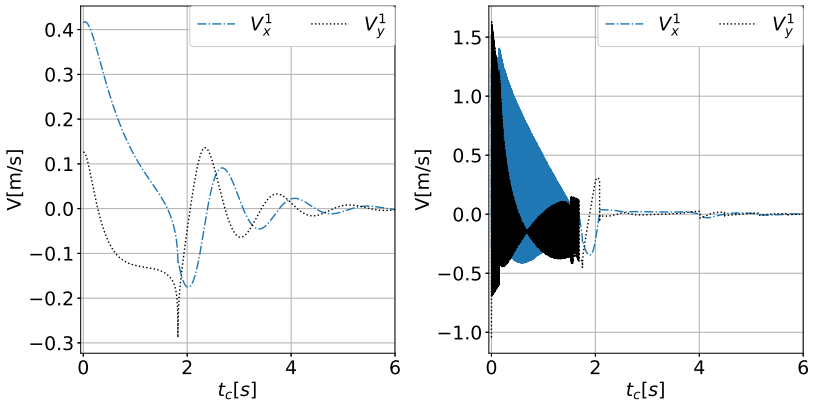
(a) (b)

Figura 4.41: En el escenario (a) se muestran las velocidades del $WMR1$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #6.



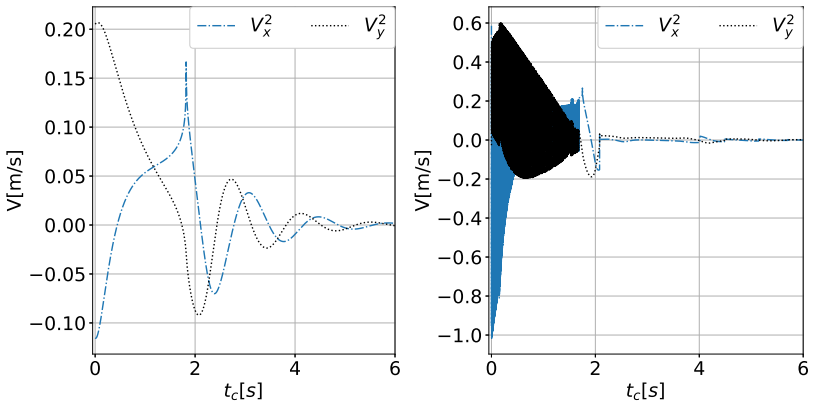
(a) (b)

Figura 4.42: En el escenario (a) se muestran las velocidades del $WMR2$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #6.



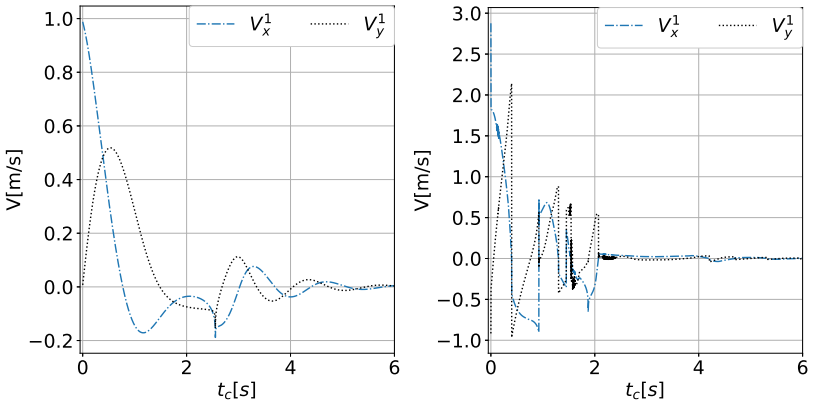
(a) (b)

Figura 4.43: En el escenario (a) se muestran las velocidades del $WMR1$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #7.



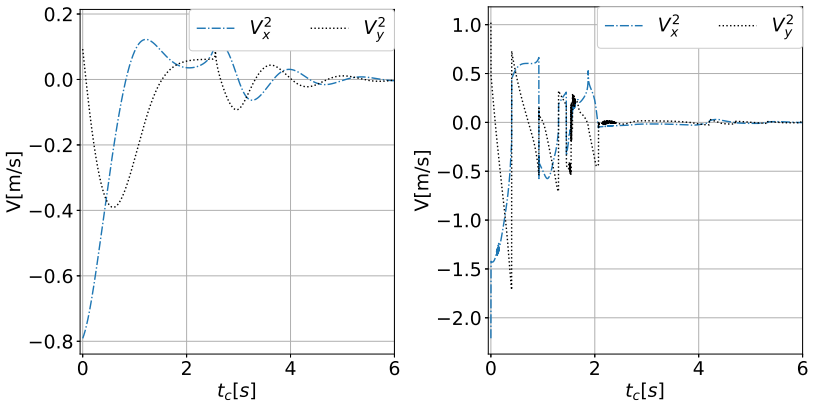
(a) (b)

Figura 4.44: En el escenario (a) se muestran las velocidades del $WMR2$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #7.



(a) (b)

Figura 4.45: En el escenario (a) se muestran las velocidades del $WMR1$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #8.



(a) (b)

Figura 4.46: En el escenario (a) se muestran las velocidades del $WMR2$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #8.

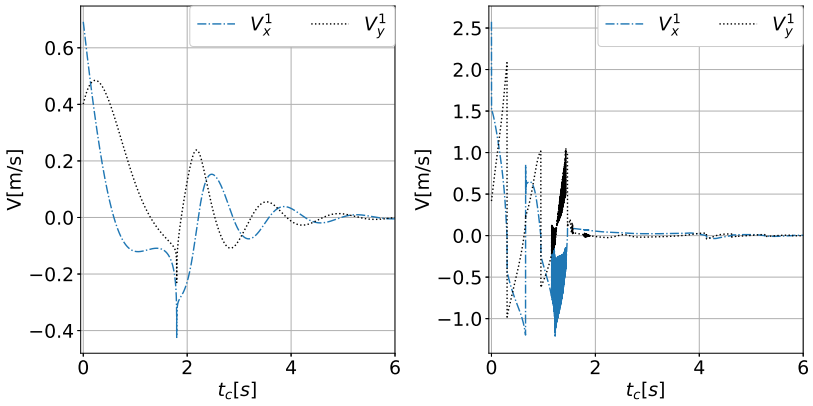


Figura 4.47: En el escenario (a) se muestran las velocidades del $WMR1$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #9.

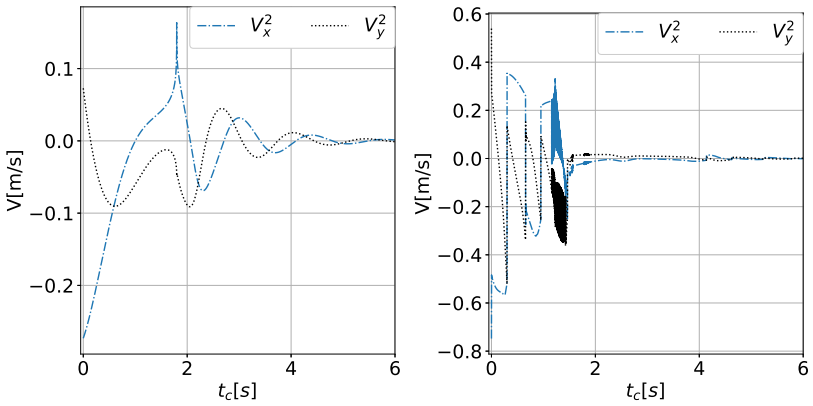
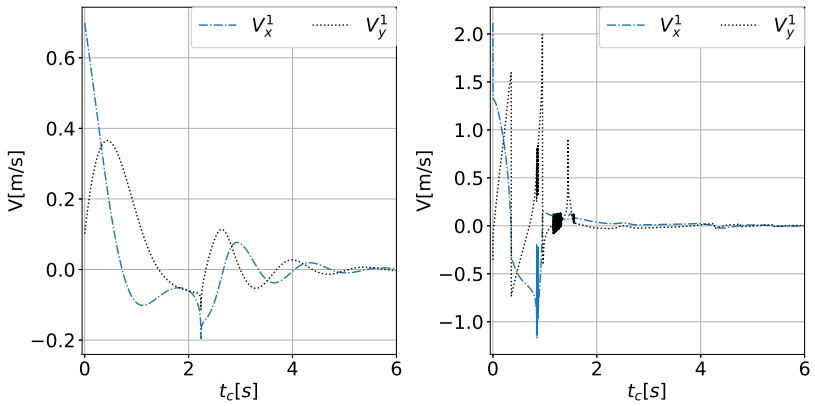
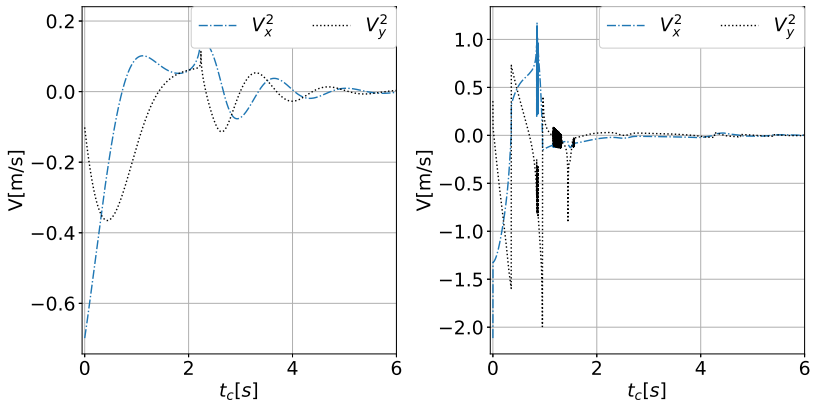


Figura 4.48: En el escenario (a) se muestran las velocidades del $WMR2$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #9.



(a) (b)

Figura 4.49: En el escenario (a) se muestran las velocidades del $WMR1$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #10.

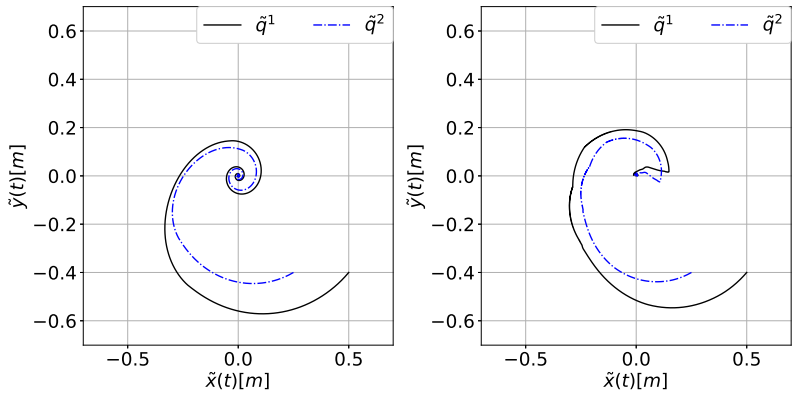


(a) (b)

Figura 4.50: En el escenario (a) se muestran las velocidades del $WMR2$ cuando no cuenta con comportamiento aprendido y en el escenario (b) se incorpora la solución u_{l_1}, ambos casos para la condición inicial #10.

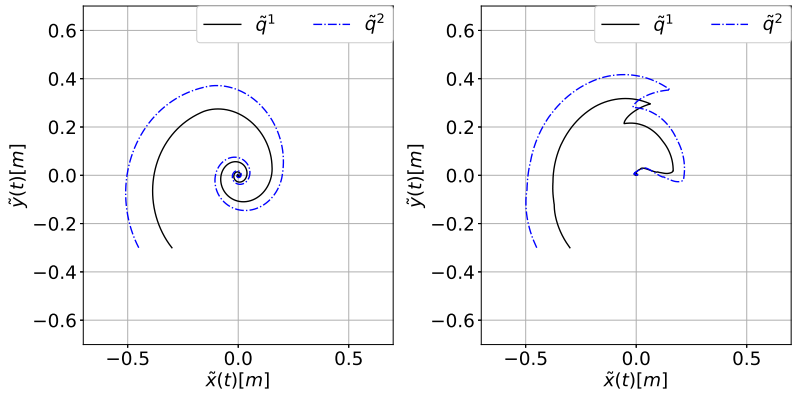
4.1.0.5 *Dinámica del error de cada WMR*

En esta sección se representa la dinámica del error q^i para dos robots móviles llamados q^1 y q^2, conforme se aproximén a la meta el error se acercará a cero.



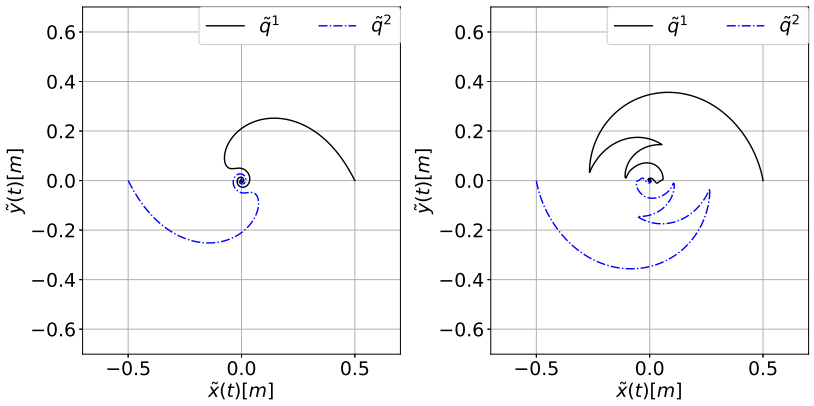
(a) (b)

Figura 4.51: En el inciso (a) se representa la dinámica del error para el escenario donde los robots tienen solo el comportamiento forzado, mientras que en el inciso (b) se incorpora el comportamiento aprendido u_{l_1}, ambos para la condición #1.



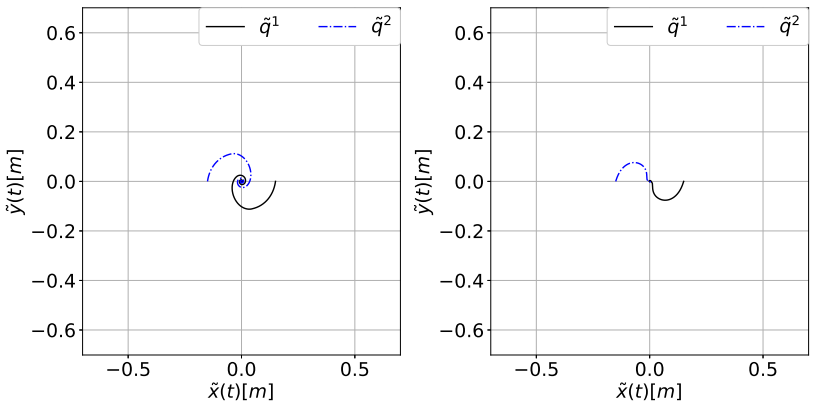
(a) (b)

Figura 4.52: En el inciso (a) se representa la dinámica del error para el escenario donde los robots tienen solo el comportamiento forzado, mientras que en el inciso (b) se incorpora el comportamiento aprendido u_{l_1}, ambos para la condición #2.



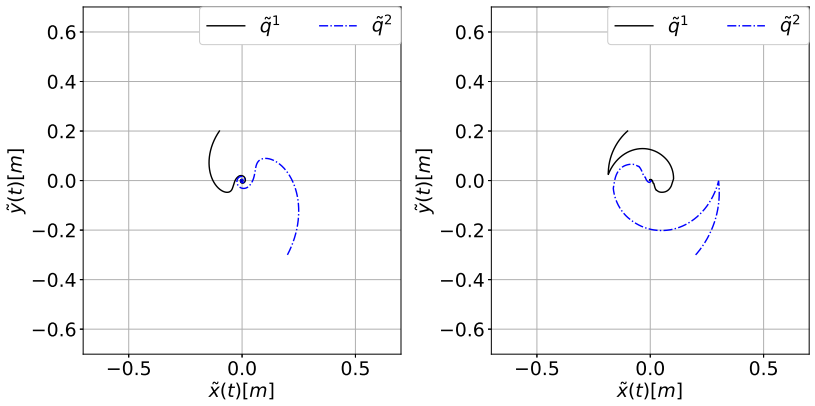
(a) (b)

Figura 4.53: En el inciso (a) se representa la dinámica del error para el escenario donde los robots tienen solo el comportamiento forzado, mientras que en el inciso (b) se incorpora el comportamiento aprendido u_{l_1}, ambos para la condición #3.



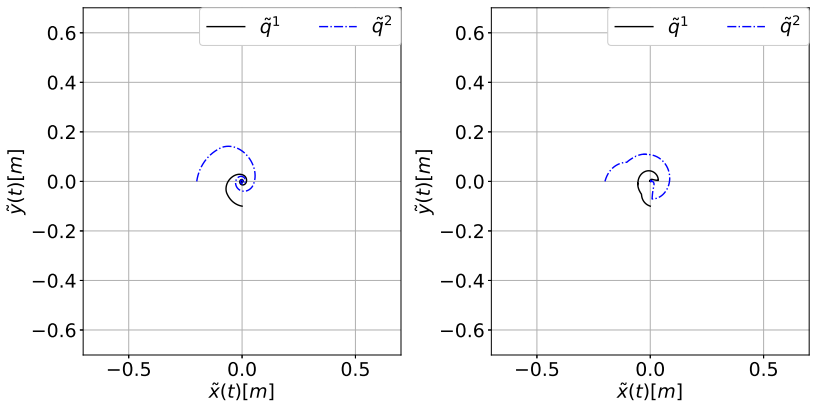
(a) (b)

Figura 4.54: En el inciso (a) se representa la dinámica del error para el escenario donde los robots tienen solo el comportamiento forzado, mientras que en el inciso (b) se incorpora el comportamiento aprendido u_{l_1}, ambos para la condición #4.



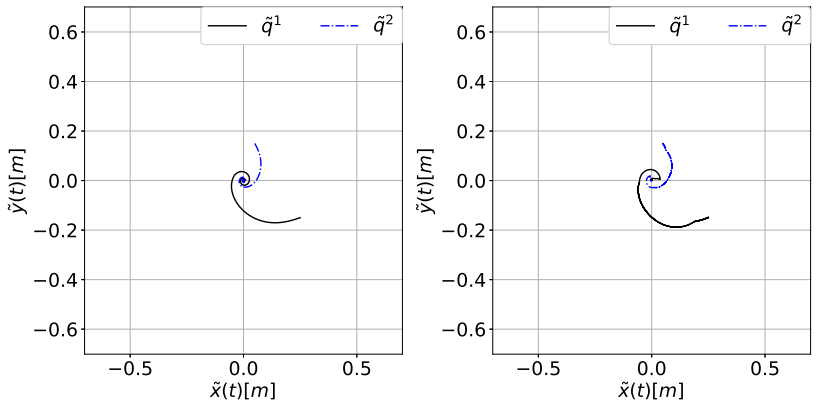
(a) (b)

Figura 4.55: En el inciso (a) se representa la dinámica del error para el escenario donde los robots tienen solo el comportamiento forzado, mientras que en el inciso (b) se incorpora el comportamiento aprendido u_{l_1}, ambos para la condición #5.



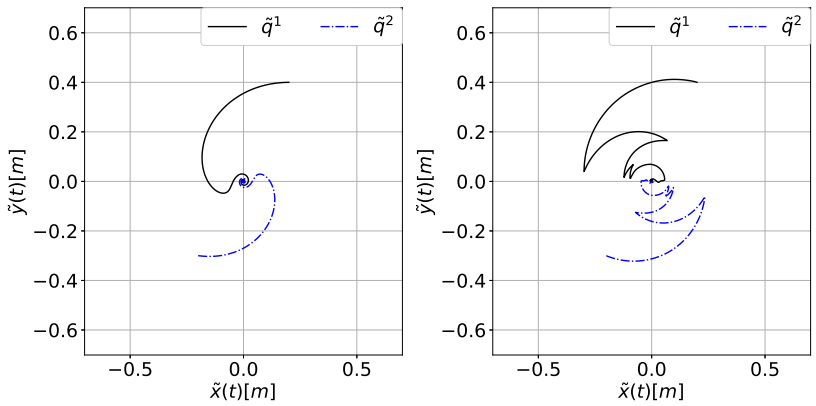
(a) (b)

Figura 4.56: En el inciso (a) se representa la dinámica del error para el escenario donde los robots tienen solo el comportamiento forzado, mientras que en el inciso (b) se incorpora el comportamiento aprendido u_{l_1}, ambos para la condición #6.



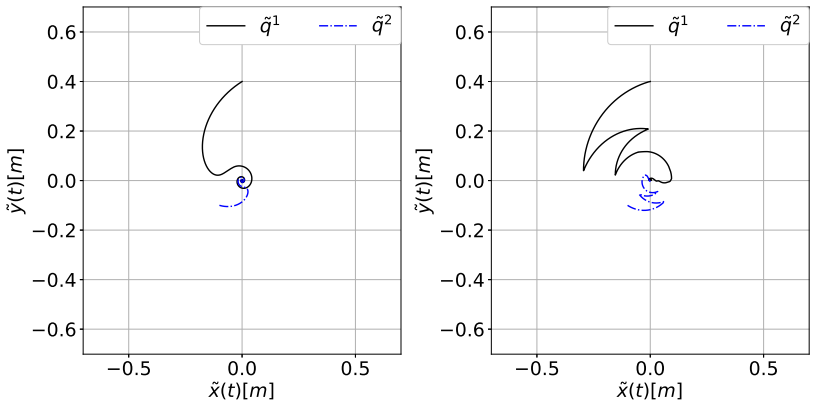
(a) (b)

Figura 4.57: En el inciso (a) se representa la dinámica del error para el escenario donde los robots tienen solo el comportamiento forzado, mientras que en el inciso (b) se incorpora el comportamiento aprendido u_{l_1}, ambos para la condición #7.



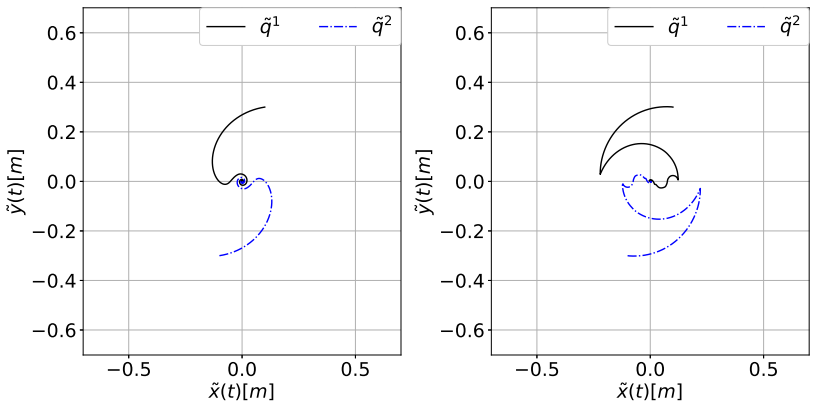
(a) (b)

Figura 4.58: En el inciso (a) se representa la dinámica del error para el escenario donde los robots tienen solo el comportamiento forzado, mientras que en el inciso (b) se incorpora el comportamiento aprendido u_{l_1}, ambos para la condición #8.



(a) (b)

Figura 4.59: En el inciso (a) se representa la dinámica del error para el escenario donde los robots tienen solo el comportamiento forzado, mientras que en el inciso (b) se incorpora el comportamiento aprendido u_{l_1}, ambos para la condición #9.

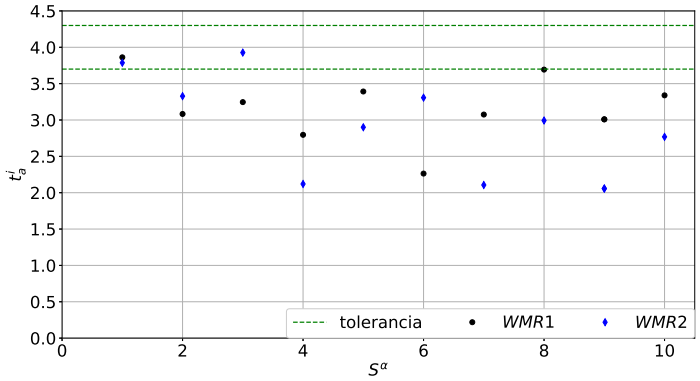


(a) (b)

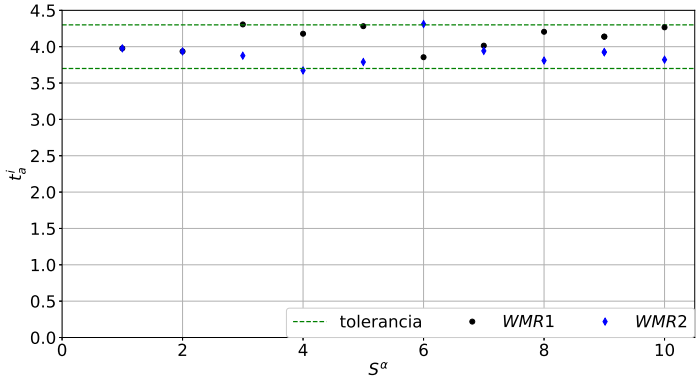
Figura 4.60: En el inciso (a) se representa la dinámica del error para el escenario donde los robots tienen solo el comportamiento forzado, mientras que en el inciso (b) se incorpora el comportamiento aprendido u_{l_1}, ambos para la condición #10.

4.1.0.6 *Validación del controlador de tiempo*

En esta sección se hace una comparación entre el tiempo de llegada a la meta para cada condición inicial cuando el sistema solo cuenta con el comportamiento aprendido y cuando cuenta solamente con comportamiento forzado, y así, validar que el sistema llegue a la meta en el tiempo entrenado.



(a)



(b)

Figura 4.61: En el inciso (a) se representan los tiempos de llegada de los $WMR1$ y $WMR2$ para cada condición inicial en el sistema con comportamiento forzado, mientras que en el inciso (b) se representan los tiempos de llegada para el sistema con comportamiento aprendido u_{l_1}.

4.2 Estadísticas

En esta sección se describen las estadísticas generales de las evoluciones, abordando la incidencia de funciones y terminales, así como, el desempeño de la ejecución donde se encontró la mejor solución. Se hará una comparación del conjunto de soluciones de las tres corridas para resaltar las diferencias presentes. Para considerar buena una solución, se definió que debe ser cercana a 0, mientras más cerca el valor sea de 0 mejor sera la solución.En el caso contrario, entre el desempeño sea mayor a 0 peor sera la solución, además, se desechan las soluciones con un desempeño mayor a 0,15, y a su vez, se considerá como mejor grupo de soluciones de todo el conjunto a aquellas 13 soluciones con el valor más pequeño de desempeño.

4.2.1 *Incidencia de Funciones y Terminales para las soluciones únicas*

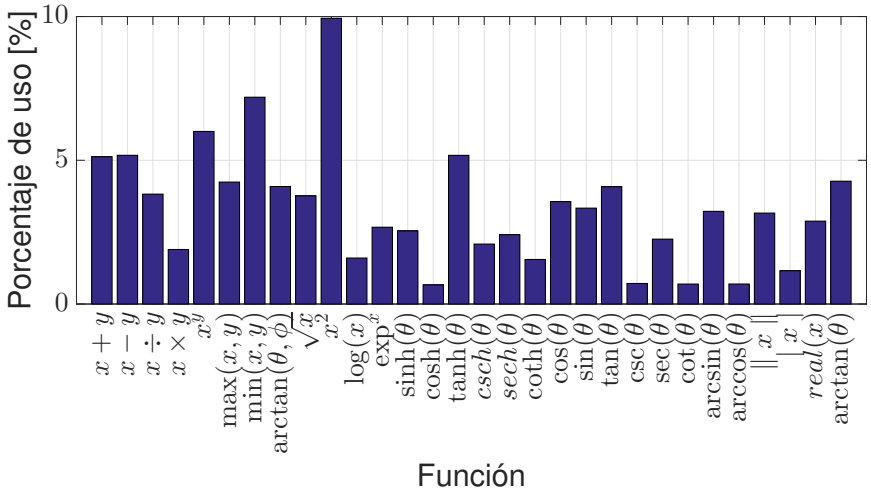


Figura 4.62: Incidencia de Funciones para las corridas de 200 generaciones y 200 de población sin individuos repetidos.

La clasificación de la incidencia entre funciones y terminales esta dada por las siguientes características:

- Se basan en las ejecuciones de 200, 300 y 400 individuos dentro de la población.
- Se clasifícan descartando soluciones repetidas.
- Abarca la cantidad de individuos encontrados en la tabla(4.1).
- Se consideran 30 funciones y 19 terminales.

En la figura(4.68), se muestra la cantidad de funciónes encontradas en las soluciones para el conjunto de 200 individuos, donde la más usada en este caso es x^2 y la función que menos se uso es el $\cosh\theta$. Por otro lado en las terminales, la mas usada fue $d_y^{2\hat{o}^1}$ y la menos usada fue la de y^2.

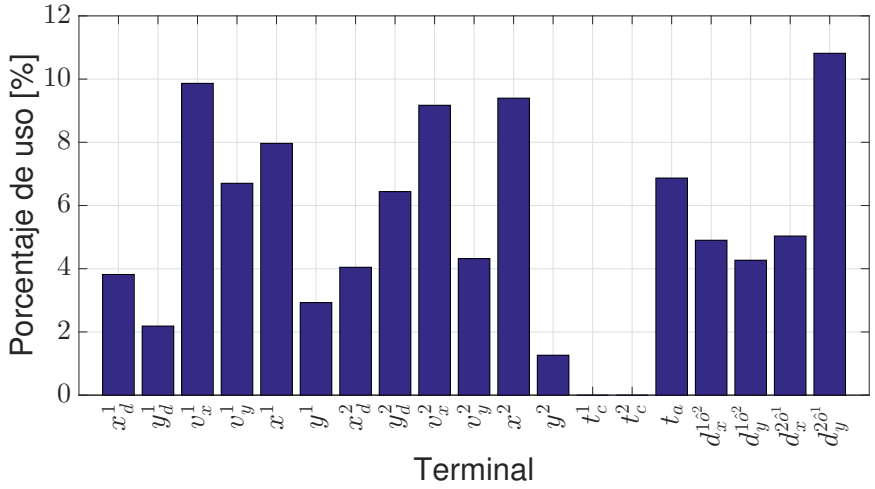


Figura 4.63: Incidencia de Terminales para las corridas de 200 generaciones y 200 de población sin individuos repetidos

En el caso de la figura(4.64), la función que más apareció fue la de v_x^1, mientras que la de menor incidencia fue la de $d_y^{1\hat{o}^2}$, para el caso de las terminales la de mayor presencia fue la suma$(x+y)$ y la de menor presencia fue la de $\csc(\theta)$.

Finalmente en el conjunto de soluciones para la figura(4.66), la función más usada fue la del minimo $min(x,y)$ y la que menos presencia tuvo fue la división. Mientras que en las terminales la que más se uso fue t_c^1 y la que se uso poco fue la de y^1.

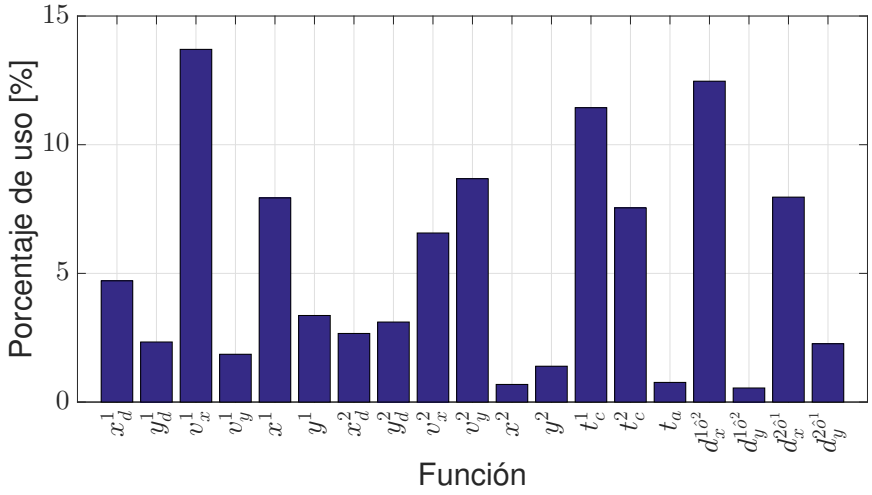


Figura 4.64: Incidencia de Funciones para las corridas de 200 generaciones y 300 de población sin individuos repetidos.

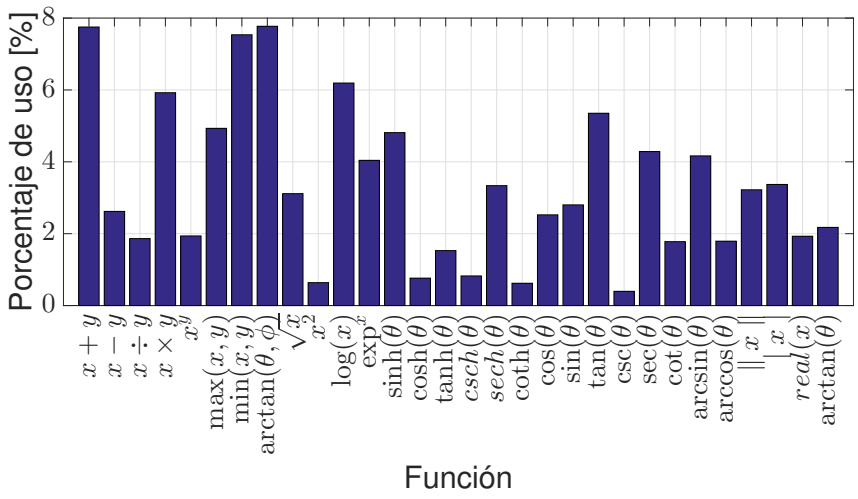


Figura 4.65: Incidencia de Terminales para las corridas de 200 generaciones y 300 de población sin individuos repetidos

De estos conjuntos analizados, se añade una condicionante para elegir las mejores soluciones y es elegir solo aquellas que tengan un desempeño menor a 0,15, esto debido a que son los individuos con un mejor comportamiento para la mayoria de las condiciones iniciales con las que se entreno el algoritmo evo-

lutivo. Entonces, podemos decir que se clasificará la incidencia de funciones y terminales con soluciones únicas y a su vez, se eligirán solo las que esten dentro de la métrica mencionada anteriormente.

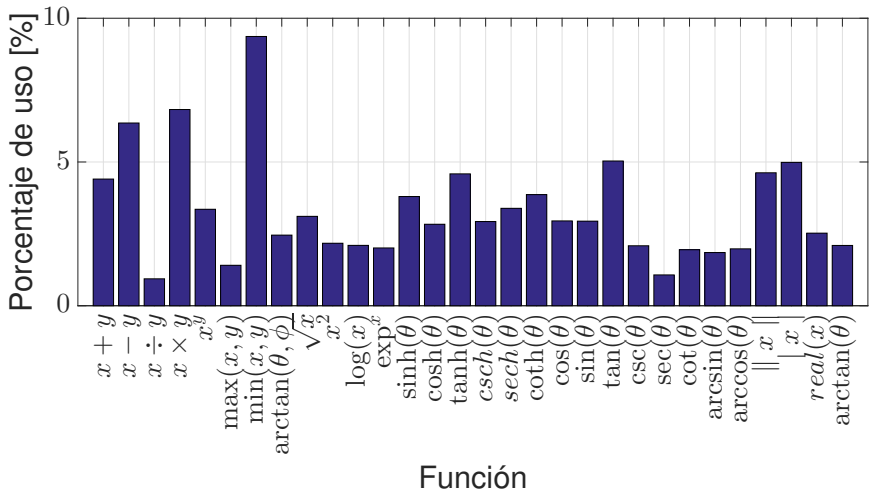


Figura 4.66: Incidencia de Funciones para las corridas de 300 generaciones y 400 de población sin individuos repetidos

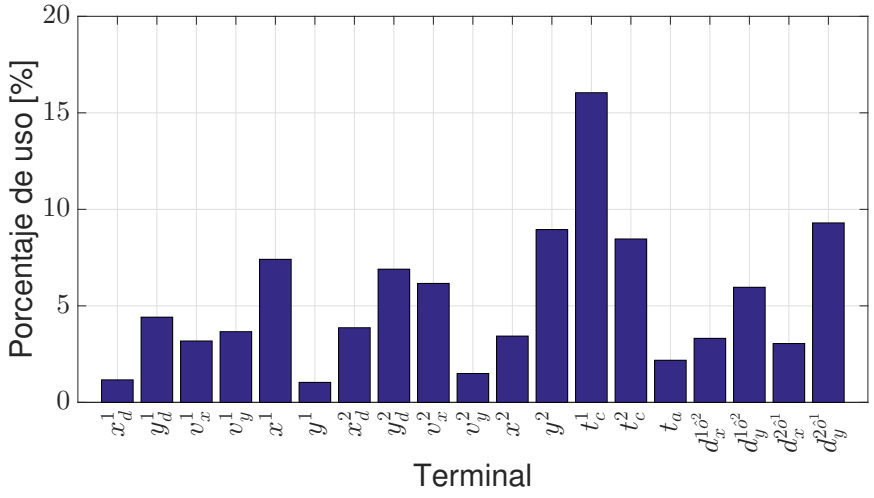


Figura 4.67: Incidencia de Terminales para las corridas de 300 generaciones y 400 de población sin individuos repetidos

4.2.2 *Incidencia de funciones y terminales para las soluciones que cumplen con los objetivos de control*

- En el conjuto de 200 generaciones y 200 de población se encontraron 0 soluciones que cuentan con un buen desempeño en la mayoria de las 10 condiciones iniciales propuestas.

- En el conjuto de 200 generaciones y 300 de población se encontraron 39 soluciones que cuentan con un buen desempeño en la mayoria de las 10 condiciones iniciales propuestas.

- En el conjuto de 300 generaciones y 400 de población se encontraron 66 soluciones que cuentan con un buen desempeño en la mayoria de las 10 condiciones iniciales propuestas.

Analizando la figura(4.68) podemos notar como la incidencia de funciones para las soluciones unicas que tienen el mejor desempeño de todo el conjunto dos, se repite la función $+$, mientras que en el caso de las terminales, la que más se repite es la de v_x^1.

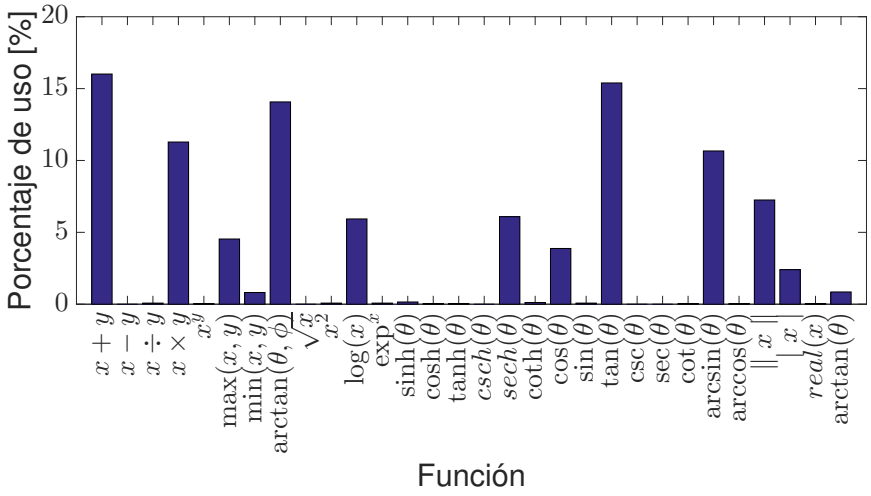


Figura 4.68: Incidencia de funciones para las soluciones que tienen un buen desempeño, de las cuales se seleccionarán las mejores para el conjunto de soluciones con 200 generaciones y 300 de población

Finalmente, para el caso del conjunto 3, tenemos que la incidencia de funciones más usadas fue la de $tanh(\theta)$ como se muestra en la figura(4.70) y para el caso de las terminales la más usada fue la de t_c^1.

Recordando que del segundo conjunto de soluciones se tomaron las 13 mejores soluciones para realizar un análisis del cumplimiento del objetivo de tiempo de llegada y al final se selecciono 1 solución para efectuar la comparación entre el uso de comportamiento aprendido contra el sistema forzado solamente.

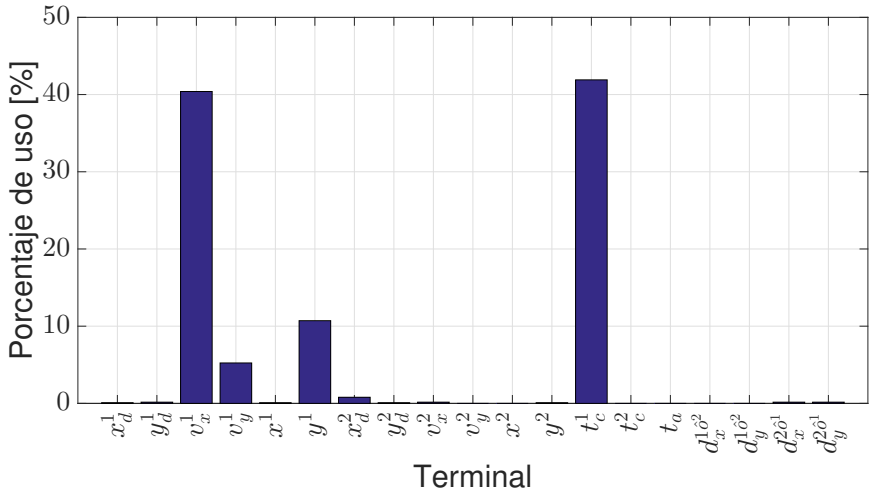


Figura 4.69: Incidencia de terminales para las soluciones que tienen un buen desempeño del conjunto de soluciones con 200 generaciones y 300 de población

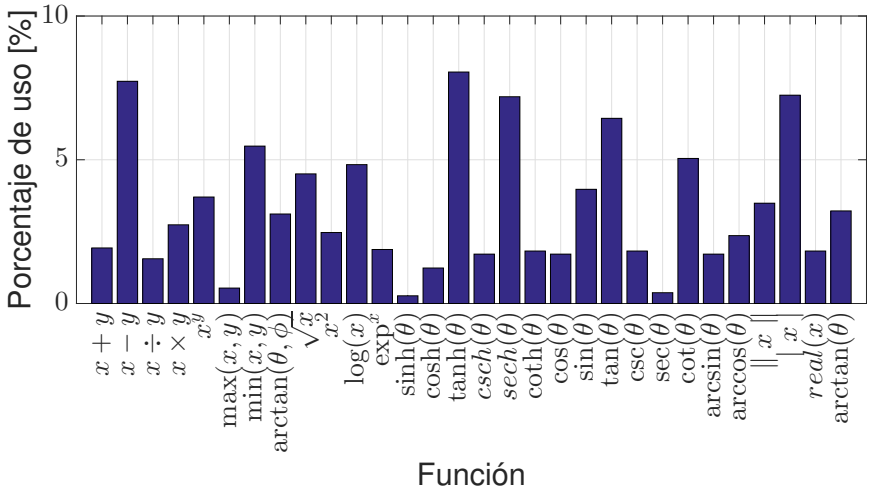


Figura 4.70: Incidencia de funciones para las soluciones que tienen un buen desempeño del conjunto de soluciones con 300 generaciones y 400 de población

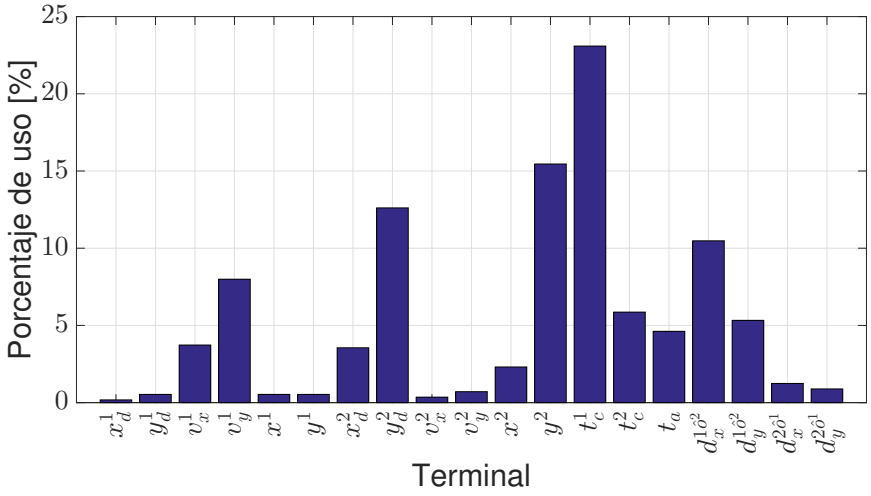


Figura 4.71: Incidencia de funciones para las soluciones que tienen un buen desempeño del conjunto de soluciones con 300 generaciones y 400 de población

4.2.3 *Estadísticas del desempeño de los individuos.*

Dentro de esta sección se describen las estadísticas generales del desempeño para la ejecución **Ev 2** de la tabla(3.5), abordando también la cantidad de nodos (funciones y terminales) para cada generación.

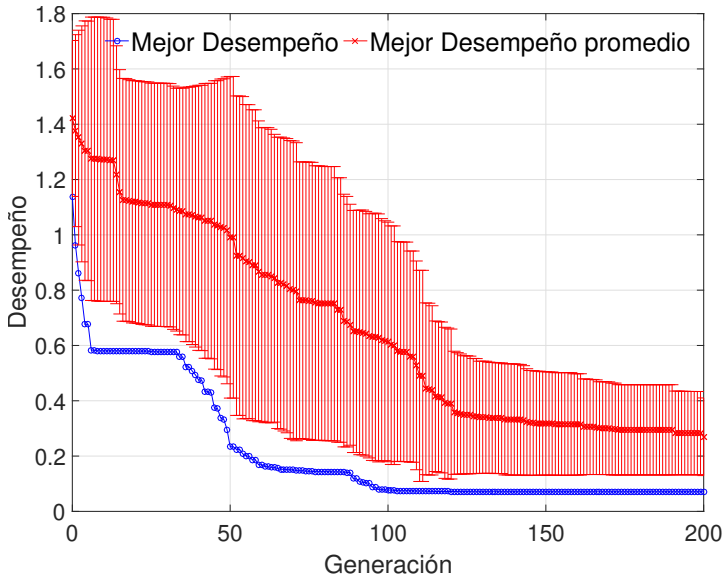


Figura 4.72: Desempeño promedio del conjunto de corridas para la ejecución **Ev 2**, donde se seleccionó el mejor individuo, basado en la tabla (3.5)

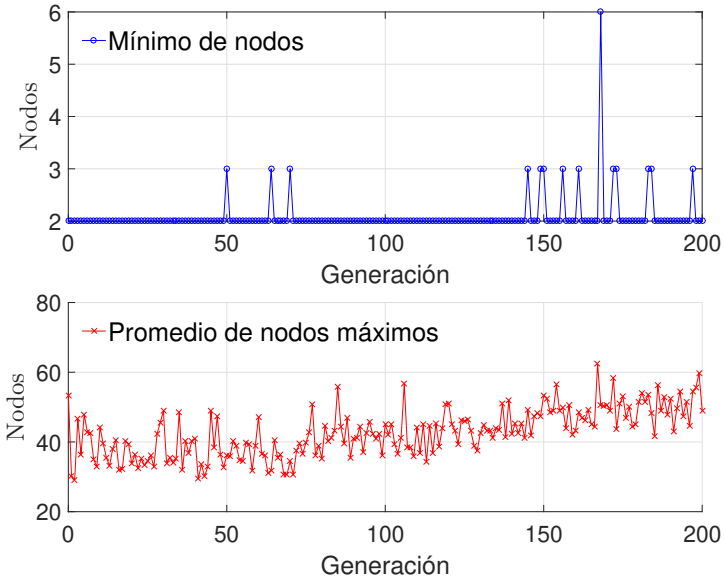


Figura 4.73: Tamaño de los nodos(funciones y terminales) usados por generación en los individuos para las cuatro corridas de la ejecución **Ev 2**.

4.3 Validación númerica con diferentes condiciones inciales a las usadas en el aprendizaje e incrementando el número de robots.

En esta sección se abordaran los resultados de validación para los casos de análisis y simulación de 2 robots, 3 robots y 4 robots respectivamente, se validará que el controlador aprendido realice un ajuste en el tiempo de navegación. Para esto, se propusieron 3 escenarios, el escenario 1 esta enfocado en validar el cumplimiento del objetivo de tiempo para el caso de 2 robots con diferentes condiciones iniciales. En el escenario 2, ademas de añadir un robot y elegir condiciones iniciales diferentes al aprendizaje, se incrementara el area de la zona donde se posicionan las metas para cada WMR. Finalmente, en el escenario 4 se repetirán los pasos del escenario 3, incrementando el numero de robots a cuatro de ellos.

Para analizae el caso de 2 robots se propone un conjunto de condiciones iniciales diferentes a las usadas en la evolución, se mantienen los parámetros originales y se simulá durante 6s.

4.3.1 *Condiciones iniciales*

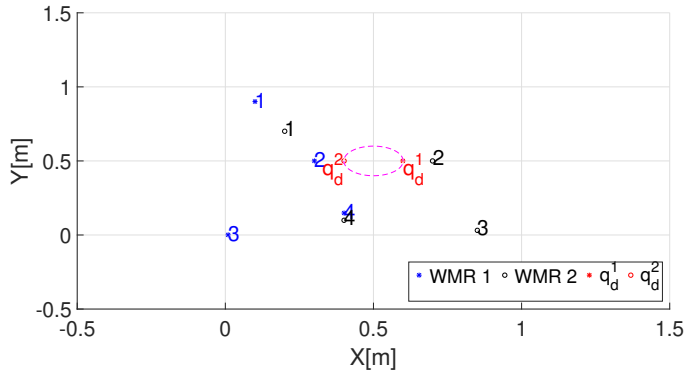


Figura 4.74: Distribución de las condiciones iniciales a lo largo del eje coordenado para el caso de 2 robots.

Las condiciones iniciales usadas para la validación de 2 robots móviles son diferentes a las usadas para el aprendizaje, se pueden visualizar en la tabla(4.4) y la figura(4.74).

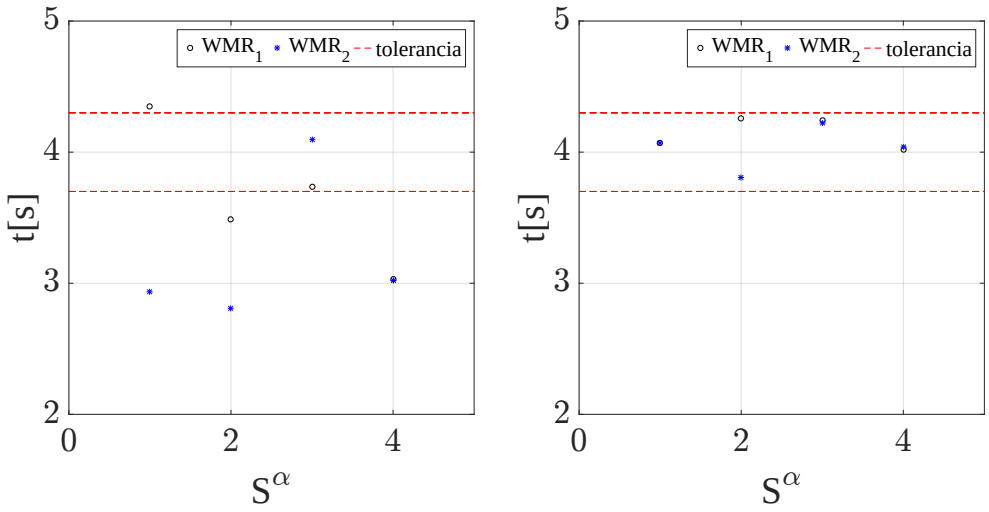
Tabla 4.4: Validación: Condiciones iniciales para 2 robots.

Condiciones iniciales					
α	WMR_1	WMR_2	α	WMR_1	WMR_2
#	$q^1(0)$	$q^2(0)$	#	$q^1(0)$	$q^2(0)$
1	$[0.1, 0.9]^T$	$[0.2, 0.7]^T$	2	$[0.3, 0.5]^T$	$[0.7, 0.5]^T$
3	$[0.01, 0.0]^T$	$[0.85, 0.03]^T$	4	$[0.4, 0.15]^T$	$[0.4, 0.1]^T$

Finalmente, en la tabla(4.5) y en la figura(4.75) se valida que se haga el ajuste del tiempo en el que llegó cada robot móvil para las condiciones propuestas.

4.3.2 *Resultados*

Por último, se presentan las trayectorias de los robots móviles en la figura(4.77) para apreciar visualmente que ambos robots llegan a las metas desea-



(a) (b)

Figura 4.75: Validación de los tiempos de llegada para ambos robots móviles; en el caso (a) se muestra el tiempo de llegada con comportamiento forzado y el caso(b) se muestran los tiempos de llegada con comportamiento aprendido.

das. Para complementar, en la figura(4.76) se presenta la dinámica de los robots móviles para cada condición inicial.

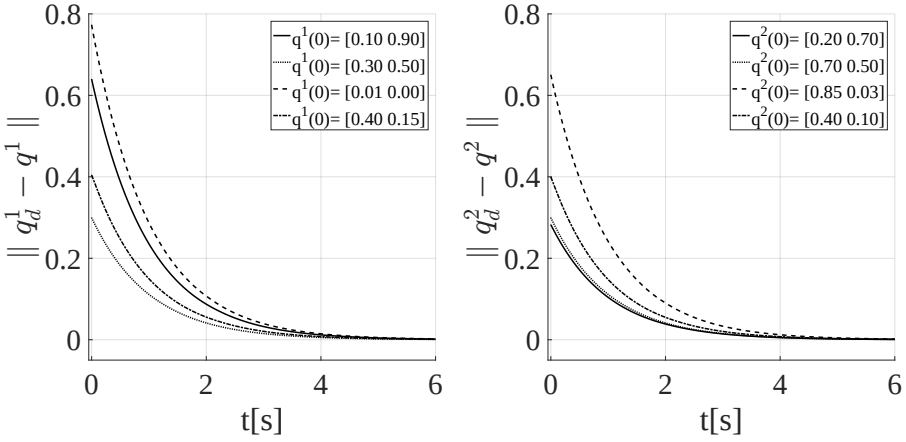


Figura 4.76: Validación: Cumplimiento de la tarea de **encuentro** con evasión de obstáculos y tiempo optimizado de llegada, para las condiciones iniciales de la tabla(4.4). (a) Norma del error de posición del WMR_1; (b) Norma del error de posición del WMR_2.

Tabla 4.5: Validación: Tiempo de llegada de 4 diferentes escenarios.

Tiempo objetivo para el encuentro, t_a= 4 segundos				
	$u^i = u_f^i$		$u^i = u_f^i + u_{l_1}$	
α	WMR_1	WMR_2	WMR_1	WMR_2
1	4.3470	2.9330	4.0680	4.0680
2	3.4850	2.8090	4.2560	3.8060
3	3.7360	4.0970	4.2440	4.2240
4	3.0310	3.0190	4.0190	4.0380
Avg	3.6497	3.2145	4.1467	4.0340

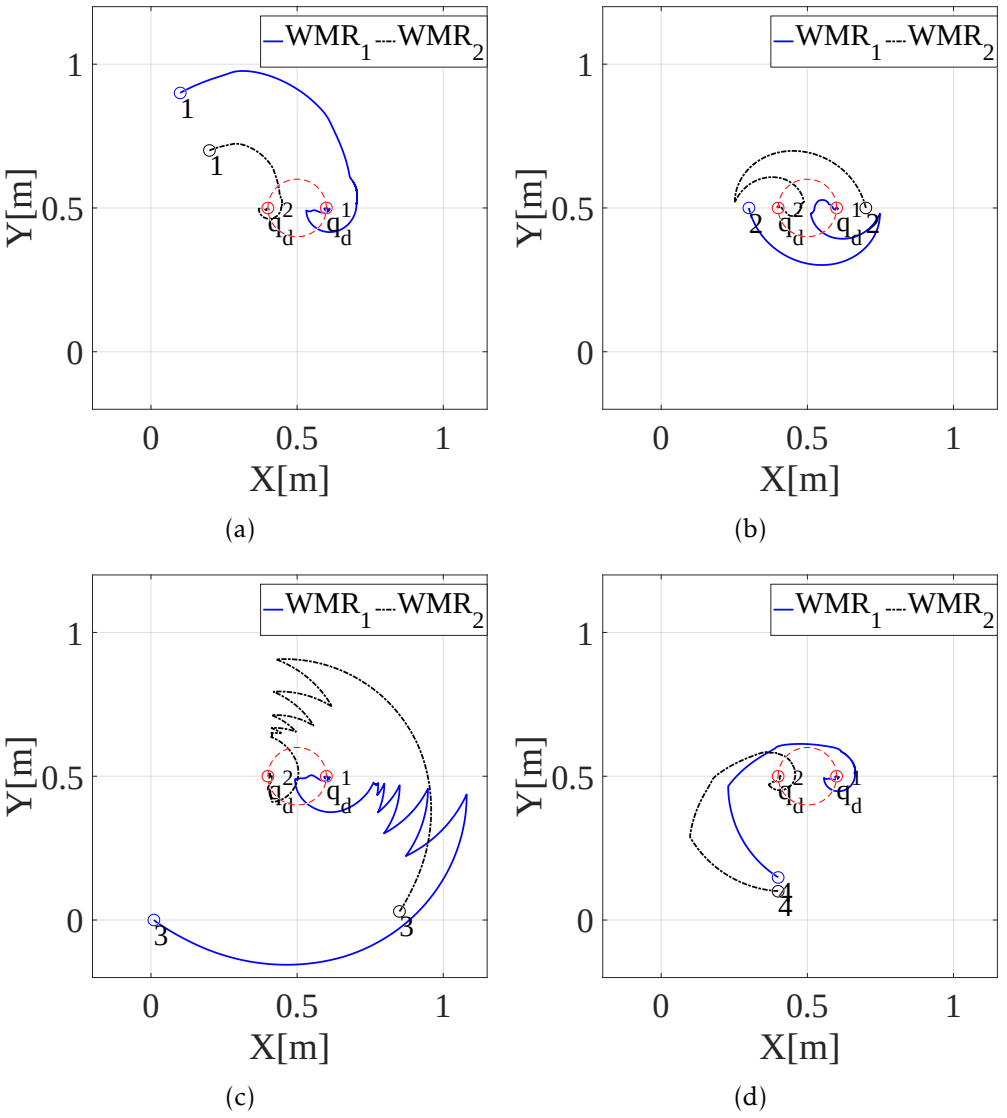


Figura 4.77: Validación: Trayectorias de los 4 escenarios seleccionados S^{α} definidos en la tabla 4.5.

4.3.3 *Validación númerica usando tres robots*

En esta sección se valida el caso de 3 robots, los cuales deben llegar a la meta en el tiempo entrenado y a su vez, los robots deben llegar a las metas deseadas evadiendose entre ellos. Para este caso de análisis se cambio el radio del área deseada a 0,5 y se implementó un controlador por robot que considera las trayectorias de los tres, siguiendo la estructura de las ecuaciones (3.4) y (3.25).

4.3.3.1 *Condiciones iniciales*

Tabla 4.6: Validación: Condiciones iniciales seleccionadas para 3 robots.

Condiciones iniciales			
α	WMR_1	WMR_2	WMR_3
#	$q^1(0)$	$q^2(0)$	$q^3(0)$
1	$[0.10, 0.10]^T$	$[0.10, 0.00]^T$	$[0.10, \text{-}0.50]^T$
2	$[0.60, 0.20]^T$	$[0.65, 0.10]^T$	$[0.35, 0.70]^T$
3	$[0.50, 0.60]^T$	$[0.50, 0.10]^T$	$[0.05, 0.70]^T$
4	$[0.00, 0.00]^T$	$[0.60, 0.00]^T$	$[1.00, 0.00]^T$

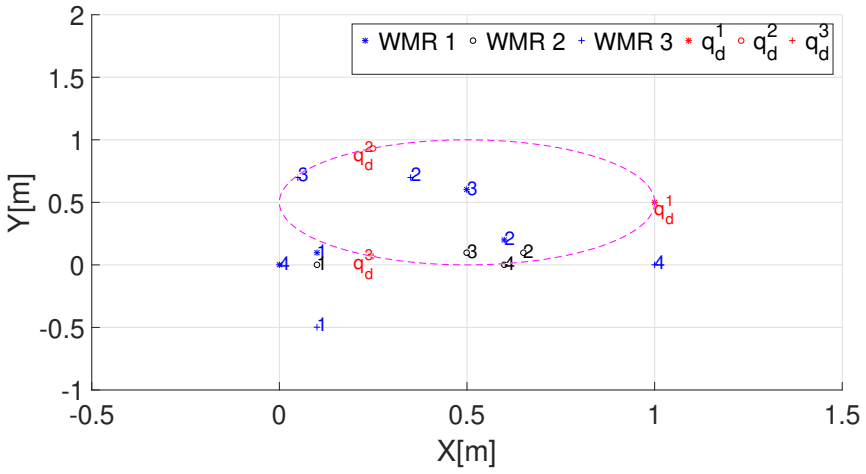


Figura 4.78: Distribución de las condiciones iniciales a lo largo del eje coordenado para el caso de 3 robots móviles.

En la figura(4.78) se muestra la forma en que estan distribuidas las condiciones iniciales para cada robot, donde los números del 1 al 4 corresponden a cada condición, y q_d^i corresponde a las posiciones deseadas para cada robot móvil dentro del área definida.

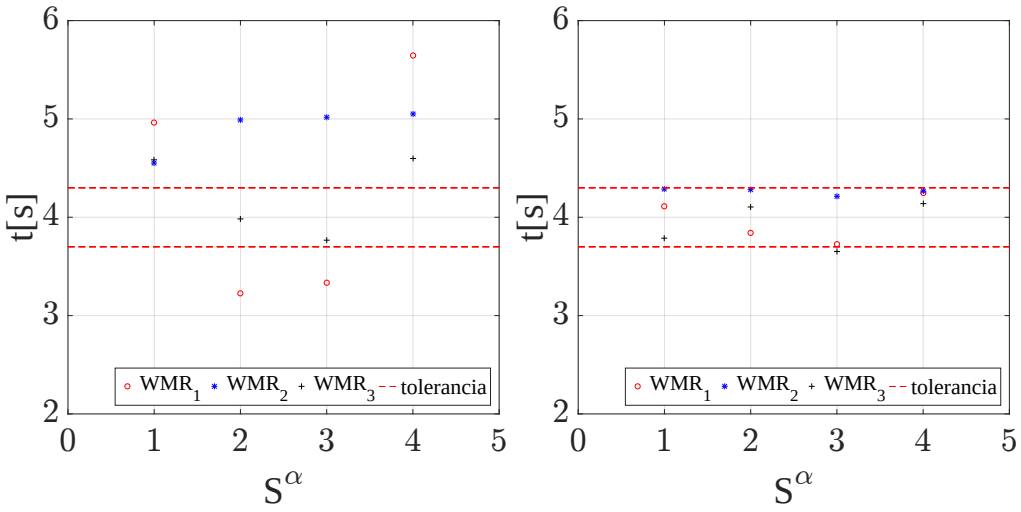


Figura 4.79: Validación de los tiempos de llegada para los tres robots móviles; en el caso (a) se muestra el tiempo de llegada con comportamiento forzado y el caso(b) se muestran los tiempos de llegada con comportamiento aprendido.

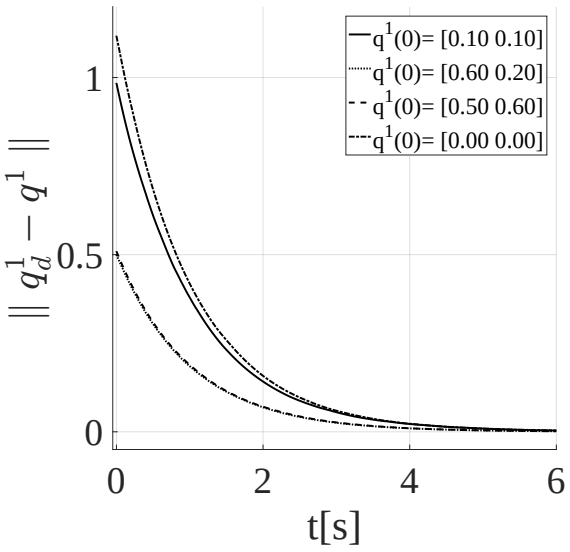
Mientras que en la figura(4.79) se aprecia como se ajusta el tiempo de llegada en el caso del comportamiento aprendido (b), donde, al menos para las condiciones iniciales validadas los robots cumplen con los objetivos de tiempo, representados en la tabla(4.7)

4.3.3.2 *Resultados*

Finalmente en la sección de resultados, se aprecia como independientemente de que los robots sean forzados a cumplir con un tiempo específico, dado, el comportamiento forzado, los robots siempre van a llegar a la meta, como se muestra en las figuras (4.80) hasta (4.83). Donde en las figuras(4.82) y (4.83) corresponden a las trayectorias de los 3 robots móviles.

Tabla 4.7: Validación: Tiempo de llegada de 4 escenarios para el caso de 3 robots móviles.

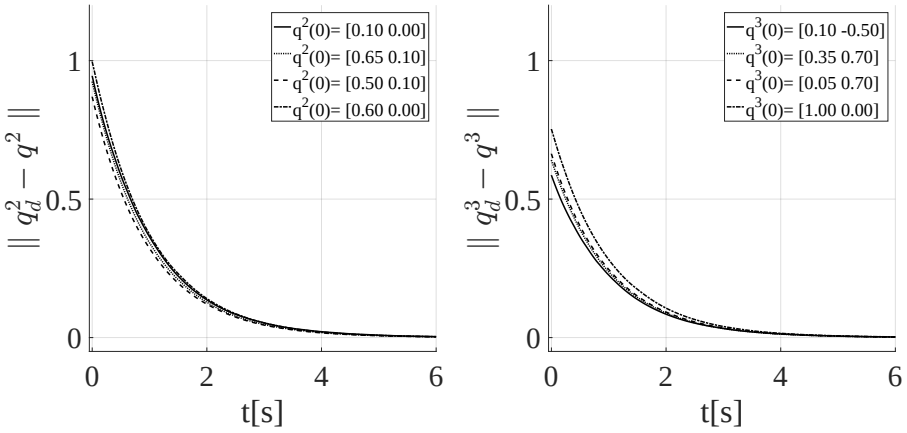
Tiempo de llegada para la tarea de **encuentro**, t_a= 4 segundos						
	$u^i = u^i_f$			$u^i = u^i_f + u_{l_1}$		
α	WMR_1	WMR_2	WMR_3	WMR_1	WMR_2	WMR_3
1	4.9620	4.5540	4.5850	4.1110	4.2860	3.7900
2	3.2230	4.9930	3.9860	3.8380	4.2830	4.1070
3	3.3330	5.0170	3.7690	3.7280	4.2130	3.6510
4	5.6460	5.0490	4.5950	4.2450	4.2660	4.1390
Avg	4.2910	4.9033	4.2337	3.9805	4.2620	3.9217



(a)

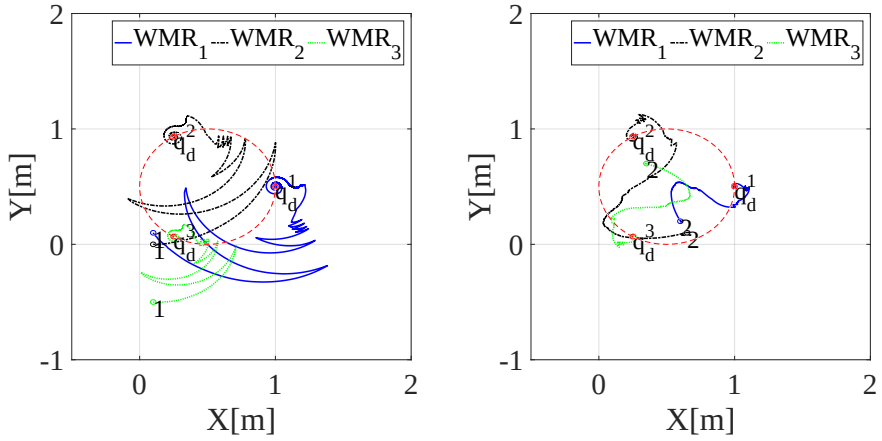
Figura 4.80: Validación: (a) Norma de error de posición del WMR_1.

Finalmente, en las figuras (4.82) y (4.83) se aprecia como sin importar donde esten posicionados los 3 robots móviles, estos llegan a las metas deseadas de cada uno respectivamente.



(a) (b)

Figura 4.81: Validación: Cumplimiento de la tarea de **encuentro** con evasión de obstáculos y tiempo de llegada optimizado. (a) Norma de error de posición del WMR_2; (b) Norma de error de posición del WMR_3.



(a) (b)

Figura 4.82: Validación: Trayectorias de los escenarios seleccionados 1 y 2 para 3 robots móviles definidos en la tabla(4.6).

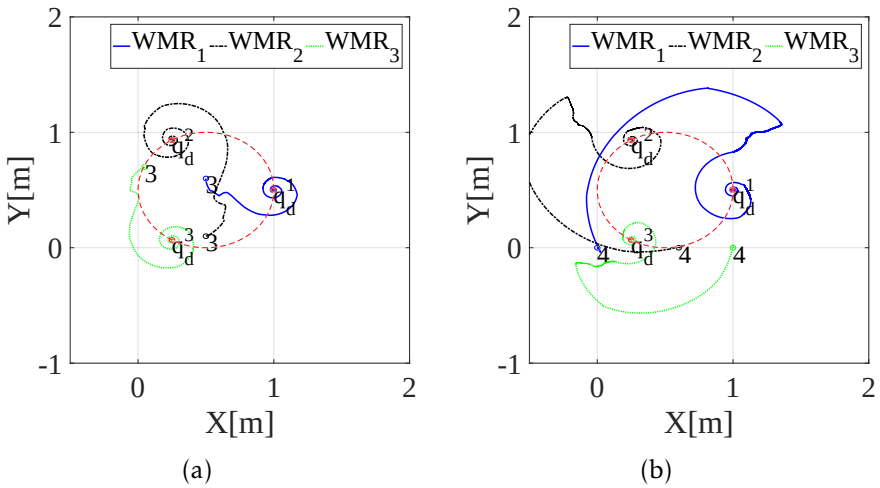


Figura 4.83: Validación: Trayectorias de los escenarios seleccionados 3 y 4 para 3 robots móviles definidos en la tabla(4.6).

4.3.4 *Validación númerica usando cuatro robots*

En esta sección se valida el caso de 4 robots, los cuales deben llegar a la meta en el tiempo entrenado y a su vez, los robots deben llegar a las metas deseadas evadiendose entre ellos. Para este caso de análisis el radio del área deseada es 0,5m.

4.3.4.1 *Condiciones iniciales*

Tabla 4.8: Validación: Condiciones iniciales seleccionadas para 4 robots.

Condiciones iniciales				
α	WMR_1	WMR_2	WMR_3	WMR_4
#	$q^1(0)$	$q^2(0)$	$q^3(0)$	$q^4(0)$
1	$[0.00, 0.10]^T$	$[0.00, 0.00]^T$	$[0.95, 0.60]^T$	$[0.70, 0.70]^T$
2	$[0.41, 0.03]^T$	$[0.65, 0.10]^T$	$[0.75, 0.70]^T$	$[0.10, 0.80]^T$
3	$[0.42, 0.92]^T$	$[0.74, 0.39]^T$	$[0.66, 0.04]^T$	$[0.85, 0.93]^T$
4	$[0.25, 0.10]^T$	$[0.00, 0.15]^T$	$[0.65, 0.60]^T$	$[0.35, 0.70]^T$

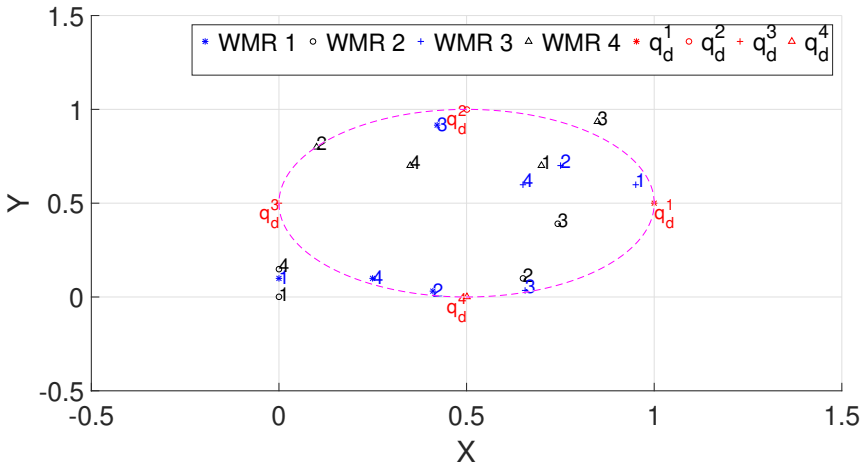


Figura 4.84: Distribución de las condiciones iniciales a lo largo del eje coordenado.

Tabla 4.9: Validación: Tiempo de llegada de 4 escenarios para cuatro robots móviles.

Tiempo de llegada para la tarea de **encuentro**, t_a= 4 segundos								
	$u^i = u_f^i$				$u^i = u_f^i + u_{l_1}$			
α	WMR_1	WMR_2	WMR_3	WMR_4	WMR_1	WMR_2	WMR_3	WMR_4
1	4.3520	4.1730	3.8810	3.6070	4.2990	4.2600	4.1350	3.7190
2	4.2240	3.9630	3.6760	4.6590	3.9010	4.2370	3.7930	4.0290
3	4.3200	3.6870	3.7080	4.6400	3.7010	3.9130	3.8780	4.1310
4	4.1450	4.0110	3.5070	4.5960	3.9560	4.3240	3.7870	3.6940
Avg	4.2603	3.9585	3.6930	4.3755	3.9643	4.1835	3.8983	3.8933

4.3.4.2 *Resultados*

Como se puede apreciar en la tabla(4.9), el controlador ajusta la navegación para que los robots móviles cumplan con la premisa de tiempo, en la figura(4.85) se aprecia de mejor manera como se ajustan los tiempos al deseado. Por otro lado, en la figura(4.86) se muestran las trayectorias que siguen los robots para llegar a sus respectivas metas y en la figura(4.87) se muestra la dinámica del error para cada robot móvil.

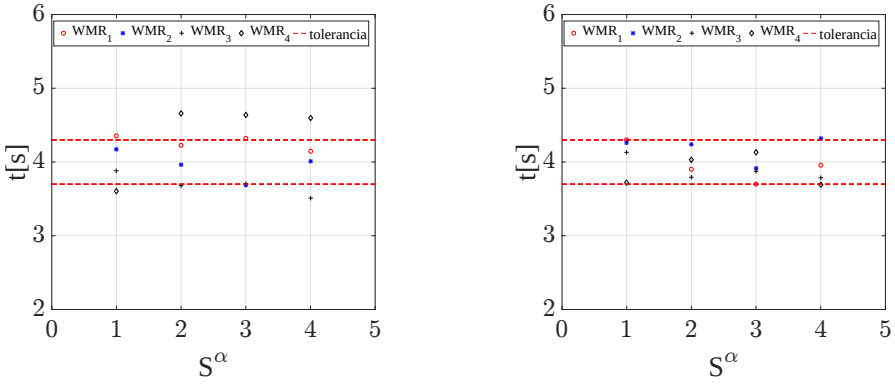


Figura 4.85: Validación de los tiempos de llegada para los cuatro robots móviles; en el caso (a) se muestra el tiempo de llegada con comportamiento forzado y el caso(b) se muestran los tiempos de llegada con comportamiento aprendido.

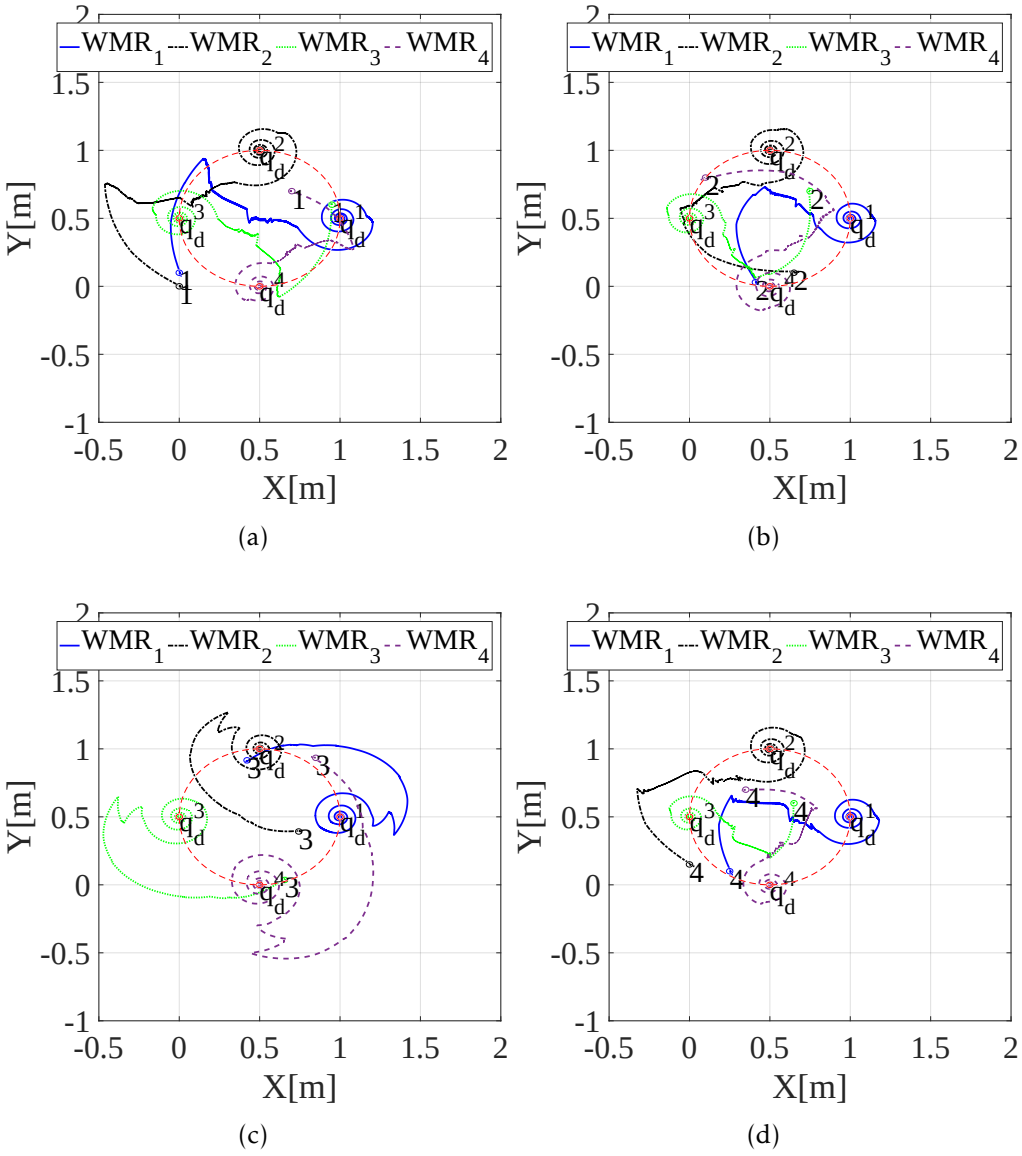


Figura 4.86: Validación: Trayectorias de los 4 escenarios para los cuatro robots móviles definidos en la tabla(4.8).

-

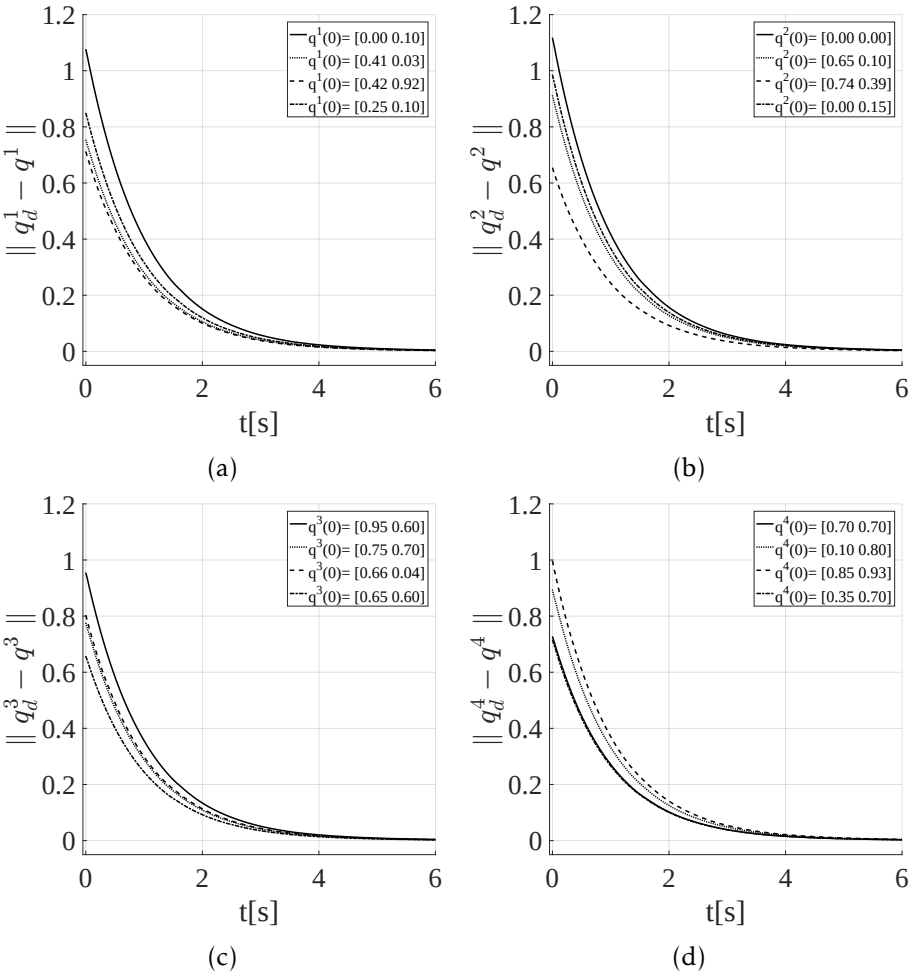


Figura 4.87: Validación: Cumplimiento de la tarea de **encuentro** con evasión de obstáculos y tiempo de llegada optimizado, para las condiciones iniciales propuestas en la tabla(4.8). (a) Norma de error del WMR_1; (b) Norma de error del WMR_2; (c) Norma de error del WMR_3; (d) Norma de error del WMR_4;.

5

Conclusiones y trabajo futuro

5.1 Discusiones

- Esta propuesta se puede extender hacia una aplicación práctica, donde la carga computacional para cualquier sistema embebido es pequeña.
- Los robots móviles puede moverse libremente hacia la posición deseada.
- Los robots móviles son tratados como obstáculos dinámicos, donde cada robot solo conoce la posición y el radio de los otros robots, esto permite añadir n robots y m obstáculos sin afectar los objetivos de control.
- La metodología de control por comportamientos analíticos abordado en este proyecto, sienta una base para el diseño de algoritmos evolutivos multi-objetivo aplicados en sistemas multi-robot.

5.2 Trabajo futuro

- Implementación de la metodología para la síntesis de controladores mediante comportamientos analíticos para sistemas multirobot aplicados en el entorno CUDA con enfoque en el lenguaje de programación python.
- Aplicación de la metodología de control por comportamientos analíticos en sistemas multi-robot para resolver tareas de navegación en otras áreas de la robótica.

- Diseño de un algoritmo evolutivo multi-objetivo capaz de proporcionar al menos dos individuos capaces de cumplir con objetivos de control predefinidos.
- Implementación de la metodología de comportamientos analíticos para sistemas multirobot en sistemas embebidos de bajo consumo.
- Implementación de la metodología de comportamientos analíticos en un sistemas en tiempo real capaz de generar un aprendizaje basado en las características del entorno.

5.3 Conclusión

El uso de la metodología de control por comportamientos analíticos permite analizar problemas en robótica que pueden ser difíciles de resolver mediante métodos tradicionales. La principal característica de esta metodología, permite obtener representaciones matemáticas del sistema, asegurando que los objetivos definidos se cumplan debido a su relación con la teoría de control. Esto se puede implementar tanto en sistemas multirobot como en sistemas de un solo robot, la principal desventaja esta ligada a una mala sintonización de parámetros al momento de plantear el problema. Si el problema no esta bien definido la metodología arrojará soluciones erroneas, por el contrario, si el problema se definió de manera correcta y los parámetros se eligierón de manera adecuada, las soluciones encontradas tienen una alta probabilidad de representar una solución general.

Debido a esto, es necesario analizar minusionamente el problema para poder equilibrar las soluciones obtenidas por la metodología y es aquí donde tomá importancia la teoría de control. Las herramientas que proporciona la teoría de control permite diseñar sistemas de control que cumplan con los objetivos propuestos y al añadir la parte evolutiva se crea un entorno capaz de proveer soluciones a diversos problemas en un tiempo relativamente mas corto, que además, esten respaldadas por las matemáticas. De esta manera, se pueden abordar problemas en robótca como el de navegación autónoma basada en **encuentro**, gracias a las caracerísticas del control por comportamientos analíticos se pueden abordar problemas con un enfoque generalizado sin la necesidad de recurrir a los métodos evolutivos cada vez que se cambia un parámetro al sistema. Por otro lado, una de las grandes ventajas de la metodología de control por com-

portamientos analíticos, es que respeta siempre las características establecidas mediante teoría de control, inclusive al modificar el controlador original.

Finalmente, esta metodología permite manipular el comportamiento del sistema para que cumpla con los objetivos deseados, sin afectar la convergencia y así obtener un modelo matemático del comportamiento del sistema. Para el caso donde la metodología de control por comportamientos analíticos parte de dos robots, generó las bases de una metodología aplicada a sistemas multi-robot. Con el modelo mas simple del sistema multirobot para nevegación basada en el problema **encuentro**, fue posible proponer una metodología con la capacidad de ser ampliada a sistemas donde se puedan agregar más robots u obstáculos. Las soluciones encontradas, demostrarón que al hacer uso del control por comportamientos analíticos es posible obtener el comportamiento general del sistema, en forma analítica. Siendo una métodología que permite analizar el cumplimiento de diversos objetivos mediante métodos analíticos tradicionales y así asegurar que siempre se cumplán debido a sus bases en la teoría de control.

Ahvanooey, M. T., Li, Q., Wu, M., and Wang, S. (2019). A survey of genetic programming and its applications. *KSII Transactions on Internet and Information Systems*, 13(4):1765–1794.

Alitappeh, R. J., Jeddisaravi, K., and Guimarães, F. G. (2017). Multi-objective multi-robot deployment in a dynamic environment. *Soft Computing*, 21(21):6481–6497.

Alonso-Mora, J., Knepper, R., Siegwart, R., and Rus, D. (2015). Local motion planning for collaborative multi-robot manipulation of deformable objects. *Proceedings - IEEE International Conference on Robotics and Automation*, 2015-June(June):5495–5502.

Anggraeni, P., Defoort, M., Djemai, M., and Zuo, Z. (2019). Control strategy for fixed-time leader–follower consensus for multi-agent systems with chained-form dynamics. *Nonlinear Dynamics*, 96(4):2693–2705.

Arkin, R. C. (1998). *Behavior-Based Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press.

Ayoub, A., El-Shorbagy, M., Eldesoky, I., and Mousa, A. a. (2020). *Cell Blood Image Segmentation Based on Genetic Algorithm*, pages 564–573. Springer.

Bae, H., Kim, G., Kim, J., Qian, D., and Lee, S. (2019). Multi-robot path planning method using reinforcement learning. *Applied Sciences (Switzerland)*, 9(15).

Bayindir, L. (2016). A review of swarm robotics tasks. *Neurocomputing*, 172:292–321.

Beasley, D., Bull, D. R., and Martin, R. R. (1993). An overview of genetic algorithms: Part 1, fundamentals. *University computing*, 15(2):56–69.

Becerra, H. M., Colunga, J. A., and Romero, J. G. (2016). Robust trajectory tracking controllers for pose-regulation of wheeled mobile robots. *IEEE International Conference on Intelligent Robots and Systems*, 2016-Novem:1041–1047.

Bechlioulis, C. P. and Kyriakopoulos, K. J. (2018). Collaborative multi-robot transportation in obstacle-cluttered environments via implicit communication. *Frontiers Robotics AI*, 5(AUG):1–17.

Borenstein, J. and Koren, Y. (1989). Real-time obstacle avoidance for fast mobile robots. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(5):1179–1187.

Brooks, R. A. (1991). Intelligence without representation. *Artificial Intelligence*, 47(1):139 – 159.

Chang, S., Wang, Y., Zuo, Z., and Yang, H. (2021). Fixed-Time Formation Control for Wheeled Mobile Robots With Prescribed Performance. *IEEE Transactions on Control Systems Technology*, pages 1–8.

Clemente, E., Meza-Sánchez, M., Bugarin, E., and Aguilar-Bustos, A. Y. (2018). Adaptive Behaviors in Autonomous Navigation with Collision Avoidance and Bounded Velocity of an Omnidirectional Mobile Robot: A Control Theory with Genetic Programming Approach. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 92(2):359–380.

Colunga, J. A., Becerra, H. M., Vázquez, C. R., and Gómez-Gutiérrez, D. (2020). Robust leader-following consensus of high-order multi-agent systems in prescribed time. *IEEE Access*, 8:195170–195183.

Ding, S., Li, H., Su, C., Yu, J., and Jin, F. (2013). Evolutionary artificial neural networks: a review. *Artificial Intelligence Review*, pages 1–10.

Eun Soo Jang, Seul Jung, and Hsia, T. C. (2005). Collision avoidance of a mobile robot for moving obstacles based on impedance force control algorithm. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 382–387.

Francis Manfredi Maggiore, B. A. (2016). Flocking and Rendezvous in Distributed Robotics. Technical report, University of Toronto.

Friudenberg, P. and Koziol, S. (2018). Mobile Robot Rendezvous Using Potential Fields combined With Parallel Navigation. *IEEE Access*, 6:16948–16957.

Gasparri, A., Priolo, A., and Ulivi, G. (2012). A swarm aggregation algorithm for multi-robot systems based on local interaction. *Proceedings of the IEEE International Conference on Control Applications*, pages 1497–1502.

Ge, S. and Cui, Y. (2002). Dynamic motion planning for mobile robots using potential field method. *Autonomous Robots*, 13:207–222.

Guldner, J. and Utkin, V. I. (1995). Sliding mode control for gradient tracking and robot navigation using artificial potential fields. *IEEE Transactions on Robotics and Automation*, 11(2):247–254.

Hawley, L. and Suleiman, W. (2019). Control framework for cooperative object transportation by two humanoid robots. *Robotics and Autonomous Systems*, 115:1–16.

Ichikawa, Y. and Ozaki, N. (1985). *Autonomous Mobile Robot.*, volume 2. IFAC.

Khatib, O. (1980). *Commande Dynamique dans l'Espace Opérationnel des Robots Manipulateurs en Présence d'Obstacles*. PhD thesis, École Nationale Supérieure de l'Aéronautique et de l'Espace (ENSAE), Tolouse, France.

Khatib, O. (1985). Real-time obstacle avoidance for manipulators and mobile robots. In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, volume 2, pages 500–505.

Kim, D. W., Lasky, T. A., and Velinsky, S. A. (2013). Autonomous Multi-mobile Robot System : Simulation and Implementation using Fuzzy Logic. *International Journal of Control, Automation and Systems*, 11:545–554.

Kim, J. and Khosla, P. K. (1992). Real-time obstacle avoidance using harmonic potential functions. *IEEE Transactions on Robotics and Automation*, 8(3):338–349.

Koza, J. R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, USA.

Kumar, A., Sharma, S., Tiwari, R., and Majumdar, S. (2012). Area exploration by flocking of multi robot. *Procedia Engineering*, 41(Iris):377–382.

Kumar, R. (2012). Blending roulette wheel selection & rank selection in genetic algorithms. *International Journal of Machine Learning and Computing*, 2(4):365.

Kunwar, F. and Benhabib, B. (2006). Rendezvous-guidance trajectory planning for robotic dynamic obstacle avoidance and interception. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 36(6):1432–1441.

Langdon, W. B. and Poli, R. (2002). *Foundations of Genetic Programming*. Springer.

Li, Q., Wei, J., Gou, Q., and Niu, Z. (2021). Distributed adaptive fixed-time formation control for second-order multi-agent systems with collision avoidance. *Information Sciences*, 564:27–44.

Luo, S., Kim, J., Parasuraman, R., Han, J., Matson, E. T., Min, B.-c., and Korea, S. (2019). Ad Hoc Networks Multi-robot rendezvous based on bearing-aided hierarchical tracking of network topology. *Ad Hoc Networks*, 86:131–143.

Masoud, A. A. (2013). A harmonic potential field approach for joint planning and control of a rigid, separable nonholonomic, mobile robot. *Robotics and Autonomous Systems*, 61(6):593–615.

Mataric, M. (1994). *Interaction and Intelligent Behavior*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA. AAI0575115.

Matarić, M. (1995). Designing and understanding adaptive group behavior. *Adaptive Behavior*, 4:51–80.

Matarić, M. and Michaud, F. (2008). Behavior-based systems. In Siciliano, B. and Khatib, O., editors, *Springer Handbook of Robotics*, pages 891–909. Springer Berlin Heidelberg.

Melanie, M. (1999). An introduction to genetic algorithms. *Cambridge, Massachusetts London, England, Fifth printing*, 3:62–75.

Meza-Sánchez, M., Clemente, E., Rodríguez-Liñán, M. C., and Olague, G. (2019). Synthetic-analytic behavior-based control framework: Constraining velocity in tracking for nonholonomic wheeled mobile robots. *Information Sciences*, 501:436–459.

Meza-Sánchez, M., Clemente, E., Villalvazo, R., and Olague, G. (2019). Bounding Velocity in Tracking Control of Unicycle Mobile Robots with Genetic Programming. In *2018 20th Congreso Mexicano de Robotica, COMRob 2018*.

Mohammad, S., Rostami, H., Sangaiah, A. K., and Wang, J. (2019). Obstacle avoidance of mobile robots using modified potential field algorithm. *Eurasip Journal on Wireless Communications and Networking*, 2019(1):1–19.

Muñoz-Vázquez, A. J., Parra-Vega, V., Sánchez-Orta, A., and Sánchez-Torres, J. D. (2021). Adaptive Fuzzy Velocity Field Control for Navigation of Nonholonomic Mobile Robots. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 101(2).

Méndez, M. (2008). *Algoritmos evolutivos y preferencia del decisor aplicados a problemas de optimización multiobjetivos discretos*. PhD thesis, Escuela de Ingenierías Industriales y Civiles.

Nedjah, N. and Silva, L. (2019). Review of methodologies and tasks in swarm robotics towards standardization. *Swarm and Evolutionary Computation*, 50(August):100565.

Osorio-Comparán, R., López-Juárez, I., Reyes-Acosta, A., Pena-Cabrera, M., Bustamante, M., and Lefranc, G. (2016). Mobile robot navigation using potential fields and lma. In *2016 IEEE International Conference on Automatica (ICA-ACCA)*, pages 1–7.

Otte, M., Correll, N., and Frazzoli, E. (2013). Navigation with foraging. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3150–3157.

Peñaloza-Mejía, O., Clemente, E., Meza-Sánchez, M., Pérez, C., and Chavez, F. (2017). Gp-based motion control design for the double-integrator system subject to velocity constraint. In *GECCO '17 Companion*, Berlin, Germany.

Peñaloza-Mejía, O., Clemente, E., Meza-Sánchez, M., and Pérez, C. B. (2019). Evolving behaviors for bounded-flow tracking control of second-order dynamical systems. *Engineering Applications of Artificial Intelligence*, 78:12 – 27.

Poli, R., Langdon, W. B., and McPhee, N. F. (2008a). *A field guide to genetic programming*. Published via `http://lulu.com` and freely available at `http://www.gp-field-guide.org.uk`. (With contributions by J. R. Koza).

Poli, R., McPhee, N. F., and Vanneschi, L. (2008b). Elitism reduces bloat in genetic programming. *GECCO'08: Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation 2008*, pages 1343–1344.

Sahare, V. and Sahare, N. (2010). An Approach Based on Clustering Method for Object Finding Mobile Robots Using ACO. In *2010 Second International Conference on Machine Learning and Computing*. IEEE.

Sahni, Y., Cao, J., and Jiang, S. (2019). *Middleware for Multi-robot Systems*. Number January 2020 in Studies in Systems, Decision and Control. Springer International Publishing.

Shang, Y. and Huang, J. (2020). Fixed-Time Stabilization of Spatial Constrained Wheeled Mobile Robot via Nonlinear Mapping. *IAENG International Journal of Applied Mathematics*, 50(4).

Siciliano, B. and Khatib, O. (2007). *Springer Handbook of Robotics*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Smirnova, E., Stepanov, D., and Goryunov, V. (2015). A technique of natural visual landmarks detection and description for mobile robot cognitive navigation. *Annals of DAAAM and Proceedings of the International DAAAM Symposium*, 2015-Janua(February 2017):905–911.

Sperati, V., Trianni, V., and Nolfi, S. (2011). Self-organised path formation in a swarm of robots. *Swarm Intelligence*, 5(2):97–119.

Thede, S. M. (2004). An introduction to genetic algorithms. *Journal of Computing Sciences in Colleges*, 20(1):115–123.

Urakubo, T. (2018). Stability Analysis and Control of Nonholonomic Systems with Potential Fields. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 89(1-2):121–137.

Vardy, A., Vorobyev, G., and Banzhaf, W. (2014). Cache consensus: Rapid object sorting by a robotic swarm. *Swarm Intelligence*, 8(1):61–87.

Villalvazo-Covián, R., Meza-Sánchez, M., Clemente, E., Rodríguez-Liñán, M. C., Monay-Arredondo, L., and Olague, G. (2021). Position control with dynamic obstacle avoidance in omnidirectional mobile robots. In *2021 XXIII Robotics Mexican Congress (ComRob)*, pages 44–49.

Whigham, P. A. and Dick, G. (2010). Implicitly controlling bloat in genetic programming. *IEEE Transactions on Evolutionary Computation*, 14(2):173–190.

Yu, X. and Gen, M. (2010). *Introduction to evolutionary algorithms*. Springer Science & Business Media.

Zavlangas, P. G. and Tzafestas, S. G. (2003). Motion control for mobile robot obstacle avoidance and navigation: a fuzzy logic-based approach. *Systems Analysis Modelling Simulation*, 43(12):1625–1637.

Zhang, H., Gong, W., Li, B., Niu, Y., and Yang, Y. (2020). Finite-time Trajectory Tracking Control of A Wheel Mobile Robot based on Integral Terminal Sliding Mode. *IEEE International Conference on Control and Automation, ICCA*, 2020-Octob(1):70–75.

Zhao, L., Jin, J., and Gong, J. (2021). Robust zeroing neural network for fixed-time kinematic control of wheeled mobile robot in noise-polluted environment. *Mathematics and Computers in Simulation*, 185:289–307.