Revista Iberoamericana de Automática e Informática Industrial

Revista Iberoamericana de Automática e Informática Industrial 23 (2026) 80-91

www.revista-riai.org

Sistema de conducción automática para el vehículo AutoMINY

Oscar González Miranda,*, Juan Manuel Ibarra Zannatha

Departamento de Control Automático, CINVESTAV, Unidad Zacatenco, Av. Instituto Politécnico Nacional, 2508, San Pedro Zacatenco, 07360, CDMX, México.

To cite this article: Gonzalez-Miranda, O., Ibarra-Zanatha, J.M. 2026. Automated driving system for the AutoMINY. Revista Iberoamericana de Automática e Informática Industrial 23, 80-91. https://doi.org/10.4995/riai.2026.23613

Resumen

En este trabajo se diseña e implementa de forma experimental un Sistema de Conducción Automática (SCA) para el vehículo robótico a escala AutoMINY capaz de conducir sobre una carretera respetando la señalización vertical. Este SCA consta de (*i*) un sistema de percepción, el cual procesa las mediciones de los sensores a bordo y obtiene información útil sobre el estado del automóvil y de su entorno; (*ii*) un selector de maniobras, el cual toma la información provista por el sistema de percepción y elige la maniobra de conducción que se debe llevar a cabo; y (*iii*) los controladores de movimiento diseñados para llevar a cabo cada maniobra. Con este SCA el AutoMINY pudo reconocer 20 clases de objetos de interés y señales de tránsito, detectar el camino y los obstáculos a su alrededor, conducir siguiendo su carril a diferentes velocidades, llevar a cabo la maniobra de rebase al detectar otros autos estacionados en el camino, estacionarse en paralelo o en batería y detenerse al detectar peatones sobre su carril o semáforos en rojo.

Palabras clave: Vehículos autónomos, Sistema de conducción automática, Sistema de toma de desiciones, Control lateral, Estacionamiento automático

Automated driving system for the AutoMINY

Abstract

In this work, an Autonomous Driving System (ADS) is designed and experimentally implemented for the AutoMINY robotic vehicle at scale 1:10. This ADS consists of: (*i*) a perception system, which processes sensor measurements on board and obtains useful information about the car and its environment; (*ii*) a maneuver selector, which takes the information provided by the perception system and chooses the driving maneuver to be executed; and (*iii*) motion controllers designed to carry out each individual maneuver. With this ADS, the AutoMINY was able to recognize up to 20 classes of objects of interest and traffic signs, detect the road and obstacles around it, drive within its lane at different speeds, perform overtaking maneuvers when detecting slow or parked cars on the road, park in parallel or in a battery formation, and stop when detecting pedestrians, stop signals, or red traffic lights.

Keywords: Autonomous vehicles, Automated driving system, Decision making system, Lateral control, Automatic parking

1. Introducción

Un Sistema de Conducción Automático (SCA) es aquel conjunto de algoritmos, métodos matemáticos y técnicas de control que le permiten a un automóvil autónomo percibir su entorno, localizarse en un mapa, calcular una ruta y una trayectoria, y controlar su movimiento (Liu et al. (2018); Levinson et al. (2011); Ibarra-Zannatha et al. (2022)). El objetivo final es conseguir lo que se conoce como un nivel 5 de autonomía (Committee et al. (2014)), es decir, un automóvil que, sin intervención humana, es capaz de conducir a cualquier lugar, detectar las señales de tránsito u otros objetos de interés, tomar sus propias decisiones en ambientes dinámicos e interpolar información ante incertidumbres en las mediciones de sus sensores. Al mismo tiempo, debe seguir el reglamento de tránsito y

*Autor para correspondencia: ogonzalez31416@gmail.mx

conducir de forma segura para llevar a los pasajeros a un sitio determinado.

Si bien no existe un consenso de cómo integrar los sensores, los actuadores, el software y el hardware para construir un SCA, en la literatura aparecen muchas propuestas de SCA modulares como el que se muestra en la Figura 1. El Sistema de Percepción es aquel que toma las mediciones provistas por todos los sensores y obtiene información relevante sobre el estado del automóvil y de su entorno. Aquí se encuentran los métodos de visión artificial que detectan, localizan y clasifican objetos y señales de tránsito en imágenes, los métodos de mapeo y localización, los detectores de obstáculos y colisiones, entre otros. El Selector de Comportamiento toma la información provista por el Sistema de Percepción y la utiliza para diferentes fines como calcular la ruta y la trayectoria para llegar de un punto a otro o seleccionar la maniobra de conducción que debe llevarse a cabo. Este sistema es muy importante ya que debe ser capaz de responder ante los imprevistos que puedan aparecer en el ambiente dinámico y no determinista que le rodea. Finalmente, los Controladores de Movimiento son los que actúan directamente con los actuadores del automóvil y se diseñan para regular el movimiento tanto longitudinal como lateral de un automóvil con estabilidad y robustez, o bien para realizar alguna maniobra específica de conducción.

Uno de los primeros trabajos en presentar un SCA modular es (Urmson et al. (2008)) utilizado para controlar al vehículo BOSS en el DARPA Urban Challenge en 2007. Este vehículo contaba con cámaras, cuatro tipos de lidars, radares y GPS; su sistema de percepción se diseñó para obtener la localización del coche en la carretera, mapear los obstáculos estáticos alrededor del auto y modelar los obstáculos en movimiento. Por otro lado, el selector de comportamiento se construyó como una máquina de estados finitos que escogía entre distintas maniobras de conducción como conducir por su carril, mantener la distancia con el vehículo de enfrente, incorporarse a un carril o estacionarse. A pesar de su gran desempeño durante la competición la metodología empleada contempla situaciones muy específicas y requiere la programación de bastantes nodos.

Un trabajo más reciente es (Bansal et al. (2018)) en el cual se utilizó un SCA para un vehículo Waymo. El sistema de percepción que proponen procesa la información de las cámaras y del lidar para hacer un mapa local en el que se muestra el camino, la trayectoria deseada, los semáforos, los límites de velocidad, la localización del auto y la posición de otros agentes como automóviles, peatones, ciclistas, etc. Entonces, una red neuronal nombrada «*ChauffeurNet*» toma esta información y calcula la pose y la velocidad deseadas para el vehículo. Finalmente, un controlador de movimiento calcula el ángulo de dirección del volante, así como la velocidad del auto. Las pruebas experimentales con este SCA muestran cómo el automóvil puede conducir siguiendo la ruta deseada, detenerse ante señales de alto, dar vueltas en cruceros y rebasar otros automóviles; pero también se reportan problemas como rebases agresivos, trayectorias con curvas incapaces de seguir, incapacidad de dar vueltas en U, entre otros. Finalmente, cabe resaltar que el trabajo no utiliza métricas cuantitativas para evaluar el desempeño del SCA.

Por otro lado en (Badue et al. (2021)) se reporta el SCA diseñado para el vehículo autónomo IARA; el cual está equipado con una cámara estéreo, un GPS, un sensor inercial y dos lidars. Su sistema de percepción es capaz de detectar carriles, identificar señales de tránsito, modelar objetos móviles, hacer un mapeo de su entorno y obtener la autolocalización del auto. Su selector de comportamiento puede calcular la ruta y la trayectoria deseadas para llevar al automóvil de un punto a otro y utiliza una máquina de estados finitos para seleccionar la maniobra de conducción deseada. Finalmente, su controlador de movimiento utiliza una red neuronal para modelar la dinámica lateral y emplea el control por modelo predictivo para conducir a una velocidad máxima de 37 km/h. Esta metodología le permitió al automóvil IARA recorrer 74 km desde Vitória a Guarapari en Brasil en la noche del 12 de mayo de 2017. A pesar de este éxito, el uso de máquinas de estados finitos como selector de maniobras presenta grandes desventajas; por ejemplo, es difícil modelar las incertidumbres de los ambientes urbanos dinámicos, requiere la programación de una gran cantidad de reglas y no son capaces de reaccionar a situaciones no contempladas.

Alternativamente a los SCA modulares se han propuesto arquitecturas de *deep learning* que procesan la información provista por los sensores y controlan directamente el ángulo de dirección del volante, así como los pedales de aceleración y freno. A estas arquitecturas se les conoce en la literatura como *End-to-end* y un ejemplo de este enfoque es el trabajo de (Hawke et al. (2020)). En dicho trabajo se diseñó una red neuronal convolucional para procesar la información provista por tres cámaras a bordo de un mini auto eléctrico, a fin de calcular el ángulo de dirección del volante y la velocidad de avance deseada. El conjunto de entrenamiento se construyó con las demostraciones de un conductor experto mientras circulaba por un ambiente urbano densamente poblado y la red se entrenó para hacer una segmentación semántica de los objetos que aparecen en el camino, calcular la profundidad a la que se encuentran y medir el flujo óptico de las mismas. Con esta arquitectura *End-to-end* el automóvil pudo navegar siguiendo una ruta dada, conducir por un carril parcialmente señalizado, girar en cruceros, ceder el paso a otros automóviles, detenerse en pasos peatonales y obedecer el semáforo. Sin embargo, es difícil interpretar las métricas que utilizaron para medir el desempeño del SCA y se reportan múltiples intervenciones humanas durante la conducción.

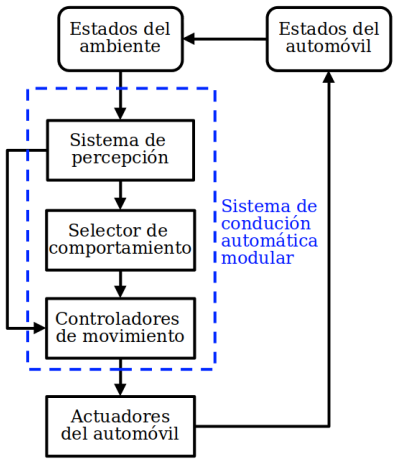

Figura 1: Sistema de conducción automático modular

En años recientes, diversas competiciones han impulsado el desarrollo de SCA para plataformas a escala 1:10 (Li et al. (2024)). Entre las más destacadas se encuentran la Bosch Future Mobility Challenge (Kilyen et al. (2021)), la Carolo-Cup (Nolte et al. (2018)) y la VDI Autonomous Driving Challenge (VDI (2025)), entre otras. Estas competiciones recrean entornos urbanos a pequeña escala con calles, intersecciones, rotondas y señalización vial; en las que se evalúan y validan algoritmos de control, percepción y planificación diseñados para vehículos autónomos.

Particularmente, en (Papafotiou et al. (2025)) se presenta el desarrollo de un SCA para un vehículo robótico a escala 1:10, cuyo sistema de percepción integra la información de los sensores para generar una representación tridimensional del vehículo y su entorno. Después, una máquina de estados finitos determina la maniobra de conducción más adecuada y se planifica la trayectoria correspondiente. El control de la dirección y la tracción se implementa mediante controladores PID, que actúan sobre el servomotor y el motor sin escobillas, respectivamente. Con esta arquitectura, el vehículo obtuvo el primer lugar en la Bosch Future Mobility Challenge 2024, superando retos como la navegación por intersecciones y rampas, el cruce de rotondas, el reconocimiento y cumplimiento de semáforos, así como la ejecución de maniobras de estacionamiento en paralelo.

En paralelo, en (Bächle et al. (2024)) se presenta un SCA para un vehículo a escala 1:10, diseñado para operar en entornos urbanos inteligentes. El sistema de percepción integra datos provenientes de una cámara RGB-D, una unidad de medición inercial (IMU) y los codificadores de las ruedas; con el objetivo de detectar objetos en el entorno mediante una red neuronal convolucional YOLOv5, identificar la carretera y estimar la pose del vehículo a través de odometría y fusión sensorial. Posteriormente, se realiza la selección de maniobras de conducción mediante un mapeo entre estados del vehículo y acciones específicas, y finalmente se utilizan controladores PID para regular la velocidad y el ángulo de dirección del vehículo. Este SCA permite al automóvil seguir su carril, ajustar su velocidad ante cruces peatonales, detenerse frente a señales de alto y ejecutar maniobras de estacionamiento en espacios delimitados.

La estructura del artículo es la siguiente. En la Sección 2 se describe el vehículo robótico utilizado en los experimentos. En la Sección 3 se presenta el SCA modular diseñado para dicho vehículo, la cual se divide en tres subsecciones. En la Subsección 3.1, Sistema de percepción, se abordan los métodos empleados para la detección del camino, el reconocimiento de objetos y señales de tránsito, así como la detección de obstáculos. La Subsección 3.2, Selector de maniobras, describe la red neuronal utilizada para seleccionar las maniobras de conducción. Por su parte, la Subsección 3.3, Controladores de movimiento, presenta el modelo matemático usado para describir la dinámica del vehículo y los controladores diseñados para cada maniobra de conducción. Después, la Sección 4 contiene los resultados experimentales, en los cuales se evalúa el desempeño tanto de cada maniobra de conducción como del selector de maniobras. En particular, para la maniobra de seguimiento de carril, se compara el desempeño de un controlador LQR basado en el modelo propuesto con el de un controlador MPC ampliamente utilizado en la literatura (Artuñedo et al. (2024); Rokonuzzaman et al. (2021); Chen et al. (2020)). Finalmente, en la Sección 5 se presentan las conclusiones del trabajo.

Las principales contribuciones de este trabajo son las siguientes. En primer lugar, se detalla la integración de diversos métodos para el diseño e implementación de un SCA modular. Dicha arquitectura abarca desde la adquisición de datos de los sensores hasta la evaluación del desempeño de cada controlador, y puede servir como base para el desarrollo de trabajos similares en el área. En segundo lugar, se demuestra que el controlador LQR, basado en el modelo matemático propuesto, presenta un mejor desempeño que un controlador MPC ampliamente reportado en la literatura, además de ofrecer una implementación más sencilla en escenarios reales ya que varios de los parámetros del modelo pueden medirse directamente. Finalmente, se presenta un selector de maniobras basado en una red neuronal entrenada a partir de una tabla de verdad; este enfoque es transparente y versátil, permite la incorporación sencilla de nuevas maniobras y requiere un proceso de entrenamiento simple. Asimismo, se incluyen resultados experimentales obtenidos con el vehículo AutoMINY, mostrando que esta estrategia constituye una alternativa viable a las máquinas de estados finitos ampliamente utilizadas en la literatura.

2. El vehículo AutoMINY

El vehículo AutoMINY es un automóvil a escala 1:10 diseñado por la Universidad Libre de Berlín para probar algoritmos de conducción autónoma (Berlin (2020)). En este trabajo se usó una versión modificada del mismo que cuenta con una cámara *fish-eye* ELP USBFHD01M, un lidar A2M8 360°, un sensor inercial Bosch BNO055, un servomotor Adafruit para controlar el sistema de dirección Ackerman, un motor sin escobillas Faulhaber 2232 012BX4SC para darle tracción a las 4 ruedas, un Arduino NANO para controlar los motores y una computadora Intel NUC8I5BEK. La computadora a bordo tiene el sistema operativo Linux Ubuntu 18.04 y el software de ROS Melodic para gestionar la comunicación entre todos los elementos del robot. En la Figura 2 se muestra el AutoMINY y algunas de sus partes.

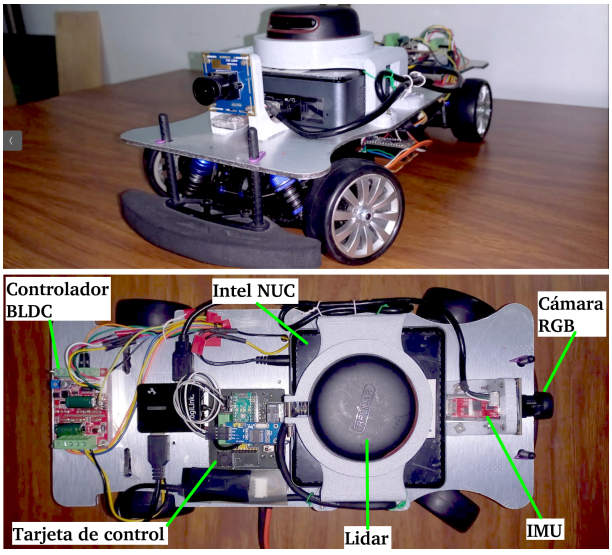

Figura 2: Vehículo AutoMiny

3. Sistema de conducción automática

El SCA diseñado para el AutoMINY le permite: conducir por una pista siguiendo su carril, evadir automóviles que aparezcan en el camino, estacionarse tanto en paralelo como en batería, reconocer 20 clases de objetos y señales de tránsito, detenerse al encontrarse peatones, señales de alto o semáforos en rojo; y cambiar su velocidad de acuerdo a los límites señalizados. Este sistema consta de tres módulos: (*i*) un sistema de percepción, (*ii*) un selector de maniobras de conducción y (*iii*) los controladores de movimiento.

3.1. Sistema de percepción

Este sistema toma la información adquirida por la cámara *fish-eye*, el lidar y el sensor inercial y la procesa para obtener mediciones relevantes acerca del vehículo y su entorno. Este módulo se compone de un sistema de detección del camino, un sistema para detectar, localizar e identificar objetos y señales de tránsito y un detector de obstáculos.

3.1.1. Detección del camino

Este módulo toma las imágenes capturadas por la cámara y las procesa para detectar las líneas del camino, siguiendo la metodología descrita a continuación. Algunos de los pasos de dicha metodología se ilustran en la Figura 3.

1. Dado que el suelo es la única región de la imagen que contiene información relevante para este módulo; se recorta la parte superior de la imagen situada encima de la línea del horizonte.

2. La imagen resultante se somete a un filtrado Gaussiano de 3×3 para reducir el ruido y se corrigen las distorsiones radial y tangencial de la cámara, utilizando los coeficientes obtenidos durante el proceso de calibración.

3. Posteriormente, se compensa la distorsión por perspectiva mediante una transformación de homografía, también obtenida durante la calibración. Como resultado, se obtiene una reconstrucción 2D del plano del suelo, en la cual todos los puntos se expresan en coordenadas métricas (centímetros).

4. El camino por el cual circuló el automóvil en estos experimentos se construyó con líneas de color naranja; por esta razón se realiza una segmentación de dicho color en el espacio HSV. Como resultado se obtiene una imagen binaria, en la que las líneas del camino aparecen en color blanco sobre fondo negro.

5. A partir de la imagen binaria se obtiene el conjunto de puntos $\{(x_i, y_i)\}_{i=1,\dots,N}$, que contiene todos los píxeles blancos correspondientes a las líneas delimitadoras del camino, expresados en coordenadas métricas.

6. Después, y solo con las primeras 5 imágenes de la cámara, se aplica el siguiente algoritmo para modelar al camino como un par de rectas. Este algoritmo se repite durante 100 épocas en cada iteración:

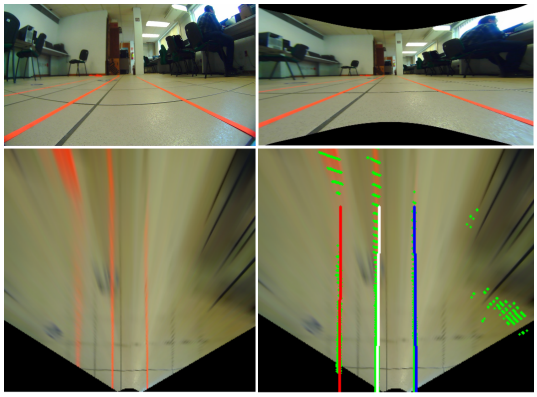

Figura 3: Sistema de detección de líneas. Se muestra la imagen original (arriba a la izquierda), la imagen sin distorsión radial ni tangencial (arriba a la derecha), la transformación de homografía (abajo a la izquierda) y las líneas detectadas (abajo a la derecha).

   6.1. Se define una matriz L de 2×2 como:

$$L = \begin{pmatrix} m_r & m_l \\ b_r & b_l \end{pmatrix}$$

   Las columnas de esta matriz contienen la pendiente m_i y la ordenada al origen b_i de la línea derecha e izquierda del camino. Estos parámetros se actualizan en cada época.

   6.2. Del conjunto $\{x_i, y_i\}_{i=1,\dots,N}$ se toma un punto $p = (x_p, y_p)$ de forma aleatoria y se crea una bola B de radio $\epsilon = 15\ cm$ centrada en dicho punto p.

   6.3. Si dentro de esta bola se tienen menos de 30 elementos se descarta p y se elige otro punto con su respectiva bola. De este modo se suprimen aquellos puntos que puedan ser ruido.

   6.4. Si la bola B tiene suficientes elementos, se hace una regresión lineal con sus elementos para obtener los parámetros m_p y b_p de la recta a la cual pertenece.

   6.5. Se mide la distancia entre el punto p y cada línea de las columnas de L, de modo que:

$$d_i = \frac{|y_p - m_i x_p - b_i|}{\sqrt{m_i^2 + 1}} \quad i = r, l$$

   Aquella columna con la menor distancia d_i, es la que debe actualizarse.

   6.6. Para actualizar las columnas de L se usa la siguiente regla de actualización:

$$m_i^{k+1} = m_i^k + \alpha(m_p - m_i^k)$$
$$b_i^{k+1} = b_i^k + \alpha(b_p - b_i^k)$$

   Teniendo a $\alpha = 0{,}85$ como parámetro de aprendizaje e $i = r, l$ según sea el caso.

7. Después de las primeras 5 iteraciones se obtiene una matriz L con las líneas derecha e izquierda que modelan al camino. A partir de aquí se usa el siguiente criterio para actualizar L con cada nueva fotografía:

7.1. Ya que la cámara funciona a 30 fps y la velocidad del vehículo es relativamente pequeña, podemos suponer que la posición de las líneas detectadas en una iteración no difiere mucho en la siguiente. Así que, para decidir si los puntos $\{x_i, y_i\}_{i=1,\dots,N}$ pertenecen a la línea de la derecha o a la de la izquierda; usamos un criterio de cercanía. Esto es, del conjunto $\{x_i, y_i\}_{i=1,\dots,N}$ se forman dos subconjuntos:

$$P_r = \left\{(x_j, y_j) | \frac{|y_j - m_r x_j - b_r|}{\sqrt{m_r^2 + 1}} \leq 30cm\right\}$$

$$P_l = \left\{(x_j, y_j) | \frac{|y_j - m_l x_j - b_l|}{\sqrt{m_r^2 + 1}} \leq 30cm\right\}$$

7.2. Después, con cada subconjunto se hace una regresión lineal con el método RANSAC; y con los parámetros obtenidos se actualizan las columnas de L correspondientes.

7.3. Para tratar el caso de que no aparezca una de las dos líneas o que una sola línea se detecte como izquierda y derecha al mismo tiempo, se añadieron instrucciones para: (*i*) tomar los parámetros m_i y b_i de la línea con mejor RMSE, (*ii*) usar esos parámetros para calcular una línea paralela, pero separada 60 *cm* (pues es el lugar donde se supone que debería estar la línea faltante) y (*iii*) actualizar L con esta nueva línea.

3.1.2. *Reconocimiento de objetos y señales de tránsito*

Para detectar, localizar e identificar objetos de interés en las imágenes capturadas por la cámara a bordo se utilizó una Red Neuronal Convolucional (RNC) con la arquitectura YOLOv3 (Redmon and Farhadi (2018); Redmon et al. (2016)). Se eligió esta arquitectura por ser compatible con las capacidades del hardware utilizado, por su buen desempeño y porque se puede utilizar en tiempo real. Con esta red se detectaron 20 clases de objetos diferentes: autos, camiones, bicicletas, motocicletas, trenes, peatones, perros, gatos, caballos; señales de tránsito como: alto (stop), no estacionarse, los tres estados del semáforo y los límites de velocidad de 20, 30, 40, 50 y 100 Km/h.

Esta red neuronal se entrenó con 43,099 imágenes, de las cuales 15,249 pertenecen al conjunto VOC-2012 (Everingham et al. (2012)); las demás se obtuvieron de diferentes fuentes y se etiquetaron manualmente con el programa LabelImg (Tzutalin (2015)). El entrenamiento se hizo con el software Darknet (Wang et al. (2022)) y su implementación como nodo de ROS se hizo con el software provisto por Bjelonic (2018). Al final del entrenamiento la precisión promedio de la red fue de $mAP = 62{,}98\,\%$.

3.1.3. *Detección de obstáculos*

El sensor lidar detecta obstáculos alrededor del automóvil y mide la distancia a la que se encuentran. Tiene una resolución de 1°, trabaja a una frecuencia de 10 Hz y puede medir distancias de entre 0,05 y 8,0 m. En cada iteración genera 360 vectores $\{(\rho_i, \gamma_i)\}_{i=1}^{360}$ en coordenadas polares, siendo ρ_i la distancia a la que se encuentra un objeto y γ_i su posición angular. Estos vectores pueden usarse para definir regiones de interés alrededor del auto y así obtener información útil mientras se realiza alguna maniobra de conducción. Por ejemplo, si existen vectores (ρ_i, γ_i) tales que $\rho_i < 0{,}9\ m$ y $-12° \leq \gamma_i \leq 12°$; entonces hay un objeto enfrente del auto a menos de 90 cm. Por otro lado, si reescribimos estos vectores en su forma cartesiana (x_i, y_i), y si aparecen objetos tales que $-0{,}3 \leq x_i \leq 0{,}2$ y $-0{,}47 \leq y_i \leq -0{,}1$ m; entonces, se puede deducir que hay un objeto a la derecha del auto. La Figura 4 ilustra estas situaciones; mientras que, en la Tabla 1 se muestran diferentes regiones de interés usadas para evadir obstáculos, estacionarse, detenerse, etc.

3.2. *Selector de maniobras*

Este módulo es el encargado de seleccionar la maniobra de conducción que debe realizarse tomando en cuenta la información provista por el sistema de percepción. En este trabajo se diseñó como una red neuronal *feed-forward* con cuatro entradas, cuatro salidas y cinco neuronas en una capa oculta. La Figura 5 muestra la arquitectura de dicha red. Cada salida está asociada a una maniobra de conducción y se denotan como: `Alto`, `Carril derecho`, `Carril izquierdo`, y `Estacionamiento`. Estas son variables binarias de las cuales solo una toma el valor de 1 a la vez (la cual corresponde a la maniobra de conducción que debe llevarse a cabo). En cambio, el vector de entradas de la red consta de 4 variables binarias denotadas como: `s_rojo`, `s_peat`, `s_ev`, y `s_est`. El valor de cada una depende de las siguientes circunstancias:

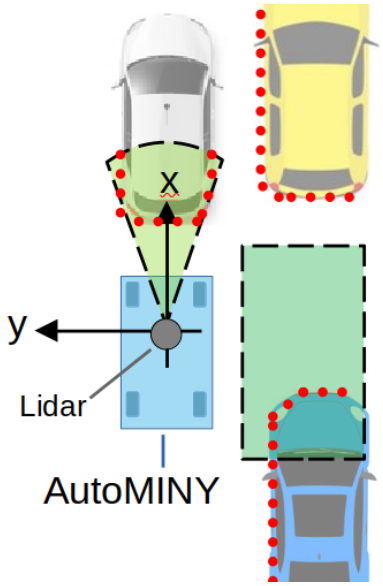

Figura 4: Ejemplos de regiones de interés alrededor del auto (áreas en verde). Para algunas maniobras de conducción, es importante detectar si un objeto (puntos rojos) está dentro o fuera de alguna de estas áreas.

Tabla 1: Regiones de interés usadas en diferentes maniobras de conducción. Unas se muestran en forma polar y otras en forma cartesiana. Las unidades de x_i, y_i y ρ_i son m.

Regiones de interés
$R_F = \{(\rho_i, \gamma_i) \mid 0{,}1 \leq \rho_i < 0{,}9 \text{ y } -12° \leq \gamma_i \leq 12°\}$
$R_f = \{(\rho_i, \gamma_i) \mid 0{,}1 \leq \rho_i < 0{,}24 \text{ y } -30° \leq \gamma_i \leq 30°\}$
$R_b = \{(\rho_i, \gamma_i) \mid 0{,}1 \leq \rho_i < 0{,}34 \text{ y } 160° \leq \gamma_i \leq 200°\}$
$R_0 = \{(x_i, y_i) \mid -1{,}0 \leq x_i \leq 1{,}0 \text{ y } -0{,}1 \leq y_i \leq -0{,}6\}$
$R_{par} = \{(x_i, y_i) \mid -0{,}3 \leq x_i \leq 0{,}2 \text{ y } -0{,}47 \leq y_i \leq -0{,}1\}$
$R_{bat} = \{(x_i, y_i) \mid -0{,}2 \leq x_i \leq 0{,}1 \text{ y } -0{,}67 \leq y_i \leq -0{,}1\}$

- Si la RNC YOLO detecta una señal de alto con una caja delimitadora de área mayor o igual a 2,450 px^2, la variable `s_rojo` toma el valor de 1; de lo contrario permanece en 0.

- Si la RNC YOLO detecta un peatón con una caja delimitadora de área mayor o igual 1,750 px^2 y el centro de la caja está dentro de cierta región de interés, la variable `s_peat` toma el valor de 1; de lo contrario permanece en 0.

- Si el lidar detecta un obstáculo en R_F y además la RNC YOLO detecta un automóvil con una caja delimitadora de área mayor o igual a 8,500 px^2; la variable `s_ev` toma el valor de 1 y permanece así hasta que el lidar detecta que el obstáculo se aleja a más de 30 cm de distancia en cualquier dirección.

- La variable `s_est` inicialmente toma el valor de 0 y de forma manual se cambia a 1 cuando se desea llevar a cabo la maniobra de estacionamiento.

Entonces, para entrenar a la red se construyó una tabla en la que se seleccionaba una salida para cada combinación de entradas. Por ejemplo, si `s_rojo = 0` y `s_peat = 1` entonces `Alto=1`; pues aunque no hay un semáforo en rojo se detecta un peatón.

Por otro lado, si `s_rojo = 0` y `s_peat = 0` entonces `Carril derecho=1` para conducir el auto por el carril derecho. Al final, se entrenó la red neuronal con esta tabla. Si bien la construcción de esta tabla puede resultar laboriosa se trata de una metodología que permite entender claramente cómo se toman las decisiones. Esto es particularmente importante cuando se analiza una falla o cuando se tiene que dilucidar las responsabilidades ante un accidente. Además, no es difícil extrapolar este método para añadir nuevas maniobras de conducción o información obtenida por otros sensores.

3.3. *Controladores de movimiento*

A continuación se describe el modelo matemático usado para representar al AutoMINY así como los controladores diseñados para realizar las siguientes maniobras de conducción: seguimiento del carril, evasión de obstáculos y el estacionamiento en paralelo o en batería.

3.3.1. *Modelado matemático del vehículo*

Para modelar el movimiento del AutoMINY primero se define un referencial $\{V\}$ montado sobre el coche de manera que el eje x_V es el eje de avance y y_V es el eje lateral. También se define un referencial inercial $\{E\}$ desde donde se mide la pose del automóvil; y un referencial $\{E'\}$ el cual es $\{E\}$ rotado en la dirección del ángulo de guiñada (*yaw*) ψ del auto. La Figura 6 ilustra esta situación en la que además aparecen: un punto sobre la trayectoria que se desea seguir $\vec{r}_R$, un vector de referencia $\vec{r}_h$ paralelo a x_V, una representación paramétrica $(x_R, f(x_R))$ de $\vec{r}_R$ desde $\{V\}$ y (x_v, y_v) es la posición de $\{V\}$ desde $\{E'\}$. Si se supone que el auto viaja a una velocidad de traslación (v_x, v_y) y con una velocidad angular ω_z; se puede deducir que:

$$\begin{aligned}
\vec{r}_R &= x_v\hat{\imath} + y_v\hat{\jmath} + x_R\hat{\imath} + f(x_R)\hat{\jmath} \\
\vec{r}_h &= x_v\hat{\imath} + y_v\hat{\jmath} + x_h\hat{\imath} \\
\Rightarrow \dot{\vec{r}}_R &= v_x\hat{\imath} + v_y\hat{\jmath} + \dot{x}_R\hat{\imath} + x_R(\omega^{ref} \times \hat{\imath}) + \dot{f}(x_R)\hat{\jmath} \\
&\quad + f(x_R)(\omega^{ref} \times \hat{\jmath}) \\
\dot{\vec{r}}_h &= v_x\hat{\imath} + v_y\hat{\jmath} + \dot{x}_h\hat{\imath} + x_h(\omega \times \hat{\imath})
\end{aligned}$$

Pero como:

$$\begin{aligned}
\vec{\omega} \times \hat{\imath} &= \omega_z\hat{\jmath} \\
\vec{\omega} \times \hat{\jmath} &= -\omega_z\hat{\imath} \\
\Rightarrow \dot{\vec{r}}_R &= v_x\hat{\imath} + v_y\hat{\jmath} + \dot{x}_R\hat{\imath} + x_R\omega_z^{ref}\hat{\jmath} + \dot{f}(x_R)\hat{\jmath} \\
&\quad - f(x_R)\omega_z^{ref}\hat{\imath} \\
\dot{\vec{r}}_h &= v_x\hat{\imath} + v_y\hat{\jmath} + \dot{x}_h\hat{\imath} + x_h\omega_z\hat{\jmath}
\end{aligned}$$

Entonces, se define al error de posición $\vec{e}$ relativo a la trayectoria como:

$$\begin{aligned}
\vec{e} &= \vec{r}_R - \vec{r}_h \\
\Rightarrow \dot{\vec{e}} &= \dot{\vec{r}}_R - \dot{\vec{r}}_h \\
&= \dot{x}_R\hat{\imath} + x_R\omega_z^{ref}\hat{\jmath} + \dot{f}(x_R)\hat{\jmath} - f(x_R)\omega_z^{ref}\hat{\imath} - \dot{x}_h\hat{\imath} - x_h\omega_z\hat{\jmath} \quad (1)
\end{aligned}$$

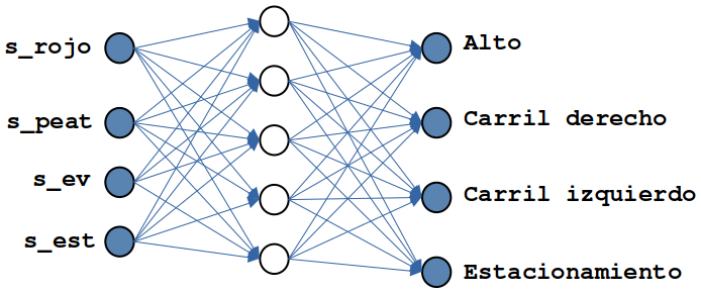

Figura 5: Arquitectura de la red *feed-forward* usada como selector de maniobras.

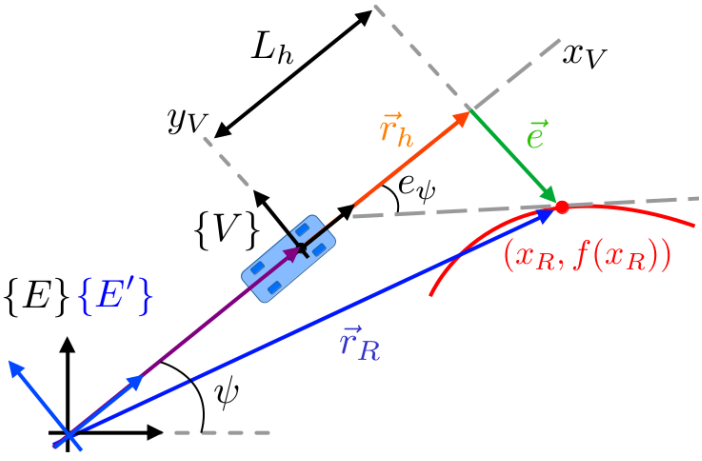

Figura 6: Modelado del movimiento de un automóvil. $\{V\}$ es un referencial montado sobre el vehículo mientras que $(x_R, f(x_R))$ es un punto sobre la trayectoria que se desea seguir (en rojo). e_ψ es el ángulo que se forma entre el eje longitudinal del auto y la tangente.

Tomando en cuenta que:

- $x_h = x_R = L_h$ es la distancia que hay entre el vehículo y el punto donde se hace la medición del error de posición $\vec{e}$.
- $f(x_R) = e_y$ es la componente lateral de $\vec{e}$.
- $\dot{x}_h = 0$ porque el punto x_h donde se hace la medición del error no se desplaza con respecto a $\{V\}$.
- $\dot{x}_R = v_x$ porque el punto x_R se desplaza conforme el auto avanza.
- e_ψ es la pendiente de la recta tangente a $(x_R, f(x_R))$ de modo que:

$$\begin{aligned}\dot{f}(x_R) &= \dot{x}_R \frac{df(x_R)}{dx_R} \\ &= v_x \tan(e_\psi)\end{aligned}$$

Entonces, se pueden concatenar estas observaciones junto con (1) para obtener:

$$\dot{\vec{e}} = \left(v_x - e_y \omega_z^{ref}\right)\hat{\imath} + \left(v_x \tan(e_\psi) - L_h\omega_z + L_h\omega_z^{ref}\right)\hat{\jmath} \quad (2)$$

O bien, separando (2) en componentes se obtiene:

$$\begin{aligned}\dot{e}_x &= v_x - e_y\omega_z^{ref} \\ \dot{e}_y &= v_x \tan(e_\psi) - L_h\omega_z + L_h\omega_z^{ref}\end{aligned} \quad (3)$$

La componente longitudinal $\dot{e}_x$ muestra la velocidad a la que se desplaza $\vec{e}$ conforme avanza el automóvil; mientras que $\dot{e}_y$ es la dinámica que se desea controlar.

Por otro lado, del modelo de la bicicleta ilustrado en la Figura 7, se puede relacionar el ángulo de dirección δ de la rueda delantera con la velocidad angular ω_z de la siguiente forma:

$$\begin{aligned}\text{Como: } \tan(\delta) &= \frac{L}{R} \\ \Rightarrow \frac{\tan(\delta)}{L} &= \frac{1}{R} \\ \text{También: } v_x &= R\omega_z \\ \Rightarrow \frac{\omega_z}{v_x} &= \frac{1}{R} \\ \Rightarrow \frac{\omega_z}{v_x} &= \frac{\tan(\delta)}{L} \\ \therefore \omega_z &= v_x \frac{\tan(\delta)}{L}\end{aligned} \quad (4)$$

Entonces, sustituyendo (4) en la segunda ecuación de (3), el error de posición lateral e_y se puede modelar como:

$$\dot{e}_y = v_x \tan(e_\psi) - v_x \frac{L_h}{L}\tan(\delta) + L_h\omega_z^{ref} \quad (5)$$

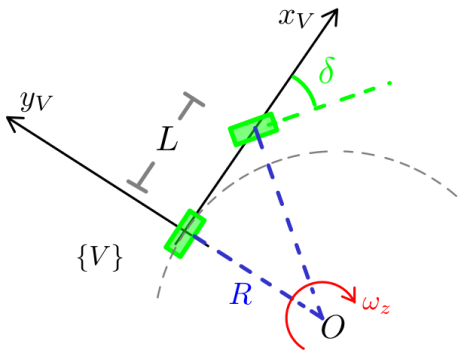

Figura 7: Modelo de la bicicleta de un automóvil en el que se sustituyen las cuatro ruedas del auto por dos. Una fija en el centro del eje trasero y una móvil en el centro del eje delantero. L es la separación de ambos ejes, δ es el ángulo de dirección, O es el centro de giro instantáneo y R es el radio de giro.

Además, si error de orientación e_ψ relativo a la carretera se define como la diferencia entre el ángulo de guiñada (*yaw*) deseado ψ^{ref} para seguir la trayectoria dada menos el ángulo de guiñada actual ψ, se tiene que:

$$\begin{aligned}e_\psi &= \psi^{ref} - \psi \\ \Rightarrow \dot{e_\psi} &= \omega_z^{ref} - \omega_z \\ \text{Definiendo: } \rho^{ref} &\triangleq \frac{1}{R^{ref}} \\ \Rightarrow \omega_z^{ref} &= V_x\rho^{ref} \\ \therefore \dot{e}_\psi &= V_x\rho^{ref} - \frac{v_x}{L}\tan(\delta)\end{aligned} \quad (6)$$

Finalmente, para ángulos $e_\psi \leq 34{,}38°$ en los que $\tan(e_\psi) \approx e_\psi$, el modelo de 2 GDL que resulta de unir (5) y (6) es:

$$\begin{aligned}\begin{pmatrix}\dot{e}_y \\ \dot{e}_\psi\end{pmatrix} &= \underbrace{\begin{pmatrix}0 & v_x \\ 0 & 0\end{pmatrix}}_{A}\underbrace{\begin{pmatrix}e_y \\ e_\psi\end{pmatrix}}_{\chi} - \underbrace{\frac{v_x}{L}\begin{pmatrix}L_h \\ 1\end{pmatrix}}_{B}\tan(\delta) + \underbrace{v_x\rho^{ref}\begin{pmatrix}L_h \\ 1\end{pmatrix}}_{W} \\ y &= \underbrace{\begin{pmatrix}1 & 0 \\ 0 & 1\end{pmatrix}}_{C}\begin{pmatrix}e_y \\ e_\psi\end{pmatrix}\end{aligned} \quad (7)$$

Este modelo es lineal si la velocidad de avance v_x se considera como un parámetro de las matrices A y B del sistema. Más aún, es fácil demostrar que es controlable siempre que $v_x > 0$. Adicionalmente, $\tan(\delta)$ se usa como una señal de control, alterada por una perturbación W que depende de L_h, v_x y de la curvatura de la trayectoria dada ρ^{ref}.

3.3.2. *Seguimiento del carril*

Del modelo definido en (7) se resalta que los estados se pueden medir utilizando la cámara a bordo del automóvil. En la Figura 8 se muestra cómo se miden estas cantidades utilizando una homografía del camino y el método de detección de líneas descrito anteriormente. Si (x_1, y_1) es un punto sobre la línea de la derecha del carril, podemos deducir que:

$$\begin{aligned}y_1 &= m_r x_1 + b_r \\ e_y &= y_1 - y_{ref} \\ e_\psi &= -\arctan(m_r)\end{aligned} \quad (8)$$

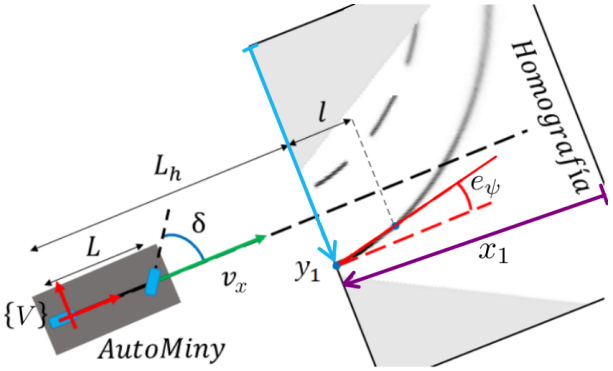

Figura 8: Medición de los errores relativos a la carretera, usando la homografía del suelo y el detector de líneas del camino.

Siendo $y_{ref} = 165\ cm$ la mitad del ancho de la homografía (150 cm) más la mitad del ancho del carril (15 cm). Así pues, para controlar al sistema definido en (7) se propone como ley de control:

$$\delta = \arctan(-K\chi) \tag{9}$$

En donde $K = (0{,}2495, 2{,}8531)$ es un vector de ganancias sintonizado mediante el método LQR.

3.3.3. *Evasión de obstáculos*

Al iniciar la maniobra de evasión el automóvil deja de seguir el carril derecho y pasa al carril izquierdo. Entonces, al medir los errores relativos definidos en (8), se sustituyen los parámetros (m_r, b_r) por (m_l, b_l) y se redefine $y_{ref} = 135\ cm$ como la diferencia entre la mitad del ancho de la homografía menos la mitad del ancho del carril. Con estos cambios, se hace la conmutación del carril con la misma ley de control (9); y, al finalizar la maniobra, la medición de los errores relativos vuelve a calcularse como se hacía antes de la maniobra de adelantamiento.

3.3.4. *Estacionamiento automático*

El auto puede realizar la maniobra de estacionamiento ya sea en paralelo o en batería, eligiendo la que sea compatible con el espacio disponible para estacionarse. El algoritmo correspondiente es el siguiente:

1. Para poder alinear al AutoMINY con los vehículos de la zona de estacionamiento, éstos se modelan como una recta con parámetros m_{lidar} y b_{lidar}. Así que, para obtener estos parámetros, se toman los datos generados por el lidar, se reescriben en forma cartesiana y se crea el subconjunto R_0 tal y como fue definido en la Tabla 1. Con ese subconjunto se hace una regresión lineal mediante el método RANSAC (Pedregosa et al. (2011)). La Figura 9 ilustra la recta así obtenida.

2. Después, mientras el automóvil se desplaza a $36{,}4 \pm 3{,}6\ cm/s$, se usa el controlador lateral (9), pero con una ganancia $K = (0{,}2, 9{,}2)$ y midiendo los errores relativos de la siguiente manera:

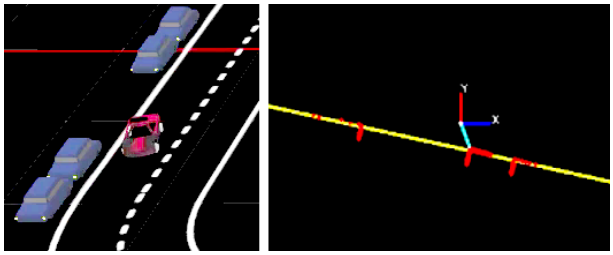

Figura 9: A la izquierda, el AutoMINY alineándose con los demás coches. A la derecha, los automóviles detectados (puntos rojos) y modelados como una línea recta (en amarillo). El referencial que se muestra esta anclado al lidar.

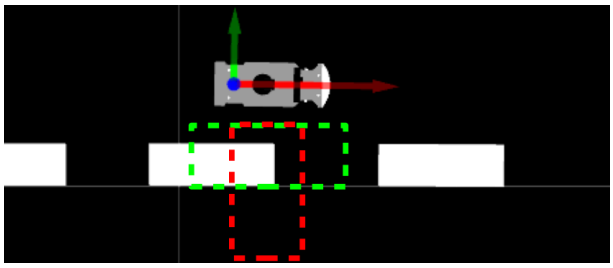

Figura 10: Regiones de interés definidas para decidir si se hace la maniobra de estacionamiento en paralelo (en verde) o en batería (en rojo).

$$\begin{aligned} e_y &= \frac{|b_{lidar}|}{\sqrt{m_{lidar}^2 + 1}} - d_{ref} \\ e_\psi &= -\arctan(m_{lidar}) \end{aligned} \tag{10}$$

De este modo el auto se mantiene separado $d_{ref} = 17\ cm$ de la línea y se orienta paralelo a la misma.

3. Pasadas las primeras 30 iteraciones el conjunto R_0 se construye con aquellos puntos (x_i, y_i) tales que su distancia a la recta de la iteración anterior es menor a 5 cm. Posteriormente, se hace una regresión lineal con RANSAC y se actualizan los valores de m_{lidar} y b_{lidar}.

4. Mientras el AutoMINY se alinea con los demás vehículos se definen dos regiones de interés para decidir cuál de las dos opciones de estacionamiento debe realizarse. Estas dos regiones son los rectángulos R_{par} y R_{bat} definidos en la Tabla 1 y que se ilustran en la Figura 10. Si ninguno de los puntos del lidar cae dentro de R_{par} (rectángulo en verde), se realiza la maniobra de estacionamiento en paralelo. Por otro lado, si ninguno de los puntos del lidar cae dentro de R_{bat} (rectángulo en rojo), pero sí dentro de R_{par}, entonces se realiza la maniobra de estacionamiento en batería.

5. Una vez seleccionada la maniobra de estacionamiento, ésta se lleva a cabo de acuerdo a los diagramas de flujo mostrados en las Figuras 11 y 12. En dichas figuras $\Delta\psi$ representa el cambio en el ángulo de guiñada medido con el sensor inercial. En cambio, R_f y R_b son las regiones de interés (definidas en la Tabla 1) usadas para buscar objetos a punto de colisionar con el auto, por enfrente o por detrás, respectivamente.

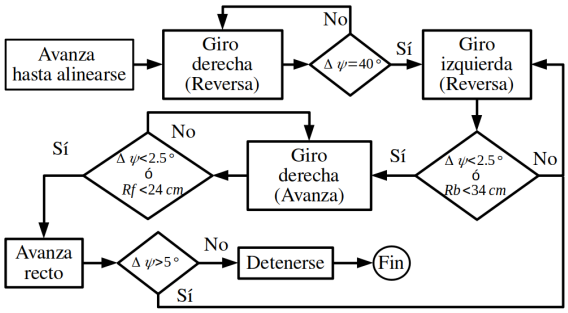

Figura 11: Diagrama de flujo usado para programar la maniobra de estacionamiento en paralelo.

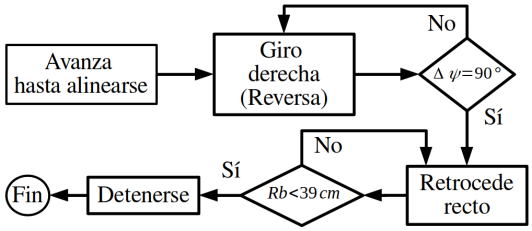

Figura 12: Diagrama de flujo usado para programar la maniobra de estacionamiento en batería.

4. Resultados experimentales

A continuación se describen los experimentos diseñados para evaluar el desempeño de cada controlador así como del selector de comportamiento. Todo el software desarrollado para este robot se puede consultar en `https://github.com/sherlock-gmo/Autominy_REAL`.

4.1. *Sistema de ground-truth*

Para poder medir el desempeño de los controladores se diseñó un sistema de *ground-truth* el cual mide constantemente la pose del AutoMINY. Este sistema consta de una cámara RGB *fish-eye* ELP USB100W03M-BL170 colocada en el techo del laboratorio y de un marcador de color montado sobre el vehículo. El sistema toma las imágenes de la cámara, les quita las distorsiones radial y tangencial, aplica una transformación de homografía calibrada a la altura del marcador y paralela al suelo; segmenta los colores del marcador y calcula los centroides de las máscaras resultantes. La Figura 13 muestra al vehículo con su marcador de color (arriba a la izquierda) y el referencial fijo desde donde se mide la pose (arriba a la derecha). La orientación del auto queda definida con el vector que une a los centroides de ambos colores. Con este sistema se pudo medir la pose del coche con un periodo de muestreo de $h = 0{,}0382$ s, una incertidumbre de $\pm 1{,}0$ cm y en un espacio de trabajo de 370×395 cm.

4.2. *Seguimiento del carril*

Para evaluar el desempeño del controlador (9), el vehículo se condujo de manera automática sobre la pista anaranjada que se muestra en la Figura 13. Cada curva tiene un radio exterior de 90 cm, un radio interior de 60 cm por lo que el ancho del carril es de 30 cm. El automóvil recorrió la pista a cinco velocidades diferentes: 30,7, 44,0, 56,6, 68,5 y 82,7 cm/s.

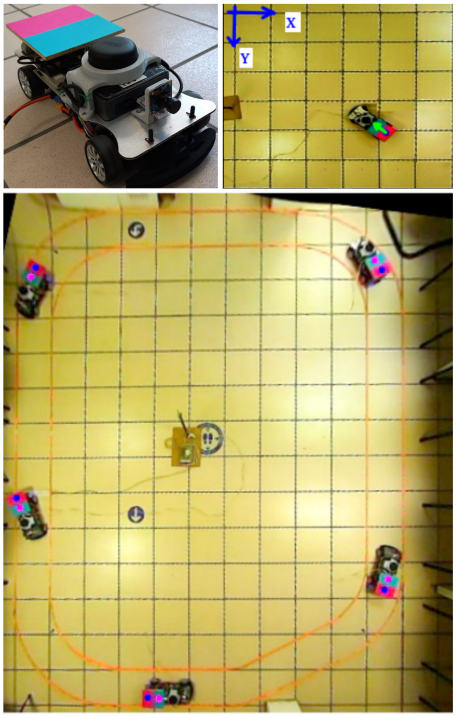

Figura 13: Arriba a la izquierda, el vehículo con su marcador de color. Arriba a la derecha, una parte de la homografía donde se ilustra el referencial fijo desde donde se mide la pose y el vector que define la orientación del AutoMINY. Abajo, pista donde se condujo al AutoMINY (en anaranjado). También se muestran los centroides del marcador de color del coche (en cian y magenta).

Para cada velocidad se hicieron seis experimentos diferentes y en cada experimento se varió la pose inicial coche de manera que $e_{y0} \in [-6, 12]$ cm y $e_{\psi 0} \in [-36{,}9°, 26{,}6°]$. La Figura 14 muestra la pose del coche mientras recorre la pista. Por su parte, la Tabla 2, en los renglones correspondientes al controlador **LQR**, presenta la velocidad de desplazamiento, el error cuadrático medio (RMSE) del error lateral e_y, la desviación lateral máxima máx(e_y) y el gasto energético del controlador (GEC). Específicamente, para cada ronda experimental con N datos, el RMSE y el GEC se calcularon como:

$$RMSE = \sqrt{\frac{1}{N}\left(\sum_{i=1}^{N} e_{yi}^2\right)}, \quad GEC = \sum_{i=1}^{N} \delta_i \cdot h$$

En el siguiente enlace se muestra un video con algunos de los experimentos realizados `https://youtu.be/U2scZhX6OP8?si=DDoSOkp3E4KNrW0-`.

Para el control lateral del AutoMINY también se realizaron experimentos utilizando el método de Control Predictivo Basado en Modelo o MPC, el cual es uno de los métodos más empleados en la literatura (Artuñedo et al. (2024); Rokonuzzaman et al. (2021); Chen et al. (2020)). Para las mismas cinco velocidades de desplazamiento se hicieron seis rondas experimentales en las que se varió la pose inicial del vehículo de manera que $e_{y0} \in [-12{,}7, 16{,}1]$ cm y $e_{\psi 0} \in [-13{,}2°, 24{,}1°]$. El método MPC se implementó con un periodo de muestreo de 0,0345 s, un horizonte de predicción de 10 pasos y un horizonte de control de 1 paso. Las matrices de ponderación del funcional de costo se definieron como $Q = \text{diag}(2, 0{,}0001, 2, 0{,}0001)$ y $R = 1{,}5$.

En la Tabla 2, en los renglones correspondientes al **MPC**, también se muestran las métricas de RMSE e_y, máx(e_y) y GEC. Como puede observarse, en casi todos los casos el controlador LQR presenta mejores resultados que el MPC, con valores menores de RMSE, menor error lateral máximo y menor GEC. Esto sugiere que el modelo matemático propuesto en la ecuación (7) y la ley de control (9) ofrecen un mejor desempeño en el control lateral del AutoMINY que el método MPC. También, el enfoque propuesto es más sencillo y de implementación más directa que el método MPC, ya que requiere menos parámetros y estos pueden ser medidos o estimados con mayor facilidad en condiciones reales.

4.3. *Evasión de obstáculos*

El arreglo experimental usado para evaluar el desempeño de la maniobra de adelantamiento y de evasión de obstáculos estáticos se muestra en la Figura 15 (arriba). Como obstáculo se utilizó un automóvil estático de 20×40 *cm* , circulando por el carril derecho desde una pose inicial arbitraria, al detectar el auto estacionado el AutoMINY se cambiaba al carril izquierdo para evadir este obstáculo y después se reincorporaba al carril derecho. En cada experimento se varió la pose inicial del auto así como la velocidad a la que se conducía. En total se hicieron cincuenta experimentos a cinco velocidades diferentes y cabe resaltar que en ninguno hubo una colisión; más aún, la separación mínima entre los centros del AutoMINY y el obstáculo se mantuvo entre 15,0 y 28,9 *cm*. Las gráficas de la Figura 15 (abajo) muestran la trayectoria del AutoMINY en algunos de estos experimentos mientras que en el siguiente enlace se muestra un video de algunas de estas pruebas `https://youtu.be/ds7bb0453BQ?si=5XRnDxllgc89EHu8`.

4.4. *Estacionamiento*

Respecto al estacionamiento en paralelo, se llevaron a cabo catorce experimentos en los que se realizó la maniobra para estacionarse en un espacio de 64×32 *cm*. En cada experimento se varió la pose inicial del auto y se condujo a una velocidad constante de 36,4 cm/s. Al final de cada experimento se midieron el error lateral e_{yf} y la orientación $e_{\psi f}$ del auto relativos al espacio designado para estacionarse. Se obtuvo que dichas variables permanecieron en los siguientes intervalos: $e_{yf} \in [1,6]$ *cm* y $e_{\psi f} \in [1{,}9, 7{,}1]°$. La Figura 16 muestra la trayectoria del coche en algunos de estos experimentos.

De manera similar, para la maniobra de estacionamiento en batería se llevaron a cabo quince experimentos para estacionarse en un espacio de 64×49 *cm*. En cada experimento se varió la pose inicial del auto y se mantuvo una velocidad constante de 36,4 cm/s. El error lateral e_{xf} relativo al espacio designado se mantuvo en el intervalo $e_{xf} \in [-5, 2]$ *cm*; mientras que el error de orientación medido fue de $e_{\psi f} \in [0{,}0, 5{,}2]°$. La Figura 17 muestra la trayectoria del auto obtenida de algunos de estos experimentos; mientras que el video del siguiente enlace ilustra las pruebas realizadas tanto en el estacionamiento en paralelo como en batería `https://youtu.be/zcEgX3Eu7hM?si=SKUnG2rCmL13o9Qw`.

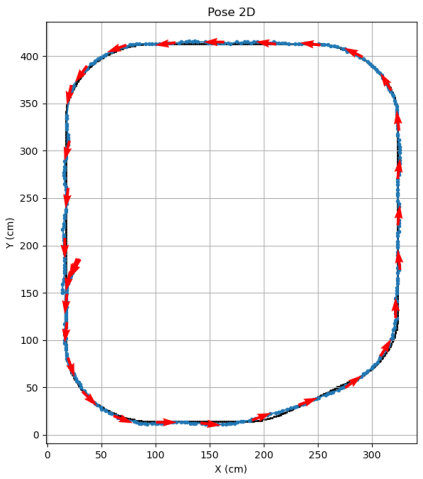

Figura 14: Pose del AutoMINY mientras recorre la pista a 82,7 cm/s. Los puntos azules representan su posición, las flechas rojas su orientación y los puntos negros es la trayectoria que se desea seguir.

Tabla 2: Comparación entre el método de control LQR y el método MPC. Para cada velocidad de desplazamiento se muestran en negritas los valores menores de RMSE e_y, máx(e_y) y GEC. Cada una de estas métricas es el promedio de las seis rondas experimentales.

Método	**Velocidad** (cm/s)	**RMSE** e_y	máx(e_y)	**GEC**
LQR	30.7	**2.03**	**6.0**	**32.6**
	44.0	**2.53**	**11.2**	21.6
	56.6	3.43	**15.2**	**15.9**
	68.5	**2.73**	**9.0**	**15.2**
	82.7	**2.75**	**8.0**	**14.5**
MPC	30.7	5.02	11.5	35.6
	44.0	2.90	13.1	**19.6**
	56.6	**2.48**	16.0	27.6
	68.5	3.43	16.2	55.4
	82.7	3.64	12.7	116.0

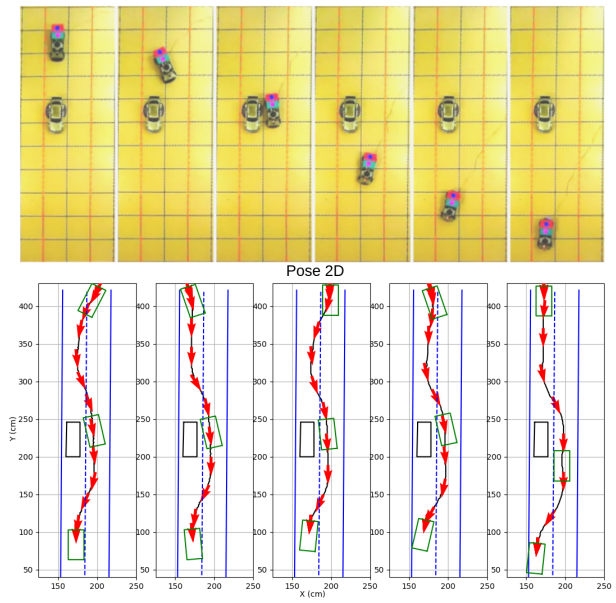

Figura 15: Arriba, arreglo experimental usado en la evasión de obstáculos. Abajo, pose del AutoMINY mientras realiza la maniobra de evasión de obstáculos a diferentes velocidades. De izquierda a derecha la velocidad es de: 30,7 , 44,0, 56,6, 68,5 y 82,7 cm/s. El rectángulo verde es el área que abarca el AutoMINY; mientras que el rectángulo negro, es el espacio que ocupa el obstáculo.

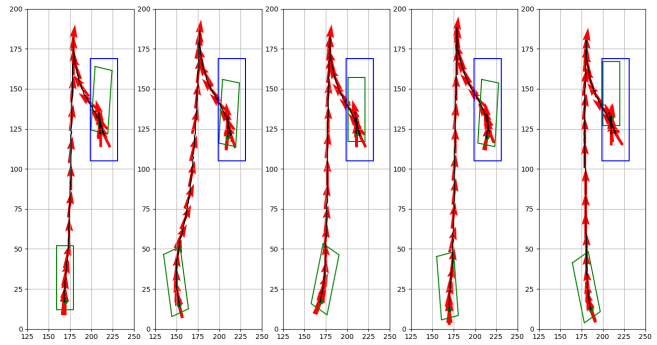

Figura 16: Pose del automóvil mientras realiza la maniobra de estacionamiento en paralelo desde diferentes poses iniciales. El rectángulo verde es el área que abarca el AutoMINY; mientras que el rectángulo azul es el espacio disponible para estacionarse.

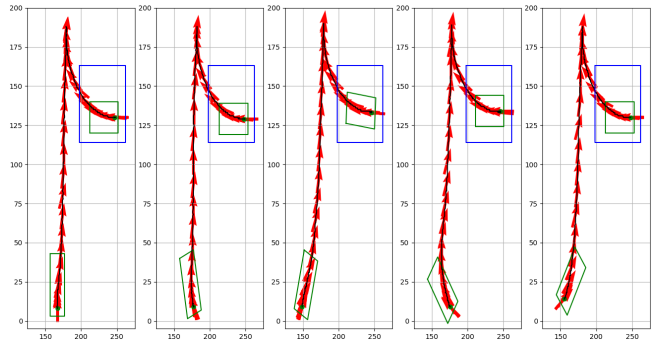

Figura 17: Pose del automóvil mientras realiza la maniobra de estacionamiento en batería desde diferentes poses iniciales. El rectángulo verde es el área que abarca el AutoMINY; mientras que el rectángulo azul es el espacio disponible para estacionarse.

4.5. Selector de maniobras

Para probar al selector de maniobras, a la pista ilustrada en la Figura 13 se le añadieron las señales de tránsito de: no estacionarse, semáforo en verde, semáforo en rojo y los límites de velocidad de 50 km/h y 100 km/h; así como un par de peatones y un automóvil estático. Entonces se condujo al AutoMINY con el SCA diseñado y se observó su comportamiento en cada situación. El video del siguiente enlace muestra este experimento `https://youtu.be/j18L-LGWZRA?si=X6Sb5hE2oVx1Kqc7`. De este experimento se puede destacar lo siguiente:

- Aunque en los alrededores de la pista se detecten peatones, el AutoMINY sólo se detiene cuando los detecta dentro de su carril y se mantiene detenido hasta que salen fuera del mismo.
- Al detectar el límite de 50 km/h el automóvil se conduce a 44,0 cm/s; mientras que al detectar el límite de 100 km/h su velocidad se incrementa a 82,7 cm/s.
- Al detectar a un automóvil estacionado en el camino, el AutoMINY inicia la maniobra de evasión. Entonces disminuye su velocidad a 44,0 cm/s independientemente del límite de velocidad antes detectado. Al finalizar la maniobra la velocidad de avance regresa a su valor anterior.
- Para probar el SCA ante múltiples estímulos se combinaron tres elementos: un automóvil estacionado en el camino, un peatón en el carril usado para rebasar y un semáforo en verde. Ante esta situación, el AutoMINY comienza la maniobra de rebase, luego se detiene enfrente del peatón a pesar de detectar un semáforo en verde y reanuda la maniobra de rebase hasta que el peatón se retira del camino.
- Al recibir una señal para estacionarse, el AutoMINY disminuye su velocidad a 36,4 cm/s y comienza a alinearse con el muro, mientras al mismo tiempo busca un espacio para estacionarse. Al detectar un espacio adecuado lleva a cabo la maniobra de estacionamiento en paralelo.

5. Conclusiones

En este trabajo se presentó el desarrollo de un sistema de conducción automática modular para el vehículo a escala 1:10 AutoMINY, con el cual fue capaz de circular de manera autónoma sobre una carretera respetando señales de tránsito, detenerse ante la presencia de peatones en su carril, esquivar obstáculos y ejecutar maniobras de estacionamiento en paralelo y en batería. Este sistema se dividió en tres módulos: el sistema de percepción, el selector de maniobras y los controladores de movimiento.

El sistema de percepción toma la información de los sensores a bordo y lleva a cabo tareas como: (*i*) detectar visualmente las líneas que delimitan el carril, (*ii*) identificar y localizar señales de tránsito u objetos de interés usando una red neuronal convolucional con la arquitectura YOLOv3, (*iii*) detectar obstáculos alrededor del vehículo y (*iv*) medir la orientación del automóvil.

La selección de la maniobra de conducción se realizó mediante una red neuronal *feed-forward* entrenada a partir de una tabla de verdad. En los experimentos en una carretera a escala, este selector de maniobras le permitió al AutoMINY conducir por una pista, detenerse ante la presencia de peatones en su carril, aumentar o disminuir su velocidad dependiendo de la señal de límite de velocidad, rebasar automóviles estacionados y estacionarse adecuadamente. Este enfoque permite interpretar de manera explícita la relación entre la información provista por el sistema de percepción y la maniobra seleccionada, al tiempo que ofrece la flexibilidad necesaria para reentrenar el modelo ante la incorporación de nuevos sensores o maniobras. En este sentido, el método propuesto constituye una alternativa frente a las máquinas de estados finitos y las arquitecturas *end-to-end* ampliamente usadas en la literatura.

Los controladores de movimiento se diseñaron para conducir por un carril (derecho o izquierdo), evadir obstáculos estáticos, estacionarse en paralelo o en batería y detenerse. Particularmente, para la tarea de seguimiento de carril se propuso un controlador visual de la posición lateral sintonizado mediante el método LQR, basado en un modelo cinemático propuesto. El desempeño de este controlador se evaluó mediante seis experimentos realizados a cinco velocidades distintas, considerando como métricas la raíz del error cuadrático medio del error lateral, el error lateral máximo y el gasto energético del controlador. Posteriormente, se repitieron los experimentos empleando un controlador predictivo basado en modelo (MPC) y

se compararon las métricas obtenidas. En términos generales, el controlador LQR presentó valores inferiores en todas las métricas evaluadas, lo que muestra un mejor desempeño. Además, el modelo matemático propuesto requiere un menor número de parámetros, los cuales resultan más fáciles de medir o estimar que aquellos necesarios para la implementación del MPC; lo que favorece su aplicación en escenarios reales.

La evasión de obstáculos se llevó a cabo utilizando el mismo esquema de control, complementado con un algoritmo que modifica dinámicamente las consignas para conducir por el carril derecho o izquierdo. En todos los experimentos realizados no se registraron colisiones con los obstáculos y se mantuvo una distancia entre los centros del AutoMINY y el obstáculo comprendida entre 15.0 y 28.9 cm, lo que confirma la efectividad de la estrategia propuesta.

En cuanto a las maniobras de estacionamiento, el desempeño de los controladores de movimiento se evaluó a partir de los errores lateral y de orientación con respecto a la posición final deseada del vehículo. En el caso del estacionamiento en paralelo, dichos errores se mantuvieron en los intervalos $e_{yf} \in [1,6]$ cm y $e_{\psi f} \in [1{,}9, 7{,}1]°$; mientras que para el estacionamiento en batería se obtuvieron valores de $e_{xf} \in [-5,2]$ cm y $e_{\psi f} \in [0{,}0, 5{,}2]°$. Si bien estos resultados confirman la viabilidad de las estrategias adoptadas, también se observa un margen considerable de mejora.

A pesar del buen desempeño del sistema de conducción automática desarrollado, su implementación está limitada a entornos estáticos. Al mismo tiempo, se navega en una carretera de solo dos carriles y no se aborda explícitamente el problema del seguimiento de rutas. Como trabajo futuro, se propone la incorporación de un planeador de trayectorias que permita al vehículo desplazarse en un entorno conocido desde un punto inicial hasta un destino final; considerando la presencia de otros vehículos en movimiento dentro de un escenario a escala. También se considera ampliar el conjunto de maniobras de conducción para añadir rebases y cambios de carril, lo que acercaría el sistema a situaciones de conducción más realistas.

En lo que respecta al reconocimiento de objetos y señales de tránsito, en este trabajo se empleó una red con arquitectura YOLOv3 debido a las limitaciones de hardware de las plataformas experimentales. El uso de computadoras más modernas permitiría explorar arquitecturas de detección más precisas y robustas; así como integrar múltiples cámaras a bordo del vehículo para incrementar el campo de visión.

Agradecimientos

Oscar González Miranda agradece al SECIHTI por su apoyo durante sus estudios en el programa de doctorado del CINVESTAV; así como al fondo FORDECYT/07SE/2018/08/06-04 otorgado al proyecto SECIHTI 296737 denominado “CONSORCIO EN INTELIGENCIA ARTIFICIAL”.

Referencias

Artuñedo, A., Moreno-Gonzalez, M., Villagra, J., 2024. Lateral control for autonomous vehicles: A comparative evaluation. Annual reviews in control 57, 100910.

Bächle, J., Häringer, J., Köhler, N., Özer, K.-K., Enzweiler, M., Marchthaler, R., 2024. Competing with autonomous model vehicles: a software stack for driving in smart city environments. Autonomous Intelligent Systems 4 (1), 17.

Badue, C., Guidolini, R., Carneiro, R. V., Azevedo, P., Cardoso, V. B., Forechi, A., Jesus, L., Berriel, R., Paixao, T. M., Mutz, F., et al., 2021. Self-driving cars: A survey. Expert systems with applications 165, 113816.

Bansal, M., Krizhevsky, A., Ogale, A., 2018. Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. arXiv preprint arXiv:1812.03079.

Berlin, F. U., 2020. Autominy. https://autominy.github.io/AutoMiny/.

Bjelonic, M., 2018. YOLO ROS: Real-time object detection for ROS. https://github.com/leggedrobotics/darknet_ros.

Chen, S.-p., Xiong, G.-m., Chen, H.-y., Negrut, D., 2020. Mpc-based path tracking with pid speed control for high-speed autonomous vehicles considering time-optimal travel. Journal of Central South University 27 (12), 3702–3720.

Committee, S. O.-R. A. V. S., et al., 2014. Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems. SAE Standard J 3016, 1.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., Zisserman, A., 2012. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

Hawke, J., Shen, R., Gurau, C., Sharma, S., Reda, D., Nikolov, N., Mazur, P., Micklethwaite, S., Griffiths, N., Shah, A., et al., 2020. Urban driving with conditional imitation learning. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). IEEE, pp. 251–257.

Ibarra-Zannatha, J. M., Miranda, O. G., Ramírez, C. B., Miranda, L. A. L., Aguilar, S. R. A., Osuna, L. Á. D., 2022. Integration of perception, planning and control in the autominy 4.0. In: 2022 19th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE). IEEE, pp. 1–6.

Kilyen, N. A., Lemnariu, R. F., Mois, G. D., Chen, Y., Morris, B. T., Muntean, I., 2021. The ieee itss and bosch future mobility challenge: A hands-on start to autonomous driving [technical activities]. IEEE Intelligent Transportation Systems Magazine 13 (3), 276–282.

Levinson, J., Askeland, J., Becker, J., Dolson, J., Held, D., Kammel, S., Kolter, J. Z., Langer, D., Pink, O., Pratt, V., et al., 2011. Towards fully autonomous driving: Systems and algorithms. In: 2011 IEEE intelligent vehicles symposium (IV). IEEE, pp. 163–168.

Li, D., Auerbach, P., Okhrin, O., 2024. Towards autonomous driving with small-scale cars: A survey of recent development. arXiv preprint arXiv:2404.06229.

Liu, S., Li, L., Tang, J., Wu, S., Gaudiot, J.-L., 2018. Creating autonomous vehicle systems. Springer.

Nolte, M., Form, T., Ernst, S., Graubohm, R., Maurer, M., 2018. The carolo-cup student competition: Involving students with automated driving. In: 2018 12th European Workshop on Microelectronics Education (EWME). IEEE, pp. 95–99.

Papafotiou, T., Tsardoulias, E., Nikolaou, A., Papagiannitsi, A., Christodoulou, D., Gkountras, I., Symeonidis, A. L., 2025. How to win bosch future mobility challenge: Design and implementation of the vroom autonomous scaled vehicle. Machines 13 (6), 514.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825–2830.

Redmon, J., Divvala, S., Girshick, R., Farhadi, A., 2016. You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 779–788.

Redmon, J., Farhadi, A., 2018. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.

Rokonuzzaman, M., Mohajer, N., Nahavandi, S., Mohamed, S., 2021. Model predictive control with learned vehicle dynamics for autonomous vehicle path tracking. IEEE Access 9, 128233–128249.

Tzutalin, 2015. Label studio is a modern, multi-modal data annotation tool. https://github.com/HumanSignal/labelImg.

Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M., Dolan, J., Duggins, D., Galatali, T., Geyer, C., et al., 2008. Autonomous driving in urban environments: Boss and the urban challenge. Journal of field Robotics 25 (8), 425–466.

VDI, 2025. Vdi autonomous driving challenge. Accessed on Agugost 5, 2025. URL: https://www.vdi-adc.de/

Wang, C.-Y., Bochkovskiy, A., Liao, H.-Y. M., 2022. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. URL: https://arxiv.org/abs/2207.02696 DOI: 10.48550/ARXIV.2207.02696