Navegación Autónoma para vehículos sin conductor

David Trejo [1] Marco Negrete [1]

[1]Facultad de Ingeniería UNAM

Introducción

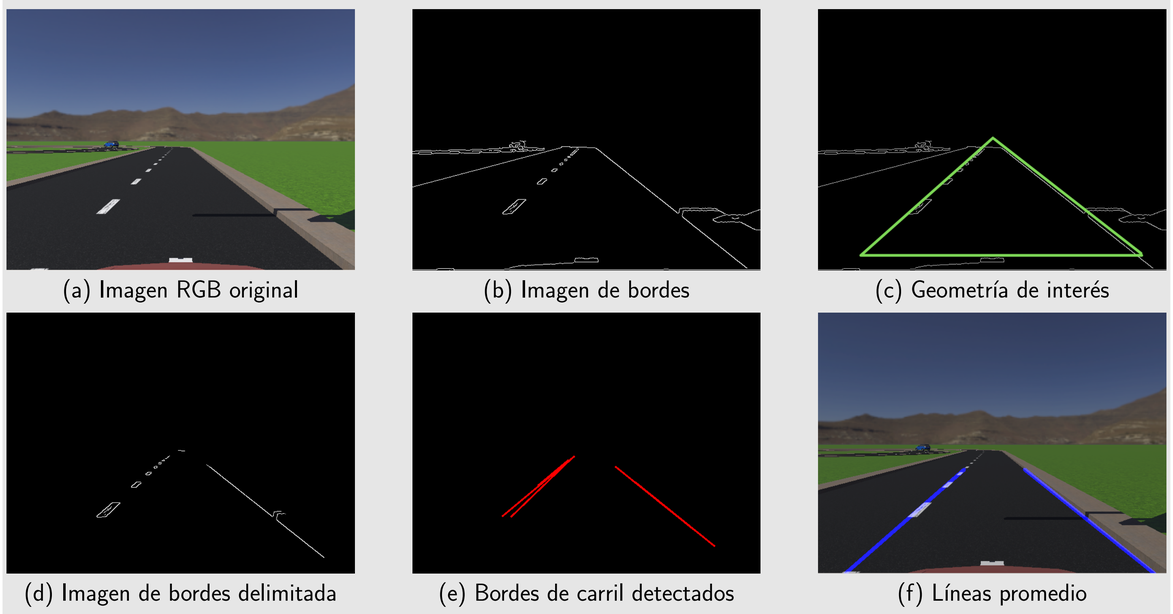
Los accidentes viales en su mayoría son provocados por errores humanos, los vehículos autónomos cuentan con el potencial de reducirlos en base a toma de decisiones bien planeadas y acciones supervisadas constantemente.
Este póster contiene algunas de las múltiples herramientas teórico matemáticas necesarias para la elaboración de un sistema de navegación autónoma para vehículos sin conductor, en este caso utilizando el simulador **Webots** [Webots n.d.] y la plataforma **ROS**. Con el propósito de crear este sistema se desarrollaron los siguientes módulos para tareas específicas.

- **Detección de Carril** por medio de algoritmos de detección de bordes de **Canny** y transformada **Hough** para procesamiento de imágenes **RGB** implementados con lenguaje **Python** y uso de funciones de **OpenCV**.
- **Diseño de leyes de control** para funciones operativas (movimiento lateral y longitudinal) del vehículo que involucran **velocidad** y **ángulo de dirección**.
- **Seguimiento de Carril** como combinación de las tareas de **Detección de Carril** y **Leyes de Control**.
- **Detección de Obstáculos** a partir de la implementación de un algoritmo de **aprendizaje no supervisado** para agrupación de datos, desarrollado en lenguaje **C++**.
- **Evasión de Obstáculos** en base al diseño de diferentes comportamientos modelados a través de **máquinas de estados finitos** implementadas en **Python**.

Detección de Carril

El objetivo de la detección de carriles es obtener una característica particular de la imagen que funcione como guía durante el movimiento del vehículo. Las líneas pertenecientes a los bordes de cada línea del carril son el parámetro necesario para mantener al vehículo autónomo dentro del circuito.
Proceso de **Detección de Carril**

1. Obtener una imagen **RGB** que ilustre la parte frontal del escenario de pruebas.
2. Extraer los bordes de la imagen con **Detector de bordes de Canny** [Gonzalez 2009].
3. Delimitar el área donde se encuentra el carril deseado por medio una geometría.
4. Utilizar **Transformada de Hough** [Forsyth and Ponce 2011] para detección de líneas rectas dentro de el área de interés.
5. Obtener una línea recta en cada borde del carril expresada en **forma polar**.



(a) Imagen RGB original (b) Imagen de bordes (c) Geometría de interés

(d) Imagen de bordes delimitada (e) Bordes de carril detectados (f) Líneas promedio

Figure 1. Líneas del carril detectadas por medio del Detector de bordes de Canny y Transformada Hough

Seguimiento de Carril

El seguimiento de carril permite la navegación del vehículo autónomo dentro del escenario de pruebas tomando en cuenta el error que existe entre las líneas detectadas y originales del carril.

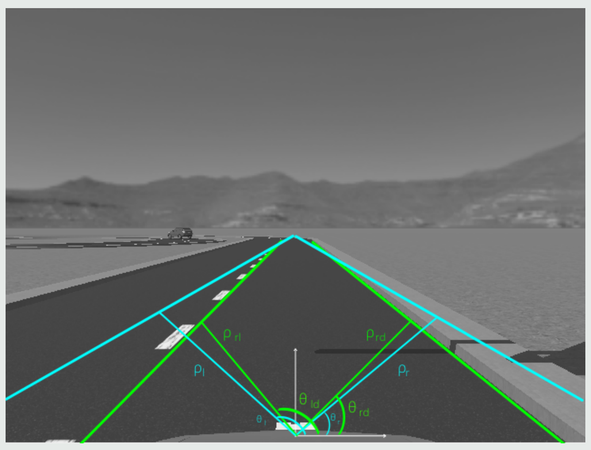


Figure 2. Leyes de Control

Movimiento lateral

Las leyes de control de control que permiten el movimiento lateral del vehículo por medio del ángulo de dirección δ corresponden a las siguientes ecuaciones.

$$\delta = K_\rho e_{\rho p} + K_\theta e_{\theta p}$$

Donde:

$$e_{\rho p} = (\rho_{ld} - \rho_l + \rho_{rd} - \rho_r)/2 \qquad e_{\theta p} = (\theta_{ld} - \theta_l + \theta_{rd} - \theta_r)/2$$

Movimiento longitudinal

El control de velocidad v del vehículo autónomo es constante en todo momento. Es decir, no existe un calculo para este parámetro mientras se encuentre en navegación sin obstáculos y es establecido al inicio de cada simulación.

$$v = C$$

Detección de obstáculos

La detección de obstáculos es alcanzada mediante el algoritmo **K-means** [Han et al. 2011], el cual utiliza una técnica de división basada en la distancia entre objetos y permite agrupar los objetos mientras más cercanos se encuentren. El conjunto de datos por agrupar es la **nube de puntos** otorgada por el sensor **LIDAR**.

Detección de obstáculos

Proceso de **Detección de Obstáculos**

1. Obtener la **nube de puntos** del sensor **LIDAR**.
2. Definir el número de grupos iniciales.
3. Aplicar **K-means** al conjunto de datos.
4. Obtener la posición de los obstáculos identificados mediante su **centroide**.

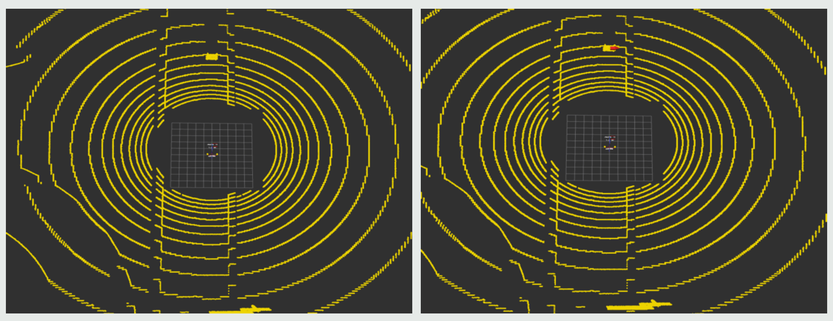


Figure 3. Detección de objetos con el algoritmo de clustering k-means.

Evasión de Obstáculos

La evasión de obstáculos se logra definiendo dos comportamientos: **Crucero** y **Rebase**. Mientras no se detecten obstáculos el vehículo se mantiene en **Crucero**, por otro lado cuando existan obstáculos en el camino del vehículo, este cambia al comportamiento de **Rebase**.

Crucero

El comportamiento de crucero hace uso del **Seguidor de carril** con las mismas leyes de control para movimiento lateral y longitudinal.

Rebase

La acción de rebase es completada por medio de una **máquina de estados** que calcula las leyes de control para movimiento lateral. En cambio, la velocidad v del movimiento longitudinal también es constante pero menor que en el caso del comportamiento de Crucero.

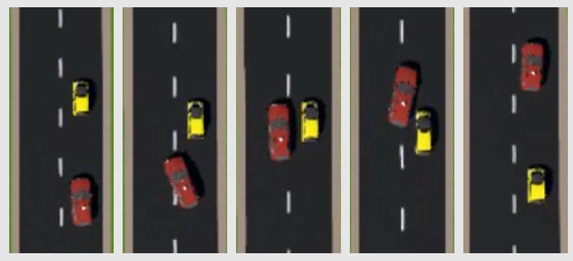


Figure 4. Comportamiento de Rebase.

Estacionamiento Autónomo

El **estacionamiento autónomo** se resolvió con el siguiente algoritmo.

1. Avanzar hasta detectar dos obstáculos no contiguos.
2. Frenar.
3. Retroceder t segundos con ángulo de dirección θ en sentido horario.
4. Retroceder t segundos con ángulo de dirección θ en sentido antihorario.
5. Frenar.
6. Avanzar un t segundos con ángulo de dirección $\theta = 0°$ para alinear.

La velocidad v siempre es constante durante la acción de estacionamiento autónomo.

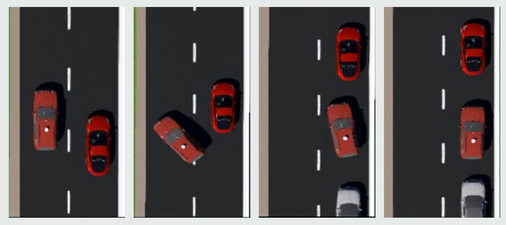


Figure 5. Estacionamiento Autónomo

References

Webots (n.d.). *http://www.cyberbotics.com*. Ed. by C. Ltd. Open-source Mobile Robot Simulation Software. URL: http://www.cyberbotics.com.

Forsyth, D. and J. Ponce (2011). *Computer vision: A modern approach*. Prentice hall.

Han, J., J. Pei, and M. Kamber (2011). *Data mining: concepts and techniques*. Elsevier.

Gonzalez, R. C. (2009). *Digital image processing*. Pearson education india.