ANÁLISIS DE PROTOTIPO DE VEHÍCULO AUTÓNOMO CON BASE EN SISTEMA DE VISIÓN Y BAJO EL CONCEPTO DEL INTERNET DE LAS COSAS EN PLATAFORMA INTEL EDISON

ANALYSIS OF PROTOTYPE OF AUTONOMOUS VEHICLE BASED ON VISION SYSTEM AND UNDER THE CONCEPT OF THE INTERNET OF THINGS IN INTEL EDISON´S PLATFORM

Enrique Gerardo Hernández Vega
Tecnológico Nacional de México, Instituto Tecnológico de Chihuahua
ehernand@itchihuahua.edu.mx

Felipe Eliacim Garay Acuña
Tecnológico Nacional de México, Instituto Tecnológico de Chihuahua
fegaray@itchihuahua.edu.mx

Vicente González Navarro
Tecnológico Nacional de México, Instituto Tecnológico de Chihuahua
vgonzalez@itchihuahua.edu.mx

Miguel Ángel Gutiérrez Velázquez
Tecnológico Nacional de México, Instituto Tecnológico de Chihuahua
magutierrez@itchihuahua.edu.mx

Recepción: 6/noviembre/2019 **Aceptación:** 23/noviembre/2019

Resumen

El creciente campo de investigación que concierne a la autonomía vehicular, demanda el continuo desarrollo de tecnología que permita la maduración de este concepto, manifestando retos de eficiencia cada vez mayores relacionados a hardware y software. Este trabajo presenta el diseño, implementación y prueba de un vehículo autónomo en una plataforma con bajos recursos de cómputo. La primera versión se basa en un sistema de visión para percibir la ruta y obstáculos, algoritmos para la toma de decisiones; incorporando el concepto de modularidad e Internet de las cosas. El vehículo se probó en un entorno controlado (iluminación, cambio de color, obstáculos), teniendo la capacidad de navegar por un camino acotado por dos líneas, con obstáculos no preestablecidos para el sistema. Se

argumenta la posibilidad de implementar un algoritmo para detectar líneas curvas eficientemente, clases de objetos y diseñar un control difuso para manejar las instrucciones de conducción.

Palabras Claves: Autonomía vehicular, Internet de las cosas, sistema de visión.

Abstract

The constant growth in the investigation field concerning to vehicular autonomy, demands the continuous development of technologies that allow the maturity of this concept, stating efficiency challenges higher every time related to hardware and software matter. The paper presents the design, implementation and testing of an autonomous vehicle in a low resources hardware platform. The first version is vision-based system to perceive the route and obstacles, algorithms for decision taking; gathering the concept of modularity and internet of things. The vehicle was proved in a controlled environment (illumination, color changes, obstacles), having the capability to navigate by a two-line bounded path, with not preset obstacles in the system. It argues the possibility of implementing an algorithm for curved lines detection, object classes and designing a fuzzy control to manage driving instructions.

Keywords**:** *Internet of Things, self-driving car, vision system.*

1. Introducción

El transporte es usualmente dado por hecho por la mayoría de la gente y difícilmente se dan cuenta que forma la base de nuestra civilización. Mientras las ciudades y la población crecen, más tráfico existe, lo cual tiene muchos efectos adversos. La necesidad de un sistema de transporte más eficiente, balanceado y seguro es obvia. Esta necesidad puede ser alcanzada e implementada mejor por sistemas de transporte autónomo [Forrest, 2007]. La literatura existente está de acuerdo en que los vehículos completamente autónomos están por aparecer dentro de la siguiente década y que dentro de los siguientes 50 años gran parte de los vehículos en el camino contarán con una conducción autónoma completa [Axhausen, 2016].

Un vehículo autónomo es un vehículo sin conductor, capaz de satisfacer las principales capacidades de transporte de un carro tradicional, puede operar sin control humano y no requiere de ninguna intervención humana. El vehículo autónomo, a su vez, puede adquirir información del ambiente, clasificar diferentes tipos de objetos que detecta y puede interpretar la información para identificar caminos de navegación apropiados [Bimbraw, 2015].

Los sistemas de visión tienen una ventaja intrínseca sobre láseres o sensores de radio para el desarrollo de un vehículo autónomo: la posibilidad de adquirir información de una manera no invasiva, por lo tanto, no altera el ambiente. Los sistemas de visión pueden ser usados para algunas aplicaciones específicas para las cuales la información visual juega un rol básico: detección de líneas, localización, reconocimiento de señales de tráfico e identificación de obstáculos [Bertozzi, 2000]. Un enfoque usado es la detección de líneas, al detectar los bordes del camino y aplicar la transformada de Hough para detectar líneas.

En [Doshi, 2018] se realizó un algoritmo el cual toma una imagen, luego es convertida a escala de grises y después a una imagen binaria usando un umbral. La imagen es procesada con varios métodos de detector de bordes entre ellos Prewitt, Sobel, Canny, LoG y Roberts. A continuación, se selecciona una región de interés para limitar el procesamiento y que se procese la parte de la imagen que importa para la detección de líneas. Por último, se aplica la transformada de Hough para obtener las líneas. Un acercamiento similar se encuentra en [Assidiq, 2008] donde se sostiene que la razón de la gran mayoría de accidentes automovilísticos es debido a que los vehículos salen de su camino, así se justifica la importancia de la detección de líneas. El algoritmo presentado es similar al mencionado anteriormente, pero cuenta con algunas mejoras. Después de la detección de líneas se usa un escaneo de límites de líneas: como entradas se tienen las líneas de Hough, así como la línea del horizonte. El escaneo comienza donde las líneas de Hough intersectan el fondo de la imagen. Así se conoce mejor el inicio de línea. Se prosigue con un ajuste mediante una hipérbola, la cual toma las líneas halladas en el paso anterior y las hace coincidir en un punto. Por otra parte, la detección confiable y exacta de obstáculos es uno de los problemas principales a resolver en

la navegación autónoma. Existen dificultades debido a una gran variedad de objetos y condiciones de la carretera, características irregulares e inestables, objetos moviéndose, cambios de iluminación y de clima. En [Chen, 2000] se emplea un algoritmo para detectar y esquivar obstáculos, el cual comienza cuando ningún objeto aparece en la imagen, se utiliza un conjunto de información de colores y combinado con líneas y técnicas de seguimiento de líneas para guiar el vehículo por el camino. Cuando nuevos objetos aparecen en la imagen se almacenan para ser evaluados como obstáculos o no en el siguiente ciclo. Para decidir si el objeto es una rémora o no se extraen las características en la forma del objeto, además de predecir el movimiento.

Se puede observar en [Barea, 2018] un prototipo de vehículo autónomo diseñado para personas mayores en ambiente urbano. Se ha empleado una arquitectura de software modular, con 5 capas que envuelven dichos módulos: la capa hardware/simulación (el vehículo real), la capa de control de dispositivos (donde se reciben las señales de control de los dispositivos), la capa de control (encargada de la navegación y el control básico, así como el procesamiento de la información de los sensores), la capa de ejecución (que coordina las acciones de los módulos) y la capa de interfaz (que provee al usuario una manera de conectarse con el proceso de control directamente). El prototipo pretende permitir al vehículo una navegación autónoma en un entorno urbano dinámico y con obstáculos estáticos. Posibilita la comunicación entre procesos de forma independiente y modular, principios abordados en el prototipo presente. Se encuentran algoritmos para alcanzar objetivos desde la adquisición de datos hasta la toma de decisiones, con la capacidad de navegar con seguridad ante diferentes situaciones como la permanencia en un carril, detener el vehículo ante la presencia de una persona pasando el cruce peatonal, etc., utilizando sensores como el LIDAR 3D, GPS, y una ECU (que recibe como información el ángulo al que debe girar y genera una señal PWM para el control de los motores).

Para el diseño del prototipo de vehículo autónomo se reflexionó sobre la información consultada y los enfoques seguidos en las referencias de información; de esta manera se extrajeron algoritmos que ayudarían a la realización del prototipo, así

como soluciones a problemas debido al ambiente y al hardware utilizado. Se establece, entonces, que se llevó a cabo una investigación cualitativa y bibliográfica. El proyecto surge debido a la necesidad de adquirir experiencia y conocimiento a nivel institucional; con la finalidad de dejar antecedentes educativos y de investigación respecto a la autonomía vehicular con base en sistemas de visión e internet de las cosas. Por ende, el alcance es exploratorio, ya que el problema planteado es poco estudiado en la Institución y no ha sido abordado antes. Según la clasificación hallada en [Lin, 2018] el prototipo de vehículo autónomo desarrollado tiene el alcance de ser nivel 3, que se define como automatización condicional, en el cual un sistema de conducción automatizado realiza el conjunto de las tareas de conducción y el conductor debe estar disponible para intervenir y conducir si es preciso, como se encuentra catalogado en la tabulación de [SAE International, 2014].

El sistema maneja las tareas, con condiciones del ambiente controladas, y espera que el humano intervenga a solicitudes que el sistema pide. El vehículo realiza el procesamiento para detectar líneas y centroides de objetos que son obstáculos, entonces se accede a una página HTML para llamar las instrucciones que moverán al vehículo. El destino al que debe llegar está dado previamente, para llegar allí fueron colocadas bandas de referencia, landmarks, las cuales son guías para que el vehículo conozca dónde se encuentra. Se tuvieron en cuenta restricciones que afectan el rendimiento, como pueden ser los cuadros por segundo y la latencia [Lin S., 2018], además de las propias del ambiente en el cual el vehículo autónomo se desplazará. Algunas de ellas son inherentes, como la inclinación del suelo, y perjudican el funcionamiento.

2. Métodos

El sistema que se presenta es resultado de la unión de diferentes módulos con aplicaciones específicas, mismos que, trabajando en conjunto, dan al sistema la característica de modularidad. Se utilizaron diferentes herramientas para lograr el correcto funcionamiento del vehículo, las cuales se presentarán a continuación conforme se vaya explicando cada módulo.

La figura 1 provee una idea de las distintas partes que conforman el sistema modular, todas ellas comunicándose con el nodo principal que es la tarjeta de desarrollo Intel Edison.

Figura 1 Descripción modular del sistema.

El vehículo en su primera fase, está compuesto por 4 partes fundamentales que operan en conjunto para manifestar la autonomía. El procesamiento (ejecución de instrucciones), el sistema de visión, el control mecánico y la transmisión en tiempo real son módulos del sistema que son administrados por la parte central del sistema. Este módulo de cómputo es el encargado de administrar los recursos y las operaciones del sistema general, siendo el nodo principal donde se recolecta la información de todo el vehículo y de donde se envían las instrucciones a ejecutar en el mismo.

El vehículo es capaz de controlar su propia velocidad, teniendo las opciones preestablecidas, según las detecciones que se obtengan del sistema de visión. El dispositivo central se encarga de procesar las imágenes para así seleccionar las instrucciones correspondientes según la detección, se puede estar observando el camino por el que se mueve el vehículo a través de la transmisión en tiempo real, el nodo central se comunica con el módulo de control mecánico vía Wi-Fi, permitiendo al usuario intervenir en el manejo del vehículo de manera remota a través de una dirección IP. Se describen de manera detallada cada uno de los módulos para su mejor comprensión, donde se señalarán los puntos de enlace de un módulo con otro y las características de cada uno de ellos.

Módulo de Procesamiento (Tarjeta Intel Edison)

Constituye la parte central del sistema, dado que es la encargada de administrar a tiempo real las tareas requeridas para el control de autonomía del vehículo.

Para trabajar en esta plataforma se requiere la instalación de varios paquetes, programas y bibliotecas de desarrollo, de los cuales vale la pena destacar Python, OpenCV y NANO, siendo los pilares del proyecto. La descarga e instalación es realizada apegándose al proceso común de obtención de repositorios en distribuciones Linux, para este caso se maneja la distribución Yocto, por lo que los comandos suelen tener pequeñas variaciones.

Con la finalidad de adquirir cualquiera de los repositorios necesarios es recomendable tener una conexión de Internet con todos los puertos abiertos y ninguna restricción de navegación. La tarjeta Intel Edison facilita la conexión por red Wi-Fi, utilizando el comando «configure_edison --wifi», esto desplegará una lista de redes a disposición y se procede a acceder a una de ellas.

Al momento de comenzar la instalación de OpenCV y extensiones de Python imprescindibles para este proyecto, es necesario ingresar una memoria SD en el puerto disponible en el módulo de Arduino, en el cual viene montada la tarjeta de desarrollo Intel Edison, esto debido a la irremediable necesidad de espacio en memoria para su descarga, extracción e instalación. Utilizando el comando «opkg install python-numpy opencv python-opencv nano», con la memoria ya instalada en el dispositivo, se comienza el proceso anteriormente planteado. Se puede observar con claridad la tarjeta utilizada en la figura 2.

Figura 2 Tarjeta de desarrollo intel Edison.

Módulo para Control Mecánico

El vehículo debe tener control sobre tres aspectos básicos para la conducción: la dirección, la velocidad del vehículo y la reversa. La dirección del vehículo consiste en el manejo de un servomotor de alto torque, Futaba S3010, con 60° de libertad, es decir, al configurarse una posición de 60° se posicionan las llantas para un giro a la izquierda, si se configura a 120° se posiciona un giro a la derecha, y si se configura a 90° entonces las llantas están alineadas para ir hacia adelante, el servomotor tarda 200 milisegundos para avanzar 60°.

La dirección se controla por medio de señales moduladas en ancho de pulso, considerando los datos del servomotor, de donde se obtienen los anchos de pulso, la frecuencia necesaria y la codificación de color de los cables de conexión. Como se mencionó anteriormente, considerando el servomotor de marca Futaba con su respectivo ángulo de libertad de 60°-120°, por lo que el ancho de pulso variará de 1ms-1.4ms para controlar el ángulo de giro, mismos que se establecerán al momento de embeber el código en el dispositivo de control.

En cuanto a la velocidad del automóvil, cabe mencionar que también es controlada por medio de dos señales moduladas en ancho de pulso, PWM, esto para controlar la cantidad de potencia entregada a los motores, una señal para cada motor. Los motores se alimentan con una batería de Tenergy compuesta de NiCd, con 7.2 V a 2200 mAh, suficiente para otorgar al vehículo una buena velocidad. Además, se coloca un capacitor de 0.1 microfaradios en paralelo con las terminales del motor, esto para aportar al arranque del mismo un mayor torque para vencer el estado de reposo del automóvil.

Se utiliza un módulo L298N, que posee un arreglo tipo puente H para el control de giro del motor. Lo que permite por medio de 4 señales digitales controlar el sentido de giro de cada uno de los motores, también recibe 2 señales moduladas en ancho de pulso como entradas para controlar la potencia entregada a cada uno de los motores. Como se había mencionado, de acuerdo con la figura 3, el módulo L298N es intermediario entre el dispositivo de control mecánico y los actuadores, motores y servomotor. El mismo módulo posee una salida de 5v que se ha utilizado para alimentar el servomotor.

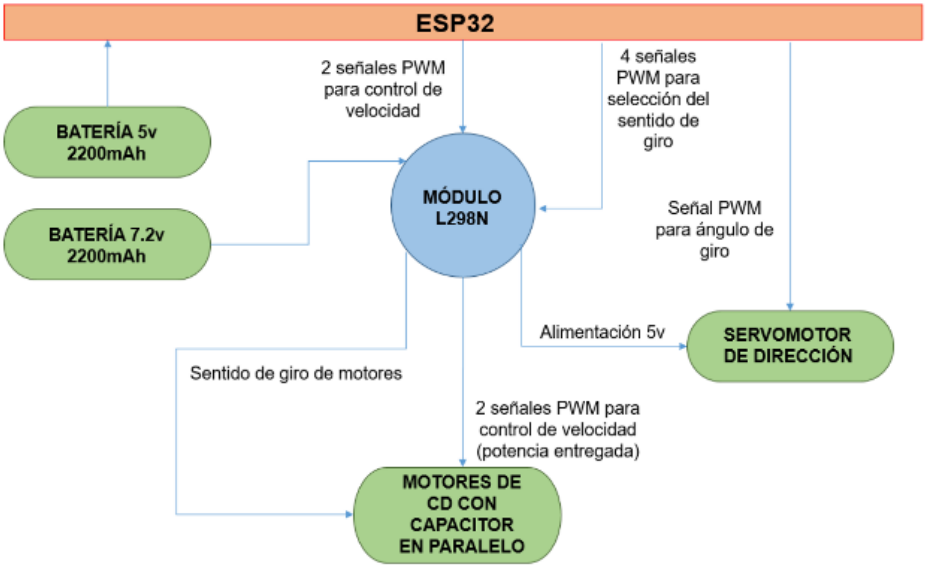


Figura 3 Esquema de módulo para control mecánico.

Asimismo, cabe mencionar que el control de toda la parte mecánica se ha centralizado en el módulo de cómputo ESP32, con procesador Xtensa LX6 de 32 bits, con la ventaja de poseer un módulo Wi-Fi integrado y ser compatible con Arduino. Así pues, es este dispositivo quien controla las acciones de toda la parte mecánica del sistema.

Se ha utilizado la arquitectura de diseño de software llamada cliente-servidor [Serain, 1995], para diseñar una página de Internet en lenguaje HTML, HyperText Markup Language. En el que se utilizó el IDE de Arduino que está basado en el lenguaje de programación C++, para establecer una página desde el módulo ESP32 como servidor. Ahí se presentan las opciones para el control del vehículo, es decir, se han preestablecido instrucciones para el control. Para la velocidad se presentan tres opciones: baja, media y alta; también se presentan las opciones para que el auto se mueve hacia enfrente, en reversa o que se detenga; hay tres opciones de dirección: izquierda, derecha y de frente (acomodar). De manera que el servidor se establece utilizando la dirección IP asignada a la tarjeta. Se ha utilizado el puerto 80 para el intercambio de petición-respuesta. La estructura de la página se diseñó de la manera más sencilla posible, para permitir que el tiempo de respuesta a cada petición fuera lo más rápido posible. La figura 4 muestra la manera en que se vería la página accediendo desde un explorador web externo al sistema.

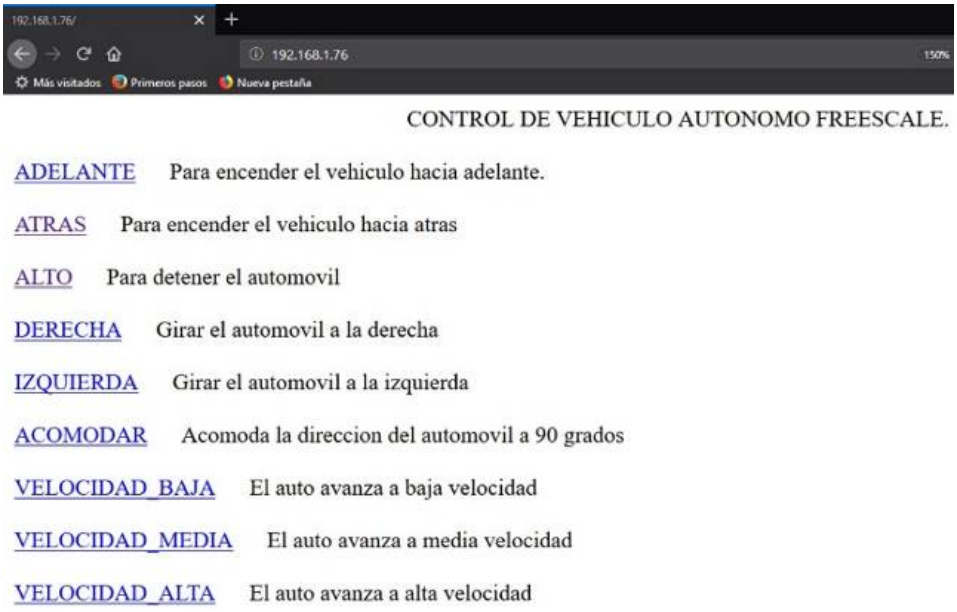


Figura 4 Diseño de página de internet.

Para facilitar el diseño de la página, se utilizaron las librerías WiFi.h y Servo.h, las cuales permiten manipular la señal PWM que controla el servomotor y el módulo Wi-Fi para la conexión a Internet. La tarjeta Intel Edison actúa como cliente, haciendo la petición de una instrucción específica a la vez, misma que es interpretada por la ESP32 (el servidor), y envía la señal correspondiente al módulo L298N para ejecutarse en el vehículo. De esta manera, puede observarse que la conexión entre el módulo mecánico y el módulo de ejecución es meramente inalámbrica, dando así la posibilidad al usuario de mantener cierto control, de ser necesario, sobre la parte mecánica del sistema.

Para poder monitorear la entrada y ejecución de una petición del cliente, se instaló un LED que estuviera cambiando su estado (toggle) cada vez que se ejecutara una instrucción. Por seguridad, se instaló un control para aislar la parte mecánica del prototipo de todos los dispositivos de alimentación, de manera que, al no tener el dispositivo de control en su posición, todo el sistema mecánico queda sin alimentación.

En la figura 5a se puede observar físicamente la localización del módulo de control mecánico del proyecto, y en figura 5b se observa de color rojo el dispositivo de control antes mencionado en la parte inferior derecha de la imagen.

a) Vista lateral *b) Vista superior*

Figura 5 Módulo de control mecánico.

Módulo del Sistema de Visión

Este módulo basa su funcionamiento en un algoritmo para detección de líneas y se divide en dos partes: el mejoramiento del frame y la detección de las líneas que aparecen en dicho frame. En primer término, se lee un frame y se cambia al espacio de color de RGB a HSV. Este modelo de color es definido en tres componentes: el matiz, la saturación y el valor. Se hace esto para resaltar el color de las líneas de interés, como se muestra en la figura 6. Se aplica una máscara con los valores mínimos y máximos que corresponden al color.

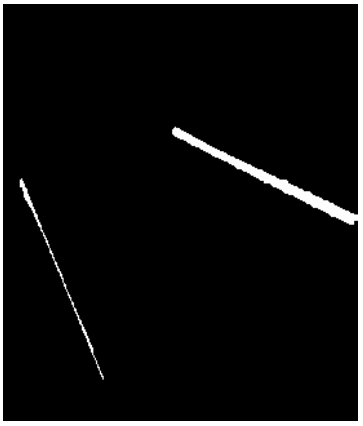


Figura 6 Obtención del color de las líneas en el espacio de color HSV.

A esta imagen se le realiza un proceso de difuminado, luego la operación morfológica de apertura; esto para eliminar el máximo ruido posible. A continuación, se aplica detección de bordes por el método de Canny, figura 7.

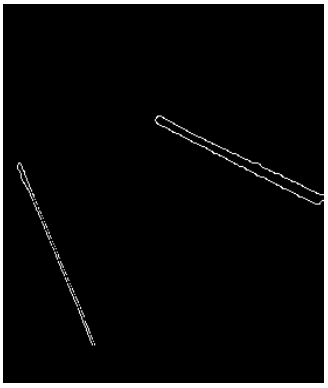


Figura 7 Detección de bordes.

Para la detección de líneas se implementa la transformada de Hough (ecuación 1), la cual es una técnica para aislar características de una forma específica dentro de una imagen.

$$\rho = xcos\theta + ycos\theta \quad (1)$$

Se trabaja con ecuación 1, donde ρ es la distancia perpendicular desde el origen del plano hasta la línea y θ es el ángulo formado por la línea y eje horizontal, figura 8.

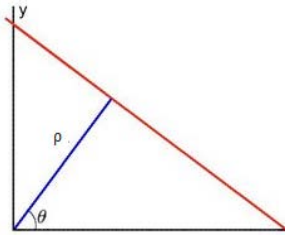


Figura 8 Cambio de parámetros de la recta.

Dados los puntos (x, y) y varios valores de θ se conoce el valor de ρ. Entonces se mapea la línea en el espacio (θ, ρ) como un punto, tal y como se muestra en la figura 9.

Todos los puntos mapeados se almacenan en un acumulador. Como se puede ver en la figura 10a, se tienen dos puntos ρ_0 y ρ_1, los cuales son mapeados con todas las posibles líneas que estos puntos pueden tener. Se describen para ambos puntos funciones sinusoidales, mostradas en el b) de la figura 10. El punto donde las curvas se intersectan representan la línea que pasa por ρ_0 y ρ_1.

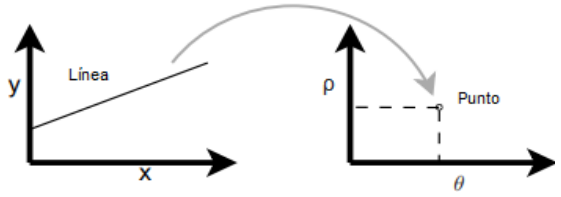
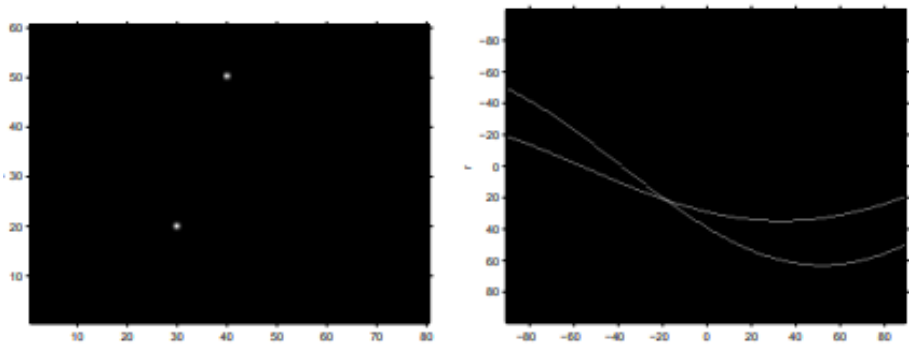


Figura 9 Mapeo de la línea en el plano (x, y) a un punto en el plano (θ, ρ).



a) Dos puntos en el plano (x, y). *b) Funciones sinusoidales*

Figura 10 Mapeo de dos puntos.

Al aplicar la transformada de Hough se mejora la imagen, figura 11.

Figura 11 Resultado de aplicar la transformada de Hough.

Al mismo tiempo que se hace la detección de líneas, se detectan los contornos, esto para que el vehículo esquive los posibles obstáculos en el camino. Sin embargo, lo que importa no es el contorno del objeto, sino el centroide; el cual se obtiene con las ecuaciones 2 y 3.

$$C_X = \frac{M_{10}}{M_{00}} \tag{2}$$

$$C_Y = \frac{M_{01}}{M_{00}} \quad (3)$$

Antes de obtener el centro del objeto, se obtiene el área cubierta por cada contorno detectado e implementando un umbral se decide si ese objeto es un obstáculo o no. Al final de ambos algoritmos se obtiene la figura 12.

Figura 12 Detección de líneas y del centroide de un obstáculo.

Para que el vehículo sepa por dónde ha de moverse, se obtiene el punto medio entre las dos líneas y se envía al control mecánico para que se mantenga ahí; si una línea no es detectada, gira dependiendo de cuál línea no aparece. Además, al saber dónde se localiza el centroide del obstáculo el vehículo se mueve en otra dirección para esquivarlo. Tal como se aprecia en la figura 13, la cámara utilizada se encuentra en la parte frontal del vehículo, mientras que en la figura 14 se puede ver el diagrama de flujo del algoritmo utilizado.

Figura 13 Vista frontal del prototipo mostrando cámara del sistema de visión.

Módulo de Transmisión en Tiempo Real

Se ha utilizado el puerto de red «5052», el cual garantiza la entrega de paquetes de datos en el mismo orden, lo cual es ideal para cubrir este aspecto. Se desarrolla

un método de transmisión de imagen en tiempo real, con la intención de subsanar la problemática de no contar con una interfaz gráfica para monitorear al vehículo.

Figura 14 Diagrama de flujo de algoritmo del sistema de visión.

Utilizando, a su vez, los «sockets» de red para lograr la comunicación por medio de la IP local y la remota. Esto debido a que, en la tarjeta de desarrollo, se cuenta con un programa local, el cual está encargado de enviar los cuadros de imagen por medio del puerto de preferencia; por otro lado, se tiene como resultado la posibilidad de visualizar, en cualquier dispositivo que cuente con la capacidad de ejecutar un archivo de Python el cual representa el archivo con la IP remota, las imágenes enviadas.

3. Resultados

Se pudo observar que, utilizando una conexión a Internet (red 1) con una latencia de entre 14 a 17 ms, con una red de 4.5 Mbps de subida y 12.89 Mbps de bajada y con un ancho de banda de 2.4 GHz, la comunicación entre los dispositivos Intel Edison y ESP32 es lenta, presentando un retardo entre ejecución de instrucciones de aproximadamente 0.5 segundos. Esta situación provoca que el vehículo pierda su camino, dado que, al recibir una instrucción retrasada, la acción ya no corresponde a la necesaria para mantener una navegación adecuada, presentando problemas de dirección y velocidad. En promedio, cuando varias instrucciones se envían seguidas el sistema presenta retardos mayores. Analizando los mismos casos con otra conexión a Internet (red 2) con una latencia menor, entre 12 a 14 ms, un aumento en la red a 20 Mbps de bajada, 1.5 Mbps de subida y conservando un ancho de banda de 2.4 GHz. Se muestran en la tabla 1 los resultados obtenidos,

con ambas redes, al enviar instrucciones y verificar el momento en que se ejecutan en el vehículo, luego se aumenta el número de instrucciones para comprobar el tiempo de retardo de ejecución.

Tabla 1 Tiempo de retardo en ejecución de instrucciones.

INSTRUCCIÓN	TIEMPO ENTRE CADA INSTRUCCIÓN	TIEMPO DE RETARDO DE EJECUCIÓN (RED 1)	TIEMPO DE RETARDO DE EJECUCIÓN (RED 2)
ADELANTE	No aplica	1 segundos	0.6 segundos
ADELANTE-ATRÁS	0.5 segundos	3 segundos	1.8 segundos
ADELANTE-ATRÁS-GIRO	0.5 segundos	3.2 segundos	2.2 segundos
ADELANTE-ATRÁS-GIRO-ALTO	0.5 segundos	3.7 segundos	2.4 segundos

El prototipo ejecuta la mayoría de las instrucciones, es decir, solamente un 5% del total de instrucciones enviadas se pierden, cuando se envía una cantidad mayor a 8 instrucciones en un intervalo de 3 segundos, 1 de cada 20 instrucciones se pierden y quedan sin ejecutarse dadas las múltiples peticiones al servidor. Lo que demuestra que el sistema es lo suficientemente seguro para conservar las instrucciones en el orden en que se enviaron. La transmisión en tiempo real presenta una alta latencia en su respuesta de actualización de cuadros, lo que dificulta poder seguir el vehículo en su recorrido remotamente. La tarjeta utilizada (Intel Edison) no logró procesar los cuadros obtenidos con la rapidez necesaria para poder hacer la toma de decisiones más eficiente, esto debido a que esta tarjeta tiene como fortaleza su capacidad de manejar sistemas con IoT y no meramente sistemas de visión.

Cuando el ángulo de inclinación de la línea es mayor de 60° y se extiende por toda la imagen, el programa no concluye de manera correcta hacia dónde debe moverse un 5% de las veces. Igualmente, se obtienen falsos positivos siempre que se presenten oclusiones, pues son detectados como un único objeto y localiza el centroide de toda la figura como si fuera un solo obstáculo. La incorporación final los módulos antes mencionados en el prototipo se puede contemplar en la figura 15.

4. Discusión

El prototipo presentado en este artículo posee cuatro módulos, que se comunican con el principal. Cada módulo realiza una función específica y trabajan de manera

paralela, tal como lo presentado en [Barea, 2018], donde la comunicación entre sus capas se presenta de forma independiente, permitiendo ejecutar algoritmos de adquisición de datos y toma de decisiones.

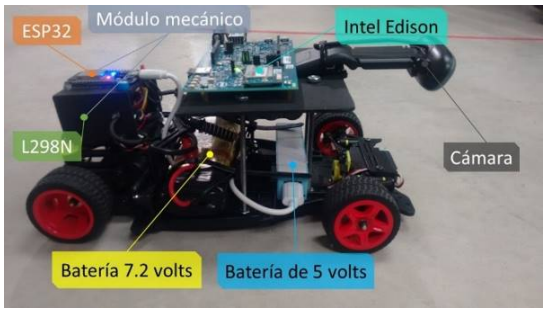


Figura 15 Prototipo funcional con módulos integrados.

El proyecto referenciado anteriormente aborda como objetivo los vehículos autónomos tripulados en un ambiente abierto. Los sensores LIDAR, para mapeos, GPS, para el posicionamiento, y un sistema de visión estéreo, para detección de profundidad y distancias, permitieron la autonomía suficiente para desplazarse por los pasillos de la universidad de Alcalá, y a diferencia de [A.S.P, 2016] donde solo se utiliza un método de detección de obstáculos basado en sensores infrarrojos y ultrasónicos. El prototipo en cuestión ha utilizado, únicamente, un sistema de visión monocular debido al objetivo inicial del proyecto; el cual es, solamente, con fines académicos. Se ha logrado la evasión de obstáculos estáticos en el ambiente, el seguimiento en un carril definido y el control completo del vehículo (en lo que a velocidad y dirección respecta). El algoritmo mostrado en [Doshi, 2018] para la detección de líneas emplea, al igual que este prototipo, la transformada de Hough. Sin embargo, existen diferencias importantes. Primeramente, en el proyecto propuesto se hace un cambio al espacio de color HSV para aislar el color de las líneas del camino, y se asegura que dichas líneas sí son las adecuadas ser detectadas. Además, se usó la operación morfológica de apertura, para eliminar el máximo ruido posible. Estos cambios llevaron una mejora significativa al comportamiento del algoritmo. Los resultados presentan la oportunidad de optimizar el comportamiento del módulo de control mecánico, implementando un control de

tipo no convencional, como lo es el control difuso, en lugar del algoritmo de control secuencial. Escalando la capacidad del sistema a un control automático de velocidad, y una mejora en la velocidad de respuesta de ejecución.

5. Conclusiones

En base a lo observado y obtenido a lo largo del desarrollo del vehículo autónomo se formularon las siguientes conclusiones:

La tarjeta de desarrollo Intel Edison, aun y cuando fue diseñada para cubrir el desarrollo de tecnologías del Internet de las cosas, no se limita exclusivamente a este uso. Se comprobó que puede ser utilizada para embeber algoritmos de sistemas de visión; sin embargo, estos tienen que ser de bajo consumo de recursos de cómputo.

Dada la información recolectada durante las pruebas de la investigación se infiere la necesidad de una conexión a Internet estable y de buena calidad, de esto depende la fiabilidad de la transferencia de información entre los módulos, ergo, la navegación idónea del vehículo.

El algoritmo del sistema de visión se diseñó con la intención de satisfacer dos puntos: ser implementado en una tarjeta que carece de recursos de procesamiento gráfico y permitir una integración sencilla con el resto de módulos del prototipo.

Para realizar un vehículo autónomo de mayor eficiencia, se propone la utilización de algoritmos más robustos para la detección del camino y distintos paradigmas para el reconocimiento de objetos y señalizaciones de tránsito: aprendizaje profundo, redes neuronales, lógica difusa.

6. Bibliografía y Referencias

[1] Assidiq A., Islam R., Khalifa O., & Khan S. Real time lane detection for autonomous vehicles. International Conference on Computer and Communication Engineering. Kuala Lumpur, Malaysia. May, 2008.

[2] Axhausen K., Ciari F., Hörl S, &. Recent perspectives on the impact of autonomous vehicles. Institute for Transport Planning and Systems. Zürich**,** Switzerland. September, 2016.

[3] A. S. P., Kharade P., Mandalollu L., Marali K., Savadatti P. Prototype Implementation of IoT based Autonomous Vehicle on Raspberry Pi. Bonfring International Journal of Research in Communication Engineering, Vol. 6, Special Issue, November 2016.

[4] Barea R., Bergasa L., López E., López J., Molinos E., Otero C., Paz E., Revenga P., Romera E., Sánchez P., Sanz R. Desarrollo de un vehículo eléctrico autónomo de código abierto para personas mayores. Jornadas Nacionales de Robótica, junio, 2018.

[5] Bertozzi M., Broggi A., & Fascioli A. Vision-based intelligent vehicles: State of the art and perspectives. Elsevier, Robotics and Autonoumus Systems. June, 2000.

[6] Bimbraw, K. Autonomous Cars: Past, Present and Future - A Review of the Developments in the Last Century, the Present Scenario and the Expected Future of Autonomous Vehicle Technology. 12th International Conference on Informatics in Control, Automation and Robotics. Colmar, Alsace, France. July, 2015.

[7] Chen K., & Tsai W. Vision-based obstacle detection and avoidance for autonomous land vehicle navigation in outdoor roads. Elsevier, Automation in Construction. 2000.

[8] Doshi N., Kathiresan S., Peddagolla Y., Prabhu N. & Zadeh M. On the lane detection for autonomous driving: A computational experiment study on performance of the Edge detectors. ResearchGate. June, 2018.

[9] Forrest A., & Konca M. Autonomous cars and society. Worcester Polytechnic Institute. May, 2007.

[10] Lin S., Hsu C., Skach M., & Zhang Y. The architectural implications of autonomous driving: Constraints and acceleration. Association for Computing Machinery. March, 2018.

[11] SAE International. Taxonomy and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems. United States. January, 2014.

[12] Serain D., Client/Server: Why? What? How? International Seminar on Client/Server Computing. October, 1995.