UNIVERSIDAD VERACRUZANA

Maestría en Ingeniería Electrónica y Computación

"Implementación de la Metodología de planificación de Rutas mediante Redes Resistivas en un Robot de Dirección Ackermann"

Tesis presentada por

Ing. Martin Carballo Flores

como requisito

para obtener el título de Maestro

en Ingeniería Electrónica y Computación

Xalapa Enríquez, Veracruz. Primavera de 2021

UNIVERSIDAD VERACRUZANA

Maestría en Ingeniería Electrónica y Computación

"Implementación de la Metodología de planificación de Rutas mediante Redes Resistivas en un Robot de Dirección Ackermann"

Tesis profesional presentada por

Ing. Martin Carballo Flores

como requisito parcial
para obtener el título de Maestro
en Ingeniería Electrónica y Computación

Supervisada por:

Dr. Roberto Castañeda Sheissa
Facultad de Instrumentación Electrónica

Dr. Héctor Vázquez Leal
Facultad de Instrumentación Electrónica

Xalapa Enríquez, Veracruz. Enero de 2021

Índice general

Capítulo 1

Introducción

La robótica está en continuo crecimiento y cada vez cobra mayor relevancia, sobre todo la robótica móvil, en diferentes industrias como la agrícola, restaurantera y la manufacturera, en la cual se ha establecido desde hace décadas con los robots manipuladores, pero que ahora con la robótica móvil se habren más posibilidades en las industrias mencionadas, cambiando la forma en que se encaminan los esfuerzos a futuro y el lugar en el que se visualiza al ser humano dentro de todo este esquema.

Así también como en la industria, gracias al desarrollo de la robótica en general, se puede avanzar y retomar investigaciones en áreas de la ciencia como la exploración marítima, arqueológica y la exploración espacial, en la que ha sido de vital importancia desde sus inicios.

Éste continuo aumento en la demanda de soluciones, conlleva a que a futuro la robótica sea la única alternativa en las industrias y se abandone la opcion de manipulación humana debido a aspectos de optimización de producción y seguridad del personal; y es precisamente por esto que en la industria automotriz se espera que en las próximas décadas haya un aumento exponencial en la producción de vehículos autónomos y semi-autónomos hasta que se cambie por completo la manufactura a vehículos autónomos, entrando dentro de la robótica móvil que se describe en la siguiente sección.

1.1. Robótica Móvil

La robótica móvil es el área de la robótica que engloba a los robots que cuentan con una completa autonomía para desplazarse mediante la aplicación de fuerza al entorno en el que se encuentran, a diferencia de los manipuladores, como lo son los brazos robóticos que están sujetos a una base de la cual no se pueden separar, es decir, están sujetos al entorno y aplican fuerza a los objetos que manipulan. De hecho, actualmente hay un debate entre la comunidad de la robótica sobre dejar de considerar robots a los dispositivos que no cumplan con las características de la robótica móvil.

Aunque haya algunos desacuerdos en la comunidad de la robótica sobre las clasificación de los robots, se pueden identificar las características que se han establecido para que un robot sea incluído en la robótica móvil, las cuales son locomoción, cinemática, localización, planificación y navegación, [1].

1.1.1. Locomoción

Locomoción es la aplicación de mecanismos que permiten que un robot pueda moverse a través de su entorno, [1]. Estos mecanismos son muy diversos y han sido ideados a partir de los mecanismos de locomoción se pueden encontrar en la naturaleza. Los mecanismos más comunes en la robótica actualmente son los basados en ruedas, ya que mediante estas que obtiene mayor estabilidad, en primer lugar porque trabajan en ambientes hechos por el hombre, que son menos complejos que los ambientes presentes en la naturaleza, para los cuales los mecanismos basados en piernas se pueden desempeñar mejor, además de que la estructura de los robots con ruedas contribuye a que tengan mayor estabilidad dinámica y estática, gracias a la ubicación del centro de gravedad y la cantidad de puntos de contacto. Dentro de los mecanismos de locomoción basados en ruedas se pueden encontrar una gran cantidad de configuraciones, como por ejemplo la configuración Ackermann empleada en automóviles.

1.1.2. Cinemática

Para obtener un modelo que describa la cinemática de un robot es necesario tomar en cuenta la aportación de fuerza de cada uno de sus actuadores, los acuales a su vez aportan restricciones de movimiento al robot. Basándose en la geometría del robot se obtiene el movimiento total del robot en función de todas las aportaciones en movimiento y en las restricciones de todos los actuadores, [1].

Para especificar la posición del robot, se le ve como un objeto coordenado con un marco de referencia local R(ϕ). Si se recorre una trayectoria $Y = F(X)$ con una rotación con respecto al eje Z del marco de referencia global ξ se puede calcular el vector de coordenadas r_ξ de un punto sobre el robot mediante la siguiente ecuación, donde la matriz de rotación R_Z es llamada tambien matriz de cosenos de dirección del vector resultante $\mathbf{r}_{R(\phi)}$ con respecto al marco de referencia global, 1.1.

$$\mathbf{r}_\xi = R_Z \mathbf{r}_{\mathbf{R}(\phi)} \tag{1.1}$$

$$R_Z = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1.2}$$

$$r_\xi = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{1.3}$$

$$\mathbf{r}_{R(\phi)} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{1.4}$$

Una vez establecidos los marcos de referencia se procede a definir el espacio de configuración, que es la forma en que el robot interpreta al mundo y de cómo se ve al robot dentro del espacio de configuración. Para hacerlo, es necesario tomar en cuenta las restricciones de movimiento del robot, como lo son las restricciones en movimiento de los actuadores, así como aquellas debido a la topología del robot.

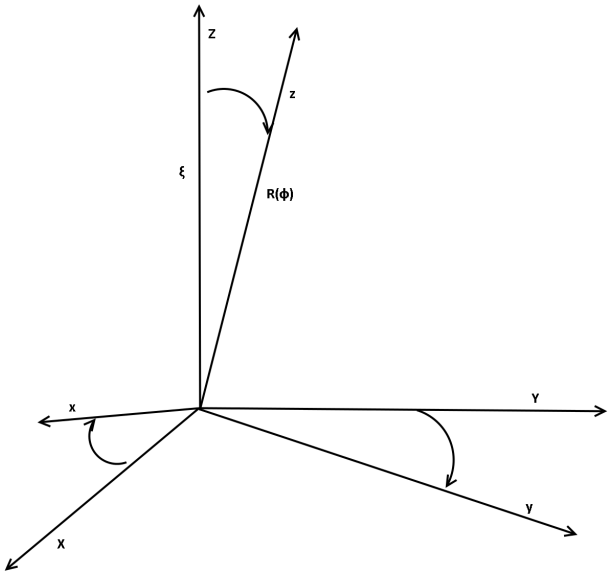

Figura 1.1: Marcos de referencia en robótica.

1.1.3. Percepción

En robótica móvil es de vital importancia la percepción, ya que permite reconstruir el entorno que rodea al robot y así poder evitar colisiones. Se lleva a cabo mediante la interpretación de los datos adquiridos por los sensores equipados. Se toman los datos de los sensores para que a partir de estos se puedan extraer caracteristicas del entorno, como bordes y obstáculos, que en su conjunto conforman el escenario en el que se encuentre el robot. La extracción de características no es algo forzoso, ya que se puede establecer la relación del robot con el entorno, asignando diferentes niveles de percepción para cada sensor.

Los sensores se clasifican de la siguiente manera: prioceptivos, encargados de medir variables internas del robot, como lo es la aceleración de un motor; exteroceptivos, que miden las variables del medio; pasivos, que se limitan a medir energía proveniente del medio; y activos, los cuales emiten una señal de excitación al medio y miden la respuesta.

La percepción es una de las tareas más dificiles de realizar, ya que todos los sensores cuentan con cierto grado de incertidumbre, por lo tanto, es necesario tomar datos de más de un sensor y así se pueda tener una interpretación más cercana al mundo real.

1.1.4. Localización

El problema de la localizacion surge debido a la incertidumbre de los sensores, lo cual aporta ruido a las lecturas; además, si se toma en cuenta el grado de incertidumbre en los actuadores, se tiene que el error en la estimación de la posición aumenta conforme avanza el robot. Por ello es que se aplican diversas técnicas para procesar los datos adquiridos y obtener una mejor estimación de la posición, algunas basadas en probabilidades y otras que en cambio manejan multiples hipótesis para la estimación de la posición.

1.1.5. Navegación y Planificación

Navegación consiste en la habilidad de usar la informacion adquirida del entorno para que el robot pueda alcanzar sus objetivos de forma eficiente y confiable. Se requieren dos acciones vitales para lograr el objetivo de la navegación: Planificación de trayectorias y evación de obstáculos [1].

1.2. Planificación de rutas

La planificación de rutas para robots móviles ha sido un problema abordado desde los inicios de ésta área. Se ha desarrollado una amplia variedad de algoritmos para la planificación de rutas, de los cuales destacan dos vertientes: los métodos probabilísticos y los métodos determinísticos. Dentro de los métodos probabilísticos se encuentran: Exploración de Árboles Aleatórios (RRT) y Mapas de Caminos Probabilísticos (PRM). Dentro de los algoritmos de tipo determinístico se encuentran la metodología de Planificación de rutas mediante redes resistivas (RGPPM) desarrollada en [17] y los basados en Campos de Potencial Artificial, propuestos en [16].

1.2.1. Métodos Probabilísticos

Los métodos probabilísticos generan rutas basándose en funciones aleatorias. La desventaja de estos métodos es que no se tiene control sobre la forma que tendrán los caminos generados, lo que implica que sean ineficientes para lugares estrechos, para los que necesitan generar una mayor cantidad de puntos aleatorios, y así cubrir dichos espacios o iterar aun más, para el caso de exploración de mapas probabilísticos.

La metodología de mapas de caminos probabilísticos inicia generando una determinada cantidad de puntos aleatorios alrededor de todo el espacio de configuración, después procede a unir los puntos entre sí y calcula cuál es el camino que recorre menos conexiones para llegar al objetivo (Figura 1.2).

La metodología de exploración de árboles aleatorios genera caminos aleatorios partiendo del origen o incluso del destino, cada iteración agrega nuevas ramas al árbol de caminos hasta que una rama este lo suficientemente cerca del destino según el margen de incertidumbre que se tenga [4].

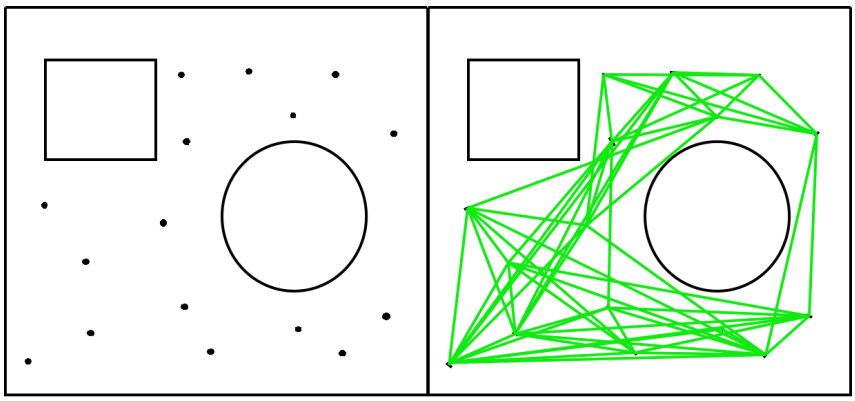

Figura 1.2: Metodología PRM.

1.2.2. Métodos Determinísticos

Los métodos determinísticos no involucran elementos aleatorios en sus procesos, ya que realizan cálculos y resuelven sistemas de ecuaciones para construir las rutas.

Un método muy usado anteriormente es la planificaación de rutas mediante campos

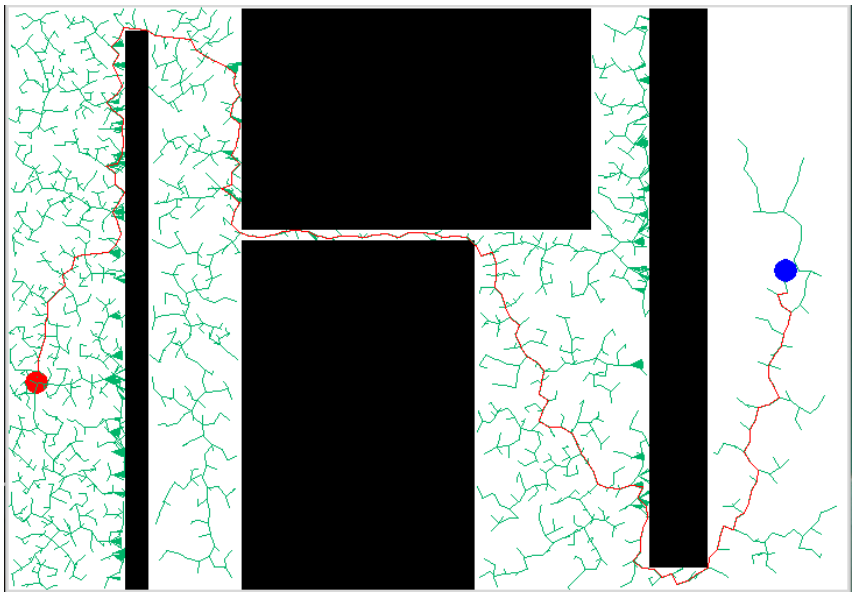

Figura 1.3: Metodología RRT.

de potencial artificial [5]. En ésta metodología se modela el ambiente como un campo de potencial electromagnético en el cual la posición del robot se representa como una partícula de carga positiva y al destino se le representa como una partícula con carga de sentido contrario, Figura 1.4.

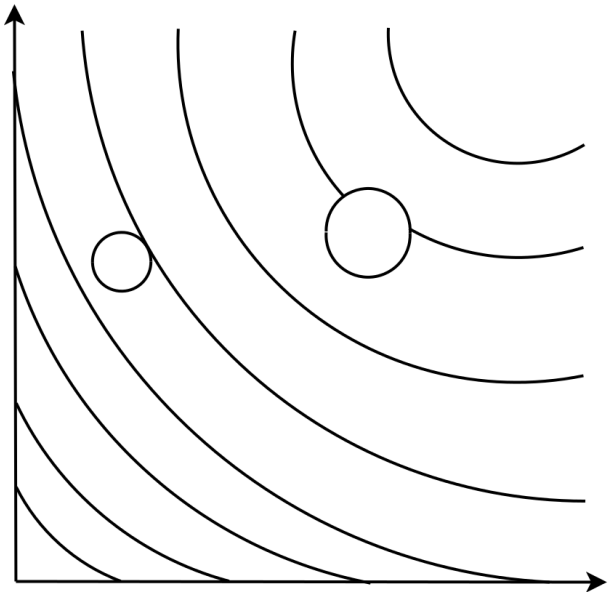

Figura 1.4: Campo de optencial artificial [5].

Los obstáculos son modelados como cargas de mismo signo y amplitud que la carga del

origen y así es como se puede llevar a cabo la evasión de obstáculos con esta metodología. Sin embargo, con éste método surgen problemas cuando la ruta avanzaba hacia espacios rodeados por cargas de signo contrario, haciendo imposible avanzar o retroceder, ya que retroceder implicaría ir en contra de la diferencia de potencial, entrando en lo que se conoce como minimo local [12].

La pendiente generada por el campo de potencial simulado, que impide que la ruta salga de la zona bloqueada, se podía eliminar recalculando el campo, pero ahora usando la última ubicación como nuevo origen se genera una diferencia de potencial que lleva a la trayectoria hacia afuera de la zona bloqueada.

1.3. Objetivo de la Tesis

1.3.1. Hipótesis

El presente trabajo de tesis plantea la hipótesis de que es posible acoplar la técnica de planificacion de rutas mediante redes resistivas RGPP con el modelo matemático de robots de dirección Ackermann con la finalidad de planificar movimientos robóticos, considerando las restricciones asociadas a robots de éste tipo.

1.3.2. Metodología

1. Revisión del estado del arte en planificación de rutas en conducción autónoma.

2. Experimentación con la generación de rutas generadas con RGPPM mediante simulación.

3. Aplicar una técnica de optimización de trayectorias a las rutas generadas con RGPP para su aplicación en el robot de dirección Ackerman.

4. Desarrollo de prototipo robótico.

5. Pruebas experimentales y validación de la metodología propuesta.

1.3.3. Objetivo General

Éste proyecto tiene como objetivo realizar una implementación de la metodología RGPP para la generación y seguimiento de rutas libres de colisiones en un robot de dirección Ackerman, tomando en cuenta las restricciones en movimiento del modelo Ackerman.

Capítulo 2

Robot Ackerman

Desde hace décadas se han dedicado muchos trabajos sobre conducción autónoma, en los que se ha probado la efectividad de diversos métodos, ya sean basados en redes neuronales como [19], enfocándose en la toma de decisiones basado en un conjunto de datos tomados de conductores humanos altamente calificados. En [20] se le da un enfoque predictivo a la planificación, tomando en cuenta el comportamiento de los vehículos cercanos y predecir su comportamiento basándose en los puntos de quiebre Bayesianos tomados del historial de los automóviles hasta el momento actual y, con base en esto, elegir una ruta segura.

En [6] y [7] se realizan aportaciones en cuanto a la planeación de trayectoria mediante la identificación de patrones en imágenes provenientes de un arreglo de cámaras, posterior a ello se obtiene una linea virtual que el automóvil debe seguir para avanzar por la autopista. Por otro lado en [8] y [10] se prueba la efectividad del uso de lasers para la detección de obstáculos, lo cual no indica que se deba excluir el uso de un tipo de sensor sobre otro, sino que cada técnica y diferentes sensores pueden acoplarse para conseguir un sistema más robusto. En el siguiente apartado se hablará sobre el tema de planificación de rutas en vehículos autónomos.

2.1. Planificación de rutas en conducción autónoma

Los métodos de planificación de rutas en conducción autónoma se dividen en metodologías incrementales y locales, ubicando dentro de las incrementales a la metodoloía RRT mencionada anteriormente y la metodología de planificacion en reja [3]. La planificación local de rutas se plantea como una alternativa de solución al alto costo computacional que conlleva el cálculo de la ruta completa debido a la exploración completa del espacio entre el origen y el destino; dado que son técnicas muy simples, se usan más como técnicas de apoyo y no son confiables para ambientes complejos.

2.1.1. Planificación incremental de rutas

La metodología de planificación con patrones de reja en la Figura 2.1 se basa en la discretización del espacio de configuraciones en celdas de estado en las que se incluye la posición X, Y, el ángulo de dirección y se apoya en otras metodologías de planificación de rutas como el método de A*.

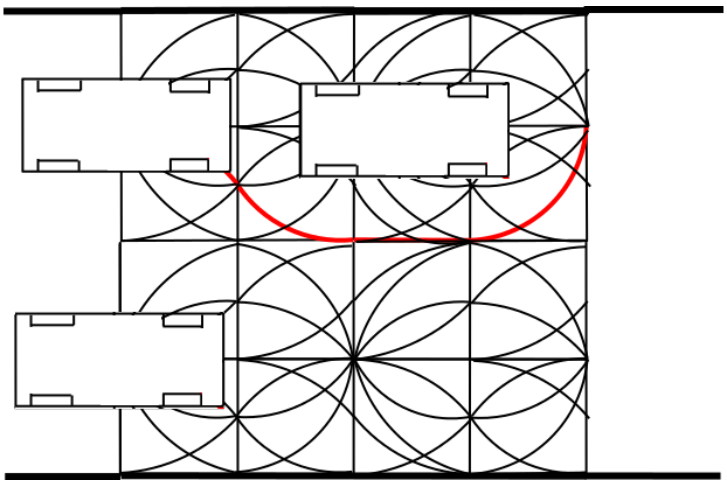

Figura 2.1: Espacio de configuraciónes en planificación con patron de reja [3].

Como se puede observar, las rutas que se generan con éste tipo de planificación se ajustan muy bien al tipo de trayectoria que puede seguir un automóvil, sin embargo, en [9] se obtuvo que la discretización del ángulo de rumbo puede llevar a tener rutas

discontinuas, lo cual se evita en [3] en la que se estableció un modelado del espacio de configuraciones dependiente del tiempo y ya no dividido en celdas discretas.

2.1.2. Planificación local de rutas

Se puede encontrar en la literatura con una técnica muy usada para planificación de movimientos dentro de conducción autónoma en carretera que se basa en determinar el grado de desplazamiento lateral que se realizará, Figura 2.2.

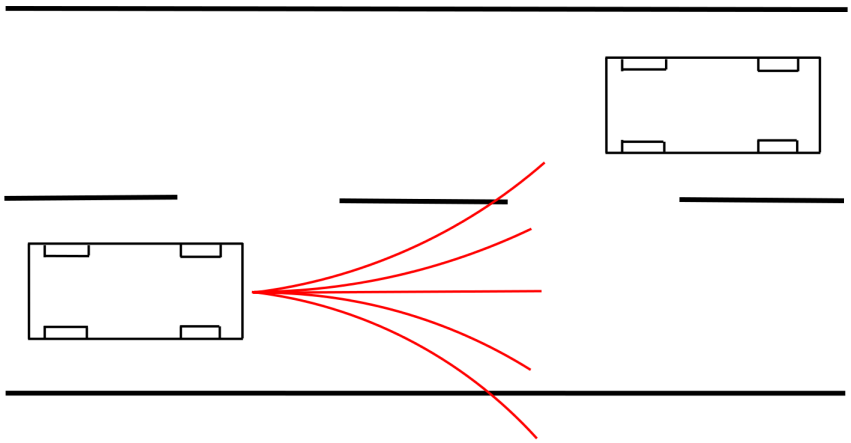

Figura 2.2: Técnica de desplazamientos laterales.

Otra técnica de planificación local es la de planificación parcial de movimiento (PMP), desarrollado por [11], cuya base es RRT y el uso del Estado de Colisión Inevitable (ICS). Cuando un estado de colisión inevitable es determinado se busca una nueva alternativa de ruta local.

2.2. Seguimiento de trayectoria

La estructura básica de los automóviles es conocida como modelo Ackerman, la cual consta de cuatro ruedas, las ruedas traseras aportan tracción mientras que las delanteras se encargan de la dirección, Figura 2.3. Los automóviles y los robots que comparten éste mismo modelo tienen la característica de que a bajas velocidades, no presentan derrape al girar, característica conocida como condición Ackerman descrita por la ecuación 2.1,

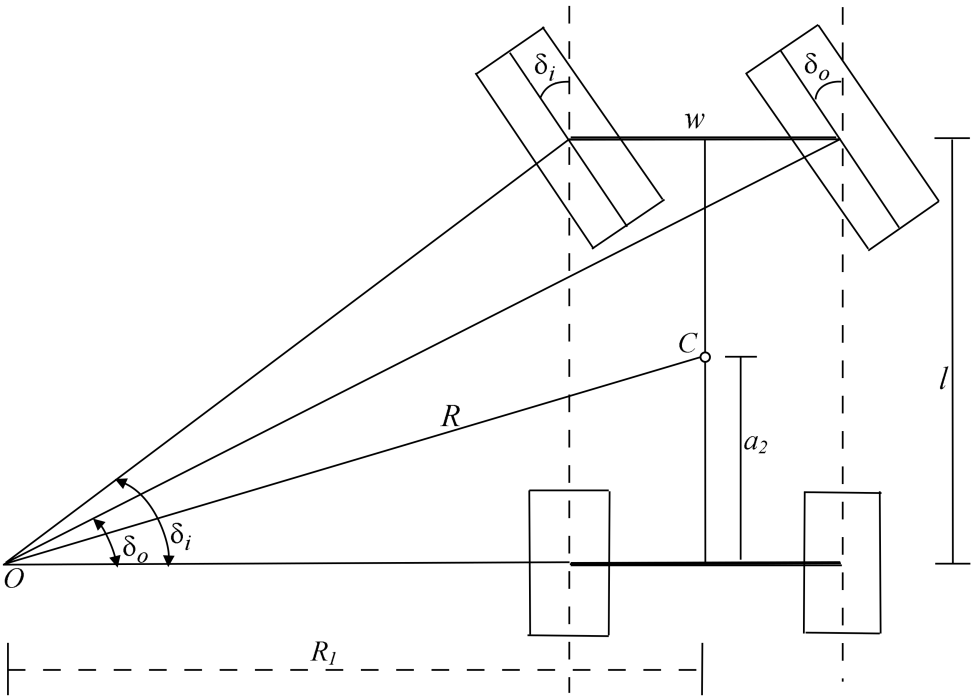

Figura 2.3: Modelo Ackerman.

donde se relaciona a ω y l, llamados ancho y longitud cinemáticos, con los ángulos interior (δ_i) y exterior (δ_o) de las ruedas delanteras de la Figura 2.3.

$$\cot \delta_o - \cot \delta_i = \frac{\omega}{l} \tag{2.1}$$

Como se puede ver, estando bajo la condición Ackerman, el radio R en el que gira el centro de masa C es obtenido mediante un cálculo de hipotenusa en la ecuación 2.2.

$$R = \sqrt{a_2^2 + l^2 \cot^2 \delta} \tag{2.2}$$

De la cual se tiene que

$$\delta = \frac{\cot \delta_o + \cot \delta_i}{2} \tag{2.3}$$

O del modelo equivalente de la bicicleta

$$\cot \delta = \frac{R_1}{l} \tag{2.4}$$

De la geometría de la Figura 2.3 se relaciona a los ángulos de dirección de las ruedas delanteras con el radio $R1$ mediante las ecuaciones 2.5 y 2.6.

$$\tan \delta_i = \frac{l}{R1 - \frac{\omega}{2}} \tag{2.5}$$

$$\tan \delta_o = \frac{l}{R1 - \frac{\omega}{2}} \tag{2.6}$$

Al realizar la planificación de una trayectoria para un robot con dirección Ackerman también es necesario tomar en cuenta el espacio de giro necesario el cual condiciona la evasión de obstáculos, además de las variables anteriormente mencionadas.

En la Figura 2.4 se muestra el espacio requerido para el giro del robot, se puede observar un radio mínimo R_{min}, que va del centro O de la curva al centro de la rueda trasera interior, así como también se muestra el radio máximo de movimiento R_{max}, cuya longitud se extiende desde el centro de giro O hasta la esquina delantera exterior del robot u automóvil.

El radio máximo de giro esta definido por

$$R_{max} = \sqrt{(R_{min} + \omega)^2 + (l + g)^2} \tag{2.7}$$

Por lo tanto, el robot se moverá en un anillo de ancho ΔR igual a

$$\Delta R = R_{max} - R_{min} = \sqrt{(R_{min} + \omega)^2 + (l + g)^2} - R_{min} \tag{2.8}$$

Sustituyendo R_{min} por

$$R_{min} = R1 - \frac{1}{2}\omega = \frac{l}{\tan \delta_i} = \frac{l}{\tan \delta_o} - \omega \tag{2.9}$$

Se tiene que el espacio necesario para girar es igual a

$$\Delta R = \sqrt{(\frac{l}{\tan \delta_i} + 2\omega)^2 + (l + g)^2} - \frac{l}{\tan \delta_i} = \sqrt{(\frac{l}{\tan \delta_o} + \omega)^2 + (l + g)^2} - \frac{l}{\tan \delta_o} + \omega \tag{2.10}$$

Al seguir una ruta definida por una función $Y = F(X)$ a velocidad $\overrightarrow{\mathbf{v}}$ y aceleración $\overrightarrow{\mathbf{a}}$, se tiene que la curvatura de la trayectoria está definida por

$$K = \frac{1}{R} = \frac{a_n}{v^2} \tag{2.11}$$

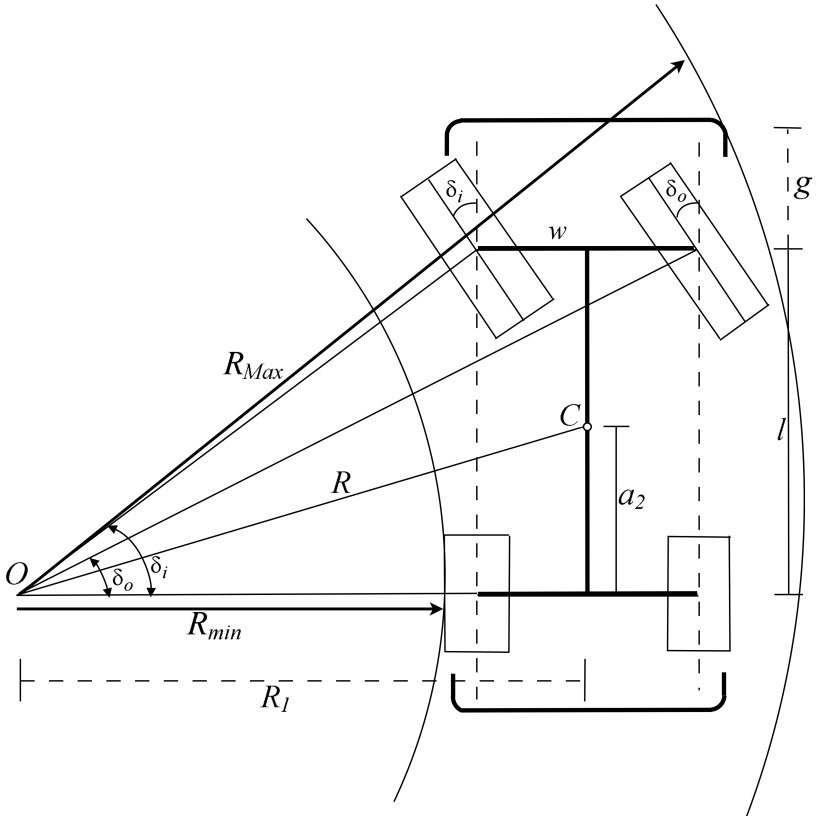

Figura 2.4: Espacio requerido para el giro del modelo Ackerman.

Donde a_n es el elemento normal de la aceleración en el marco de referencia global, X e Y, y se define en la ecuación 2.12; v es la magnitud del vector de velocidades en los ejes X e Y; y $\overrightarrow{\mathbf{a}}$ es el vector de aceleracion en X e Y.

$$a_n = \left| \frac{\overrightarrow{\mathbf{v}}}{v} \times \overrightarrow{\mathbf{a}} \right| = \frac{1}{v} \left| \overrightarrow{\mathbf{v}} \times \overrightarrow{\mathbf{a}} \right| = \frac{1}{v}(a_Y v_X - a_X v_Y) \tag{2.12}$$

$$a_n = \frac{Y'' X' - X'' Y'}{\sqrt{X'^2 + Y'^2}} \tag{2.13}$$

Por lo tanto, se puede definir a la curvatura de una trayectoria usando 2.12 o en términos de las funciones que definan la posición en el marco de referencia global mediante 2.13, dando como resultado 2.14.

$$k = \frac{Y'' X' - X'' Y'}{(X'^2 + Y'^2)^{3/2}} = \frac{Y'' X' - X'' Y'}{X'^3} \frac{1}{(1 + \frac{Y'^2}{Y'^2})^{3/2}} \tag{2.14}$$

De esta manera es posible calcular los ángulos de dirección de las ruedas delanteras a partir de 2.5 y 2.6 que satisfagan el radio requerido para el seguimiento de la trayectoria.

Capítulo 3

Metodología RGPPM

La metodología de planificación de rutas mediante redes resistivas se basa en un modelado circuital del espacio de configuración, en el que cada nodo del circuito representa una ubicación en el entorno, fue propuesta por primera vez en [16] y surge a partir de la metodología de campos de potencial artificial con el objetivo de poder realizar una implementación en chips VLSI. La propuesta de mejora a la metodología resuelve diversas problemáticas presentadas en campos de potencial y permite cómputo paralelo.

La principal problemática en campo de potencial artificial era la magnitud del gradiente del campo, lo cual genera un alto esfuerzo computacional y descartando su implementación en chips VLSI, por ello con el uso de redes 2D se evita el aumento exponencial en la carga carga computacional y permite la viabilidad de implementaciones en chips. Otra problemática era la presencia de mínimos locales, para lo cual se propuso el modelo de obstáculos como zonas no conductoras dentro de un medio conductor en vez de zonas de carga contraria a la fuente, se hizo el modelado de la fuente como una fuente de corriente S ubicada en el perímetro del medio conductor y el destino como una fuente de corriente de signo contrario G ubicada en otro punto del perímetro del medio conductor. En la Figura 3.1 se muestra un espacio de configuración propuesto como una distribución de conductividad circular para obtener soluciones en términos de armónicas circulares. Se obtienen tres rutas A, B y C partiendo de S que inyecta corriente en un punto del perímetro hacia G, hubicado en otro punto del perímetro.

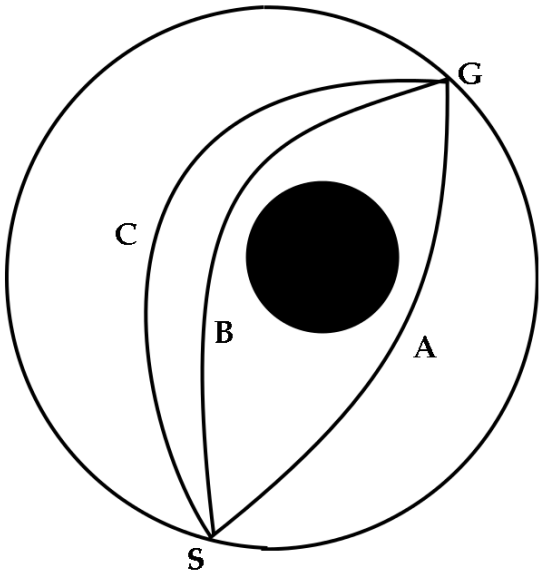

Figura 3.1: Primer propuesta de planificación mediante red resistiva en [16].

Posteriormente en [17] se presentó una optimización de la metodología mediante el uso del método multifrontal de solución de matrices, reduciendo tiempo de cómputo y con ello reduciendo el consumo de CPU. El método multifrontal fue desarrolado por [18] en 1983 como una generalización del método frontal para llevar a cabo factorizaciones LL^T o LDL^T en matrices simétricas. Poco después se desarrollaron mejoras para su aplicación en factorización de sistemas simétricos indefinidos, sistemas no simétricos usando factorizaciones LDL^T y LU y factorización QR.

El método multifrontal de solución de matrices dispersas mejora el uso de memoria debido a que en vez de realizar las operaciones sobre matrices grandes, realiza una factorización en matrices más pequeñas llamadas matrices frontales, que son bloques formados con las contribuciones del pivoteo de la etapa en la que se esté trabajando. Se parte de los nodos hijos, que se usan para construir los nodos padre dentro de un árbol de ensamblaje, las matrices frontales resultantes contendrán las contribuciones de los nodos hijos y el proceso de ensamblaje se repite hasta que se obtiene la matriz completa de todas las

contribuciones de los nodos hijos definida por

$$A = \sum_{i=1}^{n} A^i \tag{3.1}$$

Donde A^i es una contribución de un elemento finito, es decir, un nodo hijo. El árbol de ensamblaje también es llamado árbol de eliminación como en el método frontal, aunque en este caso no se realiza ningún tipo de eliminación.

El método multifrontal consta de tres pasos: el primero consiste en la formulación del árbol de ensamblaje; el segundo consiste en la factorización, en la que se obtienen las matrices frontales hijas, seguidas de sus corespondientes frontales padre y se lleva a cabo de manera transversal de abajo hacia arriba; por último se resuelve por sustitución mediante un recorrido del árbol, aplicando una solución triangular a cada frontal.

También se experimentó con diferentes tipos de modelado geométrico de la red, partiendo de modelado con figuras cuadradas, Figura 3.2, donde la posición inicial del robot es representada como una fuente de voltaje y el destino como la tierra del circuito.

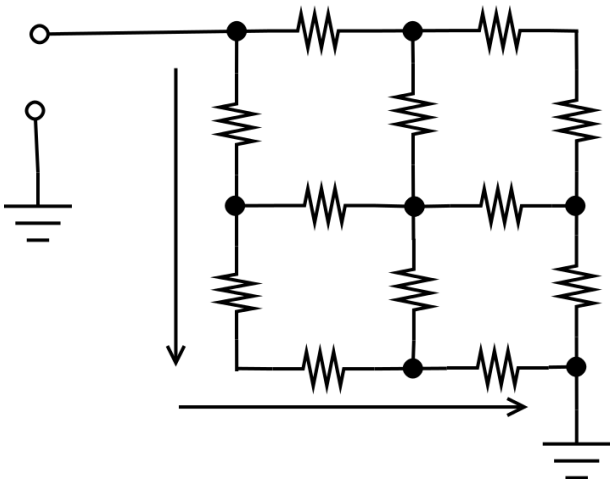

Figura 3.2: Modelado del espacio de configuración mediante una red resistiva.

En la programación se inicia con la configuración de las matrices de nodos y de resistencias, seguido de la configuracion de las ecuaciones de equilibrio con las que se crea la matriz MNA, A en la ecuación 3.2, conformada por las conductancias de cada elemento del circuito; B es el vector excitación en el que se incluyen las fuentes y se procede a

resolver para el vector $\mathbf{x}$ que contiene los valores nodales.

$$A\mathbf{x} = B \tag{3.2}$$

Una vez calculados los voltajes nodales, se inicia la búsqueda de la ruta hacia el destino mediante el cálculo de las corrientes que salen hacia todos los nodos unidos directamente al nodo en el que se esté iterando. Se comparan las corrientes de salida hacia los nodos adyacentes y el nodo hacia el que halla mayor flujo de corriente calculado será el próximo nodo de la ruta. El proceso se repite hasta que la tierra es alcanzada, Figura 3.3.

Con RGPPM se puede modelar el espacio de configuración usando diferentes formas geométricas, lo cual permite obtener trayectorias de formas diferentes y que se adaptan a diferentes ambientes, por ejemplo, en la Figura 3.4 se muestra un ejemplo de RGPP con modelado cuadrado del espacio de configuración con presencia de obstáculos. En [17] se muestran los resultados obtenidos al experimentar con modelado cuadrado, cuadrado con diagonal, cuadrado con nodo en el centro, hexagonal y hexagonal con nodo en el centro.

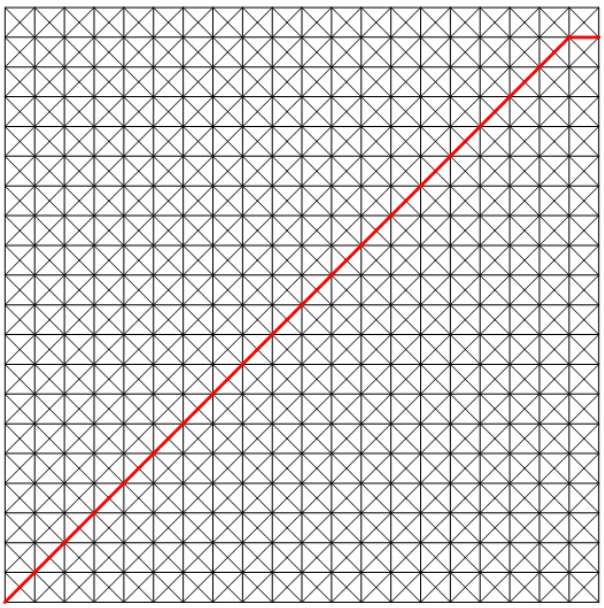

Figura 3.3: Metodologia RGPPM con modelado de cuadrado con centro.

Debido a la geometría de las rutas generadas, ésta metodología se ajusta muy bien para poder implementarse en robots omnidireccionales como drones, gracias a que estos

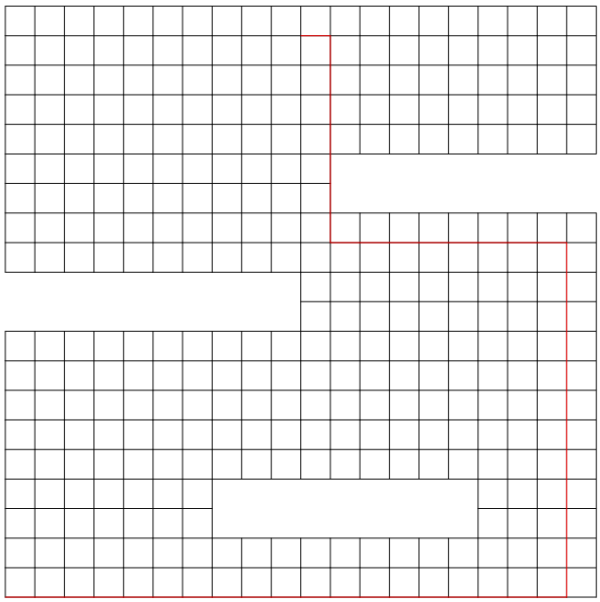

Figura 3.4: Metodologia RGPPM con modelado de cuadrado y obstáculos.

robots tienen la capacidad de moverse hacia todas las direcciones sin necesitad de cambiar su orientación actual.

Lo anterior sienta las bases para poder realizar pruebas en robots no holonómicos mediante la aplicación de alguna técnica de optimización de trayectorias para obtener trayectorias con curvas suaves de la forma mostrada en la Figura 3.5; ya que en éste tipo de robots como los robots de dirección Ackerman, no es posible seguir trayectorias en rutas como las mostradas en éste capítulo. En el presente trabajo se emplea el algoritmo planteado en [17], a partir del cual se obtienen las rutas con las cuales se trabaja posteriormente.

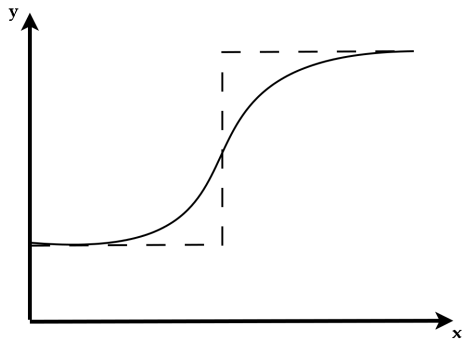

Figura 3.5: Comparación de la forma de una ruta basada en rectas, mostrada con linea punteada; y la trayectoria deseada para robots no holonónicos, en linea continua.

Capítulo 4

Seguimiento de ruta con robot de dirección Ackermann

Para obtener una trayectoria que pueda ser fácilmente seguida por el robot de dirección Ackerman se realiza un seguimiento de ruta de manera similar al seguimiento puro de una trayectoria previamente suavizada, pero que a diferencia de éste, se calcula un seguimiento de la ruta original hecha de lineas rectas de tal forma que el resultado es una trayectoria suavizada y obteniendo el ángulo de dirección a lo largo de la trayectoria. Ésta idea se basa en el algoritmo presentado en [21], en el que se propone un algoritmo para segimiento de objetivos y rutas en simulaciones basado en el cálculo de un vector de cambio de dirección $\overrightarrow{\mathbf{s}}$ a partir de un vector de dirección actual $\overrightarrow{\mathbf{v}}$ y un vector de dirección deseado $\overrightarrow{\mathbf{d}}$, con el vector de cambio de dirección se modifica el vector de velocidad actual para que con cada iteración se pueda acercar a la dirección deseada $\overrightarrow{\mathbf{d}}$. Por lo tanto, con la búsqueda constante una dirección deseada a la vez que se avanza en la dirección actual, se construye una trayectoria suavizada con cada iteración, figura 4.1.

Como primer paso, una vez que se ha calculado la ruta mediante RGPP, se empieza por asignar valores de coordenadas dentro de un plano de dos dimensiones a cada punto $P[i]$ de la ruta y crear una lista de rectas $Rects = [Rect_0, Rect_1, ..., Rect_{n-1}]$, las cuales están conformadas por pares de puntos (a, b) que unen los puntos de la ruta, con lo que se obtiene algo parecido al ejemplo de la figura 4.2. Una vez hecho esto se procede a realizar el seguimiento de las rectas establecidas entre cada par de puntos (a, b).

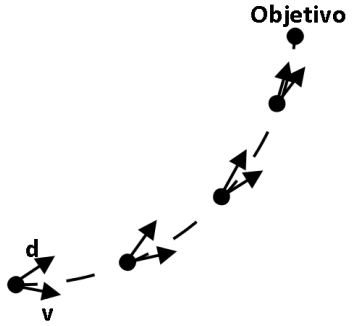

Figura 4.1: Concepto de una trayectoria suavizada a través de varias iteraciones con la propuesta de [21].

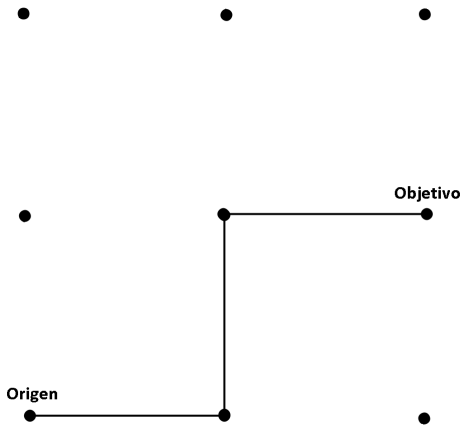

Figura 4.2: Concepto de ruta hecha de lineas rectas que conectan los nodos, previamente calculados con RGPP.

El seguimiento de la ruta se lleva a cabo mediante el seguimiento de un objetivo local $\overrightarrow{\mathbf{t}}$, éste objetivo local hará un recorrido de la ruta original, avanzando sobre las rectas que la conforman y mientras esto sucede se obtendrá la nueva posición del robot sumando $\overrightarrow{\mathbf{s}}$ al vector de velocidad $\overrightarrow{\mathbf{v}}$ y sumando el resultado al vector de posición $\overrightarrow{\mathbf{p}}$, Figura 4.3, construyendo el esquema parecido al mostrado en la Figura 4.1.

Partiendo del punto inicial, se calcula una predicción $\overrightarrow{\mathbf{pr}}$ de la posición, a cierta distancia a la que estaría el robot de seguir en la misma direción después de un tiempo. Se toma el vector $\overrightarrow{\mathbf{ap}}$ que va del primer punto $\overrightarrow{\mathbf{a}}$ de la recta que se esté trabajando al punto predictivo $\overrightarrow{\mathbf{pr}}$ de la Figura 4.6, luego se calcula la proyección escalar $\overrightarrow{\mathbf{sp}}$ mediante la ecuación 4.3 sobre la recta *ab*. Al inicio la proyeccción escalar $\overrightarrow{\mathbf{sp}}$ se hubicará en la misma

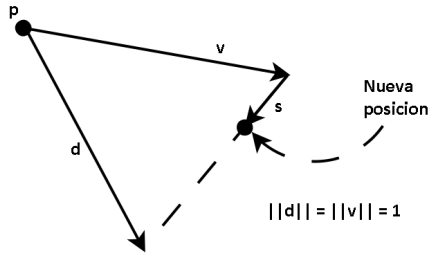

Figura 4.3: Concepto básico del seguimiento de objetivo.

posición que $\overrightarrow{\mathbf{pr}}$ ya que se avanza hacia el frente sin ningún cambio de ángulo con respecto a la recta *ab* en la Figura 4.4.

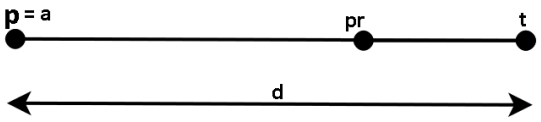

Figura 4.4: Inicio del recorrido de la ruta con la técnica propuesta.

$$\overrightarrow{\mathbf{ap}} = \overrightarrow{\mathbf{pr}} - \overrightarrow{\mathbf{a}} \tag{4.1}$$

$$\overrightarrow{\mathbf{ab}} = \overrightarrow{\mathbf{b}} - \overrightarrow{\mathbf{a}} \tag{4.2}$$

$$\overrightarrow{\mathbf{sp}} = \frac{\overrightarrow{\mathbf{ab}}}{||\overrightarrow{\mathbf{ab}}||} \cdot (\overrightarrow{\mathbf{ap}} \cdot \overrightarrow{\mathbf{ab}}) \tag{4.3}$$

$$\overrightarrow{\mathbf{np}} = \overrightarrow{\mathbf{a}} + \overrightarrow{\mathbf{sp}} \tag{4.4}$$

El objetivo local $\overrightarrow{\mathbf{t}}$ de la Figura 4.4 se hubicará sobre la recta *ab*, a cierta distancia *ds* por delante del punto $\overrightarrow{\mathbf{np}}$, que en la ecuación 4.4 es igual al punto *a* mas el vector de proyeccción escalar $\overrightarrow{\mathbf{sp}}$ calculado mediante la ecuación 4.3. La recta sobre la que se ubicará el objetivo local se cambiará a la siguiente de la ruta una vez que la proyección escalar se ubique fuera de la recta actual.

Al cambiar de recta, cuyo nuevo sentido sea diferente que la recta anterior, se cambia el sentido del objetivo local debido a la ecuación 4.5, produciendose el escenario de la Figura 4.5, en donde se puede ver que $\overrightarrow{\mathbf{v}}$ y $\overrightarrow{\mathbf{d}}$ ya difieren en su sentido.

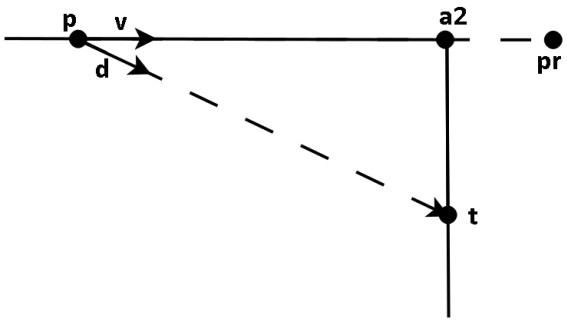

Figura 4.5: Cambio de recta para el objetivo local.

$$\overrightarrow{\mathbf{t}} = \overrightarrow{\mathbf{sp}} + \left(\frac{\overrightarrow{\mathbf{b}} - \overrightarrow{\mathbf{a}}}{||\overrightarrow{\mathbf{b}} - \overrightarrow{\mathbf{a}}||} \right) \cdot ds \tag{4.5}$$

En cada iteración del algoritmo se revisa la desviación r de la posición predictiva $\overrightarrow{\mathbf{pr}}$ con respecto a la ruta, si es mayor a una distancia máxima de desvio, establecida en 4 cm a los cuales el robot podría desviarse sin salirse del camino, se calcula el vector de cambio de dirección $\overrightarrow{\mathbf{s}}$ de la Figura 4.3 con la expresión 4.7, donde se resta el vector de velocidad $\overrightarrow{\mathbf{v}}$ a $\overrightarrow{\mathbf{d}}$ para obtener un vector de magnitud igual a la distancia entre $\overrightarrow{\mathbf{d}}$ y $\overrightarrow{\mathbf{v}}$ y con dirección que va de $\overrightarrow{\mathbf{v}}$ a $\overrightarrow{\mathbf{d}}$. El resultado $\overrightarrow{\mathbf{s}}$ se escala a $maxf$, que es la magnitud a la que se puede cambiar el sentido del vector de dirección sin que se sobrepase el ángulo máximo al que se puede mover el robot de dirección Ackerman. Se suma $\overrightarrow{\mathbf{s}}$ al vector de velocidad, con lo cual se cambia la dirección del movimiento del robot y se continúa el proceso de la forma mostrada en las figuras 4.6 y en 4.7.

$$\overrightarrow{\mathbf{d}} = \left(\frac{\overrightarrow{\mathbf{t}} - \overrightarrow{\mathbf{p}}}{||\overrightarrow{\mathbf{t}} - \overrightarrow{\mathbf{p}}||} \right) \cdot maxv \tag{4.6}$$

$$\overrightarrow{\mathbf{s}} = \frac{\overrightarrow{\mathbf{d}} - \overrightarrow{\mathbf{v}}}{||\overrightarrow{\mathbf{d}} - \overrightarrow{\mathbf{v}}||} \cdot maxf \tag{4.7}$$

$$\overrightarrow{\mathbf{v}} = \overrightarrow{\mathbf{v}} + \overrightarrow{\mathbf{s}} \tag{4.8}$$

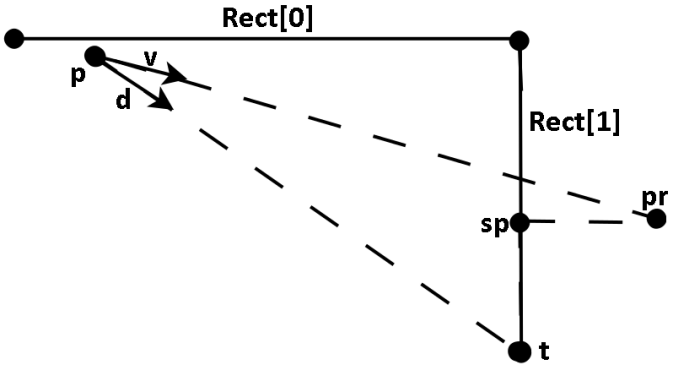

Figura 4.6: Cambio de dirección del robot.

En la Figura 4.7 se muestra el avance hacia el punto blanco de forma progresiva mostrando el avance de la posición predictiva $\overrightarrow{\mathbf{pr}}$ con puntos rojos y de la posición del robot $\overrightarrow{\mathbf{p}}$ con puntos azules. En el diagrama de la Figura 4.8 se muestra el algoritmo descrito en este Capítulo.

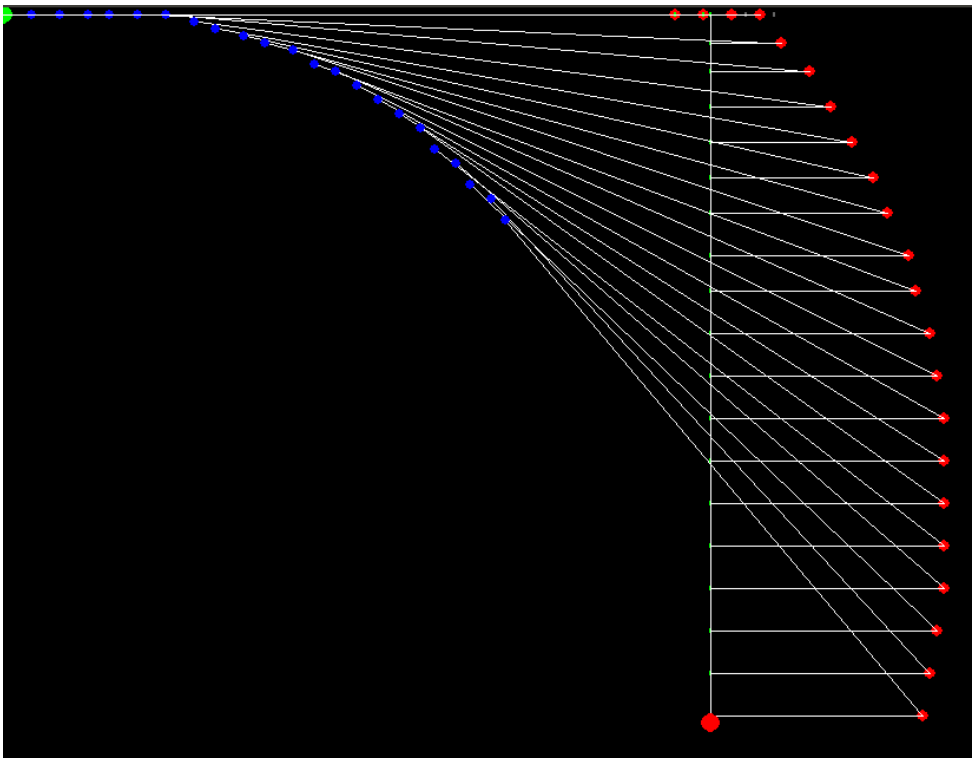

Figura 4.7: Avance de la posición del robot y de la posición predictiva de forma progresiva.

En el siguiente ejemplo sencillo se muestra el proceso descrito para el algoritmo propuesto. Se tienen dos rectas formando un ángulo de 90 grados en la Figura **??**, el punto de partida es la ubicación (0,0) y el objetivo es el punto (5.5). En este ejemplo se establece maxf=1, maxv=2, la posiciń predictiva se ubicara 3 unidades por delante de la posición p y la longitud de las rectas será de 5 unidades.

En la primera iteración la posición se mueve 2 unidades sin cambio alguno ya que la dirección deseada es la misma que $\overrightarrow{\mathbf{v}}$, la posición resultante es (2,0), la posición predictiva es (5,0), el punto $\overrightarrow{\mathbf{np}}$ es (5,0), $\overrightarrow{\mathbf{v}}$ se mantiene en (2,0) y el objetivo t es (6,0).

la posición predictiva es (7,0) pero ya que la proyección escalar se ubica fuera de la recta actual, se cambia a la siguente recta. El punto normal

Al simular el algoritmo usando python se obtienen los resultados de la Figura 4.9, en la que el objetivo es llegar del punto A al punto B, las rutas originales se muestran mediante las rectas delgadas de color blanco y la trayectoria resultante mediante la linea de color gris de mayor grosor.

Para llevar a cabo la simulación se usó una magnitud máxima para $\overrightarrow{\mathbf{s}}$ de 0.828712 cm de longitud para conseguir 30 grados de dirección del robot, que es el ángulo máximo que se puede conseguir con el robot utilizado.

Con base en las dimensiones de la estructura mecánica usada se procede a calcular el radio de curvatura mediante la ecuación 2.11 cada vez que se obtiene un nuevo punto de la trayectoria. Ya que los cálculos de la trayectoria se realizan usando vectores en lugar de funciones paramétricas, se usa la definición de a_n en la ecuación 2.12 y se usan los vectores $\overrightarrow{\mathbf{s}}$, en lugar de $\overrightarrow{\mathbf{a}}$ y el vector $\overrightarrow{\mathbf{v}}$, mencionados en el Capítulo 2. Una vez obtenida la curvatura de la ruta, se obtiene el radio de la curva con la ecuación 2.2 y el ángulo de dirección δ con la expresión 2.4.

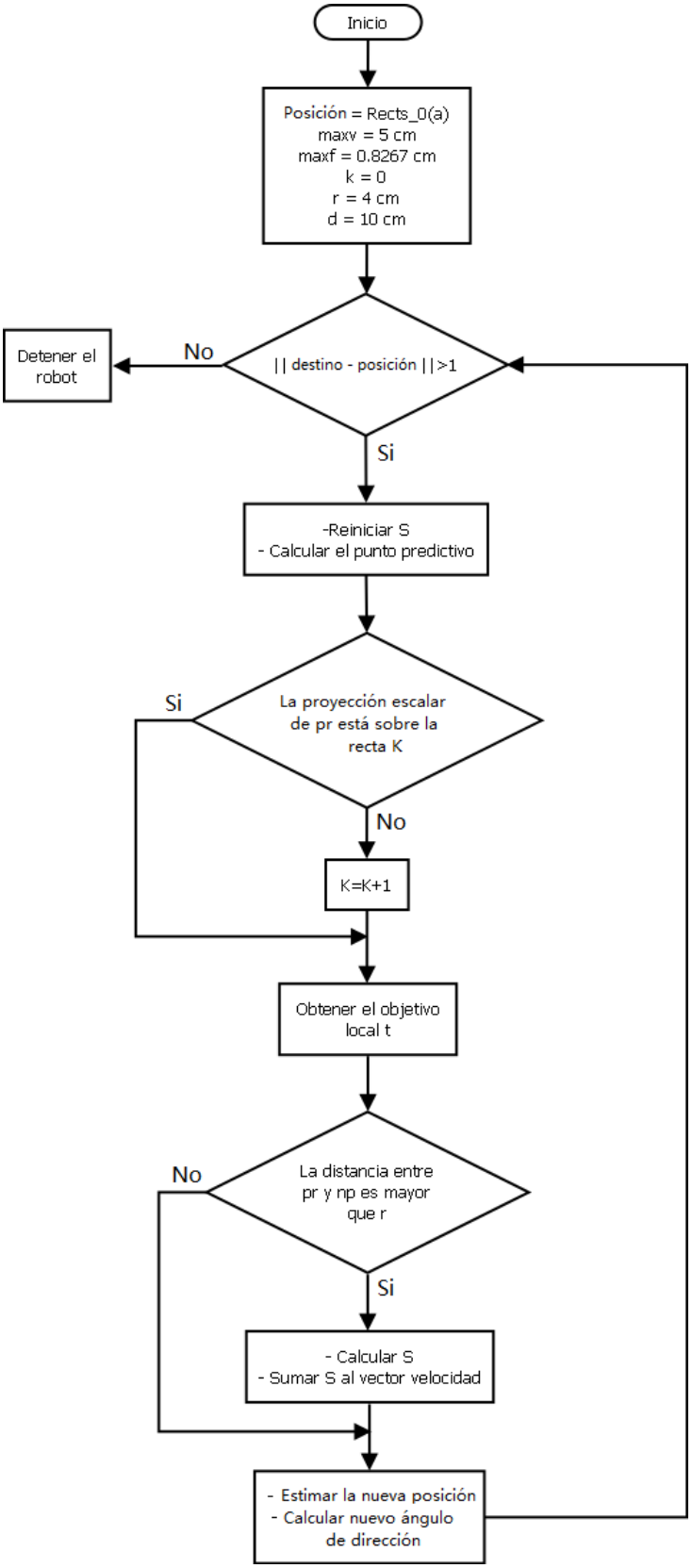

Figura 4.8: Algoritmo propuesto para el seguimiento de ruta.

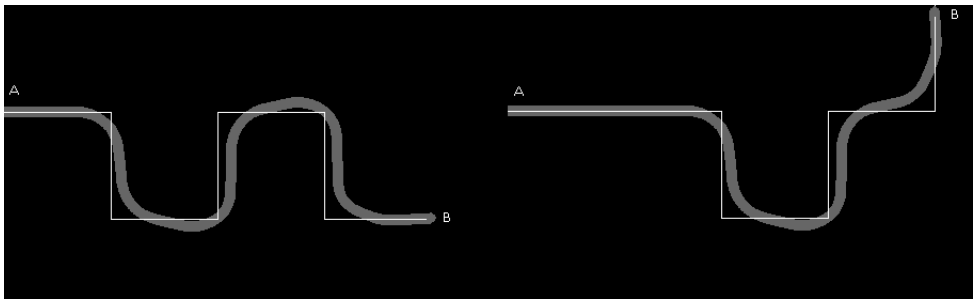

Figura 4.9: Simulación del seguimiento de ruta para un robot de dirección Ackerman, con un ángulo de dirección máximo de 30 grados. Programado en Python.

Capítulo 5

Implementación y pruebas del sistema robótico

El sistema robótico consiste de una Raspberry pi 3 B+ para realizar el cálculo de la ruta y el cálculo del seguimiento de la ruta, para después enviar el ángulo de dirección a una tarjeta Arduino Mega mediante el cual se hace el control del motor de DC 540 con escobillas, a traves de un circuito controlador de velocidad ESC. Mediante el Arduino también se controla al servomotor Mg996r, encargado de la dirección de las llantas delanteras, enviando una señal PWM directamente desde la tarjeta arduino.

Debido a limitaciones de hardware no se se cuenta con retroalimentación de sensores, por lo tanto, no se puede saber la posición exacta del robot ni la velocidad exacta en todo momento, por ello se eligió empíricamente una velocidad de 1.7 m/seg en la cual se observó que el robot no presentó derrapes. Siendo una velocidad estable esto también implica que se puede mantener en estado Ackermann sin generar mucha inercia, haciendo que el torque del motor principal no tenga problemas para mantener una velocidad constante, fiable hasta cierta distancia de entre 6 y 7 metros a la cual el error empieza a aumentar, de acuerdo a pruebas realizadas. Lo anterior, debido a la falta de retroalimentación de algún sensor que permita medir la velocidad exacta en todo momento para así poder aplicar correcciones, por ejemplo mediante un control PID. Tampoco se cuenta con sensores de proximidad con objetos, haciendo que se pierda total noción de la ubicacion del robot superada la distancia ya mencionada. En la Figura 5.1 se puede observar el esquema del

sistema robótico.

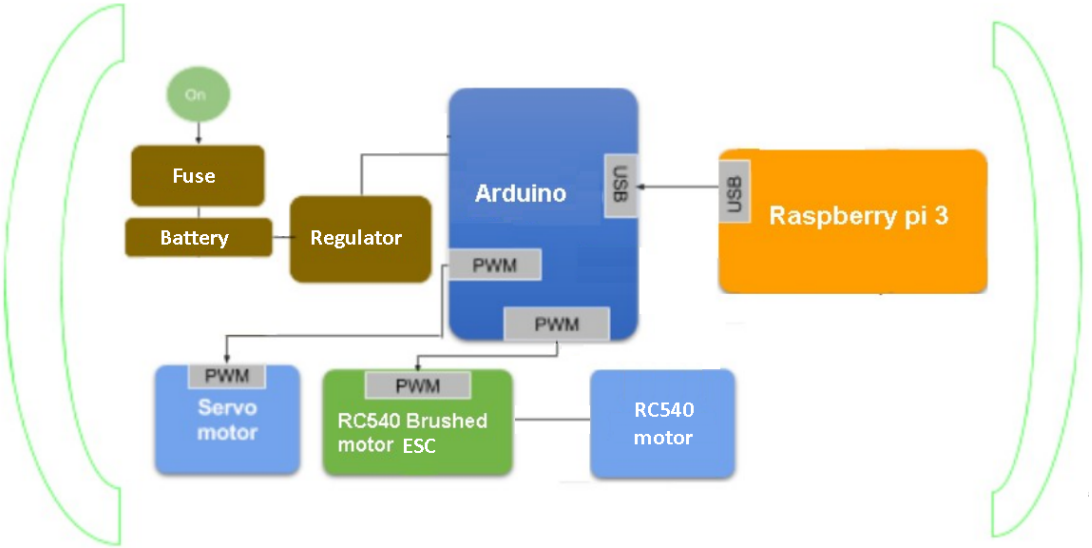

Figura 5.1: Esquema del sistema robótico.

El prototipo implementado tiene un largo cinemático de 25.5 cm y un ancho cinemático de 14 cm éstas medidas del robot se emplearon dentro del algoritmo descrito en el capítulo anterior para el cálculo del ángulo de dirección del robot. En la Figura 5.2 se muestra el prototipo ensamblado.

Se realizaron pruebas con el prototipo en tres escenarios distintos mostrados en los diagramas de las Figuras 5.3, 5.4 y 5.5, donde se muestran rutas que han sido previamente obtenidas con RGPP.

Al simular el método RGPP usando estos escenarios y aplicando el algoritmo de seguimiento de trayectorias explicado en el capítulo anterior se obtienen los resultados de la Figura 5.6; en donde se muestra al trazado que hace el avance del objetivo local representado con pundos verdes, los puntos generados de la prediccion de la posicion se muestra con puntos rojos y el seguimiento del centro de masa del robot se muestra con puntos de color blanco.

En las Figuras 5.7, 5.8 y 5.9 se muestran imágenes de las pruebas experimentales con el prototipo, por un lado se muestra al robot mientras hacía el recorrido de la trayectoria y a un lado se muestra el seguimiento de la trayectoria. Ambas pruebas fueron realizadas

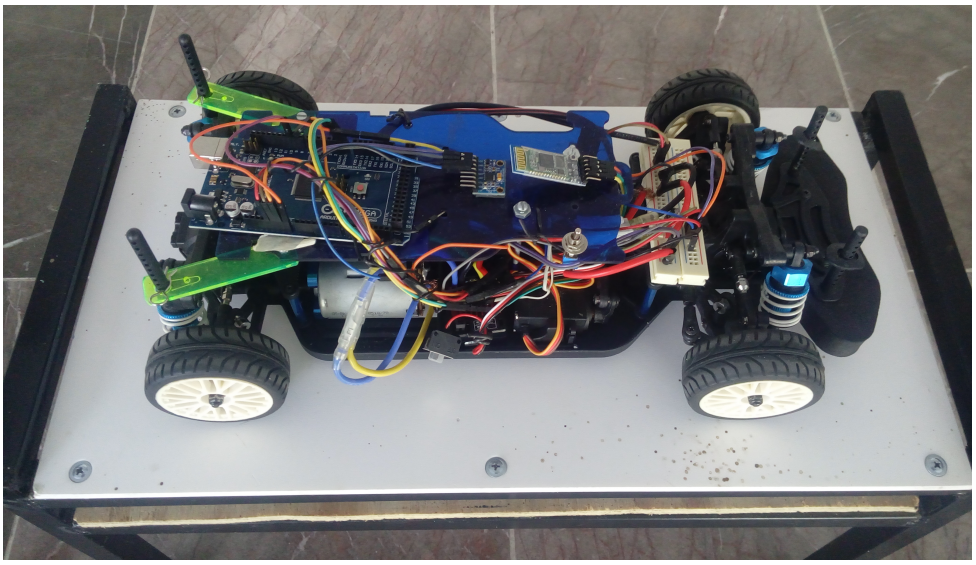

Figura 5.2: Sistema robótico implementado en este trabajo.

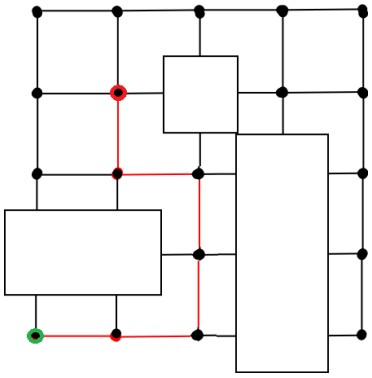

Figura 5.3: Escenario 1 con obstáculos, la ruta que va del nodo 1 al nodo 17 se muestra con lineas rojas, el nodo inicio se marca con un círculo verde y el destino con un círculo rojo.

usando Python 3 enviando el ángulo de dirección 34 veces por segundo con el objetivo de sincronizar la ejecución de la técnica de seguimiento en la Raspberry con la velocidad de movimiento del prototipo. Como se mencionó antes, al usar una velocidad baja que pueda mantenerse en la condición Ackermann, se obtuvo un error muy bajo con respecto a la hubicación del destino del robot, siempre y cuando no se superaran los 6 metros de recorrido, a partir de donde el error aumenta significativamente debido a que el robot es muy sensible a las condiciones del suelo como para mentener un control estable de circuito abierto.

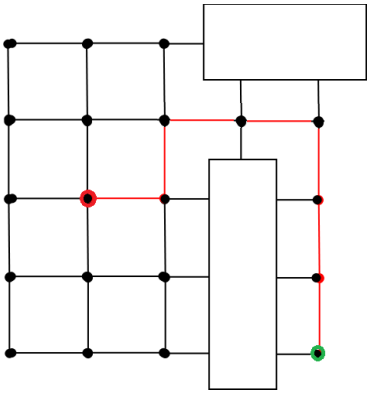

Figura 5.4: Escenario 2 con obstáculos, la ruta que va del nodo 5 al nodo 12 se muestra con lineas rojas, el nodo inicio se marca con un círculo verde y el destino con un círculo rojo.

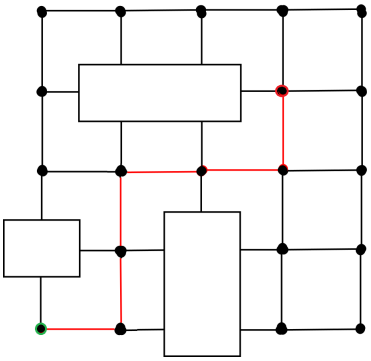

Figura 5.5: Escenario 2 con obstáculos, la ruta que va del nodo 1 al nodo 19 se muestra con lineas rojas, el nodo inicio se marca con un círculo verde y el destino con un círculo rojo.

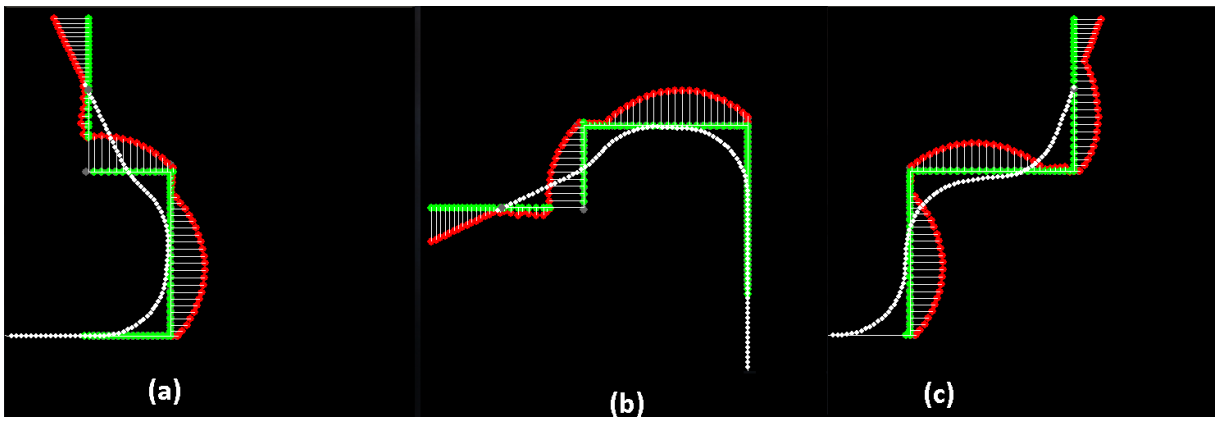

Figura 5.6: (a) seguimiento de la ruta en el escenario de la Figura 5.3, (b) seguimiento de la ruta en el escenario de la Figura 5.4 y (c) seguimiento de la ruta en el escenario de la Figura 5.5.

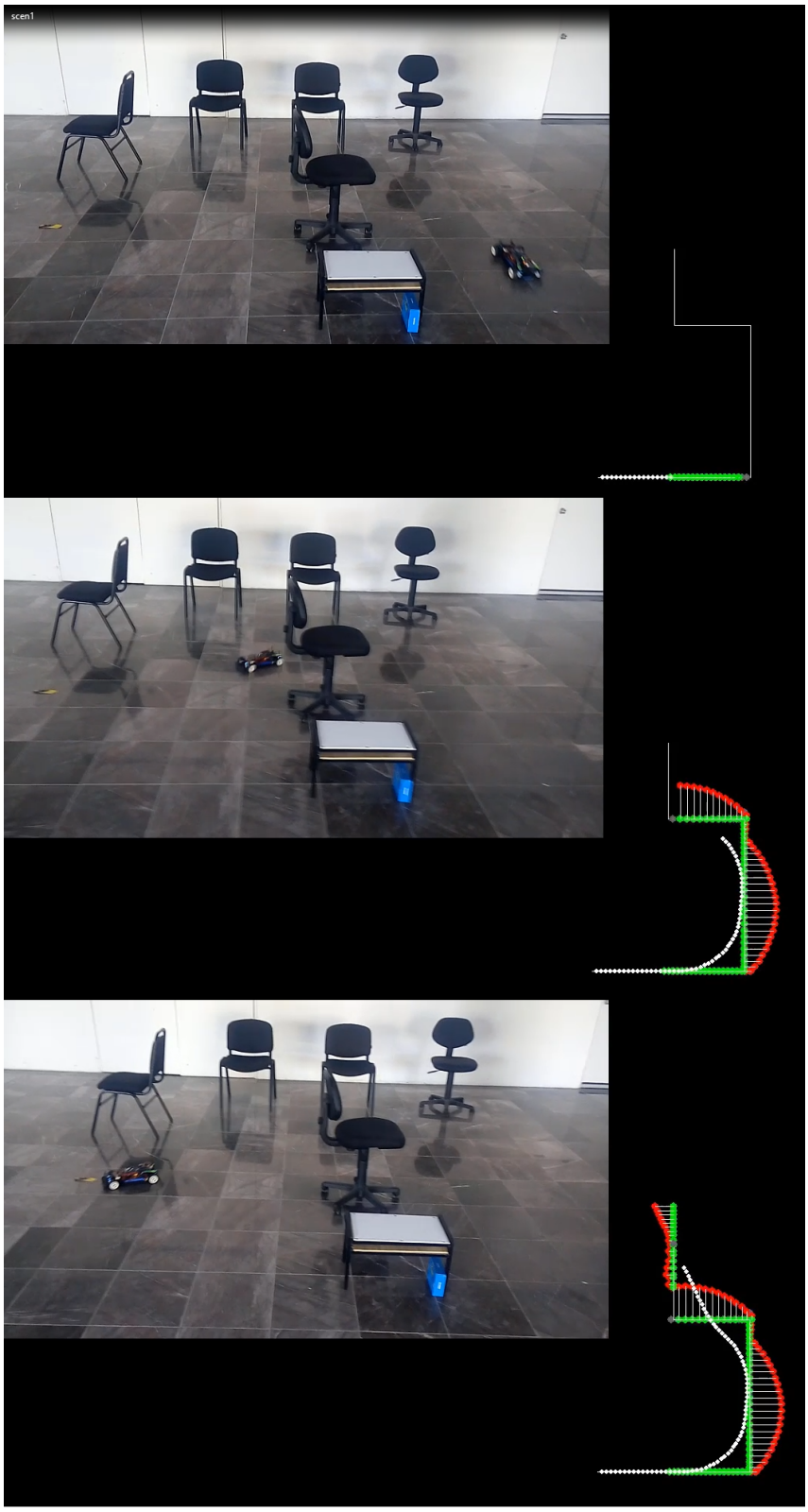

Figura 5.7: Secuencia de seguimiento en tiempo real de la ruta en el escenario de la Figura 5.3.

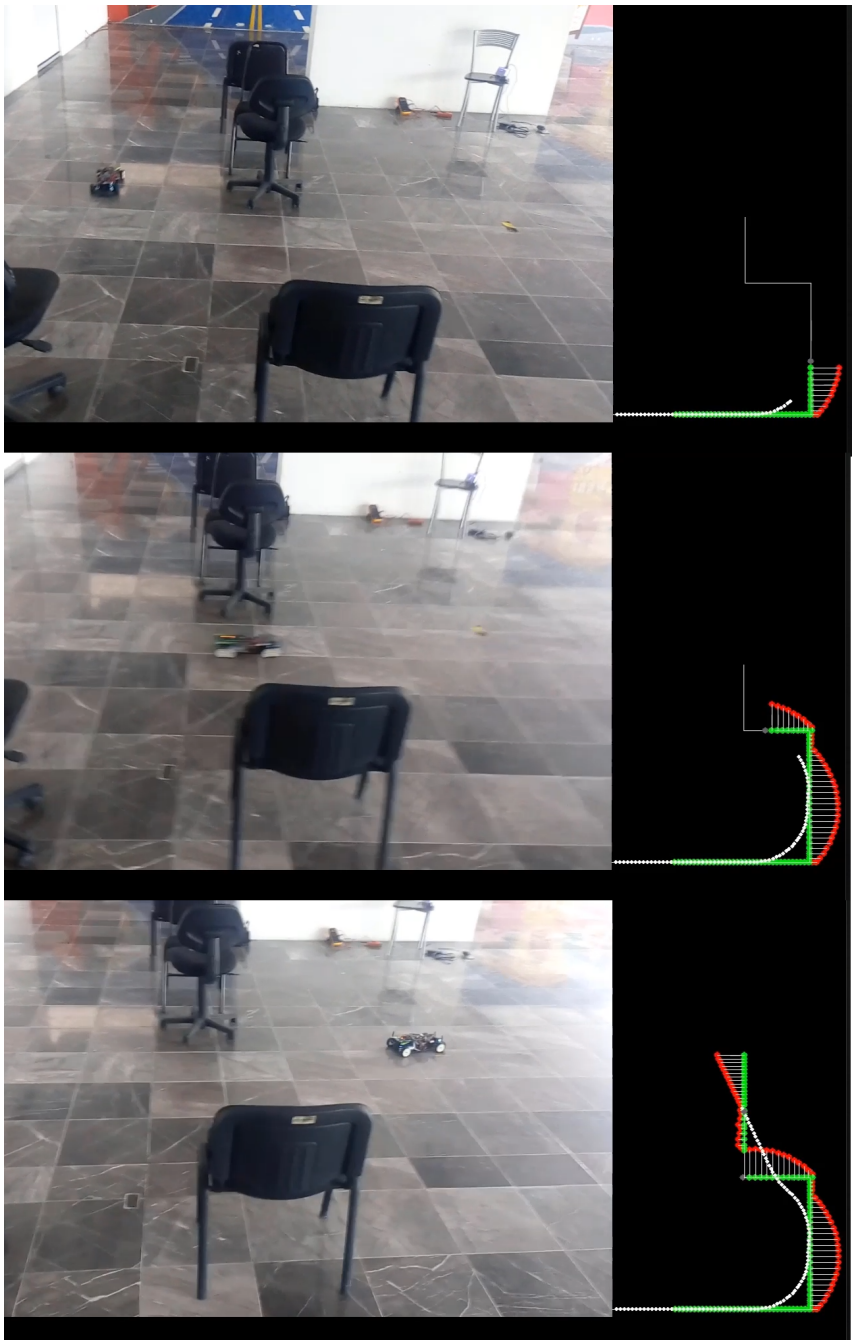

Figura 5.8: Secuencia de seguimiento de la ruta en tiempo real en el escenario de la Figura 5.4.

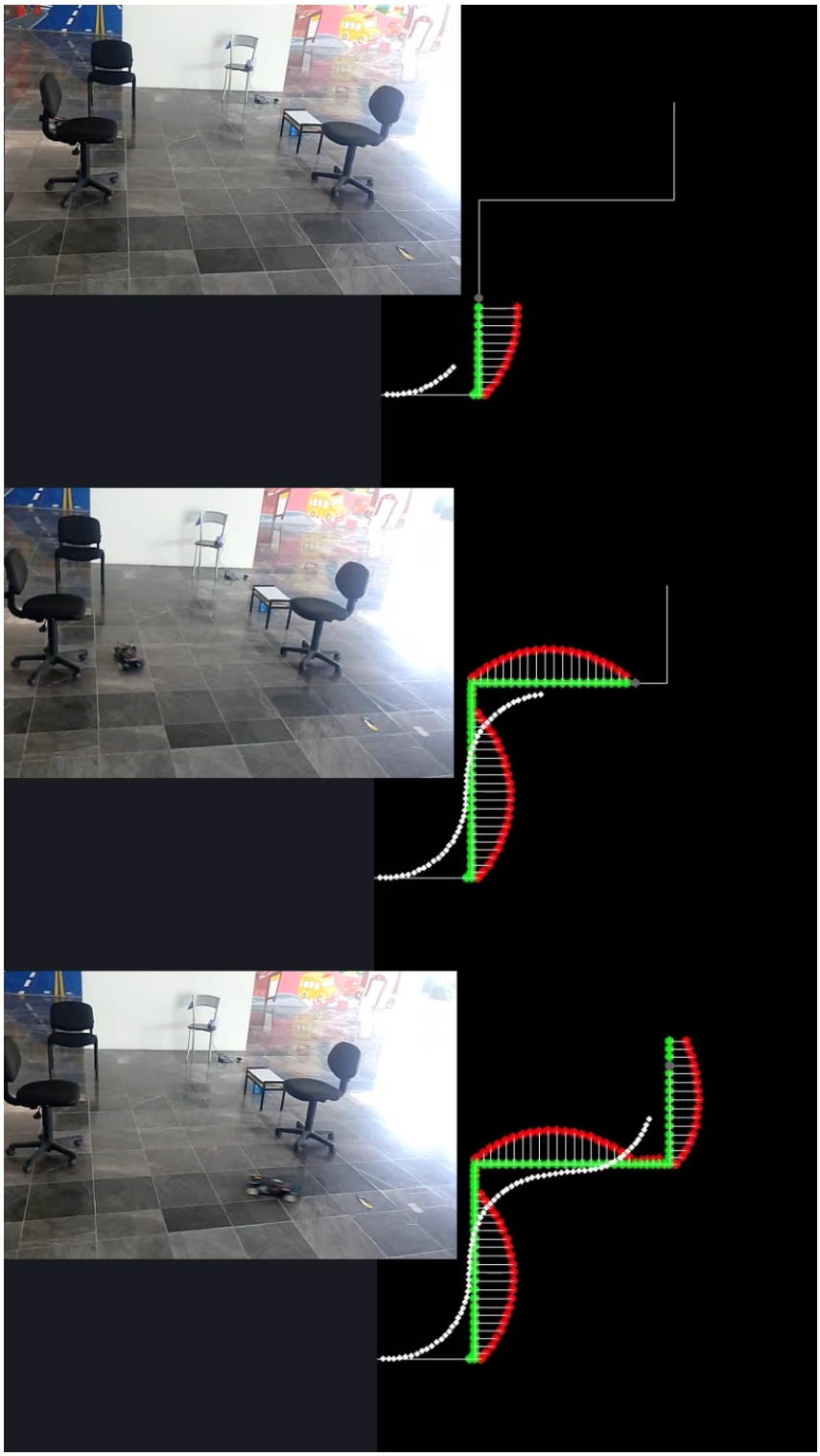

Figura 5.9: Secuencia de seguimiento en tiempo real de la ruta en el escenario de la Figura 5.5.

Capítulo 6

Conclusiones

Se demostró que es posible seguir rutas libres de colisiones basadas en RGPPM utilizando robots tipo Ackermann, lo que abre una perspectiva de cara a aplicaciones prácticas basadas en RGPPM. El procedimiento que se siguió fue el siguiente.

1 - Programar un seguimiento de ruta con curvatura máxima ajustable basado en el seguimiento de un objetivo.

2 - Se empleó una librería de Python para el cálculo de rutas mediante RGPPM y se aplicó el seguimiento propuesto.

3 - Se determinó la curvatura máxima que se puede obtener con el prototipo y con ésto se ajustó la curvatura en la técnica de seguimiento de ruta para después aplicarlo al robot de dirección Ackermann.

4 - Se experimentó con el prototipo Ackermann, realizando el seguimiento de rutas en tres escenarios distintos, validando así el método propuesto.

Los resultados obtenidos fueron adecuados, siempre y cuando se mantuviera al robot dentro de la condición Ackermann, debido a limitaciones del hardware con el que se cuenta y la falta de sensores para un mejor control. Sería importante trabajar en un esquema que utilice el método RGPP en conjunto con sensores para realizar ajustes en el modelado del ambiente y calcular rutas en tiempo real, para lo cual se sugiere lo siguente:

1 - Agregar más variables del modelo cinemático del vehículo, como son los coeficientes de fricción y la masa del robot, para así mejorar los cálculos de velocidad teniendo un

efecto directo en la estimación de su posición, obteniendo así un mejor seguimiento de la trayectoria.

2 - Agregar un motor con sensor de velocidad para poder estimar la posición con mayor fiabilidad y permitir realizar correcciones de velocidad mediante algún método adecuado.

3 - Agregar sensor láser para mejorar la estimación de la posición, así como evitar colisiones en tiempo real.

4 - El uso de una camara para mejorar la detección de obstáculos y permitir el procesamiento de imágenes con el objetivo de hacer un seguimiento de señalamientos.

Apéndice A

Apéndice

Algorithm 1 Técnica de seguimiento propuesta

$maxv \leftarrow 5cm$ {Distancia que se avanza cada vez que se ejecute el algoritmo.}
$maxf \leftarrow 0{,}826712$ {Mágnitud máxima del vector de giro permitido por el robot.}
$k \leftarrow 0$ {Número de recta inicial.}
$r \leftarrow 4$ {Radio de la ruta, usado para identificar desvios en la posición predictiva.}
$d \leftarrow 10$ {Distancia entre el objetivo local y la posición predictiva.}
$\overrightarrow{\mathbf{p}} \leftarrow Rects_0(a)$ {La posición inicial es la posición del punto (a) de la recta cero.}
{Se ejecutará el seguimiento mientras la posición sea diferente de la posición destino.}
while $||Rect_{n-1}(b) - position|| > 1$ **do**
  $\overrightarrow{\mathbf{s}} \leftarrow 0$ {Vector de asceleración.}
  $\overrightarrow{\mathbf{pr}} \leftarrow \overrightarrow{\mathbf{v}}$ {Se orienta la posición predictiva en la misma dirección que la velocidad.}
  $\overrightarrow{\mathbf{pr}} = \frac{\overrightarrow{\mathbf{pr}}}{||\overrightarrow{\mathbf{pr}}||} * 60$ {Se escala el vector predictivo en 60 cm.}
  {Se revisa si ya se ha superado la recta actual para así hacer el cambio de recta.}
  if $Rect_k(a)_x < Rect_k(b)_x$ **then**
    if $\overrightarrow{\mathbf{pr}}_x > Rect_k(b)_x$ **then**
      $K+ = 1$
    end if
  else if $Rect_k(a)_y < Rect_k(b)_y$ **then**
    if $\overrightarrow{\mathbf{pr}}_y > Rect_k(b)_y$ **then**
      $k+ = 1$
    end if
  else if $Rect_k(a)_y > Rect_k(b)_y$ **then**
    if $\overrightarrow{\mathbf{pr}}_y < Rect_k(b)_y$ **then**
      $k+ = 1$
    end if
  end if
  $\overrightarrow{\mathbf{ap}} = \overrightarrow{\mathbf{pr}} - Rect_k(a)$ {Se obtiene un vector del punto (a) al punto predictivo.}
  $\overrightarrow{\mathbf{ab}} = Rect_k(b) - Rect_k(a)$ {Crear una recta con los puntos de la recta actual.}
  $\overrightarrow{\mathbf{sp}} = \frac{\overrightarrow{\mathbf{ab}}}{||\overrightarrow{\mathbf{ab}}||} * (\overrightarrow{\mathbf{ap}} \cdot \overrightarrow{\mathbf{ab}})$ {Obtener la proyección escalar del punto predictivo sobre la recta actual.}
  $\overrightarrow{\mathbf{np}} = Rect_k(a) + \overrightarrow{\mathbf{sp}}$ {Hubicar la proyección escalar sobre la recta.}
  $\overrightarrow{\mathbf{t}} = \overrightarrow{\mathbf{np}} + \left(\frac{Rect_k(b) - Rect_k(a)}{||Rect_k(b) - Rect_k(a)||}\right) * d$ {Obtener el objetivo local.}
  {Si el punto predictivo está fuera de la ruta, se activa el seguimiento del objetivo.}
  if $\sqrt{(np(a_x) - pr(a_x))^2 + (np(a_y) - pr(a_y))^2} < r$ **then**
    $\overrightarrow{\mathbf{ds}} = \left(\frac{\overrightarrow{\mathbf{t}} - \overrightarrow{\mathbf{p}}}{||\overrightarrow{\mathbf{t}} - \overrightarrow{\mathbf{p}}||}\right) * maxv$ {Obtener un vector de la posición actual al objetivo actual y escalarlo a maxf.}
    $\overrightarrow{\mathbf{s}} = \overrightarrow{\mathbf{ds}} - \overrightarrow{\mathbf{v}}$ {Obtener el vector de asceleración.}
    $\overrightarrow{\mathbf{v}} = \overrightarrow{\mathbf{v}} + \overrightarrow{\mathbf{s}}$ {Sumar la asceleración a la velocidad.}
  end if
  $R = \frac{||\overrightarrow{\mathbf{v}}||^2}{a_n}$ {Obtener el radio de la curva.}
  $R1 = \sqrt{R^2 - w^2}$ {Obtener el radio R1.}
  $angle = 57{,}2957795 * arctan(R1/L)$ {Obtener el ángulo de dirección.}
  $\overrightarrow{\mathbf{p}} = \overrightarrow{\mathbf{p}} + \overrightarrow{\mathbf{v}}$ {Obtener la nueva posición.}
end while

Bibliografía

[1] Ronald Siegwart, Illah R. Nourbakshsh. "Introducion to Autonomous Mobile Robots". The MIT Press, 2004.

[2] Reza N. Jazar. "Veicle Dynamics: Theory and applications". *Springer Science*, Vol.1, 2008.

[3] Ziegler, J., Stiller, C. "Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios". InternationalConference on Intelligent Robots and Systems, pp. 1879-1884, 2009.

[4] Steven M. LaValle. "Rapidly-Exploring Random Trees: A New Tool for Path Planning". *International Conference on Robotics and Automation*, San Francisco, 2000.

[5] K. S. Al-Sultan, M. D. S. Aliyu. "A New Potential Field-Based Algorithm for Path Planning". Journal of Inteligent and Robotic System, No. 17, pp. 265-282, 1996.

[6] Kyung Yeop Han, Minho Lee, Young-Sup Lee. "Implementation of autonomous driving of a ground vehiclefor narrow high-curvature roads using surround view images". Springer Science+Business Media, LLC, part of Springer Nature, 30 august, 2018.

[7] Albert S. Huang, Seth Teller. "Probabilistic lane estimation for autonomous driving using basiscurves". Springer Science+Business Media, LLC, pp. 31:269-283, 2011.

[8] Taejun Song, Hyewon Lee, Kwangseok Oh. "Laser-scanner-based object state estimation and tracking controlalgorithms of autonomous truck using single-wheel driving module". Springer-Verlag GmbH Germany, part of Springer Nature, 19 july 2019.

[9] Madas, D., Nosratinia, M., Keshavarz, M., Sundstrom, P., Philippsen, R., Eidehall, A., Dahl, K. "On path planning methods for automotive collisionavoidance". IEEE Intelligent Vehicles Symposium (IV), pp. 931-937, 2013.

[10] Anna Petrovskaya, Sebastian Thrun. "Model based vehicle detection and tracking for autonomous urbandriving". Springer Science+Business Media, LLC, pp. 26: 123-139, 1 April 2009.

[11] Benenson, R., Petti, S., Fraichard, T., Parent, M. "Integrating perception and planning for autonomous navigation of urban vehicles". IEEE/RSJInternational Conference on Intelligent Robots and Systems, pp. 98-104, 2006.

[12] Min Gyu Park, Min Cheol Lee. "A New Technique to Escape Local Minimum in Artificial Potencial Field Based Path Planning". KSME International Journal, Vol. 17, No. 12, pp. 1876-1885, 2003.

[13] Christos Katrakazasa, Mohammed Quddusa, Wen-Hua Chenb, Lipika Deka. "Real-time motion planning methods for autonomous on-roaddriving: State-of-the-art and future research directions". Elsevier Ltd., Transportation Research Part C pp. 60:416-442, 2015.

[14] Zhuang Wang, Jiejin Cai. "Probabilistic roadmap method for path-planning in radioactive environmentof nuclear facilities". Elsevier, 2018.

[15] Carlos Hernández-Mejáa, Héctor Vázquez-Leal y Delia Torres-Muñoz. "A Novel Collision-Free Path Planning Modeling and Simulation Methodology for Robotical Arms Using Resistive Grids". Cambridge University Press, 2019.

[16] L. Tarassenko, A. Blake. "Analogue computation of collision-free paths". International Conference on Robotics and Automation, Sacramento, California, April 1991.

[17] Carlos Hernández-Mejáa, Héctor Vázquez-Leal, Alfonso Sánchez-González, ángel Corona-Avelizapa. "A Novel and Reduced CPU Time Modeling and Simulation

Methodology for Path Planning Based on Resistive Grids". Arabian Journal for Science and Engineering, 2018.

[18] I. S. Duff y J. K. Reid. "The multifrontal solution of indefinite sparse symmetric linear systems". ACM Transactions on Mathematical Software, 9:302-325, 1983.

[19] CHEN Xue-mei, JIN Min, MIAO Yi-song, ZHANG Qiang. "Driving decision-making analysis of car-following for autonomous vehicle under complex urban environment". Central South University Press and Springer-Verlag Berlin Heidelberg, pp. 24: 1476-1482, 2017.

[20] Enric Galceran, Alexander G. Cunningham, Ryan M. Eustice, Edwin Olson. "Multipolicy decision-making for autonomous driving viachangepoint-based behavior prediction: Theory and experiment". Springer Science+Business Media New York, pp. 41:1367-1382, 2017.

[21] Craig W. Reynolds. "Steering behaviors for autonomous characters". Game Developers Conference, 1999.