Desarrollo proyecto integrador Nuevas tecnologías

Coches autónomos con visión por computadora y procesamiento de imágenes

Gilbert Olmos Barradas
Amos Vielma Rivas
Hristo Jiménez Alcantar
Yair Rodríguez Chimal

Prefacio

El presente documento tiene como objetivo recaudar la mayor cantidad de información con respecto al desarrollo del proyecto de coches autónomos con visión artificial y procesamiento de imágenes, desde su planteamiento, el proceso de prototipado y construcción, así como las conclusiones a las que se llegó una vez finalizado e implementado el proyecto, reflexionando sobre lo aplicado y el uso de conceptos, así como tecnologías ya utilizadas en diferentes momentos de la carrera.

Preface

The purpose of this document is to gather as much information as possible regarding the development of the autonomous car project with artificial vision and image processing, from its initial conception, the prototyping and construction process, as well as the conclusions reached once the project was completed and implemented, reflecting on the application and use of concepts, as well as technologies already used at different times during the course.

Introducción

Problemática a resolver

La elección de este proyecto parte con el objetivo de resolver una problemática personal presentada a lo largo del semestre pasado y el presente. Durante este periodo de tiempo y como parte de una de nuestras materias se nos otorgó la posibilidad de trabajar con un kit de automóviles autónomos desarrollado y distribuido por Waveshare, los cuales fueron adquiridos por la universidad pero que hasta el momento no se habían logrado poner en funcionamiento, por lo que la meta era conseguir que los kits se pudieran emplear y destinar a presentarse como un proyecto atractivo realizado por parte de la facultad de ingenierías.

El equipo con el que se comenzó a trabajar fue el JetRacer Pro Powered by Jetson Nano, que se conjuntaba con el kit Waveshare Jetson Nano Dev Kit.

Sin embargo, así como el proyecto trajo un cúmulo de aprendizajes respecto al machine learning, el empleo de sistemas operativos tipo Linux, procesamiento de imágenes y desarrollo de programas de visión por computadora, también trajo consigo varias problemáticas, partiendo desde complicaciones para conseguir que la tarjeta Jetson pudiese correr el sistema operativo que contenía las librerías y

archivos necesarios, así como poder pasar de la batch y entrar a la interfaz de la misma tarjeta, estos obstáculos fueron sorteados de forma exitosa, pero siempre se mantuvo uno, la incapacidad de utilizar los puertos MicroSD, los cuales eran cruciales, ya que mediante estos se utilizaría la memoria que contenía el sistema operativo que necesitaban tanto los motores como la cámara, por lo que la única solución que se encontró fue emplear un adaptador USB y utilizar uno de los puertos disponibles.

A lo largo de los meses se continuó trabajando en el proyecto, consiguiendo grandes avances como conseguir hacer uso de la cámara, controles y motores desde la plataforma Jupyter, pero con la limitación de ser incapaces de entrenar el dispositivo para su implementación autónoma, ya que al todo estar trabajando mediante los puertos USB, el sistema mismo era inestable y no soportaba la carga de memoría que el entrenamiento requeriría.

Este proceso de intentos, fallos y desconfiguración se mantuvo durante un tiempo prolongado, hasta que sin ver avances se decidió optar por un camino diferente, adaptar los elementos que nos eran útiles, como lo era la infraestructura misma de los autos, los motores, servos y estructura del automóvil, y cambiar aquellos con los que se estaba teniendo problemas, la tarjeta Jetson y la cámara.

Fue que por esta razón que como solución a la problemática se decidió trasladar la parte lógica o el cerebro de los automóviles a una raspberry, hacer uso de una cámara web, y sacar provecho de los elementos funcionales, así como de los conocimientos que se habían adquirido pero un entorno estable y con el cual se podía comenzar desde cero.

Justificación técnica

La selección del proyecto no solo parte con la intencionalidad de solucionar la problemática ya mencionada, si no por la complejidad de este y los temas que involucra, mismo que han estado presentes a lo largo de distintas materias durante el transcurso de la carrera.

En primera instancia es el uso de programas de visión por computadora y procesamiento de imágenes, ambas necesarias dada la naturaleza del planteamiento del proyecto, la autonomía de los automóviles le corresponde al desarrollo de estas dos materias.

Ambas áreas van de la mano y correlacionadas la una con la otra, el procesamiento de imágenes se encarga de la manipulación y el mejoramiento de estos archivos con el fin de analizarlos o ser utilizados, se puede entender como la preparación antes de someter la imagen a un programa de visión por computadora.

La visión por computadora es la encargada de extraer contenido significativo de los datos que se están recibiendo y que estos se puedan interpretar y utilizar, ya sea mediante el reconocimiento de patrones u objetos, identificación de colores, etc., la interpretación de esta información tiene como objetivo ser utilizada en la toma de decisiones y automatización de procesos. En proyectos bien fundamentados de vehículos autónomos la visión por computadora es esencial para la identificación de obstáculos, otros coches, así como la señalización que hay en las autopistas y carreteras, y aunque en el proyecto no las hay como tal, es crucial para conseguir identificar el camino por donde se debe ir y la existencia de algún obstáculo. Ambas materias requieren conocimientos de programación, algunos de los cuales ya se habían adquirido en materias como visión industrial y transformadas integrales, el desarrollo de ambas partes se llevó a cabo en Python, el cual es un lenguaje de programación que dada su simplicidad así como un amplio acceso a bibliotecas de la misma área, como lo son OpenCV, una de las más comunes y la utilizada en este, a su vez de poder interactuar con NumPy, herramienta útil en el procesamiento de imágenes debido a su capacidad para mostrar las imágenes como matrices y poder realizar operaciones matemáticas de una forma sencilla, facilitando tareas que en otra instancia serían complejas, por estas razones así como otras Python resulta un entorno afable y amigable para el desarrollo de programas de visión industrial, siendo un ecosistema con una gran variedad de recursos y herramientas que reciben mantenimiento constante de parte de la comunidad y de la cual se puede contar con su apoyo.

El uso de estos elementos se tuvo que complementar con otros, por ejemplo la creación de un servidor en línea para permitir al usuario monitorear la imagen ya procesada y la capacidad de cambiar los parámetros del filtrado de la imagen si es que las condiciones de trabajo no permiten que su funcionamiento sea el adecuado; siendo la conjunción de conocimientos los que vuelven atractivo el proyecto para nuestro perfil profesional, primero como ingenieros, pues el trabajo en áreas de las cuales se tiene poco conocimiento siempre son un reto y una buena oportunidad para poner a prueba nuestra principal habilidad como profesionales, buscar soluciones a problemas; después como ingenieros mecatrónicos, ya que tanto la parte de visión por computadora se traslada al campo laboral en lo que respecta a la visión industrial, donde su uso es fundamental para la automatización de procesos, mientras que la creación de servidores como el implementado, permiten concebir proyectos que resulten más amigables para

los usuarios a los que en un futuro se les pretende vender este tipo de servicios.

Añadiendo a lo que sería el área respectiva a la programación, el proyecto requiere hacer uso de temas como los protocolos de comunicación necesarios para el funcionamiento de los motores, en este caso el protocolo i2c, ya utilizado en el desarrollo de proyectos anteriores, pero con propósitos y entornos distintos.

Finalmente, se implementó un control PI, necesario para la regulación de la trayectoria del vehículo, involucrando una de las áreas más importantes de la ingeniería mecatrónica, especialmente en los procesos de automatización y necesario para la autonomía de proyectos como el presente.

Además del uso de conocimientos y herramientas de programación, se utilizaron programas de diseño 3D para la creación de piezas necesarias para adaptar la estructura ya existente a los elementos nuevos, como lo es la Raspberry y la webcam, implementando lo aprendido en materias como Diseño por computadora y Manufactura asistida por computadora, tanto en el boceto y diseño de estas piezas, así como su manufactura mediante el uso de impresoras 3D.

Desarrollo

Tecnologías a utilizar

Las tecnologías que se utilizaron en el proyecto son las siguientes:

- Raspberry Pi Model 4
- Webcam USB
- Estructura física del kit Jetracer de NVIDIA
    - (Servomotores, base donde estaba la tarjeta Jetson y motores para el movimiento)
- Programación: Código para el procesamiento en visión por computadora, uso de los motores mediante protocolo de comunicación i2c y monitoreo de la imagen ya procesada.
    - Python
    - Numpy
    - Opencv

Tiempos de trabajo y desarrollo del proyecto

Aunque anteriormente se planteó la problemática que encauza a la implementación de la solución planteada, una gran parte de los tiempos de trabajo se concentraron especialmente en la constante investigación de documentación necesaria para conseguir que los elementos que nos estaban fallando pudiesen comenzar a trabajar.

A la par, una vez que nos percatamos el entrenamiento no podría realizarse, se intentó conseguir que se programase en Python todo lo correspondiente a la visión

por computadora y el automóvil funcionase mediante su uso pero trabajando desde la Jetson, pero esta alternativa tampoco se pudo llevar a cabo, en primer instancia las cámaras ya no eran reconocidas por la Jetson, ni en los códigos de Python, ni desde la batch, por lo que se trató de usar una webcam, que igualmente no era reconocida incluso ya instalados los drivers necesarios para que la Jetson pudiese utilizarla; por otra parte, los motores y sus librerías no podían utilizarse fuera del entorno en Jupyter, ya que al querer trabajar con códigos fuera de este, la Jetson no reconocía ninguno de los archivos necesarios para su funcionamiento, fue a partir de este punto en que se decidió adquirir una Raspberry y comenzar a trabajar.

El inicio del trabajo con la Raspberry fue directo a la elaboración de un código de visión industrial dado que no se tenían las mismas dificultades que con la Jetson y desde el primer momento se pudo trabajar con la webcam; el objetivo del código que se comenzó a desarrollar fue que procesase la imagen que se estaba recibiendo en vivo, hiciese una mascara para el color amarrillo (ya que de este color son los carriles que delimitan la pista por la cual tiene que moverse el automóvil), dibujase los contornos de las dos áreas más grandes detectadas y mediante el uso de estas áreas calculase el punto medio entre ellas.

Código: mediante este vinculo se proporciona acceso al repositorio que contiene el código de visión industrial como referencia para quienes en un futuro trabajen con el kit de automóviles y quisiesen tomar como base el trabajo realizado, a la par, se agrega como un material que nutre el entendimiento de lo que se esta por explicar sobre el funcionamiento del código y en sí mismo, del proyecto.

La sección de visión industrial del código proporcionado se divide en cuatro segmentos importantes, la conversión HSV, definición del rango de color, detección y filtrado de contornos, y cálculo de momentos y centroides.

Comenzando por la conversión de BGR a HSV es como tal la transformación de una imagen o de un video de lo que sería el formato estándar BGR (que se mantiene por sus siglas en ingles Blue, Green and Red), al espacio de color HSV, donde se toma en cuenta tonalidad, saturación y valor, especialmente útil para aislar colores específicos.

Como tal la transformación de formato a espacio es útil por lo mencionado al final, aislar colores y el procesamiento de imágenes para este tipo de procesos es mucho más sencillo en el espacio HSV pues reduce la complejidad de la conversión, independientemente de las condiciones de iluminación, las cuales suelen ser un

obstáculo en este tipo de cuestiones, este modelo facilita el trabajo debido que separa como la información respectiva al color y al brillo; dadas sus siglas HSV, la H en español se mantiene por tonalidad, que refiere a la representación del color puro o información pura de este mismo (BGR) y son estos valores insensibles a los cambios de la intensidad de la luz, la cual de mantiene para lo que sería la letra V (value), aunque como tal los datos puros o información pura que contiene un color no se ve afectada por la iluminación, al momento de procesar la imagen esta se puede ver afectada por la misma, por lo que cambiar el cociente del Value permite que se responda mejor a estos cambios, sin tener que cambiar lo que serían las constantes o valores de los colores.

Esta sencillez permite que la segmentación de un color y su aislamiento sean rápidos, haciendo que procesos que podrían ser complejos si no se mantienen condiciones idénticas sean mucho más sencillos, siendo una de las principales razones por las que se utiliza este proceso en la industria y otros entornos, como control de calidad o medicina.

La definición del rango de color es la encargada de obtener los valores de HSV, tanto mínimos y máximos, en el código esta parte esta aplicada para que el usuario sea capaz de modificar estos valores en tiempo real mediante el uso de trackbars, pero como tal estos valores pueden simplemente definirse dentro del mismo archivo y cambiarse antes del uso del programa; una vez definidos los valores se crear un arreglo NumPy que define cuales serán los limites inferior y superior, en este caso ambos se definen como amarillo en HSV para ambas partes, pues ese este color el que nos interesa aislar, ya definidos se aplica este rango de color a la imagen HSV, creando una imagen binaria (máscara) que aunque no se refleje como tal en la imagen que se ve en vivo, transforma en blanco todos los pixles que se consideren amarillos dentro de los rangos establecidos y el resto son negros, generando una imagen en blanco y negro que solo detecta el color amarillo.

La mascara binaria es crucial, pues define las regiones de interés o partes de la imagen procesada que son útiles para el fin o medio que se esté aplicando, como tal la mascara es una matriz de datos, datos ajustados según los valores definidos mediante el método HSV, la cual multiplica a la imagen (que como tal es una matriz de datos o pixeles), este procesamiento modifica la información de la imagen como tal para obtener de ella la información que se requiere.

Este tipo de procesos se utiliza en el mundo del espectáculo como lo son las pantallas verdes o azules, así como en algunos procesos de agricultura, para diferenciar plantas sanas de enfermas.

El filtrado y la detección de contornos se refiere a la detección de las formas o fronteras de los objetos blancos de la imagen a la que se le aplico la máscara, una vez detectados estos objetos hace iteraciones sobre los contornos encontrados para descartar aquellos que sean demasiado pequeños (los cuales podrían ser ruido) y seleccionar aquellos cuya área sea mayor a 700 pixeles (siendo este un valor que se puede modificar según las condiciones en las que se esta trabajando), este selecciona los dos contornos más grandes detectados, los cuales a su vez deben encontrarse en sectores diferentes de la imagen, es decir, uno debe estar del lado izquierdo y el otro del derecho, pues al ser las líneas limitantes lo que se busca es encontrar el punto medio entre ambas, por lo cual sería inútil hacer uso de dos áreas seleccionadas si estas se encuentran en la misma sección.

Como tal, la biblioteca de OpenCV tiene una función específica para encontrar contornos, que lo que hace es utilizar la lista de coordenadas que definen los límites y forma de un objeto, a su vez, mediante el uso de otra función se filtran las áreas que se encuentran para solo centrarse en objetos de un tamaño considerable.

Estos mismos procesos se utilizan para aplicaciones de recuento de objetos, como contar el número de piezas de algún dispositivo o incluso el número de bacterias en una placa de Petri; así como el control de dimensiones en proceso de manufactura.

Los cálculos de momentos y centroides es la sección encargada de determinar el centro de la masa o centroide del objeto, es como tal su coordenada principal, es la encargada de control y seguimiento, pues le indica al robot donde esta el objeto, definiendo su posición en la imagen.

En primera instancia se calculan los momentos estadísticos que son propiedades como el área, centro, etc., haciendo uso de estos momentos se calculan las coordenadas del centroide, que como tal sería el centro geométrico del objeto amarillo detectado.

Finalmente se dibujan los contornos seleccionados y el área calculada sobre el frame original, el cual se muestra visualmente al usuario para que estos funjan de indicativo grafico para este.

El conjunto de estos pasos finaliza en lo que sería la visualización central y lo que serían los primeros pasos de la lógica de control.

Una vez procesada la imagen se calcula la línea central del frame de vídeo y dos líneas extras a 30 pixeles de la primera, esto para establecer los extremos en que las líneas amarillas deberían mostrarse, una vez que estas líneas se detectan calcula el punto medio entre los dos centroides de las áreas de los objetos amarillos detectados, para después calcular la distancia entre las dos líneas limite que se establecieron a cierta

distancia de la central, buscando que el punto medio entre los centroides se encuentre siempre dentro de este rango, teniendo como objetivo que se utilice para definir cuando se avanza utilizando los servomotores o hace un recorrido en línea recta.

Por último, para resolver una problemática presentada en las curvas, pero que también brindó un movimiento más fluido; al llegar a una curva una de las líneas amarillas de los limites externos de la pista desaparece, por lo que como tal no se podría hacer un calculo del centro entre las dos áreas pues solo estaría presente una, para esta sección la solución que se dio fue hacer que el vehículo se comportase como un seguidor de línea, lo cual sucede de la siguiente forma, una vez llega a la curva y una de las líneas limites desaparece, la que se mantiene presente en la imagen capturada se toma como referencia y el automóvil comienza a seguirla hasta que ambas líneas sean visibles dentro del frame; esta solución permitió que el movimiento del automóvil fuese mucho más suave y fluido sin la necesidad de tener que afectar la lógica mediante la cual se controla en las líneas rectas.

Finalmente, la sección de visión industrial se complemento mediante la implementación de un entorno digital, el cual permite entrelazar la Raspberry Pi con una computadora externa, manipulándola sin la necesidad de hacer uso de un monitor conectado a la misma, además, esta conexión dio la posibilidad de ver en tiempo real la imagen capturada por el vehículo, a la par, se agregaron elementos con los cuales el usuario puede interactuar para modificar los parámetros para la detección del color, los cuales son la tonalidad, saturación y valor, tanto en sus límites superiores como inferiores, con el fin de permitir que en el caso de que las condiciones lumínicas cambien el usuario pueda ajustarlos hasta que se detecte el color o si es que el color a filtrarse es uno distinto, aunque por defecto viene configurada para trabajar con el color y parámetros utilizados en este proyecto; permitiendo que se use en circuitos con diseños distintos y facilitando su uso.

Una vez finalizada la sección de visión artificial, se inició con el control, el cual es un control PI que limita el PWM con el cual están trabajando los servomotores, ya que la velocidad de los motores de tracción se mantiene constante.

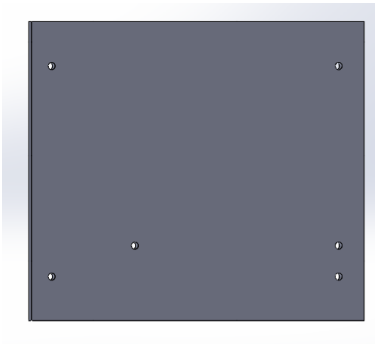
El control PI, controlador Proporcional-Integral, es un método de control muy común, los controladores PI hacen uso del cálculo del error, el cual es la diferencia entre el resultado obtenido y el esperado, en este caso, la posición a donde llega el vehículo y la esperada según lo indicado por el centro calculado, este error se utiliza para dos partes importantes, la acción

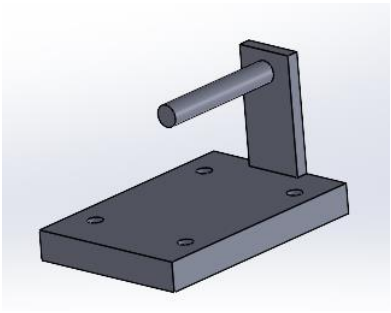
proporcional, que permite que exista una reacción instantánea al error, es decir, busca corregir el error de forma inmediata, según más grande el error mayor será la respuesta, una constante P muy pequeña va a ser incapaz de responder de forma adecuada al error, mientras que una muy grande haría que las respuestas sean desmedidas; por otra parte, la acción integral busca reducir el error a lo largo del tiempo mediante la integración del erro acumulado, ajustando la salida hasta que el error sea cero, permitiendo que el sistema sea mucho más estable y las respuestas sean adecuadas según la reducción del error.

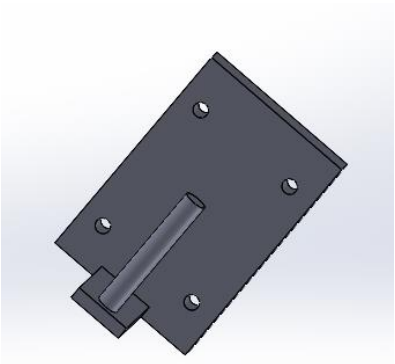
El controlador PI hace uso del error calculado para conseguir que el movimiento de los servomotores sea el adecuado, permitiendo que sus movimientos no sean muy agresivos una vez que se desvía de la posición objetivo y a la par, que pueda responder de forma idónea y rápida.

Para el funcionamiento de los motores, tanto servos como DC, se hizo uso de las librerías ya proporcionadas por el kit, las cuales son Adafruit y Servokit, ambas se encontraron en un repositorio sobre el equipo utilizado y fueron de gran utilidad para ahorrar trabajo, especialmente para los servomotores.

Por ultima instancia, se imprimieron dos piezas 3D, una para sujetar la cámara web desde una de las bases con las que ya contaba el automóvil, la otra pieza fue una base para colocar la Raspberry sobre el vehículo y que se mantuviese estable.

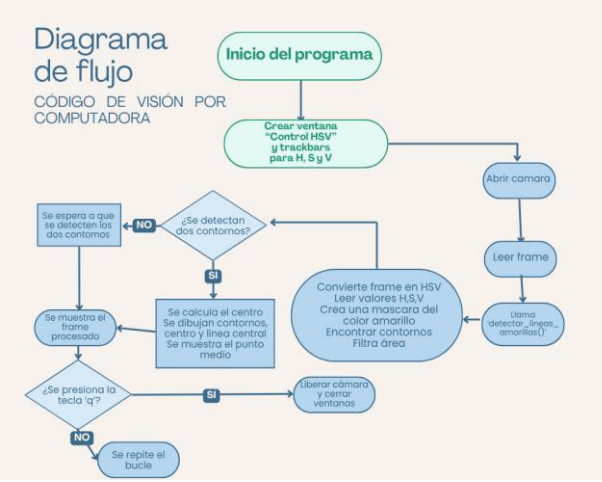




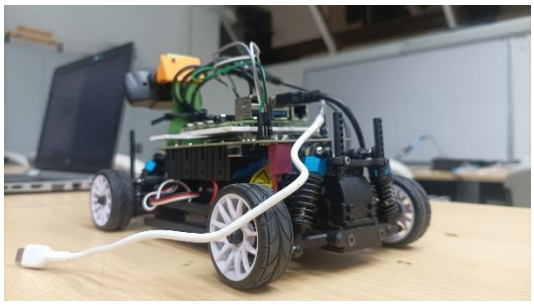


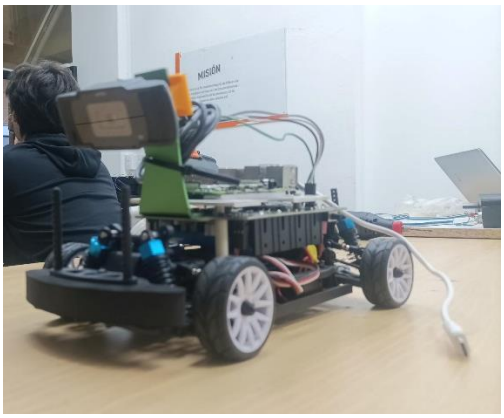
A la par, se adapto un cargador USBC para que pudiese alimentar la Raspberry con las baterías que tienen la placa de lo motores, evitando hacer uso de alguna alimentación externa; la razón por la cual se decidió hacer uso del USBC fue la seguridad, dado que mediante este método la misma placa cuenta con protocolos, los cuales, en caso de una descarga o una subida de corriente se dañe la misma.

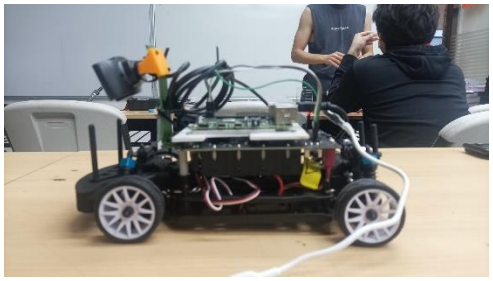
Diagrama de flujo

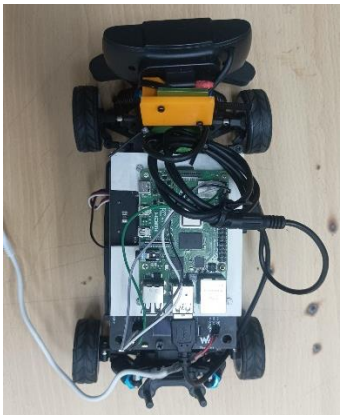


Fotografías del automóvil









Conclusiones

Finalmente, el desarrollo de este proyecto resultó como una fuente de conocimientos, así como el desarrollo de los ya adquiridos, así como la aplicación de nuevas herramientas para la búsqueda de soluciones, esperamos con gusto que quien tome este proyecto, ya sea como parte de su prácticum o una base para algún proyecto personal, que esta recopilación de información sea útil y que permita mejorar lo que nosotros conseguimos.