Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California

Maestría en Ciencias
Electrónica y Telecomunicaciones
con orientación en Instrumentación y Control

Generación de trayectorias caóticas en pequeños robots móviles

Tesis

para cubrir parcialmente los requisitos necesarios para obtener el grado de

Maestro en Ciencias

Presenta:

Eduardo López Barrón

Ensenada, Baja California, México

2022

Tesis defendida por

Eduardo López Barrón

Y aprobada por el siguiente comité

Dr. César Cruz Hernández
Codirector de tesis

M. en C. Juan José Cetina Denis
Codirector de tesis

Dr. Javier Pliego Jiménez

Dr. Rafael de Jesús Kelly Martínez

Dr. Víctor Ruiz Cortés

Dra. María del Carmen Maya Sánchez
Coordinadora del Posgrado en Electrónica y Telecomunicaciones

Dr. Pedro Negrete Regagnon
Director de Estudios de Posgrado

Resumen de la tesis que presenta **Eduardo López Barrón** como requisito parcial para la obtención del grado de Maestro en Ciencias en Electrónica y Telecomunicaciones con orientación en Instrumentación y Control.

Generación de trayectorias caóticas en pequeños robots móviles

Resumen aprobado por:

Dr. César Cruz Hernández	**M. en C. Juan José Cetina Denis**
Codirector de tesis	Codirector de tesis

En el presente trabajo de tesis, se aborda el problema de generación de trayectorias caóticas, específicamente en un robot móvil con tracción diferencial. El esquema de control empleado en este trabajo de tesis, permite sincronizar los estados del sistema caótico MACM con las velocidades lineales y angulares de un robot móvil diferencial para inducir dinámicas caóticas. Se reportan resultados numéricos, así como experimentales utilizando un modelo virtual de un robot diferencial. Se demuestra de manera numérica la existencia de caos en las trayectorias generadas. Por último, se presenta una posible aplicación de las trayectorias caóticas generadas a la detección y búsqueda rápida de objetos en un área de trabajo.

Palabras clave: Generación de trayectorias, Caos, Sistema caótico MACM, Robots móviles diferenciales.

Abstract of the thesis presented by **Eduardo López Barrón** as a partial requirement to obtain the Master of Science degree in Electronics and Telecommunications with orientation in Instrumentation and Control.

Generation of chaotic trajectories in small mobile robots

Abstract approved by:

Dr. César Cruz Hernández	**M. en C. Juan José Cetina Denis**
Codirector de tesis	Codirector de tesis

In this work, the problem of generating chaotic trajectories in a mobile robot with differential drive is addressed. The control scheme used in this work synchronizes the states of the MACM chaotic system with the linear and angular velocities of the mobile robot to induce chaotic dynamics. Numerical results are reported as well as experimental results using a virtual model of a differential drive robot. The existence of chaos in the generated trajectories is numerically demonstrated. Finally, a possible application of the generated chaotic trajectories to fast detection and search of objects in a work area is presented.

Keywords: Trajectory generation, Chaos, Chaotic system MACM, Differential mobile robots.

Dedicatoria

A mis padres Jorge y Cristina, gracias por apoyarme y confiar siempre en mí.

A mis hermanos Adán y Sandra, gracias por todo.

A mis abuelos, especialmente a mi abuelo Alberto.

Finalmente, a Luz María mi gran amor y compañera de vida.

Agradecimientos

A mi familia por apoyarme siempre en la medida de lo posible.

Al Centro de Investigación Científica y de Educación Superior de Ensenada por darme la oportunidad de realizar mis estudios de posgrado.

Al Consejo Nacional de Ciencia y Tecnología (CONACyT) por brindarme el apoyo económico necesario para llevar a cabo mis estudios.

Al Consejo Nacional de Ciencia y Tecnología (CONACyT) a través del proyecto de investigación en Ciencia Básica entre instituciones "*Sincronización de sistemas complejos y algunas aplicaciones",* Ref. A1-S-31628 (2020-2022).

A mis directores de tesis, el Dr. César Cruz Hernández y el M. en C. Juan José Cetina Denis, por su gran apoyo, enseñanzas e invaluables consejos durante la realización de esta tesis.

A los integrantes mi comité de tesis, al Dr. Javier Pliego Jiménez por su constante apoyo desde el momento que llegue a CICESE como practicante, al Dr. Víctor Ruiz Cortes y Dr. Rafael de Jesús Kelly Martínez por sus acertadas correcciones y su orientación a lo largo de todo este proceso.

A todos mis profesores del Centro de Investigación Científica y Educación Superior de Ensenada (CICESE), piezas clave en mi formación académica.

A todos mis compañeros, por hacer más amena mi estancia en la maestría.

Tabla de contenido

Lista de figuras

Lista de tablas

Capítulo 1. Introducción

En este capítulo se da un primer acercamiento al caos y los robots móviles.

Al contrario del imaginario popular, el caos no es azar, desorden o algo de naturaleza incontrolable, este puede definirse de manera matemática y puede ser aprovechado de distintas maneras. El caos tiene su origen en la teoría de los sistemas dinámicos.

Los sistemas dinámicos deben cumplir una serie de condiciones para ser considerados caóticos. En primer lugar, dichos sistemas deben ser termodinámicamente abiertos, es decir, que describan una situación en la que se producen intercambios de materia y energía entre el sistema y su entorno. En segundo lugar, el sistema debe estar en desequilibrio termodinámico (diferencias de temperatura). Por último, el sistema debe contener al menos tres dimensiones, para el caso de sistemas continuos. Otra de las características más representativas de un sistema caótico es la sensibilidad a las condiciones iniciales, esto significa que el mismo sistema tendrá evoluciones temporales sumamente diferentes, a pesar de que sus condiciones iniciales sean ligeramente distintas (Hilborn C. Robert, 2005).

Actualmente los sistemas caóticos han sido utilizados ampliamente en la ingeniería para la resolución de diferentes problemas gracias a las características tan únicas de estos sistemas dinámicos. La sensibilidad las condiciones iniciales por ejemplo, es sumamente útil al momento de codificar información y transmitirla de forma segura por canales públicos (Cruz-Hernández y Romero-Haros, 2008). Las geómetras fractales, propias de sistemas caóticos son utilizadas en el diseño de antenas de alta eficiencia (Figueroa-Torres *et al.,* 2013).

En cuanto a los robots móviles, existen diversos tipos: terrestres, aéreos, acuáticos, etc. Dos de los más estudiados y utilizados en investigación científica son el Khepera III y Elisa. El Khepera III es la tercera versión de un robot móvil diferencial terrestre pequeño que cuenta con dos ruedas y un soporte deslizante, múltiples sensores infrarrojos y ultrasónicos, y con los medios necesarios para comunicarse inalámbricamente. El cuadricóptero es un helicóptero multirotor que es levantado y propulsado por cuatro rotores; cuenta con los controladores necesarios para mantenerse en vuelo estable y comunicación inalámbrica. Al ser un robot móvil aéreo es muy deseable su aplicación en misiones de búsqueda y patrullaje. Un conjunto de varios de estos robots se encuentra disponibles en el Laboratorio de Sincronización y Sistemas Complejos del Departamento de Electrónica y Telecomunicaciones.

1.1. Antecedentes

La generación de trayectorias es un tema importante cuando se habla de robótica móvil, ya que el éxito de la tarea asignada al robot móvil depende en gran medida del correcto diseño de la trayectoria. En años recientes, la comunidad científica ha aprovechado las características de los sistemas caóticos para generar trayectorias en robot móviles con múltiples aplicaciones.

Cetina (2017) propone un método sencillo y aplicable a diferentes tipos de robots móviles en el que se inducen dinámicas caóticas a través de las entradas de control del robot. Las dinámicas caóticas pueden ser inducidas por cualquier sistema caótico con una dimensión igual o mayor al número de entradas del robot.

Li *et al.* (2017) para generar trayectorias caóticas con aplicaciones especiales utilizan una versión bidimensional del mapa de Chebyshev, el cual utilizando transformaciones mapea una serie de puntos a una región factible del área de trabajo designada, el robot después sigue estos puntos en orden generando así la trayectoria.

Petavratzis *et al.* (2019) toma un enfoque diferente en la generación de trayectorias caóticas, utilizando una versión modificada del mapa logístico para generar un generador caótico pseudo aleatorio de bits. La secuencia de bits generada es utilizada para producir comandos de movimiento en un robot en cuatro u ocho direcciones.

Majeed (2020), buscando generar una trayectoria para cubrir un área delimitada de la manera más eficiente posible utilizando un robot móvil diferencial, propone un método para generar trayectorias caóticas en el que se integran las ecuaciones de movimiento del robot y el mapa logístico.

1.2. Justificación

En este proyecto de tesis se pretende generar trayectorias no predecibles, en este caso caóticas, en un robot móvil diferencial de manera numérica y experimental, particularmente utilizando el sistema caótico MACM (Méndez-Ramírez *et al.,* 2017) para influir directamente en el comportamiento de las velocidades del robot. A diferencia de otros trabajos, en el algoritmo de control utilizado en esta tesis, no se hace uso directo del modelo matemático del robot móvil, sino que las trayectorias caóticas son

producto de la influencia directa en cada una de las entradas al sistema; en el caso del robot móvil diferencial estas entradas son las velocidades de los motores. Las ventajas de esta metodología es su sencillez y la facilidad que tiene de ser exportado hacia otros tipos de robots móviles. Necesitando solamente un sistema caótico con un numero de estados igual o mayor al número de entradas del robot a controlar, ver (Cetina Denis *et al.,* 2018).

1.3. Hipótesis

Es posible generar trayectorias no predecibles con base en dinámicas caóticas, para pequeños robots móviles para aplicaciones prácticas de detección y búsqueda rápida de objetos, patrullaje, etc.

1.4. Objetivos

1.4.1. Objetivo General

Generar trayectorias con dinámicas no predecibles con base en caos para robots móviles y dar aplicaciones en exploración eficiente de terrenos inhóspitos, detección rápida de objetos peligrosos o patrullaje.

1.4.2. Objetivos Particulares

- Seleccionar el sistema caótico apropiado para generar trayectorias impredecibles en robots móviles.
- Realizar pruebas numéricas en la sincronización de un sistema caótico y un modelo de robot móvil diferencial.
- Generar trayectorias basadas en dinámicas caóticas para robots móviles diferenciales.
- Realizar aplicaciones en la exploración eficiente de terrenos inhóspitos, detección rápida de objetos peligros o patrullaje.

1.5. Organización de la tesis

Esta tesis está organizada de la siguiente manera.

En el capítulo 2 se da una visión general de los sistemas caóticos, sus características y sus aplicaciones. Se presenta el sistema caótico MACM que será utilizado para inducir dinámicas caóticas en un robot móvil diferencial.

En el capítulo 3 se da una explicación general de la robótica móvil, los distintos tipos de robots móviles y sus aplicaciones en la vida cotidiana. Se comentan, a detalle, los robots móviles terrestres y sus distintos tipos de locomoción; además, se presenta el robot móvil Khepera III y su modelo cinemático.

Posteriormente en el capítulo 4 se explica de manera detallada el algoritmo generador de trayectorias caóticas que, en conjunto con el sistema MACM y el modelo cinemático del robot, se utilizan para las simulaciones numéricas. Se presentan diversas maneras para generar trayectorias diferentes manteniendo condiciones iniciales y se realiza la validación de las mismas en software de simulación especializado.

En el capítulo 5 se realizan pruebas para comprobar la existencia de caos en las trayectorias obtenidas de manera numérica y mediante el modelo virtual de un robot diferencial, además de mostrar una posible aplicación.

Por último, en el capítulo 6 se presentan las conclusiones y recomendaciones para futuros trabajos enfocados en la generación de trayectorias.

Capítulo 2. Caos y sus aplicaciones

Este capítulo incluye una explicación sobre los sistemas caóticos, sus características y las distintas aplicaciones en las diferentes ramas de la ciencia y de ingeniería.

El caos en matemáticas tiene su base en la teoría de los sistemas dinámicos. Podemos definir un sistema dinámico como un sistema el cual evoluciona en el tiempo. Para que un sistema dinámico sea considerado caótico debe poseer ciertas características o condiciones.

2.1. Características de los sistemas caóticos

Dentro de las muchas características de los sistemas caóticos podemos mencionar como las más representativas las siguientes:

- **Sensibilidad a condiciones iniciales:** Una diferencia, por más mínima que sea en las condiciones iniciales del sistema genera evoluciones temporales exponencialmente diferentes a medida que pasa el tiempo. Esta característica de los sistemas caóticos es la base de su uso en los campos de encriptado y comunicaciones seguras. En la Figura ***1*** se muestra esta propiedad al observar cómo evoluciona en el tiempo un estado del mapa de Hénon con condiciones iniciales muy similares.

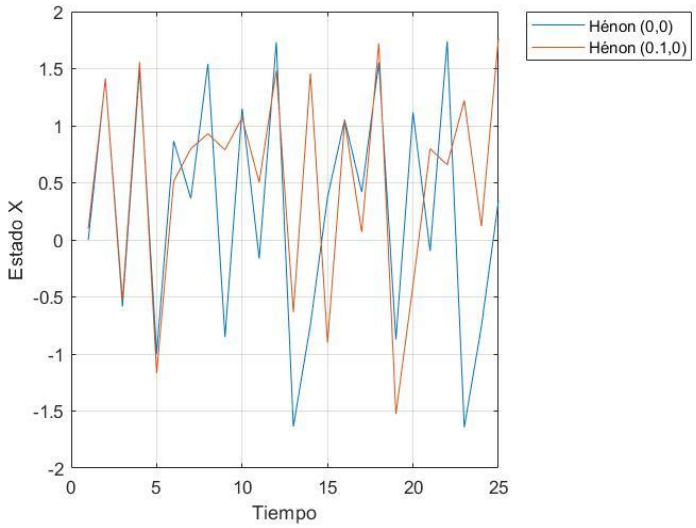
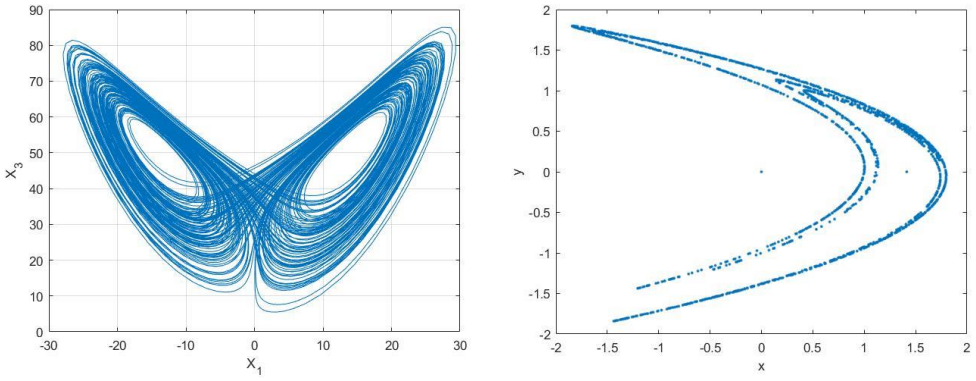


Figura 1. Evolución temporal del primer estado del mapa caótico de Hénon con condiciones iniciales ligeramente diferentes.

▪ **Generan un atractor extraño:** Las secuencias de valores producidas por un sistema caótico dibujan, en el plano de fase, un atractor extraño. Los atractores extraños son regiones en el espacio de fase a las que tienden las dinámicas del sistema, poseen una estructura geométrica inusual alejada de los objetos geométricos clásicos y con características fractales. En la Figura ***2*** se muestran los atractores extraños producidos por el sistema caótico de Lorenz en tiempo continuo (Figura ***2***a) y el atractor extraño producido por el sistema caótico de Hénon en tiempo discreto (Figura ***2***b).



(a) Atractor de Lorenz (b) Atractor caótico del mapa de Hénon

Figura 2. Atractores extraños generados por los sistemas caóticos de Lorenz (a) y el mapa de Hénon (b)

- **Deterministas:** Las dinámicas de los sistemas caóticos no son causa del azar, es posible predecir de manera perfecta estas dinámicas conociendo las condiciones iniciales y los valores en sus parámetros.

- **No linealidad:** Los sistemas caóticos son sistemas dinámicos que no cumplen con el principio de homogeneidad y superposición.

- **Exponente de Lyapunov positivo:** Los exponentes de Lyapunov son utilizados para medir el grado en el que convergen o divergen dos soluciones infinitesimalmente cercanas en el espacio de fase. Un exponente de Lyapunov negativo es indicador de una convergencia en dichas trayectorias, en caso de contar con un exponente de Lyapunov positivo las trayectorias divergen, evidenciando de esta manera una sensibilidad a las condiciones

iniciales, característica clave de los sistemas caóticos. Mientras que un exponente de Lyapunov igual a cero es resultado de una dirección neutra de las trayectorias.

- **Valores acotados:** los sistemas caóticos generan secuencias de valores acotadas, es decir permanecen en un intervalo de dos valores.

2.2. Sistema caótico MACM

El sistema caótico MACM documentado por primera vez por Méndez-Ramírez *et al.* (2017) es un sistema dinámico tridimensional derivado de las ecuaciones del sistema de Lorenz (Lorenz, 1963). El sistema caótico MACM esta descrito por las siguientes ecuaciones diferenciales no lineales.

$$\begin{aligned} \dot{x} &= -ax - byz, \\ \dot{y} &= -x + cy, \\ \dot{z} &= d - y^2 - z. \end{aligned} \tag{1}$$

El sistema está compuesto por 7 términos y 2 no linealidades. Se reporta en Méndez-Ramírez *et al.* (2017) que el sistema exhibe un comportamiento completamente caótico para los valores $a = 2, b = 2, c = 0.5, d = 4$; además, el sistema cuenta con 2 parámetros críticos de bifurcación b y d. En la Figura ***3*** se muestra el atractor extraño del sistema.

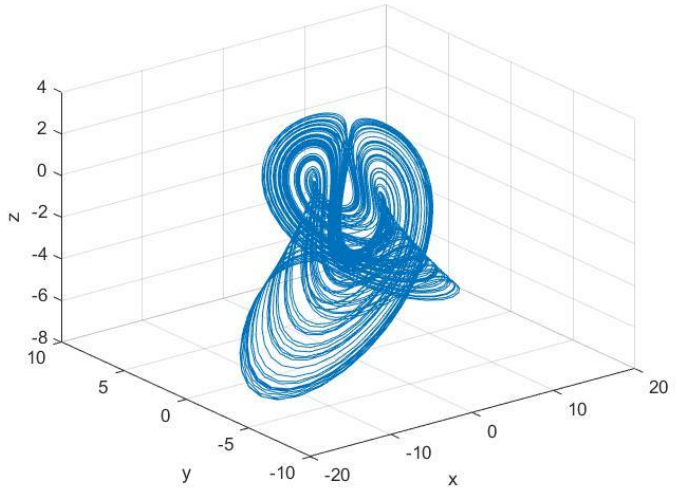


Figura 3. Atractor extraño generado por el sistema caótico MACM con parámetros $a = 2, b = 2, c = 0.5$ y $d = 4$.

2.3. Aplicaciones del caos

Por las características presentadas anteriormente y por muchas otras es que el uso de los sistemas caóticos se ha extendido ampliamente en diferentes ramas de la ingeniería, los siguientes ejemplos muestran como este tipo de sistemas han sido aprovechados.

- **Sistemas de comunicación:** La alta sensibilidad a las condiciones iniciales de los sistemas caóticos es explotada en el área de las comunicaciones para el cifrado de mensajes enviados mediante canales públicos o inseguros. Ejemplos de este tipo de usos los encontramos en Cruz-Hernández y Romero-Haros (2008) y en Murillo-Escobar *et al.* (2015).

- **Trayectorias en robots:** En el área de la robótica los sistemas caóticos han sido utilizados para generar movimientos caóticos en robots humanoides (Volos *et al.,* 2012) y en robots móviles con ruedas como se reporta en Cetina Denis (2017), trabajo del cual se desprende el algoritmo generador de trayectorias caóticas utilizado en este trabajo de tesis.

- **Diseño de antenas:** Debido a la constante miniaturización de los dispositivos de comunicación, el diseño de antenas está enfocado a mejorar el rendimiento de estas. Los diseños con geometrías fractales son una buena solución a este problema como se muestra en (Figueroa-Torres *et al.,* 2013).

2.4. Conclusiones del capítulo

En este capítulo se explican de manera breve los sistemas caóticos y las características que los hacen sistemas dinámicos únicos y extremadamente atractivos en mundo de la investigación, también se dieron una serie ejemplos de trabajos de distintas diciplinas basados en estos sistemas donde cada uno explota al máximo una característica diferente.

Capítulo 3. Robots móviles

Este capítulo incluye una breve descripción sobre robótica y su evolución a la robótica móvil, los diferentes tipos de robots móviles y los tipos de locomoción robótica, haciendo especial énfasis en los robots con ruedas dado que este trabajo de tesis está enfocado en este tipo de robots. También se presentan las ecuaciones matemáticas que describen a un robot móvil diferencial.

A pesar de que el término robot fue acuñado por primera vez en el año 1921 por el dramaturgo checo Karel Capek en su obra Rossum´s Universal Robots (R.U.R), los orígenes de la robótica datan de mucho tiempo antes. En el siglo V a.C. el matemático y filósofo griego Arquitas de Tarento construyó un autómata (máquina que imita la figura y movimiento de un ser vivo) de una paloma propulsada a vapor que imitaba el vuelo. Más recientemente, en el siglo XVIII, el relojero suizo Pierre Jaquez-Droz presentó 3 autómatas capaces de ejecutar instrumentos musicales, escribir y realizar pinturas, dando así origen a las primeras máquinas programables. Con el avance de la tecnología y la era industrial es que se da una aplicación de estas máquinas a tareas específicas originando la robótica tal y como la conocemos hoy en día.

Podemos definir a un robot como una máquina programable capaz de manipular objetos y realizar diversas acciones repetitivas que requieren cierto grado de precisión. Los primeros robots verdaderos encontraron aplicación rápidamente en el área industrial, estos eran en su mayoría brazos manipuladores anclados por alguno de sus extremos que realizaban movimientos en una secuencia determinada, el área de trabajo de estos robots estaba limitada por el largo de la estructura mecánica del brazo.

El siguiente gran paso en la robótica se da con la aparición de la robótica móvil la cual surge como una evolución de estos robots para eliminar las restricciones de movilidad, autonomía (independencia respecto a la intervención humana) y extender el campo de aplicación de la robótica.

Los robots móviles son máquinas programables capaces de desplazarse de manera autónoma en un ambiente dado y no están fijos en una ubicación física; cuentan con un sistema de sensores que les permite obtener información del entorno y poseen uno o varios sistemas de control.

3.1. Tipos de robots móviles

Podemos clasificar a los robots móviles por el ambiente en el operan: aéreos, submarinos y terrestres.

3.1.1. Aéreos

Los vehículos aéreos no tripulados (VANT), son robots móviles capaces de mantener un vuelo controlado y sostenido de manera autónoma y pueden estar impulsados por motores a reacción o eléctricos. Los primeros VANT fueron utilizados con fines militares, principalmente en misiones reconocimiento además de poseer capacidad ofensiva. Gradualmente el avance de la tecnología, así como la disminución en los costos permitió el uso de los VANT en el ámbito civil; ejemplo de ello son los drones modernos o cuadricópteros, estos tienen aplicación en la agronomía, exploración de terrenos, cartografía, monitoreo de especies silvestres, búsqueda y rescate en zonas de desastre, entre otras. En la Figura ***4*** se muestra un VANT utilizado por el ejército de los Estados Unidos.

Figura 4. Vehículo aéreo no tripulado MQ-9 Reaper en vuelo de entrenamiento sobre el desierto de Nevada. Imagen tomada por la fuerza aérea de los Estados Unidos (USAF, 2019).

3.1.2. Submarinos

Los vehículos autónomos submarinos o AUV (Autonomous Underwater Vehicle, por sus siglas en inglés) son robots submarinos no tripulados y autopropulsados capaces de completar de manera independiente tareas previamente programadas. Estos robots son utilizados principalmente en

geociencias marinas para cartografiar el fondo marino, el estudio de vulcanismo submarino, estudio de respiraderos hidrotermales y el estudio de la zona béntica (región ecológica en el nivel más bajo de un cuerpo de agua) (Wynn *et al.,* 2014). En la Figura ***5*** se muestra un AUV utilizado por la Oficina Nacional de Administración Oceánica y Atmosférica (NOAA) para cartografiar el suelo marino.

Figura 5. Vehículo autónomo submarino REMUS 600. Imagen tomada por la Oficina Nacional de Administración Oceánica y Atmosférica (NOAA, 2016).

3.1.3. Terrestres

Los vehículos terrestres no tripulados o UGV (Unmanned Ground Vehicle, por sus siglas en inglés) son la contraparte para uso en la superficie de los VANT y AUV. Al igual que las otras variantes de robots móviles los UGV cuentan con una serie de sensores para interactuar con el entorno, información que luego es utilizada para llevar a cabo la acción para la que fue programado en caso de ser autónomo o enviar esta información a un operador humano si es tele operado.

Al igual que sus contrapartes, las primeras aplicaciones de los UGV tuvieron fines militares donde realizaban tareas de reconocimiento y ataque pasando después al ámbito civil donde versiones modernas son utilizados para la cosecha de cultivos, en la minería para la exploración de túneles, en el hogar para la limpieza, también son utilizados en la exploración espacial en el programa espacial MEP (Mars Exploration Program) de la NASA como el mostrado en la Figura ***6***.

Figura 6. Vehículo de exploración Rover utilizado en el programa espacial MEP. Imagen tomada por el laboratorio de propulsión a chorro de la NASA (NASA JPL, 2002).

3.2. Tipos de locomoción terrestre

Los vehículos terrestres no tripulados pueden ser clasificados de acuerdo con su sistema de locomoción. Las 3 configuraciones básicas son: patas que imitan a las de los animales, cuerpos articulados similares a serpientes y los dispositivos rotacionales como ruedas y orugas (Hirose, 1991). Siendo estos últimos los más utilizados en la investigación debido a relativa facilidad de construcción y control.

3.2.1. Patas

Este tipo de locomoción se basa en el uso de miembros articulados con al menos 2 grados de libertad similares a las patas de los animales, los cuales permiten mantener aislado el cuerpo del robot del terreno además de brindar una ventaja sobre otros tipos de locomoción terrestre al ser capaces de atravesar terrenos accidentados imposibles de transitar con ruedas u orugas, mediante este tipo de locomoción es posible conseguir la omnidireccionalidad y el deslizamiento en la locomoción es mucho menor (Ollero Baturone, 2001). Las desventajas de la locomoción por patas es su alto consumo energético además de la complejidad en el control de las patas para mantener el equilibrio. Existen diferentes configuraciones basadas en el número de patas que van desde 1 hasta 8 patas. En la Figura ***7*** se muestra el robot cuadrúpedo Spot de Boston Dynamics.

Figura 7. Robot autónomo Spot de la compañía Boston Dynamics. Imagen tomada de www.bostondynamics.com

3.2.2. Cuerpo articulado

En este tipo de locomoción, módulos individuales son conectados de manera lineal similar a un tren mediante uniones flexibles, los módulos pueden estar impulsados mediante ruedas, orugas o patas. Las características principales de los robots móviles con este tipo de locomoción son su excelente adaptabilidad al terreno y la capacidad de carga útil (Hirose y Morishima, 1990). Ejemplo de este tipo de configuración puede encontrarse en Fu y Li (2020). En la Figura ***8*** se muestra un robot de cuerpo articulado desarrollado por la Universidad Johns Hopkins.

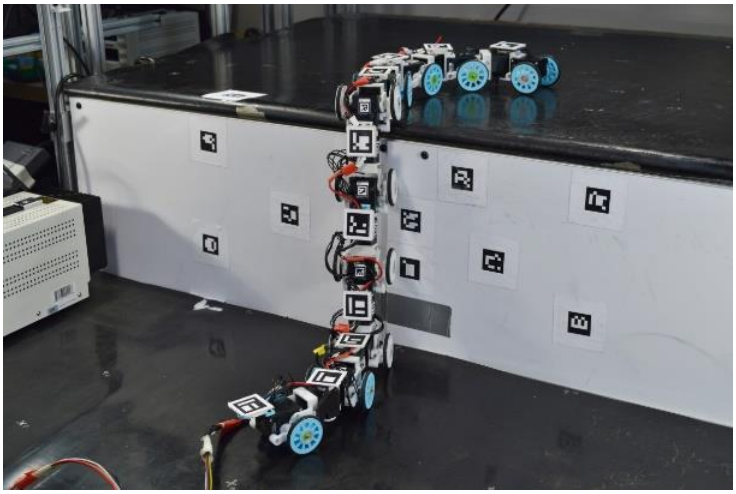


Figura 8. Robot de cuerpo articulado. Imagen tomada por laboratorio de terradinámica de la universidad Johns Hopkins (Chen Li, 2020).

3.2.3. Orugas

La locomoción mediante orugas hace uso de pistas de deslizamiento que brindan impulso al robot. Este tipo de locomoción proporciona un buen desempeño en terrenos firmes y debido a que las pistas proporcionan al robot móvil un área de contacto con el suelo mayor en comparación a las ruedas o las patas el desempeño en terrenos blandos como arena, nieve o barro es mejor. Las principales desventajas de este tipo de locomoción son su elevada potencia de rodadura y la alta vibración al transitar en un terreno irregular (González *et al.,* 2015). A pesar de las aparentes limitaciones en este tipo de locomoción se han desarrollado robots capaces de subir escaleras y son utilizados en el ámbito militar para misiones de reconocimiento y rescate en áreas urbanas. Ejemplo de esto se puede encontrar en (Helmick *et al.,* 2002), ver Figura ***9***.

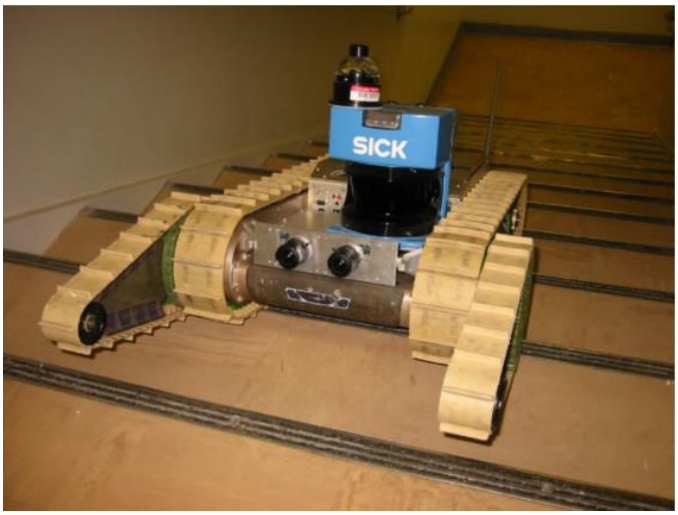


Figura 9. Robot autónomo con locomoción por urugas utilizado para misiones militares en áreas urbanas. Imagen tomada de (Helmick *et al.,* 2002).

3.2.4. Ruedas

Las ruedas son ampliamente el sistema de locomoción más común y utilizado en los robots móviles terrestres, debido principalmente a su simplicidad, eficiencia y a su relativamente fácil control. Las principales desventajas de los robots móviles con ruedas (RMR) son la poca adaptabilidad al terreno que presentan, por lo que las irregularidades sobre la superficie de trabajo pueden limitar o incluso impedir el movimiento, el deslizamiento en cierto tipo de superficies y su bajo desempeño en terrenos blandos.

Dentro de los RMR existen diferentes configuraciones cinemáticas las cuales le confieren al robot diferentes características y propiedades en su maniobrabilidad y dependen del propósito final del robot. Las siguientes configuraciones cinemáticas son las más utilizadas en los RMR.

- **Ackerman:** Esta configuración consta de 2 ruedas traseras que otorgan la tracción al robot y 2 ruedas delanteras encargadas de la dirección. Esta configuración proporciona al RMR una buena tracción en superficies inclinadas. Como ejemplo de este tipo de configuración podemos mencionar los automóviles. En la Figura ***10*** se muestra el esquema de este tipo de configuración.

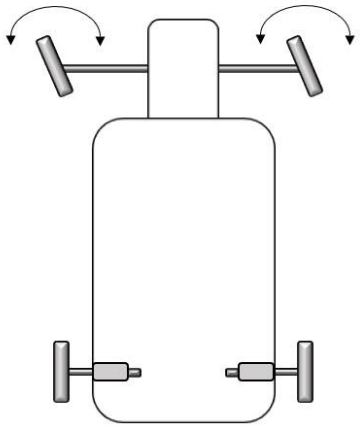


Figura 10. Configuración Ackerman.

- **Triciclo clásico:** Esta configuración cuenta con una rueda delantera encargada de la tracción y dirección del robot y 2 ruedas traseras que se mueven libremente encargadas de proporcionar la estabilidad, como se muestra en la Figura ***11***. Esta configuración ofrece una alta maniobrabilidad, sin embargo, presenta ciertos problemas de estabilidad y tracción en terrenos accidentados por lo que es utilizado principalmente en robots para interiores.

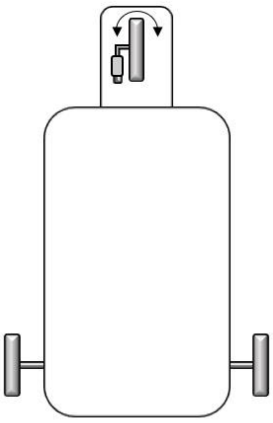


Figura 11. Configuración triciclo clásico.

- **Skid steer:** En este tipo de configuración el robot cuenta con al menos 2 ruedas en cada lado, cada lado trabaja de manera conjunta por lo que el movimiento es el resultado de la suma de todas las velocidades de las llantas en cada lado. Los giros se realizan controlando las velocidades de cada lado del robot. Esta configuración es ampliamente utilizada en vehículos enfocados en minería y agricultura. En la Figura ***12*** se muestra el esquema de un RMR con una configuración Skid steer.

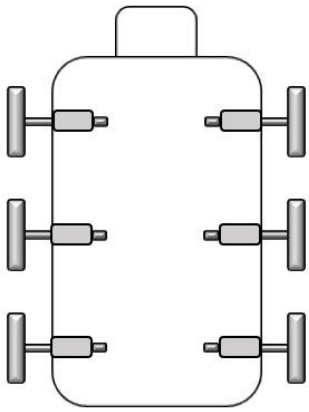


Figura 12. Configuración Skid steer.

- **Tracción síncrona:** Esta configuración consta de al menos 3 ruedas apuntando en la misma dirección que giran de manera sincronizada, además todas ellas proporcionan tracción y dirección al robot. Los cambios de dirección se logran girando las ruedas sobre un eje vertical de manera simultánea. La trasmisión en este tipo de configuración funciona mediante coronas de engranes o correas concéntricas (Ollero Baturone, 2001). La tracción síncrona puede otorgar omnidireccionalidad al robot sin la necesidad de utilizar ruedas especiales. Sin embargo, una de las principales desventajas es su complejidad mecánica. La Figura ***13*** muestra un esquema de este tipo de configuración.

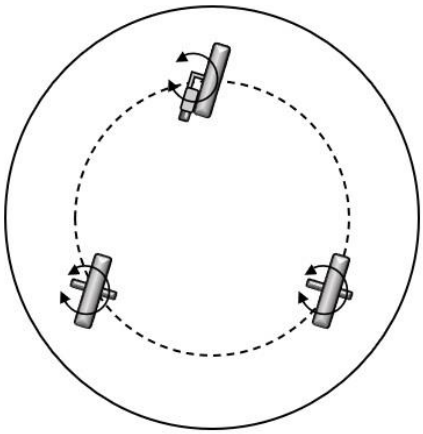


Figura 13. Configuración de tracción síncrona.

- **Omnidireccional:** Existen 2 formas de obtener omnidireccionalidad en los RMR, la primera de ellas es utilizando una configuración síncrona con un juego de ruedas como se muestra en el ejemplo anterior, mientras que la segunda se logra empleando en el robot ruedas diseñadas específicamente para este propósito como por ejemplo las ruedas suecas, de castor o esféricas. La maniobrabilidad de este tipo de configuración es su mayor ventaja ya que puede desplazarse en cualquier dirección sin la necesidad de reorientarse. La mayor desventaja que presenta esta configuración es un alto grado de complejidad en el control. La Figura ***14*** muestra un robot omnidireccional utilizando ruedas suecas.

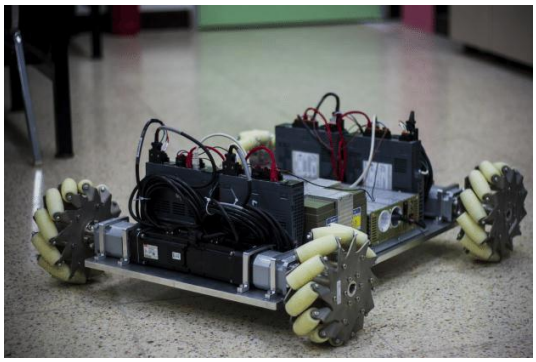


Figura 14. Robot omnidireccional con ruedas suecas actuadas. Imagen tomada de (Delgado *et al.,* 2016).

- **Tracción diferencial:** En este tipo de configuración el movimiento se consigue por la diferencia de velocidades entre 2 ruedas laterales conectadas a un mismo eje, pero actuadas de manera independiente, puede contar con una tercera rueda libre o algún otro punto de apoyo para la estabilidad. Una ventaja de esta configuración es que el robot puede cambiar su orientación sin necesidad de un movimiento de traslación por lo que principalmente utilizado en robots enfocados a trabajo en interiores. En la Figura ***15*** se muestra un esquema de este tipo de configuración.

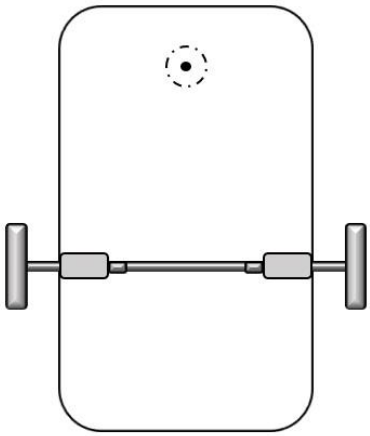


Figura 15. Configuración diferencial.

3.3. Khepera III

El Khepera III (Figura ***16***) es la tercera versión del robot móvil insignia de la compañía suiza K-Team. Es un robot compacto con un diámetro de $130mm$, $70mm$ de alto y un peso aproximado de 650 gramos. Posee una configuración diferencial por lo que cuenta con 2 ruedas encargadas de su locomoción y un soporte deslizante que le proporciona estabilidad. Cuenta con múltiples sensores infrarrojos y ultrasónicos para interactuar con el ambiente. La arquitectura del robot es modular por lo que se puede equipar con cámaras, gripers, sensores adicionales incluso un sistema de posicionamiento en interiores a partir de puntos de referencia, además posee los medios necesarios para comunicarse inalámbricamente.

El Khepera III, debido a sus dimensiones reducidas, es ampliamente utilizado con fines educativos y de investigación en temas como control (Papcun *et al.,* 2016), navegación (Prorok *et al.,* 2010) y comportamientos colectivos (Marjovi *et al.,* 2009).

Figura 16. Robot Khepera III. Imagen tomada de (Schidt *et al.,* 2015).

3.3.1. Modelo cinemático de un robot con tracción diferencial

En el modelado cinemático de un robot con configuración diferencial se toman en cuenta las siguientes consideraciones:

- El movimiento del robot se realiza sobre superficie horizontal plana.
- No existen elementos flexibles en la estructura.
- No existe deslizamiento.

En (Dhaouadi y Hatab, 2013) se describe el modelo cinemático de un robot móvil con configuración diferencial en términos de las velocidades lineal y angular, como el mostrado en la Figura ***17***, donde la ubicación del robot está determinada por las coordenadas x, y y un ángulo de orientación θ.

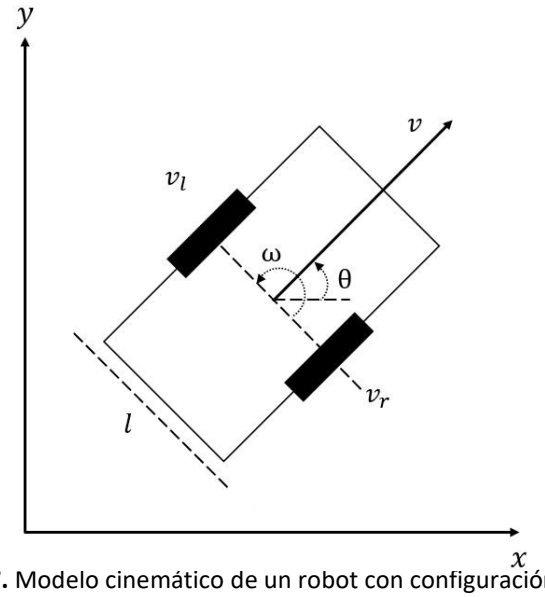


Figura 17. Modelo cinemático de un robot con configuración diferencial

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}cos(\theta) & 0\\ sen(\theta) & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}v\\ \omega\end{bmatrix} \tag{2}$$

La velocidad lineal del robot v es el promedio de las velocidades lineales de cada una de las llantas

$$v = \frac{v_r + v_l}{2}, \tag{3}$$

Mientras que la velocidad angular ω es la diferencia de velocidades de las ruedas entre la distancia entre las ruedas l

$$\omega = \frac{v_r - v_l}{l}. \tag{4}$$

Donde v_r es la velocidad lineal de la rueda derecha y v_l es la velocidad lineal de la rueda izquierda y son las entradas de control.

3.4. Conclusiones del capítulo

En este capítulo de dio un breve repaso de la historia de la robótica y su evolución hasta nuestros días. Se vio de manera general las características que hacen únicos a los robots móviles y los distintos tipos de robots móviles, se abordó de manera más detallada los robots móviles terrestres y los diferentes tipos de locomoción haciendo especial énfasis en los robots móviles con ruedas (RMR). Se presentó el modelo matemático de un RMR con tracción diferencial y sus entradas de control. Este modelo junto con el algoritmo generador que será presentado en el siguiente capítulo constituyen la base de esta tesis.

Capítulo 4. Generación de trayectorias e implementación en un modelo virtual del robot Khepera III

En este capítulo se describe el modelo de control utilizado para la generación de trayectorias caóticas en el robot Khepera III empleando un algoritmo generador de trayectorias y un sistema caótico.

Se presentan simulaciones numéricas y la implementación de las trayectorias en un modelo virtual del robot Khepera III.

4.1. Generación de trayectorias

Para la generación de las trayectorias presentadas en esta tesis se emplea el algoritmo generador presentado en (Cetina Denis *et al.,* 2018). La ventaja de este algoritmo por sobre otros métodos de generación de trayectorias es su fácil implementación y adaptabilidad a distintos sistemas caóticos y robots móviles. Para inducir las dinámicas caóticas en el robot Khepera III se utiliza el sistema caótico MACM (1), ver Anexo A.

4.1.1. Algoritmo generador de trayectorias caóticas

El algoritmo generador funciona de la siguiente manera: El primer paso es tomar la posición inicial del robot en el plano x, y para utilizarlas como condiciones iniciales del sistema caótico a utilizar. El paso 2 es calcular los valores de los estados del sistema caótico. En el paso 3 se asignan estos valores a las velocidades lineal v y angular ω en las ecuaciones de movimiento del robot para inducir las dinámicas caóticas.

En el paso 4 se calculan las velocidades de las ruedas izquierda v_l y derecha v_r a partir de v y ω. En el paso 5 se realiza un escalamiento de los datos obtenidos para ajustar las velocidades a los máximos y mínimos soportados por el robot. Estas velocidades ajustadas reciben el nombre de vd_l y vd_r y serán enviadas al robot Khepera III. El paso 6 corresponde a un tiempo de ejecución de 2 segundos, este tiempo

es necesario para que el robot tenga tiempo de desplazarse sobre la superficie de trabajo antes de cambiar nuevamente sus velocidades. La Figura ***18*** muestra el funcionamiento del algoritmo.

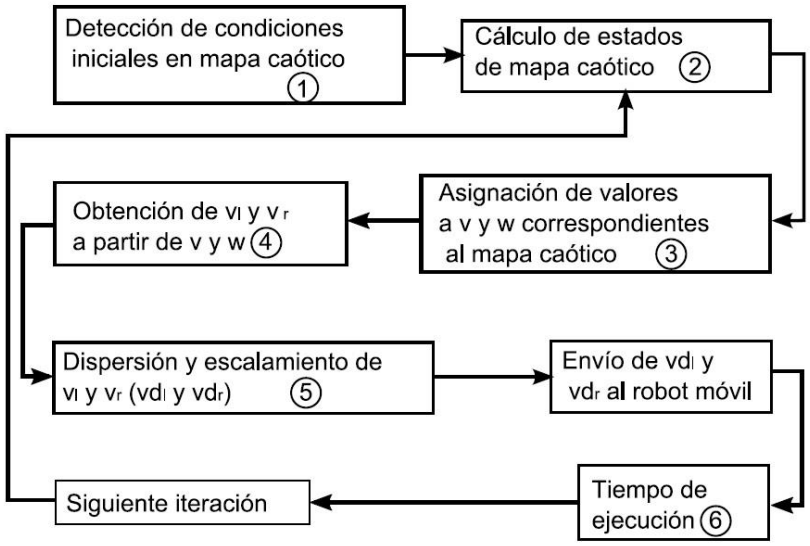


Figura 18. Diagrama a bloques algoritmo generador de trayectorias caóticas. imagen tomada de (Cetina Denis, 2017).

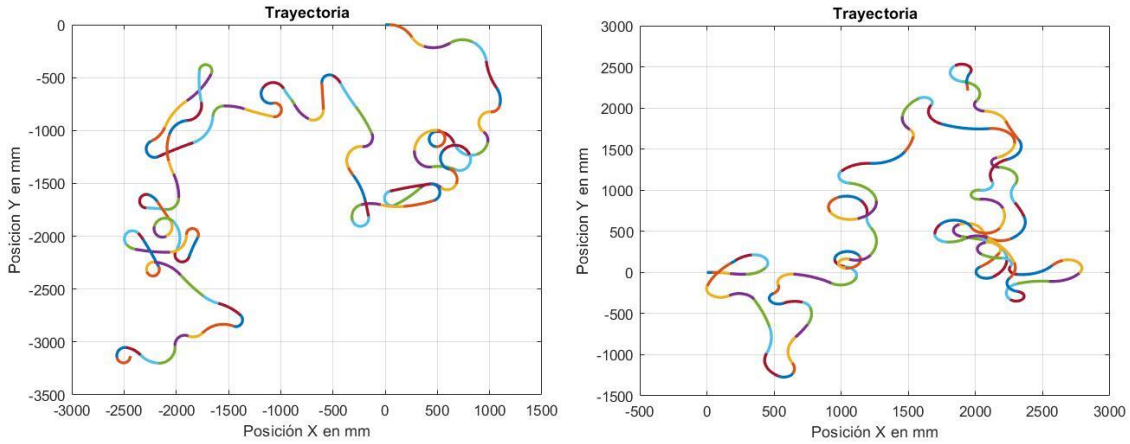
4.1.2. Simulación numérica de trayectorias

Dado que el robot Khepera III posee una configuración diferencial cuenta con 2 entradas (cada una de las llantas) y el sistema MACM tiene 3 entradas (cada uno de sus estados), se cuenta con cierta flexibilidad en la generación de trayectorias. Es posible generar un gran número de trayectorias distintas sin la necesidad de cambiar las condiciones iniciales eligiendo los estados del sistema MACM que adoptarán como condición inicial las posiciones x_r y y_r del robot o intercambiando los estados que dictarán las dinámicas del robot.

Es preciso mencionar que para todas las trayectorias mostradas en este trabajo de tesis se fijaron los estados del sistema MACM en $[1 \quad x_r \quad y_r]$. Los únicos cambios para la generación de distintas trayectorias se dan en los estados que afectan la velocidad angular ω y lineal v del robot Khepera III y en las condiciones iniciales (x_r, y_r, θ_0).

Las trayectorias mostradas en la Figura ***19*** corresponden a 100 iteraciones del algoritmo generador situando al robot en el origen del área de trabajo $(x_r, y_r, \theta_0) = (0,0,0^\circ)$. La línea de colores representa la trayectoria generada y cada color representa un tiempo de 2 segundos, tiempo suficiente para que el robot cubra una distancia significativa antes de cambiar de velocidades, durante este periodo de tiempo el robot

se mueve de acuerdo a lo dictado por los estados del sistema MACM en cada iteración del algoritmo. Se puede observar que al cambiar los estados que afectan la dinámica del robot podemos obtener trayectorias diferentes incluso manteniendo condiciones iniciales similares.



(a) (b)

Figura 19. Trayectorias caóticas de 100 iteraciones generadas por el algoritmo con diferentes estados de MACM influenciando las velocidades lineal y angular del robot: a) $v = x_{macm}\ \omega = y_{macm}$, b) $v = y_{macm}\ \ \omega = z_{macm}$.

A primera vista se puede observar que las trayectorias no siguen un patrón reconocible lo cual es deseable para las aplicaciones propuestas en esta tesis. Sin embargo, se observa que las trayectorias pueden avanzar en cualquier dirección sin tener límites aparentes, la ausencia de estos límites en una aplicación real representa un problema ya que, si se planea usar al robot, por ejemplo, en vigilancia de un terreno, el robot estará confinado a un área de un tamaño específico y no deberá sobrepasar esos límites.

Teniendo en cuenta lo anterior se propone un método para poder confinar al robot a un área de trabajo específica en el cual el robot cambia de dirección al llegar a los límites.

El robot al aproximarse a los bordes del área de trabajo lo hace con un ángulo de orientación con respecto al eje horizontal que llamaremos ángulo de entrada al borde θ_e. El método de confinamiento propuesto es un método reflexivo y consiste en obtener un ángulo de "rebote" o salida de los bordes θ_s que refleje el ángulo de entrada como se muestra en la Figura ***20*** y en la Figura ***21***.

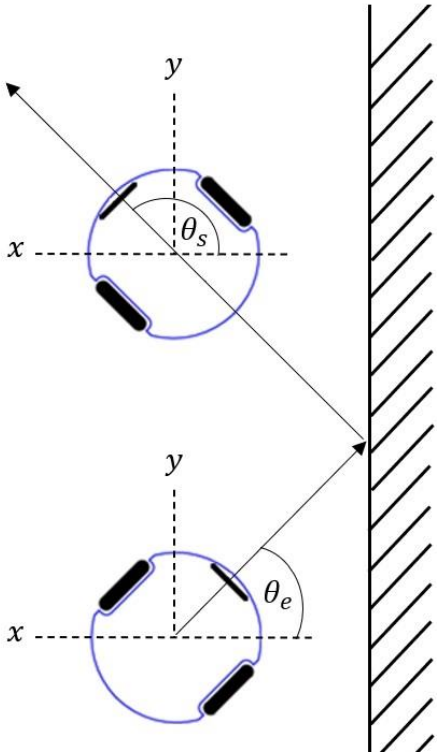


Figura 20. Método de confinamiento reflexivo propuesto donde θ_e es el ángulo de entrada y θ_s es el ángulo de salida.

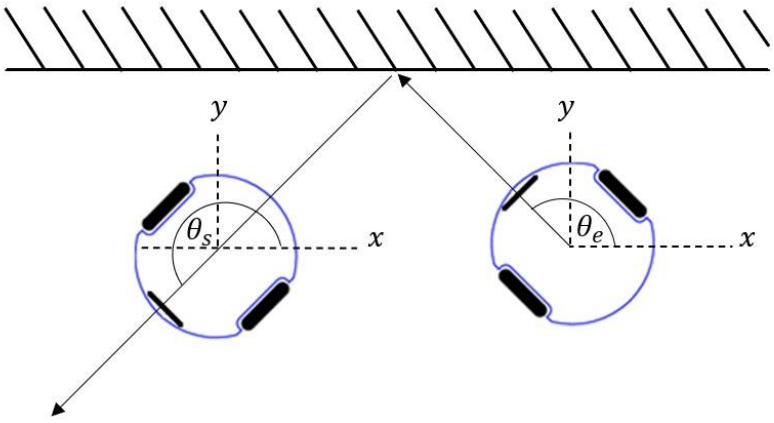


Figura 21. Método reflexivo para bordes superior e inferior donde θ_e es el ángulo de entrada y θ_s es el ángulo de salida.

Siendo θ_e el ángulo de entrada, el ángulo de salida θ_s cuando el robot se aproxima a los bordes del área de trabajo se obtiene de la siguiente manera:

$$\theta_s = \begin{cases} (360 - \theta_e) - 180 & bordes\ laterales \\ 360 - \theta_e & bordes\ superior\ e\ inferior \end{cases} \quad (5)$$

Debido a que el robot Khepera III se encuentra en el laboratorio de sincronización y sistemas complejos ubicado en el departamento de física aplicada del CICESE, el área de trabajo utilizada en esta tesis será la mesa de experimentos disponible en el laboratorio cuyas dimensiones son $900mm \times 1500mm$.Tomando en cuenta las dimensiones del área de trabajo y aplicando el método reflexivo explicado anteriormente para confinar al robot, se procede a realizar las simulaciones numéricas.

En la Figura ***22*** se muestran las mismas trayectorias de la Figura 19, pero ahora con un área de trabajo delimitada. El recuadro negro representa el área de trabajo y se puede observar que el método reflexivo propuesto funciona ya que el robot entra y sale de los bordes del área en ángulos similares y jamás traspasa los límites.

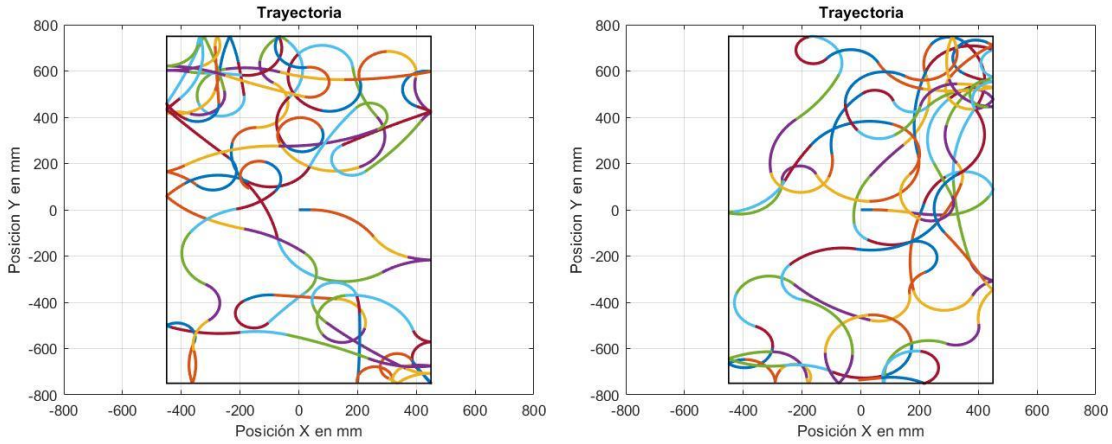


Figura 22. Trayectorias caóticas confinadas al área de trabajo utilizando el método reflexivo.

Como se mencionó anteriormente, al contar con un número mayor de entradas en el sistema caótico MACM con respecto al robot Khepera III, es posible realizar combinaciones de los 3 estados del sistema para generar trayectorias diferentes manteniendo condiciones iniciales.

En este trabajo de tesis se trabajará solo con 2 tipos de combinaciones de estados, la primera denominada combinación simple la cual es mostrada en la Figura ***23***. En este tipo de combinación durante las primeras 50 iteraciones del algoritmo los estados que actúan sobre v y ω son los estados x_{macm} y y_{macm}, mientras que en las últimas 50 iteraciones los estados que afectan a v y ω son y_{macm} y z_{macm}. Las condiciones iniciales son: $x_r = 0,\ y_r = 0$.

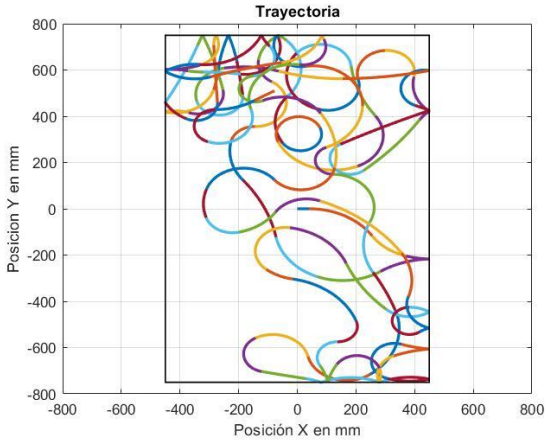


Figura 23. Trayectoria caótica generada con una combinación de estados simple.

El segundo tipo de combinación es la mostrada en la Figura ***24***. Este tipo de combinación está basada en números primos, durante las 100 iteraciones del algoritmo cuando el número de iteración n es primo, los estados que actúan sobre v y ω son x_{macm} y y_{macm} de lo contrario los estados que afectan a v y ω son x_{macm} y z_{macm} respectivamente. Las condiciones iniciales para esta trayectoria son las siguientes: $x_r = 0,\ y_r = 0$.

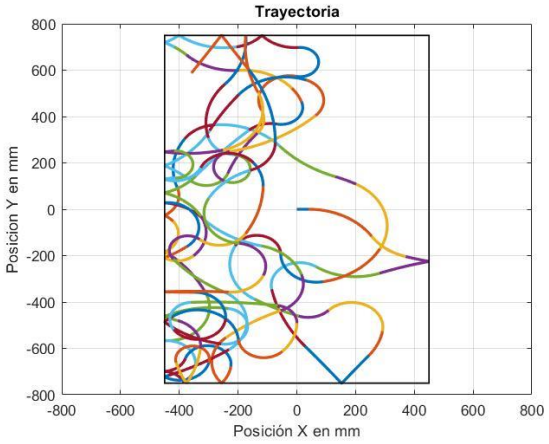
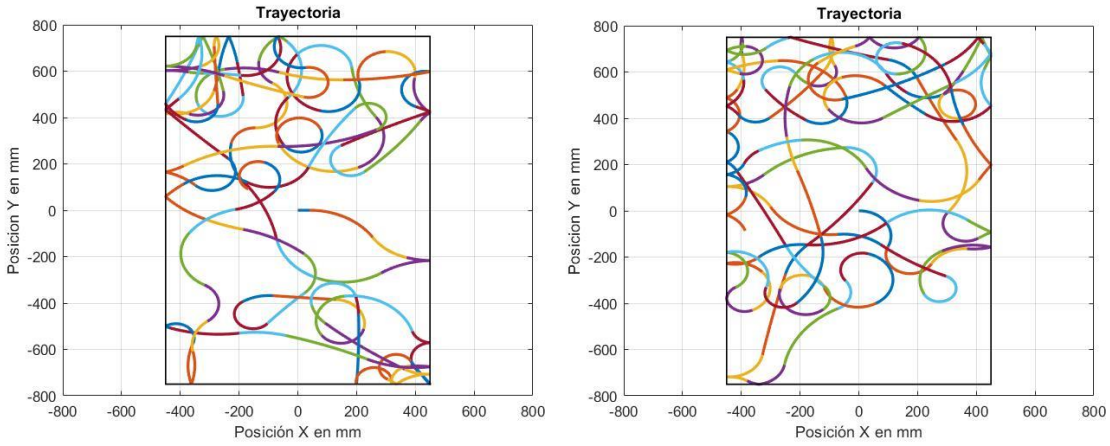


Figura 24. Trayectoria caótica generada por una combinación de estados basada en números primos.

De esta manera cuando se hable de combinación simple o de números primos se hará referencia las explicadas anteriormente.

Las posibilidades de combinar estados para generar trayectorias sin cambiar las condiciones iniciales crecen conforme se va aumentando el número de iteraciones en el algoritmo. El poder generar diferentes trayectorias con las mismas condiciones iniciales es una diferencia importante en comparación a otros métodos de generación de trayectorias en los que se depende casi en su totalidad en la diferencia de condiciones iniciales para obtener trayectorias diferentes.

Las trayectorias, al ser generadas utilizando un sistema caótico, poseen las características clave de este tipo de sistemas como se puede observar en la Figura ***25*** donde se generaron 2 trayectorias diferentes. Para las trayectorias se fijaron $v = x_{macm}$ y $\omega = y_{macm}$ y se cambiaron ligeramente las condiciones iniciales (x_r, y_r) dando como resultado trayectorias completamente diferentes.



(a) $x_r = 0, y_r = 0, \theta = 0^{\circ}$ (b) $x_r = 0.01, y_r = 0.01, \theta = 0^{\circ}$

Figura 25. Trayectorias caóticas con condiciones iniciales ligeramente diferentes.

A pesar de que la diferencia ambas trayectorias es de $0.01\ mm$ se puede observar que no guardan relación alguna entre sí, lo que deja en evidencia su extrema sensibilidad a condiciones iniciales.

4.1.3. Porcentaje de cobertura de las trayectorias

Para tener una idea de la eficiencia de las trayectorias generadas en posibles aplicaciones reales es necesario calcular el porcentaje del área de trabajo que el robot Khepera III cubre siguiendo estas trayectorias. Para lograr esto se dividió el área de trabajo en una cuadricula de 9x15 generando así 135 cuadros con medidas laterales de $100mm$ para después contar los cuadros que son visitados por la trayectoria y de este modo obtener el porcentaje. Dado que el robot Khepera III tiene un diámetro aproximado de $135mm$, los cuadros tienen el tamaño suficiente para obtener una buena aproximación del porcentaje de cobertura real.

En la Figura ***26*** se muestra el porcentaje de cobertura de una trayectoria utilizada como referencia, fue realizada con 100 iteraciones del algoritmo y con condiciones iniciales $x_r = 0, y_r = 0, \theta = 0^{\circ}$ y utilizando $v = x_{macm}$, $\omega = y_{macm}$ para inducir las dinámicas caóticas.

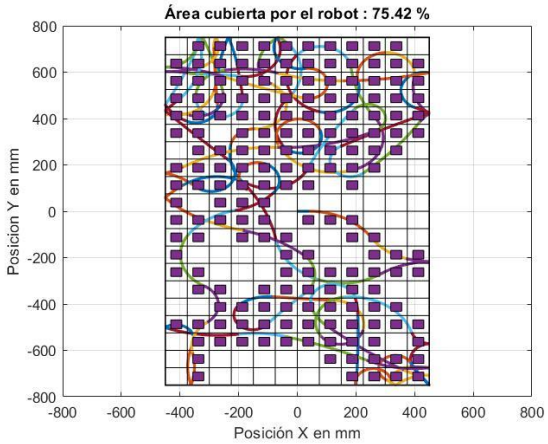
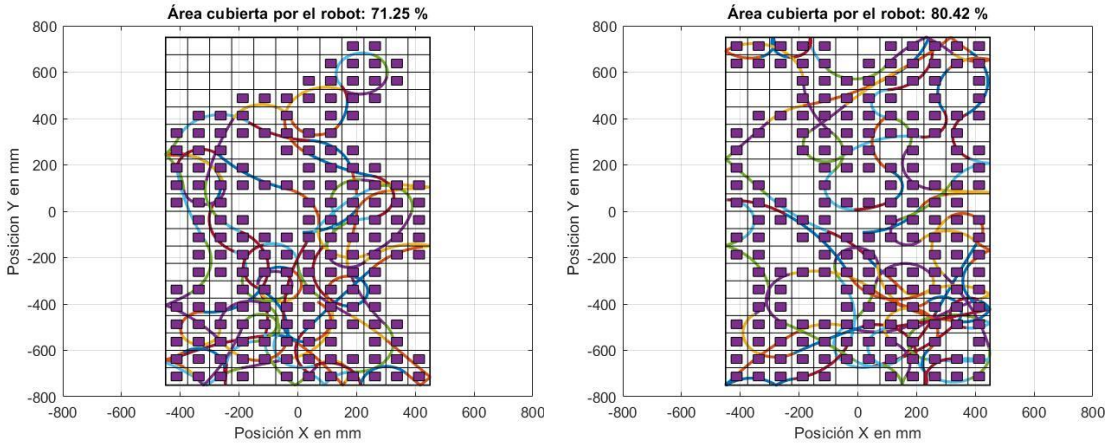


Figura 26. Porcentaje de cobertura para $(x_r, y_r, \theta_0) = 0$ con $v = x_{macm}$, $\omega = y_{macm}$

Se observa que, a pesar de ser una trayectoria relativamente corta se cubre casi toda el área de trabajo. En la Figura ***27*** se compara la eficiencia en la cobertura del área de trabajo de 2 trayectorias con condiciones iniciales x_r , y_r similares y generadas utilizando los distintos métodos de combinación de estados presentados anteriormente. Se eligieron de manera arbitraria las siguientes condiciones iniciales: $x_r = 16.1450$, $y_r = -90.8891$. Las trayectorias fueron realizadas empleando 100 iteraciones del algoritmo.



(a) (b)

Figura 27. Trayectorias realizadas utilizando diferentes métodos de combinación de estados: (a) combinación basada en números primos, (b) combinación de estados simple.

Los dos métodos de combinación de estados propuestos en esta tesis presentan un buen porcentaje de cobertura del área de trabajo para condiciones iniciales similares.

La Tabla ***1*** muestra cuantas iteraciones les toma a las 2 trayectorias de combinación de estados cubrir por completo el área de trabajo.

Tabla 1. Porcentajes de cobertura para distintas iteraciones.

Tipo de combinación utilizada	Iteración				
	100	200	300	400	500
Simple	80.42%	95%	98.33	100%	100%
Números primos	71.25%	92.08%	97.50%	98.33%	100%

4.2. Implementación de las trayectorias en un modelo virtual del Khepera III

Debido a la pandemia por COVID-19 ocurrida durante la realización de este trabajo de tesis, fue imposible validar de manera experimental los resultados obtenidos en las simulaciones numéricas; por lo tanto, se optó por recurrir al software de simulación robótica CoppeliaSim® en su versión licenciada para estudiantes.

CoppeliaSim® (antes V-rep) es un software gratuito de simulación robótica de la empresa Coppelia Robotics AG. CoppeliaSim® ofrece un entorno virtual de simulación para la creación de prototipos, simulación de automatización industrial, entre otros. Este software es multiplataforma y está basado en scripts, el lenguaje de programación utilizado es LUA; sin embargo, por medio de API, se puede crear una conexión entre CoppeliaSim y otra aplicación externa permitiendo el uso de lenguajes como C, C++, Python, Java, Matlab, Octave y Urbi. CoppeliaSim® da la opción de crear o importar modelos robóticos, además de incluir modelos de robots ampliamente utilizados en la investigación como por ejemplo el brazo robótico KUKA (Figura ***28***), el robot humanoide NAO de Aldebaran Robotics (Figura ***29***) y por supuesto el robot Khepera III de K-Team (Figura ***30***).

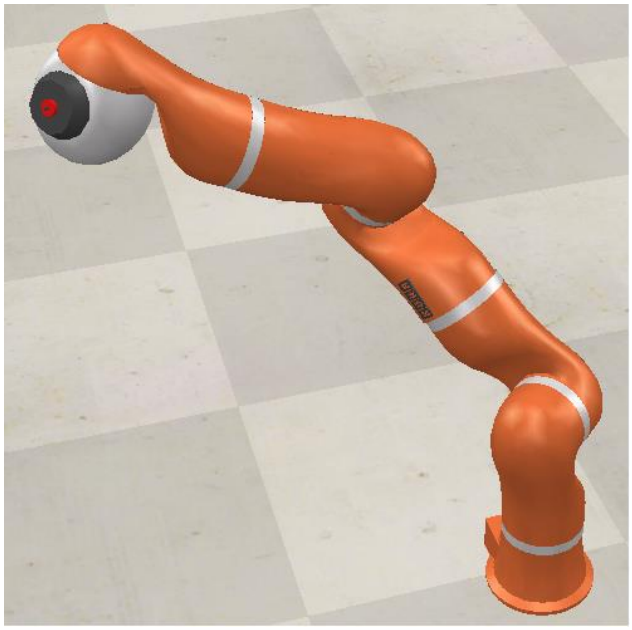


Figura 28. Modelo virtual de brazo robótico KUKA disponible en CoppeliaSim.

Figura 29. Modelo virtual del robot Humanoide NAO disponible en CoppeliaSim.

Figura 30. Modelo virtual del robot Khepera III disponible en CoppeliaSim.

Las simulaciones se realizaron utilizando el software Matlab R2017b para efectuar todos los cálculos requeridos en la generación de las trayectorias, así como los estados del sistema MACM, velocidades del robot tanto lineal como angular y las velocidades de cada llanta. La información después es enviada al robot virtual y CoppeliaSim envía la posición del robot virtual como retroalimentación a Matlab. El método reflexivo para confinar al robot al área de trabajo es calculado de igual manera por Matlab utilizando los datos de posición retroalimentados por CoppeliaSim. El uso de los múltiples sensores del robot virtual para evitar la colisión con los imites del área de trabajo se descartó debido a que los rangos de detección mínima de los sensores son de aproximadamente $15\ cm$. Cuando el objeto a detectar, en este caso el borde del área de trabajo, está a una distancia menor ocasiona lecturas erróneas y esto hace imposible que el método reflexivo funcione. La comunicación entre Matlab y CoppeliaSim se realiza utilizando un API, ver anexo B.

En la Figura ***31*** se muestra el entorno virtual de CoppeliaSim, se observa el área de trabajo y al robot Khepera III situado en el origen.

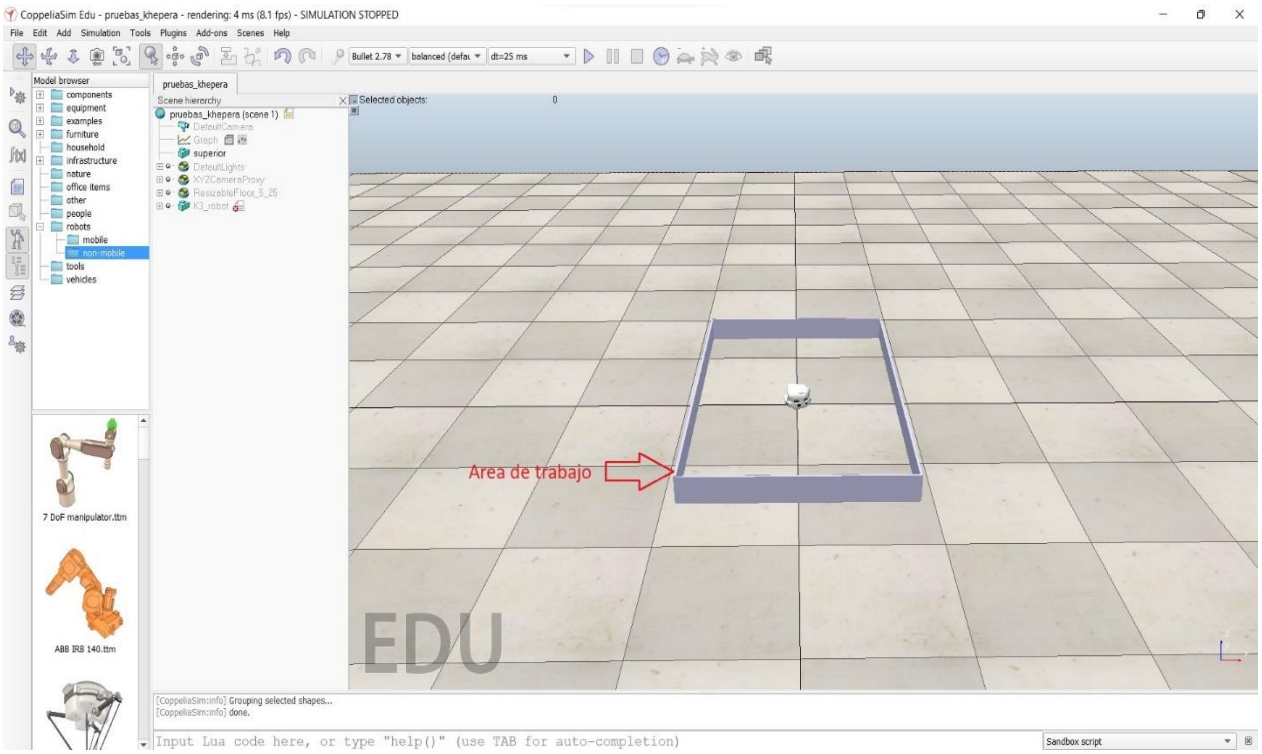


Figura 31. Entorno virtual del simulador CoppeliaSim.

Para la parte experimental con el modelo virtual se diseñó la siguiente metodología: Se escogieron de manera arbitraria 2 pares de coordenadas iniciales para el robot (x_r, y_r) y para cada par de coordenadas se realizaron 3 trayectorias distintas con una longitud de 100 iteraciones cada una. La primera trayectoria de cada par de coordenadas se denomina *Trayectoria de control* y en ella las dinámicas caóticas son inducidas utilizando $v = x_{macm}$ y $\omega = y_{macm}$. En la segunda trayectoria de cada par de coordenadas se emplea la combinación de estados simple presentada en secciones anteriores y en la última trayectoria de cada par de coordenadas se emplea la combinación basada en números primos.

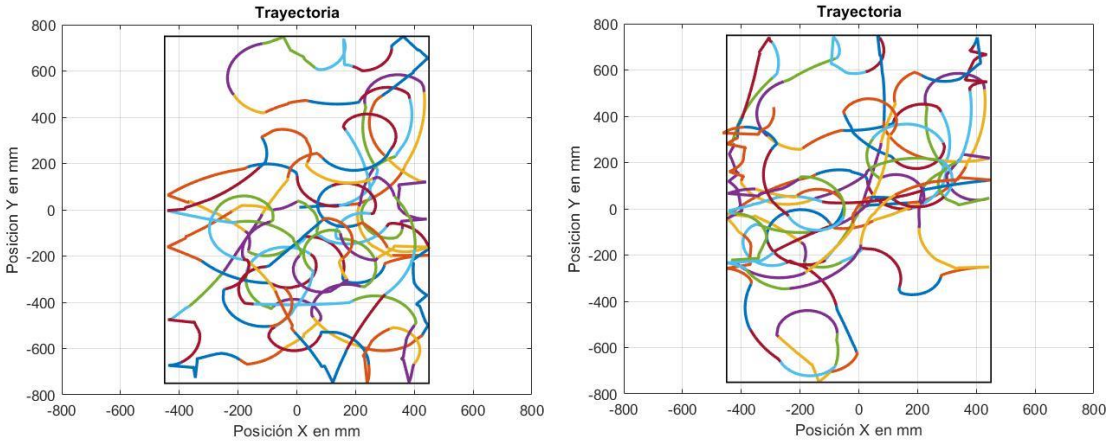
Para tener una idea de su eficiencia en patrullar el área de trabajo se muestra también su porcentaje de cobertura.

Los pares de coordenadas utilizados en las trayectorias son los siguientes:

1. $x_r = -57.4,\ y_r = 5.6$

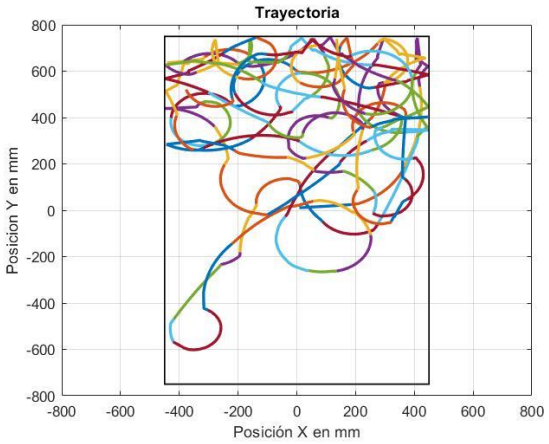
2. $x_r = -27.2,\ y_r = -309.6$

Los resultados de las simulaciones con el modelo virtual del robot para el primer par de coordenadas se muestran en la Figura ***32***.



(a) Trayectoria de control.

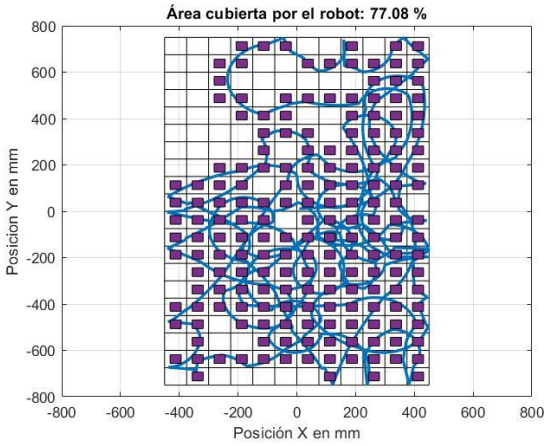
(b) Trayectoria de combinación simple.



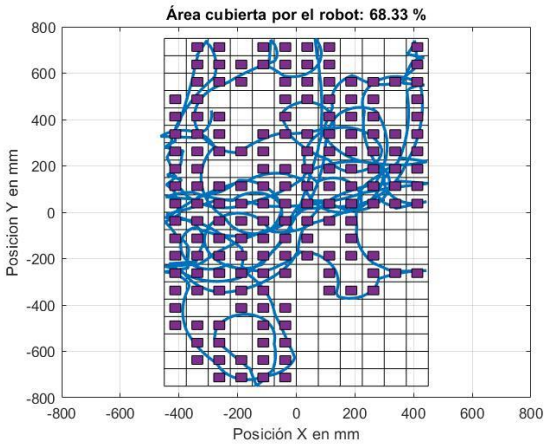
(c) Trayectoria de combinación basada en números primos.

Figura 32. Resultados experimentales para el par de coordenadas $x_r = -57.4,\ y_r = 5.6$

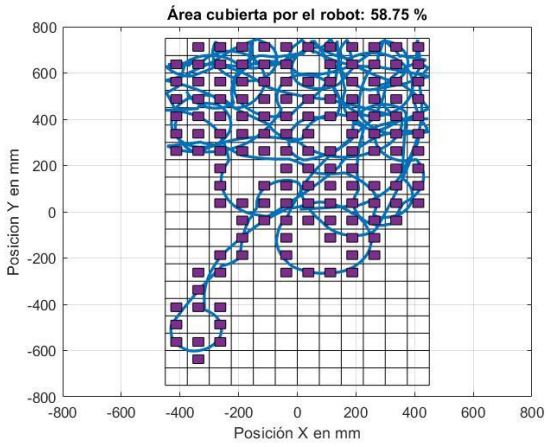
Los resultados experimentales corresponden con lo visto en las simulaciones numéricas mostrando trayectorias completamente impredecibles. Los porcentajes de cobertura para las trayectorias del primer par de coordenadas se muestran en la Figura ***33***.



(a) Trayectoria de control.



(b) Trayectoria de combinación simple.

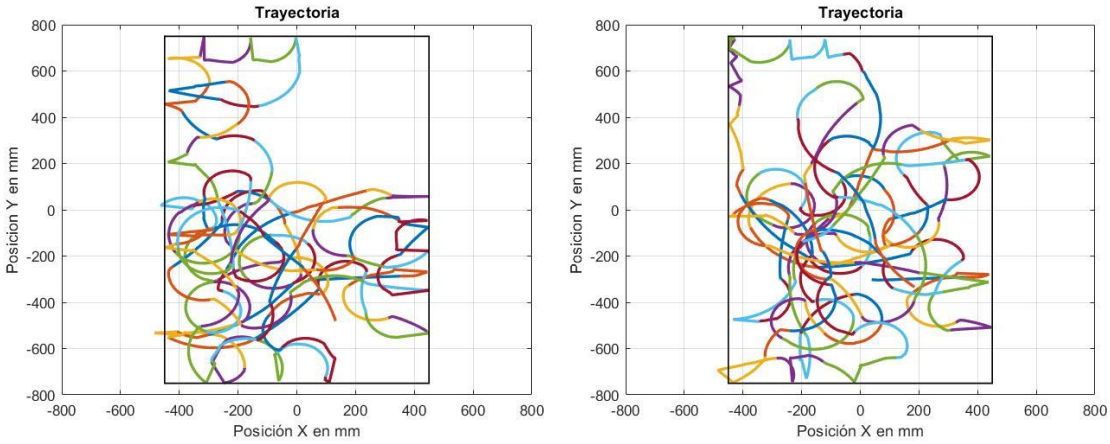


(c) Trayectoria de combinación basada en números primos.

Figura 33. Porcentajes de cobertura de los resultados experimentales para el par de coordenadas $x_r = -54.7,\ y_r = 5.6$.

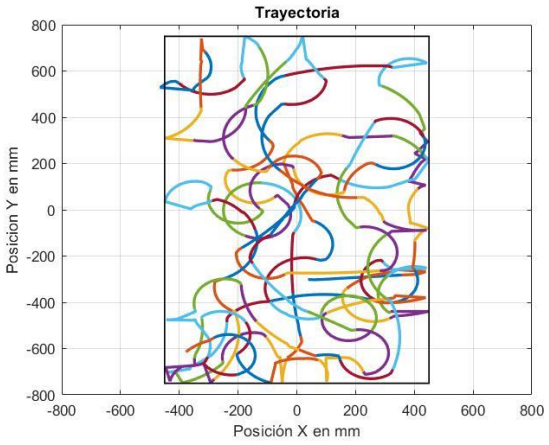
Los resultados experimentales obtenidos con modelo virtual del robot para el primer par de coordenadas muestran que en todos los casos los porcentajes de cobertura son mayores a 58%, lo que en términos generales se traduce a un desempeño aceptable.

Los resultados experimentales de las trayectorias para el segundo par de coordenadas se muestran en la Figura ***34***.



(a) Trayectoria de control.

(b) Trayectoria de combinación simple.



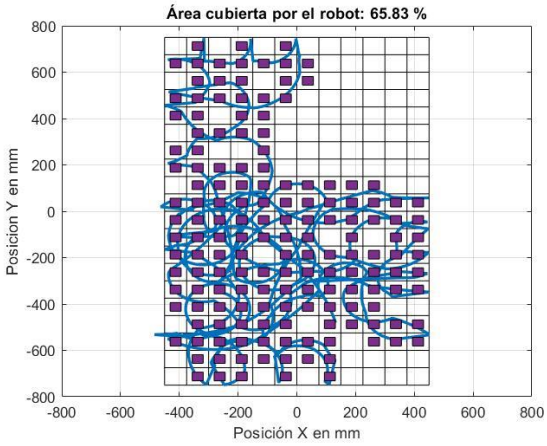
(c) Trayectoria de combinación basada en números primos.

Figura 34. Resultados experimentales para el par de coordenadas $x_r = -27.2,\ y_r = -309.6$.

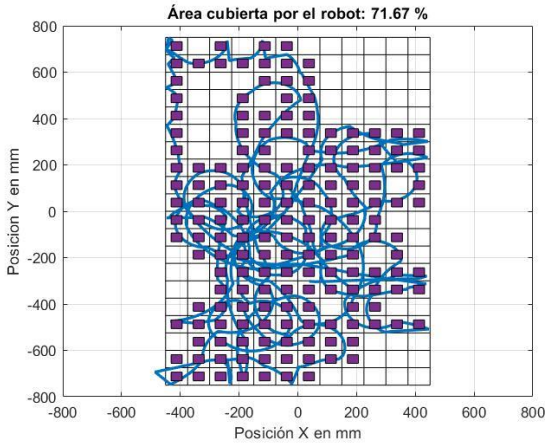
En los resultados experimentales, a diferencia de los resultados numéricos, se puede observar cómo en ciertos puntos la trayectoria se sale del área de trabajo, esto es debido a que la comunicación entre los dos programas de simulación Matlab y CoppeliaSim se realiza de manera asíncrona, es decir, los 2 trabajan en paralelo enviando información de forma bidireccional, pero ninguno de los 2 espera al otro. Por ejemplo, cuando el robot toca el borde del área de trabajo CoppeliaSim envía la posición del robot a Matlab, pero CoppeliaSim no detiene el robot. En lo que Matlab calcula el método para evitar la colisión con el borde el robot ya lo ha superado, cuando Matlab da la orden al robot que regrese este ya se

encuentra fuera de la zona permitida. Esto puede representar un serio problema si el código Matlab no está correctamente optimizado y le toma mucho tiempo a Matlab realizar los cálculos, sin embargo, factores como la capacidad de procesamiento de la computadora también pueden afectar. Una forma de evitar este problema es realizar una comunicación entre los 2 programas en modo síncrono. Sin embargo, el modo síncrono aumenta considerablemente el tiempo simulación.

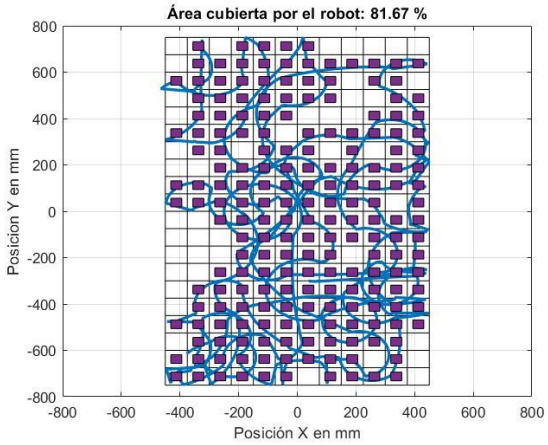
En la Figura ***35*** se muestran los porcentajes de cobertura para el segundo par de coordenadas.



(a) Trayectoria de control.



(b) Trayectoria de combinación simple.



(c) Trayectoria de combinación basada en números primos.

Figura 35. Porcentajes de cobertura de los resultados experimentales para el par de coordenadas $x_r = -27.2$, $y_r = -309.6$.

Para el segundo par de coordenadas se puede observar como la combinación de estados basada en números primos mejoro ampliamente la eficiencia de la trayectoria, aumentando el porcentaje de cobertura en un 15% con el mismo numero de iteraciones.

4.3. Conclusiones del capítulo

En este capítulo se describió el proceso de generación de trayectorias utilizando el algoritmo generador reportado en (Cetina Denis *et al.,* 2018) el cual haciendo uso de un sistema caótico induce dinámicas caóticas a un robot diferencial a través de su velocidad lineal y su velocidad angular.

Las trayectorias fueron simuladas en primera instancia de manera numérica utilizando el software Matlab donde se pudo comprobar que estas poseían las características clave de los sistemas caóticos: impredecibilidad y dependencia sensible a condiciones iniciales. Sin embargo, también se demostró que, a diferencia de otros métodos, el algoritmo generador es capaz de producir trayectorias diferentes manteniendo condiciones iniciales haciendo uso de la combinación de estados.

Los resultados numéricos fueron validados utilizando el software especializado en simulación robótica CoppeliaSim, donde se comprobó que estas trayectorias son perfectamente aplicables a un robot real sin perder ninguna de sus características.

Capítulo 5. Verificación de caos en las trayectorias y posible aplicación

En este capítulo se incluye un análisis de las trayectorias para comprobar la existencia del caos en ellas, además de proporcionar de manera numérica una posible aplicación en la detección rápida de objetos.

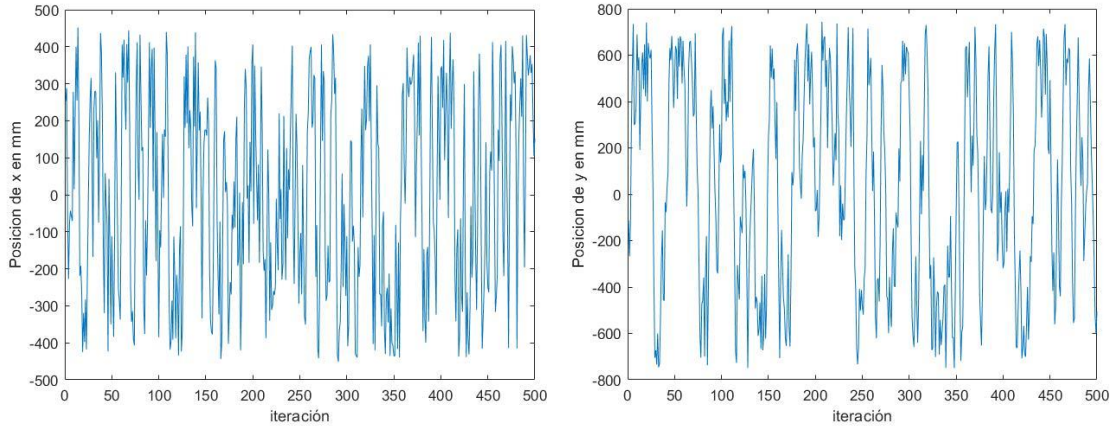
5.1. Prueba 0-1 Gottwald-Melbourne para la detección de caos en las trayectorias

Existen diferentes métodos para comprobar si un sistema dinámico es o no caótico. Por ejemplo, el cálculo del máximo exponente de Lyapunov, nos da una medida de la tasa de divergencia o convergencia exponencial de las trayectorias del sistema, un exponente de Lyapunov positivo indica la existencia de caos, sin embargo, este método puede llegar a complicarse debido a que requiere una reconstrucción espacio-fase.

Es por esto que para determinar la existencia de caos en las trayectorias mostradas anteriormente se optó por utilizar el test $0-1$ Gottwald-Melbourne presentado en (Gottwald y Melbourne, 2004). Este método, a diferencia de los exponentes de Lyapunov, trabaja directamente con las series de tiempo por lo que no es necesaria una reconstrucción espacio-fase además de que este método no toma en cuenta la dimensión o ecuaciones del sistema.

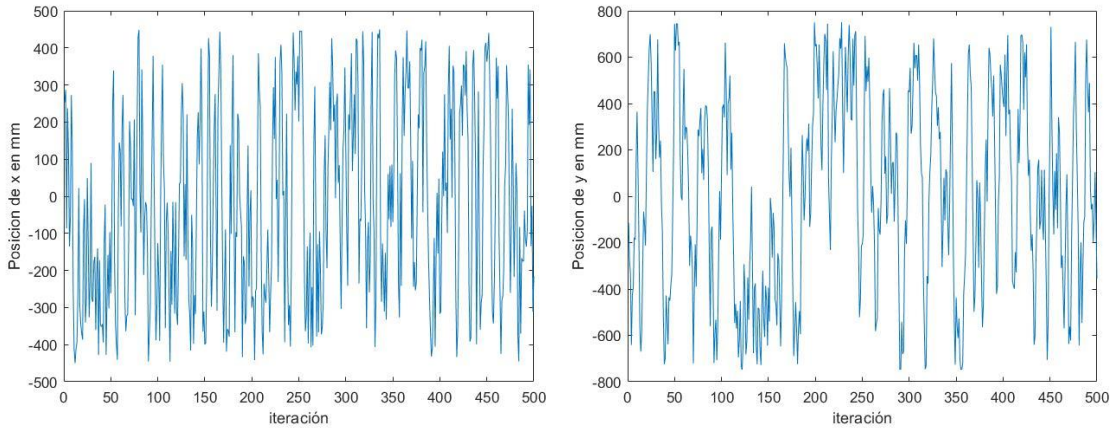
El resultado de aplicar el método $0-1$ a una serie de tiempo, como su nombre lo indica, es un valor entre 0 y 1; un resultado de 0 nos dice que la dinámica del sistema es no caótica, mientras que un resultado de 1 o muy cercano indica que la dinámica del sistema es completamente caótica.

La Prueba $0-1$ se realizó tanto para las simulaciones numéricas como para las simulaciones con el modelo virtual del robot. Para las simulaciones numéricas se utilizaron las series de tiempo de los estados x y y del modelo cinemático del robot en las trayectorias realizadas por combinación de estados simple y de números primos mostradas en la Figura ***23*** y en la Figura ***24*** respectivamente. Las series de tiempo de ambas trayectorias se muestran en la Figura ***36*** y en la Figura ***37***.



(a) Serie de tiempo de x. (b) Serie de tiempo de y.

Figura 36. Series de tiempo de los resultados numéricos con combinación de estados simple.



(a) Serie de tiempo de x. (b) Serie de tiempo de y.

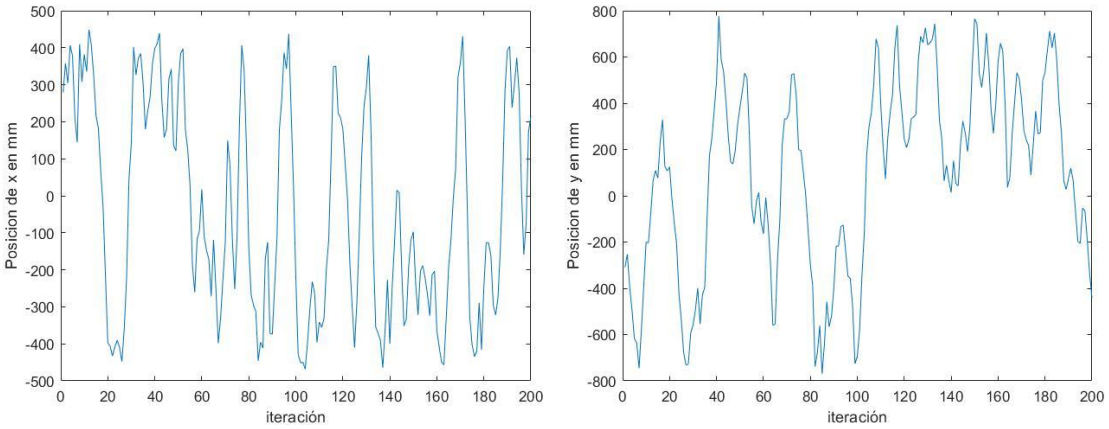
Figura 37. Series de tiempo de los resultados numéricos con combinación de estados basada en números primos.

Los resultados de la prueba $0-1$ para las simulaciones numéricas se muestran en la Tabla ***2***.

Tabla 2. Resultados de la prueba $0-1$ para la detección del caos en trayectorias con distintos métodos de combinación de estados en simulaciones numéricas.

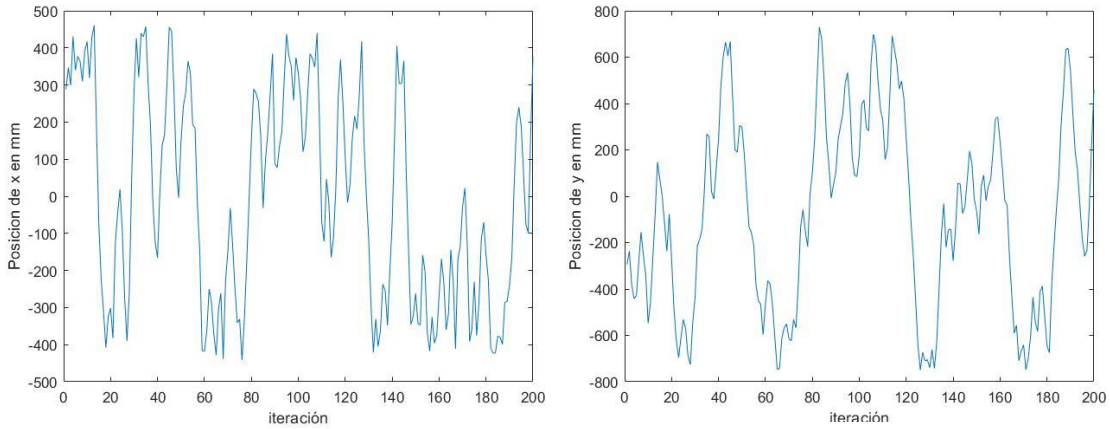
Tipo de combinación utilizada	**Resultado de la prueba para x**	**Resultado de la prueba para y**
Simple	0.9981	0.9944
Números primos	0.9974	0.9939

Para realizar la prueba $0-1$ en las simulaciones con el modelo virtual del robot se utilizaron las trayectorias mostradas en la Figura 34 (b) y Figura 34 (c) correspondiente a la combinación de estados simple y de números primos respectivamente. En la Figura ***38*** y en la Figura ***39*** se muestran las series de tiempo de las coordenadas x y y para cada trayectoria.



(a) Serie de tiempo de x. (b) Serie de tiempo de y.

Figura 38. Series de tiempo de los resultados experimentales con combinación de estados simple.



(a) Serie de tiempo de x.

(b) Serie de tiempo de y.

Figura 39. Series de tiempo de los resultados experimentales con combinación de estados basada en números primos.

Los resultados de la prueba $0-1$ para los datos experimentales se muestran en la Tabla ***3***

Tabla 3. Resultados de la prueba $0-1$ para la detección del caos en trayectorias con distintos métodos de combinación de estados en resultados experimentales.

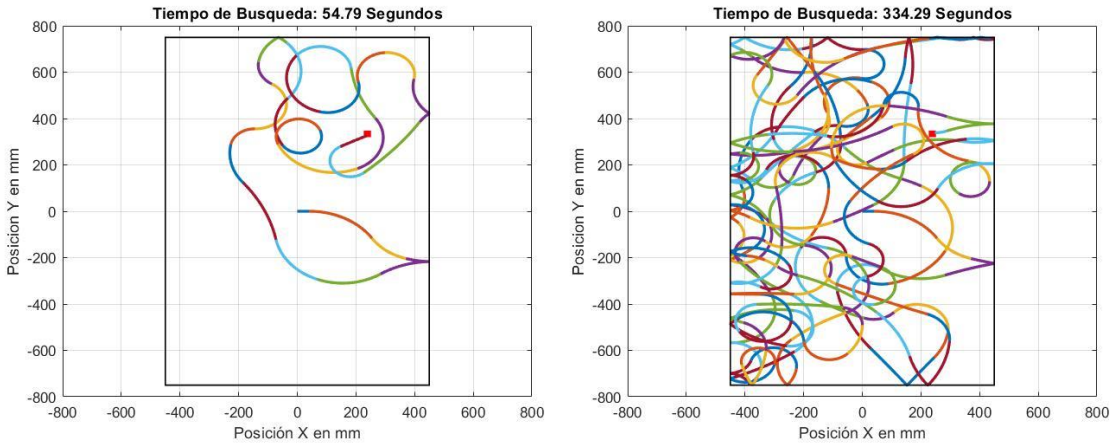
Tipo de combinación utilizada	Resultados de la prueba para x	Resultados de la prueba para y
Simple	0.9992	0.9957
Números primos	0.9897	0.9646

La cercanía a 1 de los resultados mostrados en la Tabla ***2*** y en la Tabla ***3*** son indicadores de que existe caos en las series de tiempo, por lo que se concluye que las trayectorias diseñadas en esta tesis son efectivamente caóticas.

5.2. Aplicación de trayectorias caóticas a la búsqueda de objetos en un área de trabajo

La impredecibilidad en las trayectorias es altamente deseable en aplicaciones de patrullaje y búsqueda rápida de objetos. Buscando aplicar las trayectorias generadas en este trabajo de tesis se presenta una simulación numérica enfocada en la búsqueda de objetos en el área de trabajo.

De manera arbitraria se elige la posición de un objeto (x_{obj}, y_{obj}) en el área de trabajo para ser encontrado por el robot. Se generan 2 trayectorias para el robot con condiciones iniciales $(x_r = 0, y_r = 0)$ utilizando los diferentes métodos de combinación de estados. El objetivo es mostrar el tiempo que tarda el robot en ubicar al objeto para cada tipo de trayectoria. Los resultados se muestran en la Figura ***40***, el recuadro rojo representa el objeto a encontrar.



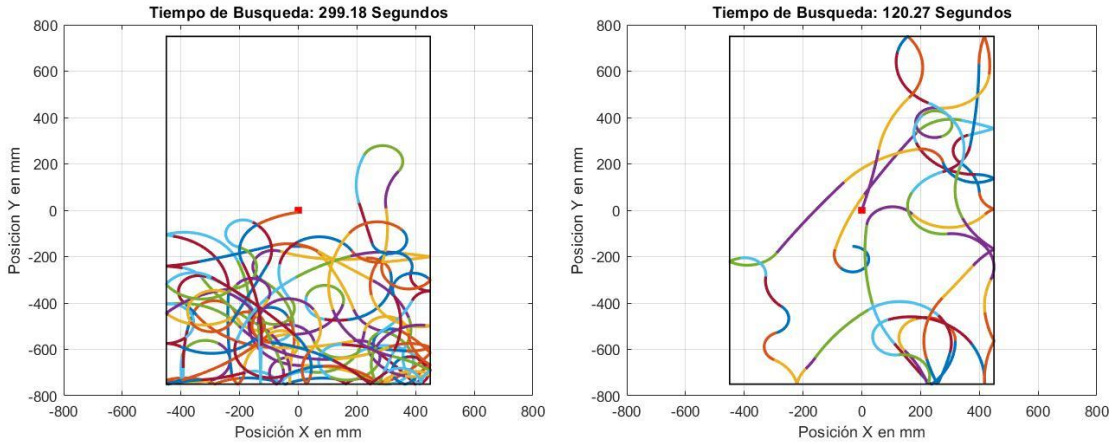
(a) (b)

Figura 40. Resultados de búsqueda con trayectorias de combinación simple (a) y de combinación basada en números primos (b) para un objeto con posición $x_{obj} = 239, y_{obj} = 334$ y condiciones iniciales $x_r = 0, y_r = 0$ para el robot.

Se observa que en el caso de la trayectoria con combinación basada en números primos le toma al robot casi 6 minutos encontrar el objeto. Sin embargo, es importante mencionar que en las simulaciones de búsqueda de objetos no se representa el tamaño real del robot Khepera y la trayectoria corresponde a un punto ubicado exactamente en el centro del robot. En la Figura ***40*** (b) Se observa que el robot pasa al menos en 3 ocasiones extremadamente cerca del objeto sin detectarlo, por lo que los tiempos de búsqueda del objeto se pueden reducir considerablemente cuando se toma en cuenta el tamaño real del

robot. Otro factor importante que influye en el tiempo de búsqueda del objeto es el ángulo de visión del robot.

Se presenta un segundo caso en el que el objeto a encontrar se ubica en el origen del área de trabajo $(x_{obj} = 0, y_{obj} = 0)$ y las condiciones iniciales del robot (x_r, y_r) son elegidas de manera arbitraria. Los resultados se muestran en la Figura ***41***.



(a) (b)

Figura 41. Resultados de búsqueda con trayectorias de combinación simple (a) y de combinación basada en números primos (b) para un objeto con posición $x_{obj} = 0, y_{obj} = 0$ y condiciones iniciales $x_r = -32, y_r = -156$ para el robot.

Se observa que para todos los casos propuestos anteriormente el robot encuentra el objeto en un tiempo relativamente aceptable.

5.3. Conclusiones del capítulo

En este capítulo se utilizó la prueba $0-1$ Gottwald-Melbourne para confirmar la presencia de caos en las distintas trayectorias presentadas en esta tesis, tanto de manera numérica como experimental.

Se presentó una posible aplicación real de las trayectorias enfocada en la búsqueda de objetos con resultados positivos ya que el robot en todos los casos fue capaz de encontrar un objeto en el área de trabajo en un tiempo aceptable.

Capítulo 6. Conclusiones

En este capítulo se presentan las conclusiones finales de este trabajo de tesis, así como el trabajo a futuro.

En esta tesis se aprovecharon las propiedades del caos para diseñar trayectorias impredecibles con diversas aplicaciones. Se utilizó el sistema caótico MACM para inducir un comportamiento caótico a un robot móvil diferencial a través de sus velocidades lineal y angular.

El algoritmo de control utilizado en esta tesis, a diferencia de otros trabajos, demostró que es posible generar trayectorias caóticas completamente diferentes manteniendo condiciones iniciales empleando combinaciones de estados.

Los distintos métodos de generación de trayectorias presentados en esta tesis fueron validados de manera experimental en un robot virtual con resultados satisfactorios. Demostrando que las trayectorias son perfectamente aplicables en situaciones reales. Por ejemplo, la búsqueda rápida de objetos en un área de trabajo.

Se demostró que todas las trayectorias mostradas poseen las características clave de los sistemas caóticos como la sensibilidad a condiciones iniciales y el determinismo, características que hacen a las trayectorias sumamente atractivas para su uso en investigación, uso civil e incluso posibles aplicaciones en el ámbito militar, empleando en cada caso un robot especifico.

Utilizando la prueba Gottwald-Melbourne se demostró de manera numérica, por medio del análisis de las series de tiempo, la existencia de caos en todas las trayectorias presentadas en esta tesis.

Con los resultados obtenidos tanto de manera numérica como experimental con el modelo virtual del robot diferencial se puede concluir que los métodos propuestos en este trabajo de tesis utilizando combinaciones de estados para la generación de trayectorias caóticas funcionan, concluyendo que se cumplieron en su totalidad los objetivos planteados en esta tesis.

6.1. Trabajo futuro

Como trabajo a futuro y enfocado en el tema expuesto en esta tesis se proponen los siguientes puntos:

- Realizar simulaciones Matlab-CoppeliaSim en modo síncrono.
- Realizar simulaciones con el robot virtual completamente en CoppeliaSim sin necesidad de una aplicación externa.
- Validar los resultados obtenidos en esta tesis en un robot Khepera III real.
- Implementar el cálculo de los exponentes de Lyapunov como una prueba adicional para la detección de caos en las trayectorias.
- Aplicar las trayectorias caóticas generadas para búsqueda rápida de objetos en otras áreas de trabajo: circular, irregular, etc.
- Emplear un grupo de robots

Literatura citada

Cetina Denis, J. J. 2017. Diseño de trayectorias caóticas en robots móviles. tesis de maestría, centro de investigacion cientifica y educacion supérior de ensenada. 70 pp.

Cetina Denis, J. J., López Gutiérrez, R. M., Arellano Delgado, A., Cruz Hernández, C. 2018. Diseño de trayectorias caóticas en robots móviles. En Congreso mexicano de robótica, Ensenada, B.C., 2018. Ensenada, B.C.

Cruz-Hernández, C., Romero-Haros, N. 2008. Communicating via synchronized time-delay chua's circuits. Communications in nonlinear science and numerical simulation, 13(3), 645–659. doi:10.1016/j.cnsns.2006.06.010

Delgado, R., Shin, W. C., Hong, H. C., Choi, W. B. 2016. Development and control of an omnidirectional mobile robot on an ethercat network. International journal of applied engineering research, 11(21), 10586–10592.

Dhaouadi, R., Hatab, A. A. 2013. Dynamic modelling of differential-drive mobile robots using lagrange and newton-euler methodologies: a unified framework. 2(2). doi:10.4172/2168-9695.1000107

Figueroa-Torres, C. A., Medina-Monroy, J. L., Lobato-Morales, H., Chavez-Pérez, R. A., Calvillo-Téllez, A. 2013. A novel fractal antenna based on the sierpinski structure for super wide-band applications. Microwave and optical technology letters, 55(11), 2562–2568. doi:10.1002/mop

Fu, Q., Li, C. 2020. Robotic modelling of snake traversing large, smooth obstacles reveals stability benefits of body compliance. Royal society, 7. doi:http://dx.doi.org/10.1098/rsos.191192

González, R., Rodríguez, F., Guzmán, J. L. 2015. Robots moviles con orugas: historia, modelado, localizacion y control. Revista iberoamericana de automática e informática industrial, 12(1), 3–12. doi:10.1016/j.riai.2014.11.001

Gottwald, G. A., Melbourne, I. 2004. A new test for chaos in deterministic systems. Proceedings of the royal society a: mathematical, physical and engineering sciences, 460(2042), 603–611. doi:10.1098/rspa.2003.1183

Helmick, D. M., Roumeliotis, S. I., Mchenry, M. C., Matthies, L. 2002. Multi-sensor, high speed autonomous stair climbing. En IEEE/RSJ international conference on intelligent robots and systems, 2002, Vol. 1, pp. 733–742. pp. 733–742.

Hilborn C. Robert. 2005. Chaos and nonlinear dynamics: an introduction for scientists and engineering (2nd ed.). Oxford University Press.

Hirose, S. 1991. Three basic types of locomotion in mobile robots. Fifth international conference on advanced robotics. Robots in unstructured environments, 1, 12–17. doi:10.1109/icar.1991.240483

Hirose, S., Morishima, A. 1990. Design and control of a mobile robot with an articulated body. The international journal of robotics research., 99–114. doi:10.1177/027836499000900208

Li, Cai-hong, Song, Y., Wang, F., Wang, Z., Li, Y. 2017. A chaotic coverage path planner for the mobile robot based on the chebyshev map for special missions. Frontiers of information technology & electronic engineering, 18(9), 1305–1319.

Li, Chen. 2020. Terradynamics lab johns hopkins university. Consultado el 16 de diciembre de 2021, de https://li.me.jhu.edu/

Lorenz, E. N. 1963. Deterministic nonperiodic flow. Journal of the atmospheric sciences, 20, 130–141. doi:10.1201/9780203734636-38

Majeed, I. A. 2020. Mobile robot motion control based on chaotic trajectory generation. Journal of engineering and sustainable development, 24(04), 48–55. doi:doi.org/10.31272/jeasd.24.4.6

Marjovi, A., Nunes, J. G., Marques, L., De Almeida, A. 2009. Multi-robot exploration and fire searching. IEEE/RSJ International conference on intelligent robots and systems, 1929–1934. doi:10.1109/IROS.2009.5354598

Méndez-Ramírez, R., Arellano-Delgado, A., Cruz-Hernández, C., Martínez-Clark, R. 2017. A new simple chaotic lorenz-type system and its digital realization using a tft touch-screen display embedded system. 2017, 13. doi:https://doi.org/10.1155/2017/6820492 Research

Murillo-Escobar, M. A., Cruz-Hernández, C., Abundiz-Pérez, F., López-Gutiérrez, R. M., Acosta Del Campo, O. R. 2015. A rgb image encryption algorithm based on total plain image characteristics and chaos. Signal processing, 109, 119–131. doi:10.1016/j.sigpro.2014.10.033

NASA JPL. 2002. Mars exploration rovers. Consultado el 8 de diciembre de 2021, de https://mars.nasa.gov/mer/multimedia/images/?t=242&&start=351

NOAA. 2016. What is an auv? Consultado el 25 de octubre de 2021, de https://oceanexplorer.noaa.gov/facts/auv.html

Ollero Baturone, A. 2001. Robótica, manipuladores y robots móviles (1a ed.). S.A. Marcombo: Barcelona, España.

Papcun, P., Zolotova, I., Tafsi, K. 2016. Control and teleoperation of robot khepera via android mobile device through bluetooth and wifi. IFAC-PapersOnLine, 49(25), 188–193. doi:10.1016/j.ifacol.2016.12.032

Petavratzis, E. K., Volos, C., Moysis, L., Stouboulos, I. N. 2019. An inverse pheromone approach in a chaotic mobile robot's path planning based on a modified logistic map. Technologies, 18–33. doi:10.3390/technologies7040084

Prorok, A., Arfire, A., Bahr, A., Farserotu, J. R., Martinoli, A. 2010. Indoor navigation research with the

khepera III mobile robot: an experimental baseline with a case-study on ultra-wideband positioning. International conference on indoor positioning and Indoor navigation, 15–17. doi:10.1109/IPIN.2010.5647880

Schidt, L., Buch, B., Burger, B., Chang, S., Otto, J. A., Seifert, U. 2015. Khepera III mobile robot – practical aspects.

USAF. 2019. MQ-9 reaper. Consultado el 16 de noviembre de 2021, de https://www.af.mil/About-Us/Fact-Sheets/Display/Article/104470/mq-9-reaper/

Volos, C. K., Kyprianidis, I. M., Stouboulos, I. N. 2012. Motion control of robots using a chaotic truly random bits generator. Journal of engineering science and technology review, 5, 6–11.

Wynn, R. B., Huvenne, V. A. I., Le, T. P., Murton, B. J., Connelly, D. P., Bett, B. J., Ruhl, H. A., Morris, K. J., Peakall, J., Parsons, D. R., Sumner, E. J., Darby, S. E., Dorrell, R. M., Hunt, J. E. 2014. Autonomous underwater vehicles (auv's): their past, present and future contributions to the advancement of marine geoscience. Marine geology. doi:10.1016/j.margeo.2014.03.012

.

Anexos

Anexo A. Código MATLAB para la creación de trayectorias caóticas

```
clear all

close all

clc



% Parametros MACM

tao=0.005;

a=2;

b=2;

c=0.5;

d=4;



% Posiciones iniciales (x,y) del robot Khepera

x(1)=0;

y(1)=0;

th(1)=0;  % Angulo inicial del robot



% Condiciones iniciales de MACM utilizando posición (x,y) de Khepera

xmacm=1;

ymacm=x(1);

zmacm=y(1);

tic
```

```
% Empiezan las iteraciones


for n=1:100
% Se calculan los estados de MACM
    xmacm(n+1)=xmacm(n)+tao*(-a*xmacm(n)-b*ymacm(n)*zmacm(n));
    ymacm(n+1)=ymacm(n)+tao*(-xmacm(n)+c*ymacm(n));
    zmacm(n+1)=zmacm(n)+tao*(d-ymacm(n).^2-zmacm(n));


% Los estados de MACM se asignan como velocidades v y w


    v(n)=xmacm(n+1);
    w(n)=ymacm(n+1);


% Se calculan las velocidades de cada rueda a partir de v y w
    vi(n)=v(n)-(90*w(n));
    vd(n)=v(n)+(90*w(n));


% Se ajustan las velocidades a los máximos y mínimos de Khepera
    vdi(n)=floor(((vi(n)*1000)-floor(vi(n)*1000))*180)+20;
    vdd(n)=floor(((vd(n)*1000)-floor(vd(n)*1000))*180)+20;


% El robot se mueve utilizando todos los datos anteriores
```

```
    for m=1:200


    x(m+1)=x(m)+((vdi(n)+vdd(n))/2)*0.01*cos(th(m));
    y(m+1)=y(m)+((vdi(n)+vdd(n))/2)*0.01*sin(th(m));
    th(m+1)=th(m)+((vdd(n)-vdi(n))/90)*0.01;


%--------------------------------------------------------------------------


% Crea límites para el robot método reflexivo


   if x(m+1)>450
       th(m+1)=((2*pi)-th(m+1))-pi;
   end


   if x(m+1)<-450
       th(m+1)=((2*pi)-th(m+1))-pi;
   end


  if y(m+1)>750
       th(m+1)=(2*pi)-th(m+1);
   end


  if y(m+1)<-750
```

```
        th(m+1)=(2*pi)-th(m+1);

  end

%--------------------------------------------------------------------------


    end % finaliza el for 200


% Se grafica

set(0,'defaultfigurecolor',[1 1 1])

figure (1)

   plot(x,y,'linewidth',2)

   ylabel('Posicion Y en mm')

   xlabel('Posicion X en mm')

   hold on

   grid on

   title('Trayectoria')

  rectangle('Position',[-450 -750 900 1500],'LineWidth',1)


ylim([-800 800]);

xlim([-800 800]);


% Nuevas posiciones del robot

   x(1)=x(m+1);

   y(1)=y(m+1);
```

```
  th(1)=th(m+1);


end
```

Anexo B. Código de comunicación Matlab-CoppeliaSim

```
% Simulacion Khepera Coppelia

clear all

close all

clc

sim=remApi('remoteApi');

sim.simxFinish(-1); % just in case, close all opened connections

clientID=sim.simxStart('127.0.0.1',19999,true,true,5000,5);



tao=0.005;

a=2;

b=2;

c=0.5;

d=4;



% Inicializa la conexión Coppelia-Matlab además de crear los handles

if (clientID>-1)

    disp('Connected to remote API server');

% Handle


[returnCode,left_Motor]=sim.simxGetObjectHandle(clientID,'K3_leftWheelMotor',s
im.simx_opmode_blocking);


[returnCode,right_Motor]=sim.simxGetObjectHandle(clientID,'K3_rightWheelMotor'
,sim.simx_opmode_blocking);
```

```
[returnCode,referencia]=sim.simxGetObjectHandle(clientID,'Referencia',sim.simx
_opmode_blocking);


[returnCode,sensor_frontal]=sim.simxGetObjectHandle(clientID,'K3_ultrasonicSen
sor3',sim.simx_opmode_blocking);



     % Sensor de proximidad


[returnCode,detectionState,detectedPoint,~,~]=sim.simxReadProximitySensor(clie
ntID,sensor_frontal,sim.simx_opmode_streaming);



     % Se obtienen posición X y Y del robot Khepera

   [returnCode,posicion]=sim.simxGetObjectPosition(clientID,referencia,-
1,sim.simx_opmode_streaming);



% Orientación del robot

   [returnCode,orientacion]=sim.simxGetObjectOrientation(clientID,referencia,-
1,sim.simx_opmode_streaming);

% Posiciones iniciales (x,y) del robot Khepera

 x(1)=posicion(1);

 y(1)=posicion(2);



% Condiciones iniciales de MACM utilizando posición (x,y) de Khepera

% Utilizando estados Y y Z

xmacm=1;

ymacm=x(1);

zmacm=y(1);
```

```
for n=1:200

    % Calculo de estados y velocidades

    % Se calculan los estados de MACM

    xmacm(n+1)=xmacm(n)+tao*(-a*xmacm(n)-b*ymacm(n)*zmacm(n));

    ymacm(n+1)=ymacm(n)+tao*(-xmacm(n)+c*ymacm(n));

    zmacm(n+1)=zmacm(n)+tao*(d-ymacm(n).^2-zmacm(n));


    % Los estados de MACM se asignan como velocidades v y w


%     v(n)=xmacm(n+1);

%     w(n)=ymacm(n+1);


    % Se calculan las velocidades de cada rueda a partir de v y w

    vi(n)=v(n)-(90*w(n));

    vd(n)=v(n)+(90*w(n));


    % Se ajustan las velocidades a los máximos y mínimos de Khepera

    vdi(n)=floor(((vi(n)*1000)-floor(vi(n)*1000))*180)+20;

    vdd(n)=floor(((vd(n)*1000)-floor(vd(n)*1000))*180)+20;


    % Convertimos las velocidades a radianes/seg

    % Se convierten las velocidades a m/s

    velocidadrueda1(n)=vdi(n)/1000; %velocidad en m/s
```

```
    velocidadrueda2(n)=vdd(n)/1000; %velocidad en m/s


    radio=20.5;  % Radio de las ruedas en mm


    % Se convierte el radio a m
    radiom=radio/1000;


    % velocidad angular


    nvdi(n)=velocidadrueda1(n)/radiom;
    nvdd(n)=velocidadrueda2(n)/radiom;


    % Movimiento del robot


    tic
 for i=1:19    %tiempo que se envian las velocidades del robot


   % Se envían las velocidades al robot

[returnCode]=sim.simxSetJointTargetVelocity(clientID,right_Motor,nvdd(n),sim.s
imx_opmode_blocking);

[returnCode]=sim.simxSetJointTargetVelocity(clientID,left_Motor,nvdi(n),sim.si
mx_opmode_blocking);


   % Se obtienen las posiciones X y Y además de la orientación
```

```
    [returnCode,posicion]=sim.simxGetObjectPosition(clientID,referencia,-
1,sim.simx_opmode_streaming);


[returnCode,orientacion]=sim.simxGetObjectOrientation(clientID,referencia,-
1,sim.simx_opmode_buffer);



 end

  tiempo(n)=toc;



end



  % Detiene el robot y obtiene la posición final


[returnCode]=sim.simxSetJointTargetVelocity(clientID,left_Motor,0,sim.simx_opm
ode_blocking);


[returnCode]=sim.simxSetJointTargetVelocity(clientID,right_Motor,0,sim.simx_op
mode_blocking);



  [returnCode,orientacion]=sim.simxGetObjectOrientation(clientID,referencia,-
1,sim.simx_opmode_buffer);

  [returnCode,posicion]=sim.simxGetObjectPosition(clientID,referencia,-
1,sim.simx_opmode_streaming);

 end

   sim.simxFinish(clientID)

   sim.delete(); % call the destructor!



   disp('Program ended');
```