ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELECTRÍCA Y ELECTRÓNICA

DISEÑO, CONTROL Y SIMULACIÓN, MEDIANTE EL USO DEL ENTORNO ROS/GAZEBO, DE UN SISTEMA MULTI-AGENTE CONFORMADO POR PLATAFORMAS TURTLEBOT3 ENFOCADO A MANTENER UNA FORMACIÓN LÍDER SEGUIDOR (TOMO 2)

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE. INGENIERA EN ELECTRÓNICA Y AUTOMATIZACIÓN

NATHALY GABRIELA ROMERO CALLE

nathaly.romero@epn.edu.ec

DIRECTOR: ING. PATRICIO JAVIER CRUZ DÁVALOS, PhD.

patricio.cruz@epn.edu.ec

DMQ, mes año

CERTIFICACIONES

Yo, Nathaly Gabriela Romero Calle declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

NATHALY GABRIELA ROMERO CALLE

Certifico que el presente trabajo de integración curricular fue desarrollado por Nathaly Gabriela Romero Calle, bajo mi supervisión.

Ing. Patricio Javier Cruz Dávalos, PhD.
DIRECTOR

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como los productos resultantes del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

Nathaly Gabriela Romero Calle

Rodrigo Alejandro Ayala Pavón (Colaborador)

Ing. Patricio Javier Cruz Dávalos, PhD.

DEDICATORIA

Todo mi trabajo, esfuerzo y tiempo invertido en este proyecto junto con el de mi compañero Rodrigo Ayala, va dedicado a mi familia en especial a mis padres y hermanos, que con gran esfuerzo me acompañaron y apoyaron en este trayecto llamado educación, que no acaba aquí, en realidad comienza.

También se lo dedico a mis amigos, maestros que me motivaron y me compartieron no solo sus conocimientos sino también su sabiduría para forjarme como profesional y como persona.

Por último, dedico mi trabajo a todos los jóvenes en busca de un reto de aprender algo nuevo. Es la única forma de descubrir de lo qué somos capaces.

¡Arriésguense a hacerlo!

AGRADECIMIENTO

Mis más grandes agradecimientos a mis padres que me han apoyado en cada decisión que he tomado y me han ayudado a llegar tan lejos como yo he deseado.

También, agradezco profundamente a la Escuela Politécnica Nacional por acogerme y permitirme conocer a personas que me ayudaron y colaboraron a lo largo de mi vida universitaria y también en el desarrollo de este trabajo de titulación. En especial al Ing. Patricio Javier Cruz Dávalos, PhD., y al MSc. Diego Maldonado, sin su ayuda y paciencia todo esto no hubiera sido posible.

A mis mejores amigos Marco y Rodrigo, gracias por su amistad y amor incondicional. No puedo describir todo lo que han hecho por mí ni todo lo que me han enseñado.

Finalmente, a todas las personas, colegas y compañeros que brindaron su tiempo y compañía para ayudarme a alcanzar el éxito.

ÍNDICE DE CONTENIDO

RESUMEN

El auge de la robótica colaborativa ha venido generando gran interés por las ventajas que representa el ocupar varios robots para que desarrollen diferentes tareas. Por esta razón, en el presente trabajo se ha enfocado en el tema referente a sistemas multi-agentes robóticos homogéneos, donde varios robots o agentes, de características similares, ejecutan acciones de modo que realizan una tarea en conjunto, como es la formación de robots. Esta se desarrolla con el fin de suplir ayuda en aspectos que representen un gran riesgo, como por ejemplo, la inspección de campos minados, transporte de objetos pesados o peligrosos, entre otros.

Existen diversos algoritmos para coordinar la formación, una de ellas y, tal vez, la más utilizada es la técnica líder seguidor. Esta se caracteriza porque el robot seguidor debe posicionarse y mantener una distancia y ángulo definido respecto del robot líder. Por esto este trabajo se centra en la aplicación de esta técnica para permite al usuario controlar la formación de robots, considerando 4 posibles configuraciones: fila, columna, V y diamante.

Para este fin se ha empleado la versión virtual del robot móvil de tracción diferencial TurtleBot3, junto con los softwares de código abierto ROS y Gazebo. De esta forma se implementa un escenario en el que se ejecutan las acciones de control para cada uno de los agentes del sistema para que puedan moverse y realizar las formaciones comandadas por el usuario.

PALABRAS CLAVE: Formación Líder Seguidor, Esquema Clásico, Referencia Virtual, Sistema Multi-agente, TurtleBot3, ROS, Gazebo

ABSTRACT

The research perspectives in collaborative robotics have been generating great interest because of the advantages of using several robots to perform different complex tasks. For this reason, this project focuses on the development of an homogeneous robotic multi-agent system where several robots or agents of similar characteristics execute actions in order to perform a task together, such as robot formation. This application is generally developed in order to help in aspects that could represent a substantial risk, such as minefield inspection, the transport of heavy or dangerous objects, among others.

There are several techniques to coordinate the formation of a robotic groups, one of them and possibly the most used is the follower-leader approach. In this one, the follower robot must maintain a position at a defined distance and angle with respect to the leader robot. Therefore, this work is based on this technique and it is applied to allow a user to control the formation through the leader robot and considering 4 possible configurations: row, column, V and diamond.

To achieve this goal, this project employs the virtual version of mobile robot platform TurtleBot3 and it is used together with the open source software ROS and Gazebo. In this fashion, a virtual scenario is implemented where it is possible to execute the control actions for the different agents of the robotic group, so they can move and perform the formations commanded by the user.

KEYWORDS: Leader-follower formation, Classic Scheme, Virtual Reference, Multi-agent System, TurtleBot3, ROS, Gazebo

1 INTRODUCCIÓN

Este documento corresponde al segundo tomo del trabajo desarrollado en conjunto por los autores Rodrigo Ayala y Gabriela Romero. Por lo que se sugiere leer inicialmente el primer tomo [1], para tener más claridad acerca de temas relacionados con el proyecto desarrollado, como es el caso de los softwares: ROS, Gazebo, la plataforma TurtleBot3, entre otros.

Similar al tomo 1, el presente trabajo está orientado hacia los robots móviles con ruedas, en particular, se centra en un tipo de robot móvil de tracción diferencial. Esta plataforma móvil posee dos ruedas fijas alineadas sobre un mismo eje y existen muchas aplicaciones en las que se ha utilizado este tipo de robot, como: transportación de objetos, exploración en distintos ambientes, reconocimiento de entornos, etc. Sin embargo, las tareas que realice un solo robot se verán limitadas por las capacidades presentes en el mismo; es aquí, donde surge la robótica cooperativa, dando lugar a los sistemas multi-agentes o también denominados como multi-robots [2].

Los sistemas multi-agentes se caracterizan por ser un conjunto de múltiples elementos informáticos, también llamados agentes. Como se lo menciona anteriormente, en este trabajo se emplean robots móviles de tracción diferencial como agentes, donde cada uno tiene como objetivo trabajar de forma cooperativa de modo que se logre cumplir con una tarea en conjunto. Dependerá de los tipos de agente con los que se trabaje para definir si el sistema multi-agente es homogéneo o heterogéneo. Esto quiere decir que, si pertenecen al mismo grupo de robots serán homogéneos caso contrario, heterogéneos [2], [3].

Las diferentes aplicaciones que se pueden desarrollar con sistemas multi-agentes se basan principalmente en las ventajas que estos prestan, como la escalabilidad, flexibilidad, autonomía, robustez, entre otras [4]. Es por esta razón que la utilización de sistemas multi-agentes da como resultado una solución robusta ante problemas que se puedan resolver de forma cooperativa. Justamente, una aplicación clásica es realizar movimientos coordinados y mantener una determinada organización o formación. Dentro de la literatura se han propuesto diferentes tipos de estructuras para emplear un control de formación. Posiblemente, el esquema más utilizado es la formación con el método líder-seguidor, cuyo funcionamiento se enfoca básicamente en definir previamente la distancia y el ángulo del robot seguidor con respecto al robot líder [5]. Dependiendo de cómo se aborde el problema de formación se encontrarán ciertas limitaciones al no poder mantener ya sea el ángulo o la distancia deseada. De modo que en este proyecto se propone utilizar una referencia

virtual en cada uno de los agentes seguidores para conseguir mayor robustez en el sistema al formar al conjunto de robots [6].

Se debe agregar que las herramientas computacionales que se necesitan para resolver la formación de un sistema multi-agente homogéneo serán los softwares de código abierto ROS y Gazebo, junto con la plataforma estándar TurtleBot3. Este robot móvil de tracción diferencial es usado activamente en áreas de investigación y educación por ser programable y de tamaño reducido. Por todo esto, este proyecto tiene como producto final demostrable un entorno de simulación basado en ROS y Gazebo, en el que se pueda comprobar el funcionamiento del control de formación líder seguidor con referencia virtual, para un sistema multi-agente homogéneo de robots móviles de tracción diferencial, empleando la plataforma Turtlebot3.

1.1 OBJETIVO GENERAL

Diseñar, controlar y simular, mediante el uso del entorno ROS/Gazebo, un sistema multi-agente conformado por plataformas TurtleBot3 enfocado a mantener una formación líder seguidor.

1.2 OBJETIVOS ESPECÍFICOS

Cabe indicar que, dado que este es el segundo tomo del trabajo desarrollado, se presentan los objetivos específicos acorde al plan aprobado pero enfocados a lo detallado en este tomo.

- Investigar las características de una formación líder seguidor y sus algoritmos en un sistema multi-agente homogéneo.
- Implementar en ROS/Gazebo un sistema multi-agente basado en el robot virtual TurtleBot3.
- Diseñar un algoritmo de control que realice una formación líder seguidor donde los robots seguidores mantengan una distancia y ángulo definidos respecto del robot líder.
- Realizar diferentes formaciones (línea, columna, V, diamante) para verificar el funcionamiento del esquema líder seguidor y determinar las restricciones del algoritmo de control implementado para su posterior análisis de resultados.

1.3 ALCANCE

Cabe indicar que, dado que este es el segundo tomo del trabajo desarrollado, se presentan los alcances acorde al plan aprobado que están relacionados con lo detallado en este tomo.

- Se realizará una investigación de la formación líder seguidor enfocado en sistemas multi-agentes homogéneos con robots móviles de tracción diferencial y el manejo del robot líder por un usuario.
- Se diseñará el algoritmo de control de formación basado en referencia virtual con el fin de garantizar que los robots seguidores se encuentren a una distancia y ángulo definidos respecto del robot líder, el cual será manejado por un usuario.
- Se implementará en ROS/Gazebo un sistema multi-agente homogéneo con una formación líder seguidor en un entorno libre de obstáculos diseñado previamente con un número mínimo de dos robots seguidores y un robot líder.
- Se realizará pruebas en un entorno de trabajo libre de obstáculos con diferente tipo de formación (fila, columna, V, diamante) con el fin de comprobar que se mantiene la formación líder seguidor en el entorno simulado.

1.4 MARCO TEÓRICO

1.4.1 SISTEMAS MULTI-AGENTES

Las estrategias de modelado, la capacidad computacional y los lenguajes de programación han evolucionado y se han ido enlazado entre sí. Esto ha permitido evolucionar de sistemas que se pensaban como un todo indivisible a sistemas con un enfoque de agregación de elementos autónomos, que se modelan y programan como agentes.

Para definir lo que es un sistema multi-agente en primer lugar se deberá definir qué es un agente. Si bien existen diversas definiciones, se entiende de manera general que, un agente es un sistema en un ambiente o entorno y que es capaz de realizar acciones de forma autónoma dentro del mismo, con el fin de alcanzar sus metas u objetivos [7].

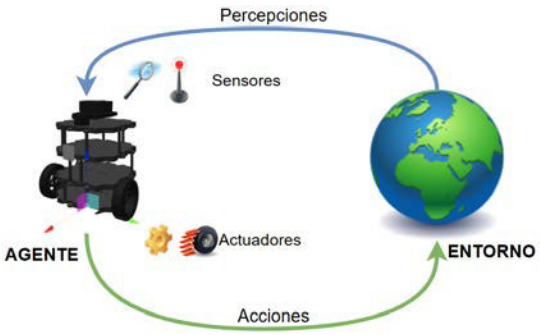


Figura 1.1. Diagrama de interacción de un agente [Fuente propia]

En la Figura 1.1 se puede apreciar de mejor manera cómo interactúa un agente con el entorno a través de percepciones y acciones, cabe notar que la representación del agente posee internamente sensores y actuadores que ayuden a completar su objetivo. Un agente, como el robot mostrado en la Figura 1.1, además de poseer sensores y actuadores es necesario que cumpla con las siguientes propiedades:

- mantener interacción constante con el entorno o ambiente,
- su comportamiento debe estar basado en cumplir con su objetivo o meta,
- intercambiar información con otros agentes,
- poseer recursos propios, y
- tener al menos un hilo de ejecución.

Teniendo en cuenta cómo está representado un agente, en consecuencia, un sistema multi-agente (SMA) es un conjunto de varios agentes, en este caso robots móviles, que interactúan entre sí con el fin de cumplir con el objetivo dado. De esta forma, se tiene que un SMA cumple con las siguientes características [8]:

- no posee un sistema global que controle al grupo de agentes,
- los datos no están centralizados, y
- cada agente puede dar solución parcial al problema.

En un SMA la asignación de tareas y recursos se enfoca en realizar un trabajo distribuido, en otras palabras, cada robot se encarga de obtener datos de los robots vecinos de forma individual. Es por esto que se deben definir los roles de cada uno de los agentes ya que pueden proveer, servir o mediar información. Una vez definido los roles de los agentes es importante analizar también la planificación de actividades que van a tener. Al no planificar las tareas correctamente el problema se verá reflejado al aumentar a un mayor número de agentes, de modo que se intensifica su complejidad, ya que necesitarán de replanificaciones que ayuden a evitar alteraciones en la ejecución de acciones de cada robot que conforme el sistema [9].

1.4.1.1 Clasificación de los Sistemas Multi-Agente

Los sistemas multi-agente abarcan un campo bastante extenso y se pueden clasificar de muchas formas, una de ellas es presentada en la Figura 1.2. En esta se observa la división entre sistemas multi-robóticos y sistemas no robóticos con múltiples elementos como, por

ejemplo, un sistema computacional. Dentro del campo de los sistemas multi robóticos, se clasifican en tres grandes categorías que se describen brevemente a continuación [10].

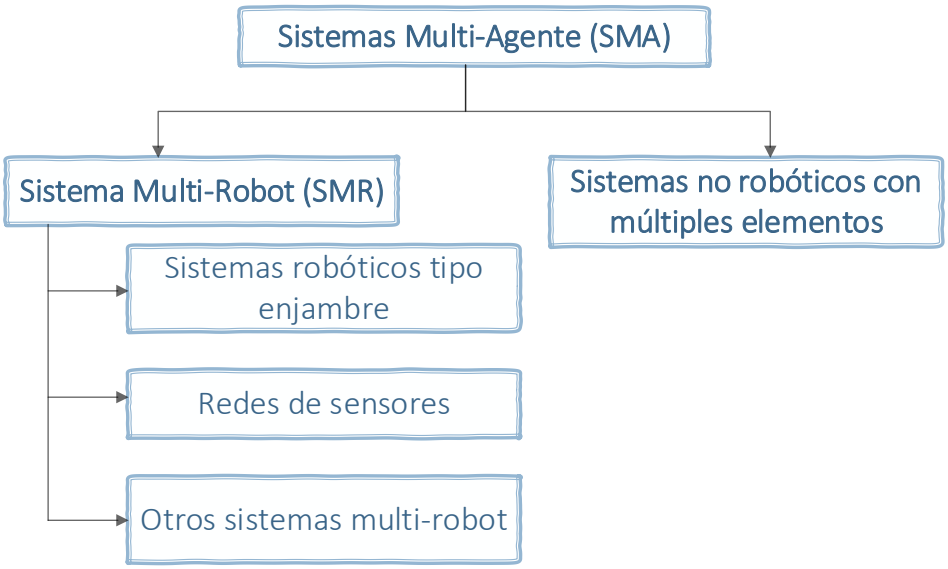


Figura 1.2. Clasificación de sistemas multi-agente [Fuente propia].

Los sistemas multi-robot (SMR) se definen como un grupo de robots enfocados en la realización de una tarea común. Estos sistemas son utilizados cuando un solo robot no es capaz de desempeñar una tarea específica, y se requiere de grupos robots. Por ejemplo, el transporte de una carga pesada [11]. La diferencia entre otros sistemas y los tipo enjambre radica en el tamaño de la población y su arquitectura de control. El número de agentes de los SMR es pequeño comparado con los sistemas tipo enjambre. Por otra parte, respecto a la arquitectura de control, en un SMR se lo hace usualmente bajo una sola entidad que hace de sistema centralizado [10]; mientras que, en un tipo enjambre se tiene una arquitectura descentralizada o distribuida. Siendo así los sistemas tipo enjambre más robustos, flexibles, escalables y con poblaciones grandes.

Con respecto a las redes de sensores, son consideradas como SMR ya que en su gran mayoría son sensores móviles que pueden comunicarse efectivamente y de forma inalámbrica. En [12], esta subdivisión consiste en cientos e incluso miles de sensores distribuidos en un determinado campo de aplicación. Cada agente puede participar activamente en el monitoreo y la comunicación con otros miembros. Esta información puede ser por ejemplo la densidad atmosférica en una determinada área.

Desde otro punto de vista, los sistemas multi-agente pueden clasificarse en función del tipo de entes que lo conforman. Siendo así, homogéneos si los agentes comparten entre sí el mismo modelo, estructura, dinámica, capacidades y poseen características de cómputo y

sensado similares como se observa en la Figura 1.3(a), donde todos los robots son móviles de tracción diferencial con un sistema de locomoción mediante ruedas. Mientras que se consideran sistemas multi-agente heterogéneos cuando las características mencionadas anteriormente difieren entre agentes, como se muestra en la Figura 1.3(b). En el campo de la robótica, un gran número de aplicaciones está enfocado al desarrollo de sistemas multi-agente por la flexibilidad y robustez que presentan. No obstante, hay que considerar que estos sistemas requieren algoritmos de consenso para poder establecer comunicación entre agentes [13].

Figura 1.3. (a) Sistemas multi-agente homogéneos (b) Sistemas multi-agente heterogéneos [Fuente propia]

1.4.1.2 Control de Sistemas Multi-Robot

En un sistema multi-robot, controlar y monitorear directamente cada agente por separado resultaría contraproducente y en determinados casos llegaría a ser imposible a medida que se escala el sistema a un mayor número de robots [14]. Por ello, es importante la implementación de algoritmos de control que ayuden en tareas de supervisión, separación, generación de trayectorias, identificación de objetos, formación, etc. De tal forma que, estas arquitecturas permitan una interacción eficaz entre un humano y un grupo de robots. La distribución de tareas de cada robot en un SMR dependerá de la arquitectura de control que se utilice, entre las que se destacan:

Arquitectura de Control Centralizado: en este tipo de arquitectura las órdenes se dan únicamente a un robot y este las distribuye al resto del sistema, como se observa en la Figura 1. 4(a).

Arquitectura de Control Jerárquico: en esta arquitectura se pueden dar órdenes a varios robots considerados líderes, y estos a su vez organizan el funcionamiento de sus robots subordinados, como se aprecia en la Figura 1.4(b).

Arquitectura de control a nivel de conjunto: se caracteriza por ser un sistema que internamente resuelve como llevar a cabo la orden dada por el humano, véase la Figura 1.4(c) [10].

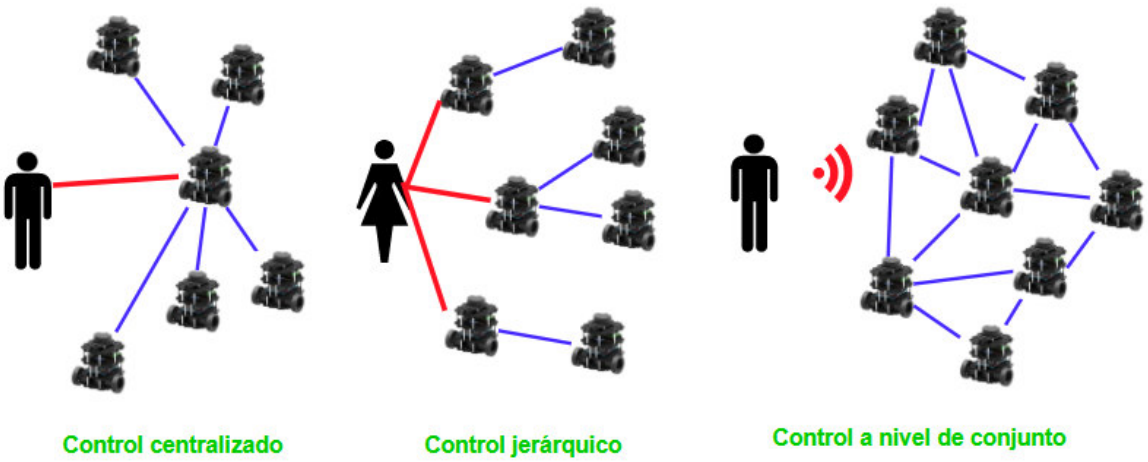


Figura 1.4. Arquitecturas de control de Sistemas Multi - Robot [Fuente propia].

Este trabajo se enfoca en el problema de control de sistemas multi-robot mediante algoritmos de formación, por los beneficios que representa tanto en aplicaciones industriales, militares o de la vida cotidiana.

1.4.2 FORMACIÓN DE ROBOTS

Está en la naturaleza realizar grupos que de alguna u otra forma realicen movimientos coordinados o formaciones como los grupos de animales sincronizados, como por ejemplo banco de peces, bandadas de pájaros, etc. Si al observarlos desde lejos pareciera como si todos sus movimientos están centralizados, esto no significa que siempre sea así. En realidad, el desplazamiento grupal que realizan resulta de todas las acciones individuales de cada ente o agente que conforme el grupo.

De manera similar sucede con los robots y, como se vio en la sección anterior, al grupo de estos se los llama sistemas multi-agentes. Realizar una formación de robots tiene varias aplicaciones y ofrece dar solución generalmente a problemas donde la salud y el bienestar de una persona se vea involucrada como la inspección de campos minados, la transportación de objetos, la vigilancia de un entorno, entre otras aplicaciones. Implementar, desarrollar o diseñar formaciones de robots se justifica básicamente por dos razones:

- un solo robot no puede realizar una tarea compleja. Por lo que al utilizar más robots se puede hacer de forma más sencilla, y

- resulta más económico un sistema que tenga varios robots simples que desarrollen tareas complejas, a tener un solo robot lo suficientemente capaz de realizar una tarea compleja por sí solo [15].

Por tanto, una formación de robots necesita de un control. Se entiende por control de la formación a estabilizar y mantener una forma o esquema de formación deseada [16]. No obstante, se debe agregar que, al realizar un control de formación el principal problema que existe es la medición de la posición entre los agentes. Para esto, existen tres maneras de hacerlo: basada en posición, desplazamiento y distancia. De las cuales, las dos últimas presentan un mayor desafío dado que usan únicamente información local para el control de formación.

Al realizar el control basado en desplazamiento, los agentes miden la posición relativa de sus vecinos con respecto a una referencia global. Así, el desplazamiento de cada agente puede ser directamente controlado para alcanzar la formación deseada. Por otro lado, en el caso del control basado en distancia, los agentes miden la posición relativa de sus vecinos con respecto a su marco de referencia local, que no está alineado con cada uno de los robots. Es por esta razón que el movimiento de cada agente no puede ser directamente controlado; en su lugar, los agentes utilizan las distancias entre ellos como variable de control para así alcanzar la formación deseada [17].

Todas estas observaciones se relacionan con el estudio de las leyes de control cooperativo para realizar una formación de robots. Se presenta a continuación las tres técnicas más estudiadas:

1.4.2.1 Estructura virtual

Este tipo de formación se basa en tratar a todos los robots individuales como partículas de una única estructura firme. La estructura se llega a considerar como virtual cuando el sistema de control es realizado por una persona, es decir que las relaciones geométricas de todo el SMA no se imponen por un sistema físico de restricciones, sino cuando son generadas por alguien externo. Por ejemplo, en la Figura 1.5 se encuentran tres robots en un sistema de referencia global E con una estructura virtual con forma de triángulo, donde la posición (x,y) de cada robot se lo toma como vértice de forma que se logre mantener la relación geométrica entre ellos [18].

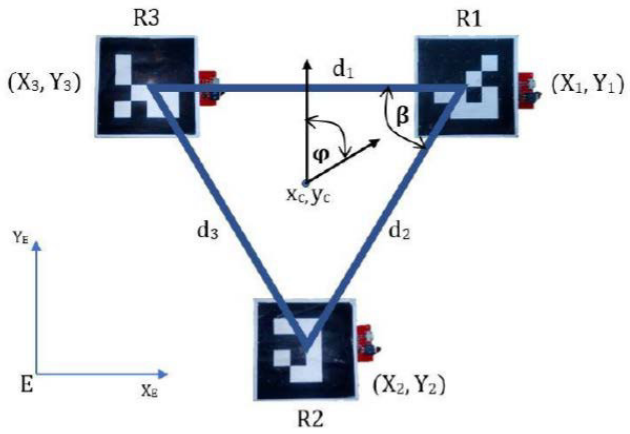


Figura 1.5. Formación de robots mediante estructura virtual en forma de triángulo [18].

1.4.2.2 Comportamiento grupal

A cada uno de los agentes o robots móviles que conforman el SMA se les asigna un comportamiento deseado; de tal forma que al realizar la ponderación relativa del comportamiento de cada uno de los robots se pueda obtener la acción de control de estos. [18], [19].

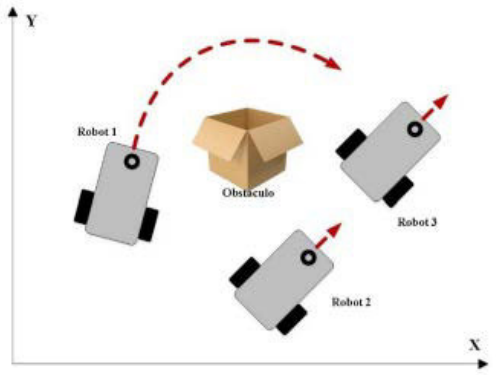


Figura 1.6. Formación de robots mediante comportamiento grupal [18].

En la Figura 1.6, se puede observar cómo los tres robots según el comportamiento asignado intentan evadir el obstáculo. La trayectoria y dirección que toma cada uno de los robots se puede observar mediante flechas rojas.

1.4.2.3 Líder seguidor

La técnica de formación líder seguidor se ha convertido en la ley de control más utilizada en virtud de los roles que adquieren los robots móviles. Uno de los robots se lo designa como líder, mientras que el resto son los robots seguidores. El robot líder tiene como

función principal guiar a todo el grupo al mismo tiempo que cada uno de estos deberán mantener su posición relativa en relación con el líder. Además, esta técnica es posible realizarla de forma centralizada y descentralizada. El presente trabajo se enfoca en realizar la formación líder seguidor de forma descentralizada, por los beneficios que presta en cuanto a robustez y escalabilidad [20].

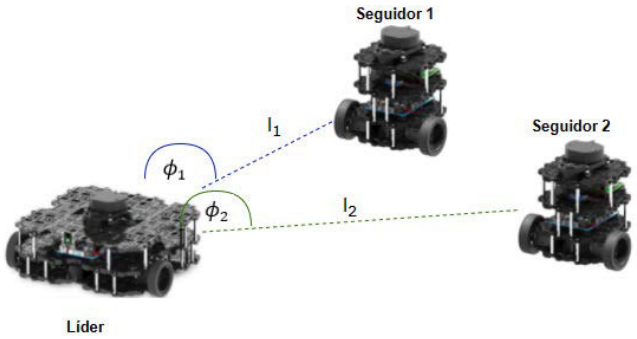


Figura 1.7. Formación de robots esquema líder seguidor [Fuente propia]

En la Figura 1.7, se puede observar cómo se utiliza el esquema líder seguidor en un SMA homogéneo (robots móviles del mismo tipo) donde los robots seguidores se mantienen a una distancia $l_{1,2}$ y ángulo $\phi_{1,2}$ respecto del robot líder.

Dado que el esquema líder seguidor es el más utilizado, se han venido desarrollando varios algoritmos que abordan el problema mediante la modelación y el control basado en el ángulo y la distancia del o los robots seguidores. Como lo es el esquema clásico basado en distancia y ángulo, mostrado en la Figura 1.7. También, se encuentra en [21], la formación de robots líder seguidor junto con evasión de obstáculos, donde para evadir los obstáculos que se encuentren con el ambiente se cambia la formación de los robots, una vez que se logra evadirlo retoman la formación principal. Por otro lado, en [22] utilizan la formación líder seguidor con varios robots líderes los cuales conocen la trayectoria que van a realizar; mientras tanto, los robots seguidores emplean algoritmos de consenso ya que no poseen información acerca de quiénes son los robots líderes y qué trayectoria seguirán. Otra propuesta interesante se la presenta en [23], donde utilizan la dinámica de robots no-holonómicos para crear un algoritmo en el que, a partir de una función que depende de la distancia y el ángulo, se obtiene el gradiente con el fin de realizar la formación de robots. Asimismo, en [24] se demuestra que la geometría de la formación líder seguidor presenta un límite en la curvatura máxima de la trayectoria del líder, de modo que los robots seguidores basan su movimiento respecto a conos centrados en el marco de referencia del robot líder.

El esquema líder seguidor también puede emplear técnicas de generación de trayectorias virtuales, creación de robots virtuales para generación de formaciones poligonales y se encuentra el caso de utilizar referencias virtuales. Este último y el esquema clásico son los algoritmos en los que este trabajo se basa, donde la posición y ángulo deseados se fijan dependiendo de la forma que tomen el grupo de robots [2].

1.4.3 FORMACIÓN LÍDER SEGUIDOR CON ESQUEMA CLÁSICO

La formación líder seguidor con esquema clásico ha venido ganando atención debido a su simplicidad ya que no requiere de conocimiento global. Su funcionamiento se enfoca en hacer que cada robot seguidor mantenga las posiciones relativas deseadas con respecto a su líder, como se muestra en la Figura 1.8. El robot líder a su vez cuando se mueve genera la trayectoria para los robots seguidores de modo que se puede diseñar un control de seguimiento para cada seguidor. Por consiguiente, controlar a un grupo de robots mediante formación líder seguidor se lo puede considerar como una extensión del problema tradicional de seguimiento de trayectorias [6].

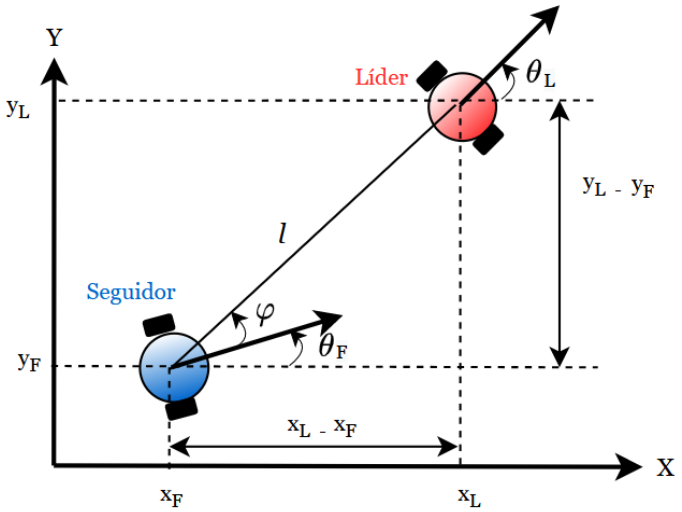


Figura 1.8. Formación de robots líder seguidor esquema clásico [Fuente propia]

Sin embargo, emplear este algoritmo como tal funciona, pero hay que destacar que posee ciertas restricciones como, por ejemplo, los valores permitidos de distancia y ángulo deseado. Es por esta razón, que se hace el uso de referencias virtuales con el fin de crear un algoritmo más robusto respecto a otros algoritmos como el esquema clásico.

1.4.4 FORMACIÓN LÍDER SEGUIDOR CON REFERENCIA VIRTUAL

Cuando se habla acerca de referencia virtual se la puede interpretar también como un robot virtual. Su funcionamiento comienza a partir de la adquisición de datos del robot líder, para

posteriormente calcular los estados de un robot virtual. Siempre que se conozca toda la información del robot virtual, como su posición y velocidad; el siguiente paso es conseguir que el robot seguidor converja a estos estados calculados [6].

Tal como se observa en la Figura 1.9, el robot de referencia virtual tiene como objetivo servir de referencia al seguidor; en otras palabras, el robot seguidor debe converger al robot virtual para que se mantenga a una distancia l_d y un ángulo φ_d deseados respecto del robot líder.

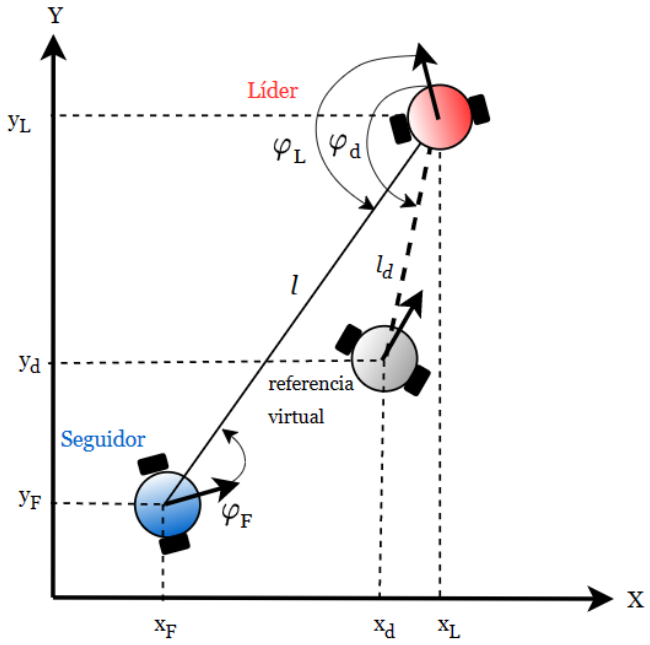


Figura 1.9. Formación de robots líder seguidor con referencia virtual [Fuente propia]

Teniendo en cuenta el proyecto a realizar, se deben buscar herramientas que sean no solamente fáciles de emplear sino también que eviten cualquier restricción y contratiempos en cuanto a licenciamiento. Es por esta razón, que para el desarrollo de un sistema multi-agente con formación líder seguidor, ya sea con esquema clásico o mediante referencia virtual, se utilizan softwares enfocados a robótica y que posean los recursos necesarios para a dar solución a comportamientos complejos y que sean de código abierto, como lo son ROS y Gazebo. El desarrollo sobre estas plataformas en cuanto a su definición, uso e instalación se lo puede encontrar en el primer tomo, del trabajo desarrollado en conjunto [1].

2 METODOLOGÍA

Este capítulo tiene por objetivo presentar los modelos matemáticos para el desarrollo de leyes de control junto con la respectiva calibración e implementación de los algoritmos para

la simulación de la formación de robots con esquema clásico y referencia virtual. Por lo que se tiene un enfoque cualitativo y cuantitativo de modo que se puede experimentar con los agentes del sistema mediante diferentes valores de distancia y ángulo para que logren realizar y mantener una formación en específico.

Para analizar de mejor manera los algoritmos se debe considerar el modelamiento matemático de cada uno de ellos de forma que permita entender la geometría que existe entre los agentes del sistema. Obteniendo así las leyes de control que se utilizarán para realizar la formación líder seguidor.

2.1 MODELAMIENTO

2.1.1 MODELO LÍDER SEGUIDOR CON ESQUEMA CLÁSICO

En la Figura 2.1 se muestra el esquema clásico de la formación líder seguidor, en el cual se busca posicionar al robot seguidor a una distancia l de separación y un ángulo φ_F deseados, ambos con respecto al robot líder.

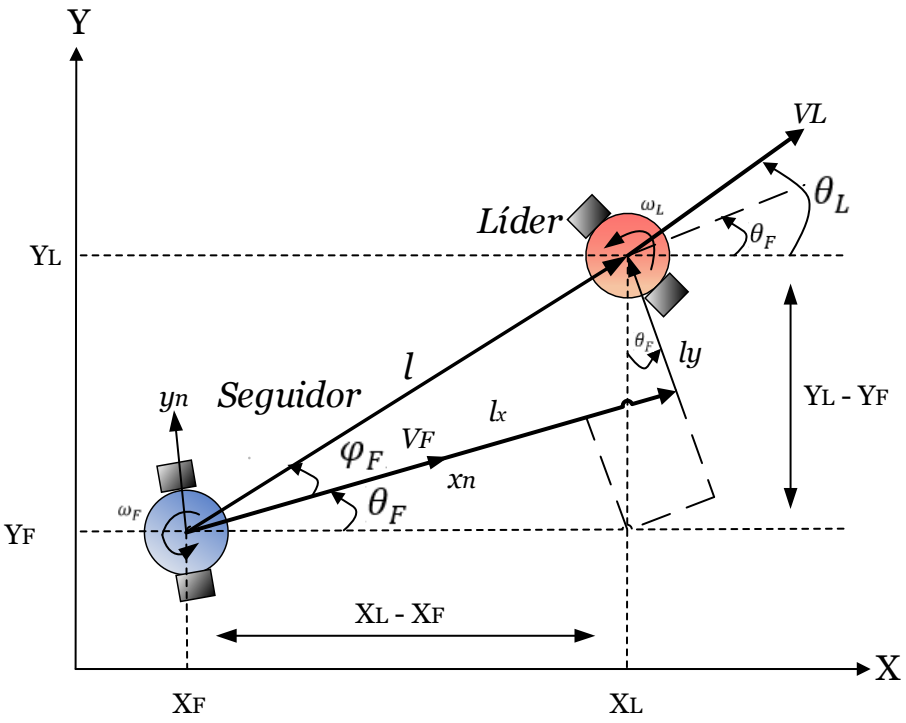


Figura 2.1. Esquema clásico de formación líder seguidor [Fuente propia]

Para iniciar con el modelamiento se considera el modelo cinemático de un robot de tracción diferencial, mostrado en la Ecuación (2.1):

$$\begin{aligned} \dot{x}_i(t) &= v_i(t) * cos\big(\theta_i(t)\big) \\ \dot{y}_i(t) &= v_i(t) * cos\big(\theta_i(t)\big) \\ \dot{\theta}_i(t) &= \omega_i(t) \end{aligned} \tag{2.1}$$

Donde:

$x_i(t)$: posición del robot i sobre el eje X del plano cartesiano

$y_i(t)$: posición del robot i sobre el eje Y del plano cartesiano

$\theta_i(t)$: orientación del robot i con respecto al eje X del plano cartesiano

$v_i(t), \omega_i(t)$: velocidad lineal y angular del robot i, respectivamente

Aquí i hace referencia a si se trata del robot líder, $i = L$, o el robot seguidor, $i = F$. Se entenderá como $i = d$ para el caso del robot de referencia virtual.

Por facilidad en la escritura se omitirá detallar la dependencia con respecto al tiempo en las demás ecuaciones.

De la Figura 2.1, se observa que la posición relativa del robot líder respecto de los ejes (x_n, y_n) del seguidor se escribe como:

$$\begin{aligned} l_x &= (x_L - x_F)\, cos(\theta_F) + (y_L - y_F)\, sin(\theta_F) \\ l_y &= -(x_L - x_F)\, sin(\theta_F) + (y_L - y_F)\, cos(\theta_F) \end{aligned} \tag{2.2}$$

Dado que se desea mantener una formación líder seguidor, es necesario que $l_x \rightarrow l_{d_x}$ y $l_y \rightarrow l_{d_Y}$, donde l_{d_x} y l_{d_y} son las distancias relativas deseadas entre los robots a lo largo de los ejes (x_n, y_n). Además, para el desarrollo del esquema clásico, se busca que $\theta_F \rightarrow \theta_L$. Pudiendo así definir los errores de seguimiento:

$$e_{l_x} = l_x - l_{d_x}, \qquad e_{l_y} = l_y - l_{d_y}, \quad e_\theta = \theta_F - \theta_L \tag{2.3}$$

Donde l_{d_x} y l_{d_y} son definidos como:

$$l_{d_x} = l_d(t)\cos\big(\varphi_d(t)\big), \quad l_{d_y} = l_d(t)\sin(\varphi_d(t)) \tag{2.4}$$

De la Figura 2.1, se puede observar que el robot seguidor se encuentra a una distancia l y ángulo φ_F respecto del robot líder. Sin embargo, se debe notar que se busca tener valores de distancia $l_d(t)$ y ángulo $\varphi_d(t)$ conocidos o deseados. Si ahora se deriva respecto al tiempo las Ecuaciones (2.3) se tiene lo siguiente:

$$\begin{aligned}\dot{e}_{l_x} &= -(x_L - x_F)\dot{\theta}_F\, sin(\theta_F) + (\dot{x}_L - \dot{x}_F)\, cos(\theta_F) + (y_L - y_F)\dot{\theta}_F\, cos(\theta_F) \\ &\quad +(\dot{y}_L - \dot{y}_F)\, sin(\theta_F) - \dot{l}_d\, cos(\varphi_d) + l_d\dot{\varphi}_d\, sin(\varphi_d) \\ \dot{e}_{l_y} &= -(x_L - x_F)\dot{\theta}_F\, cos(\theta_F) - (\dot{x}_L - \dot{x}_F)\, sin(\theta_F) - (y_L - y_F)\dot{\theta}_F\, sin(\theta_F) \\ &\quad +(\dot{y}_L - \dot{y}_F)\, cos(\theta_F) - \dot{l}_d\, sin(\varphi_d) - l_d\dot{\varphi}_d\, cos(\varphi_d) \\ \dot{e}_\theta &= \omega_F - \omega_L\end{aligned} \tag{2.5}$$

Al reemplazar en la Ecuación (2.5) la cinemática de los robots líder $(i = L)$ y seguidor $(i = F)$, dada en la Ecuación (2.1), y realizar la suma y resta de $l_{d_x}\,\omega_F$ y $l_{d_y}\omega_F$ de forma adecuada, se obtiene la siguiente expresión:

$$\begin{aligned}\dot{e}_{l_x} &= e l_y \omega_F - v_F + v_L \cos(e_\theta) + \xi_1 \\ \dot{e}_{l_y} &= -e l_x \omega_F - v_L \sin(e_\theta) + \xi_2 \\ \dot{e}_\theta &= \omega_F - \omega_L\end{aligned} \tag{2.6}$$

Donde:

$$\begin{aligned}\xi_1 &= -\dot{l}_d \cos(\varphi_\mathrm{d}) + l_d\dot{\varphi}_d \sin(\varphi_d) + l_{d_y}\omega_F \\ \xi_2 &= -\dot{l}_d \sin(\varphi_\mathrm{d}) - l_d\dot{\varphi}_d \cos(\varphi_d) - l_{d_x}\omega_F\end{aligned} \tag{2.7}$$

En las Ecuaciones (2.6) y (2.7) se puede observar la existencia de ξ_1 y ξ_2, que corresponden a términos de perturbación dependientes de la distancia y el ángulo deseado del robot seguidor. Se debe recalcar que, para las ecuaciones propuestas, la posición deseada se ha escogido sin considerar ninguna restricción. Se considera como restricción a valores de ángulo y distancia que el robot seguidor no pueda mantener mientras realice la formación, principalmente debido a sus restricciones no holonómicas.

Para realizar la formación de un sistema multi-agente, a partir de la Ecuación (2.7) se presentan la velocidad lineal y angular de los agentes seguidores en la Ecuación (2.8).

$$\begin{aligned}v_F &= k_1 e_{l_x} + v_L \cos(e_\theta) + \xi_1 \\ \omega_F &= \omega_L + k_2 v_L \frac{\sin(e_\theta)}{e_\theta} e_{l_y} - k_3 e_\theta\end{aligned} \tag{2.8}$$

Donde $k_1,\ k_2\ y\ k_3$ son constantes positivas mayores a cero. En [6] se plantea la función candidata de Lyapunov mostrada en la Ecuación (2.9), con lo cual los autores analizan y demuestran la estabilidad del sistema al aplicar las señales de control dadas en la Ecuación (2.8).

$$\begin{aligned}V(e) &= \frac{1}{2}k_2\left(e_{l_x}^2 + e_{l_y}^2\right) + \frac{1}{2}\,e_\theta^2 \\ \dot{V}(e) &= -k_1 k_2 e_{l_x}^2 - k_3 e_\theta^2 + k_2 e_{l_y}\xi_2\end{aligned} \tag{2.9}$$

El análisis de estabilidad de los errores lo efectúan con el fin de determinar las restricciones de este tipo de control, para mayores detalles se recomienda revisar [6]. En base a dicho análisis, se obtienen las siguientes posibles soluciones para garantizar la estabilidad del sistema:

1) $l_d = 0$: este caso no es aplicable para el proyecto dado que la distancia debe ser mayor a cero para evitar que los robots líder y seguidor colisionen.

2) $l_d = cte, \varphi_d = cte, \omega_F = 0$: esta opción se puede adaptar ya que se puede evitar girar con el robot líder de modo que avance únicamente con velocidad lineal, además de que se pretende realizar formaciones que cuentan con distancia y ángulo fijos, es decir, constantes.

3) $\omega_F = 0$, $l_d = \frac{-\dot{l}_d}{\dot{\varphi}_d}\tan(\varphi_d)$: esta opción no se la tomará en cuenta ya que, al no tener distancias y ángulos deseados variantes en el tiempo, su derivada se vuelve nula provocando una singularidad matemática.

4) $l_d = cte, \varphi_d = \frac{(2n+1)\pi}{2}, n \in \mathbb{Z}$: puede ser parte de la solución dado que, con una distancia constante, los valores permitidos de ángulo serían: $\frac{\pi}{2}, \frac{3\pi}{2}, \frac{5\pi}{2}, etc.$

Como se mencionó anteriormente, el algoritmo de formación líder seguidor empleando esquema clásico presenta algunas soluciones no aplicables, que corresponden al primer y tercer casos presentados, por las razones explicadas en cada uno de ellos. Por lo que, para cumplir con las formaciones propuestas en este proyecto se utilizarán la segunda y cuarta opción donde los valores de distancia y ángulo deseado son constantes. Debido a que se tiene soluciones parciales para realizar la formación, en la Sección 2.1.2 se propone el uso de un robot virtual de referencia cuya posición es obtenida a través de la información del robot líder proporcionada en tiempo real.

2.1.2 MODELO LÍDER SEGUIDOR CON REFERENCIA VIRTUAL

El objetivo de tener una formación líder seguidor es que el robot seguidor se mantenga a una distancia y un ángulo relativo respecto del robot líder, al aumentar una referencia virtual se pretende mejorar el algoritmo con esquema clásico, como por ejemplo, respecto a la rapidez en la que los robots se llegan a formar. Un robot virtual actúa como referencia virtual para el robot seguidor, de modo que sus estados de referencia se calculan mediante la información de posición y velocidad del robot líder. De esta forma, se logra que el robot seguidor converja hacia el robot de referencia virtual [6].

Considerando la formación de la Figura 2.2 se tiene tres robots no holonómicos de tracción diferencial. Sabiendo que el robot seguidor en color azul debe converger hacia el robot de referencia virtual en color gris, se tiene que la formación debe mantener una distancia l_d y ángulo φ_d deseados respecto del robot líder en color rojo. Estos parámetros son determinados por la posición del robot de referencia.

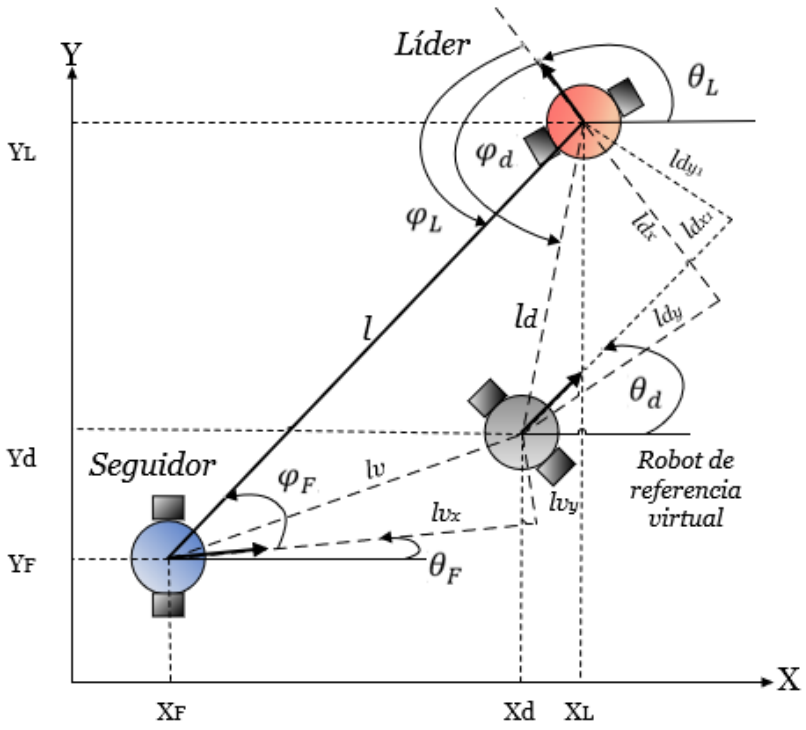


Figura 2.2. Esquema de formación líder seguidor con referencia virtual [Fuente propia]

La posición del robot de referencia respecto del robot líder se escribe como:

$$\begin{aligned} l_{dx} &= -(x_L - x_d)\cos(\theta_L) - (y_L - y_d)sen(\theta_L) \\ l_{dy} &= \ (x_L - x_d)\,sen(\theta_L) - (y_L - y_d)cos(\theta_L) \end{aligned} \tag{2.10}$$

Se tiene a l_{dx} y l_{dy} como distancias conocidas que se calculan de la misma forma a la del esquema clásico, representadas en la Ecuación (2.4)

Puesto que se desea saber la posición deseada x_d, y_d, de la Ecuación (2.10) se despejan x_d, y_d, obteniendo lo siguiente:

$$\begin{aligned} x_d &= x_L + l_{dx}\cos(\theta_L) - l_{dy}sen(\theta_L) \\ y_d &= y_L + l_{dx}\,sin(\theta_L) - l_{dy}cos(\theta_L) \end{aligned} \tag{2.11}$$

Tomando en cuenta las ecuaciones descritas en las Ecuaciones (2.4) y (2.11), se tiene finalmente la posición deseada:

$$x_d = x_L + l_d\, cos(\theta_L + \varphi_d)$$
$$y_d = y_L + l_d\, sen(\theta_L + \varphi_d) \tag{2.12}$$

La orientación del robot virtual de referencia se calcula a partir de la Ecuación (2.1), dando como resultado

$$\theta_d = \tan^{-1}\left(\frac{\dot{x}_d}{\dot{y}_d}\right) \tag{2.13}$$

Las derivadas de las posiciones del robot virtual se calculan a partir de la Ecuación (2.12)

$$\dot{x}_d = \dot{x}_L + \dot{l}_d\, cos(\theta_L + \varphi_d) - l_d(\omega_L + \dot{\varphi}_d)sen(\theta_L + \varphi_d)$$
$$\dot{y}_d = \dot{y}_L + \dot{l}_d\, sen(\theta_L + \varphi_d) + l_d(\omega_L + \dot{\varphi}_d)cos(\theta_L + \varphi_d) \tag{2.14}$$

Ya que los estados del robot virtual se encuentran definidos. A continuación, se escriben las distancias que van desde el robot seguidor hacia el robot virtual de referencia:

$$l_{vx} = (x_d - x_F)\, cos(\theta_F) + (y_d - y_F)sen(\theta_F)$$
$$l_{vy} = -(x_d - x_F)\, sen(\theta_F) + (y_d - y_F)cos(\theta_F) \tag{2.15}$$

Anteriormente se mencionó que el objetivo de la formación es que el robot seguidor converja hacia el robot de referencia virtual por lo que se pueden definir los siguientes errores:

$$e_{l_x} = l_{vx}\,, \qquad e_{l_y} = l_{vy}, \qquad e_\theta = \theta_F - \theta_d \tag{2.16}$$

Al derivar las ecuaciones descritas en la Ecuación (2.16) se obtiene:

$$\begin{aligned} \dot{e}_{l_x} = {} & (x_d - x_F)\dot{\theta}_F\, sin(\theta_F) + (\dot{x}_d - \dot{x}_F)cos(\theta_F) + (y_d - y_F)\dot{\theta}_F cos(\theta_F) \\ & + (\dot{y}_d - \dot{y}_F)sin(\theta_F) \\ \dot{e}_{l_y} = {} & -(x_d - x_F)\dot{\theta}_F\, cos(\theta_F) - (\dot{x}_d - \dot{x}_F)sin(\theta_F) - (y_d - y_F)\dot{\theta}_F sin(\theta_F) \\ & + (\dot{y}_d - \dot{y}_F)cos(\theta_F) \\ \dot{e}_\theta = {} & \omega_F - \omega_d \end{aligned} \tag{2.17}$$

Si ahora se modifica la Ecuación (2.1) tomando en cuenta los robots virtual y seguidor, a partir de varias operaciones algebraicas, se presenta en la Ecuación (2.18) la representación de los errores de seguimiento mediante sus velocidades lineales, angulares y errores:

$$\dot{e}_{l_x} = e_{l_y}\omega_F - v_F + v_d\, cos(e_\theta)$$
$$\dot{e}_{l_y} = -e_{l_x}\omega_F - v_F - v_d\, sin(e_\theta) \tag{2.18}$$
$$\dot{e}_\theta = \omega_F - \omega_d$$

Se debe considerar ahora que se puede dar solución al problema de formación líder seguidor a través de las señales de control propuestas en [6]. Para esto se debe considerar que, ante cualquier distancia y ángulo deseados no nulos, se tendrá lo siguiente:

$$v_F = k_1 e_{l_x} + v_d \cos(e_\theta)$$
$$\omega_F = \omega_d + k_2 v_d \frac{\sin(e_\theta)}{e\theta} e_{l_y} - k_3 e_\theta \quad (2.19)$$

Siendo k_1, k_2 y k_3 constantes positivas mayores a cero.

Si se parte del análisis en la Ecuación (2.1) se tiene las siguientes velocidades lineal y angular para el robot de referencia virtual.

$$v_d = \dot{x}_d \cos(\theta_d) + \dot{y}_d \sin(\theta_d)$$
$$\omega_d = \frac{\ddot{y}_d \dot{x}_d - \ddot{x}_d \dot{y}_d}{\dot{x}_d{}^2 + \dot{y}_d{}^2} \quad (2.20)$$

Los autores en [6] demuestran que, empleando las señales de control de la Ecuación (2.20), el robot seguidor converge hacia el robot de referencia virtual, ya que los errores mostrados en la Ecuación (2.18) tienden a cero. La demostración completa de la estabilidad de este controlador se la puede encontrar en la referencia [6].

Una vez revisados los modelos matemáticos de los dos algoritmos se procede a implementarlos en los softwares de ROS y Gazebo. Todo lo relacionado a la creación de paquetes, scripts, mundos, archivos .launch, entre otros, se lo puede revisar en el primer tomo del trabajo desarrollado en conjunto [1].

2.2 IMPLEMENTACIÓN

Partiendo de lo anterior, para la realización de este trabajo se procede a explicar brevemente la distribución de las carpetas y documentos que almacenan el desarrollo de la escena y simulación implementadas. Esta distribución se presenta de una forma más gráfica en la Figura 2.3.

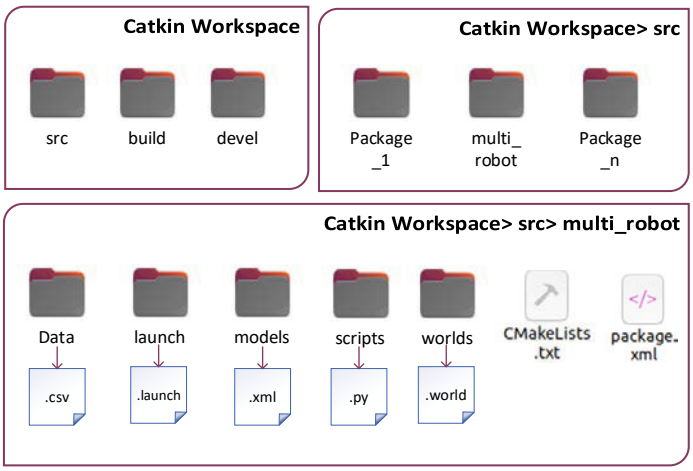


Figura 2.3. Distribución de carpetas del proyecto en ROS [Fuente propia]

La carpeta que almacena toda la información o paquete posee el nombre de ***multi_robot,*** en ella se encuentra lo siguiente:

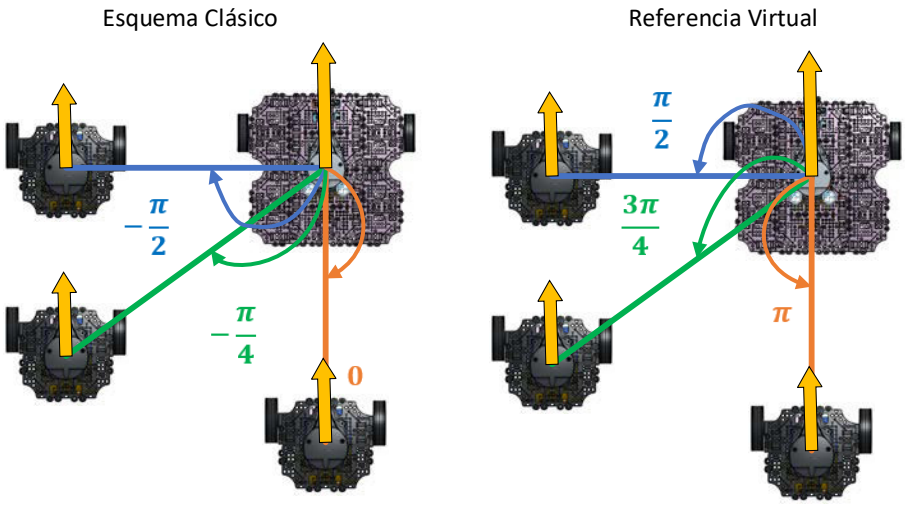
- ***Data*****:** almacena los datos de referencia como distancia y ángulo deseados junto con los valores reales de estos y la posición (x,y) de cada robot. Estos datos son de utilidad para graficar las variables a controlar y calibrar las constantes de las leyes de control vistas en la sección anterior.
- ***launch*****:** en esta carpeta se encuentran archivos .launch que ayudan a generar los modelos virtuales de los agentes del sistema, también se lo suele mencionar en inglés como: realizar un "spawn". Estos archivos también son utilizados para desplegar el entorno virtual en gazebo.
- ***models, worlds*****:** poseen archivos que sirven para generar y configurar el mundo o entorno virtual de gazebo.
- ***scripts*****:** se encuentran documentos desarrollados en Python que albergan toda la programación de los algoritmos presentados anteriormente incluyendo el movimiento del robot líder.

En el primer tomo del trabajo en conjunto [1] se encuentra la explicación más a detalle sobre el algoritmo utilizado para realizar el movimiento del robot líder y la descripción de los archivos ***CMakeLists.txt*** y ***package.xml***.

Vinculado el concepto del desarrollo de algoritmos, en la siguiente sección se presentan los diagramas de flujo de los robots seguidores para que generen diferentes formas empleando la formación líder seguidor mediante tanto el esquema clásico como el de referencia virtual.

2.2.1 IMPLEMENTACIÓN DE CONTROL DE FORMACIÓN

Antes de presentar el desarrollo de los diagramas de flujo para los diferentes algoritmos, es necesario explicar lo referente a la referencia del ángulo deseado para que los robots se formen acorde entre las tres opciones que el usuario puede escoger: fila, columna, diamante o V.



(a) (b)

Figura 2.4. Ángulo deseado en (a) esquema clásico y (b) referencia virtual [Fuente propia]

Si bien en la Sección 2.1 se presentó la geometría de la formación líder seguidor, ya sea con esquema clásico o referencia virtual, mediante la Figura 2.4, es posible entender de mejor manera qué ángulo se debe fijar para que los robots puedan variar la formación deseada. la Figura 2.4, para los dos algoritmos, el color azul representa la formación fila, el color naranja la formación columna y para la formación V o diamante se tiene el color verde.

Seguidamente se tiene los diagramas de flujo del robot seguidor con los dos algoritmos. El diagrama de flujo del movimiento del robot líder se lo puede encontrar en el tomo 1 del trabajo desarrollado en conjunto [1] . Puesto que se empleó la arquitectura de control de forma descentralizada, el algoritmo para cada robot seguidor se lo representa de forma resumida en el diagrama de flujo realizado en la Figura 2.5.

El principal y más importante paso de los algoritmos, además de adquirir librerías, declarar e inicializar variables, es iniciar el nodo con el que se va a comunicar. Esto es fundamental, ya que este nodo puede ser publicador o subscriptor de los tópicos de velocidad y odometría de los agentes del sistema. Posteriormente se tiene el control, este empieza con la adquisición de datos de la posición y orientación de los robots para calcular las señales de control. Tanto para el algoritmo con esquema clásico y de referencia virtual se lo representa con un solo diagrama de flujo de control, pero se debe destacar que, si bien el diagrama es similar, los algoritmos se diferencian en las ecuaciones de control (letras en color rojo de la Figura 2.5) ya que son diferentes para cada algoritmo por la matemática que llevan internamente. En Anexo I se puede encontrar el detalle del repositorio generado.

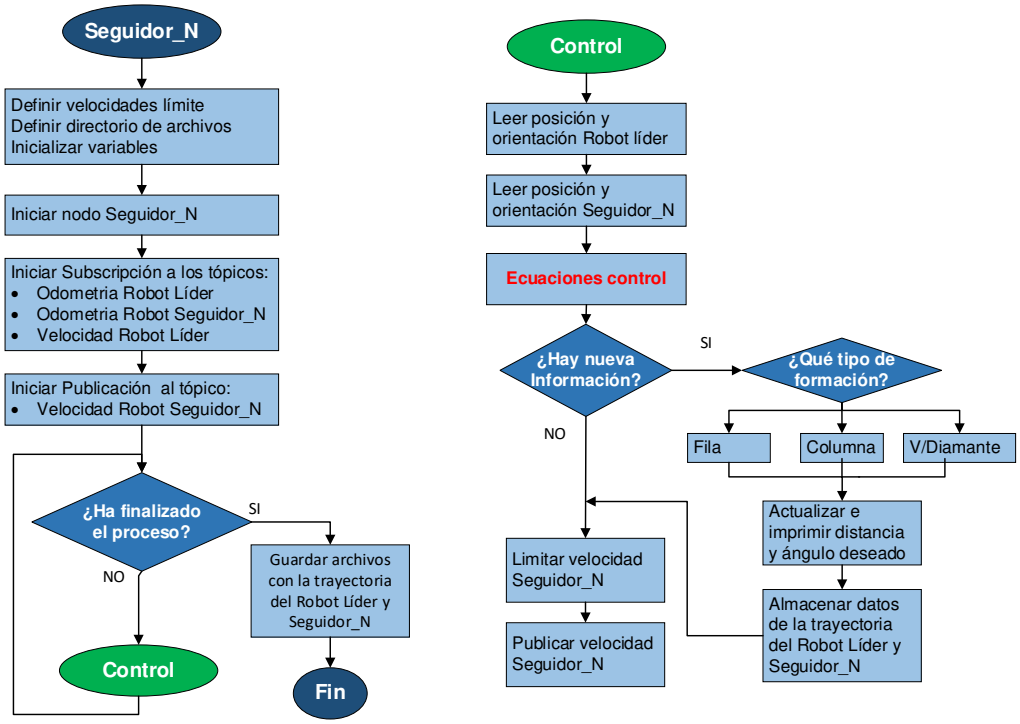


Figura 2.5. Diagramas de flujo de algoritmos implementados [Fuente propia]

Como complemento, para que la ejecución y simulación del programa sea más amigable para el usuario, se han desarrollado archivos ejecutables que desplieguen menús en la terminal, donde se pueda elegir el número de agentes y el tipo de algoritmo de formación. Los diagramas de flujos de estos se pueden apreciar en las Figuras 2.6 y 2.7.

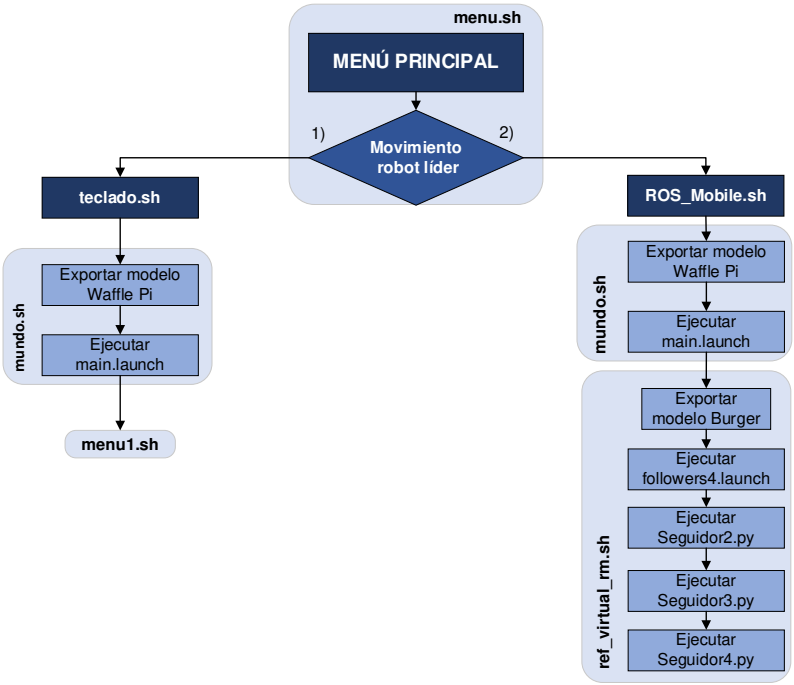


Figura 2.6. Despliegue de menú principal [Fuente propia]

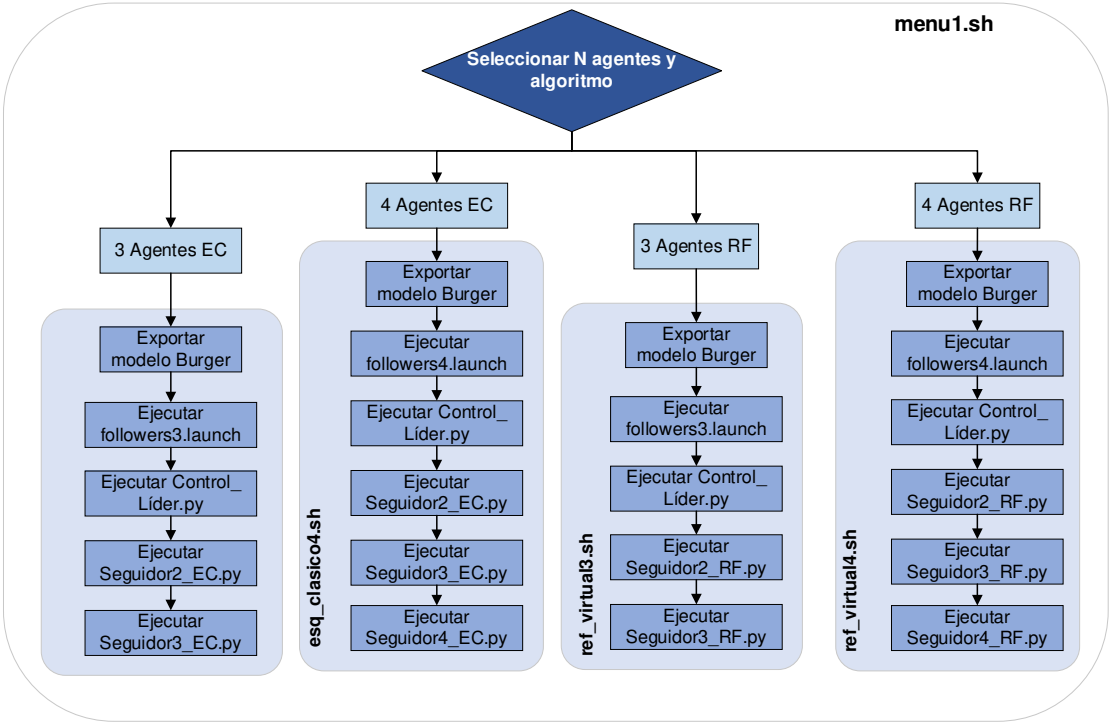


Figura 2.7. Despliegue de menú secundario [Fuente propia]

Se sugiere revisar el primer tomo [1], para entender de mejor manera los archivos internos que se presentan en los despliegues de menú. A continuación, en la Figura 2.8 se aprecia la visualización de los menús desde la terminal.

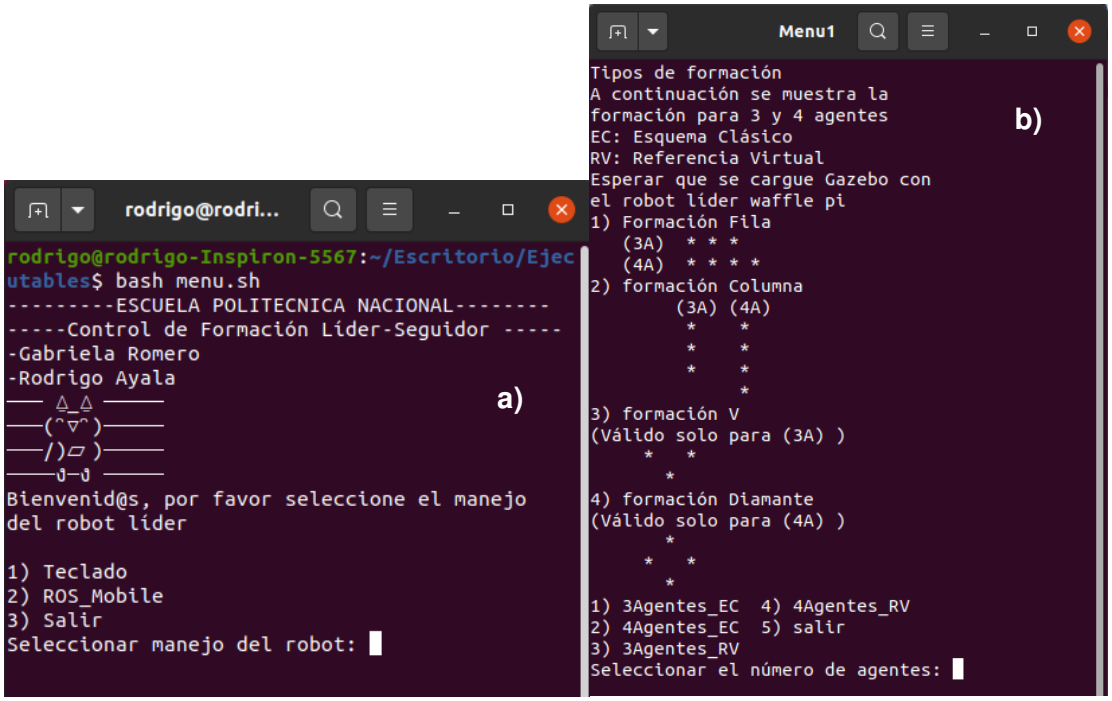


Figura 2.8. Despliegue de menús desde la terminal, (a) menú principal, (b) menú secundario [Fuente propia]

Ahora se examinará brevemente las terminales generadas al ejecutar el proyecto. Para tener más detalle e información se recomienda revisar el Anexo II, el cual corresponde al manual de usuario. La descripción de cada terminal se describe a continuación:

- **Terminal 1:** se ejecuta inicialmente el menú principal donde se podrá escoger el movimiento del robot líder, véase en la Figura 2.8 (a).
- **Terminal 2:** se despliega Roscore junto con Gazebo y el modelo del robot líder como se muestra en la Figura 2.9 (a).
- **Terminal 3:** despliegue de menú secundario (véase en la Figura 2.8 (b)), en este se permite seleccionar el tipo de formación, número de agentes y tipo de algoritmo. También se lanza el modelo de los robots seguidores para que aparezcan en Gazebo como se lo aprecia en la Figura 2.9 (b).
- **Terminal 4:** en caso de seleccionar el movimiento del robot líder por teclado. Permite mover al robot mediante teclas y hacer cambios en el tipo de formación: fila, columna, V para tres agentes y diamante en el caso de 4 agentes, tal como se observa en la Figura 2.9 (c). En caso de seleccionar el movimiento por ROS_Mobile, se omite esta terminal.
- **Terminal 5,6 y/o 7:** en el caso de escoger esquema clásico, se muestra información de distancia en [m] y ángulo deseado en [rad] y los valores reales del seguidor junto con el tipo de formación que se está realizando en ese momento. Para el algoritmo de referencia virtual se aumenta la información de distancia y ángulo de la referencia como se observa en la Figura 2.9 (d).

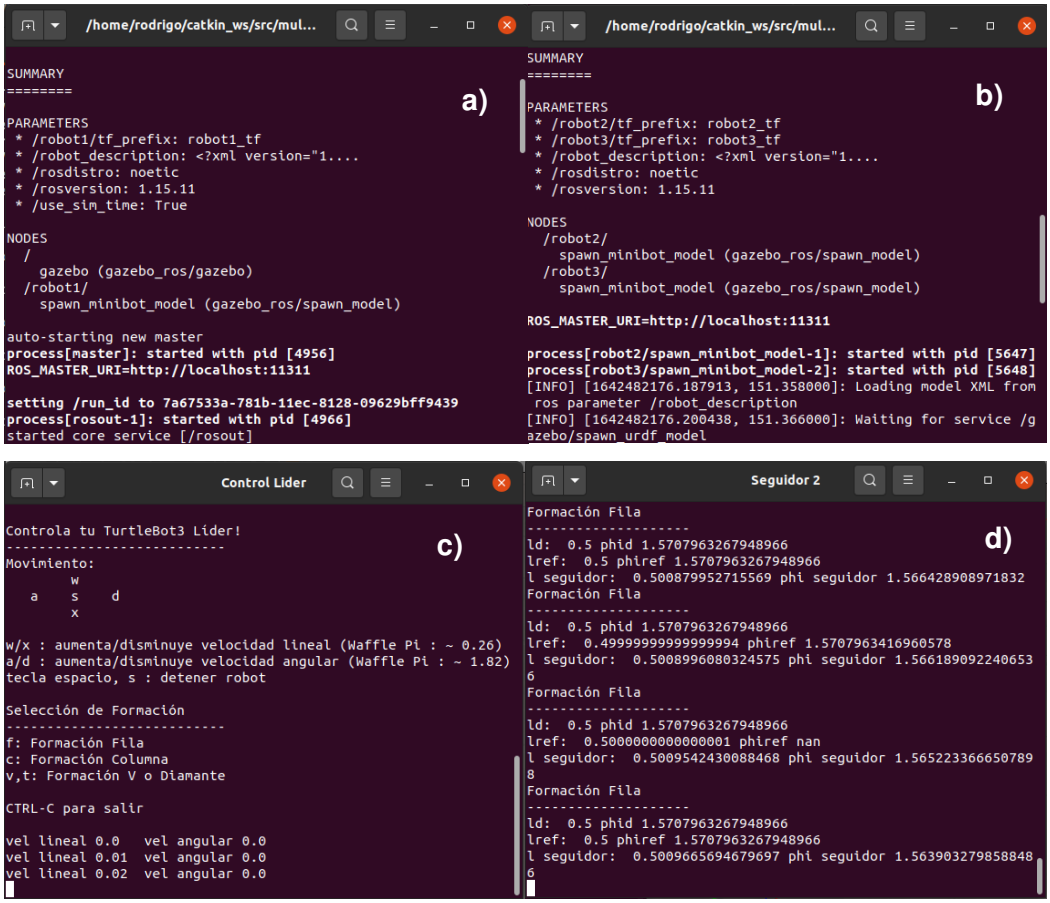


Figura 2.9. Despliegue de terminales [Fuente propia]

Respecto a la posición inicial de los robots seguidores y robot líder, se seleccionó posiciones determinadas, de manera que se pueda observar cómo llegan a la formación que haya indicado el usuario. Las posiciones se describen a continuación en unidades de [m] y de forma gráfica se lo puede apreciar en la Figura 2.10.

- Líder R1: $x = 0\ [m], y = 0[m]$
- Seguidor R2: $x = -0.5\ [m],\ y = 1\ [m]$
- Seguidor R3: $x = -2\ [m],\ y = 1.5[m]$
- Seguidor R4: $x = -1.5[m],\ y = 2[m]$

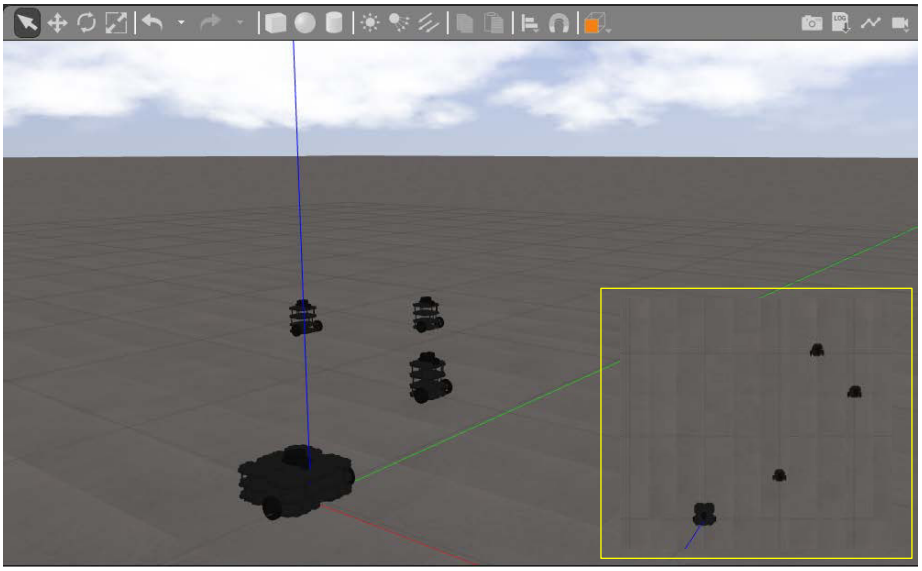


Figura 2.10. Posición inicial de los agentes del sistema [Fuente propia]

Una vez se disponga de todos los elementos necesarios dentro de la simulación se debe calibrar las leyes de control para los algoritmos de esquema clásico y referencia virtual, de modo que se realice de manera adecuada la formación líder seguidor para ambos casos.

2.3 Calibración

En la Sección 2.1 se presentaron las Ecuaciones (2.8) y (2.19) que corresponden a las leyes de control para el algoritmo de formación líder seguidor con esquema clásico y referencia virtual, respectivamente. En ambos casos, se disponen de tres constantes que permiten el ajuste de los controladores. Para su calibración, se procede a realizar pruebas a diferente distancia y ángulo con dos agentes, un robot líder y un robot seguidor. De estas pruebas se mide el índice de error al cuadrado (ISE) por sus siglas en inglés, el cual es presentado en la Ecuación (2.23)

$$ISE = \int_0^{\infty} e(t)^2 dt \tag{2.21}$$

Posteriormente, se tabulan los resultados obtenidos para diferentes combinaciones de constantes con el fin de determinar las que presentan el mejor desempeño del controlador. Para poder comparar los resultados, las pruebas se deben realizar bajo las mismas condiciones iniciales y de operación. Así, en todos los casos presentados, el robot seguidor inicia en la posición $x = -0.5\,[m]\,, y = 1\,[m]$ y el robot líder inicia en la posición $x = 0\,[m]\,, y = 0\,[m]$. Durante la ejecución, los robots avanzan 6 [m] con velocidad lineal 0.109 [m/s] y velocidad angular nula.

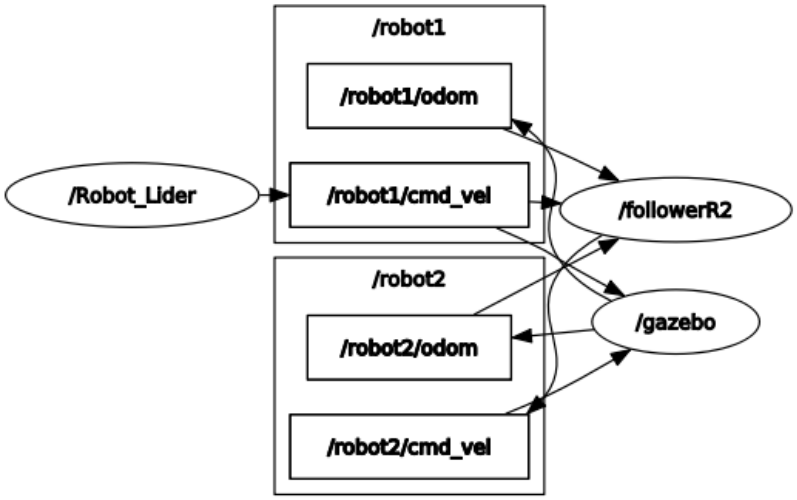


Figura 2.11. Diagrama de nodos y tópicos para un robot líder y un robot seguidor [Fuente propia]

En la Figura 2.11, se observa el diagrama de nodos y tópicos al ejecutar únicamente el robot líder y un robot seguidor para la realización de las pruebas de calibración de los controladores de formación con esquema clásico o con referencia virtual. El diagrama de nodos y tópicos resulta el mismo para los dos algoritmos, su diferencia se centra en que el nodo /followerR2 puede contener o bien el algoritmo con referencia virtual o con esquema clásico.

El nodo /Robot_Líder publica sobre el tópico de velocidad del robot1 o líder, luego el nodo /followerR2 se suscribe al mismo tópico y al de odometría del robot2 o seguidor, para luego publicar la velocidad lineal y angular del seguidor sobre su tópico /robot2/cmd_vel. Finalmente, el nodo de gazebo se comunica con los tópicos de odometría y de velocidad del robot líder y seguidor.

Considerando la comunicación de nodos y tópicos de ROS descrita anteriormente, se procede a realizar la calibración de constantes de las leyes de control para los algoritmos de esquema clásico y referencia virtual.

2.3.1 CALIBRACIÓN DEL CONTROLADOR PARA FORMACIÓN CON ESQUEMA CLÁSICO

Al ser uno de los objetivos del presente trabajo mantener una formación, las pruebas realizadas con un solo agente seguidor poseen valores de distancia y ángulo necesarios para dichas formaciones. Así, para los valores de $l_d = 0{,}5\,[m]\; y\; \varphi_d = -\frac{\pi}{2}[rad]$ que equivalen a una formación fila, se realizan 5 pruebas con diferentes constates, mostradas en la Tabla 2.1.

Tabla 2.1. Calibración de constantes para formación fila con esquema clásico

$l_d = 0,5\ y\ \varphi_d = -\frac{\pi}{2}$	Constantes			ISE	
No. Prueba	k_1	k_2	k_3	**Distancia**	**Ángulo**
P1	**2**	**5**	**1**	**857,748**	**100,971**
P2	7	15	1.5	1152,28	724,22
P3	3	8	1	801,28	459,27
P4	1.7	5.5	1.3	875,55	147,13
P5	1.9	5.1	1.2	1978,93	288.81

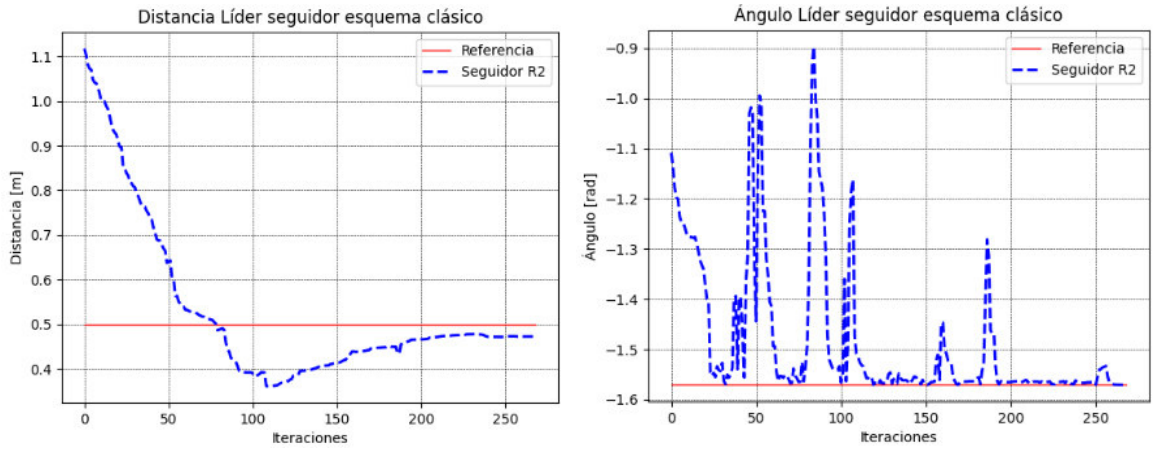


Figura 2.12. Resultados de distancia y ángulo para prueba con formación fila empleando el esquema clásico [Fuente propia]

En la Figura 2.12, se puede apreciar las variables de distancia y ángulo en color azul con su respectiva referencia en color rojo. Las gráficas escogidas corresponden a la prueba que presenta el mejor desempeño. Es decir que, los valores de distancia y ángulo presentados son producto de las constantes utilizadas en la prueba 1 de la Tabla 2.1, con los que se obtuvo el menor valor de ISE en ángulo y el segundo menor ISE en distancia. Se puede apreciar también en la gráfica de distancia que no logra llegar a la referencia establecida debido a que el algoritmo con esquema clásico necesita de un mayor tiempo de ejecución para que logre recorrer más distancia ya que inicialmente se consideró el desplazamiento del robot líder de 6[m]. Por otra parte, la variable de distancia y ángulo están ligadas de modo que, si la distancia no llega a la referencia indicada, el robot comienza a girar para acomodarse es por esta razón que la gráfica del ángulo presenta varios picos.

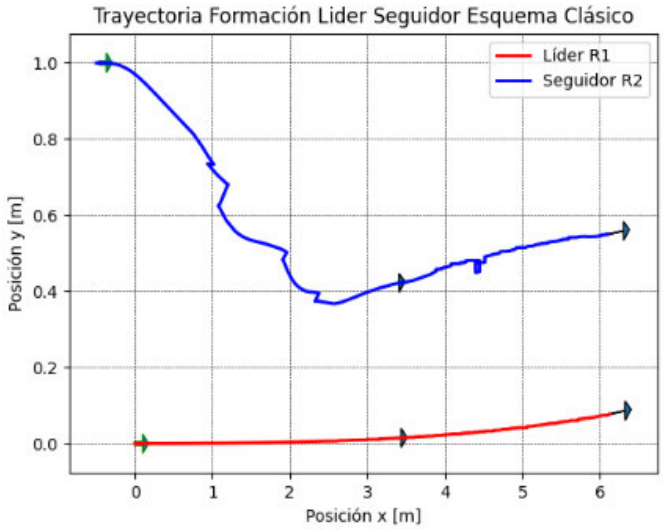


Figura 2.13. Trayectoria de robots líder y seguidor con formación fila empleando el esquema clásico [Fuente propia]

Conforme a las gráficas de distancia y ángulo, en la Figura 2.13, se puede observar la trayectoria del robot líder en color rojo y el seguidor en color azul, aquí se interpreta de mejor manera cómo el robot seguidor trata de seguir al robot líder en formación fila tratando de cumplir con los valores de distancia y ángulo previamente seteados.

Como segundo caso, se realiza una formación columna, por lo que en la Tabla 2.2 se tiene un ángulo deseado $\varphi_d = 0[rad]$ y una distancia deseada $l_d = 1\ [m]$.

Tabla 2.2. Calibración de constantes para formación columna con esquema clásico

$l_d = 1\ y\ \varphi_d = 0$	Constantes			ISE	
No. Prueba	k_1	k_2	k_3	**Distancia**	**Ángulo**
P1	2	5	1	373,09	8374,56
P2	7	15	1.5	237,74	1935,487
P3	**3**	**8**	**1**	**113,02**	**2752,88**
P4	1.7	5.5	1.3	389,16	8415,13
P5	1.9	5.1	1.2	213,88	5224,15

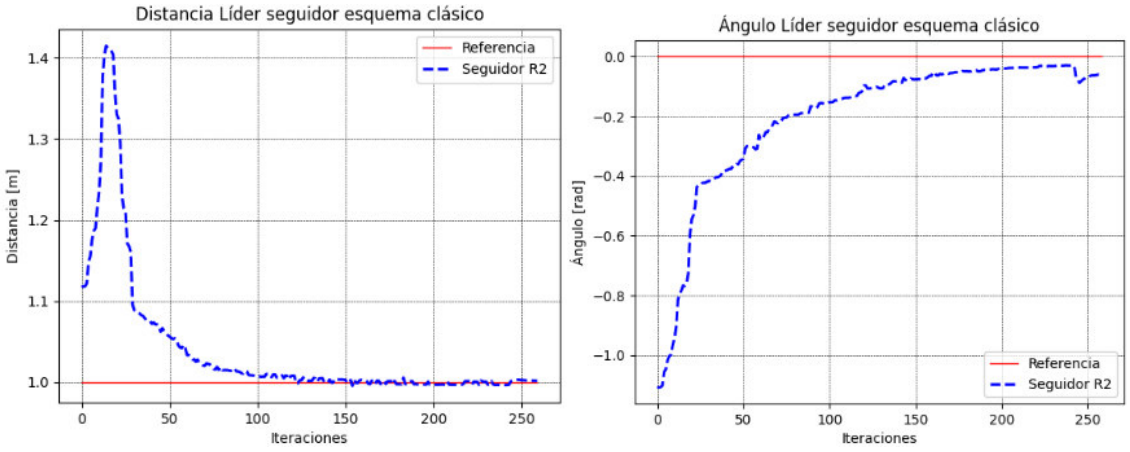


Figura 2.14. Resultados de distancia y ángulo para prueba con formación columna empleando el esquema clásico [Fuente propia]

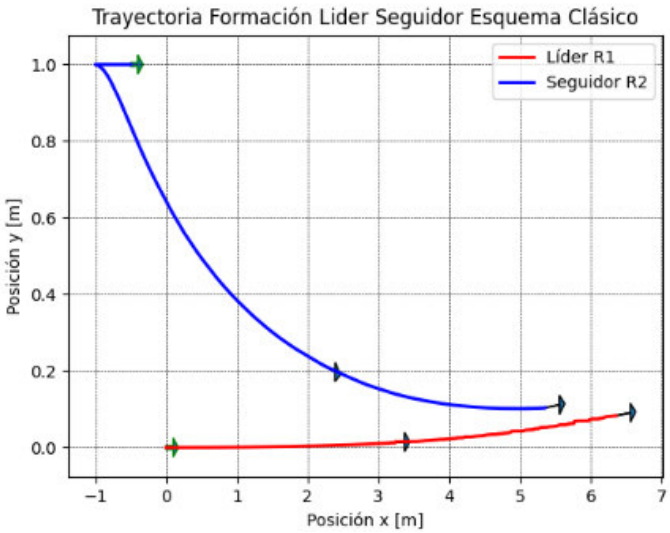


Figura 2.15. Trayectoria de robots líder y seguidor con formación columna empleando el esquema clásico [Fuente propia]

Las mejores constantes para esta formación resultan de la prueba P3, descrita en la Tabla 2.2, de la cual se obtuvieron las respectivas gráficas. Para esta prueba con formación columna, en el gráfico mostrado en la Figura 2.15, se observa que el robot seguidor no alcanza a ubicarse en el ángulo deseado detrás del robot líder debido a que el valor de 0 [rad] forma parte de las restricciones del algoritmo con esquema clásico. Sin embargo, a lo largo de 6 metros el robot seguidor logra mantener la distancia deseada, tal como se muestra en la Figura 2.14.

A continuación, para la siguiente prueba, se tiene los valores en la Tabla 2.3, con ángulo deseado $\varphi_d = -\frac{\pi}{4}[rad]$ y una distancia deseada $l_d = 0.5\ [m]$.

Esta prueba servirá de guía para realizar las formaciones en V y diamante, que podrán ser revisadas en la sección de resultados. En la formación V cuando se disponga de dos agentes, el primer robot seguidor tendrá un ángulo deseado de: $\varphi_d = -\frac{\pi}{4}[rad]$ y el segundo seguidor que estará ubicado al otro lado del robot líder deberá mantener un ángulo deseado de: $\varphi_d = \frac{\pi}{4}[rad]$. Asimismo, funcionará para el caso de formación en diamante si se dispone de tres robots seguidores, donde los dos primeros estarán ubicados a los lados y el tercer robot se ubicará en columna respecto del robot líder.

Tabla 2.3. Calibración de constantes con distancia y ángulo para formación diamante o V

$l_d = 0.5\ y\ \varphi_d = -\frac{\pi}{4}$	Constantes			ISE	
No. Prueba	k_1	k_2	k_3	**Distancia**	**Ángulo**
P1	2	5	1	1460,21	1747,87
P2	7	15	1.5	1283,47	1935,487
P3	**3**	**8**	**1**	**880,51**	**875,72**
P4	1.7	5.5	1.3	1786,41	2151,07
P5	1.9	5.1	1.2	1959,76	2600,25

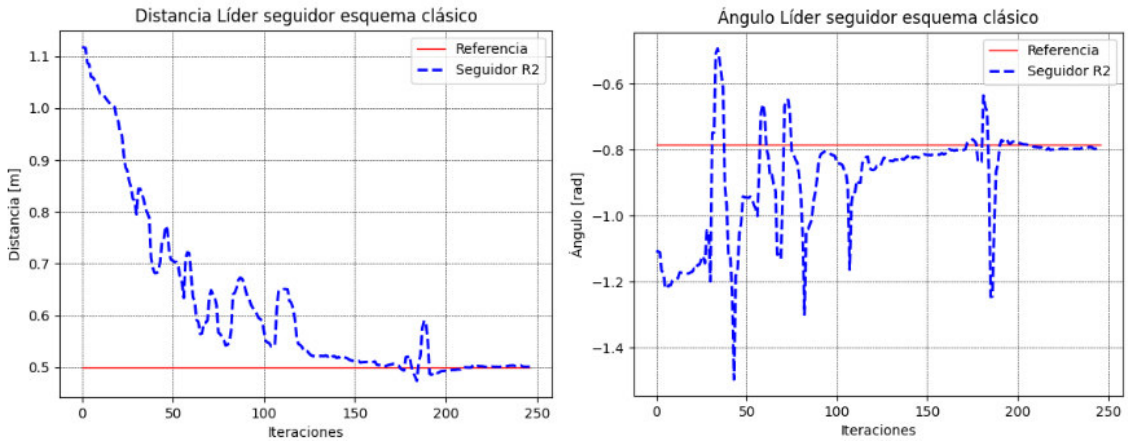


Figura 2.16. Resultados de distancia y ángulo para prueba con formación V o diamante empleando el esquema clásico [Fuente propia]

En este caso, se tiene como mejor calibración a la prueba P3, indicada en la Tabla 2.3. En la Figura 2.16 se puede observar como el robot seguidor alcanza las referencias tanto de distancia como ángulo a pesar de los picos que presenta. Esto se puede explicar de mejor manera al ver la trayectoria de los robots líder y seguidor en la Figura 2.17, ya que se puede apreciar que el robot seguidor para cumplir con los valores de distancia y ángulo tuvo que acomodarse varias veces, ya sea retrocediendo o girando.

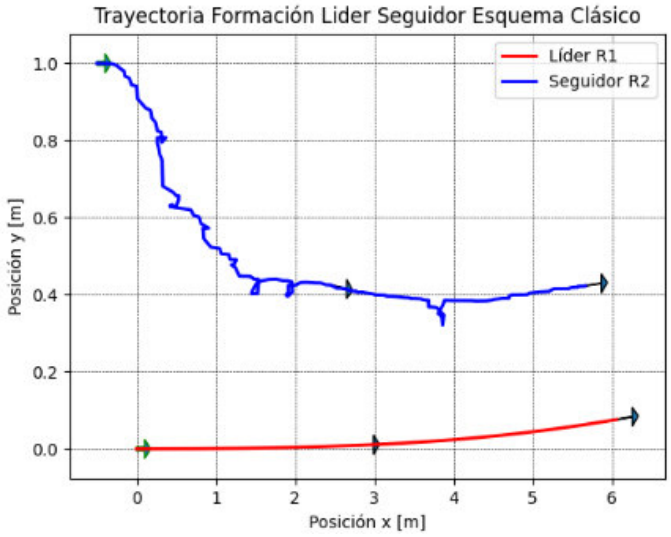


Figura 2.17. Trayectoria de robots líder y seguidor con formación V o diamante empleando el esquema clásico [Fuente propia]

Con el fin de determinar los mejores valores de constantes de calibración se presenta a continuación la comparación de los valores de ISE de ángulo y distancia en sus diferentes pruebas y formaciones.

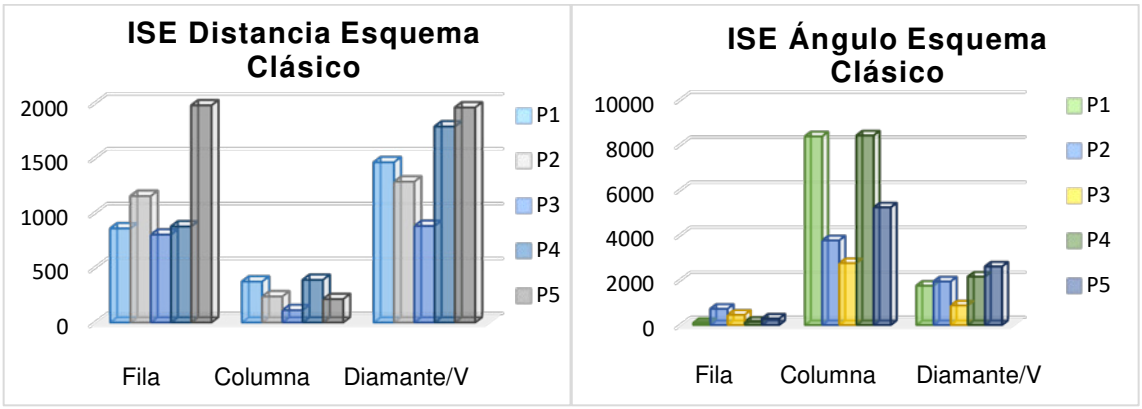


Figura 2.18. Comparación de valores ISE de distancia y ángulo de las pruebas realizadas en la formación líder seguidor con esquema clásico [Fuente propia]

A través de la Figura 2.18, se aprecia el resumen de los valores de ISE tanto de distancia como ángulo para las pruebas efectuadas con el algoritmo de esquema clásico; así se puede escoger las mejores constantes de calibración. En el caso de distancia los valores de ISE más bajos resultaron para la prueba con formación columna, esto se debe a que para esta prueba se tiene una distancia $l_d = 1[m]$, que dada las condiciones iniciales de posición es mucho más rápido alcanzar.

Por otro lado, al hablar de ángulo se tiene valores de ISE muy bajos en el caso de formación fila, esto se debe a que el valor de ángulo puesto estaba dentro de las limitaciones del control, lo que no pasa con la formación columna y diamante. En efecto, se puede observar que en todos los casos el valor de ISE más bajo corresponde a los resultados obtenidos en la prueba 3. Por lo tanto, para el algoritmo con esquema clásico las constantes escogidas para el cálculo de las leyes de control son: $k_1 = 3, k_2 = 8, k_3 = 1$

2.3.2 CALIBRACIÓN DEL CONTROLADOR PARA FORMACIÓN CON REFERENCIA VIRTUAL

Para la calibración del controlador con referencia virtual se aplica el mismo proceso de la sección anterior. Para este controlador, los valores de $l_d = 0{,}5\ y\ \varphi_d = \frac{\pi}{2}$ representan la formación fila y se realizan 5 pruebas con diferentes constates mostradas en la Tabla 2.4.

Tabla 2.4. Calibración de constantes con distancia y ángulo para formación fila

$l_d = 0,5\ y\ \varphi_d = \frac{\pi}{2}$	Constantes			ISE	
No. Prueba	k_1	k_2	k_3	**Distancia**	**Ángulo**
P1	6	15	1	997,845	237,318
P2	8	16	1,5	1110,53	168,062
P3	7,5	16,5	1,5	1061,365	157,322
P4	6,5	17	2	1970,239	203,267
P5	**7,2**	**16,3**	**1,2**	**555.739**	**149.35**

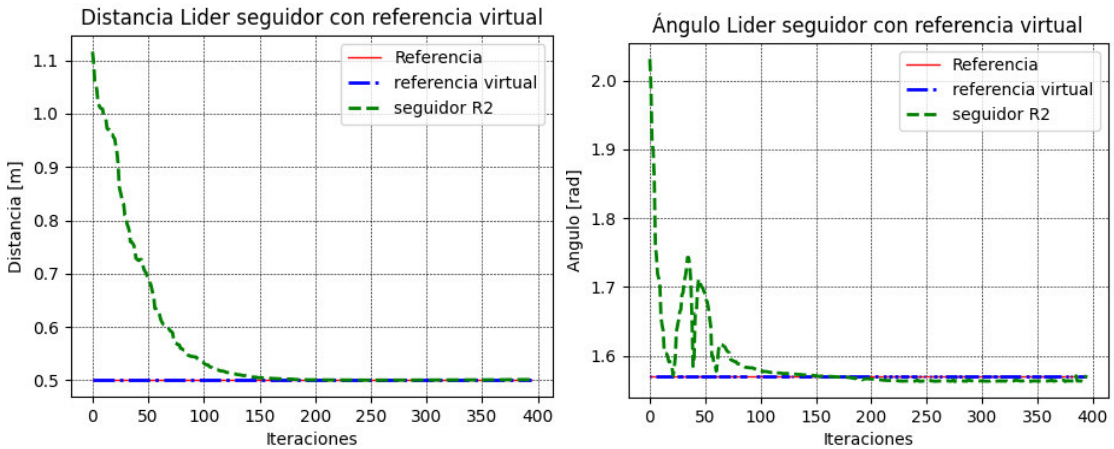


Figura 2.19. Resultados de distancia y ángulo para prueba con formación fila empleando referencia virtual [Fuente propia]

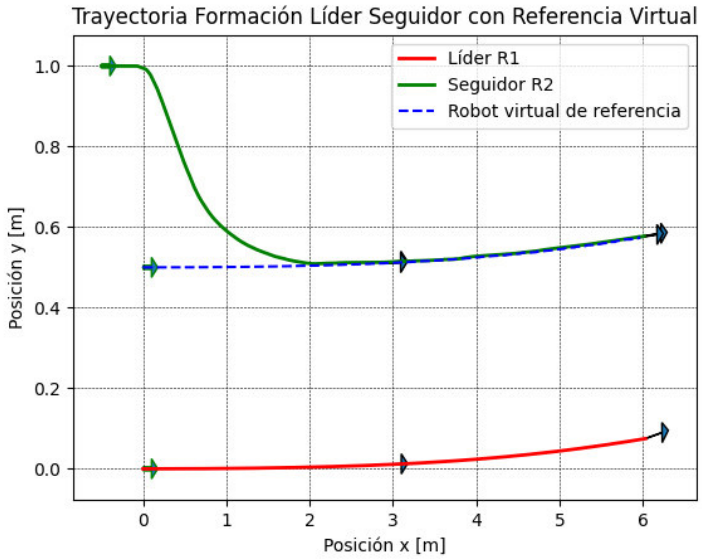


Figura 2.20. Trayectoria de robots líder y seguidor con formación fila empleando referencia virtual [Fuente propia]

Las figuras presentadas en esta sección son similares a la sección anterior, con la diferencia de que ahora se puede observar la referencia virtual. Así, para las gráficas de la Figura 2.19 se tiene de color rojo la referencia, de color azul la referencia virtual y de color verde la distancia y el ángulo del robot seguidor respectivamente. Por otra parte, en la Figura 2.20 se tiene la trayectoria del robot líder en color rojo, la trayectoria del robot seguidor en color verde y la trayectoria del robot virtual de referencia de color azul. Estas gráficas corresponden a los resultados obtenidos en la prueba 5 de la Tabla 2.4 y se observa que se alcanzan las referencias de ángulo y de distancia rápidamente. Además, es posible observar en la gráfica de la trayectoria que el robot seguidor llega a alcanzar la referencia dada a través del robot virtual, quedando así en la formación fila respecto al robot líder.

En la Tabla 2.5 se tiene un ángulo deseado $\varphi_d = \pi[rad]$ y una distancia deseada $l_d = 1\ [m]$ lo cual equivale a una formación columna.

Tabla 2.5. Calibración de constantes con distancia y ángulo para formación columna

$l_d = 1\ y\ \varphi_d = \pi$	Constantes			ISE	
No. Prueba	k_1	k_2	k_3	**Distancia**	**Ángulo**
P1	6	15	1	140.402	3081.73
P2	**8**	**16**	**1,5**	**115.751**	**2802.059**
P3	7,5	16,5	1,5	149.280	3876.524
P4	6,5	17	2	187.189	5098.282
P5	7,2	16,3	1,2	190.05	3066.216

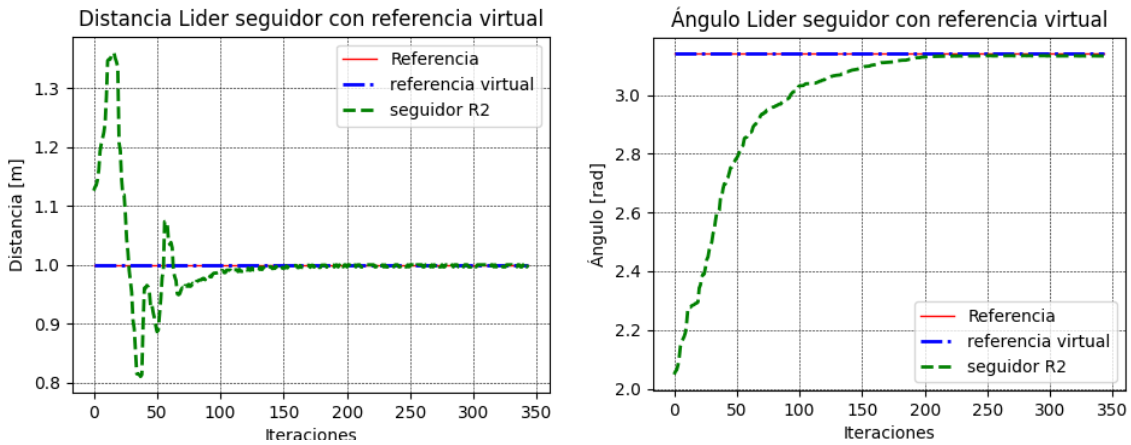


Figura 2.21. Resultados de distancia y ángulo para prueba con formación columna empleando referencia virtual [Fuente propia]

La Figura 2.21 corresponde a los resultados obtenidos en la prueba 2 de la Tabla 2.5 y se observa que se alcanzan las referencias de distancia y ángulo rápidamente y el robot seguidor logra mantenerse en formación columna detrás del robot líder, tal como se ve en la gráfica de la trayectoria en la Figura 2.22.

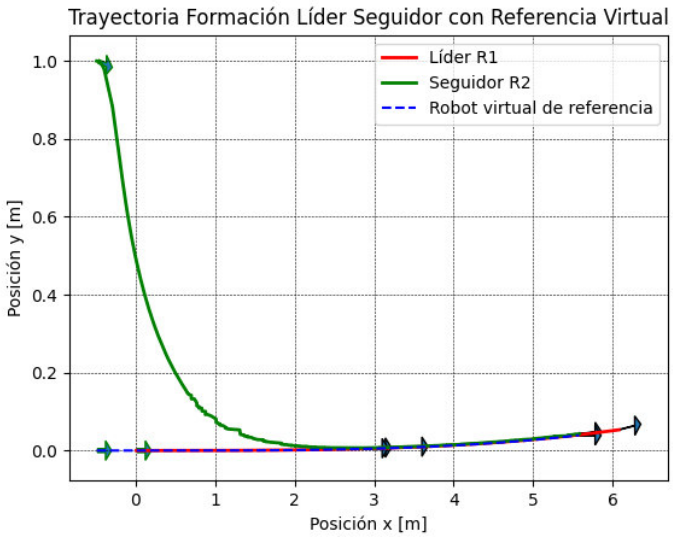


Figura 2.22. Trayectoria de robots líder y seguidor con formación fila empleando referencia virtual [Fuente propia]

Al igual que se realizó en el algoritmo con esquema clásico, la siguiente prueba servirá de guía para realizar formaciones en V si se dispone de dos agentes seguidores o una formación diamante si se dispone de tres agentes seguidores con referencia virtual. En la Tabla 2.6 se tiene un ángulo deseado $\varphi_d = \frac{\pi}{4}[rad]$ y una distancia deseada $l_d = 0.5\ [m]$.

Tabla 2.6. Calibración de constantes con distancia y ángulo para formación V o Diamante

$l_d = 0.5\ y\ \varphi_d = \frac{\pi}{4}$	Constantes			ISE	
No. Prueba	k_1	k_2	k_3	**Distancia**	**Ángulo**
P1	6	15	1	356.466	944.796
P2	8	16	1,5	871.144	2072.409
P3	7,5	16,5	1,5	700.679	1987.176
P4	6,5	17	2	924.570	2407.154
P5	**7,2**	**16,3**	**1,2**	**435.972**	**653.175**

En la Figura 2.23 se observa que el robot seguidor alcanza rápidamente la referencia de distancia y de ángulo. De la misma forma, se puede apreciar en la gráfica de la trayectoria representada en la Figura 2.24. Las gráficas corresponden a la prueba 5 de la Tabla 2.6.

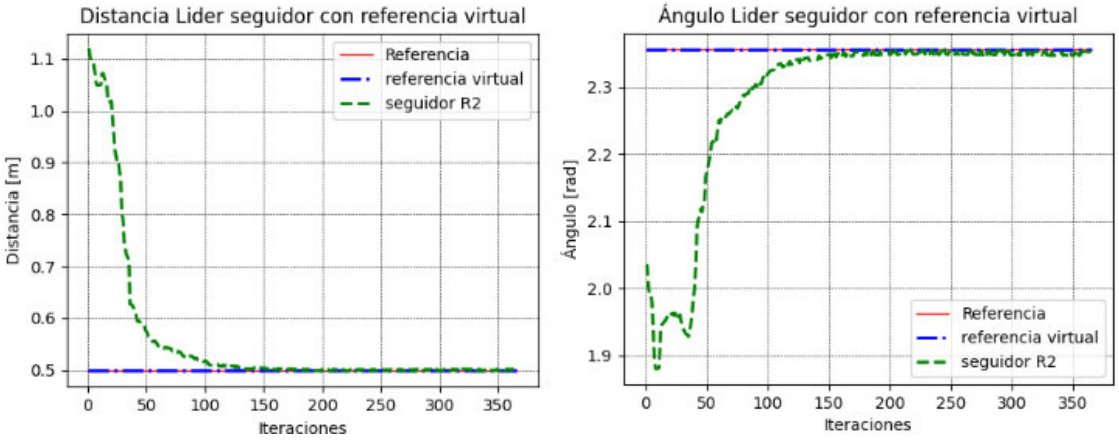


Figura 2.23. Resultados de distancia y ángulo para prueba con formación V o diamante empleando referencia virtual [Fuente propia]

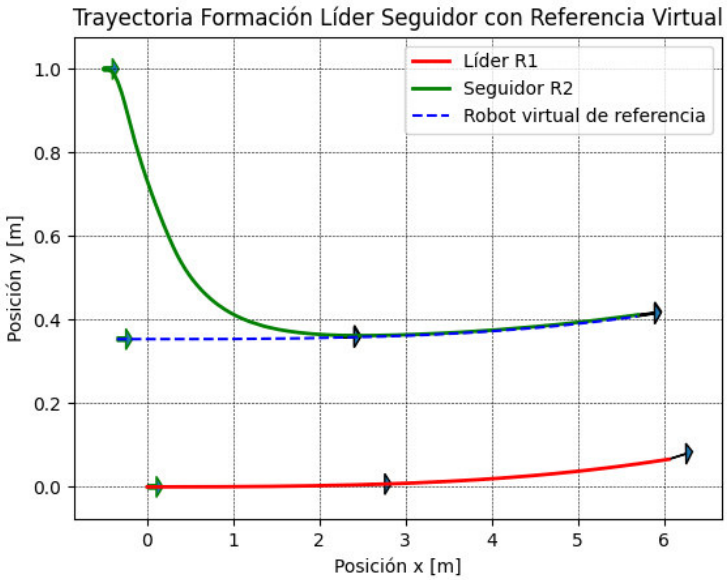


Figura 2.24. Trayectoria de robots líder y seguidor con formación V o diamante empleando referencia virtual [Fuente propia]

A lo largo de las pruebas realizadas se ha podido observar que en efecto el algoritmo con referencia virtual presenta mejoras ya que los robots seguidores alcanzan las referencias especificadas rápidamente contrario a lo que ocurre con el algoritmo con esquema clásico. Por esta razón, con el fin de identificar las mejores constantes de calibración para el algoritmo de referencia virtual se presenta a continuación el resumen de pruebas realizadas comparando los valores ISE de distancia y ángulo.

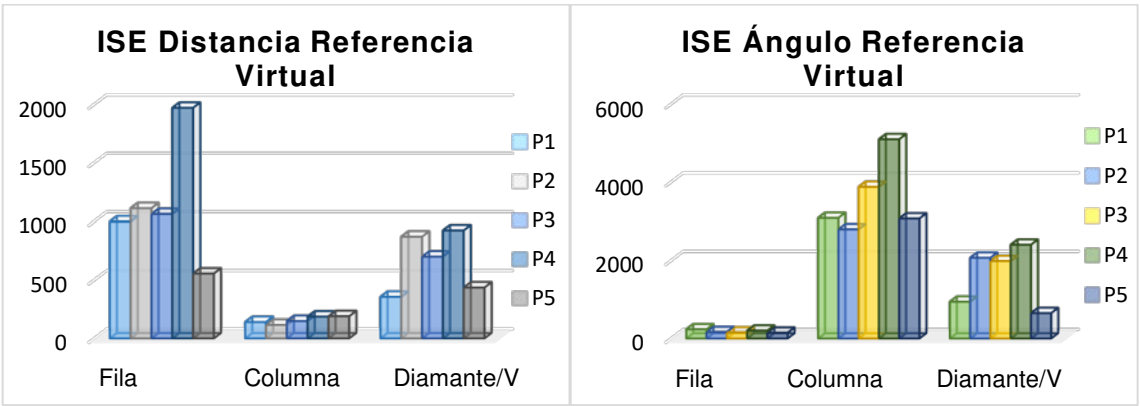


Figura 2.25. Comparación de valores ISE de distancia y ángulo de las pruebas realizadas en la formación líder seguidor con referencia virtual [Fuente propia]

Al igual que en la calibración del controlador para formación con esquema clásico, a través de los índices de desempeño obtenidos y mostrados en la Figura 2.25, se determinan las mejores constantes para el controlador con referencia virtual, siendo las de la prueba 5 las

que presentaron menor ISE y por lo tanto un mejor desempeño. Así, las constantes de control escogidas corresponden a $k_1 = 7.2, k_2 = 16.3, k_3 = 1.2$**.**

Es importante resaltar que las pruebas realizadas en la Sección 2.3.1 y 2.3.2 se efectuaron bajos las mismas condiciones iniciales y de ejecución, por lo que también son útiles para comparar el desempeño entre el esquema clásico y la referencia virtual. De modo que, se puede determinar que el algoritmo de control de formación líder seguidor empleando referencia virtual tiene un mejor desempeño ya que en general presentó un menor ISE en todos los casos, cuestión que se ve reflejada gráficamente al alcanzar rápidamente la referencia de distancia y de ángulo.

Dado que el algoritmo con referencia virtual basa sus fundamentos matemáticos en el algoritmo con esquema clásico, se puede observar a través de la Figura 2.25, que los valores ISE de ángulo para la formación fila son bastante bajos con respecto a la formación en columna. Esto se debe principalmente a que el valor de ángulo fijado para la formación fila está contemplado dentro de las restricciones del algoritmo con esquema clásico, lo que no pasa con el ángulo fijado para realizar la formación columna. Por lo que, al ocupar el algoritmo con referencia virtual se observa una mejora en la formación columna ya que el robot seguidor puede alcanzar las referencias de distancia y ángulo, es decir, que pueda estar detrás del robot líder. Así también, se observa que el ISE de distancia para la formación columna es bajo en comparación con otras formaciones debido a que es más fácil alcanzar la distancia deseada desde las posiciones iniciales dadas.

3 RESULTADOS, CONCLUSIONES Y RECOMENDACIONES

3.1 RESULTADOS

Una vez finalizada la sintonización de los controladores, con la cual se obtuvieron las constantes que se utilizan para todos los robots seguidores que aparezcan dentro del sistema multi-agente, se procede a implementar las diferentes formaciones dentro del entorno ROS-Gazebo. Todos estos resultados se podrán observar mediante videos demostrativos, el detalle de estos se especifica en Anexo III.

En la Figura 3.1, se puede observar cómo están distribuidos los nodos y los tópicos del sistema multi-agente. Todos los nodos de los robots seguidores se subscriben al tópico de velocidad y odometría del robot líder y al tópico de odometría de su respectivo robot, debido a que necesitan de esta información, como se mostró en las ecuaciones de la Sección 2.1. De manera similar, los nodos de los robots seguidores publican sobre el tópico de velocidad

de su respectivo robot a partir de los valores obtenidos de la velocidad lineal y angular de las ecuaciones de control utilizadas. En el caso del nodo del robot líder, este únicamente publica el comando de velocidad del robot 1 para realizar su control de velocidad, como se mostró en el primer tomo del trabajo en conjunto [1].

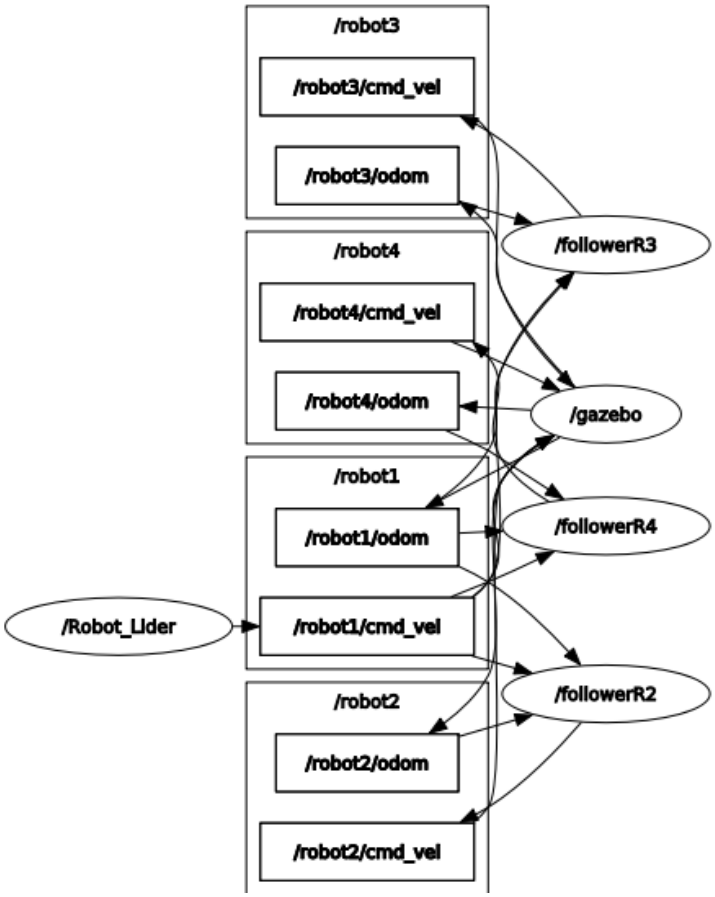


Figura 3.1. Diagrama de nodos y tópicos del sistema multi-agente implementado con ROS-Gazebo [Fuente propia]

El esquema mostrado en la Figura 3.1 es el mismo para ambos algoritmos de control. La diferencia se encuentra dentro de los nodos llamados followerRn, donde $n = 2,3,4$, ya que son los que contienen las ecuaciones de control correspondientes al algoritmo con esquema clásico o referencia virtual.

Los resultados obtenidos que se presentan a continuación fueron realizados bajo las condiciones iniciales donde el robot seguidor R2 inicia en la posición $x = -0.5\ [m]\ , y = 1\ [m]$, robot seguidor R3 inicia en $x = -2\ [m]\ , y = 1.5\ [m]$, el robot seguidor R4 en $x = -1.5\ [m]\ , y = 2\ [m]$ y el robot líder inicia en la posición $x = 0\ [m]\ , y = 0\ [m]$. Durante la ejecución, los robots avanzan 5 [m] con velocidad lineal 0.109 [m/s] y velocidad angular nula.

3.1.1 RESULTADOS FORMACIÓN FILA

En la Figura 3.2 se observan los resultados para una formación fila usando el esquema clásico y referencia virtual empleando 4 agentes, un robot líder y tres seguidores. Ambos resultados fueron tomados bajo las mismas condiciones iniciales y de operación para poder realizar una comparativa entre algoritmos. Es así como, se puede observar que el algoritmo de formación líder seguidor con referencia virtual alcanza la formación deseada con una mejor trayectoria de los robots seguidores. Por otro lado, el algoritmo de formación líder seguidor con esquema clásico presenta una gráfica con más tropiezos en la trayectoria de los robots, es decir, que representa aceleración, frenados repentinos y en determinadas ocasiones retrocesos en su trayectoria para lograr alcanzar su posicionamiento.

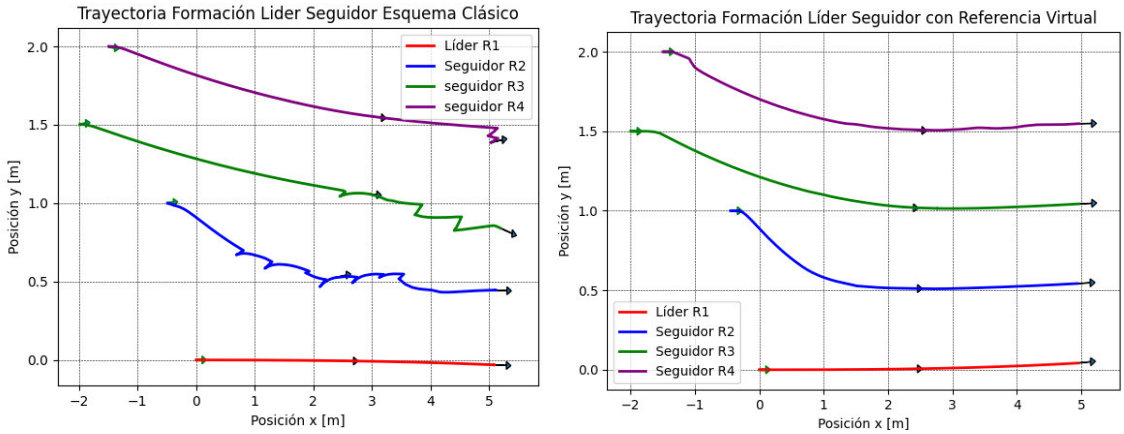


Figura 3.2. Resultados formación fila [Fuente propia]

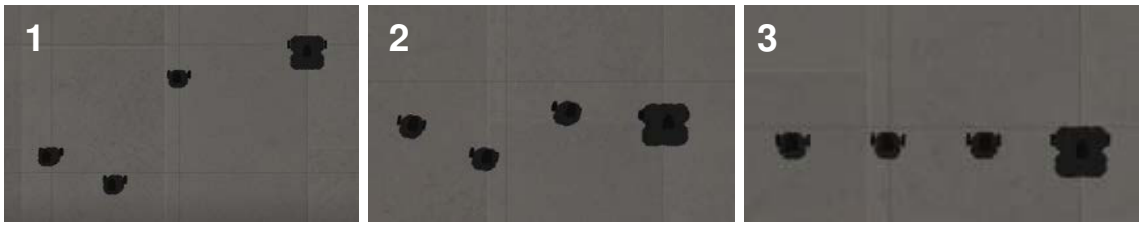


Figura 3.3. Resultados formación fila en Gazebo [Fuente propia]

La Figura 3.3 y el resto de las figuras tomadas de Gazebo y presentadas en esta sección corresponden a los resultados obtenidos con el algoritmo de formación líder seguidor con referencia virtual. Se tiene la secuencia de tres estados, en el estado inicial se puede observar cómo los robots seguidores comienzan a acercarse al robot líder desde sus posiciones iniciales, posteriormente en el segundo estado al avanzar el robot líder los robots seguidores también se empiezan a formar y finalmente en el último estado se tiene a todo el sistema multi-agente formado. No se incluyen gráficas de Gazebo del algoritmo de formación líder seguidor con esquema clásico ya que son bastante similares y no se determinaría la diferencia entre ambas resultando poco útiles para el trabajo. Estas figuras

permiten observar de una manera más gráfica el cumplimiento de las diferentes formaciones propuestas y su implementación en un entorno de simulación libre de obstáculos utilizando ROS-Gazebo.

3.1.2 RESULTADOS FORMACIÓN COLUMNA

La Figura 3.5 muestra la formación columna, al igual que en la formación fila el algoritmo de formación líder seguidor empleando referencia virtual proporciona un mejor desempeño ya que alcanza rápidamente la formación deseada. No es así el caso del algoritmo de formación con esquema clásico que no logra formarse en el lapso de distancia especifico de la prueba debido a las limitaciones de este. El sistema multi-agente formado se puede observar en el software de simulación Gazebo mostrado en la Figura 3.5, y al igual que en la formación fila, se tienen 3 estados en donde se puede ver como los robots seguidores comienzan a formarse en columna respecto del robot líder.

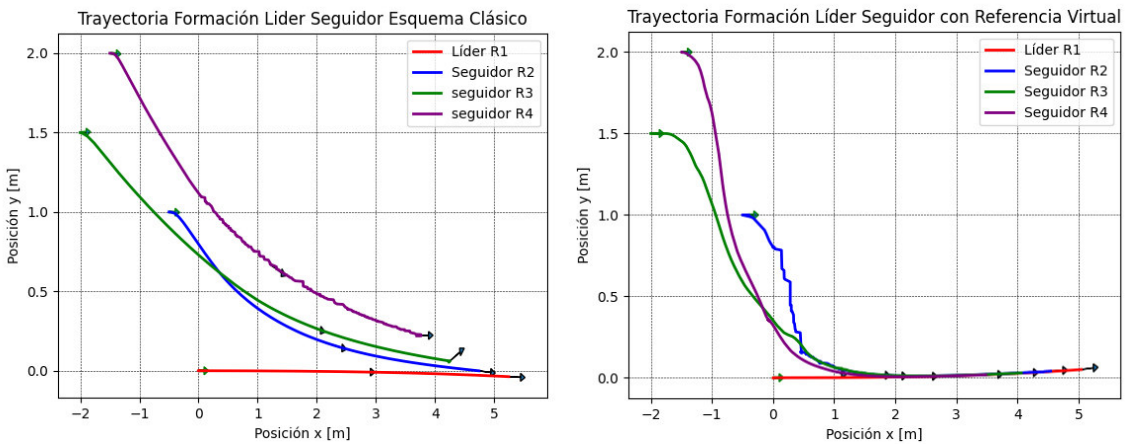


Figura 3.4. Resultados formación columna [Fuente propia]

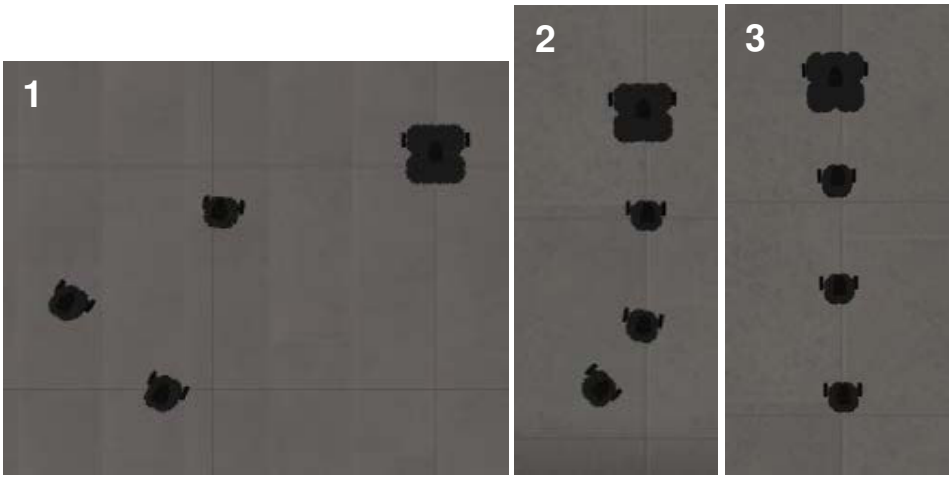


Figura 3.5. Resultados formación columna vista en Gazebo [Fuente propia]

3.1.3 RESULTADOS FORMACIÓN V

En la Figura 3.6 se presentan los resultados de la formación en V, es evidente que esta formación se logra únicamente para un número par de agentes seguidores, en este caso se lo ha realizado con el número mínimo de 2 agentes seguidores. Para esta formación ambos algoritmos de control alcanzan sin problema la formación y ofrecen un desempeño similar. La secuencia de la formación del sistema multi-agente en formación V se puede observar en el software de simulación Gazebo mostrado en la Figura 3.7. En el primer estado se observa como a los robots seguidores están en la posición inicial y seguidamente en el segundo se observa como los robots se empiezan a desplazar para finalmente en el tercer estado se realice la formación en V.

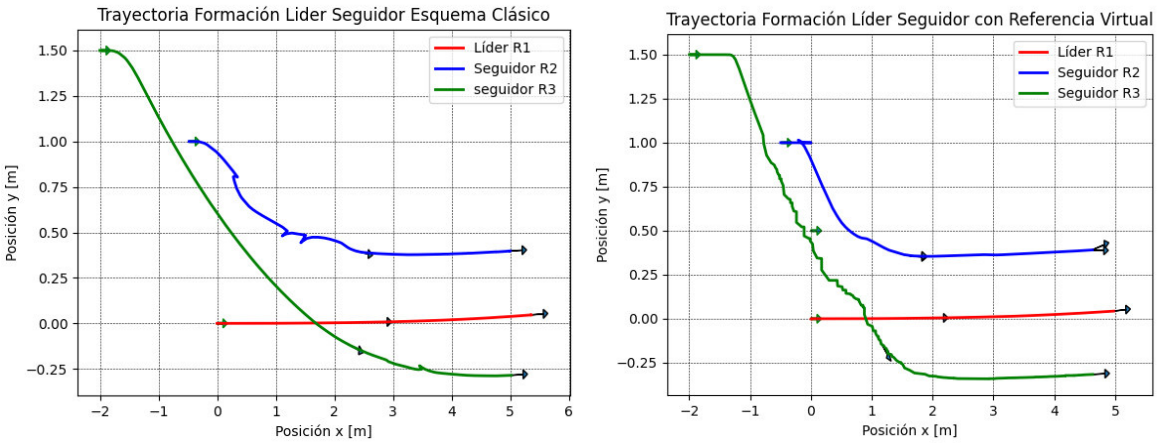


Figura 3.6. Resultados formación V [Fuente propia]

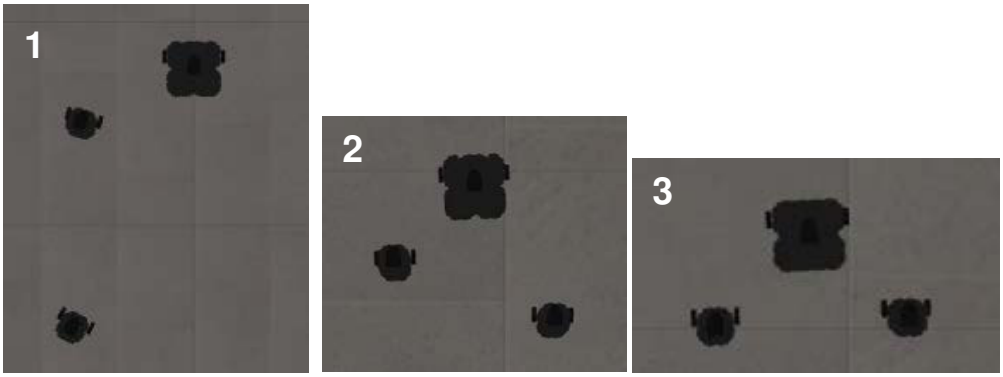


Figura 3.7. Resultados formación V, vista en Gazebo [Fuente propia]

3.1.4 RESULTADOS FORMACIÓN DIAMANTE

En la Figura 3.8 se observan las trayectorias de los robots para alcanzar una formación en diamante. Esta formación es posible únicamente para un agente líder y tres agentes seguidores. Al comparar ambos algoritmos de control se puede observar que en el

esquema clásico el robot cuatro (línea morada) no logra ubicarse detrás del robot líder, esto debido a las limitaciones de ángulo mencionadas en la Sección 2.1.1. Mientras tanto, el algoritmo que utiliza referencia virtual nuevamente proporciona un mejor desempeño ya que todos los robots alcanzan su posición deseada. El sistema multi-agente en formación diamante en el entorno de simulación Gazebo se puede observar en la Figura 3.9.

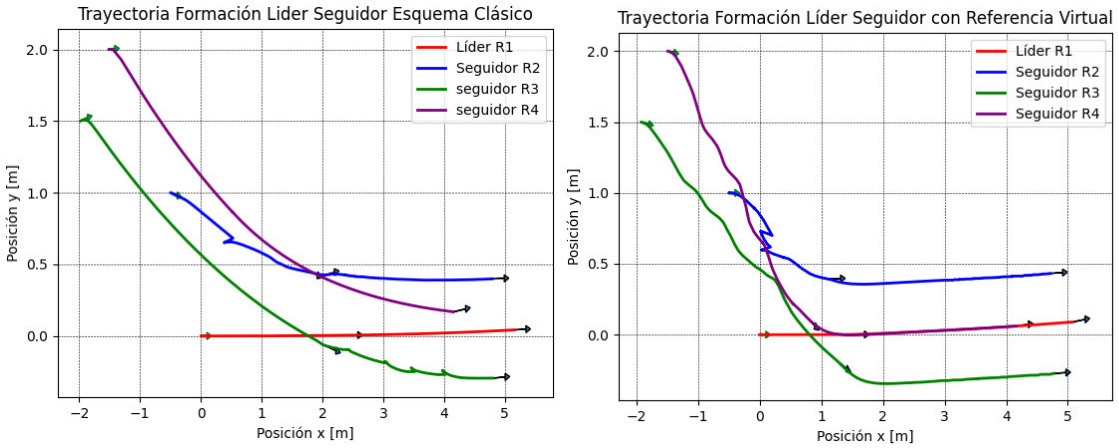


Figura 3.8. Resultados formación Diamante [Fuente propia]

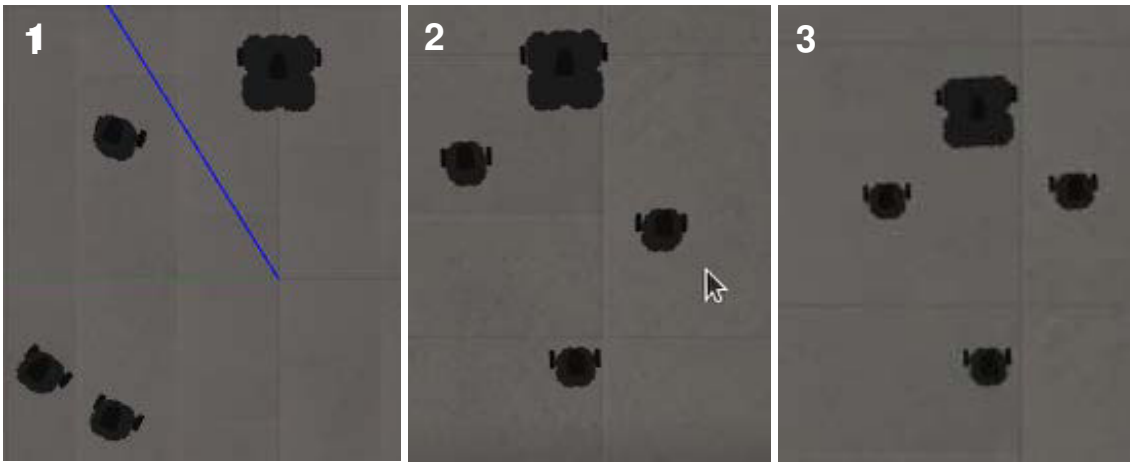


Figura 3.9 Resultados formación Diamante, vista en Gazebo [Fuente propia]

3.1.5 RESULTADOS FORMACIÓN FILA CON ROS MOBILE

En el primer tomo del trabajo desarrollado en conjunto, se presenta una alternativa para realizar el movimiento de un robot a partir de una aplicación móvil para teléfonos inteligentes con sistema Android [1]. Considerando esto, se vio oportuno realizar el movimiento del robot líder mediante esta opción, así el usuario será capaz no sólo de mover un robot sino toda la formación de un sistema multi-agente.

Para este resultado se consideraron las posiciones iniciales de los robots seguidores descritas anteriormente, pero no las velocidades lineal y angular del robot líder. Debido a que los valores de la velocidad se modificarán dependiendo del movimiento que realice el usuario con el joystick digital que presta la aplicación. Además, esta opción se ha implementado con la formación en fila y utilizando el algoritmo con referencia virtual

En la Figura 3.10, se tiene el diagrama de nodos y tópicos del sistema multi-agente con el movimiento del robot líder comandado por un joystick digital. La comunicación entre los nodos y tópicos es igual a la explicación anteriormente descrita en la parte inicial de resultados. En lo que se diferencia, es que en este diagrama ya no se presenta el nodo /Robot_Lider, en su lugar aparece el nodo de la aplicación móvil, /robot1/cmd_vel el cual publica la velocidad lineal y angular sobre el tópico /robot1/cmd_vel.

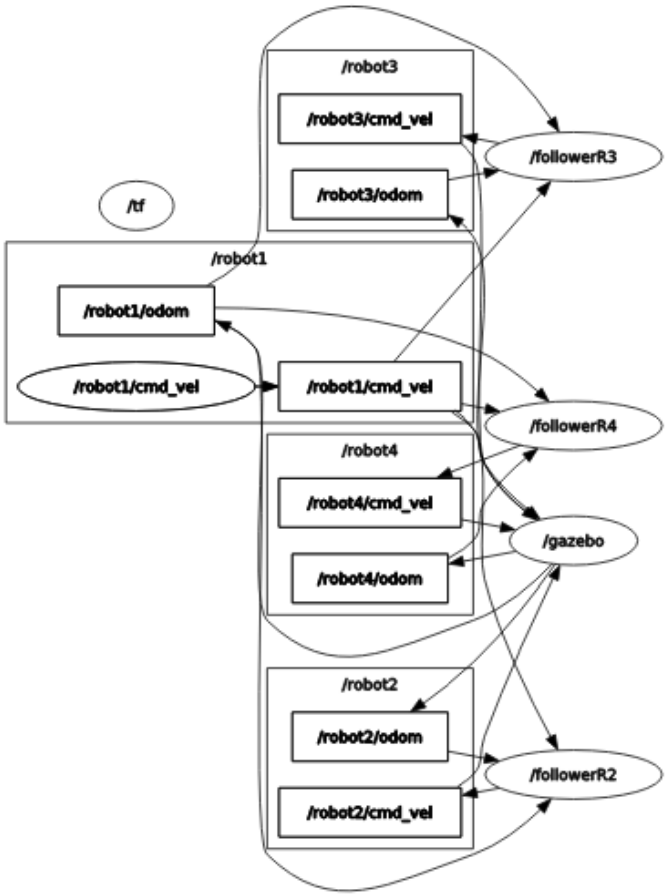


Figura 3.10. Diagrama de nodos y tópicos del sistema multi-agente implementado con ROS-Gazebo utilizando ROS Mobile [Fuente propia]

Como se puede apreciar en la Figura 3.11, la trayectoria de los robots dependerá bastante de la maniobrabilidad que el usuario tenga sobre el robot líder, es por esta razón que se observa que el robot 4 da una vuelta para cumplir con los valores de distancia y ángulo que

aseguren una formación fila. Además, a través de la Figura 3.12 y la Figura 3.13 se puede observar el comportamiento de la distancia y ángulo para el robot 2 con conducción por ROS Mobile y por teclado, respectivamente. En el caso de conducción por ROS Mobile a pesar de los movimientos libres que realiza el usuario mediante el joystick digital, se busca cumplir con la referencia de distancia. Pero, en la gráfica del ángulo se observa que tiene un error respecto a la referencia, esto se debe principalmente al manejo del robot líder por el usuario ya que mientras ocurre el desplazamiento de la formación, el robot seguidor tiene que acomodarse mediante giros para cumplir con la formación fila. Esto no ocurre en las gráficas de la Figura 3.13 ya que al conducir al robot líder por teclado es posible dejar seteado la velocidad lineal con la que se desee que avance.

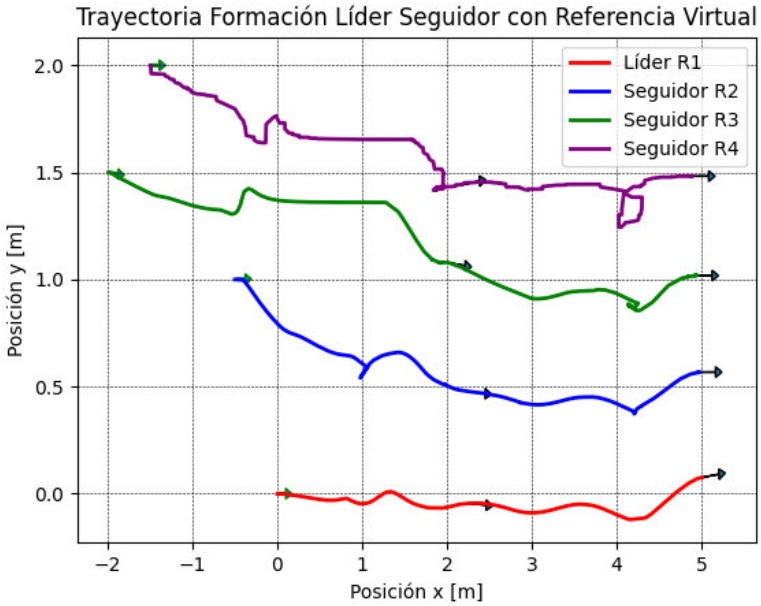


Figura 3.11. Resultados formación fila con ROS Mobile [Fuente propia]

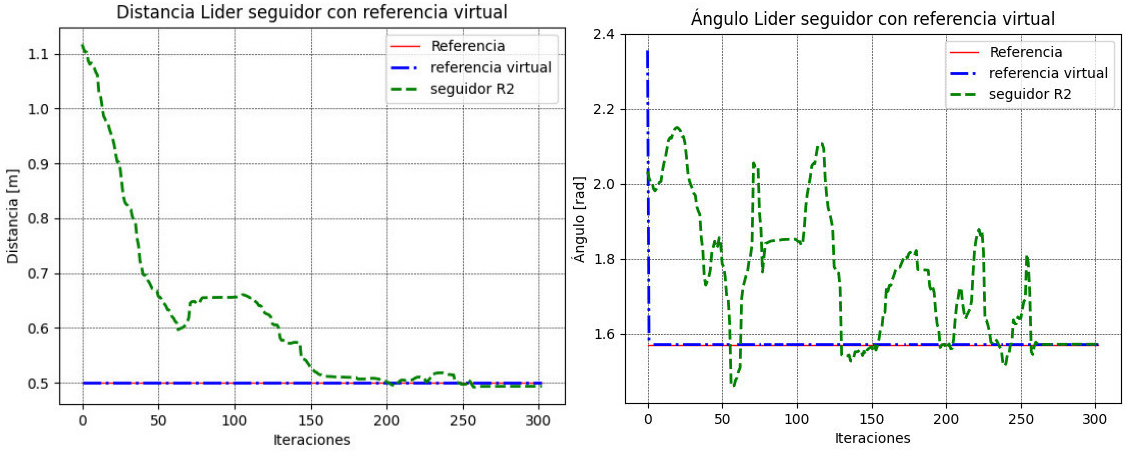


Figura 3.12. Resultados de distancia y ángulo para prueba con formación fila empleando referencia virtual con conducción por ROS Mobile [Fuente propia]

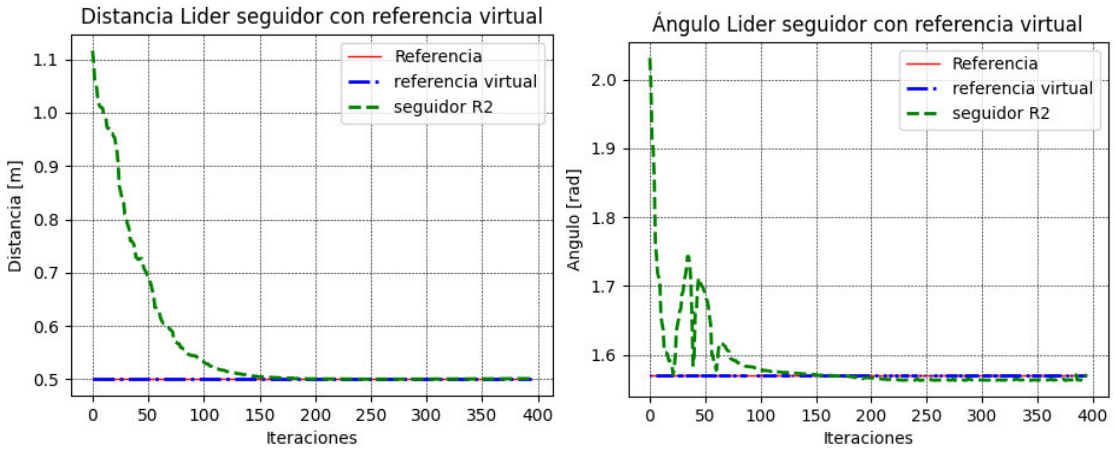


Figura 3.13. Resultados de distancia y ángulo para prueba con formación fila empleando referencia virtual con conducción por teclado [Fuente propia]

Los gráficos presentados en la sección anterior que muestran la distancia y el ángulo del robot seguidor, así como los gráficos de trayectoria utilizados en esta sección se elaboraron utilizando la librería de Python Matplotlib.

3.2 CONCLUSIONES

- Como resultado de las investigaciones realizadas sobre las características de una formación líder seguidor, se encontró que este tipo de formación es el más utilizado, ya que su funcionamiento se basa en que los robots seguidores mantengan una distancia y ángulo previamente definidos respecto del robot líder. Por lo que existen algoritmos mucho más robustos que otros como es el caso del algoritmo implementado con referencia virtual versus el algoritmo con esquema clásico.

- El uso del entorno ROS-Gazebo facilita el desarrollo de aplicaciones como la mostrada en el presente trabajo. Implementar un sistema multi-agente sin el soporte de este software sin duda hubiera presentado un reto de mayor magnitud ya que se deberían resolver problemas de comunicación entre agentes y no se contaría con una gran comunidad, como la de ROS, que facilite información y tutoriales al respecto.

- A partir de las pruebas realizadas, se puede concluir que en efecto se logró simular y controlar un sistema multi-agente homogéneo de al menos dos agentes, un robot líder y dos seguidores que logren mantener una formación líder seguidor, mediante el uso de plataformas TurtleBot3 en un entorno desarrollado a partir de los softwares ROS/Gazebo.

- En definitiva, se diseñaron e implementaron dos algoritmos de control, uno llamado esquema clásico y el otro conocido como referencia virtual, para realizar una formación líder seguidor de un sistema multi-agente, donde el robot líder es manejado mediante un usuario y los robots seguidores mantuvieron una distancia y ángulo previamente definidos respecto del robot líder.
- La comparación de los dos algoritmos al realizar diferentes formaciones (fila, columna, V, diamante), verificó el funcionamiento de estos. Evidenciando que, el algoritmo con esquema clásico presenta la limitación del ángulo definido para realizar la formación columna, mientras que en las otras formaciones fila, V o diamante requiere más tiempo para llevar a cabo la tarea. Por otro lado, el algoritmo con referencia virtual es más rápido y no posee ninguna restricción en cuanto a los ángulos previamente definidos.
- Finalmente, las restricciones en ambos algoritmos están vinculadas al hecho de que el robot líder es controlado por el usuario y dependerá de la experiencia y maniobrabilidad que tenga el usuario manejando un TurtleBot3 ya sea mediante teclado o ROS Mobile, para realizar la formación de modo que los robots no colisionen entre ellos, ya que esta acción no formó parte del alcance del presente trabajo.

3.3 RECOMENDACIONES

- Para entender de mejor manera cómo se llevó a cabo la implementación del sistema multi-agente en el entorno ROS Gazebo se recomienda revisar el trabajo escrito referente al primer tomo [1].
- Habiendo ya desarrollado la simulación de un sistema multi-agente con control de formación líder seguidor dentro del entorno ROS-Gazebo y comprobado su funcionamiento. En el caso de ser posible, se sugiere la implementación física de todo el sistema en base a plataformas TurtleBot reales.
- Teniendo en cuenta que se tiene la posibilidad de colisiones entre agentes por una mala operación del robot líder y por la ejecución de varios cambios de formación, es recomendable que en futuros trabajos se desarrolle algoritmos de control de trayectoria para el robot líder o algoritmos de evasión de inter-colisiones y obstáculos.

- Aprovechando las capacidades mejoradas del robot líder para guiar al grupo de robots, se recomienda hacer uso de la cámara que posee el Turtlebot3 Waffle Pi a través de la herramienta RViz que ofrece ROS para que el operador pueda observar el entorno en el que se desempeña el sistema multi-agente.
- Se recomienda al usuario que, al emplear el algoritmo con referencia virtual, para realizar cambios de formación se aumente la velocidad lineal del robot líder a un valor de 0.18[m/s]. Esto con el objetivo de apreciar mejor el cambio de formación y evitar posibles colisiones entre agentes del sistema por mal manejo del movimiento del robot líder.
- Antes de utilizar la opción de ROS Mobile para el control del robot líder, se recomienda ejecutar el sistema con el teclado y así poder escoger el tipo de formación que se desea puesto que la opción de ROS Mobile no permite el cambio de formación.

4 REFERENCIAS BIBLIOGRÁFICAS

[1] R. A. Ayala, «Diseño, control y simulación mediante el uso del entorno ros/gazebo, de un sistema multi-agente conformado por plataformas turtlebot3 enfocado a mantener una formación líder seguidor (Tomo 1),» Escuela Politécnica Nacional, Quito, 2022.

[2] A. O. González Medina, «Modelado y Control de Sistemas Multi-agente Heterogéneos Basado en Distancia y Ángulo entre los Agentes,» RIITEC, Coahuila Mexico, 2019.

[3] K. L.-B. Y. Shoham, Multiagent systems: algorithmic, game-theoretic, and logical foundations, Cambridge University Press, 2009.

[4] A. R. Cheraghi, S. Shahzad y K. Graffi, «Past, Presente, and Future of Swarm Robotics,» 03 Enero 2021. [En línea]. Available: https://arxiv.org/pdf/2101.00671.pdf. [Último acceso: 14 12 2021].

[5] P. Alfaro, «Control de Robots Móviles autónomos en formación usando el esquema Líder-Seguidor,» Pontificia Universidad Católica del Perú, Perú, 2020.

[6] M. Infante-Jacobo, R. Cruz-Morales, M. Velasco Villa y A. Rodríguez-Ángeles, «Formación líder-seguidor para robots diferenciales con parámetros de seguimiento de variables,» Centro de Investigación y de Estudios Avanzados del IPN, Mexico, 2019.

[7] J. Aguilar, A. Ríos, F. Hidrobo y M. Cerrada, Sistems MultiAgentes y sus Aplicaciones en Automatización Industrial, Mérida: Universidad de Los Andes - Venezuela, 2013.

[8] A. Mas, Agentes Softare y Sistemas Multiagente: Conceptos, Arquitecturas y Aplicaciones, Madrid: Pearson Prentice Hall, D.L. , 2004.

[9] J. Vidal, «Fundamentals of Multiagent Systems with NetLogo Examples,» 03 2010. [En línea]. Available: https://jmvidal.cse.sc.edu/papers/mas.pdf. [Último acceso: 11 2021].

[10] A. Cheraghi, S. Shahzad y K. Graffi, «Past, Present and Future of Swarm Robotics,» Heinrich Heine Universitat Dusseldorf, 2021.

[11] P. Faria Dias, M. Silva, G. Rocha Filho, P. Vargas, L. Cota, G. Pessin y et al, «Swarm Robotics: A Perspective on the Latest Reviewed Concepts and Applications,» 2021. [En línea]. Available: https://doi.org/10.3390/s21062062.

[12] S. Shakkottai, T. Rappaport y P. Karlsson, «Cross-Layer design for wireless networks,» *IEEE Communications Magazine,* vol. 10, nº 41, pp. 74-80, 2033 oct.

[13] M. Serrano y E. Aranda Bricaire, «Coordinación de movimiento para sistemas multiagente heterogéneos,» 2014. [En línea]. Available: http://amca.mx/memorias/amca2014/media/files/0209.pdf. [Último acceso: 8 12 2021].

[14] M. Ekelhof y P. Giacomo, «Robótica de enjambre Informe de Investigación,» UNIDIR, 2020.

[15] L. V. Vicente Calderita, V. Moreno y B. Curto, «Navegación de Robots en Formación: Un Enfoque Reactivo con Restricciones,» Universidad de Salamanca, Salamanca - España.

[16] R. Adriana y S. E. Leonardo, «Formación de Robots Móviles Mediante el Uso de Controladores,» *Ing. USBMed,* vol. 4, pp. 62-65, Julio-Diciembre 2013.

[17] K. Sung-Mo, P. Myoung-Chul, L. Byung-Hun y A. Hyo-Sung, «Distance-based formation control with a single moving leader,» IEEE, Portland, 2014.

[18] L. Arcos Enriquez y C. Calala Ayala, «Diseño e implementación del control de la formación de un grupo de robots móviles terrestres de tamaño reducido enfocado al seguimiento de trayectoria basado en el sistema operativo robótico (ROS),» Escuela Politécnica Nacional, Quito, 2020.

[19] B. Caizaluisa Guerra y M. Morocho Oña, «Diseño e Implementación de un Sistema Robótico Móvil Cooperativo para Detección y Análisis de Incendios en Ambientes Controlados,» Universidad de las Fuerzas Armadas ESPE, Sangolquí, 2016.

[20] L. E. Solaque Guzmán, D. R. Avendaño Florez, M. A. Molina Villa y C. A. Pulido Rojas, «Sistema de Transporte cooperativo desarrollado para un grupo de robots

móiles no-holonómicos usando el método Líder Virtual,» Universidad Militar Nueva Granada, UMNG, Bogotá, 2015.

[21] J. Desai, J. Ostrowski y V. Kumar, «Modeling and Control of Formations of Nonholonomic Mobile Robots,» Departamental Papers (MEAM), Pennsylvania, 2001.

[22] G. Dongbing y W. Zongyao, «Leader-Follower Flocking: Algorithms and Experiments,» *IEEE Transactions on Control Sysems Technology,* vol. 17, nº 5, pp. 1211-1219, 2009.

[23] S. Ahmadi y H. Werner, «Cascaded formation control using angle and distance between agents with orientation control part 1 and part 2,» *International Conference on Control,* 2016.

[24] L. Consolini, F. Morbidi, P. Domenico y M. Tosques, «A Geometric Characterization of Leader-Follower Formation Control,» University of Parma, Parma, Italy.

[25] J. Alberto y M. Mendoza, «Diseño de Controladores Robustos para Sistemas Electromecánicos,» IPN, 2016.

[26] C. Samson y K. Ait-Abderrahim, «Feedback Control of a Nonholonomic Wheeled Cart in Cartesian Space,» International Conference on Robotics and Automation, France, 1991.

5 ANEXOS

ANEXO I. Repositorio digital

ANEXO II. Manual de usuario

ANEXO III. Link a videos demostrativos

ANEXO I. Repositorio digital:

El anexo presentado en esta sección se puede encontrar de igual manera en el primer tomo del trabajo desarrollado en conjunto.

Dado que se ocupa softwares de código abierto como ROS y Gazebo. El proyecto desarrollado en conjunto por los autores Rodrigo Ayala y Gabriela Romero, se encuentra en el repositorio digital GitHub.

Para acceder a este repositorio se debe dirigir al siguiente link dando Ctrl+click, sobre las letras que están subrayadas y en azul:

Link: GitHub - raaptoy/Leader_Follower_Formation_TurtleBot3 at master

Al entrar al link, se encontrará con la Figura I.1:

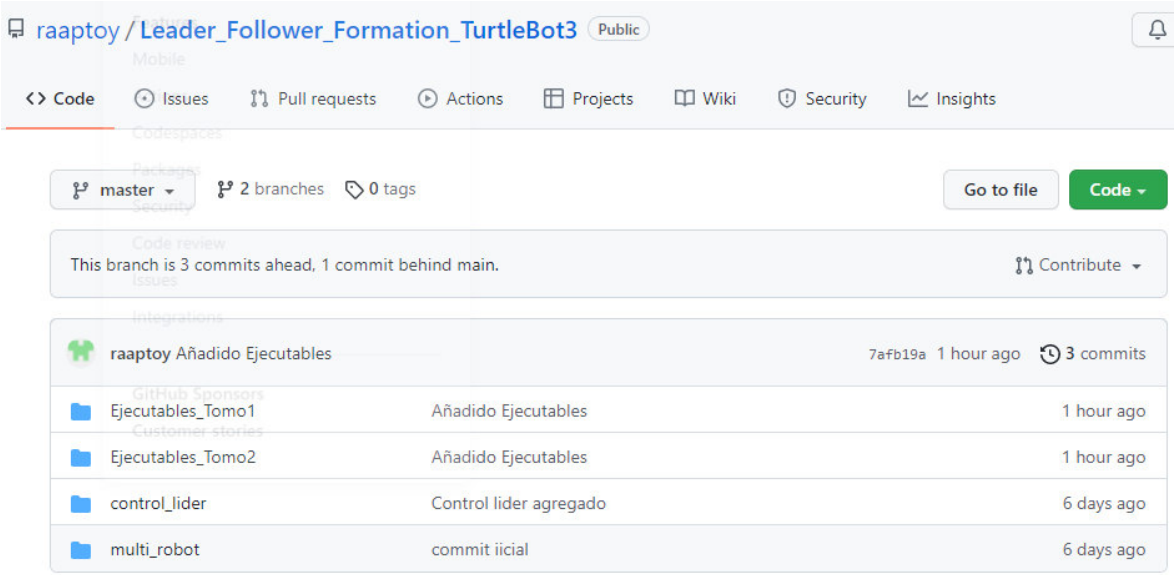


Figura I.1. Imagen general del repositorio desarrollado

En este repositorio se encuentra los paquetes desarrollados para el primer y segundo tomo, así también como los ejecutables.

Primer Tomo:

- Paquete: control_lider
- Ejecutables: Ejecutables_Tomo1

Segundo Tomo:

- Paquete: multi_robot
- Ejecutables: Ejecutables_Tomo2

Los paquetes almacenan todas las carpetas necesarias para la ejecución del proyecto, tal como se observa en la Figura I.2. En un inicio se tienen las mismas carpetas para los dos paquetes. Se tienen: Data, launch, models, scripts, worlds, CMakeLists.txt, package.xml.

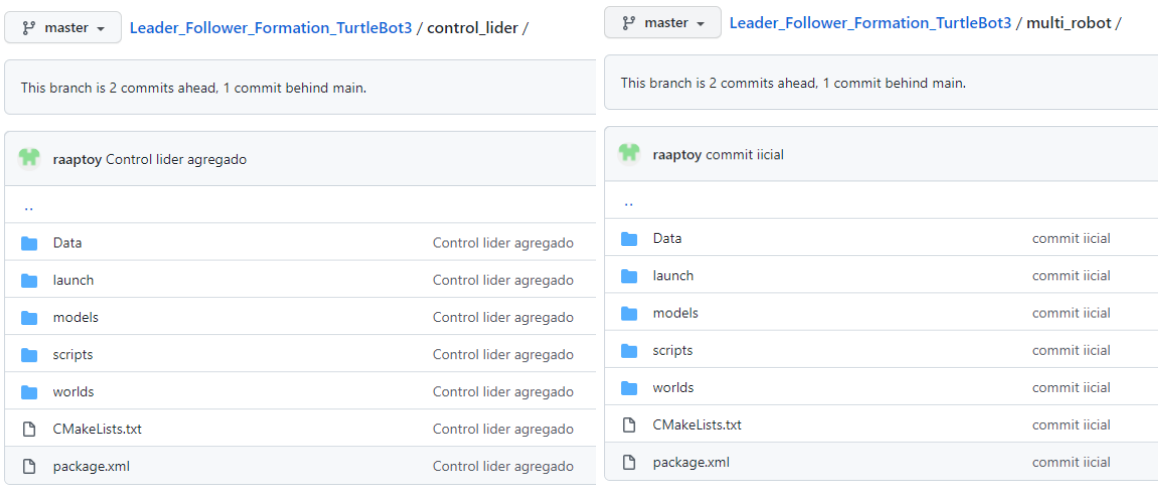


Figura I.2. Carpetas internas de los paquetes del tomo 1 y tomo2

En el repositorio se puede entrar a cada una de las carpetas para ver que contienen por ejemplo en las carpetas Data, de cada uno de los tomos almacenan archivos .csv utilizados para plotear las gráficas de trayectoria, de ángulo y distancia. La carpeta scripts almacenan los códigos generados para cada tomo, como se puede observar en la Figura I.3:

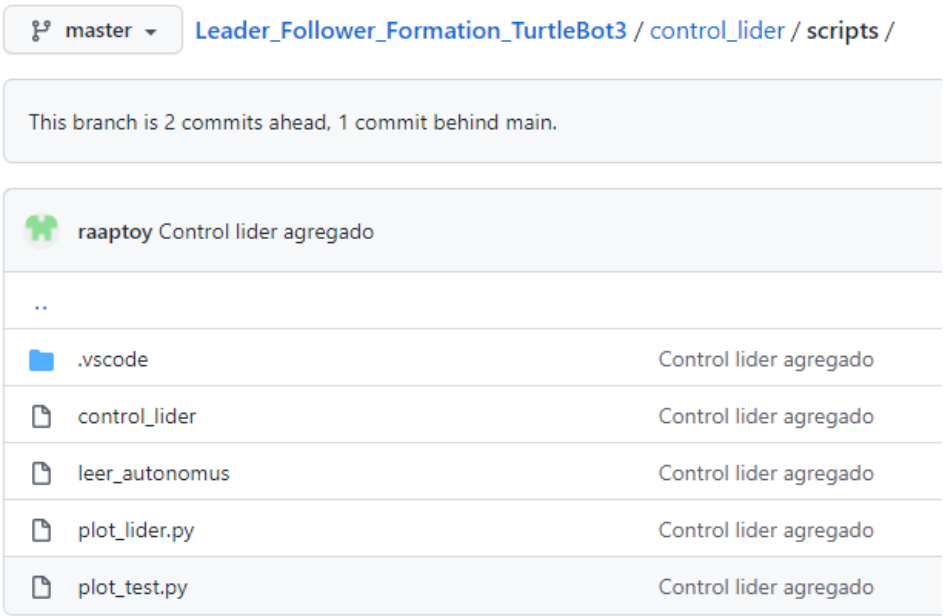


Figura I.3. Scripts desarrollados para el tomo1 en el paquete control_lider

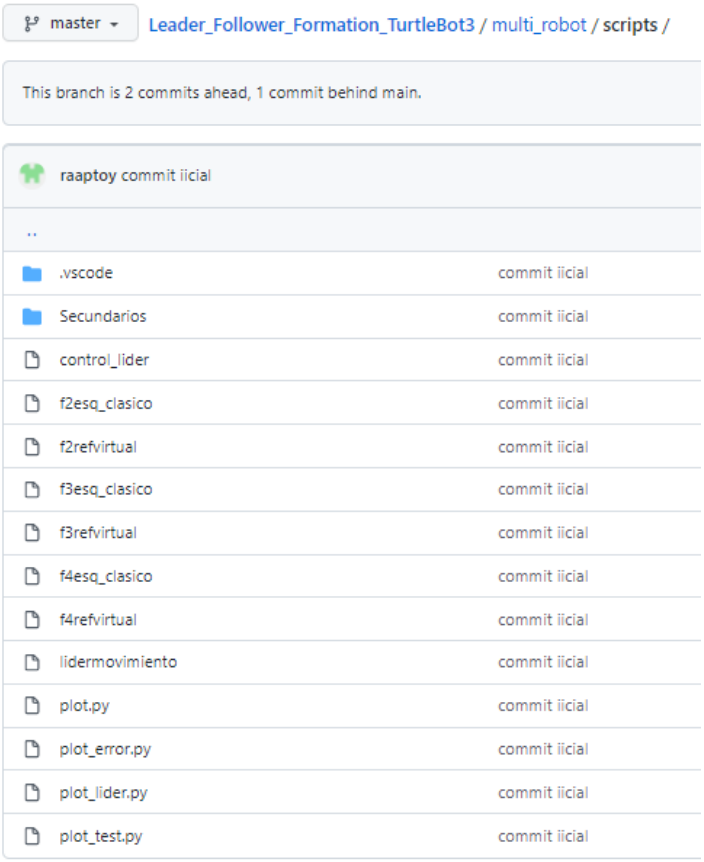


Figura I.4. Scripts desarrollados para el tomo1 en el paquete multi_robot

Finalmente, en la carpeta de ejecutables, se encuentran todos los archivos bash (.sh) que ayudan a ejecutar los proyectos, se los puede observar en las Figuras I.4 y Figura I.5.

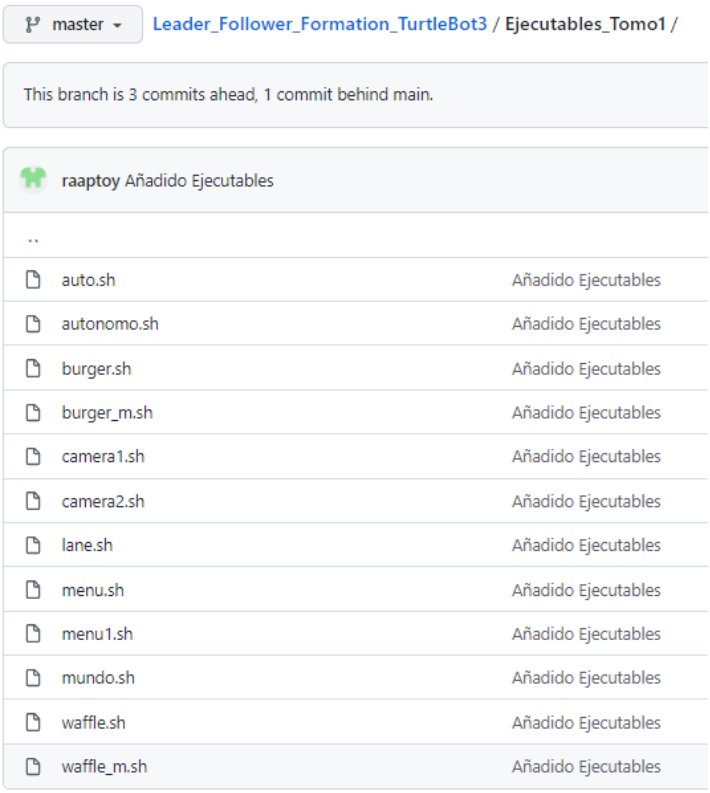


Figura I.5. Archivos ejecutables desarrollados para el tomo 1

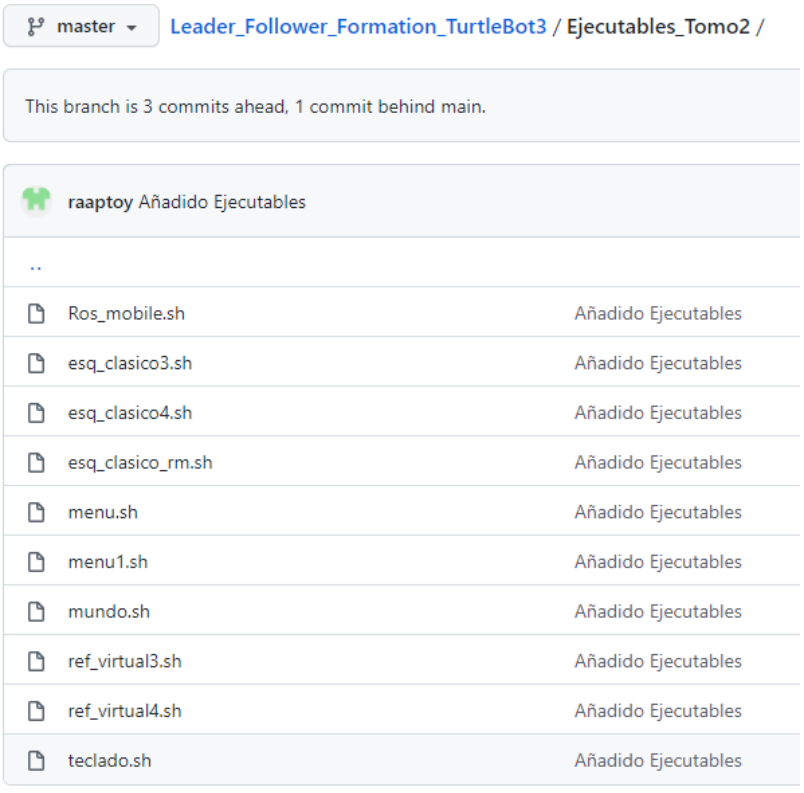


Figura I.6. Archivos ejecutables desarrollados para el tomo 2

ANEXO II. Manual de usuario:

El anexo presentado en esta sección se puede encontrar de igual manera en el primer tomo del trabajo desarrollado en conjunto.

El manual de usuario presentado a continuación servirá de ayuda para poder ejecutar el trabajo realizado en el primer y segundo tomo del trabajo desarrollado en conjunto.

Requisitos del sistema

- Sistema operativo: Linux Ubuntu 20.04 LTS
- ROS: Noetic Ninjemys
- Gazebo: V11.0.0

Paquete desarrollado en ROS

Con todos los requisitos del sistema, para poder ejecutar el programa desarrollado en su computador se recomienda clonar el paquete de ROS del repositorio:

repositorio: GitHub - raaptoy/Leader_Follower_Formation_TurtleBot3 at master

En el repositorio se encontrará con la carpeta multi_robot, esta deberá copiarse en catkin workspace, con lo que deberá tener el siguiente path:

/home/USER/catkin_ws/src/multi_robot

Se entiende por **USER** al nombre del usuario que se ha puesto en el sistema operativo Linux.

Una vez copiado el paquete, es necesario ejecutar los siguientes mandos, dentro de la terminal:

```
roscd
catkin make
```

Ejecución

Para asegurar que todos los datos se guarden es necesario modificar el usuario o USER del directorio, dentro de los scripts de Python. Así también como los paths dentro de los archivos de la carpeta bash. Como, por ejemplo, en un archivo bash el dato que se debe cambiar se lo subraya en color amarillo y el user está subrayado en color rojo, esto se lo puede observar en la Figura II.1. y en la Figura II.2

master Leader_Follower_Formation_TurtleBot3 / Ejecutables_Tomo2 / ref_virtual4.sh

raaptoy Añadido Ejecutables

1 contributor

16 lines (15 sloc) | 699 Bytes

```
#!/bin/sh
roscd
source ./devel/setup.bash
export TURTLEBOT3_MODEL=burger
#Lanzo Robots Diamante
roslaunch multi_robot followers4.launch
#Lanzo el control del robot 1 (Teleoperación)
gnome-terminal --tab --title="Control Lider" --command="rosrun multi_robot lidermovimiento 'cd /etc; ls; $SHELL'"
#Follower2
gnome-terminal --tab --title="Seguidor 2" --command="rosrun multi_robot f2refvirtual  'cd /etc; ls; $SHELL'"
#Follower3
gnome-terminal --tab --title="Seguidor 3" --command="rosrun multi_robot f3refvirtual  'cd /etc; ls; $SHELL'"
#Follower4
gnome-terminal --tab --title="Seguidor 4" --command="rosrun multi_robot f4refvirtual  'cd /etc; ls; $SHELL'"
cd /home/rodrigo/Escritorio/Ejecutables

```

Figura II.1. Archivo bash que contiene directorio a cambiar

master Leader_Follower_Formation_TurtleBot3 / multi_robot / scripts / f2esq_clasico

raaptoy commit iicial

1 contributor

Executable File | 199 lines (162 sloc) | 7.45 KB

```
#!/usr/bin/env python3
import time
import sys
import rospy
from geometry_msgs.msg import Twist
from nav_msgs.msg import Odometry
from threading import Thread

# Other Imports
import numpy as np
import math as m
from tf.transformations import euler_from_quaternion, quaternion_from_euler

max_linear_velocity = 0.2
max_angular_velocity = 2.8

LOG_FILE_DIR = '/home/rodrigo/catkin_ws/src/multi_robot/Data'
```

Figura II.2. Archivo bash que contiene directorio a cambiar

Una vez se hayan hecho todos los cambios realizados. En la carpeta Ejecutables del tomo 1 o tomo 2 dar click derecho, abrir en una terminal. Seguidamente se seguirán los siguientes pasos

Tomo 1:

Al abrir la carpeta Ejecutables_Tomo1, en la terminal generada dar el comando

```
bash menu.sh
```

Se podrá visualizar el menú principal como se ve en la Figura II.3

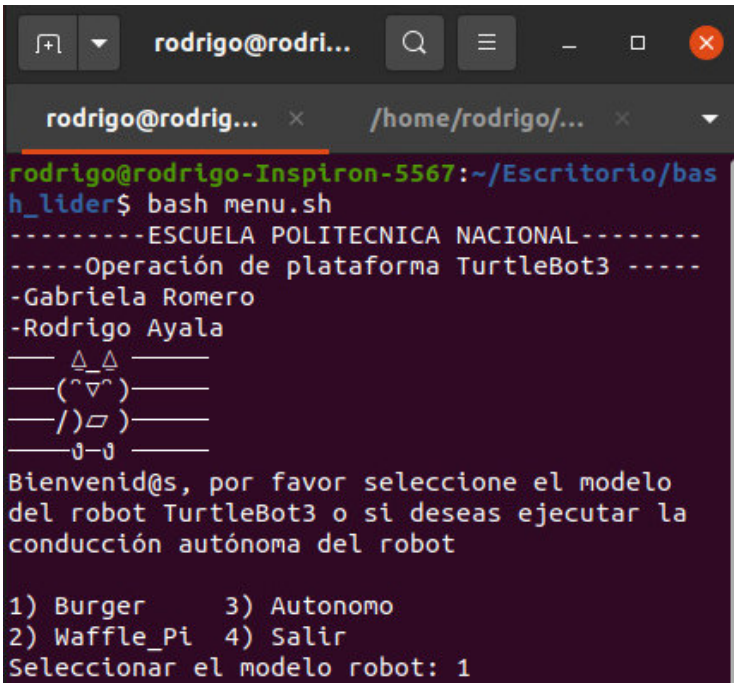


Figura II.3. Menú principal del tomo 1

Al seleccionar la opción Burger o Waffle_Pi se despliega el menú secundario mostrado en la Figura II.4, de aquí al escoger la opción Teclado se ejecutará el script del control de movimiento del robot a través del teclado. Si por el contrario se escoge ROS Mobile, se imprime un mensaje que indica al usuario que se conecte mediante la aplicación móvil.

En caso de seleccionar la opción de Autonomo, se ejecuta todos los nodos necesarios para visualizar la conducción autónoma del robot.

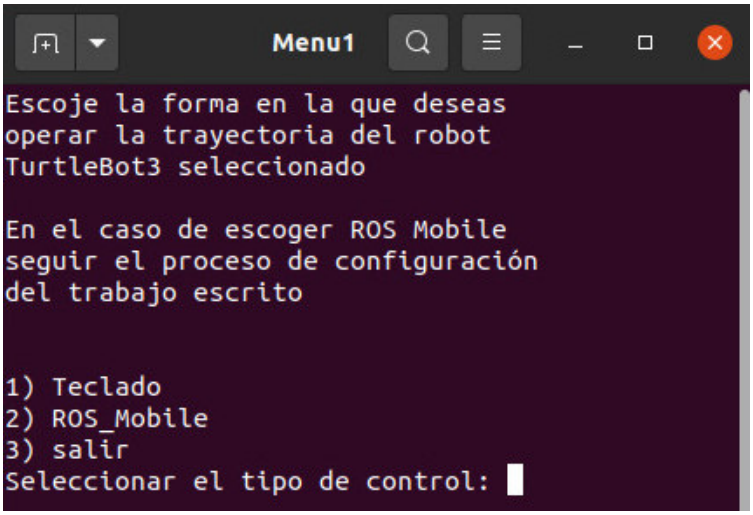


Figura II.4. Menú secundario del tomo 1

Descripción de terminales

- **Terminal 1:** ejecución de menú principal donde se podrá escoger el modelo del robot, véase en la Figura II.3.
- **Terminal 2:** despliegue de Roscore junto con Gazebo y el modelo del robot como se muestra en la Figura II.5.
- **Terminal 3:** despliegue de menú secundario (véase en la Figura II.4), en este se permite seleccionar el modo de operación del robot por teclado o por la aplicación ROS Mobile. Si se escoge la opción de teclado, en esta misma terminal se ejecutará el script necesario. En el caso de ROS Mobile, se mostrará un mensaje indicando que se conecte con la aplicación.

Si en el menú principal se selecciona Autónomo, se tienen de igual forma a las terminales 1 y 2 y luego se despliega lo siguiente:

- **Terminal 3:** despliegue del archivo launch que contiene la configuración de: camera 1 intrínseca. Ver la Figura II.7(a)
- **Terminal 4:** despliegue del archivo launch que contiene la configuración de camera 2 extrínseca. Ver la Figura II.7(b)
- **Terminal 5:** despliegue del archivo launch que contiene el script llamado Lane. Ver la Figura II.7(c)
- **Terminal 6:** despliegue del archivo launch que contiene el script llamado control_lane. Ver la Figura II.7(d)

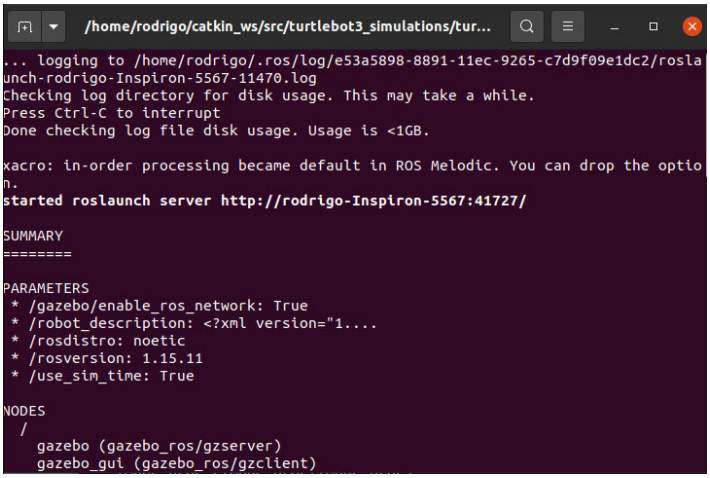


Figura II.6. Despliegue del robot en el mundo de Gazebo en el tomo 1

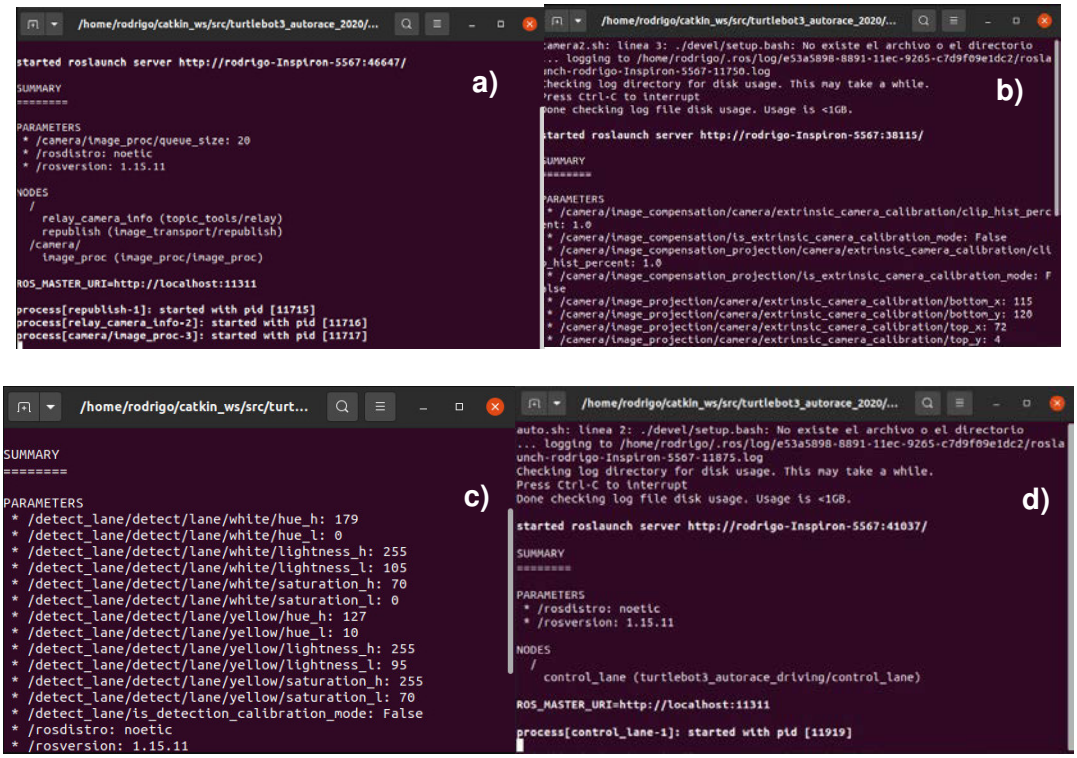


Figura II.7. Terminales generadas con la opción Autónomo en el tomo 1

En la Figura II.8 se presenta el resultado de la simulación del robot TurtleBot3 modelo Waffle pi con operación por teclado y en la Figura II.9, se aprecia la simulación del modelo Burger al seleccionar el modo de operación Autonomo en el menú principal.

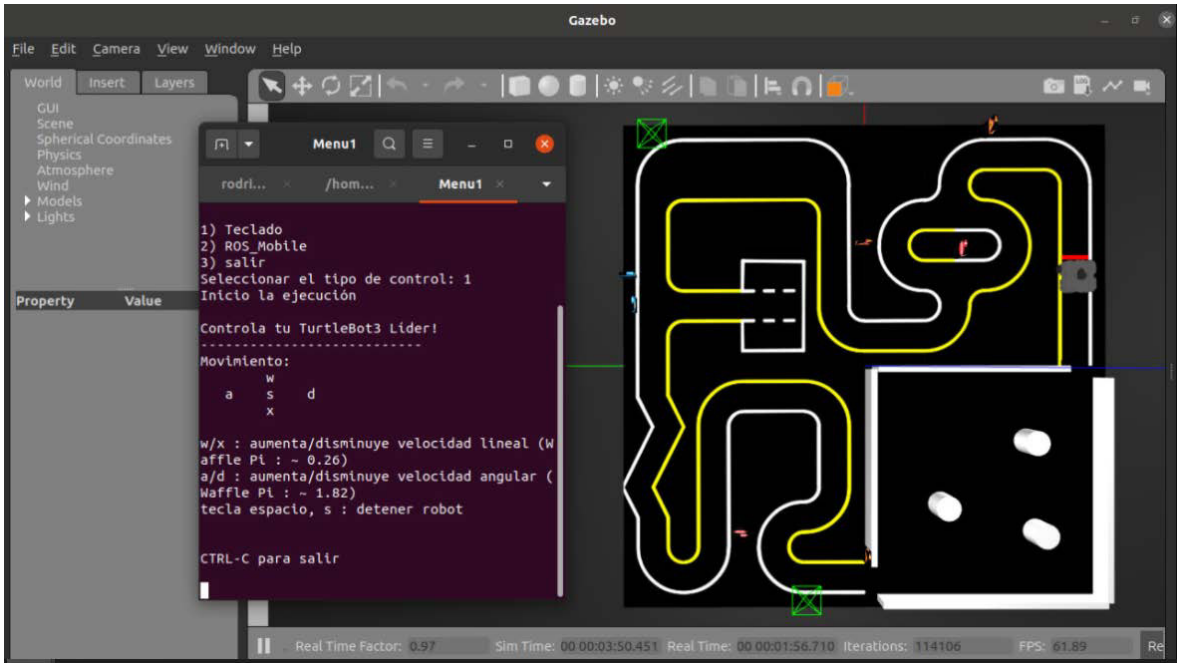


Figura II.8. Resultado de simulación con operación por teclado del robot TurtleBot3 modelo Waffle pi

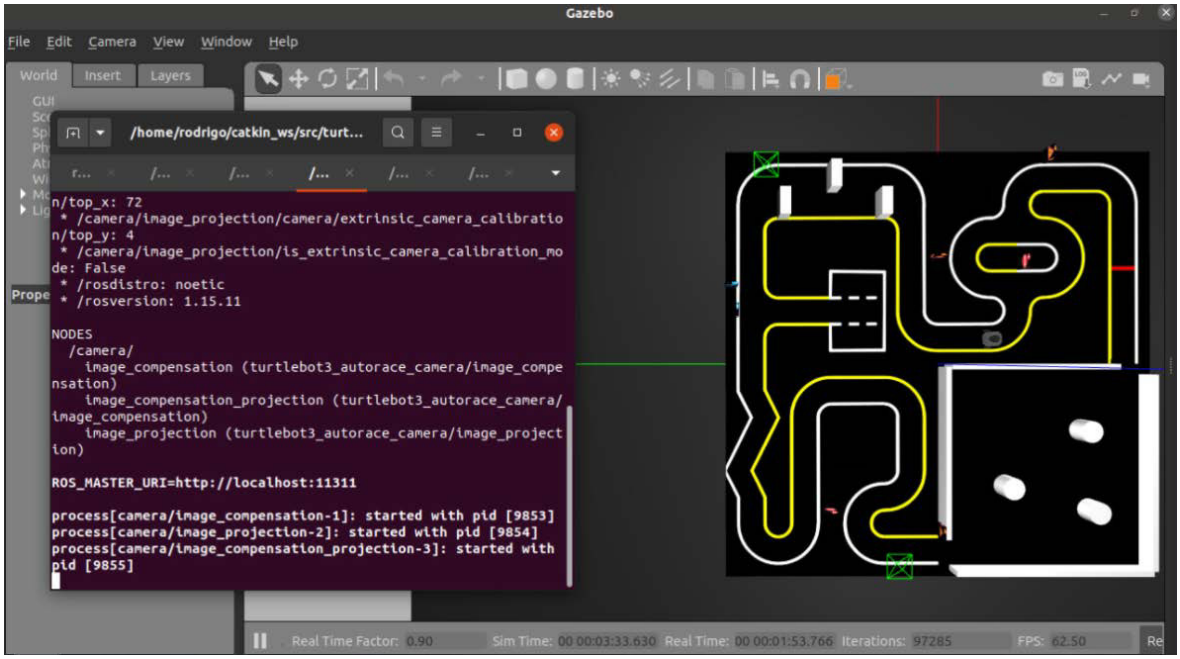


Figura II.9. Resultado de simulación de opción Autónomo del robot TurtleBot3 modelo Burger

Tomo 2

En la terminal generada dar el comando

```
bash menu.sh
```

Aparecerá entonces una gráfica del menú principal del tomo 2 mostrado en la Figura II.10.

Figura II.10. Menú principal del tomo 2

Al ejecutar la opción Teclado se ejecutará el entorno de Gazebo con el robot líder TurtleBot3 Waffle pi y seguidamente se abrirá otra terminal mostrando el menú secundario, se lo podrá realizar en la Figura II.11.

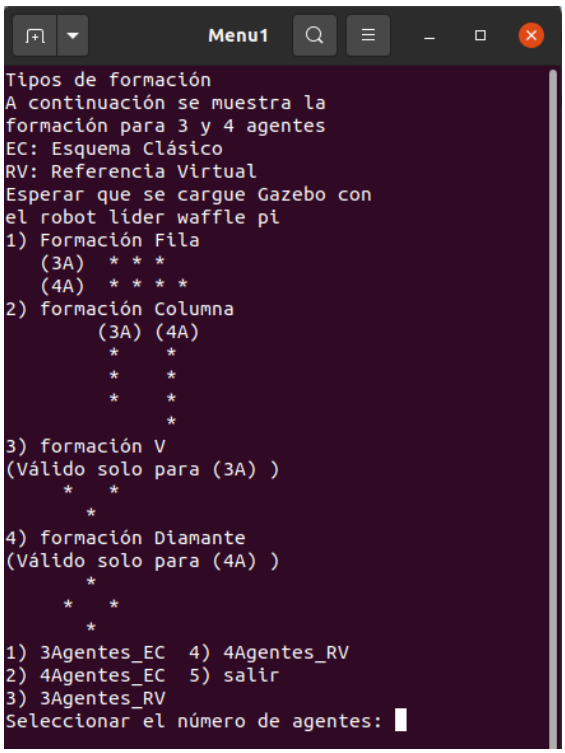


Figura II.11. Menú secundario del tomo 2

Luego se podrá elegir cualquier opción mostrada en el menú secundario y se desplegarán las demás terminales. A continuación, se presenta una breve descripción de las terminales que se generan:

- **Terminal 1:** ejecución de menú principal donde se podrá escoger el movimiento del robot líder, véase en la Figura II.10.
- **Terminal 2:** despliegue de Roscore junto con Gazebo y el modelo del robot líder como se muestra en la Figura II.12(a).
- **Terminal 3:** despliegue de menú secundario (véase en la Figura II.11), en este se permite seleccionar el tipo de formación, número de agentes y tipo de algoritmo. También se lanza el modelo de los robots seguidores para que aparezcan en Gazebo como se lo aprecia en la Figura II.12(b).
- **Terminal 4:** en caso de seleccionar el movimiento del robot líder por teclado SE ejecuta el scrip que realiza el movimiento del robot líder a través del teclado, tal como se observa en la Figura II.12(c). En caso de seleccionar el movimiento por ROS_Mobile, se omite esta terminal.
- **Terminal 5,6 y/o 7:** en el caso de escoger esquema clásico, se muestra información de distancia en [m] y ángulo deseado en [rad] y los valores reales del seguidor junto con el tipo de formación que se está realizando en ese momento. Para el algoritmo de referencia virtual se aumenta la información de distancia y ángulo de la referencia como se observa en la Figura II.12(d).

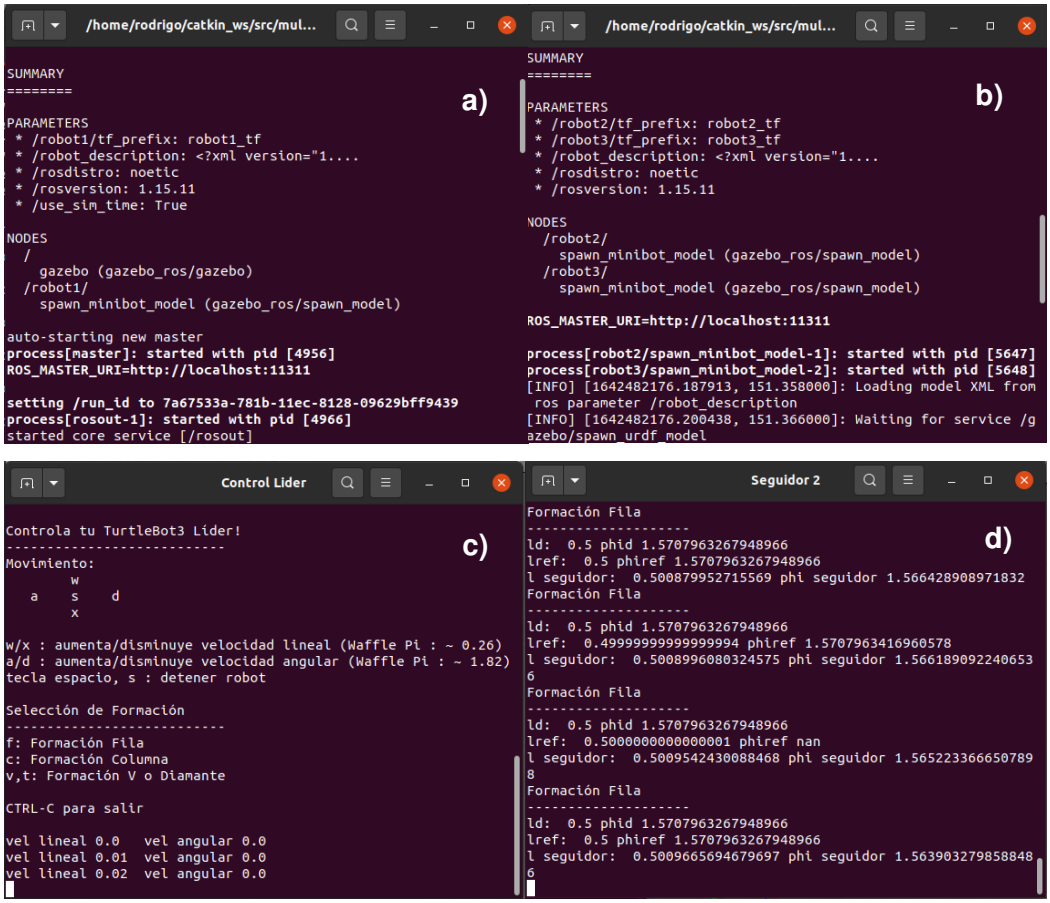


Figura II.12. Despliegue de terminales del tomo 2

Para terminar la simulación del proyecto se deberá cerrar las terminales generadas escribiendo en cada una de ellas Ctrl+C.

En la Figura II.13 y Figura II.14, se pueden observar los resultados obtenidos de la simulación en el software Gazebo junto con las terminales.

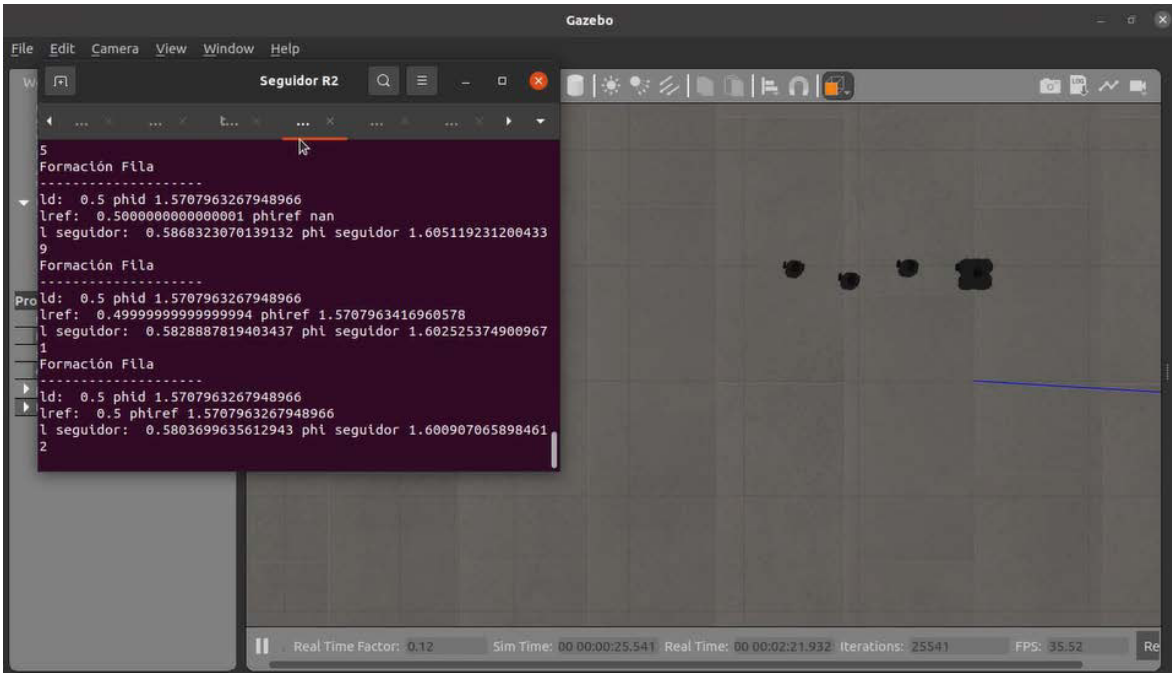


Figura II.13. Resultado de simulación en formación fila con muestra de terminal del seguidor R2

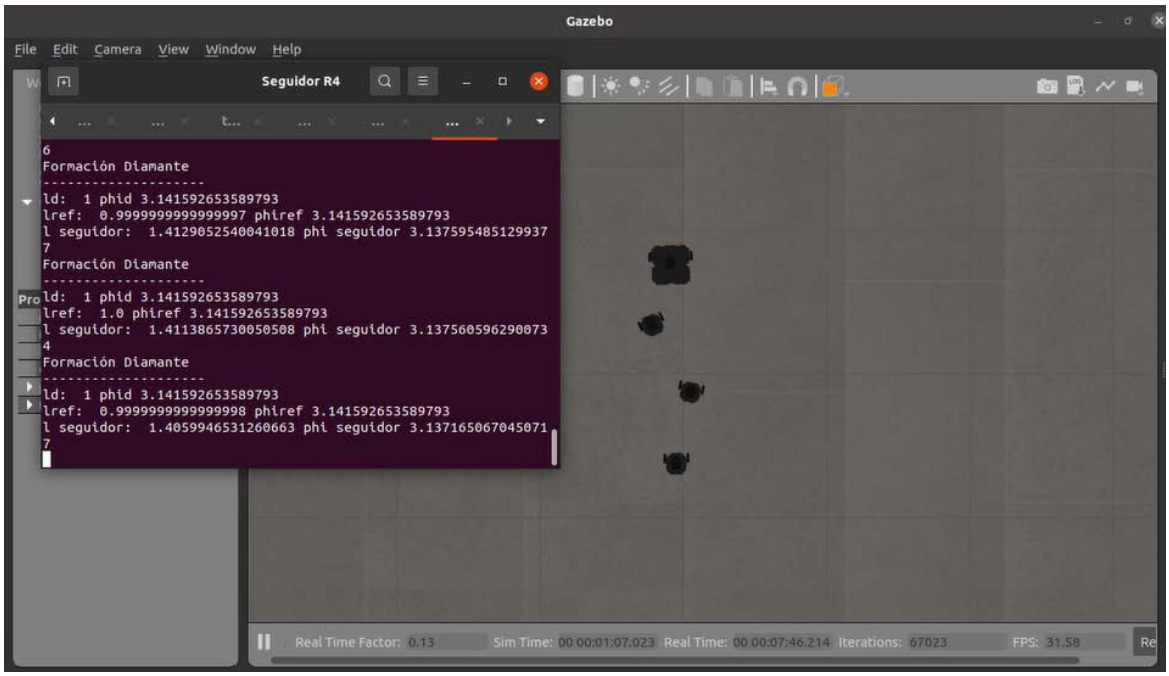


Figura II.14. Resultado de simulación en formación columna con muestra de terminal del seguidor R4

ANEXO III. Link a videos demostrativos:

El anexo presentado en esta sección se puede encontrar de igual manera en el primer tomo del trabajo desarrollado en conjunto.

Los videos principales que confieren a los resultados obtenidos del primer y segundo tomo se encuentran en el siguiente link:

Link: Leader-Follower Multi-Agent System - YouTube

En un inicio se tienen los videos de los resultados del tomo 1, con la descripción:

- Video 1: comparación del control de movimiento por teclado y ROS Mobile del robot TurtleBot3-Burger
- Video 2: comparación del control de movimiento por teclado y ROS Mobile del robot TurtleBot3-Waffle pi
- Video 3: ejecución de la conducción autónoma del paquete Autorace

Para el segundo tomo se tiene lo siguiente:

- Video 4: comparación de algoritmos con referencia virtual y esquema clásico para la realización de la formación en fila
- Video 5: comparación de algoritmos con referencia virtual y esquema clásico para la realización de la formación en columna
- Video 6: comparación de algoritmos con referencia virtual y esquema clásico para la realización de la formación en V
- Video 7: comparación de algoritmos con referencia virtual y esquema clásico para la realización de la formación en diamante
- Video 8: comparación de algoritmos con referencia virtual y esquema clásico para la realización de diferentes cambios de formación