


Generación de trayectorias y control no lineal aplicado a vehículos móviles con ruedas empleando Webots y vehículo autónomo Autominy

Trajectory generation and nonlinear control applied to wheeled mobile robots using Webots and the Autominy autonomous vehicle

J. A. Rodríguez ^a, V. D. Cruz ^a, R. Miranda-Colorado ^{b,*}, L. T. Aguilar ^a

^aInstituto Politécnico Nacional-CITEDI, Av. Instituto Politécnico Nacional No. 1310, Nueva Tijuana, Tijuana, Baja California, México, 22435.

^bCONACyT-Instituto Politécnico Nacional-CITEDI, Av. Instituto Politécnico Nacional No. 1310, Nueva Tijuana, Tijuana, Baja California, México, 22435.

Resumen

Este artículo presenta una metodología para la generación y seguimiento de trayectorias empleando Redes Neuronales Artificiales (RNA) aplicadas en robots móviles con llantas utilizando técnicas de control no lineales. Para la generación de trayectorias se utilizaron dos estructuras de RNA que permiten analizar el entorno donde opera el vehículo, segmentarlo y generar una trayectoria empleando un sistema de visión. Por otro lado, se estudian e implementan dos metodologías de control no lineales tanto en simulaciones numéricas como en una plataforma experimental. La primera ley de control emplea dos estados adicionales para generar un algoritmo de realimentación dinámica. Por su parte, el segundo esquema de control es un algoritmo Proporcional Integral Derivativo (PID) combinado con un observador de perturbaciones. Para verificar la efectividad de la metodología de diseño de trayectorias, así como de los algoritmos de control, se emplea el simulador Webots y, como plataforma experimental, el vehículo autónomo Autominy. Las simulaciones numéricas y los resultados experimentales demuestran la efectividad de las metodologías desarrolladas.

Palabras Clave: Vehículo con ruedas, generación de trayectorias de referencia, control basado en observador, control no lineal.

Abstract

This paper develops a methodology for designing a reference trajectory and trajectory tracking utilizing Artificial Neural Networks (ANNs) applied to Wheeled Mobile Robots (WMR) through nonlinear control methodologies. For generating the reference trajectories, two ANNs architectures were utilized. These ANNs allow analyzing the environment where the WMR operates, segmenting it, and generating a trajectory using a vision system. Also, two control schemes are studied and implemented via numerical simulations and with a real-time experimental platform. The first controller employs two additional states for generating a dynamic feedback algorithm. The second control methodology uses an observer-based Proportional Integral Derivative (PID) controller. The Webots simulator and the experimental platform Autominy are used to analyze the effectiveness of the trajectory generation procedure, as well as that of the control algorithms. The numerical simulations and experimental results show the effectiveness and efficiency of the proposed methodologies.

Keywords: Wheeled mobile robot, reference trajectory generation, observer-based control, nonlinear control.

1. Introducción

Los vehículos autónomos son sistemas que tienen distintas aplicaciones, que van desde la exploración de entornos desconocidos, conducción autónoma, hasta el despliegue de vehículos en el espacio (Lu *et al.* (2022); Kramer *et al.* (2007)). El seguimiento de trayectorias en vehículos móviles con llantas

es un problema con diversas aplicaciones prácticas. Por ello, se han desarrollado distintas investigaciones que afrontan esta problemática considerando perturbaciones en el entorno como deslizamientos en las llantas (Miranda-Colorado (2022), Yoo (2013)). En Liu *et al.* (2022) implementan un controlador robusto para el seguimiento de trayectorias utilizando métodos

*Autor para correspondencia: rmirandaco@conacyt.mx, rmirandaco@gmail.com

Correo electrónico: jrodriguez@citedi.mx (Jesus A. Rodriguez), vcruz@citedi.mx (Victor D. Cruz), rmirandaco@conacyt.mx (Roger Miranda-Colorado), lagui-larb@ipn.mx (Luis T. Aguilar).

inteligentes para compensar los efectos de parámetros desconocidos en las señales de control. Por otro lado, en Luca *et al.* (1998) desarrollan controles por realimentación dinámica, mientras que en Rosas-Vilchis *et al.* (2020) emplean el método *backstepping* combinado con modos deslizantes. En Lee *et al.* (2013) desarrollan métodos de seguimiento aplicados a vehículos con ruedas. Sin embargo, para que el vehículo autónomo pueda realizar una tarea de seguimiento de modo apropiado, no solo basta con tener un algoritmo de control robusto, sino que también se requiere una metodología eficiente para generar la trayectoria de referencia, misma que debe satisfacer ciertas condiciones para ser factible (Luca *et al.* (1998)).

Diversos algoritmos inteligentes se han empleado también en vehículos con ruedas. Por ejemplo, en Alomari *et al.* (2021) reportan una técnica para el control de un vehículo autónomo por medio de redes neuronales, empleando la plataforma Gazebo como entorno de simulación para evaluar el desempeño del algoritmo de seguimiento de trayectoria. Por su parte, en Hyung *et al.* (2013) aplican un algoritmo adaptable con la finalidad de resolver el problema de navegación autónoma buscando que dos vehículos sean capaces de seguirse entre sí. Además, en Davy *et al.* (2018) desarrollan metodologías de detección de líneas de carril utilizando segmentación de instancias.

Del estudio bibliográfico realizado se observa que muchos de los trabajos reportados no consideran el efecto de las perturbaciones para el diseño de algoritmos de seguimiento de trayectorias. Además, es importante contar con una trayectoria de referencia propiamente diseñada, de modo que sea factible para que el vehículo pueda seguirla apropiadamente. Con base en lo anterior, es importante contar con metodologías eficientes para generar trayectorias de referencia, así como algoritmos de control para seguimiento de trayectorias que tomen en cuenta el efecto de las perturbaciones que afectan al sistema. Aunado a lo anterior, es importante considerar que hay diversas herramientas que pueden emplearse en entornos de simulación y en plataformas experimentales, las cuales simplifican el proceso de control del vehículo de forma autónoma.

Con base en lo anterior, las contribuciones principales de este trabajo son:

- Presentar una metodología para el diseño de trayectorias de referencia para un vehículo con ruedas empleando la cámara a bordo del vehículo. Ésta metodología emplea algoritmos inteligentes previamente reportados en la literatura para la detección y segmentación de un carril.
- Mostrar el procedimiento de implementación de dos esquemas de control tomados de la literatura, los cuales permiten al vehículo seguir la trayectoria de referencia a pesar de las perturbaciones que le afectan.

La metodología de generación de trayectorias se basa en dos algoritmos inteligentes que emplean Redes Neuronales Artificiales (RNAs). Por su parte, los algoritmos de control estudiados consisten en un controlador con realimentación dinámica y un control Proporcional Integral Derivativo (PID) combinado con un observador de perturbaciones. El desempeño de la metodología de diseño de trayectorias y de los algoritmos de control se muestra por medio de simulaciones numéricas y experimentos. Las simulaciones se efectúan por medio de la herramienta

Webots, mientras que los experimentos se llevan a cabo con un vehículo autónomo a escala que reproduce el funcionamiento de un vehículo autónomo real. Con base en los resultados obtenidos se verifica que el desempeño de las metodologías estudiadas es satisfactorio, por lo que puede considerarse un enfoque útil para el control de vehículos tipo auto.

De acuerdo al estudio bibliográfico realizado, no existe otro trabajo en la literatura que haya reportado una metodología de implementación de controladores, así como de generación de trayectorias, con resultados numéricos y experimentales en un vehículo de cuatro ruedas. Por lo tanto, el estudio realizado en este documento contribuye al estado del arte para el control de vehículos con ruedas de forma autónoma.

Notación: En este trabajo se emplea \mathbb{R}, \mathbb{R}^+ para representar los números reales y los números reales positivos. Además, $|\cdot|$ denota el valor absoluto de su argumento, mientras que $\|\cdot\|$ representa la norma Euclidiana de su argumento. Los vectores se representan con letras en negritas, mientras que las matrices se describen por medio de letras mayúsculas. Dado el vector \mathbf{v} , su traspuesta se representa como \mathbf{v}^T . Finalmente, las notaciones $s_\theta, c_\theta, t_\theta$ corresponden a $\sin(\theta), \cos(\theta), \tan(\theta)$, respectivamente.

2. Generación de Trayectorias Empleando Métodos Inteligentes

En esta Sección se describe una metodología que permite generar trayectorias de referencia para el vehículo autónomo. El procedimiento se basa en el uso de algoritmos inteligentes. En específico, se utilizan los algoritmos inteligentes *HybridNets* (Dat *et al.* (2022)) y *Ultra Fast Structure-aware Deep Lane Detection* (Zequin *et al.* (2020)), los cuales permiten llevar a cabo detección y segmentación de un carril. A partir de estos algoritmos, se muestra el procedimiento que genera una trayectoria que el vehículo puede seguir.

2.1. HybridNets

El algoritmo *HybridNets* emplea una red neuronal de tipo *end-to-end* con PyTorch, que le permite obtener una respuesta multitarea. Cuenta con un codificador y dos decodificadores para resolver dos diferentes problemas, la detección y segmentación de carril, así como la detección de vehículos en el camino. La estructura general de la red se muestra en la Figura 1. Primero se tiene una imagen de entrada. Para su procesamiento se emplean dos redes: la red *EfficientNet-B3* y la red *EfficientDet*. La primera red lleva a cabo la extracción de características, mientras que la segunda utiliza la mezcla de características en diferentes resoluciones con un módulo BiFPN (*Bi-directional Feature Pyramid Network*). Como resultado, el algoritmo *HybridNets* genera una salida que detecta los vehículos presentes en la imagen y otra salida muestra el resultado de la segmentación. Ésta segunda salida es la que permite detectar las líneas del carril, así como el área disponible para conducir.

Para entrenar el algoritmo *HybridNets* se emplea el conjunto de datos BDD100K (Fisher *et al.* (2020)), el cual cuenta con cien mil videos recopilados en alrededor de 50 mil viajes, con una duración de 40 segundos cada uno. Este conjunto de datos fue desarrollado con la finalidad de resolver el problema de detección de objetos, detección de líneas y segmentación semántica, entre otros.

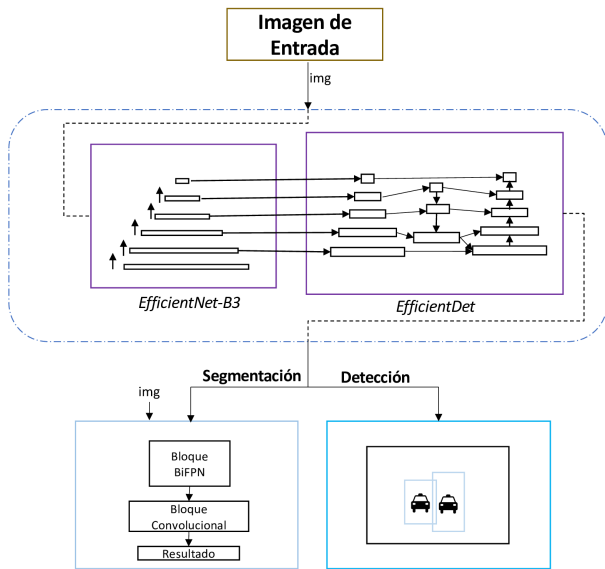


Figura 1: Diagrama de arquitectura de método *HybridNets* [Dat et al. (2022)].

2.2. Ultra Fast Structure-aware Deep Lane Detection

El algoritmo *Ultra Fast Structure-aware Deep Lane Detection* consiste en una red neuronal desarrollada por medio de PyTorch, con la finalidad de obtener una solución a la problemática de detección de carril utilizando la técnica de segmentación de píxeles. El algoritmo se conforma por dos ramas, una principal y una auxiliar. La primera rama se encarga de obtener las líneas de carril en la imagen, mientras que la rama auxiliar se utiliza únicamente en el entrenamiento de la red neuronal para generar las tareas de segmentación. El esquema general de la arquitectura para la red neuronal se muestra en la Figura 2.

El conjunto de datos utilizado para el entrenamiento del algoritmo es *TuSimple* (TuSimple (2022)), el cual consiste en 3626 video clips con duración de un segundo con veinte cuadros. Cada clip consiste en caminos de más de dos carriles en condiciones de tráfico variable, diferentes condiciones de iluminación, entre otros.

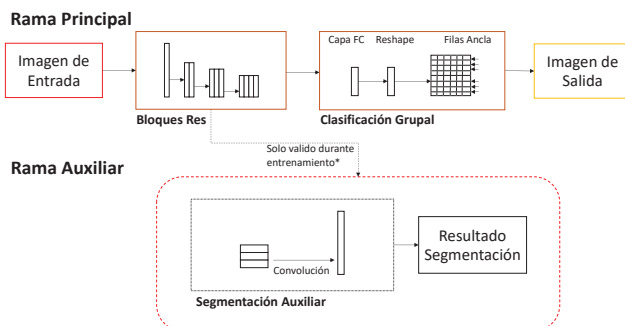


Figura 2: Diagrama de arquitectura de método *Ultra Fast Structure-aware Deep Lane Detection* (Zequin et al. (2020)).

2.3. Generación de Trayectoria

Los algoritmos anteriores se emplean para llevar a cabo la generación de trayectorias para el vehículo autónomo. El pro-

cedimiento propuesto para obtener la trayectoria de referencia del vehículo consiste en los siguientes pasos:

1. Primero, empleando la cámara que lleva integrada el vehículo, se adquiere una imagen. Esta imagen es procesada por alguno de los dos algoritmos descritos previamente.
2. Empleando las propiedades de cada algoritmo, se lleva a cabo la detección de carril y la segmentación del área disponible para conducción. Posteriormente, se procede a detectar el área disponible para conducir, que se usa para generar la trayectoria de referencia. Como resultado, cada algoritmo (*HybridNets* y *Ultra Fast Structure-aware Deep Lane Detection*) proporciona una imagen con líneas de carril y el área segmentada para conducir.
3. Con la imagen generada por alguno de los algoritmos inteligentes se realiza una transformación de perspectiva a vista de ojo de ave. El objetivo aquí es delimitar el área visible por la cámara del vehículo, para después seleccionar una región de interés enfocada en el carril frontal actual. Además, se determina el área máxima disponible para conducir, la cual se obtiene a partir del área máxima segmentada por el método inteligente en la región de interés seleccionada.
4. Sobre la región anterior se determinan cinco puntos de referencia, los cuales sirven para generar la trayectoria de referencia, como se muestra en la Fig. 3. Tres de estos puntos son variables con respecto al *frame* que adquiere la cámara, mientras que los primeros dos puntos son fijos, con la finalidad de servir de soporte a la trayectoria.
5. Posteriormente, los cinco puntos de referencia (representados en el plano imagen con transformación de perspectiva) se cambian al plano imagen original, obteniendo las coordenadas en píxel de su posición original, como se muestra en la Fig. 3.
6. Luego se genera la trayectoria de referencia con los coeficientes generados por los puntos de referencia, con lo que se representa la trayectoria como una poli-línea en el área segmentada del carril para conducir.
7. Finalmente, se obtiene la ecuación de la trayectoria obtenida.

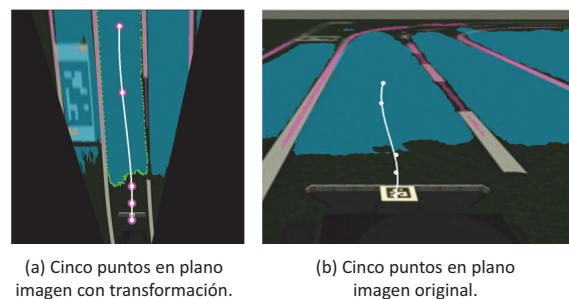


Figura 3: Diagrama mostrando los cinco puntos de referencia, en el plano imagen con transformación y en el plano imagen original, que sirven para generar la trayectoria de referencia.

En la Figura 4, se puede observar un diagrama general de los pasos para la generación de la trayectoria.

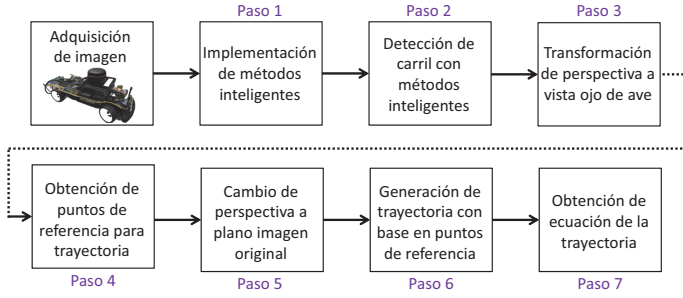


Figura 4: Diagrama general para el método propuesto de generación de trayectorias.

Posteriormente, en la Sección de simulaciones numéricas y experimentos, se muestran los resultados que se obtienen al aplicar la metodología de generación de trayectorias anterior, empleando los algoritmos inteligentes *HybridNets* y *Ultra Fast Structure-aware Deep Lane Detection*.

3. Modelo Cinemático

Este trabajo presenta una metodología para generar trayectorias de referencia en un vehículo autónomo, así como la implementación de diversos algoritmos de control en una plataforma experimental. Para ello, primero debe considerarse el modelo cinemático del vehículo, por medio del cual se diseñan los controladores.

En la Fig. 5 se muestra un diagrama que describe un vehículo tipo auto. De modo general, el modelo cinemático de un vehículo tipo auto con perturbaciones cinemáticas se describe como (Luca *et al.* (1998); Yoo (2013):

$$\dot{\mathbf{q}}(t) = \mathbf{S}(\mathbf{q})\mathbf{v}(t) + \mathbf{d}(t), \quad (1)$$

$$\mathbf{S}(\mathbf{q}) = \begin{bmatrix} c(\theta) & 0 \\ s(\theta) & 0 \\ \frac{l}{c(\theta)} & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{v}(t) = \begin{bmatrix} v_1(t) \\ v_2(t) \end{bmatrix},$$

$$\mathbf{d}(t) = \begin{bmatrix} d_1(t) & d_2(t) & d_3(t) & d_4(t) \end{bmatrix}^T,$$

donde $\mathbf{q}^T(t) = [x, y, \theta, \phi] \in \mathbb{R}^4$ es el vector de coordenadas generalizadas del vehículo, $\dot{\mathbf{q}}(t) \in \mathbb{R}^4$ es la derivada con respecto al tiempo de $\mathbf{q}(t)$. La posición del vehículo se determina por medio de las coordenadas (x, y) que se ubican en el punto medio del eje anterior, como se muestra en la Fig. 5; l corresponde a la distancia entre el eje anterior y posterior del vehículo; $\theta(t)$ denota la orientación con respecto al eje x del vehículo, mientras que el ángulo de orientación de las llantas frontales está representado por $\phi(t)$. Se emplea el vector $\mathbf{d}(t)$ para englobar las perturbaciones cinemáticas que afectan al vehículo (Miranda-Colorado (2022)). Además, las entradas de control del sistema son $v_1(t)$ y $v_2(t)$, que corresponden a la velocidad lineal y angular del vehículo.

El vehículo autónomo es un sistema mecatrónico. Debido a ello, puede suponerse que posee restricciones mecánicas, así como límites superiores e inferiores en los actuadores. Por lo tanto, es factible suponer que existen constantes $V_i, \varphi \in \mathbb{R}^+$ tales que se satisface $|v_i(t)| \leq V_i, |\phi(t)| \leq \varphi < \pi/2$ con $i \in 1, 2$. Además, se supone que las perturbaciones cinemáticas, junto

con su primera derivada temporal, son acotadas, es decir, existen constantes $D_j, \bar{D}_j \in \mathbb{R}^+$ tales que $|d_j(t)| \leq D_j, |\dot{d}_j(t)| \leq \bar{D}_j, j \in 1, 2, 3, 4$ [Miranda-Colorado (2022)].

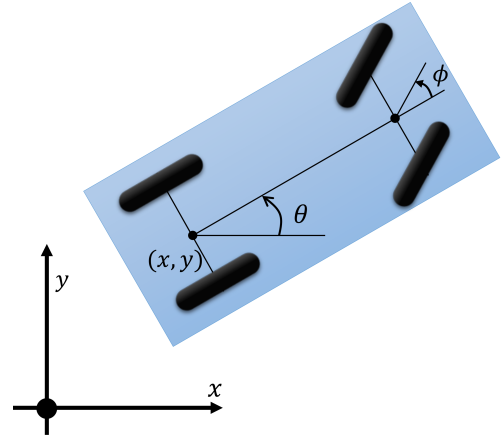


Figura 5: Diagrama general mostrando las coordenadas de posición $(x(t), y(t))$ y orientación $(\theta(t), \phi(t))$ de un vehículo autónomo tipo auto.

4. Descripción de Controladores

Para el control del vehículo autónomo se emplearon los controladores propuestos en Luca *et al.* (1998) y Miranda-Colorado (2022), los cuales están basados en el modelo cinemático (1). Independientemente del algoritmo de control, el objetivo es hacer que el vehículo siga una trayectoria de referencia de modo que el error de seguimiento converja a cero asintóticamente. Por lo tanto, el objetivo de control puede plantearse de modo matemático como se indica a continuación:

Objetivo de control: Sea el modelo cinemático de un vehículo tipo auto (1), el cual se encuentra afectado por perturbaciones. Entonces, diseñar las entradas de control $v_1(t)$ y $v_2(t)$ de modo que se satisfagan las siguientes relaciones:

$$\lim_{t \rightarrow \infty} |\tilde{x}(t)| = 0, \quad \lim_{t \rightarrow \infty} |\tilde{y}(t)| = 0. \quad (2)$$

Para implementar los controladores que se describen a continuación se utiliza un modelo cinemático alternativo en lugar del descrito en (1). Para ello se emplea la siguiente transformación de coordenadas (Luca *et al.* (1998)):

$$x_1(t) = x, \quad x_2(t) = \frac{t(\phi)}{lc_\theta^3}, \quad x_3(t) = t_\theta, \quad x_4(t) = y, \quad (3)$$

$$u_1(t) = v_1 c_\theta, \quad u_2(t) = \frac{v_2}{lc_\phi^2 c_\theta^3} + 3 \frac{t_\phi^2 s_\theta}{l^2 c_\theta^4} v_1.$$

Al derivar con respecto al tiempo las nuevas coordenadas $x_i(t), i = 1, 2, 3, 4$, se obtiene el denominado *modelo cinemático en cadena*, descrito por:

$$\dot{x}_1(t) = u_1, \quad \dot{x}_2(t) = u_2, \quad \dot{x}_3(t) = x_2 u_1, \quad \dot{x}_4(t) = x_3 u_1. \quad (4)$$

Al diseñar un controlador, se emplea una trayectoria de referencia que debe ser factible, es decir, que cumpla las restricciones cinemáticas del vehículo, para asegurar que el auto pueda seguirla. Por lo tanto, la trayectoria de referencia debe generarse por medio de un conjunto de ecuaciones análogas al modelo

(4). De este modo, la descripción matemática del modelo cinemático de referencia es:

$$\dot{x}_{d1} = u_{d1}, \quad \dot{x}_{d2} = u_{d2}, \quad \dot{x}_{d3} = x_{d2}u_{d1}, \quad \dot{x}_{d4} = x_{d3}u_{d1}, \quad (5)$$

donde $x_{di} \in \mathbb{R}$ con $i \in 1, 2, 3, 4$ son las coordenadas de referencia y $u_j \in \mathbb{R}$ con $j \in 1, 2$ son las entradas de control deseadas.

Ahora se define la siguiente variable de salida:

$$\kappa(t) = \begin{bmatrix} \kappa_1 \\ \kappa_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_4 \end{bmatrix}. \quad (6)$$

Al derivar $\kappa(t)$ tres veces con respecto al tiempo se obtiene:

$$\ddot{\kappa} = \begin{bmatrix} 0 \\ 3x_2\gamma_1\gamma_2 \end{bmatrix} + \underbrace{\begin{bmatrix} 1 & 0 \\ x_3 & \gamma_1^2 \end{bmatrix}}_A \begin{bmatrix} \dot{\gamma}_2 \\ u_2 \end{bmatrix}, \quad (7)$$

donde $\gamma_1(t) = u_1(t)$, $\dot{\gamma}_1(t) = \gamma_2(t)$ son integradores.

La matriz A se denomina matriz de desacoplamiento. Con ella, es posible considerar una nueva variable de control $\mathbf{r}(t)$, de modo que el sistema (7) se transforma en el siguiente modelo linealizado:

$$\ddot{\mathbf{r}} = \mathbf{r}, \quad (8)$$

donde $\mathbf{r}(t) = [r_1(t), r_2(t)]^T$. Entonces, de la ecuación (7), se obtienen las siguientes relaciones:

$$\begin{aligned} u_1 &= \gamma_1, & u_2 &= (r_2 - x_1 r_1 - 3x_2\gamma_1\gamma_2)/\gamma_1^2, \\ \dot{\gamma}_1 &= \gamma_2, & \dot{\gamma}_2 &= r_1. \end{aligned} \quad (9)$$

Para el diseño de la primera ley de control se emplea la señal \mathbf{r} . Entonces, se utilizan las siguientes ecuaciones [Luca et al. (1998)]:

$$\begin{aligned} r_i &= \ddot{\kappa}_{d1} + k_{a1}(\dot{\kappa}_{d1} - \dot{\kappa}_1) + k_{v1}(\dot{\kappa}_{d1} - \dot{\kappa}_1) \\ &+ k_{p1}(\kappa_{d1} - \kappa_1), i = 1, 2. \end{aligned} \quad (10)$$

Definiendo la variable de error $\tilde{\kappa}(t) = \kappa_{di}(t) - \kappa_i(t)$, el sistema (8) en lazo cerrado con (10) se representa en el espacio de estados como:

$$\underbrace{\begin{bmatrix} \dot{\tilde{\kappa}}_i \\ \dot{\tilde{\kappa}}_i \\ \dot{\tilde{\kappa}}_i \end{bmatrix}}_{M_1} = \underbrace{\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -k_{p1} & -k_{v1} & -k_{a1} \end{bmatrix}}_{M_1} \begin{bmatrix} \tilde{\rho}_i \\ \tilde{\rho}_i \\ \tilde{\rho}_i \end{bmatrix}, i = 1, 2. \quad (11)$$

Los polinomios característicos de los sistemas descritos en (11) son:

$$p_i(\lambda) = \lambda^3 + k_{a1}\lambda^2 + k_{v1}\lambda + k_{p1}, \quad i = 1, 2. \quad (12)$$

Entonces, las ganancias del controlador k_{p1} , k_{v1} y k_{a1} , se calculan de modo que los valores propios de los polinomios característicos (12) se ubiquen en el semiplano complejo izquierdo. Lo anterior se logra si se satisfacen las siguientes condiciones:

$$k_{a1}k_{v1} > k_{p1} > 0, \quad (13)$$

lo cual, de acuerdo al criterio de Routh-Hurwitz, garantiza que el origen de (11) sea asintóticamente estable. En adelante, dado que el algoritmo de control (9)-(10) emplea realimentación

dinámica, será referido como algoritmo de *Control por Realimentación Dinámica (CRD)*.

Ahora se describe el segundo algoritmo de control de acuerdo a Miranda-Colorado (2022). Para ello, se define la nueva variable de salida:

$$\rho(t) = \begin{bmatrix} x + lc_\theta + \delta c_{\theta+\phi} \\ y + ls_\theta + \delta s_{\theta+\phi} \end{bmatrix}. \quad (14)$$

Empleando el modelo cinemático con perturbaciones (1), se obtienen las primeras dos derivadas temporales de (14):

$$\begin{aligned} \dot{\rho}(t) &= A(\theta, \phi)\mathbf{v}(t) + \Lambda(t), \\ \ddot{\rho}(t) &= A(\theta, \phi)\dot{\mathbf{v}}(t) + \bar{A}(\theta, \phi)\mathbf{v}(t) + \dot{\Lambda}(t), \end{aligned} \quad (15)$$

con las siguientes definiciones:

$$\begin{aligned} A(\theta, \phi) &= \begin{bmatrix} c_\theta - t_\phi(s_\theta + \frac{\delta}{l}s_{\theta+\phi}) & -\delta s_{\theta+\phi} \\ s_\theta + t_\phi(c_\theta + \frac{\delta}{l}c_{\theta+\phi}) & \delta c_{\theta+\phi} \end{bmatrix}, \\ \bar{A}(\theta, \phi) &= \begin{bmatrix} \dot{a}_{11} & \dot{a}_{12} \\ \dot{a}_{21} & \dot{a}_{22} \end{bmatrix} \mathbf{v}(t) = \begin{bmatrix} v_1(t) \\ v_2(t) \end{bmatrix}, \quad \dot{\mathbf{v}}(t) = \begin{bmatrix} \dot{v}_1(t) \\ \dot{v}_2(t) \end{bmatrix}, \\ \Lambda(t) &= \begin{bmatrix} -ls_\theta d_3 - \delta s_{\theta+\phi}(d_3 + d_4) + d_1 \\ lc_\theta + \delta c_{\theta+\phi}(d_3 + d_4) + d_2 \end{bmatrix}, \quad \dot{\Lambda}(t) = \begin{bmatrix} \dot{\Lambda}_1 \\ \dot{\Lambda}_2 \end{bmatrix}. \end{aligned}$$

Ahora se considera la nueva señal de referencia descrita como:

$$\rho_d(t) = \begin{bmatrix} x_d + lc_{\theta_d} + \delta c_{\theta_d+\phi_d} \\ y_d + ls_{\theta_d} + \delta s_{\theta_d+\phi_d} \end{bmatrix}. \quad (16)$$

La primera y segunda derivada con respecto al tiempo de $\rho_d(t)$ están dadas por:

$$\begin{aligned} \dot{\rho}_d(t) &= A(\theta_d, \phi_d)\mathbf{v}_d(t), \\ \ddot{\rho}_d(t) &= A(\theta_d, \phi_d)\dot{\mathbf{v}}_d(t) + \bar{A}(\theta_d, \phi_d)\mathbf{v}_d(t), \end{aligned} \quad (17)$$

donde:

$$\begin{aligned} A(\theta_d, \phi_d) &= \begin{bmatrix} c_{\theta_d} - t_{\phi_d}(s_{\theta_d} + \frac{\delta}{l}s_{\theta_d+\phi_d}) & -\delta s_{\theta_d+\phi_d} \\ s_{\theta_d} + t_{\phi_d}(c_{\theta_d} + \frac{\delta}{l}c_{\theta_d+\phi_d}) & \delta c_{\theta_d+\phi_d} \end{bmatrix}, \\ \bar{A}(\theta_d, \phi_d) &= \begin{bmatrix} \dot{a}_{d11} & \dot{a}_{d12} \\ \dot{a}_{d21} & \dot{a}_{d22} \end{bmatrix}, \mathbf{v}_d(t) = \begin{bmatrix} v_{d1}(t) \\ v_{d2}(t) \end{bmatrix}, \quad \dot{\mathbf{v}}_d(t) = \begin{bmatrix} \dot{v}_{d1}(t) \\ \dot{v}_{d2}(t) \end{bmatrix}. \end{aligned}$$

Empleando (14) y (16) se define la siguiente variable de error de seguimiento:

$$\tilde{\rho}(t) = \rho(t) - \rho_d(t). \quad (18)$$

Utilizando (15) y (17) se obtiene la siguiente dinámica del error:

$$\begin{aligned} \ddot{\tilde{\rho}}(t) &= A(\theta, \phi)\mathbf{v}(t) + \bar{A}(\theta, \phi)\mathbf{v}(t) + \dot{\Lambda}(t) \\ &- A(\theta_d, \phi_d)\mathbf{v}_d(t) - \bar{A}(\theta_d, \phi_d)\mathbf{v}_d(t). \end{aligned} \quad (19)$$

Ahora se definen las siguientes variables:

$$\begin{aligned} x_1(t) &= \dot{\tilde{\rho}}_1(t), & x_2(t) &= \dot{\tilde{\rho}}_1(t), \\ x_3(t) &= \dot{\tilde{\rho}}_2(t), & x_4(t) &= \dot{\tilde{\rho}}_2(t). \end{aligned} \quad (20)$$

Al derivar las variables dadas en (20) con respecto al tiempo, empleando (15) y (17), se obtienen los siguientes subsistemas de segundo orden:

$$\begin{aligned} \dot{x}_1(t) &= x_2(t), & \dot{x}_2(t) &= u_1(t) + \zeta_1(t), \\ \dot{x}_3(t) &= x_4(t), & \dot{x}_4(t) &= u_2(t) + \zeta_2(t), \end{aligned} \quad (21)$$

donde:

$$\begin{aligned} \mathbf{u}(t) &= [u_1(t), u_2(t)]^T = A(\theta, \phi)\dot{v}(t), \\ \zeta_1(t) &= \dot{a}_{11}v_1 + \dot{a}_{12}v_2 + \dot{\Lambda}_1 - a_{d11}\dot{v}_{d1} - a_{d12}\dot{v}_{d2} \\ &\quad - \dot{a}_{d11}v_{d1} - \dot{a}_{d12}v_{d2}, \\ \zeta_2(t) &= \dot{a}_{21}v_1 + \dot{a}_{22}v_2 + \dot{\Lambda}_2 - a_{d21}\dot{v}_{d1} - a_{d22}\dot{v}_{d2} \\ &\quad - \dot{a}_{d21}v_{d1} - \dot{a}_{d22}v_{d2}, \end{aligned} \quad (22)$$

En (22), $\zeta_i(t)$ son las perturbaciones del sistema y $u_i(t)$ son las entradas de control, con $i = 1, 2$. Para compensar el efecto de las perturbaciones se emplea un observador de perturbaciones cuya estructura está basada en (Park *et al.* (2019)). Las ecuaciones matemáticas del observador son:

$$\frac{d}{dt} \begin{bmatrix} \hat{e}_i \\ \hat{e}_i \\ z_{ci} \end{bmatrix} = \begin{bmatrix} k_i \tilde{e}_{i1} + \hat{e}_i \\ q_i \text{sign}(\tilde{e}_{i1}) \\ u_i \end{bmatrix}, \quad i = 1, 2, \quad (23)$$

donde:

$$\begin{aligned} \tilde{e}_1(t) &= x_2(t) - z_{c1}(t), & \tilde{e}_2(t) &= x_4(t) - z_{c2}(t), \\ \tilde{e}_{11}(t) &= e_1(t) - \hat{e}_1(t), & \tilde{e}_{21}(t) &= e_2(t) - \hat{e}_2(t), \\ \tilde{e}_{12}(t) &= \dot{e}_1(t) - \hat{\dot{e}}_1(t), & \tilde{e}_{22}(t) &= \dot{e}_2(t) - \hat{\dot{e}}_2(t). \end{aligned} \quad (24)$$

Para que el observador (23) sea capaz de estimar las perturbaciones se supone que se satisfacen las siguientes desigualdades:

$$\zeta_i(t) \leq \mathfrak{N}_i < q_i \quad i = 1, 2, \quad (25)$$

donde $\mathfrak{N}_i \in \mathbb{R}^+$ y $k \gg 0$. Bajo estas suposiciones, los errores $\tilde{e}_{i2}(t)$ convergen globalmente uniformemente asintóticamente a cero, independientemente de la condición inicial (Park *et al.* (2019)).

Para el diseño de las señales de control, se dividen las entradas $u_i(t)$ como sigue (Miranda-Colorado (2022)):

$$u_i(t) = u_{i1}(t) + u_{ib}(t), \quad i = 1, 2, \quad (26)$$

donde, utilizando los errores (24) y las señales del observador (23), se define $u_{ib}(t) = -\hat{e}_i(t)$ con $i \in 1, 2$, permitiendo estimar las perturbaciones $\zeta_i(t)$. Por lo tanto, los subsistemas (21) se reescriben de la siguiente manera:

$$\begin{aligned} \dot{x}_1(t) &= x_2(t), & \dot{x}_2(t) &= u_{1a}(t) + \tilde{e}_{12}(t), \\ \dot{x}_3(t) &= x_4(t), & \dot{x}_4(t) &= u_{2a}(t) + \tilde{e}_{22}(t). \end{aligned} \quad (27)$$

Para el diseño de la entrada de control $u_{ia}(t)$ se emplea la siguiente estructura (Miranda-Colorado (2022)):

$$\begin{aligned} u_{1a} &= k_{p1}x_1(t) + k_{d1}x_2(t) + k_{i1}\varkappa_1(t), & \varkappa_1 &= \int_0^T x_1(\tau)d\tau, \\ u_{2a} &= k_{p2}x_3(t) + k_{d2}x_4(t) + k_{i2}\varkappa_2(t), & \varkappa_2 &= \int_0^T x_3(\tau)d\tau. \end{aligned} \quad (28)$$

Por lo tanto, al sustituir (28) en (27), se obtiene el siguiente sistema en lazo cerrado:

$$\begin{aligned} \dot{x}_1(t) &= x_2(t), \\ \dot{x}_2(t) &= k_{p1}x_1(t) + k_{d1}x_2(t) + k_{i1}\varkappa_1(t) + \tilde{e}_{12}(t), \\ \dot{x}_3(t) &= x_4(t), \\ \dot{x}_4(t) &= k_{p2}x_3(t) + k_{d2}x_4(t) + k_{i2}\varkappa_2(t) + \tilde{e}_{22}(t). \end{aligned} \quad (29)$$

Para analizar la estabilidad del origen de (29) se emplea el enfoque de Lyapunov. Entonces, se consideran las funciones candidatas de Lyapunov:

$$\begin{aligned} V_1(x_1, x_2, \varkappa_1) &= \frac{1}{2}(k_{p1} + k_{i1})x_1^2 + \frac{1}{2}(k_{d1} + k_{21})x_2^2 + \frac{1}{2}(k_{i1} \\ &\quad + k_{31})\varkappa_1^2 + k_{41}x_1x_2 + k_{51}x_1\varrho_1 + k_{61}x_2\varrho_1, \\ V_2(x_1, x_2, \varkappa_2) &= \frac{1}{2}(k_{p2} + k_{i2})x_3^2 + \frac{1}{2}(k_{d2} + k_{22})x_4^2 + \frac{1}{2}(k_{i2} \\ &\quad + k_{32})\varkappa_2^2 + k_{42}x_3x_4 + k_{52}x_3\varrho_2 + k_{62}x_4\varrho_2, \end{aligned} \quad (30)$$

cuyas derivadas temporales a lo largo de las trayectorias de (29) son:

$$\begin{aligned} \dot{V}(x_1, x_2, \varkappa_1) &= -\eta_{11}x_1^2 - \eta_{12}x_2^2 - \eta_{13}\varkappa_1^2 + \tilde{e}_{12}, \\ \dot{V}(x_3, x_4, \varkappa_2) &= -\eta_{21}x_1^2 - \eta_{22}x_2^2 - \eta_{23}\varkappa_2^2 + \tilde{e}_{22}, \end{aligned} \quad (31)$$

donde:

$$\begin{aligned} k_{1j} &= k_{pj}^2 + k_{pj}(k_{dj}^2 + k_{dj} - 1) - \frac{k_{pj}(1 - k_{dj}k_{ij}) + k_{ij}}{k_{pj}}, \\ k_{2j} &= k_{pj}, \\ k_{3j} &= k_{pj}, \\ k_{4j} &= k_{pj}k_{dj}, \\ k_{5j} &= k_{ij}(k_{pj} + k_{dj}) + \frac{k_{dj}(k_{pj}(1 - k_{dj}k_{ij}) + k_{ij})}{k_{pj}}, \\ k_{6j} &= \frac{k_{pj}(1 - k_{dj}k_{ij}) + k_{ij}}{k_{pj}}, \\ \eta_{j1} &= k_{4j}k_{pj} - k_{5j}, \\ \eta_{j2} &= k_{2j}k_{dj} + k_{dj}^2 - k_{4j}, \\ \eta_{j3} &= k_{6j}k_{ij}. \end{aligned} \quad (32)$$

Empleando las definiciones (32), se verifica que las funciones de Lyapunov (30) son positivas definidas y sus derivadas temporales son definidas negativas siempre que se satisfagan las siguientes condiciones:

- $k_{pj}, k_{dj}, k_{ij} > 0$,
- $k_{dj}k_{ij} < 1$,
- k_{pj} suficientemente grande.

Por lo tanto, se concluye que el origen del sistema (29) es un punto de equilibrio Asintóticamente Estable en el sentido de Lyapunov (Khalil (2014)), lo cual implica que se satisface el objetivo de control para seguimiento de trayectorias.

De este modo se completa la descripción del segundo algoritmo de control empleado en las siguientes simulaciones numéricas y experimentos. En adelante, el algoritmo de control PID basado en observador (26) será referido como algoritmo PID.

5. Simulaciones Numéricas

En esta sección se mostrarán los resultados de simulación numérica obtenidos con los algoritmos de generación de trayectorias descritos en la Sección 2, con el objetivo de verificar la efectividad de dicha metodología. Las simulaciones se llevaron a cabo empleando la plataforma *Webots*. Dicha plataforma consiste en una aplicación de escritorio multi-plataforma de código abierto enfocada en la simulación de robots móviles tridimensionales. Originalmente se desarrolló como una herramienta de investigación para aplicar diversos algoritmos de control en robots móviles (cyb (2023)). *Webots* provee un ambiente de desarrollo completo para modelar, programar y simular robots. El entorno de simulación se puede observar en la Figura 6, donde se aprecia la vista obtenida por la cámara a bordo del vehículo en el simulador.

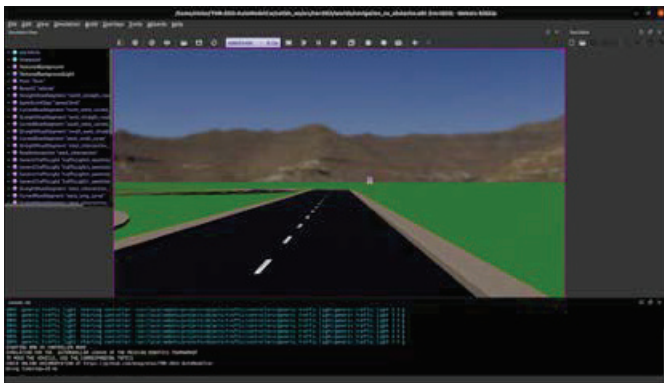


Figura 6: Entorno de simulación con plataforma *Webots*.

El método de generación de trayectorias descrito en la Sección 2 se aplica en el simulador *Webots*. En la Figura 7 se muestra el resultado obtenido con el método *Ultra Fast Structure-aware Deep Lane Detection*. Se observa que se genera la trayectoria de referencia para el vehículo situado en el entorno de simulación, la cual consiste en una trayectoria recta.

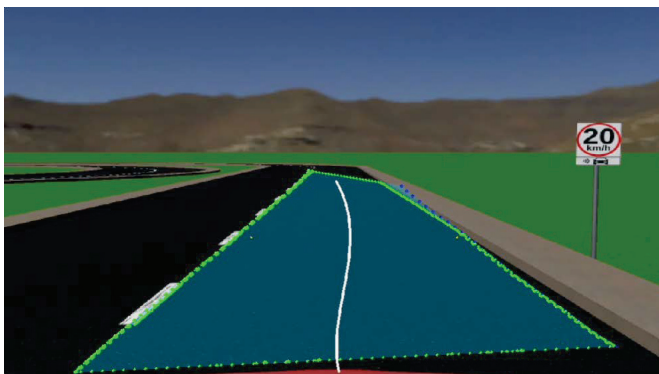


Figura 7: Resultados de simulación para método de generación de trayectorias con método *Ultra Fast Structure-aware Deep Lane Detection*.

Los resultados de generación de trayectorias cuando se emplea el algoritmo *HybridNets* se muestran en la Figura 8. Se aprecia que, al presentarse una curva, el método de generación de trayectorias se adapta y genera una trayectoria adecuada.

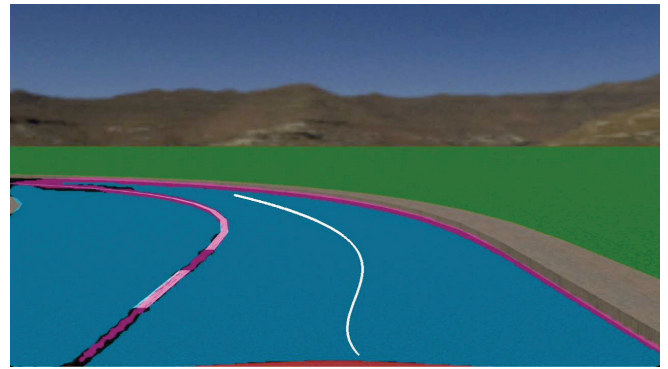


Figura 8: Resultados de simulación para método de generación de trayectorias con método *HybridNets*.

6. Resultados Experimentales

A continuación se muestran los resultados obtenidos con el método de generación de trayectorias descrito en la Sección 2, así como de la implementación de los algoritmos de control presentados en la Sección 4. Los siguientes experimentos se llevaron a cabo en la plataforma experimental *Autominy V3* (Aut (2023)), que se muestra en la Fig. 9. Este vehículo consiste en una versión a escala de un vehículo real. Para llevar a cabo la lectura de los sensores, así como el envío de las señales de control, se utiliza el Sistema Operativo Robótico ROS en su distribución Noetic. El vehículo cuenta con una computadora Intel NUC, la cual emplea el sistema operativo Ubuntu 18.04 *Bionic Beaver*.

Los resultados obtenidos para la generación de trayectorias se muestran en la Fig. 10 cuando se utiliza el algoritmo *Ultra Fast Structure-aware Deep Lane Detection*. En este caso se observa que, aún en las condiciones reales, bajo el efecto de diversos tipos de perturbaciones (perturbaciones cinemáticas, variaciones de iluminación, ruido de medición), se obtiene una trayectoria adecuada para que el vehículo pueda seguir el camino en el carril de la pista, demostrando la efectividad de la metodología de generación de trayectorias propuesta utilizando la cámara a bordo del vehículo.



Figura 9: Entorno experimental para verificar la efectividad de los algoritmos de generación de trayectorias y control.

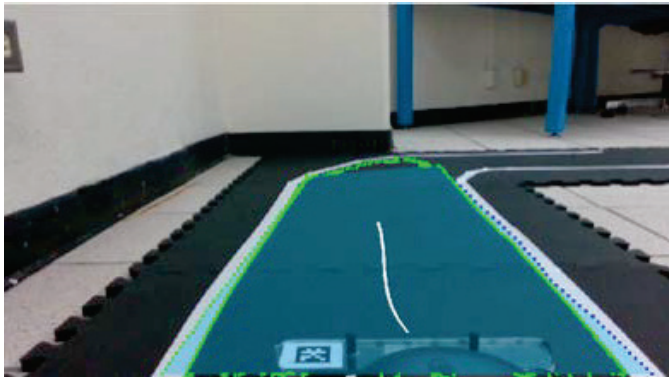


Figura 10: Resultados experimentales en vehículo Autominy utilizando el método de generación de trayectorias con el algoritmo *Ultra Fast Structure-aware Deep Lane Detection*.

Los resultados del método de generación de trayectorias empleando el algoritmo *HybridNets* se muestran en la Fig. 11. En este caso, se observa se hace una segmentación adecuada del carril en el que se ubica el vehículo. Debido a ello, se genera una señal de referencia satisfactoria.

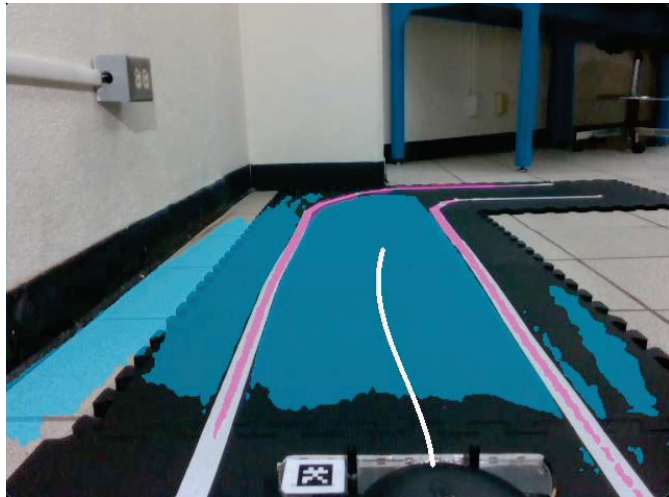


Figura 11: Resultados en vehículo Autominy para método de generación de trayectorias con el algoritmo *HybridNets*.

Para utilizar los algoritmos de control se genera una trayectoria de referencia descrita por medio de las siguientes ecuaciones:

$$\begin{aligned} x_d(t) &= 0,0052t^2 + 0,0413t + 0,62, \\ y_d(t) &= -4,4444e^{-5}t^2 + 0,0837t + 0,625, \end{aligned} \quad (33)$$

junto con las condiciones iniciales $x(0) = 0,3$, $y(0) = -0,93$, $\theta(0) = 0,7$, $\phi(0) = 0,3$. Las ecuaciones (33) se proponen siguiendo el procedimiento descrito en [Luca et al. (1998)], donde se indica que las trayectorias de referencia deben ser al menos tres veces diferenciables. Las condiciones iniciales se proponen para que el vehículo se encuentre fuera del punto de inicio de la trayectoria de referencia, así como con una orientación arbitraria.

En la Figura 12 se observan los resultados obtenidos con la plataforma experimental cuando se emplean los algoritmos de control CRD y PID junto con la trayectoria de referencia

(33). Se visualiza que el controlador PID (26) sigue satisfactoriamente la trayectoria de referencia en cada coordenada. Por su parte, el control CRD sigue satisfactoriamente la trayectoria en el plano x, y ; sin embargo, en la coordenada $\theta(t)$ se observa un elevado tiempo de convergencia, mientras que en la coordenada $\phi(t)$ se exhiben errores persistentes que no convergen a la trayectoria deseada.

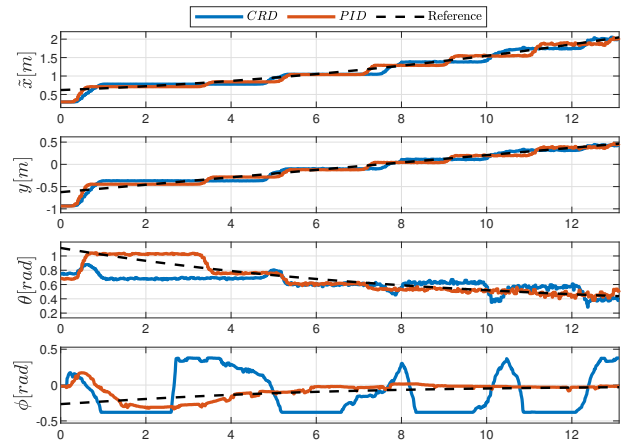


Figura 12: Resultados experimentales para seguimiento de la trayectoria de referencia (línea punteada negra) empleando los controladores CRD (señal azul) y PID (señal naranja).

En la Figura 13 se muestran los errores de seguimiento cuando se emplean los algoritmos de control PID y CRD. Se observa que los errores en las coordenadas $x(t)$ y $y(t)$ se mantienen en una vecindad cercana al origen para ambos controladores, mientras que en $\theta(t)$ y $\phi(t)$ el algoritmo PID presenta errores de menor magnitud en comparación con los exhibidos por el controlador CRD.

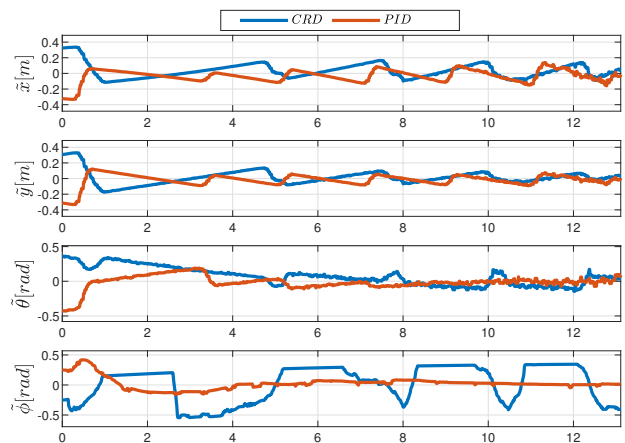


Figura 13: Seguimiento de trayectoria de referencia (línea punteada negra) empleando los controladores CRD (señal azul) y PID (señal naranja).

Los resultados anteriores muestran la efectividad tanto de la metodología de generación de trayectorias propuesta, como de los algoritmos de control para el seguimiento de trayectorias.

7. Conclusión

En este artículo se presentó una metodología que permite la generación de trayectorias para un vehículo autónomo con la

cámara a bordo del vehículo empleando algoritmos inteligentes. Además, se analizaron dos metodologías de control para seguimiento de trayectorias. Para mostrar la efectividad de las metodologías de generación de trayectorias se empleó el simulador Webots. Además, para la generación de trayectorias y algoritmos de control, se hicieron experimentos en un vehículo autónomo a escala para demostrar la efectividad de ambas metodologías. Los resultados obtenidos muestran que es posible obtener una trayectoria de referencia adecuada para el vehículo tanto en simulaciones numéricas como en experimentos. Por su parte, los resultados experimentales demuestran que, a pesar de las perturbaciones inherentes a la plataforma real, se obtienen buenos resultados tanto en la generación de trayectorias como en el seguimiento de trayectorias. Por lo tanto, se demuestra la efectividad de las metodologías desarrolladas.

Agradecimientos

Este trabajo fue apoyado por el Consejo Nacional de Humanidades, Ciencias y Tecnologías (CONAHCYT), bajo el programa “Investigadoras e Investigadores por México”, Cátedras CONAHCYT, con el proyecto 1537.

Referencias

- (2023). Autominy. *AutoMiny/AutoMiny: Software stack based on ROS for the AutoMiny model cars*.
- (2023). Simulating your robots with webots. <https://cyberbotics.com/webots>, Cyberbotics.
- Alomari, K., Mendoza, R., Goehring, D., y Rojas, R. (2021). Path following with deep reinforcement learning for autonomous cars. *Proceedings of the 2nd International Conference on Robotics, Computer Vision and Intelligent Systems - Volume 1: ROBOVIS, SciTePress*.
- Dat, V., Bao, N., y Hung, P. (2022). Hybridnets: End-to-end perception network. *arXiv*, <https://arxiv.org/abs/2203.09035>, Creative Commons Attribution 4.0 International.
- Davy, N., De, B. B., Stamatios, G., Marc, P., y Van, G. L. (2018). Towards end-to-end lane detection: an instance segmentation approach. *2018 IEEE Intelligent Vehicles Symposium (IV)*.
- Fisher, Y., Haofeng, C., Xin, W., Wenqi, X., Yingying, C., Fangchen, L., Vashisht, M., y Trevor, D. (2020). Bdd100k: A diverse driving dataset for heterogeneous multitask learning. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Hyung, L. M., Gyu, P. H., Hee, L. S., Sup, Y. K., y Soo, L. K. (2013). An adaptive cruise control system for autonomous vehicles. *International Journal of Precision Engineering and Manufacturing*.
- Khalil, H. K. (2014). Nonlinear control. *Pearson Education Limited*.
- Kramer, J., Scheutz, M., Kramer, J., Scheutz, M., y Scheutz, M. (2007). Development environments for autonomous mobile robots: A survey. <http://smart.informatik.uni-ulm.de/MIRO/>.
- Lee, M. H., Park, H. G., Lee, S. H., Yoon, K. S., y Lee, K. S. (2013). An adaptive cruise control system for autonomous vehicles. *International Journal of precision engineering and manufacturing*.
- Liu, D., Tang, M., y Fu, J. (2022). Robust adaptive trajectory tracking for wheeled mobile robots based on gaussian process regression. *Systems and Control Letters*.
- Lu, Q., Chen, J., Wang, Q., Zhang, D., Sun, M., y Su, C. Y. (2022). Practical fixed-time trajectory tracking control of constrained wheeled mobile robots with kinematic disturbances. *ISA Transactions*.
- Luca, A. D., Oriolo, G., Samson, C., y Laumond, J.-P. (1998). Feedback control of a nonholonomic car-like robot motion planning and control feedback control of a nonholonomic car-like robot. *Lectures Notes in Control and Information Sciences*.
- Miranda-Colorado, R. (2022). Observer-based proportional integral derivative control for trajectory tracking of wheeled mobile robots with kinematic disturbances. *Applied Mathematics and Computation*.
- Park, J. H., Kim, S. H., y Park, T. S. (2019). Asymptotically convergent switching differentiator. *International Journal of Adaptive Control and Signal Processing*.
- Rosas-Vilchis, A., de Loza, A. F., Aguilar, L. T., Cieslak, J., Henry, D., y Montiel-Ross, O. (2020). Trajectory tracking control for an autonomous vehicle using a decoupling approach. *2020 28th Mediterranean Conference on Control and Automation (MED)*.
- TuSimple, A. (2022). Tusimple-benchmark. <https://github.com/TuSimple/tusimple-benchmark>.
- Yoo, S. J. (2013). Adaptive neural tracking and obstacle avoidance of uncertain mobile robots with unknown skidding and slipping. *Information Sciences*.
- Zequan, Q., Huanyu, W., y Xi, L. (2020). Ultra fast structure-aware deep lane detection. *Computer Vision – ECCV 2020, Springer International Publishing, 276–291*, available at <https://github.com/cfzd/Ultra-Fast-Lane-Detection>.