Colegio de Ciencia y Tecnología

DISEÑO, IMPLEMENTACIÓN Y CONTROL DE UN VEHÍCULO AUTÓNOMO MEDIANTE VISIÓN ARTIFICIAL

TESIS

Que para obtener el título de:

Licenciados en Ingeniería en Sistemas Electrónicos Industriales

PRESENTAN:

Daniel Lechuga Rosales
y Eleasar Gamaliel Rivera Morales

Director:
Dr. Mario Villafuerte Bante
Codirector:
Dr. Pablo Vera Bustamante

Ciudad de México, febrero 2021.

UACM
Universidad Autónoma de la Ciudad de México
Nada humano me es ajeno

CA Coordinación Académica

SISTEMA BIBLIOTECARIO DE INFORMACIÓN Y DOCUMENTACIÓN

UNIVERSIDAD AUTÓNOMA DE LA CIUDAD DE MÉXICO
COORDINACIÓN ACADÉMICA

RESTRICCIONES DE USO PARA LAS TESIS DIGITALES

DERECHOS RESERVADOS©

Integración del jurado:

Presidente: Dr. Ramírez Solíz Efrén Bernardo

Secretario: Dr. Villafuerte Bante Mario

Vocal: Dr. Vera Bustamante Pablo

Plantel de adscripción: Cuautepec

Agradecimientos

- **Lechuga Rosales Daniel**

  Agradezco a mis padres por el apoyo que me brindaron durante mi estancia en la Universidad Autónoma de la Ciudad de México, a cada uno de mis profesores que hicieron de mi un mejor estudiante y sobretodo mejor persona, a mis compañeros de clase, a mi director (Dr. Mario Villafuerte Bante) y codirector de tesis (Dr. Pablo Vera Bustamante) quienes fueron pilares fundamentales durante mi estancia final de este proceso.

- **Rivera Morales Eleasar Gamaliel**

  Agradezco principalmente a mi madre, que mediante su esfuerzo y dedicación me brindó el apoyo suficiente durante mi estancia en la Universidad Autónoma de la Ciudad de México (UACM), y así poder culminar mi carrera universitaria.

  Asimismo agradezco a mi familia en general por el apoyo mostrado, de igual forma agradezco a la UACM por permitirme formar parte de ella, agradezco además a todos y cada uno de los profesores que me ayudaron a crecer como estudiante, pero aún más como persona, en especial quiero agradecer al Dr. Mario Villafuerte Bante (Asesor) y al Dr. Pablo Vera Bustamante (Co-Director), quienes permitieron hacer posible este proyecto de tesis.

Resumen

En este trabajo se realiza el diseño, control e implementación de un vehículo autónomo mediante visión artificial. El cual está dotado con sensores y actuadores, además de la implementación en Robotic Operating System (ROS) de los algoritmos de control y de este modo lograr su correcto funcionamiento en dos aplicaciones autónomas. La primera consiste en el seguimiento de trayectorias definidas; mientras que la segunda se basa en el rebase en lazo abierto de un obstáculo estático sobre una línea recta.

Uno de los principales motivos para la elaboración de este trabajo, es contribuir en la innovación y el desarrollo tecnológico en vehículos autónomos.

Abstract

In this work the design, control and implementation of an autonomous vehicle through artificial vision is carried out. Which is equipped with sensors and actuators, in addition to the implementation in ROS of the control algorithms and thus achieve its correct operation in two autonomous applications. The first consists of tracking defined trajectories; while the second is based on the open loop overflow of a static obstacle on a straight line.

One of the main reasons for the development of this work is to contribute to innovation and technological development in autonomous vehicles.

Índice general

Índice de figuras

Glosario

1. **Actuador:** La definición de actuador es aquel dispositivo que puede transformar energía eléctrica, hidráulica o neumática cuando se activa un proceso con el objetivo de crear un efecto sobre un determinado proceso automatizado [30].

2. **Algoritmo:** es una serie de pasos organizados que describe el proceso que se debe seguir, para dar solución a un problema específico [31].

3. **Autonomía:** hace referencia a la condición de aquel o aquello, que en determinados contextos, no tiene dependencia de nadie [32].

4. **Calibración:** consiste en comprobar las desviaciones de indicación de instrumentos y equipos de medida por comparación con patrones con trazabilidad nacional o internacional [33].

5. **Catkin bloom:** es una herramienta de automatización de lanzamiento, diseñada para facilitar la generación de artefactos de lanzamiento específicos de la plataforma a partir de proyectos fuente. Bloom está diseñado para funcionar mejor con proyectos catkin, pero también puede acomodar otros tipos de proyectos [34].

6. **Centroíde:** es un punto que define el centro geométrico de un objeto [35].

7. **Concéntrico:** Se utiliza en el ámbito de la geometría para calificar a una figura que dispone del mismo centro que otra [36].

8. **Control:** significa medir el valor de la variable controlada del sistema y aplicar a este un proceso de realimentación para corregir la desviación del valor medido,

respecto al valor deseado [37].

9. **Diagrama:** es un gráfico que presenta los vínculos existentes entre los distintos componentes de un sistema o de un conjunto [38].

10. **Error:** es la diferencia que surge entre el valor deseado y el valor obtenido [39].

11. **Homografía:** En geometría, se denomina homografía a toda transformación proyectiva que determina una correspondencia entre dos figuras geométricas planas, de forma que a cada uno de los puntos y las rectas de una de ellas le corresponden, respectivamente, un punto y una recta de la otra [40].

12. **Interfaz:** es una conexión entre dos sistemas [41].

13. **Irreversibilidad:** es aquello que no se puede revertir; es decir, que no es posible que recupere la condición, el estado o la propiedad que tuvo con anterioridad [42].

14. **Lazo abierto:** se caracteriza por que no recibe ninguna información o retroalimentación sobre el estado de la variable, por lo regular estos se utilizan cuando la variable es predecible y tiene un amplio margen de error, ya que se puede calcular el tiempo o las veces que se debe de repetir el ciclo para completar el proceso [43].

15. **Microcontrolador:** es un circuito integrado que es el componente principal de una aplicación embebida [44].

16. **Pixel:** es la superficie homogénea más diminuta que forma parte de una imagen [45].

17. **PWM:** una señal de modulación de ancho de pulso (PWM) es un método para generar una señal analógica utilizando una fuente digital [46].

18. **Resolución:** es la precisión del detalle en las imágenes de mapa de bits, que se mide en píxeles por pulgada (ppp). Cuantos más píxeles por pulgada, mayor resolución [47].

19. **Reflectividad:** es la fracción de radiación incidente reflejada por una superficie [48].

20. **Rosbuild:** rosbuild contiene scripts para administrar el sistema de compilación basado en CMake para ROS [49].

21. **Señal digital:** son variables eléctricas con dos niveles bien diferenciados que se alternan en el tiempo transmitiendo información según un código previamente acordado [50].

Capítulo 1

Introducción

1.1. Motivación

Durante nuestra formación en la carrera de Ingeniería en Sistemas Electrónicos Industriales, se desarrollaron las habilidades para contribuir en la innovación y el desarrollo tecnológico en vehículos autónomos, capaces de seguir trayectorias definidas y realizar el rebase en lazo abierto de un vehículo estático; la solución se basa en la implementación de un prototipo retroalimentado con visión artificial.

1.2. Planteamiento del problema

Por todo el mundo circulan diariamente millones de vehículos, propensos a sufrir o provocar accidentes, gran cantidad de ellos son provocados por errores humanos. Es por esto que se estudia, investiga y trabaja en el desarrollo e implementación de vehículos autónomos.

1.3. Objetivos

1.3.1. Objetivo general

Diseñar, controlar e implementar un vehículo autónomo dotado con sensores, actuadores y algoritmos de visión artificial. Capaz de realizar el segumiento de rutas establecidas y rebases en lazo abierto de vehículos en reposo a lo largo de una trayectoria definida.

1.3.2. Objetivos particulares

- Se realizará el Diseño Asistido por Computadora (CAD) del vehículo autónomo.
- Se realizará la programación y validación de los algoritmos de visión artificial.
- Se implementará un algoritmo de programación para el sistema de control del vehículo autónomo en el microcontrolador Arduino mega.
- Se calibrarán los sensores y actuadores que conforman el vehículo autónomo.
- Se validarán las pruebas y funcionamiento del vehículo autónomo.
- Se implementará Robotic Operating System (ROS) para la comunicación y acoplamiento entre Arduino y Raspberry.
- Se implementará un sistema embebido para el procesamiento de información.

1.4. Organización de la tesis

El capítulo 2 menciona toda la teoría necesaria para llevar acabo la implementación y el control del vehículo autónomo. Se abordan temas fundamentales como visión artificial, ROS, Modelo Ackermann etc.

En el capítulo 3 se menciona cada uno de los dispositivos que conforman el vehículo autónomo que se diseñó en la UACM tanto en forma física como en CAD, la rela-

ción que tienen estos dispositivos con ROS asi como relación que existe entre estos componentes.

En el capítulo 4 se describe el tipo de control que se diseñó e implementó en el vehículo para realizar de manera adecuada las aplicaciones autónomas.

En el capítulo 5 se muestran las pruebas y resultados obtenidos con el vehículo autónomo, se describen imágenes del vehículo al momento de realizar las aplicaciones autónomas establecidas en el capítulo 1.

En el capítulo 6 se mencionan las conclusiones obtenidas de este proyecto y se habla acerca del trabajo a futuro; es decir se planea adaptar nuevos dispositivos y algoritmos de programación para hacer de este vehículo autónomo uno mejor.

1.5. Estado del arte de los vehículos autónomos

1.5.1. Vehículos autónomos

Se define vehículo autónomo, como aquel que es capaz de recorrer cierto camino sin depender de un conductor, el automóvil es capaz de llegar a su destino analizando durante intervalos de tiempo cualquier obstáculo, condición climática, entorno que lo rodea, etc. El análisis de todos los elementos se realiza de manera inteligente (inteligencia artificial) [1].

1.5.2. Antecedentes

Los vehículos autónomos están englobados en un concepto que no es novedoso; estudios, análisis y aplicaciones fueron realizados con anterioridad. Los primeros experimentos fueron llevados a cabo en el siglo XX, en la década de los años 20, cuando se intentó desarrollar vehículos autónomos que eran controlados a distancia, sin el éxito esperado [2].

El primer “vehículo autónomo”, precursor de los actuales, fue exhibido en la feria de las muestras Futurama en el año 1939, que fue patrocinada por General Motors. Este proyecto era controlado por radio y un circuito eléctrico, el vehículo fue expuesto

por el diseñador industrial Norman Bel Geddes [2].

En 1950 se desarrollaron los primeros controladores de velocidad, retroalimentados (feedback), con base en el movimiento del acelerador, fue posible desarrollar sistemas de frenado automático [2]..

En 1980 investigadores de la universidad de Múnich desarrollaron una camioneta guiada por visión artificial, el encargado del diseño fue el investigador Erns Dickmanns. Este proyecto tuvo un éxito relativo pues logró alcanzar una velocidad de hasta 100km/h en una carretera donde no circulaban vehículos [2].

En 1987 se crearon los laboratorios HRL, en donde se construyó un vehículo capaz de recorrer más de 600 metros, creando su propia ruta y desplazándose por terrenos inestables. Este vehículo tenía instalado un Sistema de Posicionamiento Global (GPS) [2].

En 2007 el cientifico mexicano Dr. Raúl Rojas desarrolló el vehículo autónomo *AutoNOMOS*, que ha recorrido la capital alemana gracias a su moderno sistema de sensores láser, radares y cámaras de video que le permiten detectar personas, las líneas del carril, obstáculos, el estado de los semáforos y la distancia a la que se encuentra cualquier objeto. En Alemania cuenta con licencia de manejo desde el 2010. El *AutoNOMOS* ha estado en las calles de Texas, Nevada, Suiza e incluso en México [27].

En Europa, distintos países y fabricantes de automóviles se dedicaron al análisis en profundidad de los vehículos autónomos, de tal forma que en 1994 un vehículo Mercedes 500 SEL, llamado "VAmP" recorrió más de 1000 km en Paris, alcanzando una velocidad de 130 km/h, rebasando automóviles más lentos. En 2014 el modelo RS7 autónomo de Audi, alcanzó una velocidad de 240km/h en el circuito de Hockenheim en Alemania. La empresa Audi realizó una prueba de comparación al colocar un conductor 0detrás del volante, siendo éste 5 segundos más lento que el vehículo autónomo [1].

En 2015 un vehículo Audi SQ5 autónomo, controlado mediante un sistema de conducción creada por Audi y Delphi, recorrió 5400 km en un lapso de 9 días sin que los ingenieros intervinieran, salvo en dos ocasiones [1].

En 2015 Tesla Motors consiguió brindarle capacidades de autonomía a su coche eléctrico Tesla Model S. Se trató de un carro con la opción de piloto automático que le permitió al Model S cierta independencia gracias a sensores, cambios de carril y una conducción más versátil, pero sin prescindir completamente de alguien al volante. Recién hace un tiempo es que una versión mejorada del vehículo ha salido a la luz bajo el sencillo nombre de Tesla Autopilot, que no es más que el mismo Tesla Model S pero con las innovaciones necesarias que lo convierten en un coche autónomo en todo sentido de la palabra [28].

1.5.3. Niveles de autonomía

Actualmente, todos los fabricantes de vehículos incorporan a sus marcas, sistemas de control que brindan mayor seguridad y confiabilidad al conductor. Algunos de estos son; frenado de emergencia autónomo y mantenimiento de carril, haciendo sentir más apoyado al conductor.

Los sistemas de ayuda van encaminadas a la conducción totalmente autónoma. La sociedad de ingenieros del automóvil (SAE) ha realizado una clasificación de los niveles de autonomía de los automóviles, como se muestra a continuación:

- **Nivel 0.** El automóvil no cuenta con ningún sistema de automatización, solamente cuenta con algún sistema de advertencia.
- **Nivel 1.** El automóvil únicamente cuenta con sistemas como el control de crucero o la ayuda para mantenerse en el carril por el que circula.
- **Nivel 2.** Al automóvil se le puede considerar como 'semiautónomo'. El conductor debe tener atención a la conducción autónoma por si se da una situación de inseguridad. El sistema autónomo se desactivará cuando el conductor tome el control. Un ejemplo de este nivel de autonomía es el Mercedes clase 'E' con su sistema Drive Pilot.
- **Nivel 3.** El automóvil puede conducir de manera totalmente autónoma bajo condiciones controladas, por ejemplo, sobre autopistas. En este nivel aparece el

Tesla Model S, que tiene un sistema Autopilot, el cual se encuentra desactivado por defecto, el conductor puede activarlo cuando lo requiera. Este sistema verifica que el conductor se encuentre alerta y con las manos al volante (supervisando el sistema).

- **Nivel 4.** El vehículo puede circular sin la intervención del conductor, esto es para terrenos completamente definidos, donde la información proporcionada por la tecnología sea completamente confiable para la circulación del automóvil.

- **Nivel 5.** El vehículo puede conducir de manera totalmente autónoma por cualquier carretera o ciudad, siempre y cuando la ley del lugar sobre donde circula lo permita.

  Empresas que no se dedican a la fabricación de vehículos también están interesadas en este tipo de automóviles, como Microsoft que desarrolla aplicaciones con este propósito [3].

1.6. Funcionamiento

En términos generales el funcionamiento de vehículos autónomos consta de tres etapas: localización, planificación y ejecución.

- **Localización:** se encarga de ubicar el automóvil en el mapa, además de detectar cualquier objeto que esté a su alrededor, ya sea algún peatón, un animal, las líneas de vialidad de la carretera, etc. Para conseguir esto, utiliza GPS muy detallados y sensores de alta precision llamados LIDAR. Los LIDAR son escáneres de láser infrarrojo que miden el grado de reflectividad de la superficie de los objetos. Además se utilizan cámaras que permiten percibir el entorno.

- **Planificación:** se encarga de estudiar y analizar los movimientos de los automóviles que lo rodean y con base en esto tomar la mejor decisión para el movimiento del vehículo.

- **Ejecución:** tiende a realizar lo que la etapa de planificación analiza. Para evitar un posible accidente del vehículo se crea un sistema de seguridad, el cual consta de un procesador y sensores alrededor del vehículo; sensor laser de escaneo a 360 alrededor del vehículo; un procesador que lee y procesa cada dato del vehículo; sensor para medir la cantidad de revoluciones que ha dado el automóvil, de esta forma se mide distancia y a su vez se conoce la ubicación del vehículo; sensor para medir la velocidad de los vehículos de enfrente y un sensor de orientación [1].

1.7. Partes que conforman un vehículo autónomo

Para que un vehículo pueda conducirse de manera autónoma se debe contar con un sistema basado en inteligencia artificial que permita estudiar el medio que lo rodea y ejecutar las decisiones necesarias para la navegación del automóvil [7].

Algunos de los sistemas necesarios para el funcionamiento de un vehículo autónomo son los siguientes:

- **GPS.** El sistema de Posicionamiento Global o GPS. Es un sistema de navegación satelital que sirve para determinar la posición de un objeto en cualquier parte del mundo [4].

- **Sensores ultrasónicos.** Se utilizan para la detección de obstáculos en el entorno del vehículo autónomo, inicialmente su función era prevenir colisiones. Actualmente son utilizados para completar la información de otros sensores que se requieren en la percepción del vehículo autónomo [2].

- **Sensor LiDAR.** Un sistema LiDAR (Ligth Detection and Ranging) está basado en la emisión de pulsos de luz láser desde una plataforma aérea o terrestre. La medición precisa del tiempo de retorno de las porciones del pulso al sensor permite calcular la distancia que separa a éste de la superficie terrestre y de los objetos que existen sobre ella. Su funcionamiento es similar, por tanto, al

de una estación total topográfica. Dado que la posición y orientación del sensor son conocidas para cada pulso emitido, cada señal de retorno tiene unas coordenadas tridimensionales únicas, lo cual permite la captura remota de la información topográfica [5].

- **Cámaras de infrarrojo.** es un dispositivo que, a partir de las emisiones de rayos infrarrojos medios del espectro electromagnético de los cuerpos detectados, forma imágenes luminosas visibles por el ojo humano [6].
- **Estereovisión.** Son cámaras que reconstruyen una visión 3D de la carretera, lo que permite la detección de obstáculos [29].
- **Radar.** Mide la velocidad de los vehículos que circulan por delante [1].
- **Procesador.** Procesa los datos recopilados por los sensores del vehículo autónomo y regula el comportamiento del mismo; es decir, manda las señales necesarias para la ejecución óptima del automóvil [1].
- **Encoder.** Estos sensores se encargan de suministrar datos de la conducción mediante el número de rotaciones de las ruedas del vehículo [4].

  Los vehículos autónomos hoy en día aun no son 100 por ciento eficientes, se piensa que en los próximos 30 años estarán acaparando el mercado automovilístico. Realizar esto, representaría un cambio generacional, para muchas personas tendría suficientes ventajas, también es cierto que para muchas otras les traería repercusiones [1].

1.8. Ventajas de los vehículos autónomos

Cuando los vehículos autónomos sean 100 por ciento eficientes y cumplan con las condiciones legales para poder circular libremente en cada país. Las ventajas que tendrían los vehículos autónomos se muestran a continuación.

- Los tiempos de traslado se reducirían en un porcentaje definido.

- Los accidentes se reducirían en un porcentaje considerable.
- Durante el traslado será posible realizar cualquier tipo de actividades como estudiar, jugar, ver tv, navegar por internet, etc.

1.9. Desventajas de los vehículos autónomos

- Las desventajas recaen más en aquellas personas que tienen algún tipo de empresas que tienen relación directa con los automóviles que si necesitan conductor, por ejemplo, todos los seguros de autos contra accidentes desaparecerían o por obvias razones, cambiarían sus servicios.
- Al estar conectados a la red, corren el riesgo de ser hackeados.

1.10. Incertidumbres legales y aceptación del vehículo autónomo

Los vehículos autónomos han marcado un antes y un después, tras más de 100 años de autos motorizados el cambio parece aproximarse.

Las normas y leyes de circulación que rigen actualmente a los vehículos convencionales tendrán que cambiar y adaptarse a este nuevo paradigma. Posiblemente, este cambio forzado en la legislación y normativa, sea el reto más complicado para los vehículos autónomos.

Capítulo 2

Marco teórico

2.1. Modelo Ackermann

2.1.1. Sistema de dirección

El sistema de dirección es un conjunto de mecanismos capaces de orientar las ruedas directrices del vehículo para que éste sea capaz de seguir la trayectoria deseada [8].

- Características de un sistema de dirección

  La dirección es uno de los componentes más importantes en el vehículo, así como lo son la suspensión y el sistema de frenos, esto debido a que la seguridad depende de estos elementos. A continuación se enlista una serie de cualidades que debe de tener este sistema [8].

  1. **Seguridad:** depende del grado de fiabilidad del mecanismo y la calidad de los materiales empleados [8].
  2. **Suavidad:** Quiere decir que una dirección puede moverse sin la necesidad de aplicar una gran fuerza al volante si hay una perfecta alineación de las ruedas y mecanismos de enlace [9].
  3. **Precisión:** Se consigue haciendo que la dirección no sea muy dura ni excesivamente suave [8].

4. **Irreversibilidad:** El volante debe mandar el giro de las ruedas, pero por el contario, las oscilaciones que se generan en estas, no deben ser transmitidas hacia el volante [8].

2.1.2. Principio Ackermann

Cuando un vehículo gira, las trayectorias que recorren las ruedas frontales son distintas, la rueda exterior recorre una mayor distancia que la rueda interior, dado que su radio de giro es superior. Para que ambas ruedas sigan la misma trayectoria deseada, se debe cumplir que sus radios de giro converjan en un mismo punto (Centro Instantáneo de Rotación) en cualquier momento de su orientación [10].

Rudolf Ackerman descubrió y definió este principio a principios del siglo XIX. El principio de la dirección Ackerman es la relación entre la rueda frontal interior y la rueda frontal exterior de un vehículo al tomar una curva. Para formar la geometría correcta, los brazos de direccionamiento se desplazan para que la rueda interior gire un mayor ángulo que la rueda exterior. El concepto es que los ejes perpendiculares de las cuatro ruedas coincidan alrededor de un punto en común denominado Centro Instantáneo de Rotación (CIR) durante una vuelta. Si los ejes no coinciden en el mismo punto, las ruedas frontales experimentarán deslizamiento [10].

En la figura 2.1a se ilustra un esquema de dirección Paralelo, en él, las ruedas delanteras giran el mismo ángulo, lo que trae como consecuencia que los ejes de las ruedas frontales no coincidan con la prolongación del eje trasero, es decir, no convergen en un sólo centro instantáneo de rotación y las trayectorias de ambas ruedas no son concéntricas. La figura 2.1b presenta la configuración Ackerman, como se puede notar los ejes perpendiculares a las ruedas frontales coinciden con la prolongación del eje trasero en un solo CIR, por tanto, los radios de giro de cada rueda serán concéntricos en todo momento [10].

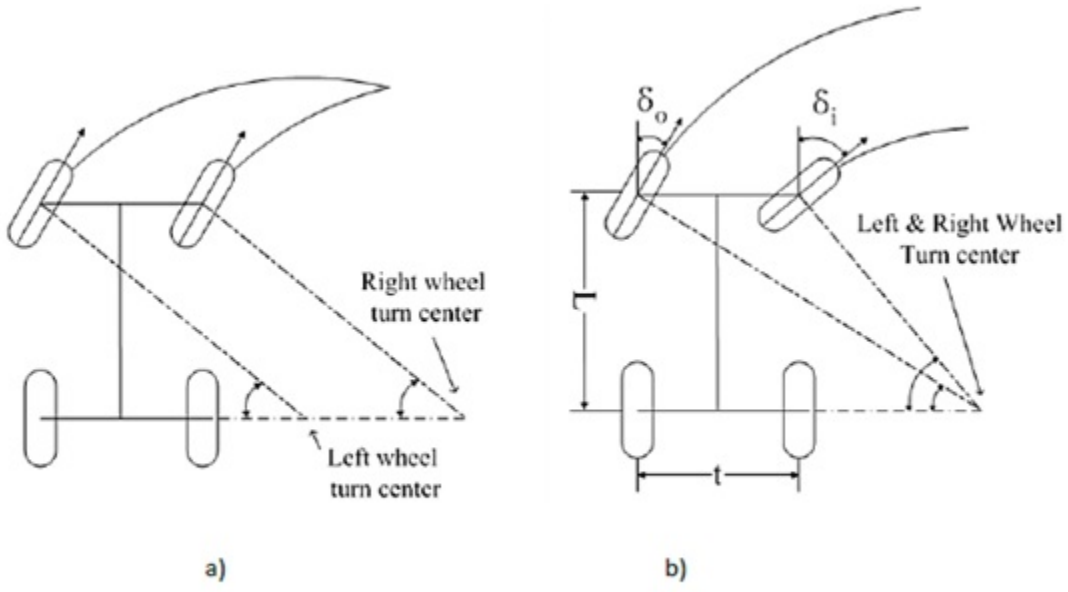

Figura 2.1: a) Sistema de dirección en paralelo; b) sistema de dirección Ackerman.

2.2. Visión Artificial

2.2.1. Visión

La visión es el sentido más importante que tiene el ser humano. Así, mientras que para el oído se tiene alrededor de treinta mil terminaciones nerviosas, en la vista hay más de dos millones. La radiación exterior recibida por el ojo debe ser transformada en señales que sean procesadas por el cerebro. El ojo es el elemento transductor mientras que el cerebro es el que procesa dicha información. [11].

2.2.2. Visión artificial

Se define "Visión Artificial" como un campo de la "Inteligencia Artificial", mediante la utilización de las técnicas adecuadas, permite el procesamiento y análisis de imágenes digitales para obtener información [12].

Los dos pilares del sistema físico de visión artificial son: el sistema de formación de las imágenes y el sistema de procesamiento de éstas.

2.2.3. Niveles de visión

El proceso de visión comienza con la obtención de una imagen hasta llegar a una descripción adecuada de acuerdo al propósito o aplicación seleccionado. El proceso consta en ir reduciendo la mayor cantidad de información posible y obtener solo la información de interés de dicha imagen.

- **Procesamiento de imagenes:** se trabaja directamente con los pixeles para extraer propiedades como orillas, gradiente, profundidad, textura, color, etc. Consiste generalmente en agrupar los elementos obtenidos, para obtener líneas y regiones, generalmente con el propósito de segmentación.

- **Analisis de imagenes:** está generalmente orientada al proceso de interpretación de los entes obtenidos en los niveles inferiores y se utilizan modelos y/o conocimiento a priori del dominio [13].

El objetivo del procesamiento y análisis de imágenes, es hacer transformaciones directamente sobre la imagen para obtener información de las propiedades físicas de los objetos que están en ella y que sean de mayor utilidad. Los principales atributos que se consideran importantes para obtener de una imagen son:

- Discontinuidades u orillas
- Color
- Textura
- Gradiente y profundidad

Existen técnicas que ayudan a mejorar la imagen, las cuales son capaces de resaltar aspectos deseados y eliminar aspectos no deseados, el ruido es un ejemplo de aspecto no deseado en una imagen.

Algunas técnicas son:

- Operaciones puntuales

- Filtrado
- Ecualización por histograma

2.2.4. Operaciones puntuales

Una operación puntual transforma una imagen de entrada E a una imagen de salida S de forma que cada pixel de la imagen de salida sólo depende del correspondiente pixel de entrada. Un ejemplo de operación puntual es la binarización.

- Binarización por umbral

  La tarea de binarización, es una típica operación puntual. Para obtener una imagen binaria se hace una transformación no lineal de la imagen de entrada, obteniéndose una imagen de salida en la cual cada pixel puede tomar alguno de dos valores 0 o 1, negro y blanco, 0 y 255, etc. Para esto, se toma un valor de umbral T (threshold).

  Matemáticamente la binarización se puede expresar como:

$$S(X,Y) = \begin{cases} 0 & \text{si } E(X,Y) \leq T \\ 1 & \text{si } E(X,Y) \geq T \end{cases} \tag{2.1}$$

  El "separar" un aspecto u objeto de interés de la imagen es la aplicación más sencilla de esta técnica, el principal problema de esta aplicación es determinar el umbral.

  En general, esta técnica se considera de prueba y error, esto se debe a que el valor de umbral no es un valor constante para todas las imágenes, además de que las condiciones de iluminación son distintas también para cada una de ellas. Es posible realizar una binarización mediante el uso de histogramas [13].

En el procesamiento de nivel intermedio, la principal técnica que se realiza es la segmentación, a continuación se tratará más a fondo esta técnica.

- Segmentación

  La segmentación es el proceso que consiste en dividir o separar una imagen digital en regiones homogéneas que tienen en común una o más características, ejemplo de estas características pueden ser el brillo o el color. El objetivo de esta técnica es facilitar un análisis o reconocimiento automático, mientras que el problema principal que es el determinar las regiones, es decir; las partes o segmentos que se consideran significativos de la imagen.

  El obtener estas regiones es de gran ayuda debido a que en lugar de trabajar con miles o millones de pixeles es posible llegar a trabajar con decenas de regiones, lo que facilita el análisis de la información de la nueva imagen.

  Las características más comunes para delimitar o segmentar regiones son: intensidad de los pixeles, textura, color, gradiente, etc. [14].

La visión de alto nivel busca encontrar una interpretación consistente de las características obtenidas en visión de nivel bajo e intermedio. Se conoce también como visión sofisticada. Básicamente la visión de alto nivel tiene que ver con reconocimiento, es decir, hacer una correspondencia de la representación interna del mundo con la información sensorial obtenida por medio de la visión [13].

2.2.5. Aplicaciones de la visión artificial

Área de producción	Aplicación
Control de calidad	Inspección de productos, identificación de piezas, etc.
Robótica	Control de soldaduras, guiado de robots.
Aplicaciones militares	Seguimiento de objetivo, vigilancia por satélites.
Control de tráfico	Matrícula de coches, tráfico viario.
Apliaciones biomédicas	Resonancias magnéticas, tomografías, etc.

Tabla 2.1: Aplicaciones de la visión artificial

2.3. Robotic Operating System (ROS)

2.3.1. ROS

La definición de ROS de acuerdo a su página oficial es:

"ROS es un sistema de código abierto, meta-operativo, para robótica. Proporciona los servicios que usted esperaría de un sistema operativo, incluyendo la abstracción de hardware, el control de dispositivos de bajo nivel, la implementación de la funcionalidad comúnmente utilizada, el paso de mensajes entre procesos y la administración de paquetes. También proporciona herramientas y bibliotecas para obtener, construir, escribir y ejecutar código en varios equipos" [15].

2.3.2. Objetivos de ROS

1. **Sencillez:** ROS está diseñado para ser tan sencillo de modo que el programa escrito en ROS se pueda utilizar en otros software.

2. **Modelo de bibliotecas:** Consiste en escribir una serie de programas con interfaces funcionales que permitan una rápida modificación.

3. **Independencia de lenguajes:** el marco de ROS es fácil de implementar en cualquier lenguaje de programación moderno. Se ha implementado en Python, C++ y LISP, además de que actualmente existen bibliotecas experimentales en JAVA y LUA.

4. **Escala:** ROS es adecuado para sistemas grandes y ejecución de procesos de gran tamaño [16].

2.3.3. Conceptos de ROS

ROS tiene tres niveles de conceptos: el nivel del sistema de archivos, nivel de computación gráfica y el nivel comunitario.

A continuación se explica cada uno de estos niveles de conceptos.

- **Sistemas de archivos**

  1. **Paquetes:** los paquetes son la unidad principal para organizar software en ROS. Un paquete puede contener procesos ejecutables (nodos), una biblioteca dependiente, conjuntos de datos, archivos de configuración, etc.
  2. **Meta-paquetes:** son paquetes especializados que solo sirven para representar un grupo de otros paquetes relacionados. Normalmente, los metapaquetes se utilizan como un marcador de posición compatible con versiones anteriores para las pilas de rosbuild convertidas.
  3. **Manifiestos de paquetes:** proporcionan metadatos sobre un paquete, incluido su nombre, versión, descripción, información de licencia, dependencias y otra información como paquetes exportados.
  4. **Repositorios:** Una colección de paquetes que comparten un sistema Version Control System (VCS) común. Los paquetes que comparten un VCS comparten la misma versión y se pueden lanzar juntos utilizando la herramienta de automatización de lanzamiento de catkin bloom. A menudo, estos repositorios se asignarán a las pilas de Rosbuild convertidas. Los repositorios también pueden contener un solo paquete.
  5. **Tipos de mensajes:** definen las estructuras de datos para los mensajes enviados en ROS.
  6. **Tipos de servicios:** definen las estructuras de datos de solicitud y respuesta para los servicios en ROS [15].

- **Computación a nivel gráfico**

  1. **Nodos:** Los nodos son procesos que realizan cómputo. ROS está diseñado para ser modular en una escala de grano fino; un sistema de control de robot usualmente comprende muchos nodos. Por ejemplo, un nodo controla un buscador láser de rangos, un nodo controla los motores de las ruedas, realiza la localización, realiza la planificación de la trayectoria, proporciona

una vista gráfica del sistema, etc. Un nodo ROS se escribe con el uso de una biblioteca de cliente ROS, como roscpp o rospy.

2. **Maestro:** proporciona el registro de nombres y la búsqueda del resto del gráfico de computación. Sin el maestro, los nodos no podrían encontrarse, intercambiar mensajes o invocar servicios.

3. **Servidor de parámetros:** permite almacenar los datos por clave en una ubicación central. Actualmente forma parte del Máster.

4. **Mensajes:** Los nodos se comunican entre sí pasando mensajes. Un mensaje es simplemente una estructura de datos, que comprende campos escritos. Se admiten los tipos primitivos estándar (entero, punto flotante, booleano, etc.), al igual que las matrices de los tipos primitivos. Los mensajes pueden incluir estructuras y matrices arbitrariamente anidadas (al igual que las estructuras C).

5. **Topicos:** Los mensajes se enrutan a través de un sistema de transporte con semántica de publicación/suscripción. Un nodo envía un mensaje publicándolo en un tema determinado. El tema es un nombre que se utiliza para identificar el contenido del mensaje. Un nodo que esté vinculado con cierto tipo de datos se suscribirá al tema apropiado. Puede haber múltiples editores y suscriptores concurrentes para un solo tema, y un solo nodo puede publicar y/o suscribirse a múltiples temas. En general, los editores y suscriptores no conocen la existencia de los demás. La idea es desacoplar la producción de información de su consumo. Lógicamente, uno puede pensar en un tema como un bus de mensajes fuertemente cifrado. Cada bus tiene un nombre, y cualquiera puede conectarse al bus para enviar o recibir mensajes siempre que sean del tipo correcto.

6. **Servicios:** El modelo de publicación/suscripción es un paradigma de comunicación muy flexible, pero su transporte unidireccional de muchos a muchos no es apropiado para las interacciones de solicitud/respuesta, que a menudo se requieren en un sistema distribuido. La solicitud/ respuesta

se realiza a través de los servicios, que se definen mediante un par de estructuras de mensajes: una para la solicitud y otra para la respuesta. Un nodo proveedor ofrece un servicio con un nombre y un cliente lo utiliza enviando el mensaje de solicitud y esperando la respuesta. Las bibliotecas cliente de ROS generalmente presentan esta interacción al programador como si fuera una llamada a un procedimiento remoto.

7. **Bolsas:** Las bolsas son un formato para guardar y reproducir datos de mensajes ROS. Las bolsas son un mecanismo importante para almacenar datos de sensores, que pueden ser difíciles de recopilar, pero son necesarios para desarrollar y probar algoritmos [15].

- **Comunidad ROS**

1. **Distribuciones:** Las distribuciones de ROS son colecciones de pilas que puede instalar. Las distribuciones desempeñan un papel similar a las distribuciones de Linux: facilitan la instalación de una colección de software y también mantienen versiones coherentes en un conjunto de software.
2. **Repositorios:** ROS se basa en una red federada de repositorios de código, donde diferentes instituciones pueden desarrollar y lanzar sus propios componentes de software de robot.
3. **La wiki de ROS:** es el foro principal para documentar información sobre ROS. Cualquiera puede registrarse para obtener una cuenta y contribuir con su propia documentación, proporcionar correcciones o actualizaciones, escribir tutoriales y más [15].

2.3.4. Sistemas operativos compatibles con ROS

Como se mencionó ROS no es un sistema operativo como tal, por tanto, a continuación se muestra una lista de sistemas operativos compatibles con ROS:

- UBUNTU
- DEBIAN

- OS X (HomeBrew)
- Gentoo
- Android NDK
- Windows

2.3.5. Distribuciones de ROS

Se ha mencionado ya el significado de una distribución de ROS en seguida se muestra una lista de las distribuciones más importantes:

- ROS Lunar
- ROS Kinetic
- ROS Jade
- ROS Indigo

  La distribución de ROS utilizada para este proyecto es ROS Kinetic.

2.3.6. Sistemas embebidos adecuados para ROS

- Raspberry pi
- Beaglebone
- Odroids
- Nvidia Jetson TK1

2.3.7. Protocolos de comunicación entre sistemas embebidos

1. SPI

   Serial Peripheral Interface (SPI) es una interfaz bus usada principalmente para enviar información entre microcontroladores y circuitos periféricos integrados.

En SPI, la señal de reloj se genera por un dispositivo con el rol de "maestro", mientras que el otro dispositivo adquiere el rol de "esclavo", se destaca que siempre hay un solo maestro, pero puede haber más de un esclavo.

2. I2C

   Inter-Integrated Circuit (I2C) es un bus de datos utilizado para la comunicación entre microcontroladores y circuitos periféricos integrados.

   La transferencia de datos se realiza desde un dispositivo denominado "maestro" hacia otro llamado comúnmente "esclavo", la comunicación se realiza a través de dos conexiones, es posible que haya más de un maestro, aunque la comunicación entre ellos no es posible.

3. UART

   Universal Asynchronous Receiver/Transmitter (UART) se requieren solo dos conexiones: una línea de transmisión (Tx) y una línea de recepción (Rx).

   En el lado de transmisión, se debe de crear el paquete de datos añadiendo bits de sincronización y de paridad. En el extremo de recepción, se debe de obtener la información de acuerdo con la tasa de transmisión y seleccionar el bit de sincronización [17].

Después de todo lo anterior se puede aclarar y afirmar que ROS no es:

- ROS no es un lenguaje de programación.
- ROS no es una librería
- ROS no es un sistema de desarrollo integrado [17].

Capítulo 3

Implementación del vehículo autónomo

El vehículo, requiere de sensores y actuadores para lograr su correcta implementación y con estos generar la autonomía del automóvil. Tanto sensores como actuadores funcionan e interactúan unos con otros, dependiendo de las tareas que se desean.

3.1. Sensores

3.1.1. Sensor

Dispositivo capaz de detectar magnitudes físicas en su entrada y transformarlas en señales eléctricas. Las señales o variables de salida pueden ser: temperatura, presión, distancia, aceleración, fuerza, resistencia eléctrica etc. [18]. Los sensores y actuadores utilizados para la implementación del vehículo autónomo se describen a continuación.

3.1.2. Sensor ultrasónico HC-SR04

En el vehículo autónomo a escala se implementaron dos sensores ultrasónicos HC-SR04.

El primero en la parte delantera del auto y otro en la parte trasera, ambos tienen la función de detectar algún objeto a una distancia determinada. Esto se consigue

porque el sensor emite pulsos de alta frecuencia de 40KHz, los cuales no son audibles por el oído humano y mide el tiempo que tardan en llegar al obstáculo y volver al sensor figura 3.1 [19].

La distancia medida se obtiene de la siguiente expresión:

$$e = \frac{(v * t)}{2} \tag{3.1}$$

donde:

e=Distancia entre el sensor y el objeto

v=velocidad del sonido en el aire (343.2 m/s)

t=tiempo de ida y regreso de los pulsos

Figura 3.1: Sensor Ultrasónico

La función que tiene el sensor ultrasónico, HC-SR04 en la implementación del vehículo autónomo colocado en la parte frontal es detectar un automóvil estático a una distancia menor a 30cm.

Al cumplirse esta condición, el sensor emite una señal al microcontrolador, el cual se encarga de accionar el servomotor MG995 y el motor DC con la finalidad de realizar el rebase del obstáculo en lazo abierto.

En el caso del sensor colocado en la parte trasera del robot, la función que tiene en el vehículo autónomo consiste en detectar objetos los cuales solo son leidos por el microcontrolador.

Los sensores colocados en la parte frontal y trasera del vehículo se muestran en la Figura 3.2.

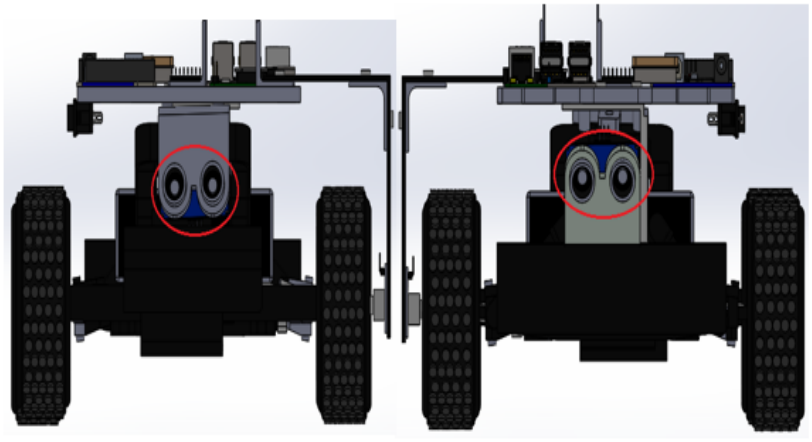

Figura 3.2: Sensor ultrasónico, parte frontal (lado derecho) y parte trasera (lado izquierdo) en CAD

Relación de ROS con el sensor ultrasónico HC-SR04

Se realizó un algoritmo de programación en Arduino para el sensor ultrasónico, el cual incluye la librería <**ros.h**>, se publicó la información (mensaje), mediante el tópico **"ultrasonic"**.

Para el sistema embebido se generó un algoritmo de programación en python, la función principal de este programa fue recibir la información enviada desde Arduino, a través del tópico **"ultrasonic"**. Los nodos en ROS son elementos muy importantes, son procesos que realizan cómputo, para este caso el nombre del nodo es **"vehiculo-autonomo"**. El nodo de este programa se muestra en la figura 3.3. Mientras que el tópico se muestra en la figura 3.4.

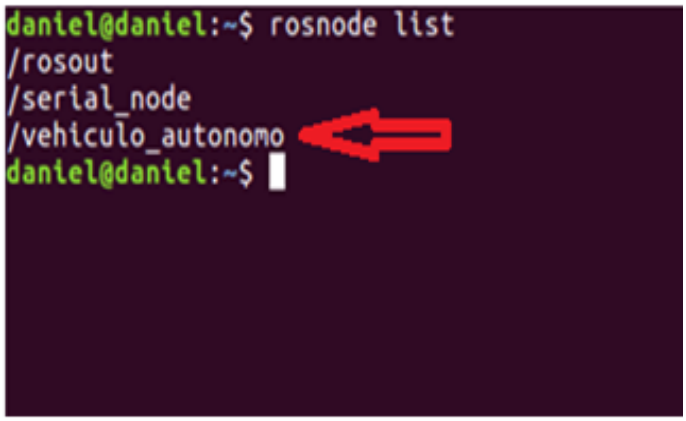

Figura 3.3: Nodo del programa.

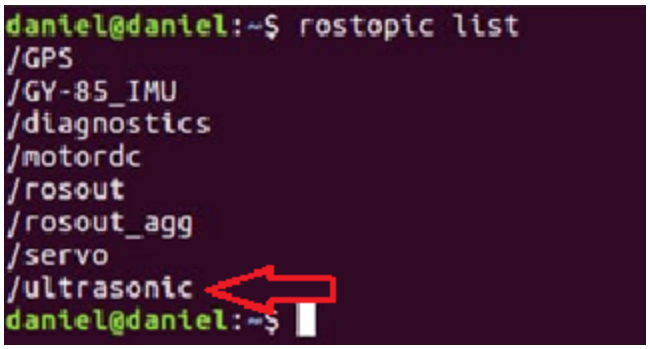

Figura 3.4: Tópico del programa.

Después, se realiza el procesamiento de la información obtenida, lo cual implica accionar el servomotor MG995 y el motor DC para obtener el resultado esperado.

3.1.3. Servomotor MG995

Para el funcionamiento adecuado de las aplicaciones del vehículo autónomo, una de las partes más importantes es el control de la dirección (*steering*), la cual se diseñó haciendo uso del servomotor MG995 (figura 3.5).

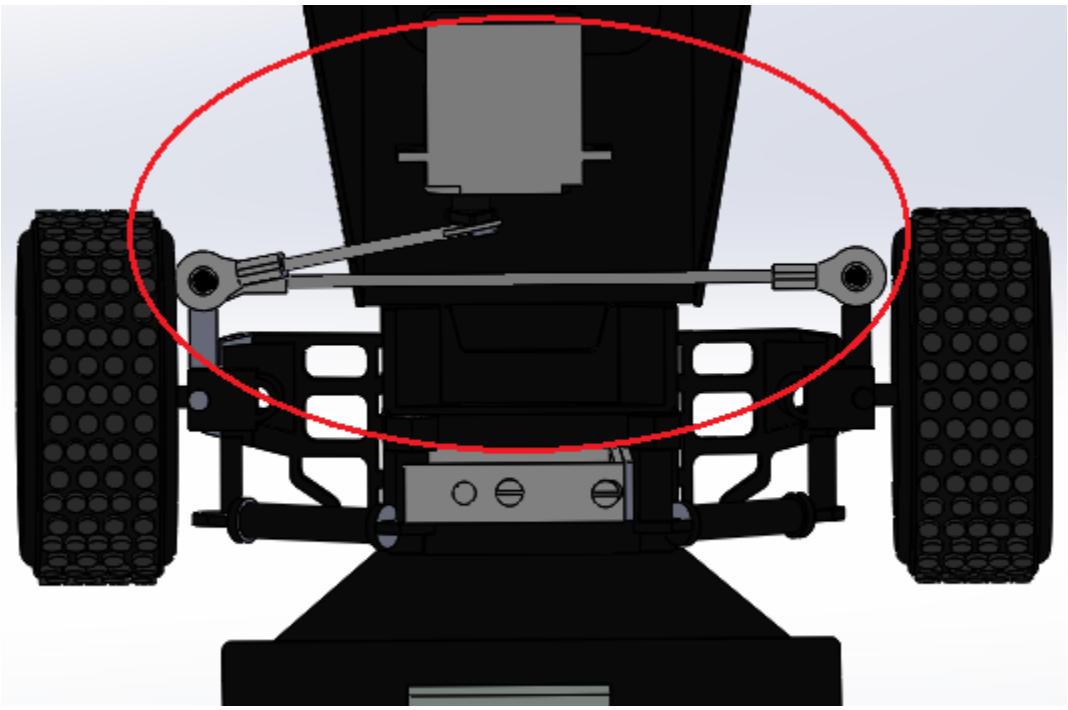

Figura 3.5: Steering del vehículo en CAD

El servomotor MG995 (Figura 3.6), es un dispositivo que dispone en su interior de un motor con un multiplicador de fuerza y reductor de velocidad, es decir, cuenta con un sistema de engranajes y está dotado de un sistema de control y un ángulo de giro es de 0-180 grados [20].

El giro del vehículo a la izquierda se realiza en el rango de 90-180 grados, el giro a la derecha está dado en el rango de 90-0 grados , mientras que los 90 grados indican que el vehículo circula en línea recta. El valor exacto de giro es directamente proporcional al tipo de controlador utilizado (tema que se abordará en el siguiente capítulo).

Figura 3.6: Servomotor MG995

Relación de ROS con el servomotor MG995

Al igual que con el sensor ultrasónico, para el servomotor se generó un programa en Arduino, el cual también incluye la librería <**ros.h**> y se generó el tópico **"servo"** de tipo *Subscriber*.

Para la Raspberry pi 3B, se creó un algoritmo de programación en python, el cual determina el valor de giro del servomotor. Mediante el tópico **"servo"** (Figura 3.7) se publica el valor del *steering*.

Los datos son recibidos por el microcontrolador, el cual se encarga de accionar el actuador cuando sea requerido.

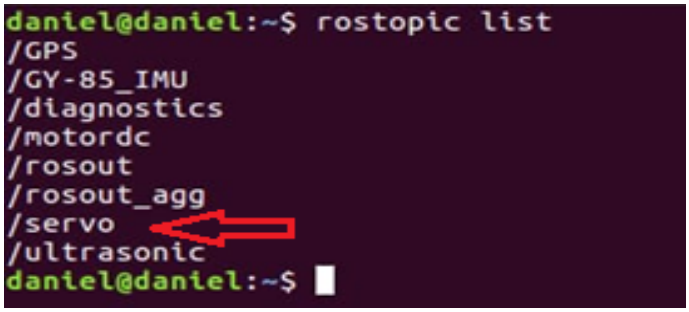

Figura 3.7: Tópico del programa.

3.1.4. Arduino Mega2560

Gran parte del procesamiento de la información se realizó mediante el microcontrolador Arduino Mega2560 (Figura 3.8), cuya capacidad es la más alta de la familia Arduino. Cuenta con 54 terminales digitales que funcionan como entrada/salida; 16 entradas analógicas, un cristal oscilador de 16 MHz, una conexión USB, un botón de reset y una entrada para la alimentación de la placa [21]. El microcontrolador se encarga de accionar sensores y actuadores en tiempo y forma y con esto garantizar un correcto funcionamiento del vehículo autónomo.

Arduino se comunica con los sensores, actuadores y el sistema embebido mediante el protocolo de comunicación I2C. Para cada uno de los sensores fue necesario realizar un algoritmo de programación ideal para el funcionamiento de cada uno de estos.

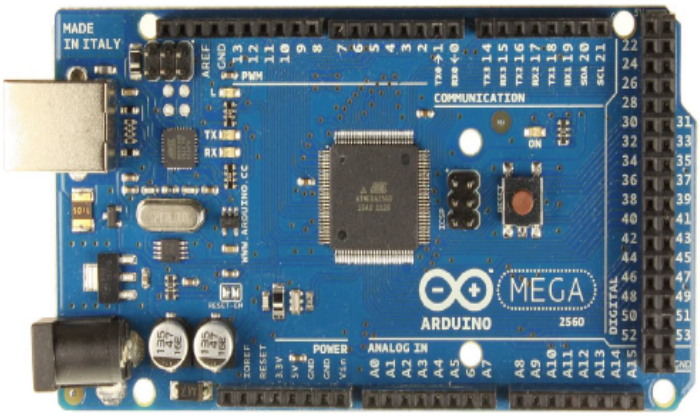

Figura 3.8: Arduino Mega2560

En la implementación del vehículo autónomo el Arduino Mega se colocó en la parte superior trasera como se muestra en la Figura 3.9.

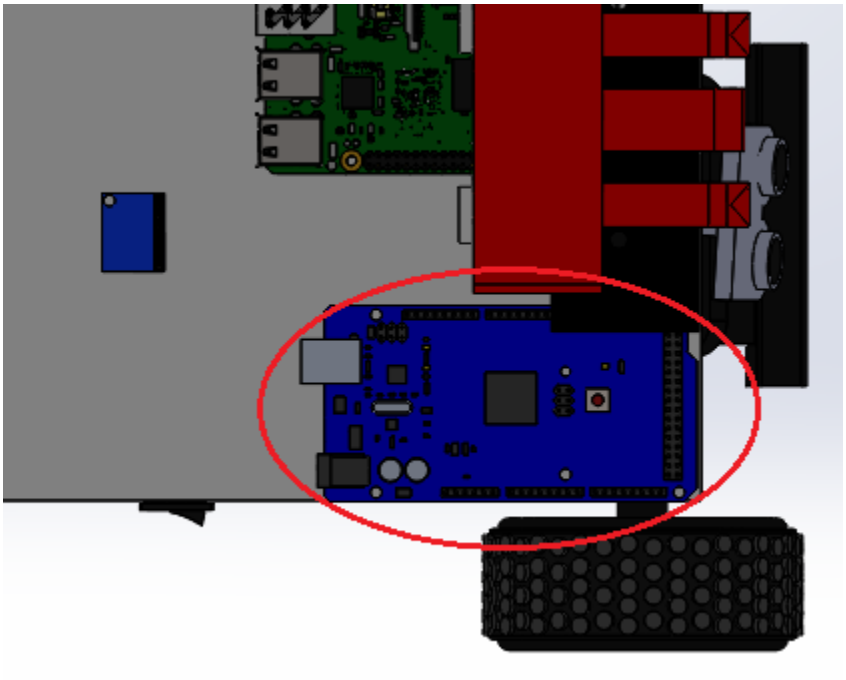

Figura 3.9: Arduino Mega en el vehículo autónomo en CAD

3.1.5. Relación de Arduino con ROS

Arduino es el microcontrolador, compatible con UBUNTU, encargado de accionar los sensores y actuadores del vehículo autónomo, los algoritmos de programación son cargados desde UBUNTU.

Arduino tiene relación directa con el sistema embebido, debido a la comunicación que hay entre ellos. De acuerdo al tópico que se ocupe, Arduino puede funcionar como *Publisher* o como *Subscriber*. *Publisher* cuando el microcontrolador envía datos al sistema embebido para ser procesados y *Subscriber* para recibir información y de esta forma accionar dispositivos electrónicos.

3.1.6. Puente H L298N

Para el movimiento del vehículo autónomo es necesaria la implementación de un motor de corriente directa (CD), el cual no puede ser accionado de manera directa, por lo que es necesario hacer uso de un *driver* para la manipulación de la velocidad y la dirección del vehículo.

El modulo para el accionamiento del actuador es el puente H L298N (Figura 3.10) el cual es un controlador de puente completo de alto voltaje, alta corriente diseñado para aceptar los niveles lógicos TTL estándar. Cuenta con dos canales de puente H, con ello se pueden controlar dos motores DC o un motor paso a paso, controlando el sentido de giro y velocidad [22].

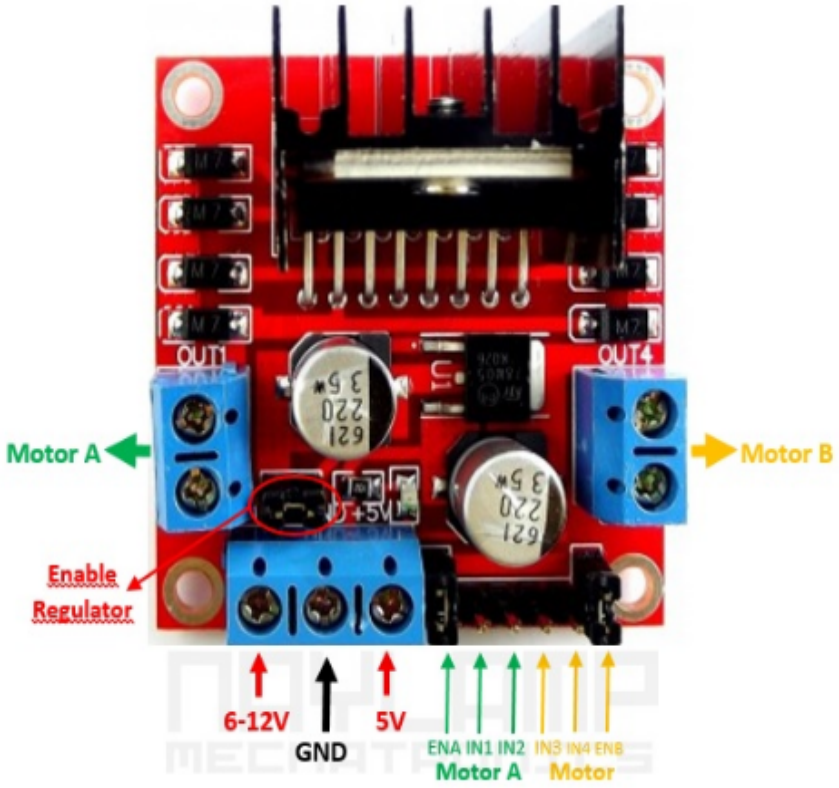

Figura 3.10: Puente H L298N

Para la instrumentación del vehículo, el puente H se colocó a la mitad del vehículo (Figura 3.11). Se alimenta con una fuente externa de 14.8V; este módulo recibe una señal del PWM desde el microcontrolador, que se encarga de controlar la velocidad

del actuador y recibe dos señales digitales en sus pines de habilitación, encargadas de la dirección del motor DC.

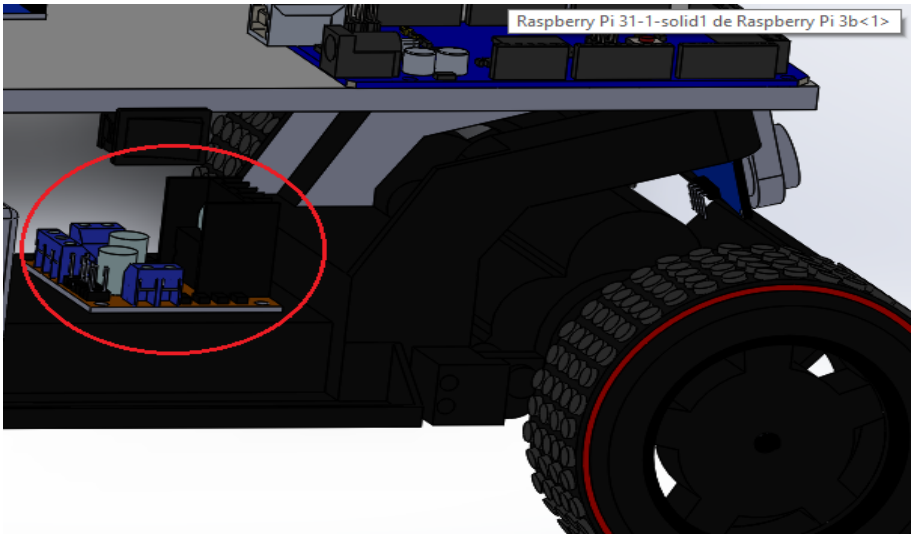

Figura 3.11: Puente H en el vehículo autónomo en CAD.

Relación del ROS con el puente H

Para el puente H se generó un programa en Arduino, en el cual también se incluyó la librería <**ros.h**>, se generó el tópico **"motordc"** de tipo *Subscriber*, esto es debido a que Arduino se encarga de accionar el actuador, en este caso el motor DC.

Para el sistema embebido se creó un algoritmo de programación en python, donde el nodo tiene por nombre **'vehiculo-autonomo'**. Mediante el tópico **"motordc"** se envía el valor del PWM que va de 0 a 255, los datos son enviados al microcontrolador mediante *Publisher*, el cual hace un escalamiento de 0-255 a 0V-5V y así poder controlar la velocidad del motor, además de la dirección. El tópico se muestra en la Figura 3.12.

3.1.7. Motor DC

Para el movimiento del vehículo se utilizó un motor DC, cabe mencionar que el actuador ya se encontraba implementado en el cuerpo de un vehículo de control remoto, el cual se modificó y se convirtió en vehículo autónomo.

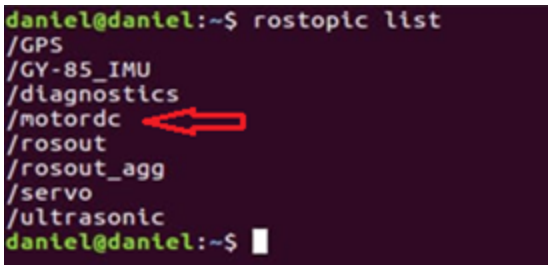

Figura 3.12: Tópico del programa

El motor DC es una máquina capaz de transformar energía eléctrica, suministrada en forma de corriente continua, en energía mecánica [23]. En el vehículo autónomo, el motor DC se muestra en la Figura 3.13.

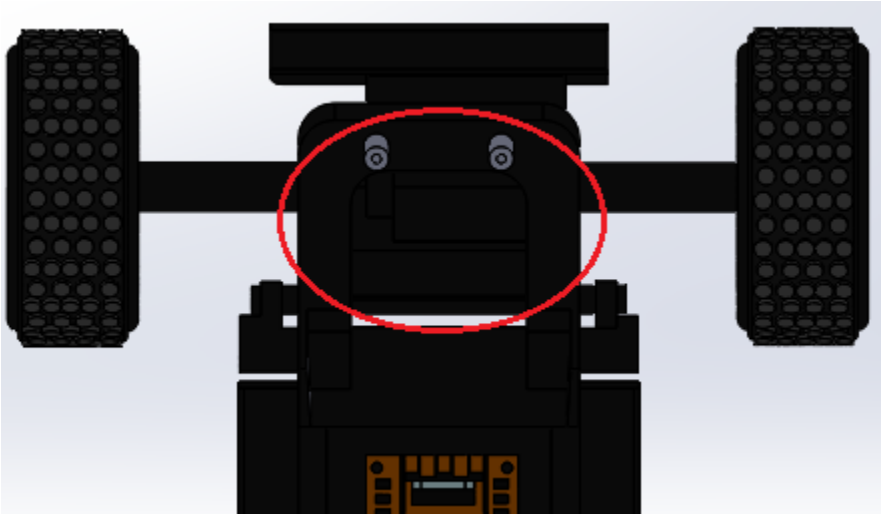

Figura 3.13: Motor DC del vehículo autónomo en CAD

El motor no tiene conexión directa con el microcontrolador y el sistema embebido, es decir; que esté conectado a alguno de estos dos dispositivos. El motor DC se acciona cuando recibe señal del módulo Puente H, el cual obtiene señales de habilitación del Arduino.

En este caso la relación con ROS es de forma indirecta. El módulo puente H es el que tiene relación directa con ROS.

3.1.8. GPS GY-GPS6MV2

Es común que los dispositivos inteligentes incluyan un módulo GPS en su sistema. Como se mencionó en el capítulo 1, el GPS es un sistema de navegación satelital que sirve para encontrar la posición de algún dispositivo. El GPS funciona mediante una red de 27 satélites (24 operativos y 3 de respaldo) en órbita a 20,200km sobre el globo terráqueo. El método para calcular la posición se le conoce como "triangulación" [4]. El modulo GPS que se implementó en el proyecto se muestra en la Figura 3.14.

Figura 3.14: GPS GY-GPS6MV2

En el vehículo autónomo se implementó un módulo GPS en su modelo GY-GPS6MV2 (Figura 3.15) con la finalidad de conocer la ubicación del vehículo autónomo sobre el globo terráqueo. El modulo es controlado con Arduino, el cual se comunica con el sistema embebido para procesar los datos.

Relación de ROS con el GPS

Mediante UBUNTU, para el módulo GPS se generó un programa en Arduino, en el que también incluye la librería <**ros.h**>, se generó el tópico **"GPS"** de tipo *Publisher*, esto es debido a que Arduino se encarga de hacer la lectura del dispositivo. El tópico se muestra en la Figura 3.16.

Para el sistema embebido se creó un programa en python, donde el nodo tiene por nombre **'vehiculo-autonomo'**. Mediante el tópico **"GPS"** se reciben los datos obtenidos por el modulo mediante un *Subscriber*, los cuales son mostrados en pantalla,

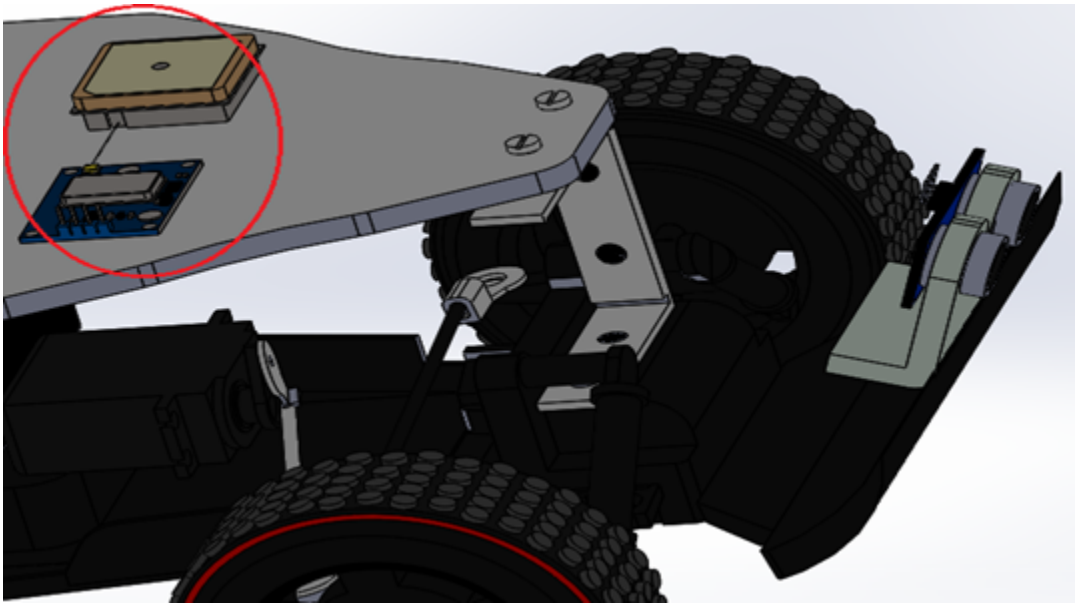

Figura 3.15: Implementación del GPS en el vehículo autónomo en CAD

indicando la posición del vehículo autónomo.

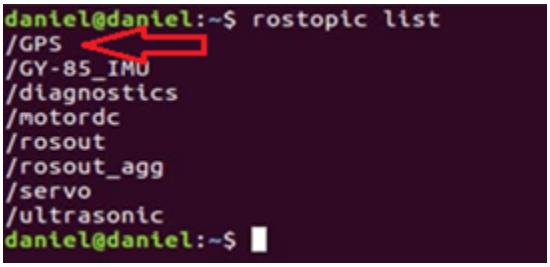

Figura 3.16: Tópico del programa

3.1.9. Cámara

Para obtener el funcionamiento correcto de las aplicaciones autónomas del vehículo, se instaló una cámara en la parte trasera del vehículo (Figura 3.17), a una distancia de 20 cm de altura. Esto con el fin de obtener una visualización adecuada del entorno de trabajo.

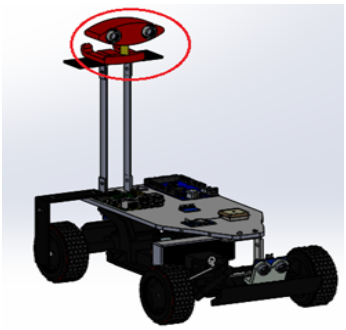

Figura 3.17: Cámara en vehículo autónomo en CAD.

Relación de la cámara con ROS

Se generó un algoritmo de programación en python, para la obtención y el procesamiento de las imágenes, dichas imágenes se obtuvieron mediante el comando **cv2.capture()** y se leyeron mediante la instrucción **cap.read()**, posteriormente mediante el comando **cv2.imshow()** se mostraron en pantalla como se muestra en la Figura 3.18.

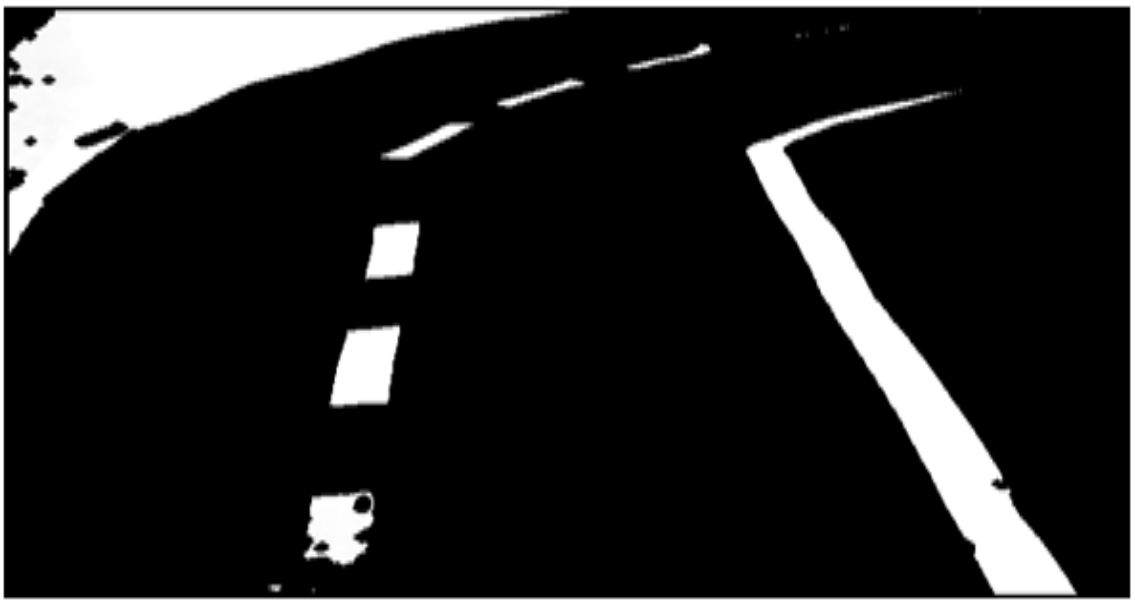

Figura 3.18: Imagen capturada por la cámara

3.1.10. Raspberry Pi 3B

Como todo sistema inteligente, necesita de un ordenador principal para procesar información y ejecutar acciones necesarias.

En el vehículo autónomo se implementó como procesador principal una Raspberry pi en su modelo 3B (Figura 3.19) que es un ordenador de placa reducida, ordenador de placa única u ordenador de placa simple (SBC) de bajo costo desarrollado en el Reino Unido por la Raspberry Pi Foundation, con el objetivo de estimular la enseñanza de informática en las escuelas. Cuenta con un Quad-Core, pero pasa de 900MHz a 1.20GHz.

Mantiene la RAM en 1GB. Su mayor y principal novedad de este modelo, fue la inclusión de Wi-Fi y Bluetooth (4.1 LowEnergy) sin necesidad de incluir adaptadores [24].

Figura 3.19: Raspberry pi 3B

La función principal de la Raspberry en el vehículo autónomo es procesar, enviar y recibir información. No se utilizan las I/O de la Raspberry debido a que se optó por usar las I/O del Arduino, con el fin de no saturar la memoria del microprocesador. Para un correcto funcionamiento de este sistema embebido, fue necesario realizar la instalación de un sistema operativo; el SO que se instaló fue Ubuntu Mate, esto per-

mitió realizar la instalación de distintas herramientas (software), como lo son OPEN CV, IDE de Arduino, ROS, etc.

La relación directa que existe entre el microprocesador y el microcontrolador se logra mediante ROS, los algoritmos de programación de control del vehículo autónomo fueron generados desde un editor sobre Ubuntu, el lenguaje de programación utilizado es Python, que se comunica con los programas generados en Arduino mediante el uso de "tópicos". La Raspberry Pi 3 modelo B en el vehículo autónomo se muestra en la Figura 3.20.

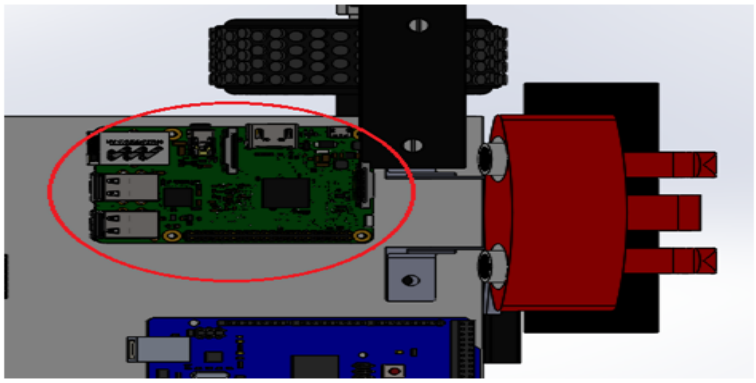

Figura 3.20: Raspberry pi 3B en vehículo autónomo en CAD

3.1.11. Encoder KY-040

Para sensar la velocidad en rpm del vehículo autónomo se implementó un encoder en la rueda trasera del vehículo. El dispositivo utilizado para realizar la medición es el Encoder KY-040 (Figra 3.21), el cual es un dispositivo de entrada giratorio (perilla) que proporciona una indicación de cuánto ha sido girada la perilla, así como la dirección de giro. Un encoder giratorio tiene un número fijo de posiciones por revolución. Estas posiciones son medidas fácilmente por pequeños clicks cuando gira el encoder. Éste módulo tiene treinta de estas posiciones [25].

El sensor se implementó con la finalidad de tener un registro de la velocidad a la que circula el automóvil. La Figura 3.22. muestra el sensor acoplado al vehículo autónomo.

Figura 3.21: Encoder KY-040

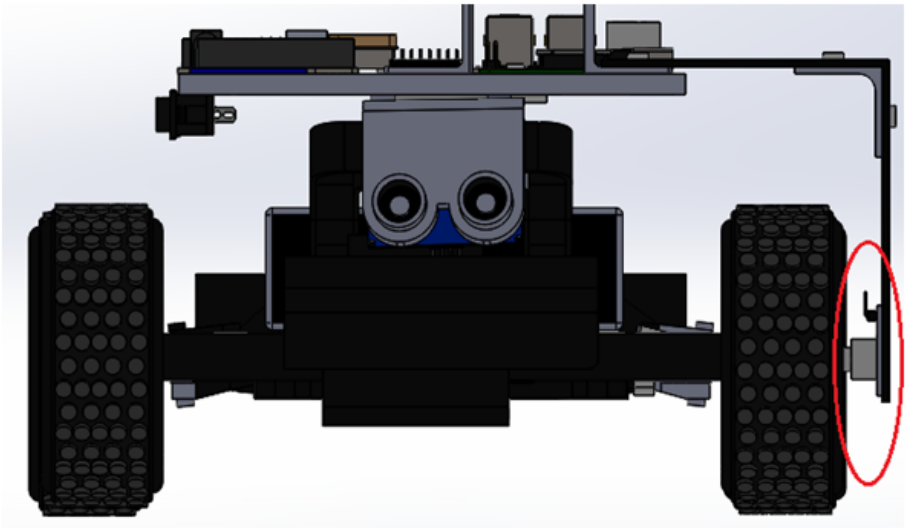

Figura 3.22: Encoder KY-040 en el vehículo autónomo

Relación con ROS del encoder KY-040

El encoder tiene una relación directa con ROS, mediante UBUNTU se generó un código en Arduino para obtener la velocidad del robot en rpm, en el programa se agregó la librería <**ros.h**>, encargada de la comunicación. La lectura de los datos se hace mediante Arduino, posteriormente son enviados a un código en Python a través de un Publisher mediante el tópico **"encoder"**; en python se hace la recepción de datos usando un Subscriberademás del mismo tópico. Una vez obtenidos los datos, haciendo uso de una terminal en ROS los datos de la velocidad se muestran en pantalla de la Raspberry.

3.1.12. IMU GY-85

En los vehículos autónomos, el conocer datos como aceleración lineal, velocidad angular y posiciones angulares con respecto al norte magnético, son de gran importancia para garantizar una correcta navegación del vehículo autónomo, en este caso la obtención de estos datos tienen una finalidad más simple la cual es mostrar los datos leídos en pantalla. Para la adquisición de los datos mencionados anteriormente, se implementó la IMU GY-85 (Figura 3.23), que es módulo compacto que incluye un giroscopio, acelerómetro, brújula digital, y un sensor de presión barométrica / temperatura. Todos los sensores individuales son accesibles a través del protocolo de comunicación I2C lo que sólo necesita 4 conexiones para acceder a todos los sensores. SDA, SCL, tierra y VCC.

El sensor es ideal para aplicaciones de robótica, medición de vibración, sistemas de medición inercial (IMU), detector de caídas, sensor de distancia y velocidad, vehículosaéreos no tripulados (UAVs), entre otras [26].

Figura 3.23: IMU GY-85

En la instrumentación del vehículo autónomo, el sensor GY-85 se colocó justo en el centro del robot (Figura 3.24). La principal función de la IMU es tomar los datos otorgados por el magnetómetro HMC5883L, el cual se calibró previamente como se muestra a continuación.

3.1.13. Calibración magnetómetro HMC5883L

Para la calibración del magnetómetro, se realizó un programa en el software Matlab para la captura de datos medidos por el sensor los cuales fueron graficados, para la recopilación de datos, el sensor fue girado en todas las direcciones del norte magnético

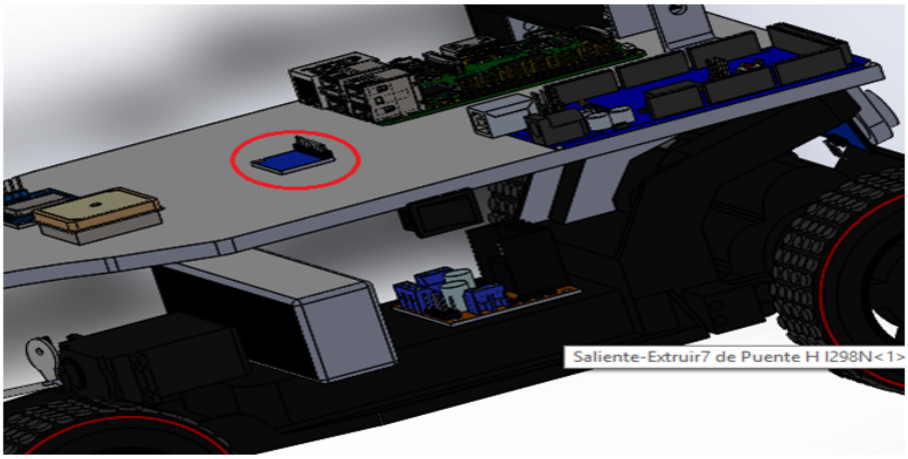

Figura 3.24: GY-85 en vehículo autónomo en CAD

de la Tierra. En la Figura 3.25 se muestra que la gráfica no es una circunferencia perfecta además se encuentra fuera del origen.

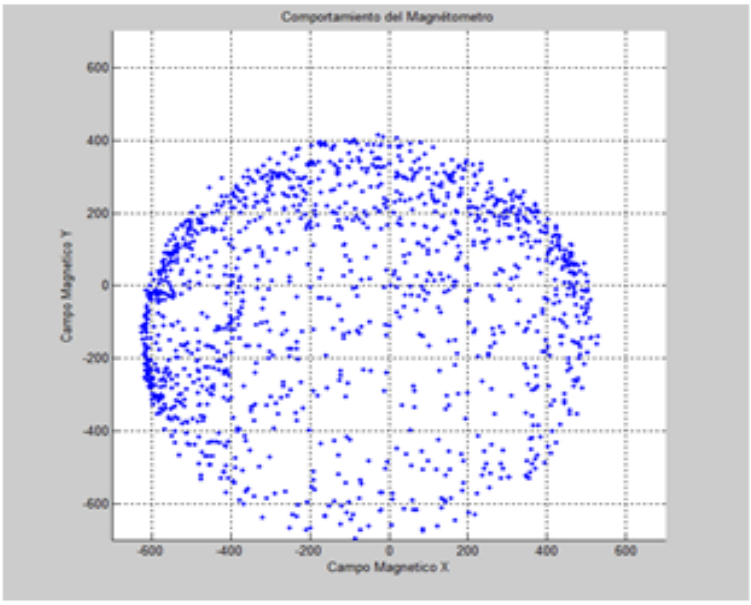

Figura 3.25: Magnetómetro sin calibrar

El proceso de calibración consiste en determinar el desplazamiento del centro de la circunferencia con respecto al origen del sistema de referencia, un factor de escala para el eje X fue calculado para mantener la misma longitud del eje Y.

Los datos para la calibración del sensor fueron:

$X_M = 534$, $Y_M = 417$

X_m = -623, Y_m = -691

donde:

X_M es el punto máximo de la circunferencia, sobre el eje X.

X_m es el punto mínimo de la circunferencia, sobre el eje X.

Y_M es el punto máximo de la circunferencia, sobre el eje Y.

Y_m es el punto mínimo de la circunferencia, sobre el eje Y.

La coordenada del centro de la esfera en el eje x está dado por la ecuación 3.2.

$$C_X = (X_M - X_m)/2$$

$$C_X = (534 - (-623))/2$$

$$C_X = 578.5 \tag{3.2}$$

La coordenada del centro de la esfera en el eje Y está dado por la ecuación 3.3.

$$C_Y = (Y_M - Y_m)/2$$

$$C_Y = (417 - (-691))/2$$

$$C_Y = 554 \tag{3.3}$$

Por lo tanto, las coordenadas del centro de la esfera son:

$$(C_X, C_Y) = (578.5, 554) \tag{3.4}$$

El siguiente paso consistió en calcular la distancia que existe entre el centro de la circunferencia sin calibrar, al origen.

La distancia que se debe desplazar sobre el eje X es:

$$d_x = 623 - 578.5 = 44.5 \tag{3.5}$$

La distancia que se debe desplazar sobre el eje Y es:

$$d_y = 691 - 554 = 137 \tag{3.6}$$

El factor de escala se calculó de la siguiente manera:

El diámetro en X es:

$$U_X = X_M - X_m$$

$\mathrm{U}_X = 534 - (-623)$

$$U_X = 1157 \tag{3.7}$$

El diámetro en Y es:

$$U_Y = Y_M - Y_m$$

$$U_Y = 417 - (-691)$$

$$U_Y = 1108 \tag{3.8}$$

El factor de escala consiste en lograr que UX=UY. Entonces:

$$U_X = (\alpha * U_Y)$$

$$\alpha = U_X/U_Y$$

$$\alpha = 1157/1108$$

$$\alpha = 1.044 \tag{3.9}$$

Por lo tanto las ecuaciones que dan como resultado la calibración del magnetómetro son:

$$m_x = (x + d_x)\alpha$$

$$m_y = (y + d_y)$$

donde:

x, y son los valores del sensor.

d_x, d_y son los desplazamientos del dentro de la esfera.

α es el factor de escala.

Sustituyendo los valores obtenidos las ecuaciones anteriores quedan:

$$m_x = (x + 44.5) * 1.044 \tag{3.10}$$

$$m_y = (y + 137) \tag{3.11}$$

En la figura 3.26 se muestra la gráfica de la calibración del magnetómetro HMC5883L, es decir, se puede observar y comparar con la gráfica que se obtuvo previo a la calibración.

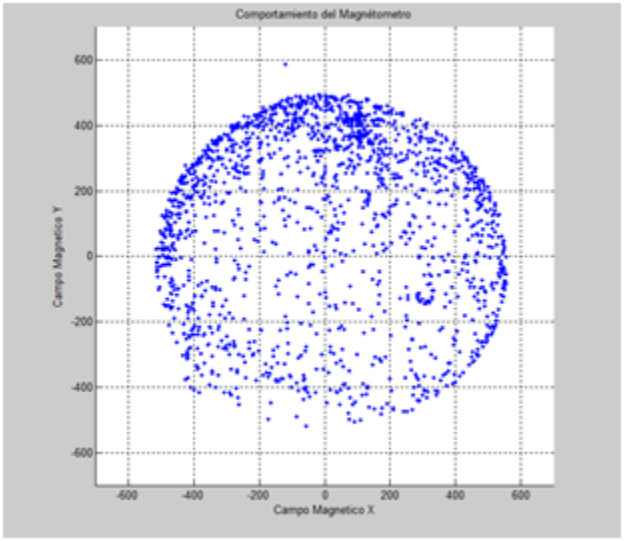

Figura 3.26: Magnetómetro calibrado

Relación del magnetómetro HMC5883L con ROS

Para el funcionamiento de este sensor en ROS, se creó un programa en Arduino mediante UBUNTU, el objetivo es recopilar los datos del magnetómetro HMC5883L. Para la comunicación con ROS se incluyó la librería <**ros.h**>, los datos se enviaron a un código en python usando un Publisher, a través del tópico **"magnetómetro"**.

En python se reciben los datos usando un Subscriber, con el mismo tópico usado en Arduino **("magnetómetro")**, mientras que el nodo es **"vehiculo-autonomo"**. De esta manera haciendo uso de una terminal desde ROS, es posible mostrar los datos en pantalla de la raspberry.

3.1.14. Batería LIPO

Se colocó una batería LIPO, de 4 celdas a 14.8V y 1800 mAh en el vehículo autónomo (figura 3.27), para la alimentación de los siguientes componentes:

- Arduino MEGA
- Puente H

La alimentación de los demás sensores se realizó a través de Arduino MEGA.

Figura 3.27: Bateria LIPO

3.1.15. RPi Power Pack V1.2

Para la alimentación de la Raspberry se utilizó una batería independiente (Figura 3.28), con el fin de aprovechar el máximo rendimiento del dispositivo.

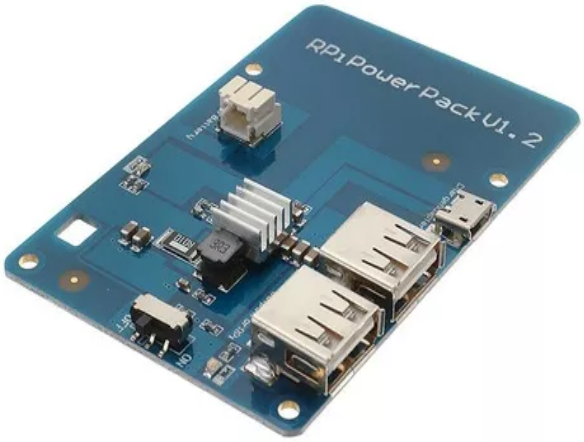

Figura 3.28: Batería RPi Power Pack

Capítulo 4

Control del vehículo autónomo

En este capítulo se describe el sistema de control del vehículo autónomo, el cual se encarga de realizar la manipulación del sistema de dirección (Steering), a través del servomotor.

4.1. Aplicaciones autónomas a controlar

4.1.1. Primera aplicación: Seguimiento de trayectorias

Consiste en realizar el seguimiento de una trayectoria bien definida, de manera autónoma (Figura 4.1).

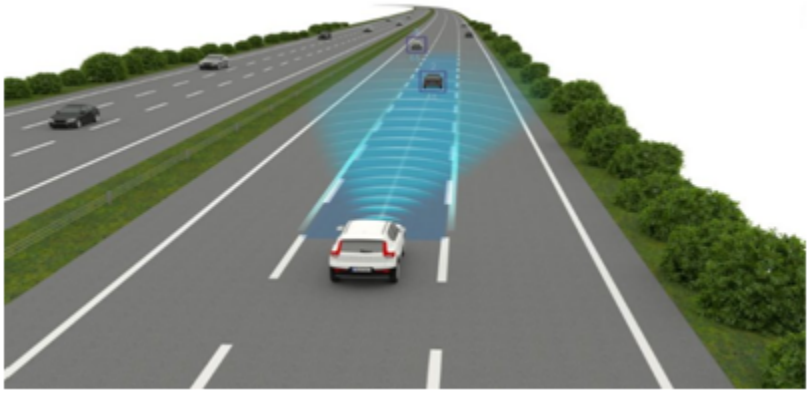

Figura 4.1: Vehículo autónomo en trayectoria definida. [Imagen de vehículo autónomo]. Recuperado de https://tecvolucion.com/que-es-conduccion-asistida/

Para ello es importante el diseño de un controlador, encargado de manipular el sistema. Más adelante se llevará a cabo el diseño de dos controladores, un proporcional (P) y un proporcional derivativo (PD). La programación de los controladores se realiza de manera independiente, con el propósito de hacer un análisis comparativo.

4.1.2. Segunda aplicación: Rebase en lazo abierto de un obstáculo estático sobre una línea recta.

Consiste en realizar el rebase de un obstáculo en lazo abierto (Figura 4.2), de igual forma es necesario desarrollar el diseño de un sistema de control. Se realizarán dos controladores, uno proporcional (P), y otro proporcional derivativo (PD). Cabe mencionar que el sistema de control solo es utilizado para el seguimiento de trayectorias. El rebase del obstáculo se realiza en lazo abierto. Los controladores P y PD, se ejecutan de manera independiente, con la intención de analizar el comportamiento de ambos.

Figura 4.2: Rebase con vehículo autónomo. [Rebase autónomo]. Recuperado de https://revistacar.es/tag/vehiculo-autonomo/feed/

4.2. Identificación del entorno

El vehículo sigue trayectorias de manera autónoma, siempre y cuando estas sean totalmente definidas, con la ayuda de una pista que es visualizada e identificada por el vehículo autónomo. A través del sensor de visualización se captura una imagen original del entorno; es decir, sin ser procesada (Figura 4.3).

Figura 4.3: Captura del entorno de trabajo

Las imágenes están constituidas por pixeles, la resolución de la cámara del vehículo autónomo es de 640 x 480 pixeles. Para obtener un mejor procesamiento de las imágenes se redujo a una resolución de 320 x 240 pixeles. Cada pixel tiene asociado un respectivo valor RGB lo cual genera una imagen a color.

Para el procesamiento de la imagen reducida es necesario hacer una transformación de ella a escala de grises. Esta transformación no es suficiente para una ejecución correcta de las aplicaciones autónomas del vehículo.

En el Capítulo 2 se mencionaron las operaciones puntuales, tal es el caso de la binarización. La imagen binaria (Figura 4.4) se obtuvo fijando un valor de umbral de 75.

Figura 4.4: Imagen binarizada de la trayectoria

4.2.1. Determinación de la ROI (Region of Interes o ROI por sus siglas en inglés)

Una región de interés (ROI) es parte de una imagen que se desea analizar.

Sobre la imagen binaria se definió una ROI con un rango en "x" de 160-320 pixeles mientras que en "y" se estableció en el pixel 200. Como se muestra en la Figura 4.5.

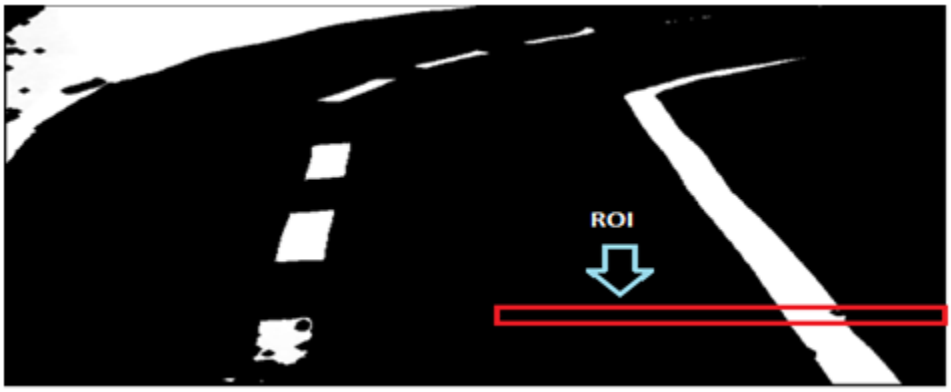

Figura 4.5: Región de interés (ROI) de la imagen.

Esta región es la más importante, pues se analizan las condiciones que determinan el valor de giro del servomotor y así generar la dirección correcta del vehículo.

4.3. Error

4.3.1. Punto de referencia

La trayectoria a seguir por el vehículo autónomo está dada por una línea blanca (Figura 4.6), sobre la cual se estableció un punto de referencia con coordenadas (xref,yref) de (240, 200) pixeles.

La ubicación del punto de referencia sobre la línea blanca, se realiza de forma manual, ya que el control de la dirección se realiza sobre esta área. El punto de referencia (color azul) se muestra en la Figura 4.7.

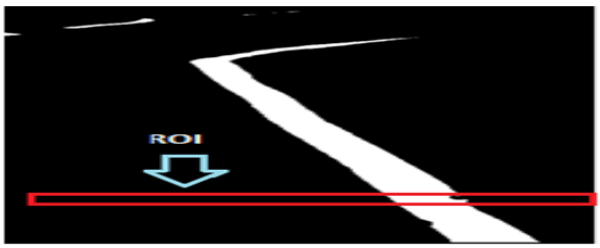

Figura 4.6: Línea blanca de referencia de la trayectoria

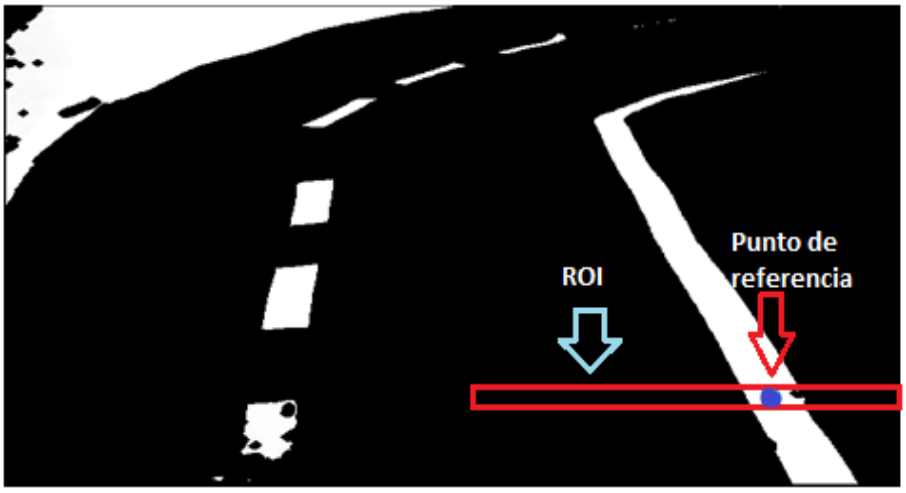

Figura 4.7: Punto de referencia sobre la ROI

4.3.2. Punto actual

El punto actual es la lectura en cada instante de tiempo que realiza la cámara sobre la ROI, este punto está dado en color verde como se muestra en la Figura 4.8. Es importante señalar que la cámara se debe encontrar en la posición correcta, de modo que sea posible visualizar el camino.

4.3.3. Cálculo del error

Para realizar el diseño del controlador del sistema de dirección del vehículo autónomo, es fundamental calcular el error en intervalos tiempo.

Matemáticamente el error se define como:

$$Error = x_{ref} - x_{act} \tag{4.1}$$

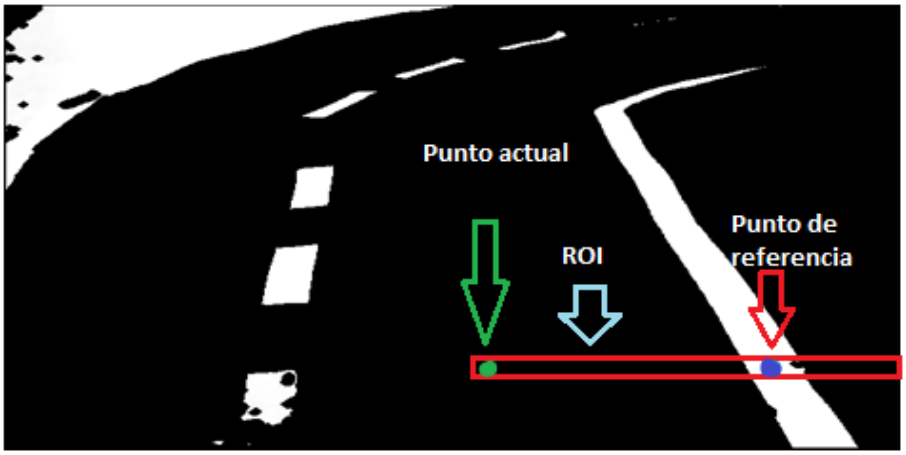

Figura 4.8: Punto actual sobre la ROI

donde:

x_{ref} = punto de referencia

x_{act} = punto actual

Existen 3 casos a considerar para el error:

Caso 1.

$$Error = \mathbf{0}$$

Cuando:

$$x_{ref} = x_{act}$$

Caso 2.

$$Error < \mathbf{0}$$

Cuando:

$$x_{ref} < x_{act}$$

Caso 3.

$$Error > \mathbf{0}$$

Cuando:

$$x_{ref} > x_{act}$$

4.4. Controlador proporcional (P)

Como se mencionó anteriormente, el cálculo del error es fundamental para el diseño de cualquier controlador. Para las aplicaciones autónomas del vehículo se diseñó un controlador proporcional, como se muestra en el diagrama de bloques de la figura 4.9

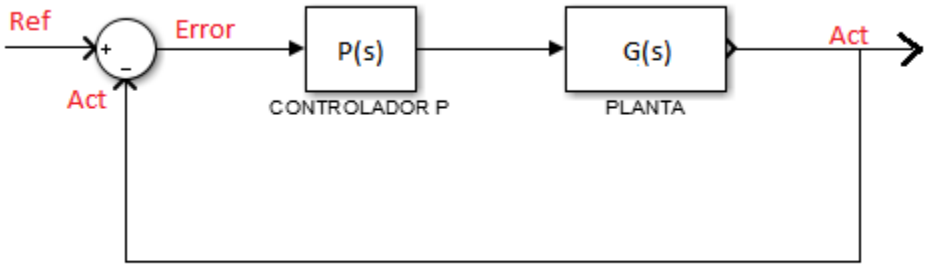

Figura 4.9: Diagrama de bloques del controlador P

Idealmente el rango de giro del servomotor encargado de la dirección del vehículo va de 0-180 grados.Para el control de giro hacia la derecha va de 0-90 grados, el control hacia enfrente es un valor constante de 90 grados y el control de giro hacia la izquierda va de 90-180 grados.

El servomotor que se implementó en el vehículo no reacciona idealmente. Su rango de respuesta va de 0-170 grados. El control hacia la derecha va de 0-95 grados, mientras que el control hacia la izquierda va de 85-170 grados. El valor del servomotor para el control del vehículo cuando va en línea recta es de 90 grados.

Con base en los rangos mencionados del servomotor, con la identificación de la ROI y el cálculo del error, resultan suficientes para la implantación un controlador proporcional.

Matemáticamente el controlador proporcional se define como:

$$u(t) = k_p e(t) \tag{4.2}$$

Donde:

$u(t)$ es el controlador

k_p es la ganancia proporcional del controlador

$e(t)$ es el error del sistema de dirección

Anteriormente se mencionó como calcular el error $e(t)$, resta por determinar la ganancia k_p. De acuerdo a los rangos de giro de la dirección del vehículo autónomo (derecha e izquierda), fue necesario realizar el diseño de dos controladores proporcionales, uno para cada sentido de giro. Los cálculos se muestran a continuación.

4.4.1. Controlador para giro hacia la izquierda

De acuerdo con la determinación de la ROI, está definida en el rango de 160-320 pixeles, sobre el eje X. Para el control hacia la izquierda el rango va desde el pixel 160 hasta el pixel 240, esto debido a que el punto de referencia se estableció en 240 y el inicio de la ROI en 160. En este intervalo es posible conocer el tamaño del error.

Matemáticamente el error máximo en este rango está definido por:

$$Error_M = 240 - 160 \tag{4.3}$$

$$Error_M = 80 \tag{4.4}$$

Lo anterior significa que para este rango el error puede tomar hasta 80 valores distintos. De acuerdo con esto y con el rango de giro hacia la izquierda del servomotor, es posible generar una gráfica que muestre el comportamiento del sistema (Figura 4.10)

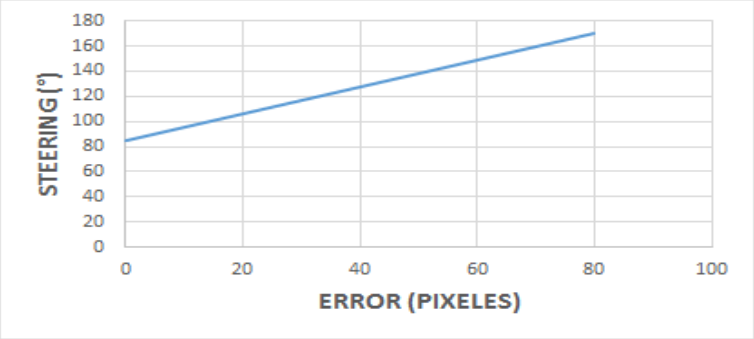

Figura 4.10: Gráfica Error vs Steering

La figura anterior muestra una línea recta, la ecuación general que la representa está dada de la siguiente forma.

$$y = m \cdot x + b \tag{4.5}$$

De acuerdo a la ecuación del controlador proporcional (ecuación 4.2), es posible hacer una comparación con la ecuación 4.5, se agrega la ordenada al origen (b) y se calcula el valor de la pendiente "m", en comparación con la ecuación 4.2 corresponde al valor de la ganancia proporcional (k_p) del controlador. La variable independiente sobre el eje "x" corresponde al error, mientras que la variable independiente "y" corresponde al steering.

La pendiente m se calcula de la siguiente forma.

$$m = \frac{(y_2 - y_1)}{(x_2 - x_1)} \tag{4.6}$$

$$m = \frac{170 - 85}{80 - 0}$$

$$m = 1.0625$$

La ordenada al origen "b" es 85 que se obtiene directamente de la gráfica.

Sustituyendo los datos en la ecuación 4.5 .

$$y = 1.0625 \cdot x + 85$$

Entonces, la ecuación 4.2 se puede escribir de la siguiente manera.

$$u(t) = 1.0625 \cdot e(t) + 85$$

4.4.2. Controlador de giro hacia la derecha

La región de ROI que se ocupa va del punto de referencia ($x_{ref} = 240$) al pixel 320. El error máximo se calcula de la siguiente forma.

$$Error_M = 240 - 320$$

$$Error_M = -80 \tag{4.7}$$

El error máximo representa la cantidad de valores maximos que puede tener el error; es decir 80 valores. Teniendo en cuenta esto y el rango de giro hacia la derecha del servomotor (95-0 grados), es posible obtener una gráfica que represente el comportamiento de estas variables (Figura 4.11).

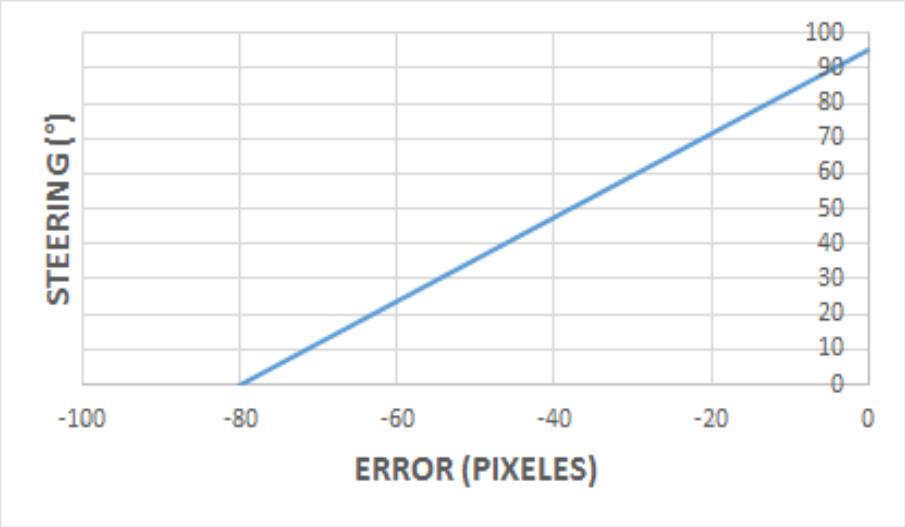

Figura 4.11: Gráfica Error vs Steering

Siguiendo el mismo procedimiento que se utilizó para el cálculo del controlador anterior, es posible obtener los cálculos para el control de giro hacia la derecha del vehículo autónomo.

En la gráfica anterior, al ser una recta, se representa por la ecuación 4.5. Por lo tanto es posible obtener la pendiente mediante la ecuación 4.6.

$$m = \frac{y_2 - y_1}{x_2 - x_1}$$

$$m = \frac{0 - 95}{-80 - 0}$$

$$m = 1.1875 \tag{4.8}$$

La ordenada al origen "b" es 95 que se obtiene directamente de la gráfica.

Sustituyendo los datos en la ecuación 4.5.

$$y = 1.1875 \cdot x + 95$$

Entonces, la ecuación 4.2 se puede escribir de la siguiente manera.

$$u(t) = 1.1875 \cdot e(t) + 95$$

4.5. Controlador Proporcional Derivativo

Para el control de seguimiento de trayectorias del vehículo autónomo, además del control proporcional, se diseñó un controlador proporcional derivativo (PD) con el propósito de observar y comparar el comportamiento del sistema.

Matemáticamente el controlador PD se define de la siguiente manera:

$$u(t) = K_p \cdot e(t) + k_D \cdot e'(t) \tag{4.9}$$

Donde:

$e'(t)$ es la derivada del error

k_D es la ganancia derivativa

k_p es la ganancia proporcional

$e(t)$ es el error del sistema de dirección

$\tilde{e}(t)$ es el error anterior

Matemáticamente $e'(t)$ se define como:

$$e'(t) = \tilde{e}(t) - e_{act} \tag{4.10}$$

Los valores k_D son propuestos de manera heuristica. El valor de k_D propuesto cuando el vehículo gira hacia la derecha es de 0.11875, mientras que k_D cuando el vehículo gira hacia la izquierda es de 0.10625. k_P se mantuvo con el mismo valor que se utilizó en el controlador P.

4.6. Determinación de la posición del vehículo autónomo mediante visión artificial.

Es importante conocer la posición que tiene el vehículo autónomo sobre la pista, para ello se implementó un algoritmo de programación en python, capaz de identificar dicha posición. El procedimiento que se llevó acabo fue el siguiente. Se colocaron tres círculos, cada uno con 17.5 cm de diámetro, sobre una superficie rectangular en color blanco. (Figura 4.12)

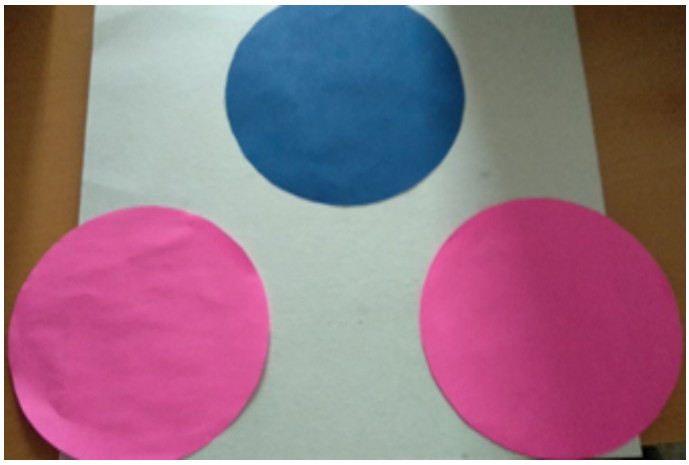

Figura 4.12: Circulos utilizados para la determinación de la posición del vehículo autónomo

La figura 4.12 muestra dos círculos en color rosa y uno en azul. Los cuales se colocaron sobre el vehículo autónomo (Figura 4.13).

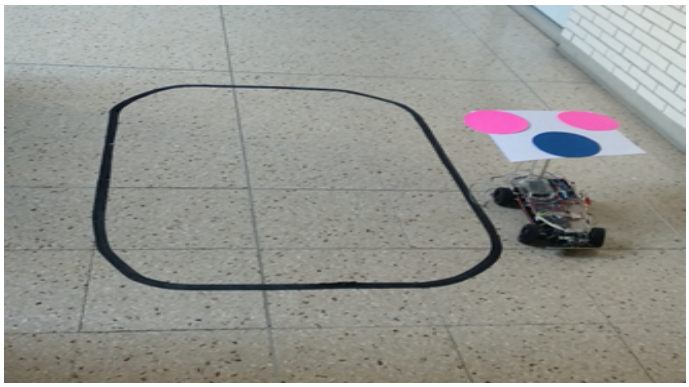

Figura 4.13: Circulos utilizados para la determinación de la posición del vehículo autónomo

El proceso de homografía se realizó con una cámara independiente a la del vehículo. Se utilizó una resolución de 640x480 pixeles, se tomaron 4 puntos fuente de la imagen original, que en la homografía son las 4 esquinas. Además se establecieron 4 puntos destino los cuales están dados por medidas reales en cm (tabla 4.1). El objetivo de obtener una imagen homografiada es hacer una traslación de los puntos fuente a los puntos destino establecidos. La figura 4.14 muestra la imagen homografiada resultante.

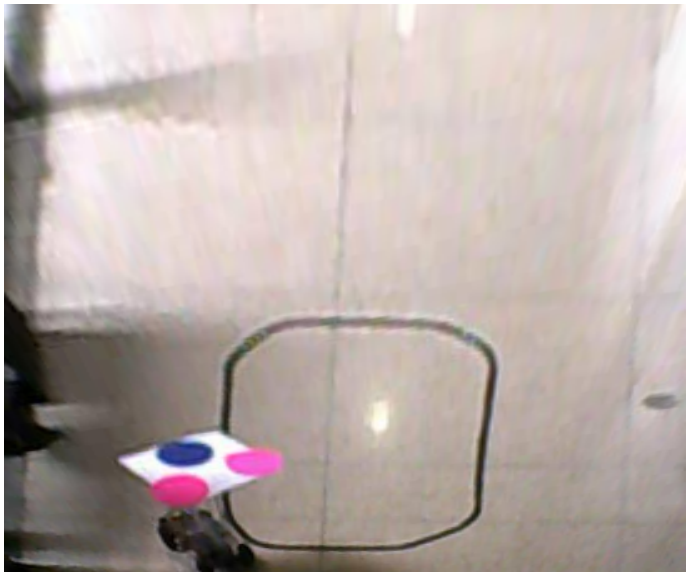

Figura 4.14: Imagen homografiada

Mediante visión artificial se realizaron dos segmentaciones, la primera para los círculos color rosa y la segunda para el circulo azul; como se muestra en la figura 4.15

Puntos fuente	Puntos destinos
(207,106)	(0,0)
(487,134)	(273,0)
(637,400)	(273,404)
(4,329)	(0,404)

Tabla 4.1: Tabla de datos curva steering (0-90 grados)

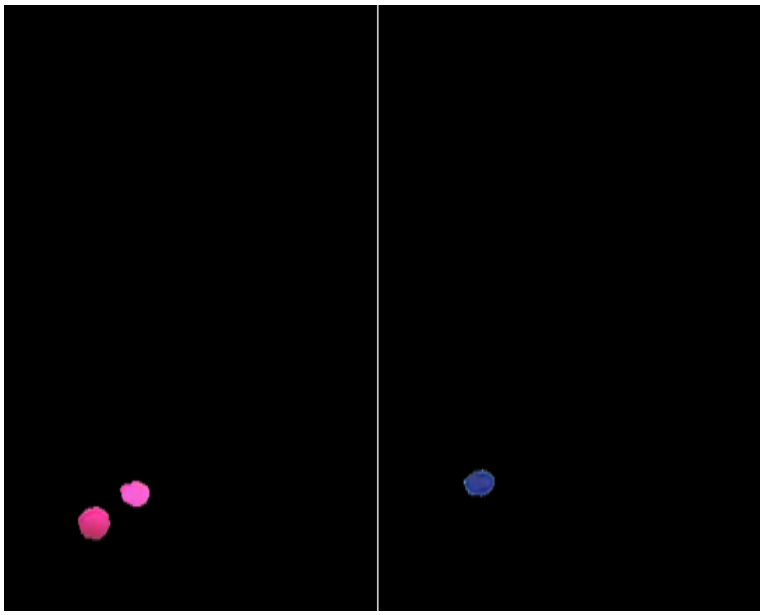

Figura 4.15: Segmentación de los círculos

El siguiente paso consistió en calcular el centroide de todos los círculos. Para el caso de los círculos rosas, se trazó una línea color azul entre sus centroides y se calculó el punto medió de esa línea (punto blanco).

Posteriormente fue posible trazar una línea entre el centroide del circulo azul y el punto blanco, finalmente se obtiene el punto medio de esta nueva línea (Figura 4.16), el cual es el dato más importante ya que da las coordenadas “x,y” sobre las que se encuentra el vehículo autónomo y con esto obtener la gráfica de la trayectoria que realiza.

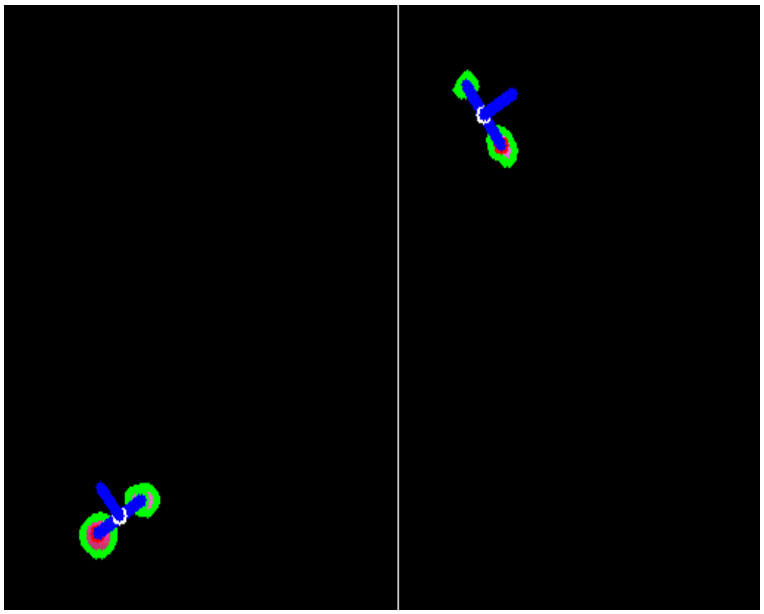

Figura 4.16: Obtención de datos de la posición del vehículo autónomo mediante visión

4.7. Diagrama de flujo de los algoritmos

La figura 4.17 muestra el diagrama de flujo del algoritmo que realiza el seguimiento de trayectorias (aplicación uno) utilizando un controlador proporcional. Mientras la figura 4.18 muestra el diagrama de flujo de la aplicación uno, utilizando un controlador PD.

El diagrama de flujo que representa el algoritmo de programación para la aplicación dos, haciendo uso de un controlador P se muestra en la figura 4.19. Mientras que la figura 4.20 muestra el diagrama de flujo de la segunda aplicación autónoma, haciendo uso de un controlador PD.

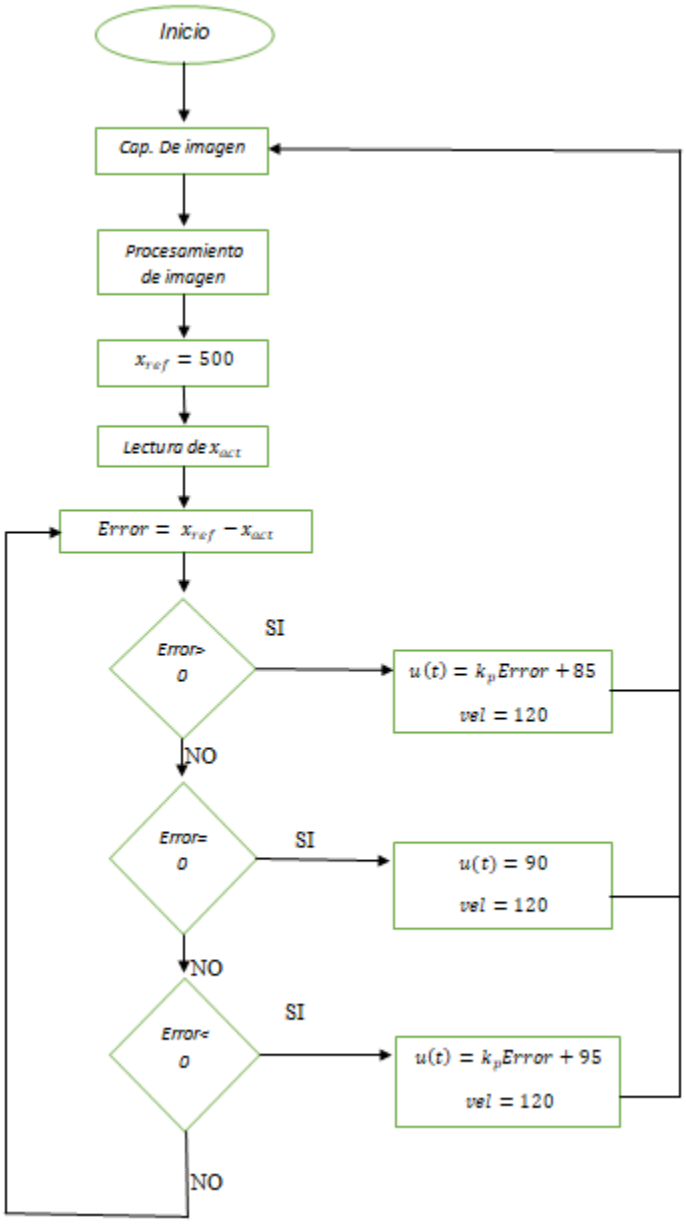

Figura 4.17: Diagrama de flujo para la primer aplicación con controlador P

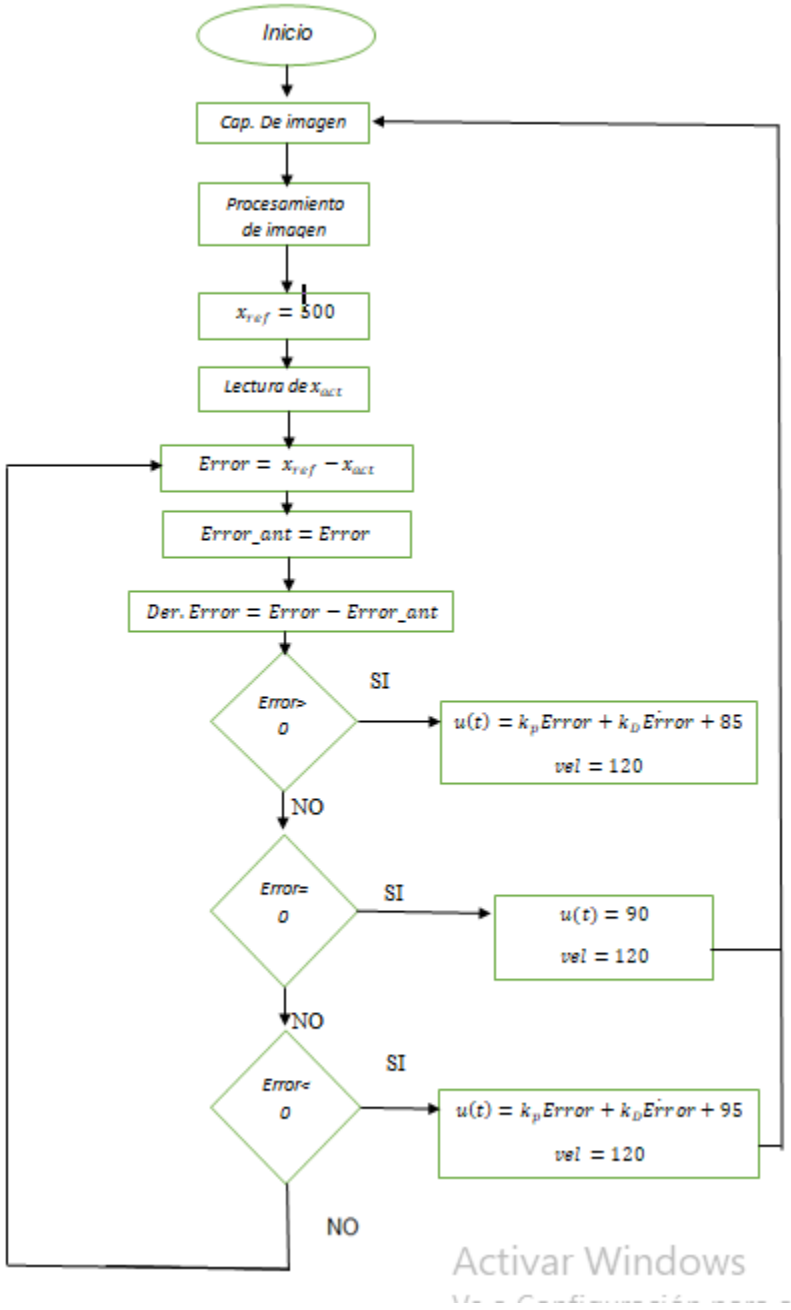

Figura 4.18: Diagrama de flujo para la primer aplicación con controlador PD

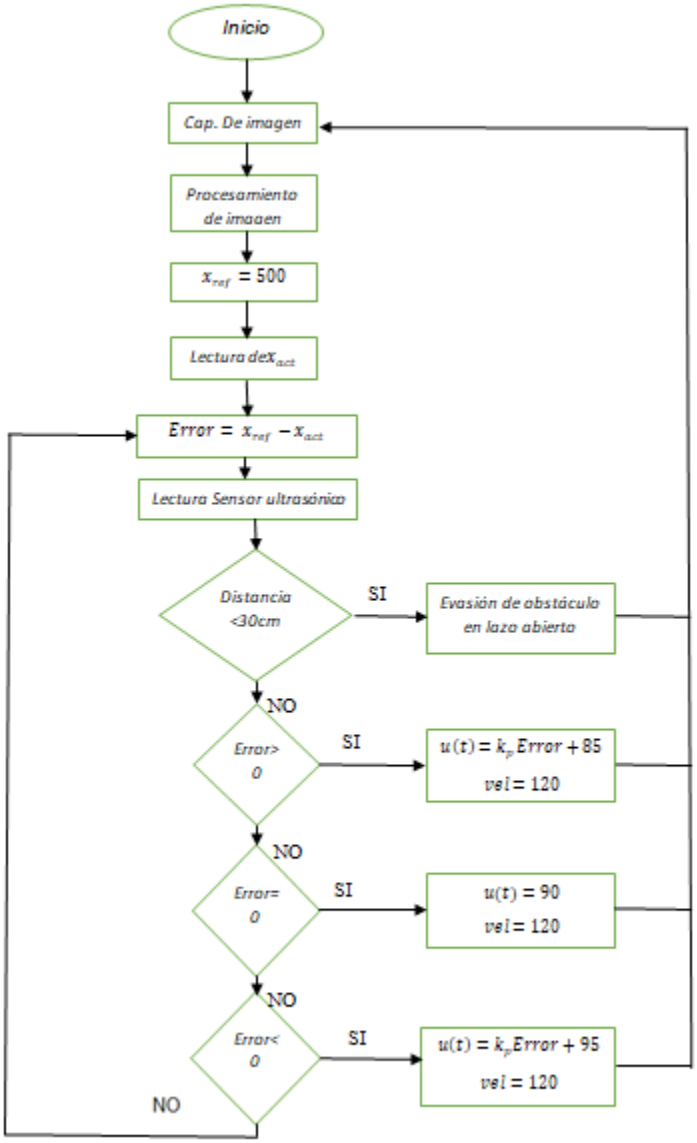

Figura 4.19: Diagrama de flujo para la segunda aplicación con controlador P

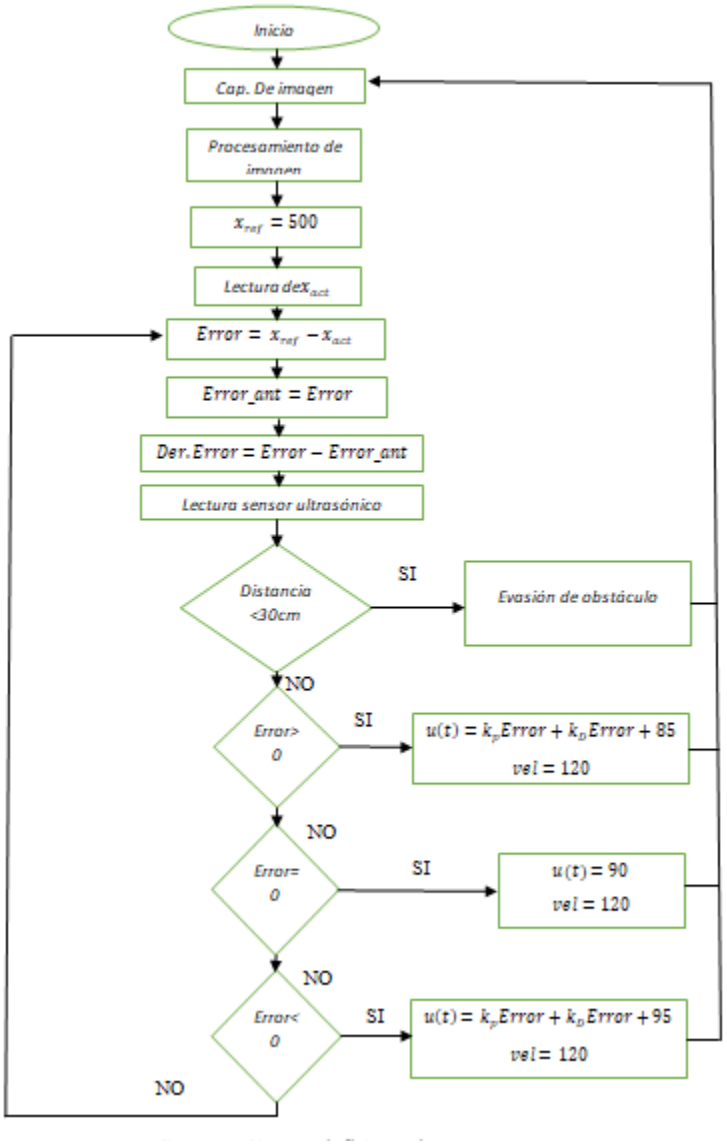

Figura 4.20: Diagrama de flujo para la segunda aplicación con controlador PD

Capítulo 5

Experimentos y resultados

5.1. Curva *steering*

El servomotor se encarga de realizar el giro del vehículo autónomo, en este caso el rango de giro del actuador es de 0-180 grados, donde 0 grados indica el giro total hacia la izquierda mientras que 180 grados es el giro total del vehículo hacia la derecha. Para conocer el radio de giro que el vehículo genera a un cierto valor del servomotor, se realizaron pruebas físicas para obtener la curva de *steering* (radio vs grados).

5.1.1. Curva *steering* hacia adelante (0-90 grados)

Para obtener la gráfica que representa la curva de *steering* se colocó el vehículo de tal forma que avance hacia adelante.Se obtuvieron los siguientes datos como se muestra en la tabla 5.1, mientras que la gráfica se muestra en la figura 5.1.

Grados	Radio(cm)
0	75
30	85
60	141.5
90	inf (se tomó un valor de 1000)

Tabla 5.1: Tabla de datos curva *steering* (0-90 grados)

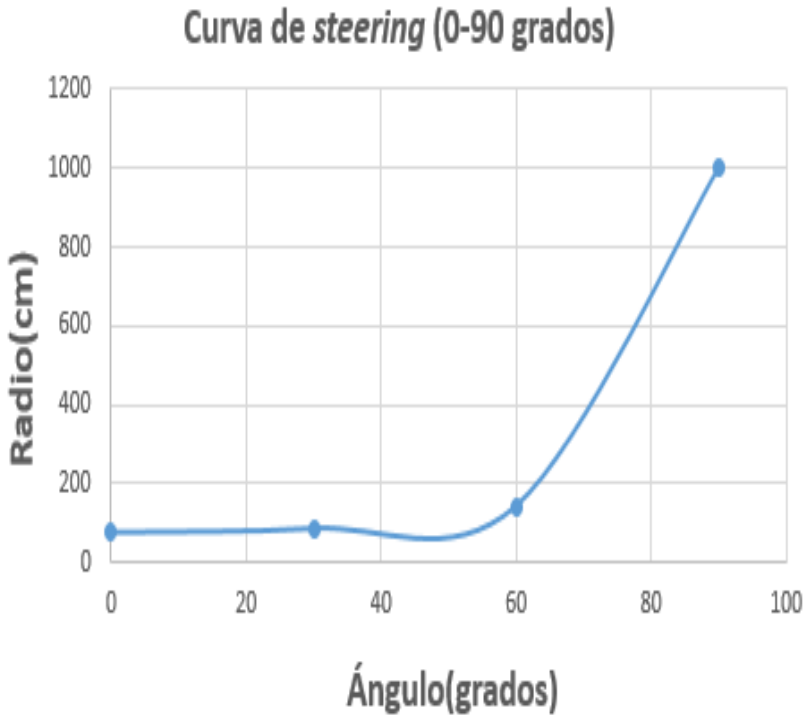

Figura 5.1: Curva de *steering* hacia adelante (0-90 grados)

5.1.2. Curva *steering* hacia adelante (90-180 grados)

Los datos obtenidos cuando el vehículo avanza hacia enfrente y el ángulo de giro es de 90-180 grados se muestran en la siguiente tabla 5.2.

Grados	Radio(cm)
90	inf (se tomó un valor de 1000)
120	107.5
150	85
180	70

Tabla 5.2: Tabla de datos curva *steering* (90-180 grados)

El rango de giro del servomotor para este caso es de 90 a 180 grados, en 90 grados el vehículo autónomo avanza en línea recta indefinidamente, en este caso se tomó un valor de 1000 cm como dato para analizar el comportamiento de la curva de *steering*. Entre mas grande sea el valor que se le asigna al servomotor (180 grados valor maximo), menor es el radio de giro que genera el vehículo. En 180 grados el

radio de giro es de 70 cm siendo este el valor mas pequeño. La velocidad a la que se hicieron las pruebas fue constante.

La gráfica que representa el comportamiento de la curva de *steering* se muestra en la figura 5.2

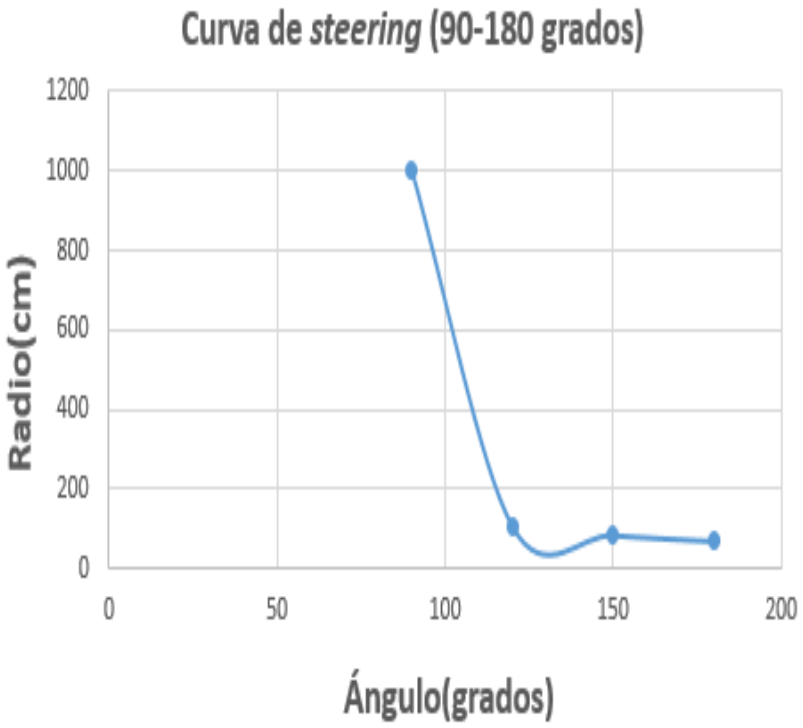

Figura 5.2: Curva de *steering* hacia adelante (90-180 grados)

5.1.3. Curva *steering* hacia atras (0-90 grados)

El radio de giro que se generó cuando el vehículo autónomo va de reversa, en un rango de 0-90 grados se muestra en la tabla 5.3.

La gráfica que representa los resultados obtenidos de la curva de *steering* cuando el vehículo va hacia atrás (reversa) en el rango de 0-90 grados se observa en la figura 5.3

Grados	Radio(cm)
0	71.5
30	73.5
60	116.5
90	inf (se tomó un valor de 1000)

Tabla 5.3: Tabla de datos curva *steering* (0-90 grados)

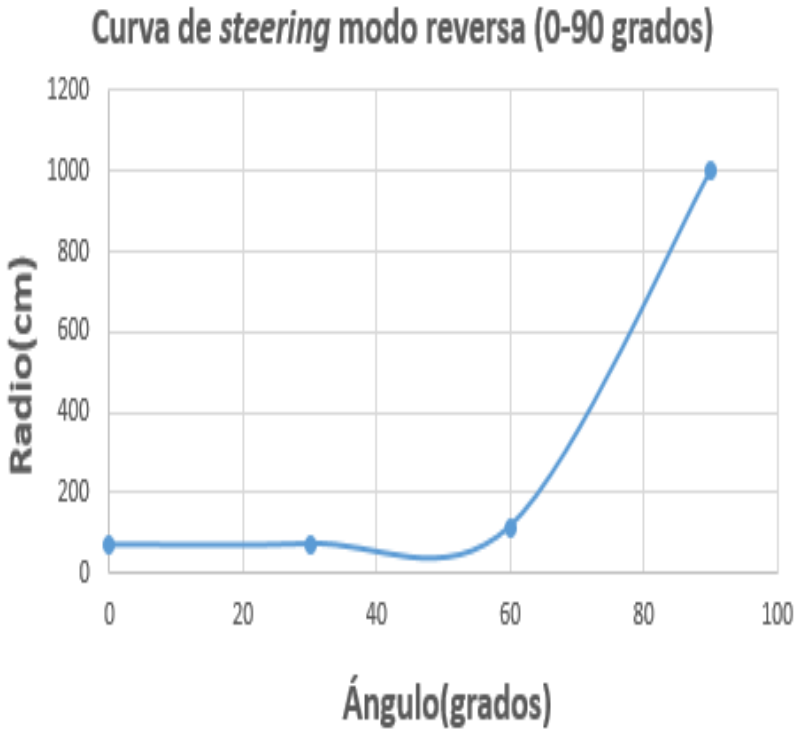

Figura 5.3: Curva *steering* reversa (0-90 grados)

5.1.4. Curva *steering* hacia atras (90-180 grados)

Los resultados que se obtuvieron cuando el vehículo avanza hacia atras (reversa) y el ángulo de giro del vehículo es de 90-180 grados se muestran en la tabla 5.4. Mientras que la gráfica que representa la curva de *steering* en reversa se muestra en la figura 5.4

Grados	Radio(cm)
90	inf (se tomó un valor de 1000)
120	145
150	85
180	64

Tabla 5.4: Tabla de datos curva *steering* de reversa (90-180 grados)

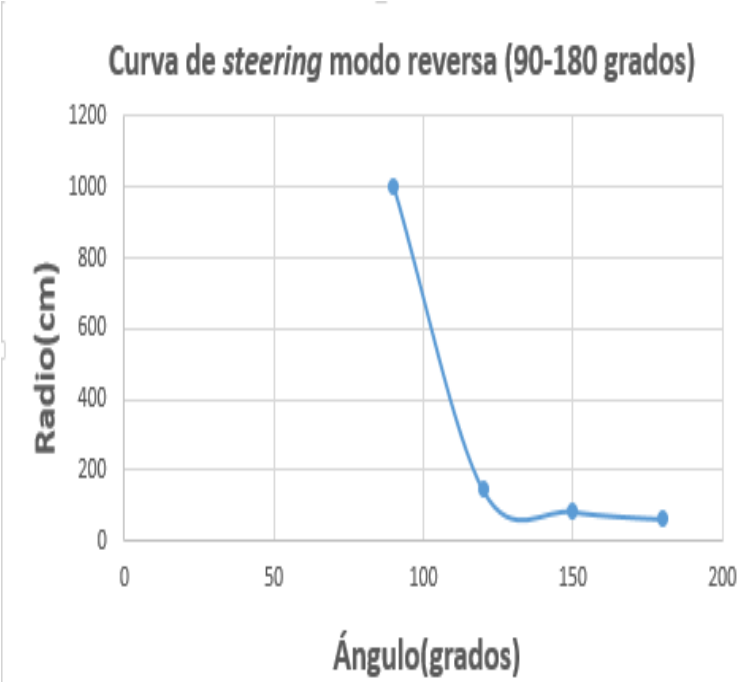

Figura 5.4: Curva *steering* reversa (90-180 grados)

5.2. Gráfica de velocidad vs tiempo

La velocidad constante con la que circula el vehículo autónomo se obtiene mediante la siguiente expresión.

$$v = d/t$$

Donde:

v= velocidad del vehículo autónomo (m/s)

d=distancia que recorre el vehículo autónomo (m)

t=tiempo en el que recorre la distancia (s)

El vehículo se desplazó una distancia de 5m a diferentes valores de PWM, se tomó el tiempo de recorrido y con los datos recopilados se calculó la velocidad. Se capturaron datos tanto para el avance hacia enfrente como de reversa. La tabla 5.5 muestra los datos recopilados y calculados. Mientras que la figura 5.5 muestra la gráfica de la velocidad vs tiempo cuando el vehículo avanza hacia enfrente.

PWM	T. frente(s)	T. reversa(s)	Vel. frente (m/s)	Vel. reversa (m/s)
100	8.88	—-	0.5630	—-
138.5	6.02	—-	0.8305	—-
177.5	3.69	7.79	1.3550	0.6418
216.25	3.50	5.06	1.4285	0.9881
255	2.93	3.93	1.7064	1.272

Tabla 5.5: Tabla con los datos recopilados y calculados

Con un PWM de 100 y 138.5 no es posible que le vehículo logre avanzar en reversa debido al peso de este.

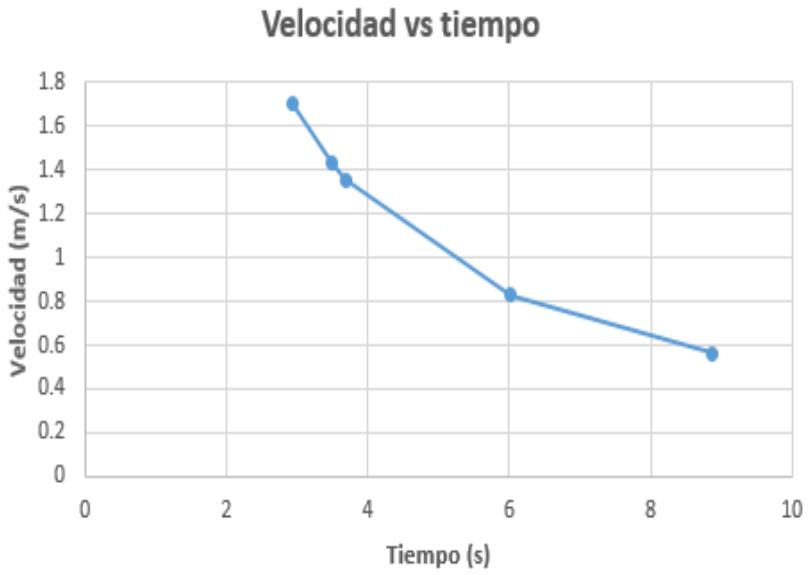

Figura 5.5: Gráfica velocidad vs tiempo cuando el vehículo avanza hacia enfrente

La siguiente gráfica (figura 5.6) muestra el comportamiento de la velocidad vs el tiempo cuando el vehículo autónomo avanza de reversa.

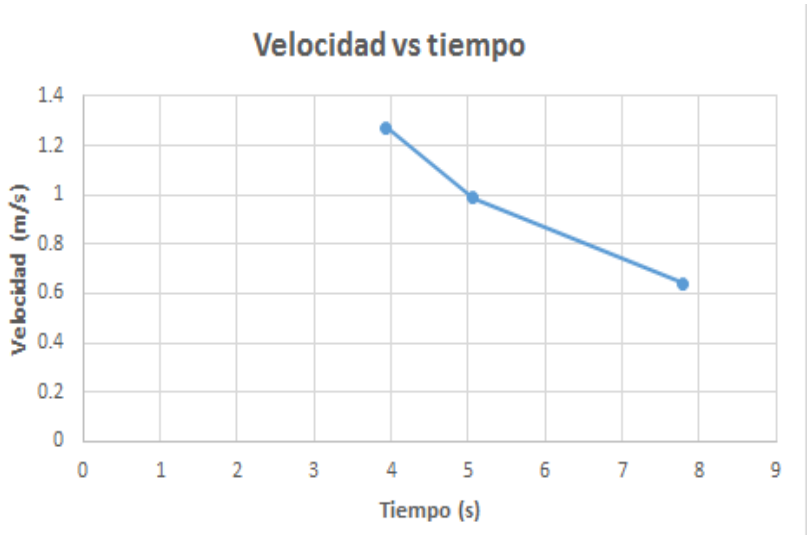

Figura 5.6: Gráfica velocidad vs tiempo cuando el vehículo avanza hacia atras

5.3. Pruebas de funcionamiento del vehículo autónomo

5.3.1. Primera prueba. Seguimiento de trayectorias bien definidas de manera autónoma con un controlador Proporcional

Para esta prueba se estableció una pista en forma de ovalo como se muestra en la figura 5.7, con dimensiones de 1.5m de ancho por 3.1m de largo. En el capítulo 4 se mencionó acerca del control para seguir trayectorias, se estableció una línea de color blanco, para hacer pruebas finales se cambió por una línea de color negro con un ancho de 4cm.

Figura 5.7: Pista en forma de ovalo

El vehículo autónomo tardó 32s en dar una vuelta sobre la pista, a una velocidad de 0.5630m/s. Manteniendo una distancia entre el vehículo y la pista de 13cm, para esta prueba se utilizó un procesador externo al del vehículo (computadora laptop). La figura 5.8 muestra el vehículo autónomo circulando sobre distintos puntos de la pista.

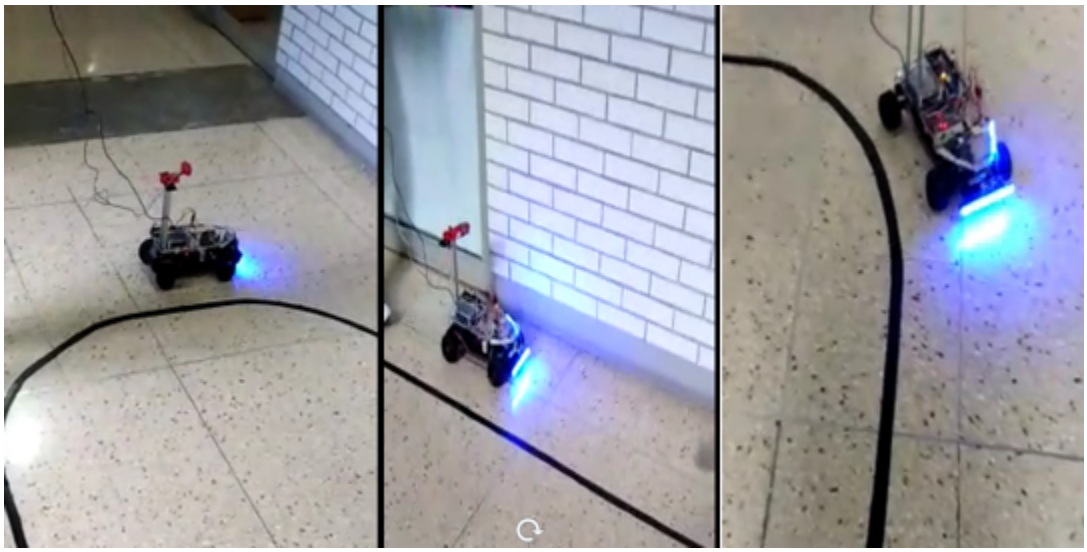

Figura 5.8: Pruebas sobre pista en forma de ovalo

Para el seguimiento de trayectorias del vehículo autónomo se binariza la imagen y sobre ella se realiza el control de dirección, tal como se muestra en la figura 5.9.

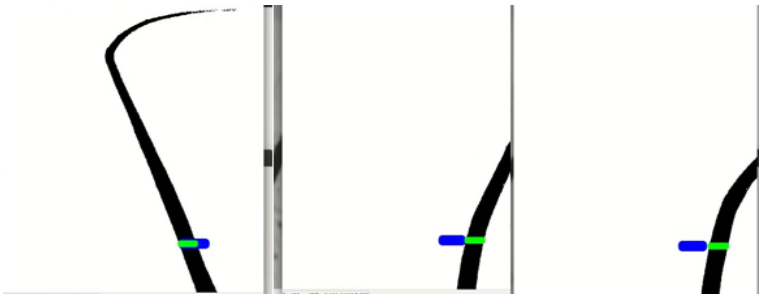

Figura 5.9: Imagen binarizada para el control del vehículo autónomo

Donde la línea azul representa el punto de referencia y la línea verde el punto actual, con estos datos se calcula el error y se realiza el control de giro del vehículo.

Se realizó la misma prueba, ahora con la Raspberry pi 3B como procesador del vehículo autónomo, el cual tardó 32s en recorrer una vuelta completa a una velocidad

de 0.5630 m/s. La distancia entre la línea de la carretera y el vehículo no siempre fue la misma, en algunas ocasiones fue variable debido a la velocidad de procesamiento de la Raspberry. La figura 5.10 muestra el vehículo autónomo circulando sobre la pista.

Figura 5.10: Vehículo autónomo circulando sobre la pista en forma de ovalo

5.3.2. Gráfica de posición del vehículo autónomo para la primera prueba con control Proporcional

Como se mencionó en el capítulo anterior, mediante visión artificial y obteniendo una imagen homografiada del entorno se logró generar un algoritmo de programación para estimar la posición del vehículo autónomo. Para ello la pista se ajustó con un ancho de 107cm y 159cm de largo. La figura 5.11 muestra la gráfica de posición con un controlador P.

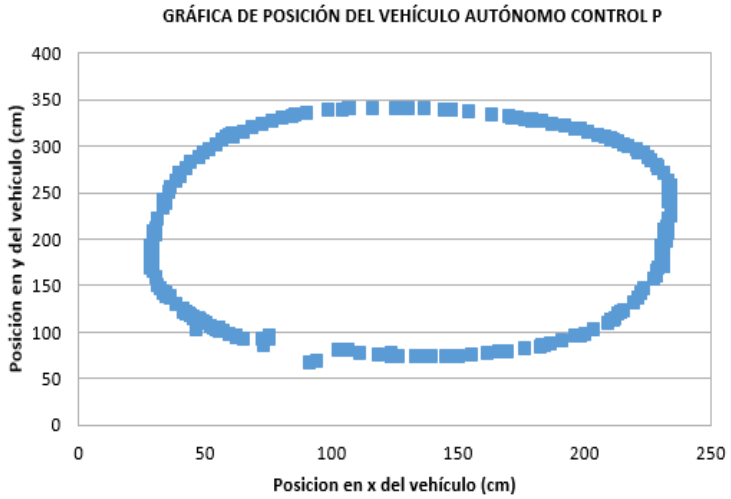

Figura 5.11: Gráfica de posición para la primera prueba con control P

La grafica anterior representa el comportamiento del vehículo autónomo en su recorrido sobre la pista, haciendo uso del procesador externo.

5.3.3. Seguimiento de trayectorias bien definidas de manera autónoma con un controlador Proporcional Derivativo (PD)

De la misma manera que se obtuvieron los resultados para la primera prueba con un controlador P se realizó para un controlador PD, utilizando un procesador externo al del vehículo. La figura 5.12 muestra el vehículo autónomo desplazándose sobre la pista en diferentes puntos de ella. Mientras que en la figura 5.13 se observa la imagen procesada (binarizada) para el control de la trayectoria.

Figura 5.12: Pruebas en pista en forma de ovalo con controlador PD

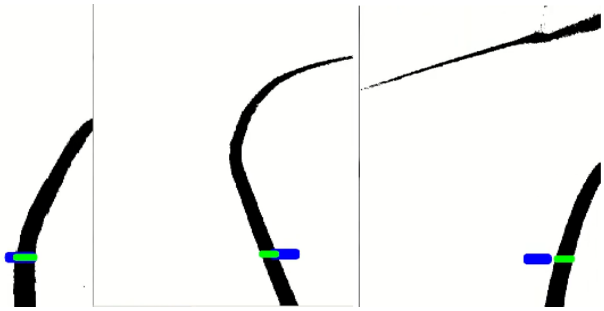

Figura 5.13: Imagen binarizada para el control del vehículo autónomo control PD

De igual forma para el controlador PD se obtuvieron resultados del vehículo autónomo siguiendo la trayectoria bien definida, utilizando como ordenador principal la Raspberry pi 3B.(figura 5.14)

Figura 5.14: Pruebas en pista en forma de ovalo con controlador PD

5.3.4. Gráfica de posición del vehículo autónomo para la primera prueba con control Proporcional Derivativo

La figura 5.15 muestra la gráfica que representa la trayectoria que el vehículo recorrió al desplazarse sobre la misma pista en forma de ovalo que se utilizó para el control P.

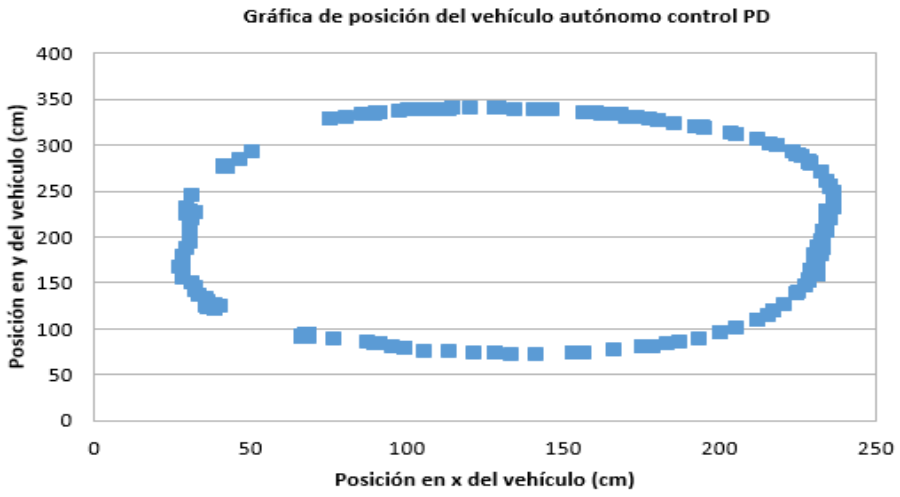

Figura 5.15: Gráfica de posición para la primera prueba con control PD

La gráfica anterior representa la trayectoria que siguió el vehículo autónomo haciendo uso de un ordenador externo.

El vehículo autónomo tiene un mejor comportamiento con la implementación del controlador Proporcional (P), debido a que la respuesta al sistema es más rápida, sin importar que no se alcance el valor de referencia. Es necesario este tipo de controlador, dado que el valor del error se va actualizando rapidamente. Mientras que el controlador PD genera una respuesta más brusca.

5.3.5. Segunda prueba. Rebase de un obstáculo estático en lazo abierto sobre línea recta con un Controlador P.

Para la segunda aplicación se colocó una pista de color negro de 4cm de ancho y 350 cm de largo como se muestra en la figura 5.16.

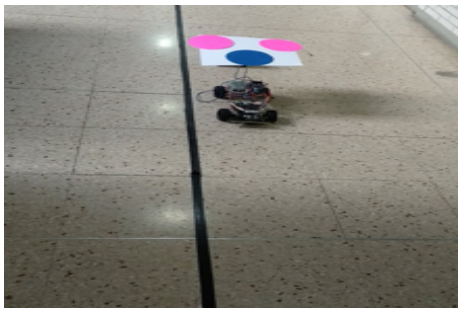

Figura 5.16: Pista para la segunda aplicación con control P

La figura 5.17 muestra al vehículo autónomo haciendo el rebase del obstáculo estático.

Figura 5.17: Vehículo autónomo realizando el rebase de obstaculo

5.3.6. Gráfica de posición del vehículo autónomo para la segunda aplicación con control P

De la misma manera que se obtuvo la gráfica de la posición en la primera aplicación, se generó para la segunda aplicación con el control Proporcional, como se muestra en la figura 5.18.

Figura 5.18: Gráfica de posición para la segunda prueba con control P

La evasión del obstáculo se realizó en lazo abierto, sobre una línea recta. El vehículo primero detectó el obstáculo, las maniobras de rebase se realizaron durante un tiempo establecido. Estas maniobras fueron girar hacia la izquierda, girar a la derecha, quedar de frente para realizar nuevamente la búsqueda de la línea y seguirla.

5.3.7. Segunda prueba. Rebase de un obstáculo estático en lazo abierto sobre línea recta con control PD

La figura 5.19 muestra al vehículo autónomo haciendo el rebase del obstáculo estático, con la implementación del controlador PD.

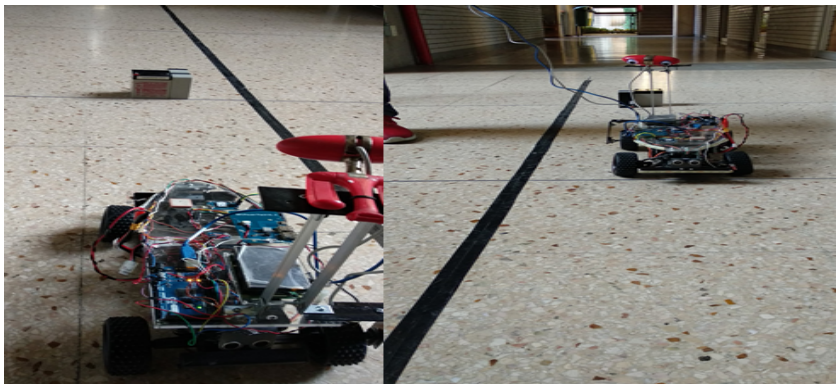

Figura 5.19: Vehículo autónomo realizando el rebaso de un obstaculo con contro PD

5.3.8. Gráfica de posición del vehículo autónomo para la segunda aplicación con control PD.

La figura 5.20 muestra la gráfica para la segunda aplicación con control PD.

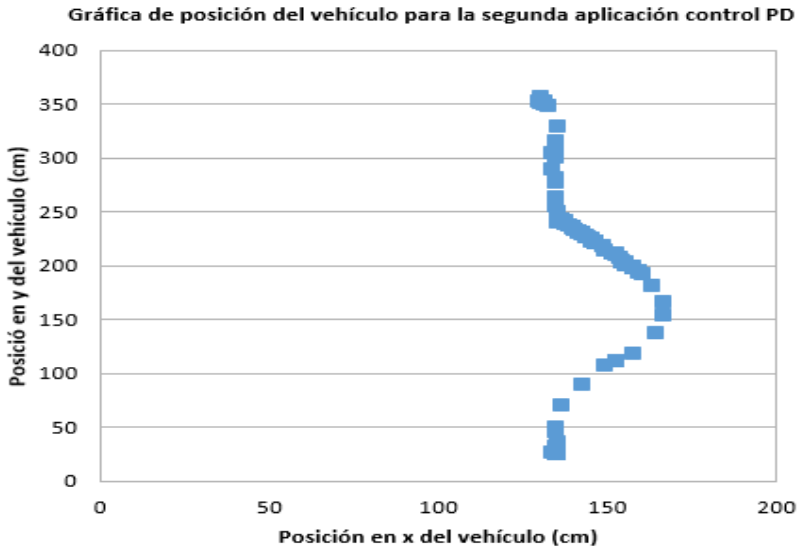

Figura 5.20: Gráfica de posición para la segunda prueba con control PD

Al igual que en la primera aplicación autónoma, el controlador P resulta mejor que el controlador PD.

5.3.9. Captura de datos de sensores del vehículo autónomo

- **Magnetómetro** Los datos recopliados por el sensor IMUGY-85 del vehículo autónomo, visualizados en la terminal mediante ROS, se muestran en las siguientes figuras; en donde el ángulo de orientación varia respecto al norte geográfico dependiendo de la posición del vehículo.

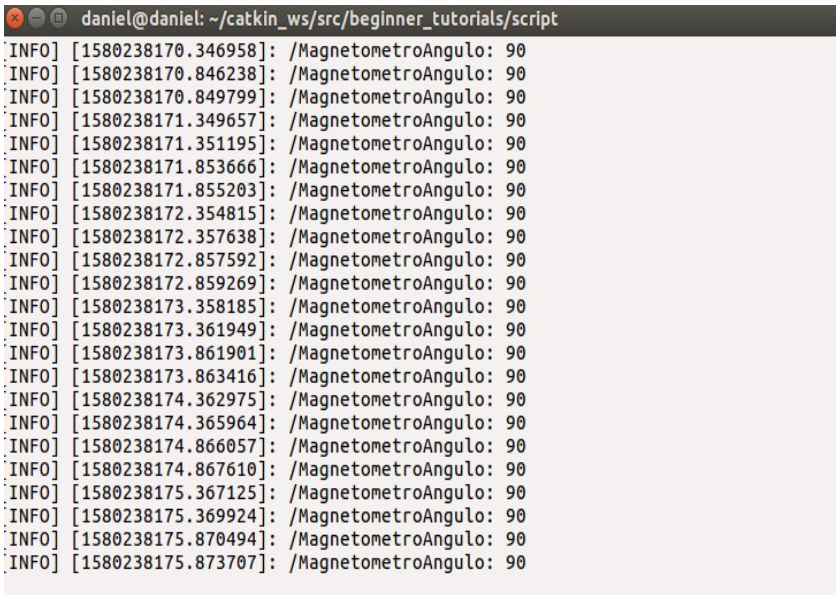

Figura 5.21: Datos del magnetómetro, 90 grados respecto al norte geográfico

- **Encoder KY-040** La velocidad del vehículo se midió a través del sensor KY-040, las unidades son m/s. Los datos son visualizados en la terminal, los cuales varian entre.40 y .54 m/s y se muestran en la siguiente figura.

- **GPS GY-GPS6MV2** Los datos recopilados por el modulo GPS que se implementó en el vehículo autónomo se muestran en la siguiente figura.

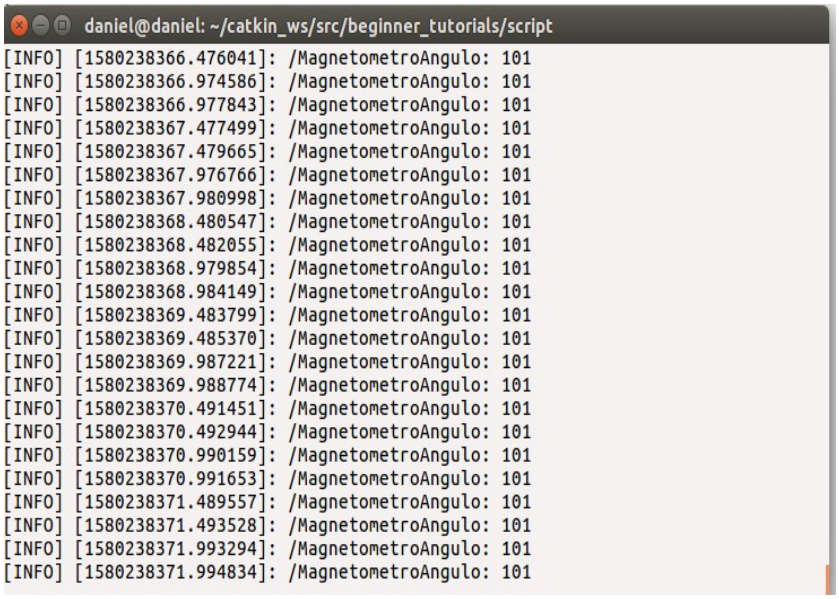

Figura 5.22: Datos del magnetómetro, 101 grados respecto al norte geográfico

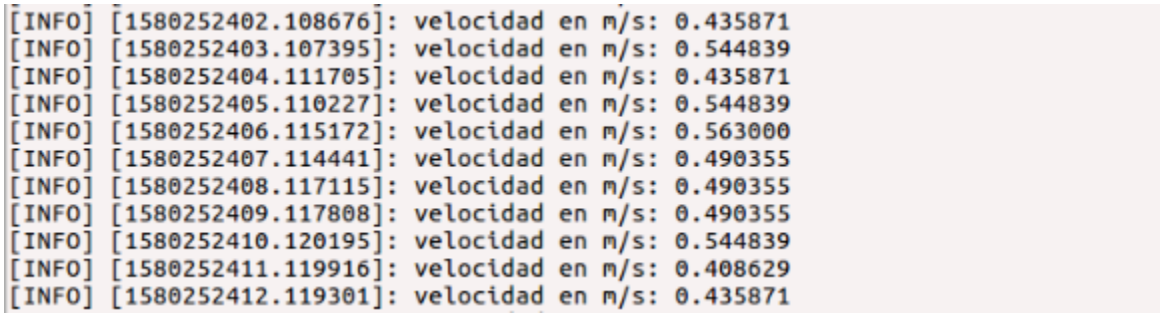

Figura 5.23: Datos de la velocidad del vehículo autónomo en m/s

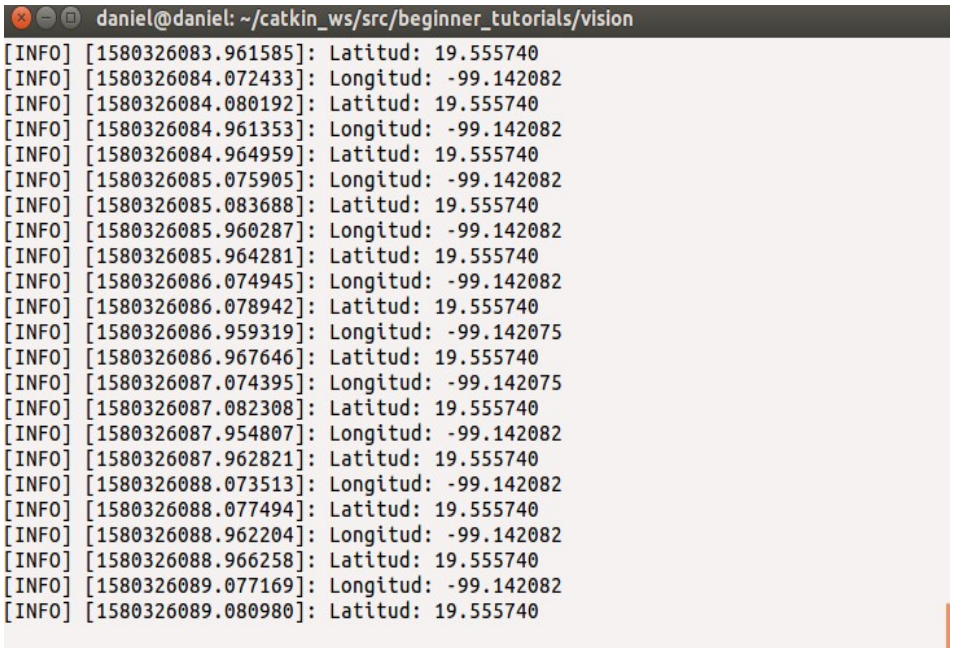

Figura 5.24: Datos del GPS GY-GPS6MV2

Capítulo 6

Conclusiones y trabajo a futuro

En este trabajo se realizó la adaptación de componentes sobre la base de un vehículo de control remoto, control e implementación de un vehículo autónomo. Se han implementado los sensores y actuadores necesarios para la autonomía del vehículo, interactuando entre sí y con el Sistema Operativo de Robótica (ROS).

Se ha comprobado el funcionamiento correcto del sistema de control del vehículo autónomo, el cual incluye sensores, actuadores y el diseño del controlador proporcional (P) y proporcional derivativo (PD).

Mediante las pruebas realizadas de manera física, resultados obtenidos y mostrados (gráficas), se determina que el funcionamiento del vehículo autónomo es el correcto en cada una de las aplicaciones autónomas.

Como trabajos a futuro se propone implementar mejores algoritmos de control, así como la implementación del sensor *Lidar* para tener un escaneo de 360 grados del entorno. Se espera utilizar el GPS para determiar la posición del vehículo autónomo al seguir una trayectoria y llegar al punto de referencia establecido.

Bibliografía

[1] Ramírez, L. (2016).Vehículos Autónomos,1-5. Recuperado de: http://jeuazarru.com/wp-content/uploads/2016/11/Vehiculos-autonomos.pdf

[2] Montoro, L., Martí-Belda, A., Lijarcio, I., Bosó, P., López, C.,Viladrich, R. y Suárez, J. (). Coche autónomo, seguridad vial y formación de conductores, 5-7. Recuperado de: https://www.cnae.com/ficheros/files/noticias/INFORME20 Coche20autoCC81nomo20seguridad20vial20y20formacioCC81n20de20conductores 20INTRAS-CNAE.pdf

[3] (2016). Vehículo Autónomo clasificación/restricciones, 1-3. Recuperado de: http://www.centro-zaragoza.com:8080/web/sala-prensa/revista-tecnica/hemeroteca/articulos/R70-A7.pdf

[4] Tamara, R. y Ros, M. (2009-2010). Sistema de Posicionamiento Global (GPS), 1.Recuperado de: https://webs.um.es/bussons/GPSresumen-TamaraElena.pdf

[5] Magdaleno, F. y Martínez, R. (2006).Aplicaciones de la teledetección láser (LIDAR) en la caracterización y gestión del medio fluvial, 1. Recuperado de: http://ambiental.cedex.es/docs/Ingenieria-Civil-142-2006-Aplicaciones-LiDAR.pdf

[6] Cámara infrarroja. (2016, agosto 7). EcuRed, . Consultado el 21:18, febrero 11, 2020 en https://www.ecured.cu/index.php?title=CC3A1mara-infrarrojaoldid=2684552.

[7]] Zannatha, J., Vera, P., Orozco, S., Cureño, A., Picasso, R. y Ramírez, L.(). Generación de trayectorias para vehículos autónomos, 1. Recuperado de: http://comrob.org/2018/openconf/modules/request.php?module=oc-proceedingsaction=view.phpid=1840file=1/1840.pdfa=Accept

[8] Celada Sanz, F., Análisis del sistema de dirección de un automóvil mediante MULTIBODY de SIMULINK, [Legánes]: Escuela Politécnica Superior de la Universidad Carlos III de Madrid.

[9] Cristóbal Villanueva A., Rangel Cervantes AJ., Valente Méndez JF., "ESTUDIO NUMÉRICO DEL SISTEMA DE DIRECCIÓN PARA UN VEHÍCULO ARENERO", [México, D.F.]: Escuela Superior de Ingeniería Mecánica y Eléctrica Unidad Ticomán, 2009.

[10] Rodríguez Huitrón S., MODELO EXPERIMENTAL DE VEHÍCULO CON DOS MODOS DE DIRECCIONABILIDAD, [Ciudad de México]: Universidad Nacional Autónoma de México, 2017.

[11] Platero, C., Apuntes de Visión Artificial, 2005-08. Recuperado de: http://www.ieef.upm.es/webantigua/spain/Asignaturas/Robotica/ApuntesVA/cap1IntroVA.pdf

[12] http://www.etitudela.com/celula/downloads/visionartificial.pdf

[13] Sucar, LE, Gómez G., Visión computacional.

[14] Gómez, Rodríguez F., Domínguez Morales MJ., Fundamentos de la visión.

[15] http://wiki.ros.org

[16] García Cazorla, A., ROS: Robotic Operating System. [Cartagena]: Universidad Politécnica de Cartagena, 2013.

[17] Gutiérrez Pérez, J., Introducción a ROS en Raspberry pi, 2017.

[18] Programación de interfaces. Sensores. [Archivo PDF]. Recuperado de: http://www.laurence.com.ar/artes/programacion/archivos/sensores.pdf

[19] Arduino. Sensor ultrasónico de distancia. [Archivo PDF]. Recuperado de: https://blogsaverroes.juntadeandalucia.es/iesbellavista/files/2016/02/SENSOR-ULTRASC393NICO.pdf

[20] Diciembre 2003. SERVOMOTORES. [Archivo PDF]. Recuperado de: http://www2.elo.utfsm.cl/ minedu-cagv/docs/ListaDetalladadeModulos/servos.pdf

[21] MB0014: OEM ARDUINO MEGA 2560. [Archivo PDF]. Recuperado de: http://www.agspecinfo.com/pdfs/M/MB0014.PDF.

[22] DUAL FULL-BRIDGE DRIVER. [Archivo PDF]. Recuperado de: https://www.sparkfun.com/datasheets/Robotics/L298-H-Bridge.pdf

[23] Unidad didáctica. Motores de corriente continua. [Archivo PDF]. Recuperado de: http://myelectronic.mipropia.com/CICLO2/M0804MOTORESDECORRIENTE.PDF?i=1

[24]] Raspberry Pi. (2019, 10 de octubre). Wikipedia, La enciclopedia libre. Fecha de consulta: 04:05, octubre 17, 2019 desde https://es.wikipedia.org/w/index.php?title=RaspberryPioldid=120134031.

[25] Keyes KY-040 Arduino Rotary Encoder User Manual. Recuperado de: http://henrysbench.capnfatz.com/henrys-bench/arduino-sensors-and-input/keyes-ky-040-arduino-rotary-encoder-user-manual/

[26] IMU GY-85 9DOF ITG3205 ADXL345 HMC5883L. Recuperado de: https://hetpro-store.com/imu-gy-85/

[27] Vázquez, C., ¡OTRO MEXICANO EN LA CIMA! GANA RAÚL ROJAS EL "DESAFÍO MUNDIAL DE MANEJO AUTÓNOMO", Octubre 20, 2019. Recuperado de: https://www.sopitas.com/noticias/gana-raul-rojas-en-premio-de-manejo-autonomo/

[28] Febrero 2016. 7 características del coche autónomo de Tesla. Recuperado de https://blog.nubecolectiva.com/7-caracteristicas-del-coche-autonomo-de-tesla/

[29] Rodriguez, P. (2017). Cuando los automóviles se conducen solos, 6. Recuperado de: https://publiadmin.fundaciontelefonica.com/.../add-descargas?...pdf...pdf

[30] ¿Qué es actuador?. Recuperado de: https://diccionarioactual.com/actuador/

[31] Bastidas, N. Fundamentos de programación. Técnicas de algoritmos para el diseño de procesos computacionales. Recuperado de: https://slideplayer.es/slide/3448545/

[32] Julián Pérez Porto y Ana Gardey. Publicado: 2016. Actualizado: 2017. Definicion.de: Definición de autonomía (https://definicion.de/autonomia/)

[33] Definición de calibración. Recuperado de: https://www.alpemetrologia.com/consultas-frecuentes/

[34] Bloom. Recuperado de: https://bloom.readthedocs.io/en/0.5.10/

[35] Centroíde: Recuperado de: https://html.rincondelvago.com/centroide.html

[36] Julián Pérez Porto y Ana Gardey. Publicado: 2016. Actualizado: 2018. Definicion.de: Definición de concéntrico (https://definicion.de/concentrico/)

[37] Glosario de términos de control. Recuperado de: http://materias.fi.uba.ar/6722/SISTCONTRPRIM2003.pdf

[38] Julián Pérez Porto y María Merino. Publicado: 2017. Actualizado: 2019. Definicion.de: Definición de diagrama (https://definicion.de/diagrama/)

[39] Instrumentación II. Recuperado de: https://es.slideshare.net/josemanuelvaldez5/instrumentacion-ii

[40] Homografía (geometría). Recuperado de: https://esacademic.com/dic.nsf/eswiki/1311779

[41] Ïnterfaz". En: Significados.com. Disponible en: https://www.significados.com/interfaz/ Consultado: 16 de marzo de 2020, 09:03 pm.

[42] Julián Pérez Porto y Ana Gardey. Publicado: 2018. Actualizado: 2019. Definicion.de: Definición de irreversible (https://definicion.de/irreversible/)

[43] Sistema de control en lazo abierto. Recuperado de: https://www.coursehero.com/file/p2r3lmcd/Un-sistema-de-control-de-lazo-abierto-se-caracteriza-por-que-no-recibe-ninguna/

[44] Microcontrolador – qué es y para que sirve. Recuperado de: https://hetpro-store.com/TUTORIALES/microcontrolador/

[45] Pixel. Recuperado de: http://www.pulso.uniovi.es/wiki/index.php/Pixel

[46] ¿Qué es una señal de modulación de ancho de pulso (PWM) y para qué se utiliza?. Recuperado de: https://knowledge.ni.com/KnowledgeArticleDetails?id=kA00Z0000019OkFSAUl=es-MX

[47] Tamaño y resolución de imágenes. Recuperado de: https://helpx.adobe.com/mx/photoshop/using/image-size-resolution.html

[48] Reflectividad. Recuperado de: http://www.educaplus.org/elementos-quimicos/propiedades/reflectividad.html

[49] Rosbuild. Recuperado de: http://wiki.ros.org/rosbuild

[50] CONCEPTO BASICO DE SEÑALES ANALOGICAS Y DIGITALES. Recuperado de: https://sites.google.com/site/electronicaanalogicagerman/unidad-iii-convertidores/concepto-basico-de-senales-analogicas-y-digitales