Universidad Autónoma de Querétaro
Facultad de Ingeniería
Maestría en Inteligencia Artificial

PROPUESTA METODOLÓGICA PARA EL DESARROLLO DE LA ETAPA DE PERCEPCIÓN AMBIENTAL EN UN VEHÍCULO AUTÓNOMO

TESIS
Que como parte de los requisitos para obtener el grado de
Maestro en Ciencias en Inteligencia Artificial

Presenta:
Ing. Juan Ignacio Ortega Gómez

Dirigido por:
Dr. Luis Alberto Morales Hernández
Co-Director
Dr. Irving Armando Cruz Albarrán
SINODALES

Dr. Luis Alberto Morales Hernández
Presidente

Dr. Irving Armando Cruz Albarrán
Secretario

Dr. Sebastián Salazar Colores
Vocal

Dr. Saúl Tovar Arriaga
Suplente

Dr. Andras Takacs
Suplente

Centro Universitario, Querétaro, QRO
Abril 2024

México.

La presente Tesis está dedicada a Dios, por otorgarme vida, salud y bienestar y a mis Padres por guiarme con el ejemplo por el camino del esfuerzo y dedicación.

Agradecimientos

El desarrollo de este proyecto de tesis fue posible gracias al apoyo de muchas personas de mi entorno social y familiar, les brindo un agradecimiento a cada uno de los que directa o indirectamente me motivaron y me brindaron su comprensión a lo largo de la investigación.

Mi agradecimiento al Dr. Luis Alberto Morales Hernández, en un principio por creer en mi trabajo y permitirme sentar las bases para el desarrollo de investigaciones en vehículos autónomos en la institución. Posteriormente, por guiarme, asesorarme y otorgarme consejos junto con herramientas para facilitar la investigación y el método de creación de evidencias de resultados.

Mi agradecimiento al Dr. Irving Armando Cruz Albarrán, por su paciencia, tiempo y análisis a detalle de cada retroalimentación relacionada con esta investigación.

Agradecimientos al CONAHCYT y a la institución, por brindarme su apoyo económico durante mi estancia como estudiante de posgrado bajo el número de becario y CVU 1144283.

Mis agradecimientos a mi familia por su apoyo y comprensión durante las decisiones que me permitieron realizar esta investigación.

Agradecimiento especial a mi pareja, por creer en mi, apoyarme y motivarme para siempre dar lo mejor.

Resumen

En la actualidad han surgido sistemas de manejo autónomo con la promesa de prevenir accidentes viales, la creación de estos sistemas al más alto nivel implica el desarrollo de una serie de funciones. La primera función es la percepción ambiental, encargada de percibir los alrededores del vehículo mediante la detección y clasificación de cada objeto de interés en una escena urbana. Con el objetivo de generar esta primera función, se desarrolla una metodología para generar un software de percepción ambiental; junto con un despliegue de componentes necesarios para su construcción, de tal forma que funja como base para posteriores trabajos dentro del área del manejo autónomo. El método propuesto consiste en los siguientes pasos: selección de componentes base, generación de base de datos en base a KITTI Benchmark Suite, desarrollo del modelo para la percepción, desarrollo del sistema de percepción ambiental y montaje del sistema de percepción ambiental. Como resultados de este proyecto se generan innovaciones en tres áreas: Primero, durante la creación de la base de datos especializada, desarrollada con manipulaciones matemáticas, que presenta métricas de Puntaje-F1: 95.77%, Precisión-promedio: 92.54%, Exactitud: 97.53%, Precisión: 94.34%, y Sensitividad: 97.25%. Segundo, durante la implementación del sistema de percepción ambiental, con métricas de Exactitud: 97.18%, Media-de-precisión-promedio: 97.03%, Media-de-intersección-sobre-unión: 89.00% para la clasificación, Media-de-precisión-promedio: 85.11% y Media-de-intersección-sobre-unión: 70.51% para la detección. Tercero, durante la propuesta del desarrollo del sistema de percepción ambiental, donde se obtuvieron métricas de segmentación del camino superiores al de trabajos previos y similares para la detección de vehículos, pero considerando la rotación de los autos, innovando en ese aspecto. En este trabajo, se llena un vacío en el estado-del-arte sobre métodos descriptivos que faciliten la producción de un sistema de percepción ambiental y se mejora la capacidad de producción de análisis y entrega de resultados por segundo.

Palabras clave: percepción, autónomo, vehículo, segmentación, detección

Abstract

Today, autonomous driving systems have emerged with the promise of preventing road accidents, the creation of these systems at the highest level involves the development of a series of functions. The first function is environmental perception, in charge of perceiving the vehicle's surroundings by detecting and classifying each object of interest in an urban scene. To generate this first function, a methodology is developed to generate an environmental perception software; along with a deployment of components necessary for its construction, it serves as a basis for further work in the area of autonomous driving. The proposed method consists of the following steps: selection of base components, generation of the database based on the KITTI Benchmark Suite, development of the model for perception, development of the environmental perception system, and assembly of the environmental perception system. As result of this project, innovations are generated in three areas: First, during the creation of the specialized database, developed with mathematical manipulations, which presents metrics of Score-F1: 95.77%, Accuracy-average: 92.54%, Accuracy: 97.53%, Precision: 94.34%, and Sensitivity: 97.25%. Second, during the implementation of the environmental perception system, with metrics of Accuracy: 97.18%, Mean-of-precision-average: 97.03%, Mean-of-intersection-above-union: 89.00% for classification, Mean-of-precision-average: 85.11%, and Mean-of-intersection-above-union: 70.51% for detection. Third, during the proposed development of the environmental perception system, where road segmentation metrics superior to previous and similar works were obtained for vehicle detection, but considering the rotation of cars, innovating in that aspect. This work fills a gap in the state-of-the-art descriptive methods that facilitate the production of an environmental perception system and improves the production capacity of analysis and delivery of results per second.

Keywords: perception, autonomous, vehicle, segmentation, detection

Contenido

Índice de figuras

Índice de Tablas

1. Introducción

1.1 Justificación

Este proyecto de investigación es parte de un trabajo global que tiene como intención desarrollar un vehículo autónomo. Una parte medular para desarrollar este vehículo, de acuerdo con [1] y [2], es la percepción ambiental, la cual permite al auto conocer el ambiente en donde se encuentra en cada instante e identificar así el camino y otros objetos de interés; como pueden ser: autos, peatones, entre otros, para poder tomar decisiones durante la planeación de su movimiento.

De acuerdo con lo anterior, el aporte general de este proyecto consistió en una propuesta metodológica que funge como base para la transformación de un auto común a vehículo autónomo, en particular al auto eléctrico de la facultad de Ingeniería de la UAQ. Además, se aportaron innovaciones en el área de segmentación del camino urbano; y detección más exacta de posición de objetos de interés en áreas urbanas, ambos a partir de nubes de puntos.

En cuanto a aportaciones sociales y económicas, este trabajo aterrizó los conceptos sobre percepción ambiental en el manejo autónomo, lo que permitió y permitirá, tanto una posterior colaboración con el sector privado y educativo en proyectos asociados, como iniciar investigaciones en el área de vehículos autónomos en el país que sigan los niveles propuestos por la SAE, buscando propiciar la inversión en investigación en el sector privado y educativo sobre estos temas.

Además, es destacable la necesidad del manejo de conceptos y habilidades relacionadas con el aprendizaje máquina y en específico con inteligencia artificial, al ser un proyecto que tiene como base el uso, manipulación y desarrollo de redes neuronales junto con las herramientas que estas conllevan. Siendo un trabajo de investigación enteramente relacionado con la Maestría en Ciencias en Inteligencia Artificial.

1.2 Planteamiento del problema

De acuerdo con [3], el 94% de los accidentes viales son ocasionados por el mal desempeño del piloto, surgiendo la idea de un manejo autónomo con la promesa de prevenir accidentes [4]. Estos sistemas autónomos están actualmente en desarrollo y relacionados con investigaciones en inteligencia artificial, de acuerdo con el análisis expuesto en la Figura 1.1. La figura resultó de una búsqueda en

la base de datos de 'ScienceDirect', en donde se seleccionaron los principales 100 artículos con las palabras clave de inteligencia artificial y vehículo autónomo.

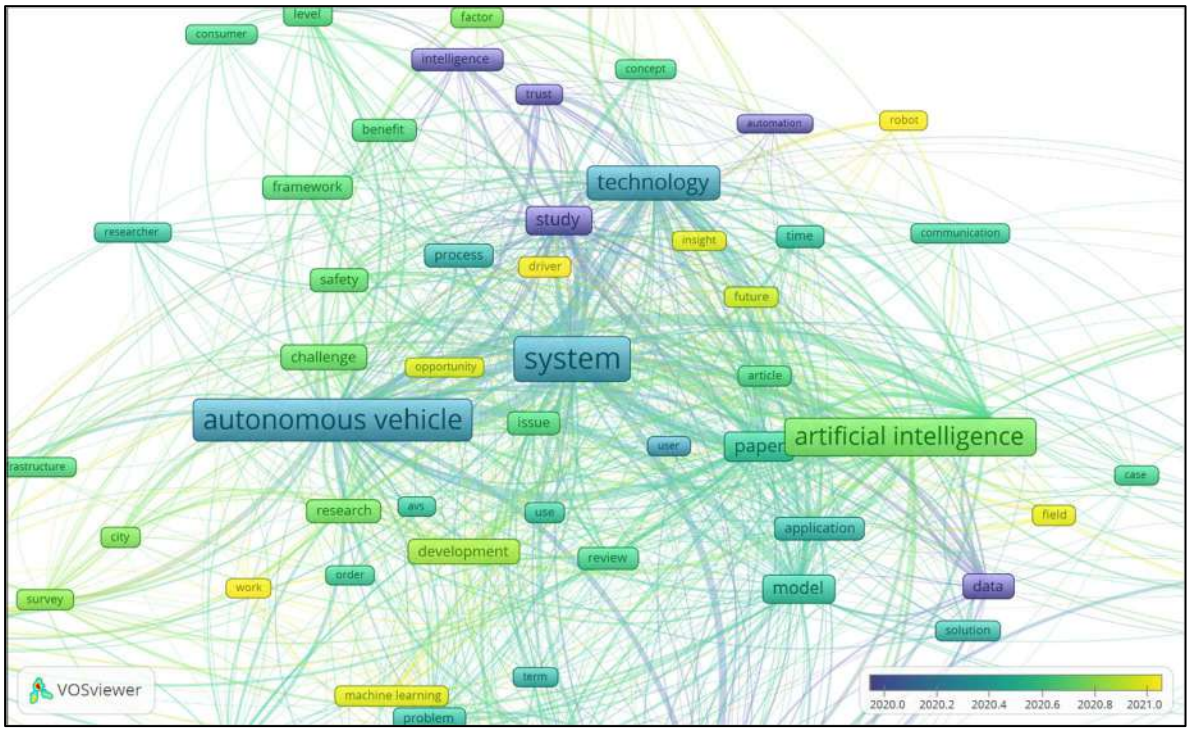


Figura 1.1 Análisis mediante el software VOSviewer [6] de los principales 100 artículos, del 2020 al 2021, relacionados con las palabras clave inteligencia artificial y vehículos autónomos.

Ahora, para el desarrollo de un vehículo autónomo, existen varias tareas que deben ser cumplidas para culminar su construcción [5]. Entre estas, la inteligencia artificial es necesaria en la mayoría, siendo la percepción ambiental y la planeación del movimiento donde se tiene una mayor complejidad de desarrollo; debido a que en conjunto genera el movimiento autónomo inicial de un vehículo, sin embargo, la percepción ambiental es la tarea prioritaria al ser la entrada que necesita la planeación de movimiento. A pesar de esto, el estado del arte presenta investigaciones solamente sobre bases de datos orientadas al manejo autónomo, modelos de inteligencia artificial orientados a la percepción ambiental e implementaciones en computador y RaspberryPi, denotando la falta de información y recursos en el estado del arte, hasta donde este trabajo concierne, sobre:

1. Bases de datos de nubes de puntos con escenas de vista aérea de segmentación.
2. Descripciones del método de desarrollo de modelos de inteligencia artificial.
3. Implementaciones de percepción ambiental en hardware especializado para inteligencia artificial.

1.3 Hipótesis

Si se investiga y se genera una apropiación del conocimiento sobre sistemas de percepción ambiental del estado-del-arte y se recopila información sobre hardware de innovación especializado para aplicaciones de inteligencia artificial, será posible desarrollar una metodología para la creación de un sistema de percepción ambiental descriptiva y que aporte innovaciones al estado-del-arte.

1.4 Objetivos

1.4.1 Objetivo general

Desarrollar una metodología que permita el desarrollo del software de la etapa de percepción ambiental para un vehículo autónomo y el despliegue de sus componentes de hardware necesarios; a través de la apropiación del conocimiento del estado del arte sobre modelos de percepción ambiental y hardware de innovación, de tal forma que el procedimiento propuesto funja como la base para transformar un auto común a uno autónomo y facilitar posteriores trabajos dentro del área del manejo autónomo.

1.4.2 Objetivos específicos

1. Integrar el conocimiento del estado del arte sobre investigaciones asociadas con la percepción ambiental; mediante la recopilación de los artículos relacionados más recientes, para realizar la apropiación del conocimiento ya existente.
2. Recabar información sobre hardware embebido de innovación especializado para trabajos de inteligencia artificial; investigando las características de la variedad de componentes existentes en el mercado, con la meta de elegir el hardware con mejores recursos para la ejecución de la percepción ambiental.
3. Proponer una serie de pasos que permitan el desarrollo del software de la etapa de percepción ambiental y el despliegue de los componentes de hardware necesarios, utilizando los conocimientos adquiridos sobre percepción ambiental y hardware de innovación para definir el método de desarrollo de percepción ambiental en un vehículo.
4. Poner a prueba la propuesta metodológica a través de un caso de estudio para comprobar que el método definido para el desarrollo de la percepción ambiental esta completo, es entendible y replicable.
5. Realizar retroalimentación sobre la metodología propuesta para afinar cada uno de los pasos metodológicos, en base a los resultados obtenidos durante el caso de estudio.

1.5 Estructura de tesis

Esta tesis está organizada de forma general como se muestra en la figura 1.2 y de manera más particular de la forma en que se describe a continuación:

Capítulo 1. Toda la información referente al planteamiento de la investigación es mostrada en este capítulo, fungiendo como introducción a lo tratado en esta investigación.

Capítulo 2. Se describen las investigaciones del estado del arte que mayor asociación se les encontró con los vehículos autónomos y con la percepción ambiental, de forma tal que, cada uno forma parte de la historia que derivó a las proposiciones y conceptos que hoy se tienen sobre estas áreas, al menos hasta lo que a esta investigación le concierne.

Capítulo 3. En esta sección general, se plantean una serie de pasos que hacen posible el desarrollo del software y hardware para un sistema de percepción ambiental. Cada sección de este capítulo cuenta con descripciones específicas de cómo cumplir cada uno de los pasos propuestos, partiendo de explicaciones generales que permiten variar lo que será utilizado, hasta lo particular proponiendo exactamente qué elementos utilizar y considerar. Todo el procedimiento recomendado, basado siempre en el caso de estudio realizado, donde se puso a prueba la metodología propuesta y en base a estas pruebas se hicieron correcciones en la misma.

Capítulo 4. Finalmente, se muestran los resultados que son posibles de obtener a partir de la metodología propuesta, utilizando exactamente cada uno de los conceptos descritos en el capítulo 3. Además, se analizan las aportaciones que esta investigación logra en diferentes rubros y las aportaciones resultantes de esta investigación.

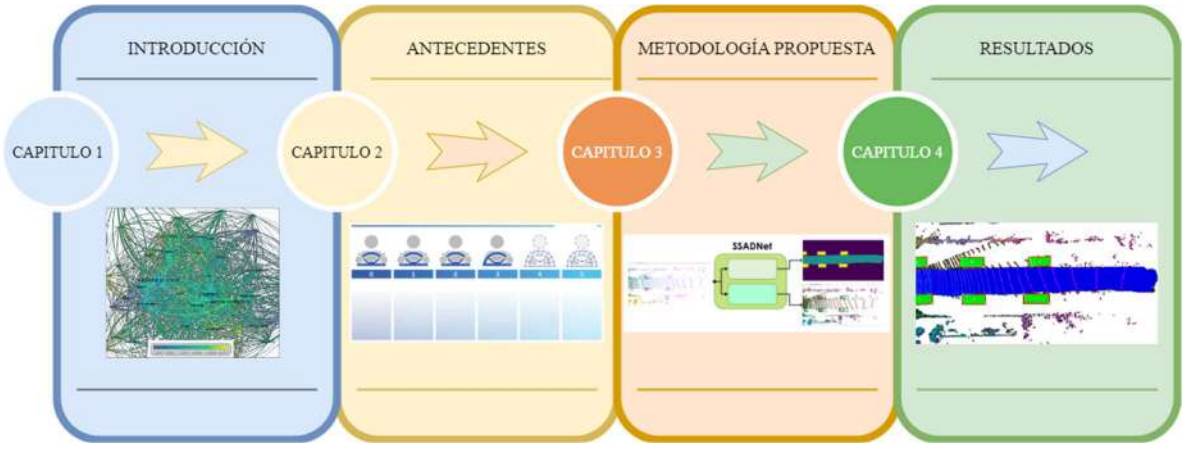


Figura 1.2. Diagrama de la propuesta metodológica planteada en este trabajo, de forma secuencial.

2. Antecedentes

2.1 Primeras investigaciones sobre la conducción autónoma

El desarrollo del proyecto ALVINN marco el inicio de las investigaciones sobre conducción autónoma en 1989, dentro de sus propuestas principales estaba el uso de redes neuronales con propagación hacia atrás, o back-propagation en inglés, para resolver problemas de la vida diaria, en su caso, aplicándolo a un auto que seguía el camino de forma automática bajo condiciones de camino específicos; utilizando una cámara y un láser [7]. Posteriormente, en 1991, se creó el proyecto VITA I en Europa con el objetivo de desarrollar un vehículo automatizado, resultando en múltiples enseñanzas que se vieron reflejadas entre 1995 y 1997 en una segunda versión del proyecto, VITA II, en donde generaron un vehículo capaz de evitar obstáculos; diseñando el control con un hardware robusto que utilizaba cámaras de video como señales de entrada. Este proyecto, tuvo buenos resultados en aplicaciones donde no existían intersecciones en el camino, sin embargo, solo era funcional con las condiciones de iluminación y clima adecuados; ya que fue diseñado para trabajar en un ambiente controlado con señales viales de colores [8].

2.2 Raíz de las recientes investigaciones

Una vez planteada la idea de un vehículo autónomo, muchas fueron las expectativas al respecto, sin embargo, la evolución se vio estancada por falta de resultados aplicables; siendo hasta que un estudio realizado por la Administración Nacional de Seguridad del Tráfico de Alta Vía (NHTSA) entre 2005 y 2007, despertó nuevamente la ilusión de la creación de un vehículo autónomo, ya que, recopilando información sobre los eventos y factores que causaron un accidente en vehículos ligeros; determinaron la cadena de sucesos que suelen llevar a un accidente, resultando en que el 94% de los sucesos fueron generados a causa de errores humanos, como lo son: errores de decisión, de reconocimiento, de rendimiento, entre otros [3]. Sugiriendo la idea de quitar a los humanos de la ecuación, para reducir los percances viales.

Guiados por la estadística de la causa raíz de los accidentes y el objetivo de desplazar a los humanos de la conducción, sistemas de manejo autónomo han surgido con la promesa de prevenir accidentes; generando investigaciones que se preocupan, en primera instancia, de integrar las técnicas y métodos que ya se conocen [4] para el desarrollo de un auto autónomo.

Actualmente, las investigaciones sobre la conducción autónoma ha creció tanto, que la sociedad de Ingeniería Automovilística (SAE por sus siglas en inglés) recomienda dividir los niveles de los autos autónomos en seis, que van desde la ausencia de la automatización, hasta una completa automatización de la conducción [9]. Estos niveles, presentados en la Figura 2.1, son aterrizados de una mejor manera por el departamento de transporte de EUA [10].

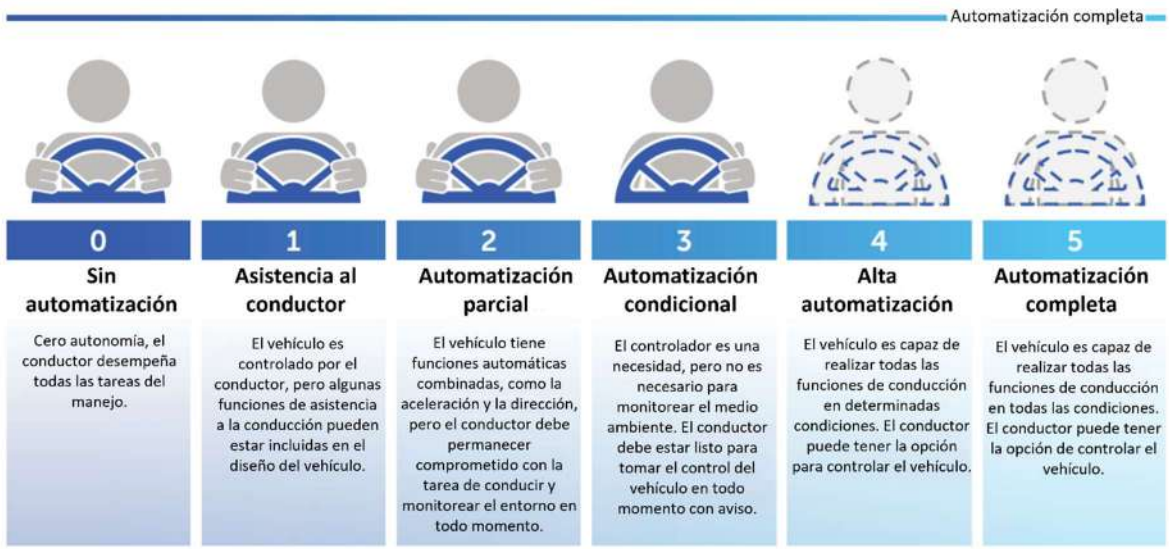


Figura 2.1 Niveles de automatización, adaptada de [10].

2.3 Investigaciones relevantes para los sistemas de manejo

Una vez fijado el objetivo de la creación de vehículos autónomos, las investigaciones relacionadas con el tema comenzaron a surgir, resultando en un estado del arte con contenido relevante para el manejo autónomo.

Un primer ejemplo de contenido de interés, es la investigación sobre sistemas de frenado autónomo desarrollado en 2018, dicho sistema utilizó una red-Q profunda para decidir, en base a obstáculos detectados por los sensores, cuándo generar un break en el auto para detenerlo progresivamente; tomando la velocidad del vehículo y la posición del obstáculo como las variables clave para definir el progreso del frenado [11].

Dentro del mismo camino, se encuentra el manejo autónomo con aprendizaje imitado, que consiste en tomar imágenes de la conducción de una persona; para aprender a controlar en línea recta un vehículo de forma autónoma, sin embargo, a este tipo de manejo no le era posible tomar decisiones correctas al buscar girar en una intersección, por lo que en 2018 se publicó una innovación a este sistema, donde mejoraron el proceso al desconectar el aprendizaje imitado durante la intersección

para guiarlo de forma manual o mediante direcciones dadas por una guía topológica de forma automática [12].

Siguiendo la búsqueda de innovaciones relevantes en el campo de la conducción autónoma, en el estado del arte retomaron los sistemas de asistencia de manejo, DAS por sus siglas en inglés, los cuales proporcionan cierta autonomía al vehículo en un área específica a la vez; siendo necesario combinar diferentes tipos de DAS para generar autonomía en ciertas tareas del manejo, por lo que la innovación propuesta en 2018, consiste en dar respuesta a la interrogante de cuándo cada tipo de DAS tiene que entrar en acción. Como resultado desarrollaron un método para entrenar a un supervisor seleccionador del mejor asistente de manejo, mediante aprendizaje-Q profundo, que selecciona el DAS más apropiado según el momento durante la conducción, dicho método obtuvo buenos resultados simulados en los que el automóvil pudo circular en una autopista a buena velocidad y fue capaz de adelantar otros vehículos sin cambios innecesarios de línea [13].

Aparte de las investigaciones relacionadas con la manera en que el vehículo debe ser conducido o cómo se puede mejorar la conducción, en el estado del arte se encuentra un artículo, publicado en 2018, en que el objetivo principal es una tarea que puede ayudar a automatizar el manejo. En dicho trabajo se propuso una estrategia de arreglos matemáticos combinados para realizar el aprendizaje multitarea, abreviada como MTL, técnica que se encarga de detectar objetos y predecir su distancia con modelos separados, generando un producto cartesiano basado en la estrategia de combinación multitarea, abreviada como CP-MTL, para modelar conjuntamente, mediante una red neuronal convolucional, las tareas del MTL clásico [14].

Avanzando en el tiempo, en 2019, se continuaron publicando artículos relevantes para la creación de vehículos autónomos. En una primera investigación de ese mismo año a considerar, desarrollaron un sistema de navegación y localización variable a través del diseño de una nueva red convolucional variable de extremo a extremo, que integra datos rudos sensados con mapas de rutas marcadas y no señalizadas, permitiendo habilitar la navegación y la localización en complejos ambientes de manejo. También dentro del mismo trabajo, formularon un algoritmo de localización que utiliza la red desarrollada de entrenada para tomar decisiones sobre la ruta topológica asignada y de esa forma, predecir la posición del vehículo a través de la creación de correspondencias entre el mapa y la apariencia del camino visualizado. Finalmente, en esa misma investigación, evaluaron los algoritmos con una base de datos del mundo real, demostrando que es posible la navegación con volante controlado y mejorando la localización de la posición, incluso en lugares con información de GPS limitada [15].

En otra investigación del mismo año, por primera vez demostraron buenos resultados aplicando el aprendizaje profundo reforzado en el manejo automatizado. El modelo que desarrollaron es capaz de

aprender una política de seguimiento de carril con algunas iteraciones de entrenamiento, utilizando imágenes monoculares individuales como entrada, en este trabajo plantean que es probable que se necesite diseñar una función de recompensa más efectiva para que el modelo sea capaz de aprender a ser un agente de manejo súper humano [16].

Siguiendo con la tendencia de investigaciones en el área de vehículos autónomos, en 2019 desarrollaron una nueva metodología integrando dos dominios, robótica y tráfico ingenieril. Dicho método, consistía en crear una red de nivel de riesgo que utiliza un aumento en la percepción del ambiente, combinado con la obtención de información de seguridad en tiempo real, para segmentar la carretera en la que se viaja; mejorando la evaluación de riesgo de los vehículos autónomos. Con las mejoras mencionadas, aumentaron la probabilidad de detectar amenazas y con el uso de los niveles de redes de predicción; facilitaron una mayor frecuencia de datos, lo cual permitió a los autos autónomos reducir la velocidad, cambiar su trayectoria o dar la opción de que el pasajero tome el control, cumpliendo con el objetivo de generar un viaje más seguro [17].

Las investigaciones relacionadas con los vehículos autónomos continuaron en el siguiente año 2020, con el desarrollo de un nuevo método 3D de detección de vehículos, denominado 'multi-task faster R-CNN' en inglés, que utiliza los puntos de coordenadas de un cuadro 2D delimitador, obtenido de una imagen simple, para delimitar la geometría y generar las coordenadas 3D del objeto mediante una transformación inversa de la perspectiva proyectada. Posteriormente, utilizando aprendizaje multitarea, integraron la detección 2D y 3D de un vehículo en la misma 'faster R-CNN' y obtuvieron un rendimiento competitivo contra otros modelos similares en el estado del arte [18].

En el mismo año de 2020, en un artículo presentaron una nueva alternativa para la obtención de la información y por ende, una nueva manera de detección de vehículos. En el trabajo presentado, propusieron un sistema que utiliza dos cámaras montadas como una cámara estéreo en el vehículo anfitrión para medir la distancia entre este y el auto detectado en tiempo real. La detección de vehículos se realiza mediante dos pasos, primero utilizan algoritmos de detección con una sola cámara y luego, el mismo vehículo se detecta en la segunda cámara; con la técnica de coincidencia de plantillas. Así, la distancia entre vehículos se calcula mediante un método matemático basado en la posición del vehículo en ambas cámaras. Esta propuesta arrojó los mejores resultados de precisión de distancia entre vehículos comparada con otros métodos que hasta la fecha se habían publicado en el estado del arte [19].

En una última investigación de 2020 que se encontró fuertemente relacionada con los vehículos autónomos, se presentó la propuesta de otra alternativa para la obtención de la información. Con esta propuesta, surge un nuevo método para la detección de obstáculos mediante un radar mmWave y un sensor de visión, donde la escasez de puntos que se presenta al usar un radar se contrarresta con el

método de fusión de atención espacial (SAF por sus siglas en inglés), propuesto en esta investigación. Además, construyeron un método que convierte los puntos de radar en imágenes de radar para el entrenamiento de una red neuronal de aprendizaje profundo. La investigación, presentí resultados numéricos que sugieren que el método de fusión que desarrollaron logró un rendimiento superior en la evaluación, en comparativa con otros trabajos similares del estado del arte [20].

2.4 Enfoque actual sobre las tareas de un vehículo autónomo

Actualmente se considera que existen diferentes tareas, mostradas en la figura 2.2, que un vehículo autónomo debe cumplir para realizar la conducción autónoma, para cada una de estas tareas se proponen algoritmos que las desarrollen. El primer aspecto crítico es la tarea de la percepción, que se encarga de percibir los alrededores del vehículo autónomo mediante la detección, clasificación y definición de ubicación correspondiente de cada objeto [5].

Figura 2.2 Tareas del manejo autónomo, adaptada de [5].

2.5 Investigaciones relevantes en la percepción ambiental

De acuerdo a lo planteado, para poder crear un vehículo autónomo, primeramente, es necesario generar la percepción ambiental en el auto, por lo cual es necesario analizar el estado del arte en búsqueda de los trabajos más relacionados con esta tarea. Una primer investigación la encontramos en [21], trabajo de 2017 donde realizaron el primer resumen enfocado en el uso de Aprendizaje Profundo o Deep Learning en inglés, para llevar a cabo la segmentación. En el artículo que publicaron muestran bases de datos, métodos de Deep Learning y evaluaciones de desempeño de diferentes pruebas realizadas para la segmentación.

En un segundo artículo, realizaron una descripción general de los problemas de confiabilidad y solidez relacionados con el procesamiento y la percepción de los sensores en 2017. Detallan las etapas críticas de percepción y proporcionan una presentación de sensores integrados utilizables. Además, en este estudio del estado del arte de los sistemas altamente automatizados, se proporcionan algunas observaciones y comentarios sobre los límites de estos sistemas [1].

En otra investigación de 2020 relacionada con la percepción ambiental [2], presentaron un sistema de conducción autónomo asistido por comunicación V2X para vehículos. Este sistema general, se compone de tres subsistemas: más allá de la percepción de la línea de visión (abreviado como BLOS), planificación ampliada y control. Específicamente, el subsistema de percepción BLOS facilita la percepción ambiental, a través de la fusión de dos métodos de extracción de datos, el primero que consta de sensores integrados para una percepción local y el segundo que utiliza la comunicación V2X para una percepción a distancia.

Durante el mismo año de 2020, en otro trabajo de investigación propusieron una red neuronal multitarea que permitió una segmentación semántica y afinación profunda simultáneas (SSDNet multitarea). Toda la red se implementó de un extremo a otro y se evaluó en conjuntos de datos tanto sintetizados como reales. Los resultados ablativos y comparativos muestran que el modelo SSDNet multitarea es capaz de mejorar eficazmente el rendimiento de las tareas de segmentación semántica y de afinación profunda en tiempo real. En este trabajo utilizaron la base de datos 'KITTI', con la que presentaron una precisión de 78.967 y una mIoU% de 46.041, mientras que al usar la base de datos 'CityScapes' presentaron una mIoU% de 86 [22].

En otra investigación en la que se presentaron avances importantes para la percepción ambiental en 2020, se planteó un método eficaz de detección de obstáculos que fusiona las características de un 3D-LIDAR y un sensor de visión para garantizar que se complementen entre sí. En este trabajo, idearon y propusieron una tecnología de ajuste del ángulo de visión activo para resolver el problema de velocidad de procesamiento causado por el uso de varios sensores, verificaron la superioridad en comparación con otros estudios sobre detección de objetos, aumentando la velocidad de

procesamiento aproximadamente entre un 30% y un 40%. Además, en el mismo trabajo obtuvieron una tasa de detección de peatones resultante del 56.2%, mientras que la tasa de detección de vehículos lograron fijarla en un 78.3% y todo mientras utilizaron solo el 81% del sistema de GPU [23].

Una última investigación en percepción ambiental del 2020, que representó la principal guía para iniciar esta investigación, es el artículo [24], en donde se propuso un método de percepción ambiental que se puede utilizar para reconocer tanto las áreas en las que se puede conducir, como los obstáculos. Este método, cuya estructura es mostrada en la figura 2.3, es denominado red de detección y segmentación simultánea o SSADNet por sus siglas en inglés. El método está basado en el aprendizaje profundo y utiliza datos de nube de puntos de detección y rangos de luz. A diferencia de otros métodos, SSADNet puede realizar la segmentación y la detección de forma simultánea basándose en una única red neuronal. El rendimiento de segmentación de la red propuesta tenía una precisión de clasificación de 96,9%, el mAP fue de 96,9%, mientras que el rendimiento de detección tuvo un mIoU de 70,5% y un mIoU de 79,0% con distancias de 0.5 y 0.7 respectivamente. La velocidad de inferencia fue de 33 FPS.

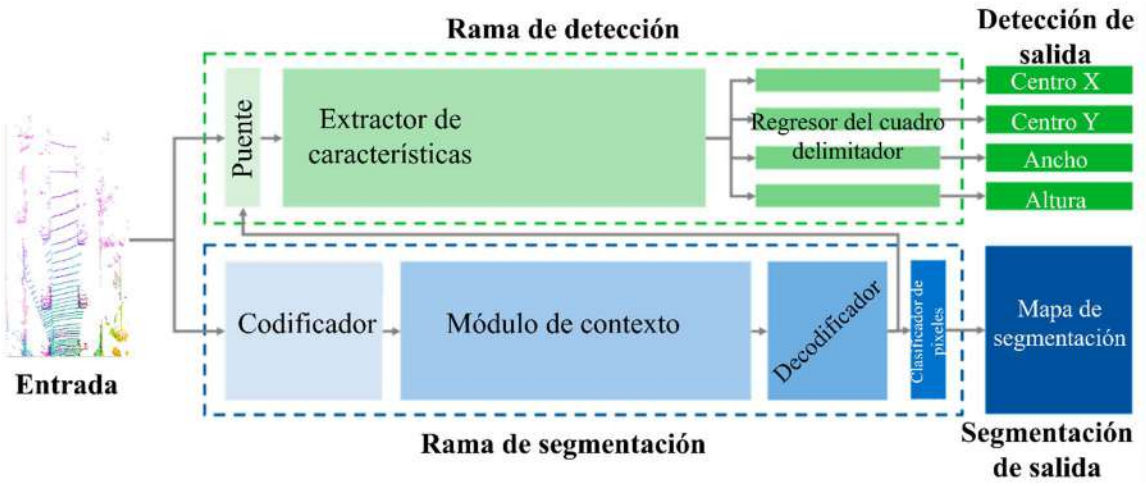


Figura 2.3 Estructura de una SSADNet, adaptada de [24].

2.6 Metodología de percepción ambiental

De acuerdo con [5], la metodología actual a seguir para el desarrollo de la percepción en un vehículo autónomo es la mostrada en la Tabla 2.1, que propone las técnicas de SSADNet y RANSAC para la resolución de esta tarea.

Tabla 2.1 Resumen de algoritmos de percepción, tabla adaptada de [5].

Tarea del auto autónomo	Nombre del algoritmo	Detalles del algoritmo	Ventaja	Limitaciones
Percepción ambiental	Red de segmentación y detección simultanea (SSADNet)	1. Es una red de aprendizaje profundo 2. Comprime dos redes separas, las cuales incluyen las ramas de segmentación y detección	1. Puede distinguir ambos, las regiones para manejar y los alrededores 2. Logra simultáneamente la segmentación y el reconocimiento de la posición de los objetos	La precisión es menor y no siempre puede otorgar la salida óptima
	Consenso de muestra aleatoria (RANSAC)	Es una herramienta de modelado predictivo ampliamente usada en el procesamiento de imágenes	1. Puede ser usado para quitar ruido de base de datos 2. Usa la base de datos más pequeña posible	Requiere muchas iteraciones y repeticiones de la construcción del modelo para obtener el mejor modelo

2.7 Investigaciones con conceptos relevantes

Durante este trabajo de investigación, se utilizan conceptos especializados. El primero de ellos es el sensor LIDAR, utilizado en las investigaciones, que tienen como propósito aportar conocimiento en el área de vehículos autónomos, como principal fuente de adquisición de datos para generar las o la entrada para un modelo de inteligencia artificial y por ende se utiliza como base para la representación de resultados, ejemplos de esto lo podemos encontrar en [25], [26] y [27]. El segundo concepto relevante, es el uso del término vista aérea o Top-View en inglés, utilizado en trabajos relacionados con la representación de escenas a partir de nubes de puntos para que generar mapas de segmentación y así, ofrecer una manera de analizar las formas y mediciones a partir de otro ángulo. Esta variación permitió facilitar los análisis al colocar la información tridimensional, en un plano de dos dimensiones que forma el plano donde los vehículos se desplazan, evidencia de estos se encuentra en los artículos [28], [29] y [30]. Un tercer y último concepto relevante a destacar, es la base de datos KITTI Vision Benchmark Suite disponible en [31]. Este concepto, como su nombre lo indica, hace referencia a una base de datos con escenas adquiridas a partir de múltiples sensores y capturada alrededor de ciudades

medianas, áreas rurales y autopistas. Su información, fue capturada a partir de sistemas de adquisición de datos de tipo cámaras, escáneres LiDAR, un par de cámaras RGB que forman un sistema de estéreo cámara, sensor de flujo óptimo, sensor de edometría visual y detector de objetos tridimensionales. Esta investigación fue publicada en [32], liberando la base de datos creada solo para fines educativos.

3. Metodología propuesta

En la figura 3.1, se destacan las etapas clave de la metodología propuesta. La primera fase, selección de componentes base, es la etapa donde se selecciona y se investigan a fondo los instrumentos de hardware y todo lo relacionado con el software. Posteriormente, se tiene la etapa de generación de base de datos, donde se les realiza todo el tratamiento necesario a los datos de origen para que sean aprovechados como insumo para una red neuronal. La tercera etapa, desarrollo del modelo, es la sección del método donde se genera el modelo de inteligencia artificial que arroja las inferencias de percepción ambiental. Luego, se encuentra el cuarto punto donde utilizamos el modelo de percepción ambiental generado para desarrollar el sistema de percepción ambiental que analiza el ambiente continuamente. Finalmente, en la fase 5 se encuentran todas las instrucciones y recomendaciones para el momento de montar el sistema de percepción ambiental en un vehículo.

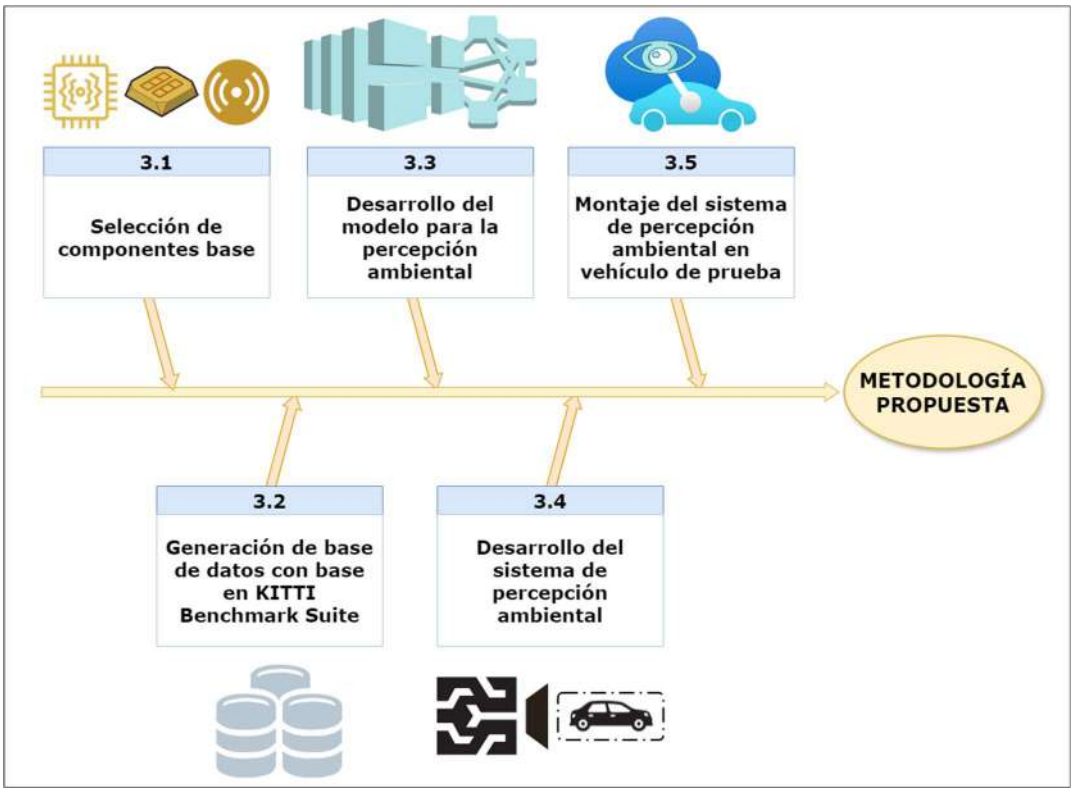


Figura 3.1 Diagrama de las secciones principales que componen a la metodología propuesta.

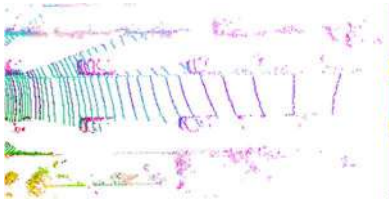
3.1 Selección de componentes base

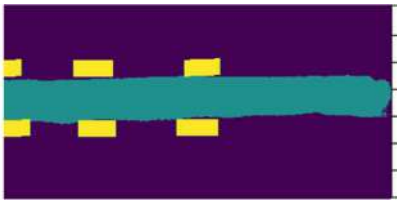
3.1.1 Selección de base de datos

Existen diversas bases de datos creadas para el trabajo con vehículos autónomos que se pueden escoger para desarrollar la percepción ambiental en un automóvil. Para el caso de esta propuesta metodológica, es vital que la base de datos seleccionada cuente con tres tipos archivos (Fig. 3.2):

- Binarios de nube de puntos o de imágenes aéreas (por su nombre en inglés), de la nube de puntos de cada escena.
- Del mapa de segmentación de cada escena.
- De etiquetas de detección en el espacio tridimensional o directamente de forma bidimensional sobre la imagen aérea de cada escena.

Los cuales contengan la misma cantidad de escenas.





Información de interés	Ejemplo I	Ejemplo II
Clase	Car	Car
Centro x	0	6
Centro y	65	126
Ancho	35.9	42.8
Largo	16.1	16.5
Ángulo de rotación	3.1008	3.1008

a) b) c)

Figura 3.2 Ejemplos de los tres tipos de archivo que se requieren en la base de datos seleccionada. a) Imagen RGB de vista de agila de una sección de calle pública, capturada por un sensor LiDAR en formato de nube de puntos. b) Mapa de segmentación de la imagen a), segmentando los autos y el camino de la escena. c) Tabla con ejemplos de los datos de la posición de vehículos dentro de una escena.

En caso de que cuente con los elementos básicos como archivos de nubes de puntos y/o etiquetas tridimensionales, será necesario transformar estos a los tipos de archivos que se requieren durante el entrenamiento. Si la base de datos no cuenta con mapas de segmentación, será posible generarlos a partir de las imágenes aéreas y etiquetas de detección. Todo esto, siguiendo las propuestas de la sección 3.2, que fue utilizada para la creación de la base de datos del artículo [33].

Ahora bien, cada escena puede contener múltiples clases de objetos por identificar y a su vez, varios objetos de cada clase. Sin embargo, se debe tomar en cuenta que entre más clases se plantee identificar, más complejo será tener una identificación correcta. Además, tanto un aumento de la cantidad de objetos a detectar, como un incremento en la cantidad de clases a identificar en cada

escena, representa un crecimiento en los requerimientos mínimos de capacidad de hardware durante la ejecución del software en un sistema embebido y durante el entrenamiento en un computador. De la misma forma, se debe buscar contar con tantas escenas como sea posible, ya que esto ayudará a mejorar las métricas de evaluación al evitar el sobre entrenamiento, pero se debe tomar en cuenta que una mayor cantidad de estas puede significar un aumento en los requerimientos de hardware.

Es necesario mencionar que existen diferentes tipos de formatos de anotaciones, sin embargo, en este escrito se propone una manera de extraer solo la información necesaria de cualquier formato para agruparla tal como se requiere durante el entrenamiento.

Una vez elegida la base de datos, es necesario considerar la siguiente información:

- Es necesario validar que se cuente con escenas que incluyan todas las clases que se buscan, tratando de que se tenga un número aproximadamente equilibrado de objetos a detectar de cada clase en toda la base de datos. Además, se debe considerar que existan escenas donde no se encuentre ni un solo objeto de las clases que se buscan.
- Se debe tomar en cuenta que el número total de escenas se dividirá en muestras de entrenamiento, prueba y validación dentro del total etiquetados en la base de datos.

3.1.2 Selección del modelo de Inteligencia Artificial

Durante este proceso, se llegó a la selección del modelo SSADNet como base para el modelo final para obtener la información de la posición y dimensión de cada uno de los objetos de interés mediante la detección y su clase a través de la segmentación. Sin embargo, es posible remplazarlo fácilmente por cualquier otro, tomando la sección de “desarrollo del modelo” como referencia durante su construcción, siempre y cuando cumpla con los siguientes puntos:

- Debe ser un modelo que tome imágenes de nubes de puntos como entrada.
- Se busca que, a la salida del modelo se obtengan las distancias (centrales o algún punto de referencia) de la posición y dimensión de cada uno de los objetos de interés, así como su clase.
- La velocidad de inferencia durante la predicción de una sola imagen sea tan alta como sea posible, tomando como referencia que la velocidad que se busca para todo el proceso, es decir, adquirir los datos, transformarlos al tipo de entrada del modelo e inferir los datos de salida de una sola escena sea posible en 45.4545 milisegundos, para poder analizar 22 escenas por segundo.

3.1.3 Selección de los dispositivos de cómputo especializados

Entrenamiento

En particular para entrenar el modelo SSADNet con 14,962 grupos de imagen, etiqueta y mapa de segmentación con un tamaño de lote igual a ocho, se ha usado una laptop con *procesador Intel CORE i7*, 32 Gigabytes de memoria RAM, y una *GPU RTX 3070*. El tiempo por ciclo de entrenamiento o época es de poco menor a 20 minutos con estos valores. Los componentes mencionados son los más importantes de tomar en cuenta al momento de seleccionar el computador con el cual se realizará el entrenamiento, los cuales pueden variar afectando el rendimiento de la siguiente manera:

- Procesador: Afecta directamente en los tiempos de entrenamiento, debido a que no todos los datos pueden ser procesados por el hardware acelerador GPU, sin embargo, el nivel de cambio entre procesadores no es muy grande durante un entrenamiento.
- Memoria RAM: De esta memoria depende la cantidad de muestras de la base de datos que pueden ser utilizados durante la ejecución de un programa en el cual se planea un entrenamiento. Se recomienda ampliamente buscar un equipo con 32 o más Gigabytes de esta memoria para evitar problemas con el uso de datos durante la ejecución de un programa de entrenamiento y el procesamiento general del computador.
- GPU: Este es el principal componente del cual dependerá el tiempo de entrenamiento en un equipo, siempre y cuando este configurado el sistema operativo y el programa para ejecutarse con este hardware acelerador. Por lo general, este tenderá a ser más potente y por ende más veloz entre más reciente sea el hardware o mayor sea el número de identificador. Es necesario destacar que las GPU, a pesar de ser del mismo número o serie, las diseñadas para laptop son menos *potentes* que las creadas para CPU de escritorio.

Implementación

Existen varios hardware compatibles con librerías de TensorFlow [34] y *pytorch*, las cuales son utilizadas durante la creación de modelos de inteligencia artificial. Sin embargo, por la complejidad del tema tratado, se recomienda ampliamente utilizar hardware *Nvidia* durante la implementación, o con características superiores, lo cual asegura que los componentes internos de estos sistemas soportaran la ejecución del programa final de percepción. Como referencia, al utilizar el hardware Jetson Nano la velocidad de ejecución del programa final de la sección 3.4, ascendía sobre los 5 minutos 40 segundos por ejecución, optando por utilizar una *Jetson Orin Nano* con un rendimiento 10 veces mayor que una *Jetson Nano*.

Además del hardware embebido, es necesario considerar la batería con la cual se estará alimentando el sistema de percepción. De forma general, esto dependerá del voltaje y amperaje que el embebido requiere como máximo, para asegurar el funcionamiento sin importar las necesidades energéticas. En el caso particular del hardware seleccionado en esta sección para la implementación, se recomienda el uso de una batería recargable con salida de tipo Jack de exactamente 12.x volts de corriente directa y que cuente con mínimo 5 amperes.

3.1.4 Selección de sensores

Todo el sistema de percepción propuesto está basado en entradas de nubes de puntos, datos que se adquieren a través de sensores LiDAR tridimensionales. Estos sensores se pueden encontrar en diversas marcas y de diferentes modelos, sin embargo, son de un alto costo, por lo que se recomienda en primera instancia utilizar el sensor más económico accesible en el mercado para verificar el funcionamiento del sistema de percepción desarrollado, para posteriormente buscar una posible mejora en la calidad del sensor y con esto en la exactitud de los resultados. Los requerimientos básicos para tomar en cuenta durante la elección de este sensor son los siguientes:

- Debe asegurarse una adquisición de datos tridimensional, es decir, que se generen puntos a largo, ancho y alto, ya que existen sensores LiDAR de dos dimensiones.
- Los rangos de adquisición de datos dependen de la base de datos utilizada, ya que se deben encontrar los máximos en cada dimensión que esta considera al momento de generar las nubes de puntos o imágenes de prueba. Existen rangos de longitud a lo largo y ancho al ser sensores de giro de 360 grados, mientras que por lo general el rango de altura depende del ángulo de visión seleccionado.

  En caso de buscar realizar una base de datos propia, se pueden decidir los valores de los rangos en cada dimensión siguiendo las instrucciones de la hoja de datos del sensor, en la cual se marcan los rangos en que el sensor funciona según la precisión que se busca.

  En el caso específico de la implementación probada, se tienen los rangos $X_{min} = 6, X_{max} = 46, Y_{min} = -10, Y_{max} = 10, Z_{min} = -2.5, Z_{max} = 2.5$. Los valores de intensidad i de cada punto, se encuentran en rango entre 0 y 1.
- La cantidad de puntos que el sensor es capaz de generar, de esto depende la exactitud de las predicciones nuevas que se realicen. De la misma manera que con las GPU, entre mayor número de puntos mejores resultados se tendrán, sin embargo, es posible trabajar con el sensor que menos puntos genere por temas económicos, para posteriormente buscar una mejora en calidad.

Como base, se recomienda tomar en cuenta el sensor con el que se realizó la base de datos buscando utilizar el mismo o el más parecido posible en cuanto a recursos, buscando asegurar nubes de puntos similares con las cuales se entrenó el modelo de inteligencia artificial.

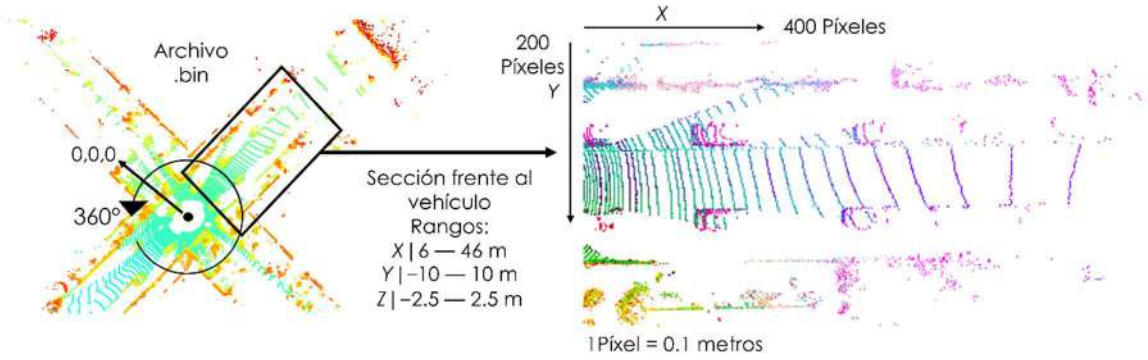
En particular, durante el desarrollo de la propuesta, se utilizó el sensor *LiDAR Pandar Hesai XT-16*.

3.2 Generación de base de datos con base en KITTI Benchmark Suite

En esta sección se presentarán los pasos metodológicos necesarios para la creación de la base de datos propuesta. En algunos pasos se explicará el procedimiento de forma breve a través de un diagrama de flujo del proceso y en otros a través de explicaciones básicas, en caso de necesitar una explicación más a fondo sobre cada sección; se sugiere analizar el artículo [33], donde se hace la primicia de este procedimiento. En Anexo se encuentra la primera hoja de ese artículo.

3.2.1 Generación de la visión superior a partir de los datos de nube de puntos Velodyne

El objetivo de esta sección es tomar los archivos binarios de nubes puntos de cada escena de la base de datos original para transformarlos a las imágenes de vista aérea bidimensionales finales (Figura 3.3), las cuales representan el primer tipo de archivo de la base de datos propuesta.



a) b)

Figura 3.3 Ejemplo de escena de nube de puntos con la sección a generar como vista aérea aproximada destacada en a) y su respectiva escena de vista aérea final en b) donde cada píxel corresponde a 0.1 metros, destacando los rangos en los que los puntos dentro de estos son tomados en cuenta.

Como propuesta para el desarrollo de esta sección del método, se presenta el diagrama de flujo para transformar la nube de puntos en una imagen de vista superior, Fig. 3.4.

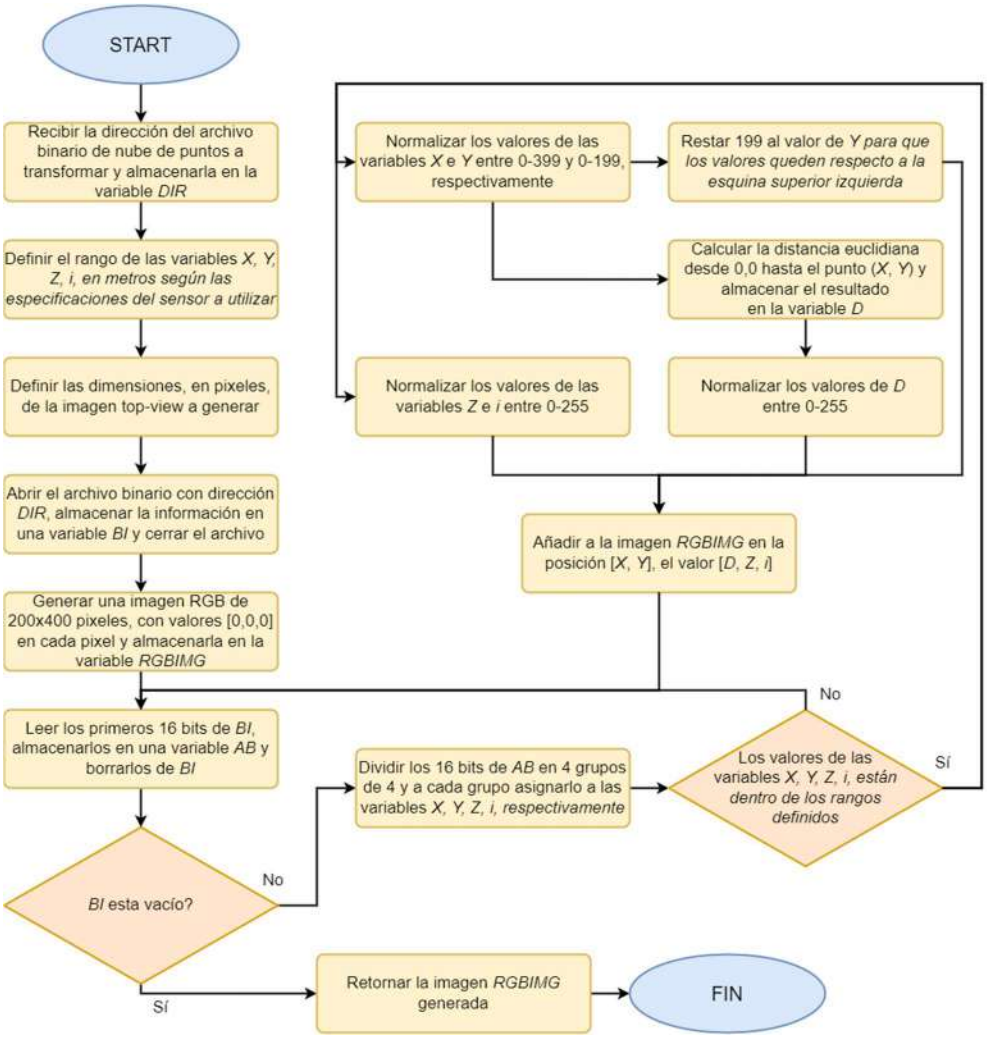


Figura 3.4 Diagrama de flujo para transformar un archivo de nube de puntos binarios en una imagen de vista superior.

3.2.2 Extracción y acomodo de la información de detección a partir de las etiquetas de entrenamiento

En esta subsección, se busca tomar las etiquetas que proporciona la base de datos KITTI para seleccionar solo la información de interés (Figura 3.5); en este caso, la necesaria para ubicar tanto el centro bidimensional de los objetos de interés en la imagen de vista superior, como su cuadro delimitador. Con esto se busca generar los archivos de etiquetas que representan el segundo tipo de archivo disponible en la base de datos propuesta.

Al igual que en la sección anterior se presenta un diagrama de flujo para llevar a cabo el proceso de este paso metodológico, Figura 3.6.

Información disponible original	Ejemplo I	Ejemplo II
Clase	Car	Cyclist
Truncamiento	0.96	0.00
Oclusión	0	1
Alpha	−0.86	−1.60
Cuadro delimitador xmin	0.00	991.70
Cuadro delimitador ymin	201.30	147.32
Cuadro delimitador xmax	303.88	1029.63
Cuadro delimitador ymax	369.00	217.27
Altura	1.50	1.72
Ancho	1.78	0.78
Longitud	3.69	1.71
Centro x	−3.16	10.48
Centro y	1.68	0.90
Centro z	3.35	18.35
Ángulo de rotación	−1.56	−1.08

Información de interés	Ejemplo I	Ejemplo II
Clase	Car	Car
Centro x	0	6
Centro y	65	126
Ancho	35.9	42.8
Alto	16.1	16.5
Ángulo de rotación	3.1008	3.1008

Figura 3.5 Ejemplo de etiqueta original de la base de datos KITTI [31] a la izquierda y su respectiva etiqueta ya reducida con los objetos y valores de interés.

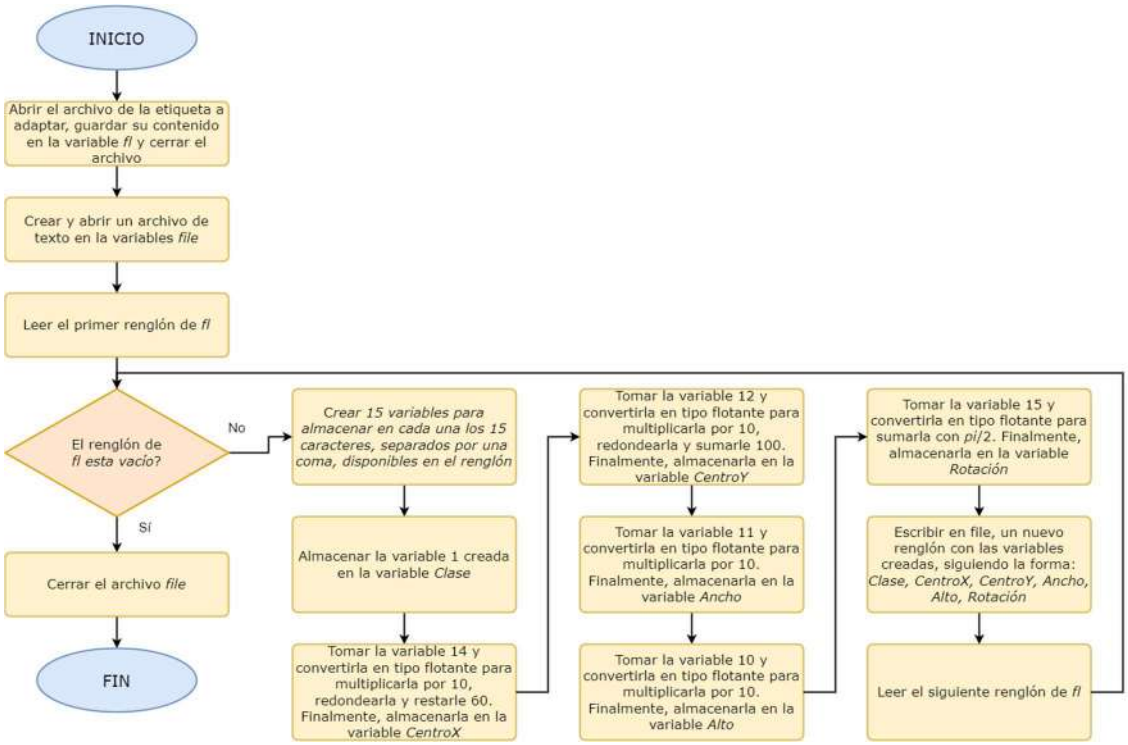


Figura 3.6 Diagrama de flujo para generar, a partir de un archivo de las etiquetas originales, un nuevo archivo con las variables de los objetos de interés.

3.2.3 Segmentación manual del camino a partir de las imágenes de visión superior

Primeramente, mediante el análisis de cada una de las escenas de las imágenes top-view generadas, se seleccionan 100 imágenes con el camino de un solo carril notablemente marcado, otras 100 con el camino de varios carriles notablemente marcados y unas últimas 100 con el camino no marcado.

En esta subsección, se utilizan las imágenes seleccionadas para, mediante el software en línea Make Sense [35], segmentar manualmente el camino, tomando como guía la figura del camino fácilmente observable en algunos casos y en otros, donde no es tan claro, buscar el camino mediante la comparación de las imágenes de visión aérea; con las imágenes frontales tomadas con cámara proporcionadas como parte de la base de datos KITTI [31]. La segmentación que se realiza con Make Sense, se lleva a cabo a través de múltiples puntos alrededor de cada figura considera de una clase en particular, sin embargo, es necesario tomar en cuenta que para este caso en especial, solo es necesario crear dos tipos de clases, alrededores y camino, siendo necesario solamente segmentar el camino de manera exacta, mientras que en el caso de los alrededores; solamente con generar un recuadro en alguna parte fuera de la sección considerada camino, servirá para generar el mapa segmentado de cada escena como verdad base.

El resultado de la segmentación con el software propuesto se recomienda obtener como un archivo en formato COCO, para posteriormente generar los mapas de segmentación, Figura 3.7, es decir, una imagen con las mismas dimensiones que la imagen segmentada manualmente, en la cual, a cada píxel se le asigna un valor entero acorde a la clase que corresponde, siendo en este caso, 0 o 1.

Figura 3.7 Ejemplo de mapa de segmentación del camino.

3.2.4 Creación de los datos de entrenamiento y validación para la segmentación del camino

Se utilizan cada uno de los mapas de segmentación creados recién en la subsección pasada y las imágenes de vista aérea de la primera subsección, para realizar un aumento de datos, que permitirá entrenar un modelo en miras de realizar un mapa de segmentación de cada una de las escenas de vista aérea generadas.

El aumento de datos de ambos tipos de escena se realiza siguiendo las instrucciones de la Figura 3.8.

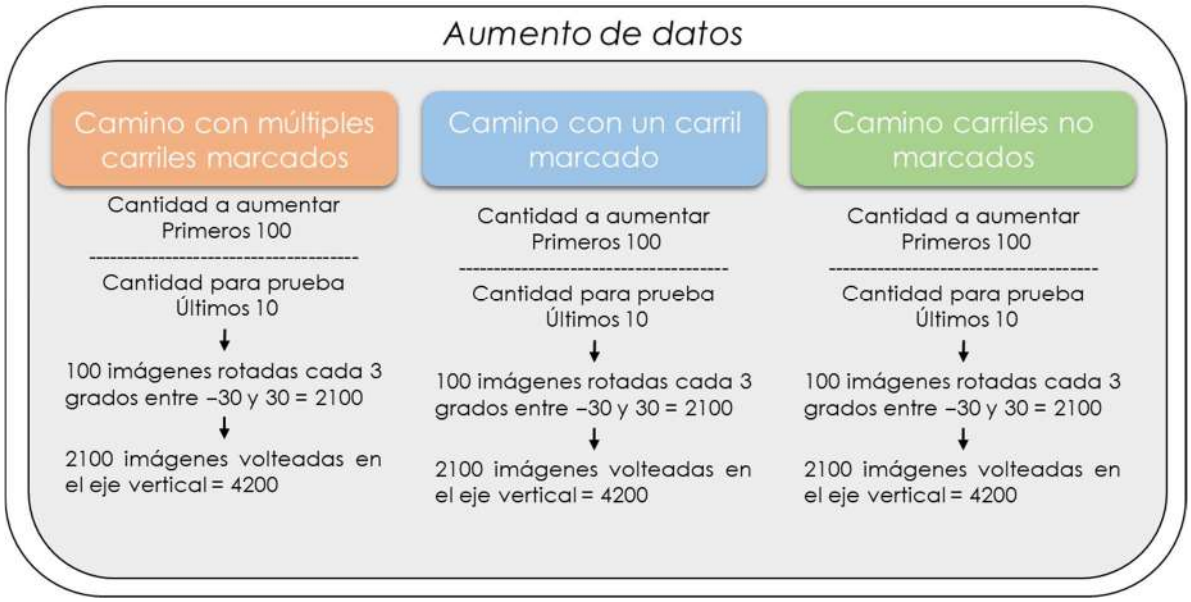


Figura 3.8 Descripción general del procedimiento para el aumento de datos, destacando que se deben separar datos de prueba desde antes del procedimiento.

Al finalizar el procedimiento del aumento de datos, se separan los números de escena de acuerdo con lo descrito en la Figura 3.9, tomando en conjunto tanto a la imagen de vista aérea como el mapa segmentado del mismo número de escena.

Como fase final de esta subsección, se generan cinco bases de datos de entrenamiento con diferentes números de escena para el entrenamiento y la validación, en miras de generar las bases de datos para una validación *5-fold*.

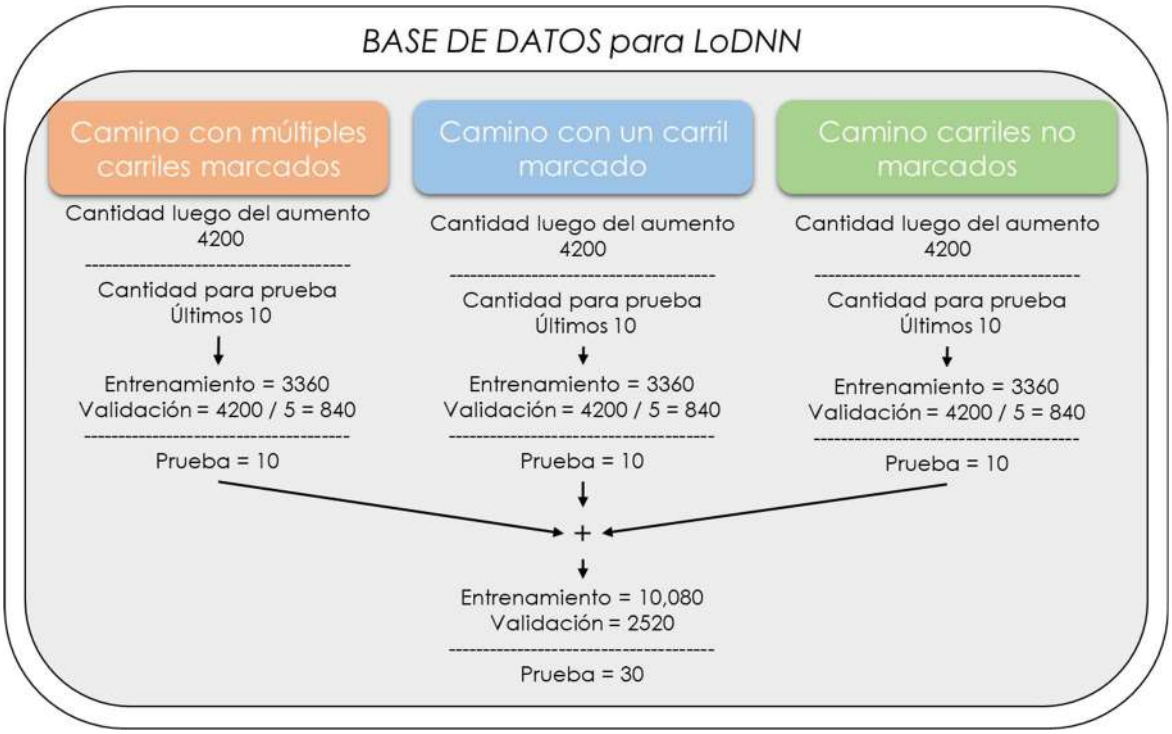


Figura 3.9 Descripción de la división de escenas generadas, en miras de la creación de datos de entrenamiento, validación y prueba.

3.2.5 Construcción del modelo LoDNN

Una vez generada la base de datos, es posible pasar a la generación de la estructura de un modelo de inteligencia artificial, modelo que una vez entrenado será capaz de inferir el mapa de segmentación del camino de cada escena de vista aérea generada en la subsección uno. La estructura sugerida es la del modelo LoDNN propuesto en [36], cuya estructura exacta de implementación con hiperparámetros se presenta en la Figura 3.10 con Tensor Flow y Keras. Es necesario resaltar que las figuras relacionadas con estructuras de modelos de inteligencia artificial contienen texto en inglés que hacen referencia al nombre exacto con el cual se encuentran en el software de Python, con el objetivo de facilitar la réplica de la construcción del modelo.

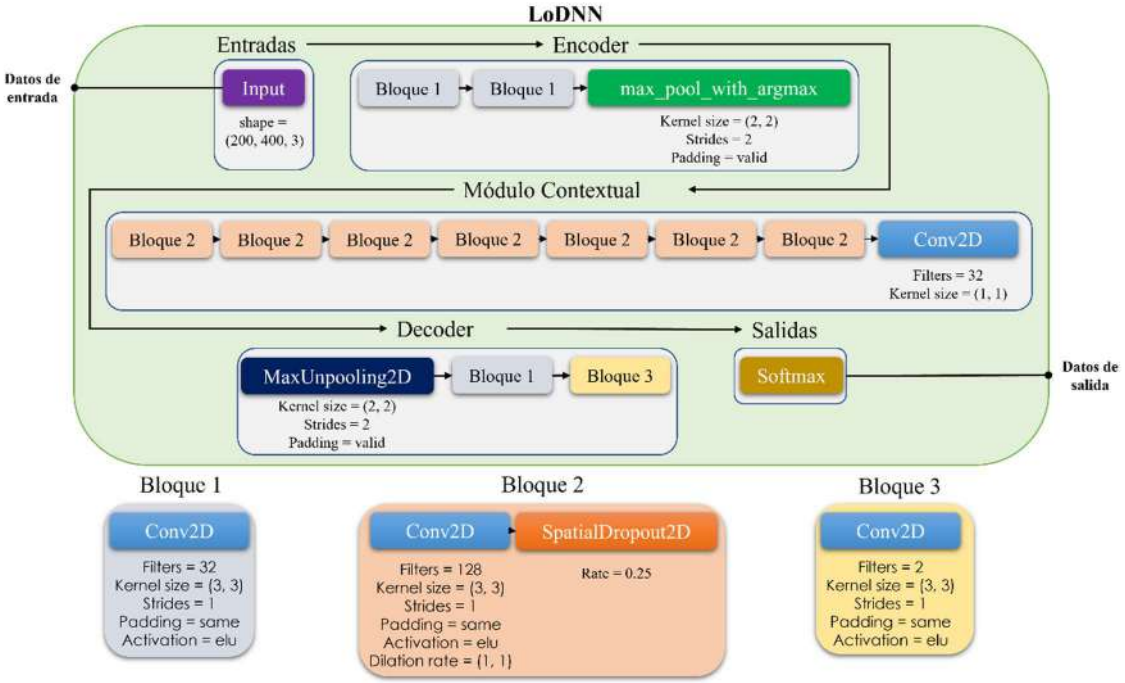


Figura 3.10 Descripción gráfica de la estructura del modelo LoDNN, con cada una de las capas en el correcto orden y sus correspondientes hiperparámetros.

3.2.6 Entrenamiento de la red neuronal para segmentación del camino y ajustes para optimización

Además de estructurar el modelo, es necesario generarlo, compilarlo y entrenarlo, considerando que para cada una de esas acciones es necesario tener bien definidos los hiperparámetros que necesitan. Como guía se presenta en la Figura 3.11 la recopilación de las instrucciones para cada acción y sus hiperparámetros. Es necesario aclarar que los datos guardados en '*x*' son las matrices de las imágenes aéreas normalizadas entre 0 y 1, mientras que los guardados en '*y*' son las matrices de los mapas de segmentación de cada escena de '*x*'.

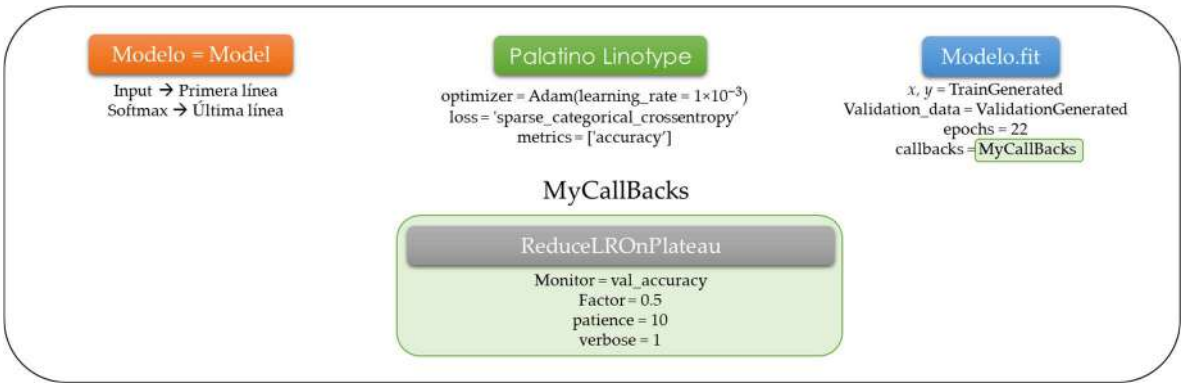


Figura 3.11 Descripción gráfica de los comandos necesarios para generar, preparar y entrenar el modelo, así como sus correspondientes hiperparámetros. Se considera que se le coloca el nombre de 'Modelo' a la red neuronal generada.

3.2.7 Métricas de rendimiento del aprendizaje máquina

Matriz de confusión binaria. Existen cuatro diferentes tipos de resultados que la matriz de confusión otorga, los cuales se representan y agrupan como se muestra en la Tabla 2.2. Cada uno de esos resultados se definen como:

1. Verdadero positivo (TP)

   Se espera que tenga in valor positivo de su característica y se obtiene un valor positivo también.
2. Verdadero negativo (TN)

   Se espera un valor negativo de su característica y se predice un valor negativo también.
3. Falso positivo (FP)

   Se tiene un valor negativo a pesar de haber predicho un valor positivo.
4. Falso negativo (FN)

   Se tiene un valor positivo a pesar de haber predicho un valor negativo.

Tasa de clasificación. Es la manera más simple de evaluar un modelo con características nominales y discretas, el método para su cálculo es la siguiente:

$$Tasa\ de\ clasificación = 1 - \frac{Clasificaciones\ incorrectas}{Total\ de\ predicciones\ realizadas} \quad (3.1)$$

Exactitud. Es el cálculo de la tasa de clasificación, pero calculado a partir de la matriz de confusión. En inglés conocida como 'Accuracy' (ACC). Se puede utilizar para calcular la exactitud de un modelo utilizando la siguiente formulación:

$$Exactitud = \frac{TP+T}{TP+TN+FP+F} \tag{3.2}$$

Precisión. Se puede definir como la tasa de las muestras predichas que son relevantes, en inglés conocida como 'Precision' (PRE) y se calcula como sigue:

$$Precisión = \frac{TP}{TP} \tag{3.3}$$

Sensitividad. La sensitividad, también llamado 'recall' (REC), se puede definir como la tasa de las muestras seleccionadas que son relevantes a la prueba, es posible calcular utilizando la siguiente formulación:

$$Sensitividad = \frac{TP}{TP+FN} \tag{3.4}$$

Puntaje F1. El puntaje-F1, F1-Score en inglés, puntaje F-beta con una beta de 1, se puede definir como la media harmónica entre sensitividad y precisión, y se calcula como se muestra en la ecuación 2.5 [37].

$$Puntaje\text{–}F1 = \frac{2\times TP}{2\times TP+FP+FN} \tag{3.5}$$

Tabla 3.1 Descripción gráfica de una matriz de confusión [37].

		Predicción	
		Positivos	**Negativos**
Valor esperado	**Positivos**	Verdaderos Positivos (VP)	Falsos Negativos (FN)
	Negativos	Falsos Positivos (FP)	Verdaderos Negativos (VN)

3.2.8 Generación de la base de datos con camino segmentado a partir de predicciones del modelo

Se utiliza el modelo entrenado resultante de la subsección anterior para inferir el mapa de segmentación de cada una de las escenas de vista aérea generadas en la subsección 3.2.1, Figura 3.12. Cada una de las matrices de los mapas inferidos se almacenan en una nueva variable que en este caso llamaremos 'Escenas con camino segmentado'.

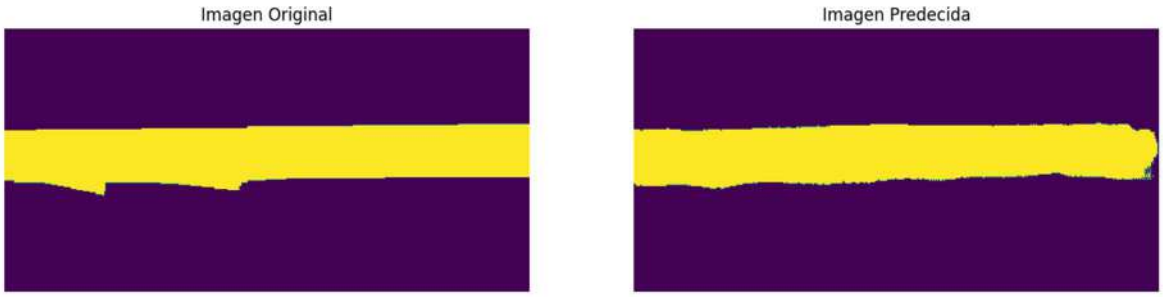


Figura 3.12 Ejemplo de mapa de segmentación original e inferido con el modelo LoDNN con los hiperparámetros propuestos.

3.2.9 Creación de la base de datos final con el camino, vehículos y obstáculos detectados tomando como referencia la información de detección.

Como último paso, se procede a generar el último tipo de archivo de la base de datos propuesta, utilizando los mapas de segmentación generados en la sección previa para incluir los objetos de interés en cada escena, en este caso, alrededores, camino, carro, furgoneta y camioneta.

Para ello, se toma cada mapa de segmentación del camino y se añaden los objetos de interés presentes según el número de escena. Se considera el número porque se tendrán en cuenta las etiquetas generadas en la subsección 3.2.2, tanto para determinar el número de objetos a añadir en cada escena como para conocer la ubicación correspondiente del cuadro delimitador de cada objeto en la escena. Consideramos que cada píxel dentro de un cuadro delimitador corresponderá a una clase específica y, por tanto, el valor de cada uno de esos píxeles dentro del mapa de segmentación deberá cambiarse por el valor correspondiente a la clase del objeto. Los números de objeto propuestos son los siguientes: alrededores−0, camino−1, carro−2, furgoneta−3 y camioneta−4.

Al igual que en las subsecciones 3.2.1 y 3.2.2, se agrega el siguiente diagrama de flujo, Figura 3.13, como método para llevar a cabo este último paso de la creación de la base de datos propuesta.

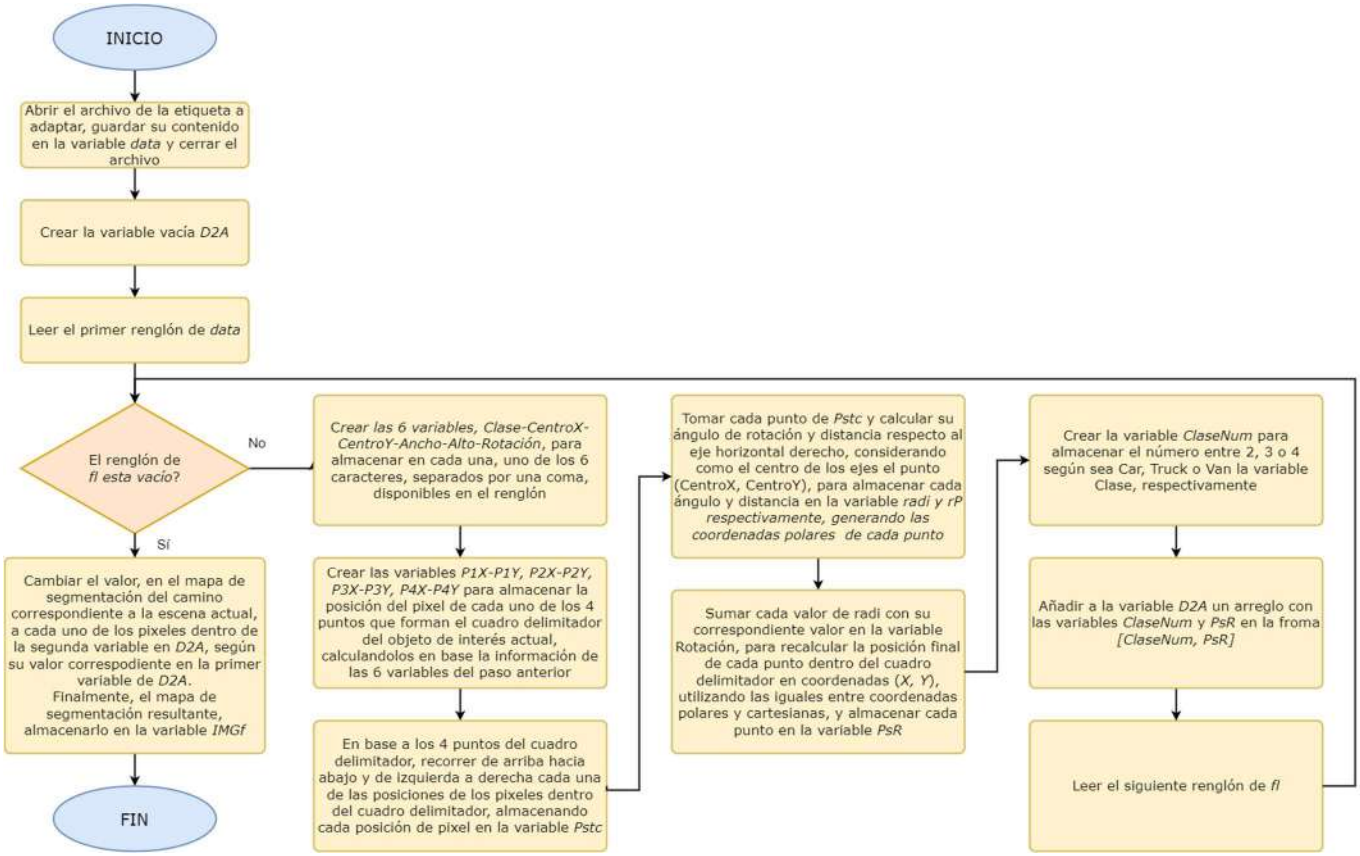


Figura 3.13 Diagrama de flujo para generar el mapa segmentado final a partir de los mapas del camino predichos.

3.3 Desarrollo del modelo para la percepción ambiental

En esta sección se propuso una serie de pasos con los cuales se busca generar un modelo entrenado capaz de predecir la posición y dimensión en el plano, así como la clase de cada uno de los objetos de interés en cualquier escena nueva que corresponda a una sección vial. Como referencia, se hace uso del modelo SSADNet y la base de datos generada en la sección pasada, buscando ejemplificar el proceso a través de información puesta a prueba de manera práctica.

3.3.1 Adaptación de la base de datos para su uso con el modelo

En primera instancia, es necesario transformar las etiquetas de la base de datos a un formato que coincida con las salidas esperadas del modelo a utilizar, en el caso particular del modelo SSADNet propuesto, las salidas serían seis: mapa de segmentación de la escena, posiciones en *X*, posiciones en *Y*, largos, anchos y ángulos de cada objeto identificado en la escena.

Analizando las etiquetas originales de la base de datos, por cada escena contamos con un mapa de segmentación y un archivo donde se encuentran todos los objetos de interés detectados y por cada objeto, los 5 datos relativos a su posición, dimensión y ángulo.

De acuerdo con los párrafos anteriores, se tienen que generar variables en las cuales se almacene un tipo diferente de conjunto de datos por variable y un total de conjuntos en cada variable equivalente al número de escenas disponibles. Los tipos de conjuntos de datos se identifican según cada una de las salidas esperadas. Es por lo anterior, que en el caso del modelo SSADNet y la base de datos propuesta, se generan seis variables con 7481 conjuntos de datos.

En el caso de decidir generar un aumento de datos, se sugiere simplemente invertir cada una de las 7481 escenas, invirtiendo las imágenes de vista aérea y los mapas de segmentación, mientras que en los archivos de etiquetas se modifica solamente el valor de la posición en *X* y su rotación siguiendo las fórmulas de fórmula 1. Lo anterior, generaría 14,962 escenas y por tanto, la misma cantidad de conjunto de datos en cada una de las seis variables.

$$Nueva\ posición = \ 399 - Posición\ X\ original \tag{3.6}$$

$$Nuevo\ ángulo = \ \pi - Ángulo\ original \tag{3.7}$$

Fórmula 1. Fórmulas para invertir los valores de la posición en X y el ángulo de rotación en las etiquetas durante un aumento de datos.

Finalmente, se generan simplemente dos variables denominadas '*X*' e '*Y*', en las cuales se almacenarán las imágenes de cada escena de vista aérea en '*X*' y las seis variables creadas en '*Y*' en el orden en que se esperan a la salida del modelo de inteligencia artificial, en el caso particular del SSADNet se esperan de la siguiente manera: posiciones en *X*, posiciones en *Y*, largos, anchos y ángulos.

3.3.2 Separación de la base de datos en conjuntos de datos

Este paso se propuso con la intención de separar la cantidad de escenas disponibles en '*X*' e '*Y*' en datos de entrenamiento y prueba, mientras que, a su vez los datos de entrenamiento, se busca dividirlos en datos de entrenamiento y validación, de tal forma que permitan generar cinco grupos con distintos datos de validación. Lo anterior, en miras de generar un análisis *K-fold de K=5* con los datos de entrenamiento y validar los resultados con datos de prueba no analizados previamente por el modelo, lo cual asegura que sus métricas de rendimiento serán más generalizadas y apegadas a las futuras nuevas escenas capturadas para su inferencia. La mejor forma de describir la separación propuesta para la base de datos es mediante la siguiente imagen (Figura 3.14):

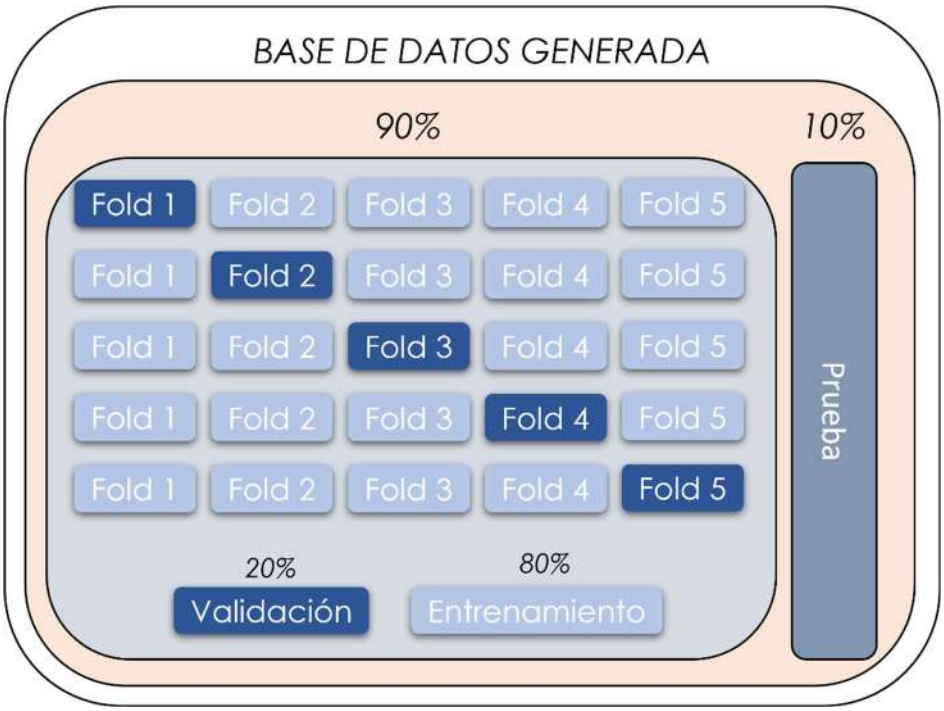


Figura 3.14 Divisiones propuestas para la base de datos a utilizar durante el entrenamiento de un modelo de inteligencia artificial.

Siguiendo la división propuesta, es que se generan las siguientes variables:

- $X-Entrenamiento$, $X-Validación$, $X-Prueba$
- $Y-Entrenamiento$, $Y-Validación$, $Y-Prueba$

Destacando que los datos en las variables de entrenamiento y validación variarán según el número K de análisis en el que se encuentre.

3.3.3 Construcción del modelo

En esta etapa, se propuso la construcción del modelo de inteligencia artificial en el lenguaje Python y a través de las librerías de TensorFlow [34] y Keras [38], especificando capa por capa cómo se estructura el modelo, incluyendo incluso los hiperparámetros de cada una de las capas. Lo anterior, con el propósito tanto de aclarar exactamente cómo utilizar cada función (capa) de red neuronal, como de permitir crear de una forma más intuitiva y sencilla la estructura de la red, lo que permite incluso modificar la estructura al añadir o quitar nuevas capas en búsqueda de mejores métricas de resultados de entrenamiento en futuros trabajos.

El modelo seleccionado es conocido como SSADNet [24], el cual toma una escena de nube de puntos, representada en formato aéreo como imagen RGB, como entrada y arroja a la salida del mapa de segmentación de la escena y los datos de detección (Centro X, Centro Y, Ancho, Alto y Rotación del rectángulo delimitador) de cada uno de los vehículos dentro de la escena. Para lograr lo anterior, en su interior procesa los datos de entrada a través de dos modelos distintos denominándolos rama de segmentación y rama de detección, Figura 3.15.

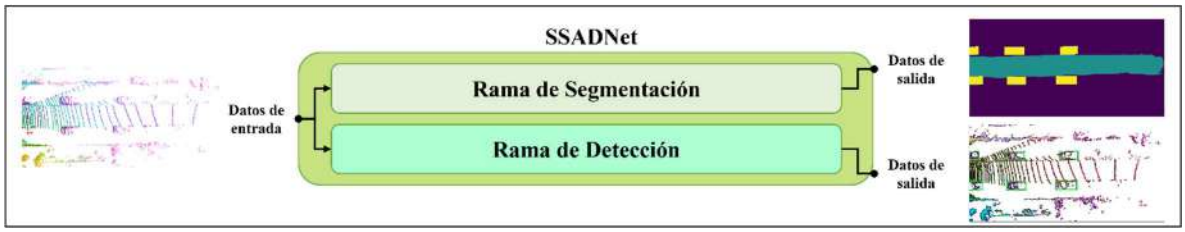


Figura 3.15 Representación simplificada del modelo SSADNet propuesto en [24].

Dentro de cada una de las ramas dentro del modelo SSADNet, existen bloques que pueden ser analizados a profundidad en [24], los cuales son tomados en cuenta durante la representación capa

por capa de las figuras 3.16, 3.17 y 3.18, donde se muestran las estructuras de cada una de las ramas, así como sus subconjuntos de capas.

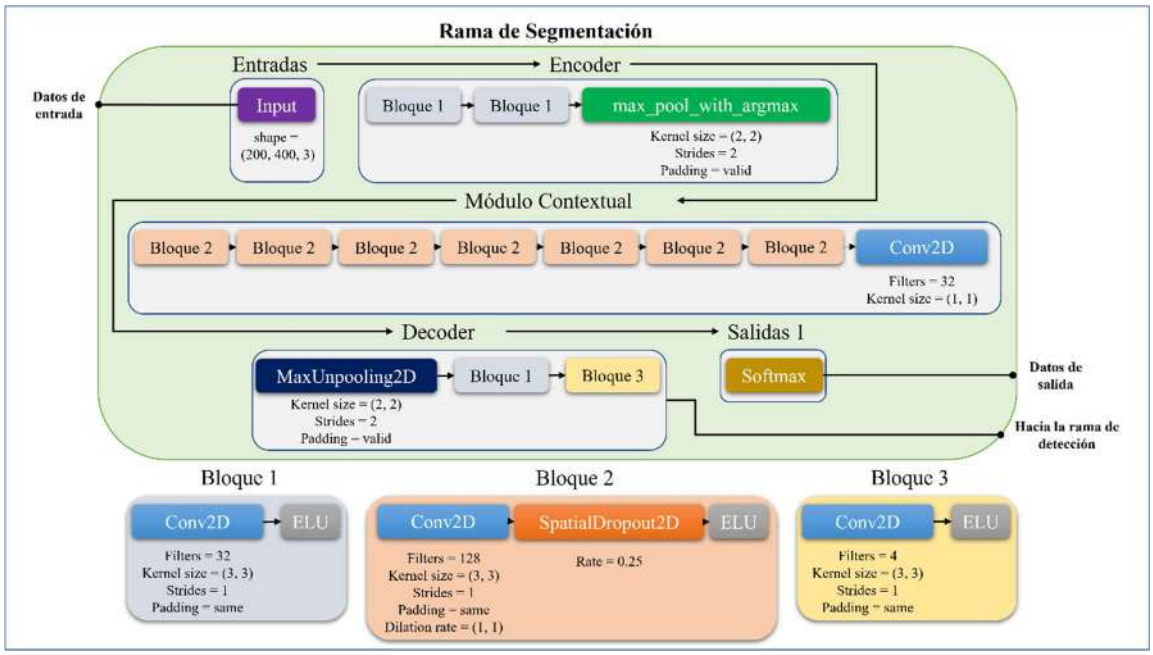


Figura 3.16 Representación capa por capa de la rama de segmentación del modelo SSADNet propuesto en [24].

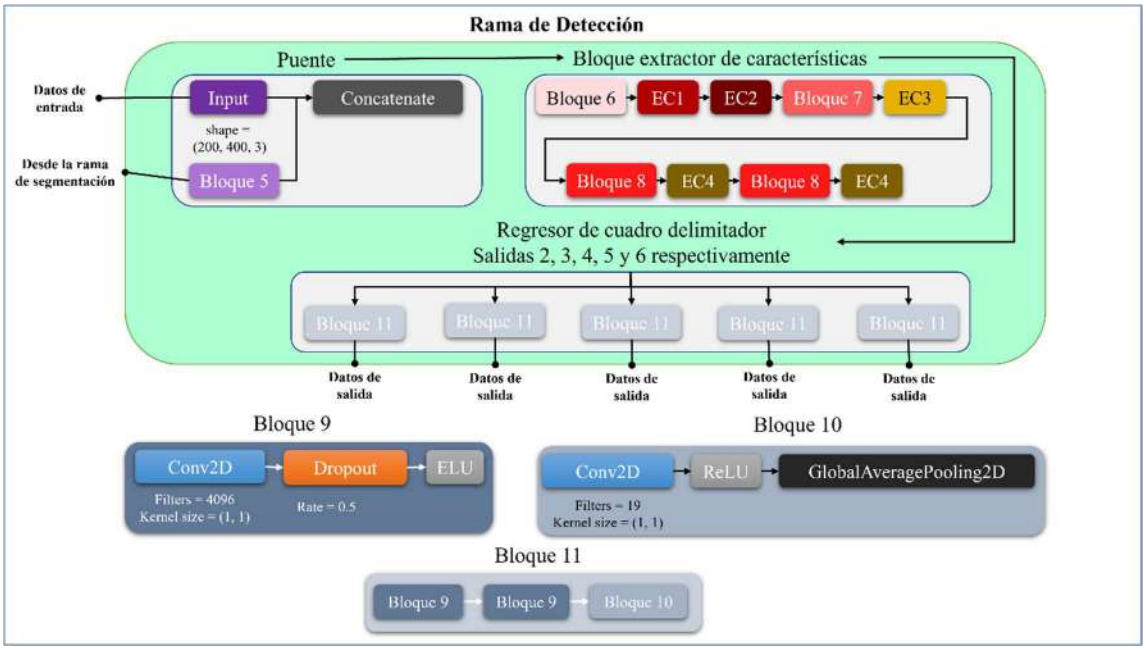


Figura 3.17 Representación capa por capa de la rama de detección del modelo SSADNet propuesto en [24].

El desarrollo capa por capa del modelo aquí utilizado, sirve como guía durante la construcción de algún otro modelo, ya que estos suelen construirse, probarse y modificarse, hasta llegar a buenos resultados. Durante la construcción de un nuevo modelo, se puede utilizar como guía en la construcción de la estructura la definición de cada capa existente en la librería de TensorFlow [34], así como estructuras de otros modelos ya existentes.

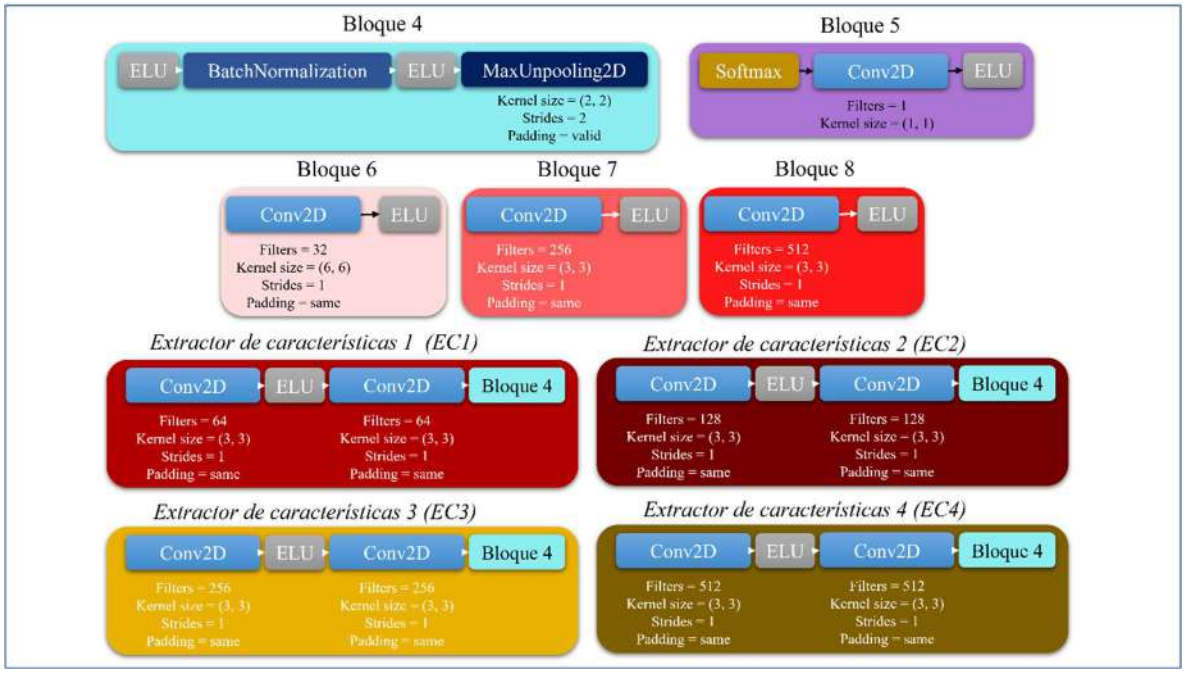


Figura 3.18 Descripción de los bloques de la rama de detección capa por capa de la figura anterior.

3.3.4 Entrenamiento y ajustes para optimización

Dentro de la librería de TensorFlow [34] se encuentra un comando, 'modelo.train', para llamar a la función que realiza el entrenamiento utilizando las variables X e Y de la sección 3.3.1 con cada conjunto de datos generado en la sección 3.3.2. Esta función de entrenamiento necesita una serie de hiperparámetros que definen los resultados del entrenamiento y por ende los resultados de las inferencias. En esta investigación se proponen los hiperparámetros mostrados en la figura 3.19, los cuales proporcionan los mejores resultados de inferencia de acuerdo con la métrica de exactitud para el modelo SSADNet, hasta donde esta investigación pudo probar. Sin embargo, es importante mencionar que los hiperparámetros de esta sección se colocan de forma aleatoria en los modelos en

un principio, para luego ajustarlos según el rendimiento mostrado en cada prueba, guiados por el comportamiento de la función de pérdida y el cálculo de la exactitud de entrenamiento y validación.

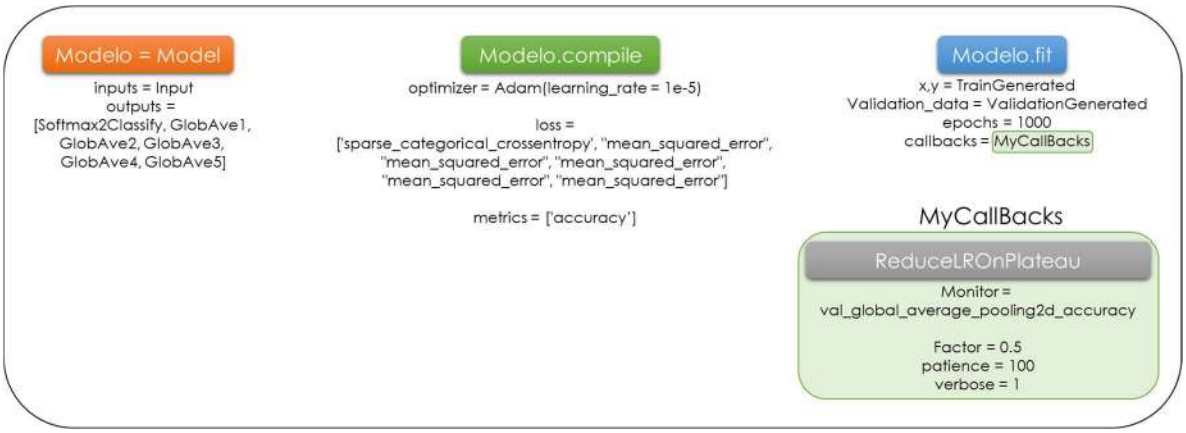


Figura 3.19 Descripción de los hiperparámetros de entrenamiento para el modelo SSADNet.

Se destaca en la sección de los hiperparámetros del comando '*Model*' de la fig. 11, el uso de un arreglo de seis variables para representar las salidas del modelo construido, así como el uso de seis funciones de pérdida, una por cada variable de salida, en el comando 'compile'.

3.3.5 Generación de métricas

En la subsección 3.2.7, se destacan las principales métricas de evaluación de modelos, las cuales son necesarias para saber si los hiperparámetros elegidos durante el entrenamiento son los óptimos. Además de utilizar la división de datos mediante *k-Fold* para asegurar el desempeño de un modelo, algunos métodos estadísticos son necesarias para asegurar que los resultados de las métricas obtenidas son verídicos, el método recomendando debido a su facilidad de comprensión y ejecución es el descrito por el teorema de Jacob Bernoulli y conocido como la 'Ley de los grandes números'; que de acuerdo con [39] propone en palabras simples que entre mayor es el número de ocasiones que un experimento se repite, el resultado más frecuente será el más acercado a la realidad o al resultado verdadero del experimento.

El método que en esta tesis se propuso para obtener las métricas óptimas de un modelo es el siguiente:

- Al finalizar el primer entrenamiento con hiperparámetros aleatorios, se calculan las métricas de las predicciones que permiten validar el rendimiento del modelo y con esto, generar ajustes en los hiperparámetros en el siguiente y en cada uno de los próximos entrenamientos, calculando en cada ocasión las métricas de evaluación para seguir ajustando hasta llegar a las mejores métricas.

Las métricas pueden ser programadas desde cero por el usuario, siguiendo las fórmulas de cada métrica, sin embargo, existen librerías que ya te permiten obtener las métricas de un modelo directamente definiendo las variables verdaderas y predichas. Una de estas librerías con métricas ya programadas en Python y la recomendada es '*scikit-learn*' [40].

3.3.6 Generación del archivo de predicciones con el modelo entrenado

Para finalizar la etapa del desarrollo del modelo de inteligencia artificial, es necesario guardar el modelo entrenado, validado y optimizado como un archivo del tipo *.h5* o algún otro que sea propio de modelos de inteligencia artificial, buscando que el tipo de archivo escogido es compatible con el tipo que el sistema embebido seleccionado puede procesar. Para generar el tipo sugerido, se utiliza la librería *h5py de Python* [41], con la cual guardaremos el modelo en un solo archivo que podremos cargar y utilizar cada vez que lo ocupemos dentro del sistema embebido seleccionado.

Para evitar tener problemas de compatibilidad entre el computador utilizado para entrenar el modelo y el sistema embebido seleccionado para ejecutar inferencias con el modelo, se recomienda utilizar la misma versión de *Python* [42] y *Tensorflow* o *Tensorflow-gpu* según el dispositivo, además de buscar asimilar lo mejor posible las versiones de *cuDNN y CUDA*, en caso de ser necesarias para ejecutar el modelo con la *GPU* del hardware.

3.4 Desarrollo del sistema de percepción ambiental

3.4.1 Lectura de sensores para la generación de datos de entrada válidos

En esta sección, se centra en generar los datos de entrada en bruto necesarios para que una vez transformados tengan el formato que el modelo de inteligencia artificial espera a la entrada. Para lograrlo, es necesario entender la forma en que el sensor funciona al momento de analizar la escena y generar la información de esta, siendo necesario seguir las instrucciones de conexiones y aplicación sugerida por la empresa fabricante del sensor, buscando asegurar en primera instancia el correcto funcionamiento y una obtención de mediciones acertadas. Posteriormente, se busca imitar los resultados de las mediciones de la escena con la aplicación conservando las conexiones, pero utilizando el software de programación con el lenguaje donde se generó el modelo de inferencia, de tal manera que se genere un programa totalmente propio o que utilice alguna librería para generar lecturas del sensor y adquiera la información de la escena de la misma manera que la aplicación y conserve las mediciones en una variable del programa para utilizarla posteriormente.

En el caso particular del sensor elegido en la sección 3.1.4, se busca generar la lectura de la escena frontal al sensor de forma tridimensional, Figura 3.20, generada por una nube de puntos, de forma tal que cada punto cuente con su información *X, Y, Z, i* de referencia. Para lograr obtener dicha información en una variable para su posterior transformación, se siguen los procedimientos generales, utilizando la aplicación y conexiones para el LiDAR seleccionado y posteriormente generando el software en Python para la lectura de los datos, recomendando siempre utilizar librerías disponibles en la web que te facilitan la adquisición de datos en el formato deseado.

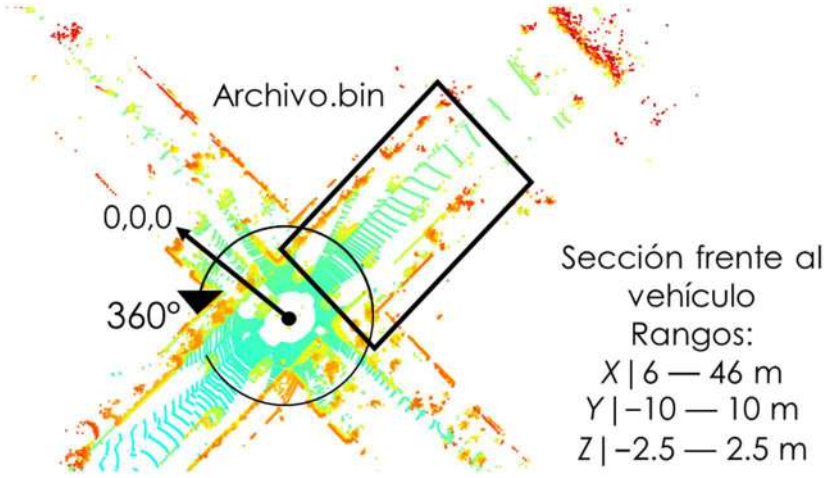


Figura 3.20 Representación de la delimitación de la escena frontal al sensor siguiendo ciertos rangos, en una escena generada por una nube de puntos en 360 grados y de forma tridimensional.

3.4.2 Predicción con nuevas entradas

Una vez que es posible la adquisición de datos crudos, siguiendo la sección anterior, y su transformación al formato que la red neuronal espera a la entrada, en particular siguiendo el apartado 3.2.1, se usa el archivo del modelo generado en la sección 3.3.6 para cargarlo en un nuevo programa y utilizarlo junto con la función de predicción, en particular en Python 'modelo.predict', para inferir la percepción ambiental de una escena capturada durante la adquisición de datos.

Por cada escena capturada durante la adquisición de datos es posible hacer una inferencia, igualmente es posible capturar varias escenas e inferir todas al mismo tiempo en una solo ejecución de predicción.

Para validar que las inferencias son realmente correctas, es necesario armar un escenario controlado, donde se conozca con anterioridad las posiciones de cada uno de los objetos de interés, siendo así que existirá información de referencia para confirmar que las posiciones inferidas son acertadas y obtener márgenes de error a partir varios experimentos realizados.

Es necesario conectar el sensor principal en el computador de entrenamiento y realizar las pruebas del software generado en este dispositivo, para asegurar el funcionamiento y validar las inferencias, para posteriormente, pasar el software y el sensor al sistema embebido seleccionado, en particular el escogido en la sección 3.1.3, y comprobar que el hardware embebido es capaz de ejecutar el software y obtener los mismos resultados de percepción ambiental, o bastantes aproximados, que en el computador de entrenamiento.

En caso de no contar aún con una correcta adquisición de datos de la sección anterior, se pueden utilizar archivos de la base de datos que cuenten con escenas en crudo como ejemplo, para validar todo el proceso de inferencia de nuevas escenas, en espera a que se pueda alimentar este proceso con escenas adquiridas a partir de los sensores y terminar el software correctamente. El método para validar las inferencias sería utilizando las etiquetas de las posiciones de los objetos de interés pertenecientes a la escena que se use de ejemplo.

3.4.3 Predicción en tiempo real

Para un correcto procesamiento de percepción ambiental, es necesario realizar inferencias de escenas en tiempo real, por lo tanto, se debe definir la cantidad de análisis de percepción ambiental por segundo que el modelo debe predecir, de acuerdo con las capacidades combinadas del modelo de inteligencia artificial junto con el sensor principal.

En este apartado es necesario definir que solo un análisis de percepción ambiental consta de la adquisición de datos de una sola escena, la transformación de estos a los esperados por el modelo, el procesamiento de la inferencia de la escena y por último la traducción de la inferencia a datos organizados entendibles por el usuario. En particular, se busca obtener al menos 22 análisis de percepción ambiental por segundo con el sensor LiDAR de la sección 3.1.3 y la Jetson de la sección 3.1.4.

Una vez definido el objetivo mencionado, se procede a realizar un programa para ejecutar el análisis de percepción ambiental de forma iterativa, guiados por el software generado en la sección anterior. En particular, se utiliza el software de predicciones nuevas de la sección de arriba y se coloca dentro de un ciclo ‘mientras’ en Python, para asegurar que se ejecute constantemente hasta que se decida detener manualmente o se cierre el programa. Es necesario hacer mediciones de tiempo de ejecución de cada iteración, en miras de optimizar el programa para que sea capaz de generar todas las predicciones pretendidas por segundo o que se acerque lo más posible.

Como método de validación se pueden generar inferencias de escenario controlados e ir guardando cada predicción en una variable para completar al menos unos segundos, para posteriormente mostrarlos y analizarlos como en el caso de un solo análisis de la sección anterior.

Al igual que en la sección pasada, se recomienda primeramente probar el software creado en el computador de entrenamiento, para posteriormente pasar al hardware embebido, considerando que la optimización del software robusta se realiza en el computador principal para luego hacer pequeños ajustes que mejoren aún más el rendimiento al ejecutar en el sistema embebido. Además, también se puede probar de forma preliminar el software con escenas de ejemplo de la misma base de datos, en caso de tener alguna dificultad con la adquisición de datos del sensor.

3.5 Montaje del sistema de percepción ambiental en vehículo de prueba

3.5.1 Posicionar el sensor y hardware embebido en el vehículo de prueba

Una vez que tenemos el hardware en armonía con el software, es posible proceder al montaje el hardware embebido al auto de prueba, siguiendo las especificaciones de posicionamiento del sensor de la Figura 3.21. Se recomienda la construcción de una base para la fijación del sensor en su posición.

En cuanto al hardware embebido, su posición dentro del vehículo puede variar, siempre y cuando se encuentre correctamente protegido y fijado, considerando que entre más cerca se encuentren los periféricos de entrada (sensor) del procesamiento central (hardware embebido), mejores son los tiempos de adquisición de datos y por ende la velocidad del procesamiento de un solo análisis de percepción ambiental se incrementa.

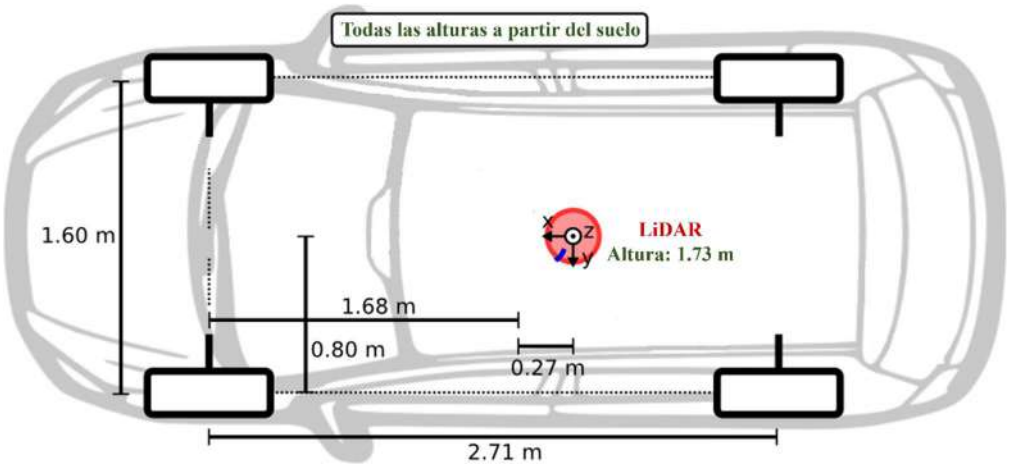


Figura 3.21 Descripción gráfica de la posición del sensor LiDAR durante la captura de la base de datos KITTI [31]*, figura adaptada de* [43].

Es posible cambiar la ubicación del sensor en el vehículo, realizando las mediciones correspondientes a su desplazamiento en cada eje para considerarlas tanto durante la adquisición de datos del sensor como en los resultados que arrojan las inferencias, es decir, sumarle o restarle la longitud desplazada en algún eje al valor que arroja las longitudes de cada punto durante la adquisición de datos en el eje desplazado y por otro lado, hacer la operación contraria al momento de obtener los resultados de la inferencia para los ejes que fueron desplazados. Un factor importante para considerar en el caso del desplazamiento del sensor es la necesidad de modificar los rangos en los que aceptamos datos del sensor para construir los archivos que se esperan a la entrada del modelo de inteligencia artificial de la sección 3.2.1, restando al rango mínimo el desplazamiento realizado en cierto eje, todo en miras de asegurar que las mediciones durante la adquisición de datos sigan estando dentro del rango que el sensor presenta precisiones superiores.

Además, para la batería seleccionada en la sección 3.1.3 se recomienda colocarla cerca del embebido, pero en una base de contención separada, de forma tal, que sea posible quitarla de manera sencilla para cargarla con facilidad.

3.5.2 Desarrollo de pruebas y validación final

Utilizando el mismo escenario de pruebas de la sección 3.4.2 y colocando el vehículo de tal forma que todos los cálculos de posiciones del escenario se calculen tomando el sensor como el origen, se procede a realizar pruebas de ejecución del software.

En primera instancia, se realiza solo un análisis de percepción ambiental para comprobar que las posiciones inferidas son correctas y por ende que el sistema de percepción ambiental esta en funcionamiento. En caso de que los experimentos presenten fallos, es necesario verificar el cumplimiento de la sección 3.5.1 y validar las mediciones de las posiciones reales dentro del escenario de prueba, si el problema persiste, será necesario realizar ajustes dentro del software siguiendo la sección 3.4.2, si el problema persiste luego de esta validación, se deben realizar ajustes para calibrar a detalle con el sistema de percepción ambiental montado en el vehículo de prueba.

Luego de obtener buenos resultado con un solo análisis de percepción ambiental, se ejecutan análisis en tiempo real imitando las instrucciones de la sección 3.4.3 pero con todo el sistema de percepción montado. En caso de presentar fallas, será necesario volver a la sección 3.4.3, para verificar el funcionamiento en tiempo real, si el problema persiste luego de esta validación, se deben realizar ajustes para calibrar a detalle con el sistema de percepción ambiental montado en el vehículo de prueba.

Finalmente, con el sistema de hardware y software de percepción ambiental funcionado en tiempo real, se procede a salir a un ambiente controlado donde se pueda manejar el vehículo para ir haciendo pruebas de mediciones aleatorias en el camino respecto a otros vehículos para comprobarlas con las inferencias.

Para una validación extra, se recomienda posicionar una cámara como en la Figura 3.22 y agregar al programa principal una opción para imprimir una imagen frontal RGB de la escena analizada, añadiendo en esta los cuadros delimitadores de los objetos de interés, generados en base a la inferencia de la escena realizada, sumando también a la imagen, las áreas de color de acuerdo con el mapa de segmentación inferido.

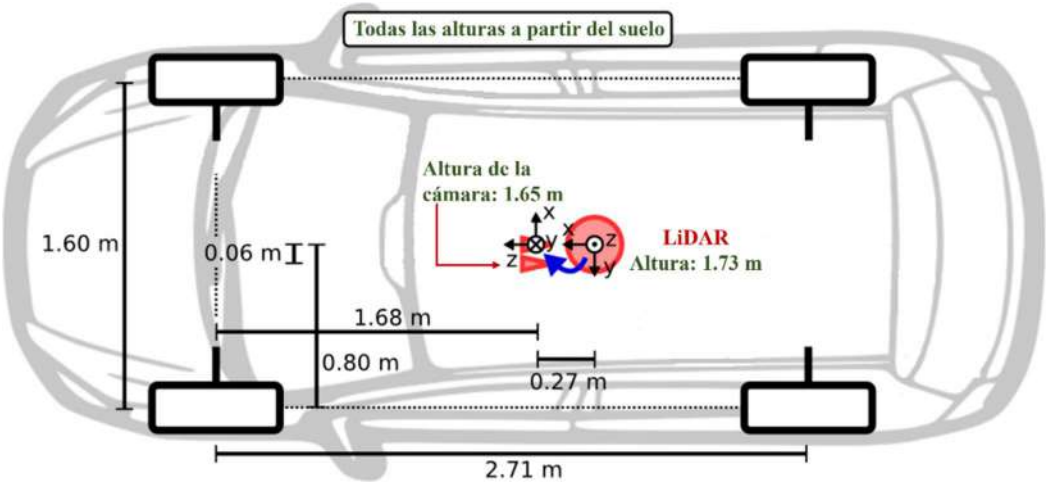


Figura 3.22 Descripción gráfica de la posición del sensor LiDAR y la cámara izquierda durante la captura de la base de datos KITTI [31]*, figura adaptada de* [43].

4. Resultados y discusión

4.1 Resultados

Al buscar afinar la propuesta metodológica, se desarrolló un caso de estudio en donde se puso a prueba cada sección metodológica. El resultado, fue una retroalimentación sobre el mismo método en cada paso que, de manera práctica, permitió comprobar que seguir los pasos de cada sección del capítulo tres generará un sistema de percepción ambiental, el cual presenta los resultados mostrados a continuación.

4.1.1 Base de datos especializada

El rendimiento de las inferencias de segmentación del camino para cada una de las escenas de la base de datos original KITTI, durante la creación de la base de datos, es el presentado en la Tabla 4.1 y en la Figura 4.1, con las métricas de evaluación y la matriz de confusión normalizada, respectivamente. En la tabla se reportan valores estables entre las métricas, lo que sugiera un rendimiento estable del modelo durante las inferencias, además, muestra valores en las métricas de alrededor del 95%, lo cual es el resultado de inferencias altamente acertadas como salidas del modelo. En cuanto a la matriz de confusión de la figura, destaca un mayor porcentaje de pixeles detectados correctamente, tanto falsos cuando deben ser falsos (No camino), como positivos cuando tienen que serlo (Camino), dejando solo un pequeño porcentaje de pixeles o secciones de la imagen mal inferidas o segmentadas. Ambas muestras de rendimiento fueron creadas a partir de la inferencia de escenas no analizadas por el modelo llamado LiDAR-only deep neural network - Improved (LoDNN-I) en inglés durante el entrenamiento, es decir, a partir de datos de prueba.

Tabla 4.1 Métricas del rendimiento de la segmentación del camino urbano, a partir de datos de prueba.

Método	**Puntaje F1**	**PRE promedio**	**ACC**	**PRE**	**REC**
LoDNN-I (propuesta)	95.77	92.54	97.53	94.34	97.25

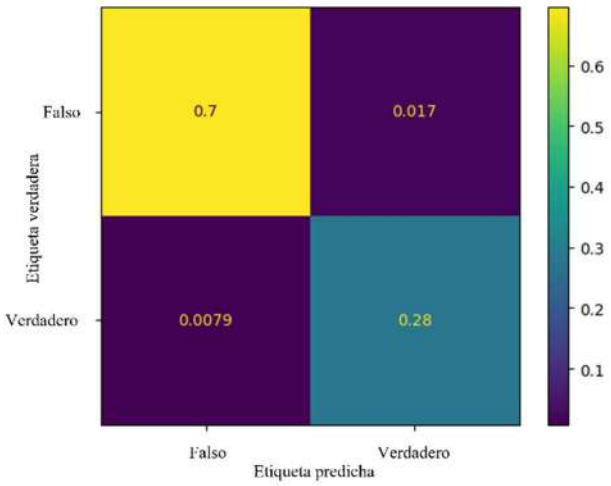


Figura 4.1 Matriz de confusión de los resultados del rendimiento de la segmentación del camino urbano a partir de datos de prueba, normalizada entre cero y uno.

Posterior al proceso de segmentación del camino, el resto de los objetos de interés para la creación de los mapas de segmentación son añadidos haciendo uso de técnicas matemáticas, lo cual genera una compatibilidad totalmente exacta de los cuadros delimitadores entre los objetos de interés del mapa de segmentación y los delimitadores generados a partir de las etiquetas de la base de datos propuesta. Como muestra de lo mencionado, se puede observar en la escena de ejemplo de la Figura 4.2, que cuenta con cada uno de los tres tipos de archivo disponibles en la base de datos final propuesta.

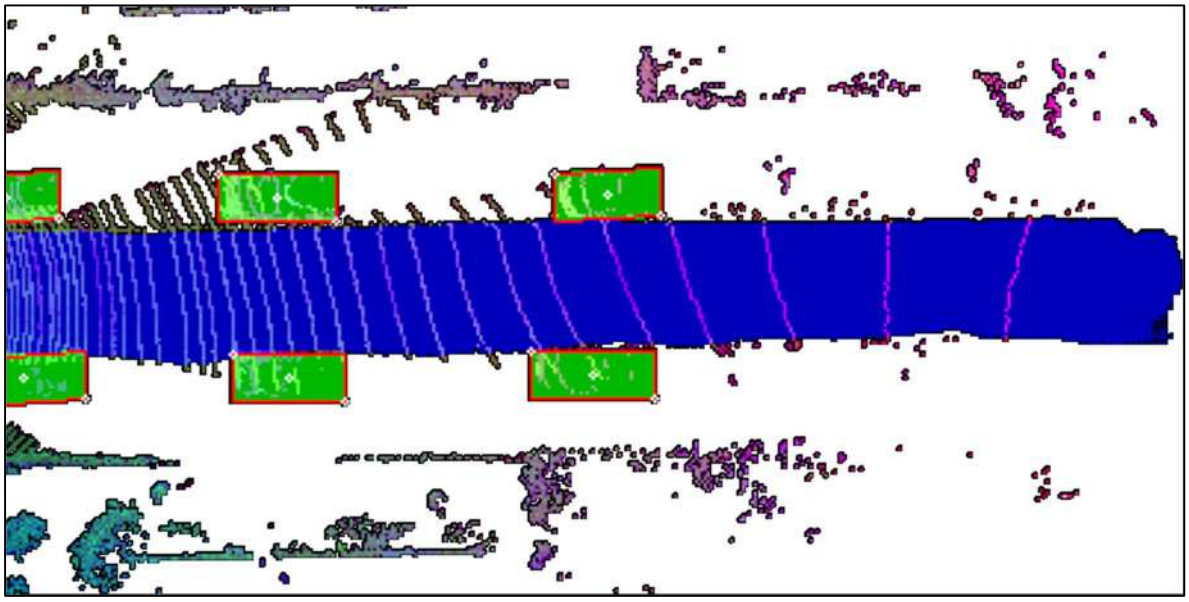


Figura 4.2 Escena de la base de datos final propuesta, que evidencia los resultados de utilizar los tres tipos de archivo disponibles de manera conjunta.

4.1.2 Sistema de percepción ambiental

Al igual que en la sección previa, pero refiriéndose a la percepción ambiental, los resultados de rendimiento del modelo Simultaneous Segmentation And Detection Network – Improved (SSADNet-I) en inglés; son mostrados en las Tablas 4.2 y 4.3 con las métricas de evaluación. Una vez más, resultados a partir de datos de prueba. En la primera tabla, se observan métricas altas que destacan una correcta segmentación o una asignación de pixeles acertada, que permite identificar a qué objeto pertenece cada uno de los pixeles de la escena capturada. En la segunda tabla, se presentan buenas métricas para la detección de los vehículos dentro de la escena capturada, lo que permite identificar de forma aproximadamente correcta las posiciones de los vehículos, pero dejando aún oportunidades de mejora para una ubicación espacial más precisa.

Tabla 4.2 Métricas del rendimiento de la segmentación de objetos de interés durante la percepción ambiental, a partir de datos de prueba.

Métodos para segmentación	**ACC [%]**	**Media precisión promedio [%]**	**Media IoU [%]**
SSADNet-I (propuesta)	97.18	97.03	89.00

Tabla 4.3 Métricas del rendimiento de la detección de objetos de interés durante la percepción ambiental, a partir de datos de prueba.

Métodos para detección	**Media precisión promedio [%]**	**Media IoU [%]**
SSADNet-I (propuesta)	85.11	70.51

La red neuronal SSADNet-I que genera los resultados de detección y ubicación tridimensional del vehículo, tiene valores modificables que cambian la precisión de sus resultados llamados hiperparámetros. En el método se presentan los hiperparámetros ideales o finales a utilizar, sin embargo, se realizaron experimentos de ablación donde se pusieron a prueba diferentes valores en la búsqueda de mejoras en el rendimiento del modelo. El resultado de estos experimentos se muestra en la siguiente tabla 4.4:

Tabla 4.4 Resultados de los experimentos de ablación sobe el modelo SSADNet.

Hiperparámetros	Resultados
Ritmo de aprendizaje (learning_rate)	Mayor que 1e-5: Se estanca en una menor precisión. Menor que 1e-5: La precisión tarda mucho más en llegar, pero logra el mismo resultado.
Función de pérdida para la detección (loss)	Al utilizar la función de pérdida "log_cosh", en lugar de "mean_squared_error" para la sección de la red de detección resultados, no presentan diferencias en la precisión resultante.
Pesos de las pérdidas (loss_weights)	Colocar pesos mayores a $\frac{1}{10}$ en la sección de detección en contraste al valor añadido para la sección de segmentación, genera un sobre entrenamiento en el modelo.

4.1.3 Implementación del sistema de percepción ambiental

El sistema de percepción ambiental presenta tiempos de un solo análisis, sin contar el tiempo de la adquisición de datos, de 0.5 segundos en el computador principal de entrenamiento, mientras que el sistema embebido Jetson Nano presenta un tiempo de 5.40 segundos por análisis, es decir, más de 10 veces más. La razón del aumento en el tiempo de procesamiento en el sistema embebido es debido al agotamiento de recursos de memoria RAM tanto del GPU como la general del embebido, es necesario utilizar ambos recursos de RAM debido a que, si se pretende llevar a cabo el análisis solo con GPU, la ejecución falla al exceder las capacidades de la gráfica. Estas pruebas se lograron ejecutar de forma remota en el embebido, utilizando un computador conectado a la misma red como método de monitorización.

4.1.4 Componentes base

En este proyecto se sugiere el uso de la base de datos KITTI Vision Benchmark Suite debido a que es ampliamente usada en otros proyectos como base de datos de referencia. Además, está disponible para su uso en la web de forma gratuita para fines educativos y presenta una cantidad grande de datos. En el mismo camino, la red neuronal elegida es el resultado de buscar un equilibrio entre velocidad de procesamiento y precisión de detección y ubicación tridimensional, lo cual se facilita con la SSADNet al otorgar resultados de segmentación y detección con una misma imagen de entrada.

En cuanto a temas de hardware, se realizaron pruebas de entrenamiento con una GPU GTX 1050 en combinación con tarjetas RAM de 16 y 32 Gb. El resultado de esta GPU en combinación con una tarjeta de 16 Gb, es que al equipo le es imposible procesar la cantidad de datos que se manejan, generando errores en el sistema operativo e imposibilitando el procesamiento. Por otro lado, el uso de una RAM de 32 Gb hace posible el procesamiento de lo planteado en este método, sin embargo, resulta extremadamente tardado el entrenamiento, llegando a tiempos en los cuales resulta imposible realizar las pruebas necesarias para obtener conclusiones. Lo anterior, generó la necesidad de probar con una GPU 3070 que cuenta con mejores recursos y permitió las pruebas y entrenamiento final.

4.2 Discusión

Los resultados de la segmentación del camino descritos en la sección 4.1.1, presentan innovación en comparación al estado del arte, como se muestra en la Tabla 4.5 con la comparación de los resultados obtenidos, junto con otras investigaciones que también utilizan nubes de puntos; para representar las escenas de análisis.

Tabla 4.5 Tabla comparativa de las métricas del rendimiento del estado del arte durante la segmentación del camino.

Métodos	**F1 Score**	**AP**	**PRE**	**REC**
LoDNN-I (propuesta)	**95.77**	**92.54**	**94.34**	**97.25**
LoDNN [36]	94.07	92.03	92.81	95.37
Up-Conv-Poly [44]	93.83	90.47	94.00	93.67
DDN [45]	93.43	89.67	95.09	91.82
FTP [46]	91.61	90.96	91.04	92.2
FCN-LC [47]	90.79	85.83	90.87	90.72
HIM [48]	90.64	81.42	91.62	89.68
NNP [49]	89.68	86.5	89.67	89.68
RES3D-Velo [50]	86.58	78.34	82.63	90.92

En cuanto a los resultados tanto de detección como de segmentación de las inferencias de percepción ambiental, presentados en la sección 4.1.2, destacan una similitud con los presentados en el trabajo [24] que propone la red SSADNet utilizada. Los mismos, en comparación con otros trabajos del estado del arte, Tablas 4.6 y 4.7, tienen resultados similares o inferiores, sin embargo, se compara con trabajos donde el modelo infiere solamente la detección o la segmentación, mientras que el modelo empleado infiere ambos al mismo tiempo. Además, la diferencia principal en los resultados obtenidos en esta investigación es la consideración de la rotación de los vehículos detectados, concepto que transforma brevemente la estructura de la red SSADNet original.

Tabla 4.6 Tabla comparativa de las métricas del rendimiento del estado del arte durante la segmentación de objetos de interés.

Métodos para segmentación	**mIoU [%]**
SSADNet-I (propuesta)	**89.00**
SSADNet [24]	83.57
FCN-VGG16 [51]	94.14
LoDNN [36]	85.68

Tabla 4.7 Tabla comparativa de las métricas del rendimiento del estado del arte durante la detección de objetos de interés.

Métodos para detección	**mIoU [%]**
SSADNet-I (propuesta)	**70.51**
SSADNet [24]	70.48
RetinaNet-ResNet50 [52]	81.36
YOLO v3 [53]	75.74

En el caso de los resultados de la implementación del sistema de percepción ambiental, presentados en la sección 4.1.3, se presenta una ausencia de pruebas con un sistema embebido más potente y con la adquisición de datos de un sensor LiDAR, dichas prueba se vieron frustradas por temas económicos, sin embargo, gracias a la adquisición del conocimiento en el área fue posible proponer el hardware más accesible para futuros trabajos en esta área. Se destaca que el embebido final propuesto, cuenta con recursos 10 veces más potentes que la Jetson Nano originalmente utilizada durante la obtención de los resultados descritos en la sección 4.1.3.

Finalmente, es necesario mencionar que la idea de tiempo real presentada en el artículo [24], es diferente al tiempo real que se busca en esta investigación, ya que en el artículo presenta esta idea

como la inferencia de múltiples escenas al mismo tiempo, mientras que en este trabajo se busca realizar al menos 22 análisis de percepción ambiental por segundo para ser considerado tiempo real, recordando que solo un análisis de percepción ambiental consta de la adquisición de datos de una escena, transformación de la escena al formato esperado por el modelo, inferencia de la escena y finalmente la interpretación de los resultados de inferencia como datos de percepción ambiental.

4.3 Valor agregado

4.3.1 Impacto tecnológico

Se aporta una propuesta metodológica que funge como base para el desarrollo del hardware y software que permite la transformación de un auto común a vehículo autónomo.

4.3.2 Impacto institucional

Propuesta metodológica que funge como base para la transformación del auto eléctrico de la facultad de Ingeniería de la Universidad Autónoma de Querétaro.

4.3.3 Impacto social

Permite la apropiación de conocimientos en el área de vehículos autónomos, para una posterior colaboración con el sector privado y educativo en proyectos asociados.

4.3.4 Impacto económico

Iniciar investigaciones en el área de vehículos autónomos que sigan los niveles propuestos por la SAE en el país, buscando propiciar la inversión en investigación en el sector privado y educativo sobre estos temas.

4.4 Publicaciones

En base a la necesidad de una base de datos que contara con las secciones de imágenes de vista aérea, mapas de segmentación y etiquetas por cada una de las escenas del conjunto de datos, se siguió el método descrito de forma general en la sección general 3.2 para su creación. Dicho método y resultados fueron descritos con mayor detalle y presentados a la comunidad científica a través de la publicación del artículo '*A Specialized Database for Autonomous Vehicles Based on the KITTI Vision Benchmark*' [33], en la revista '*Electronics*' [54].

4.5 Alcances

El presente trabajo va dirigido a estudiante e investigadores que busquen una guía para el desarrollo de un sistema de percepción ambiental y con esto sentar las bases para el desarrollo de un vehículo autónomo, de forma tal que, al seguir el método propuesto será posible tanto segmentar una escena urbana diferenciando entre camino, carros, camionetas, vehículos tipo van y espacio sin importancia, como detectar la posición bidimensional del centro de carros, camionetas y vehículos tipo van, su rotación y sus dimensiones de largo y ancho. Lo anterior, se hará con un porcentaje de exactitud de un 89 y 70 por ciento, en el caso de la segmentación y detección, respectivamente. Además, será posible implementar el sistema de percepción ambiental utilizando un sensor LiDAR tridimensional junto con un sistema embebido con mayores recursos que una Jetson Nano de Nvidia [55], del cual dependerá la velocidad de cada análisis de percepción ambiental, según los recursos de los que disponga.

5. Conclusiones

En este trabajo se propuso una serie de pasos metodológicos para producir un sistema de percepción ambiental, con el cual se detectan los vehículos frente al conductor y el camino conducible con un 89% en promedio de la métrica Intersección Sobre Unión y al mismo tiempo se arrojan las posiciones tridimensionales de cada uno de los vehículos detectados respecto al conductor con un 70.51% en promedio de Intersección Sobre Unión. Además, con esta propuesta metodológica, es posible alcanzar una frecuencia de análisis y entrega de resultados de 2 fotogramas por segundo, al ejecutar la inferencia con un computador Intel i7, 32 Gb de RAM y una GPU RTX 3070. Con estas medidas, se mejora el estado del arte en cuanto a capacidad de producción de análisis y entrega de resultados por segundo. También, en este proyecto se considera la posición rotada de los vehículos al momento de su detección y determinación de posición tridimensional con la red neuronal, innovando el trabajo que introdujo la SSADNet en este aspecto.

Con la creación y planteamiento de la metodología propuesta, se llena un vacío en el estado del arte sobre explicaciones, instructivos o métodos que faciliten la producción de un sistema de percepción ambiental. Además, se presentan evidencias sobre el método de construcción de una red neuronal capa por capa, en específico la SSADNet-I, y todos los hiperparámetros que se necesitan para que el proceso sea funcional, lo cual, es una tarea de innovación en el estado del arte.

El sistema Jetson Nano de NVIDIA, a pesar de ser un hardware especializado, carece de la potencia necesaria para poder ejecutar el sistema de percepción ambiental; producto del método planteado. Sin embargo, la idea de buscar sistemas embebidos comerciales para implementar el sistema de percepción ambiental es nueva en el estado del arte. Además, debe ser considerado que un sensor LiDAR tridimensional, necesario al momento de buscar reproducir el método planteado, tiene costos elevados de adquisición.

Referencias

[1] D. Gruyer, V. Magnier, K. Hamdi, L. Claussmann, O. Orfila, and A. Rakotonirainy, "Perception, information processing and modeling: Critical stages for autonomous driving applications," *Annu. Rev. Control*, vol. 44, pp. 323–341, 2017, doi: 10.1016/j.arcontrol.2017.09.012.

[2] C. Jung, D. Lee, S. Lee, and D. H. Shim, "V2x-communication-aided autonomous driving: System design and experimental validation," *Sensors (Switzerland)*, vol. 20, no. 10, pp. 1–21, 2020, doi: 10.3390/s20102903.

[3] S. Singh, "Critical Reasons for Crashes Investigated in the National Motor Vehicle Crash Causation Survey," pp. 1–2, 2015.

[4] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A Survey of Autonomous Driving: Common Practices and Emerging Technologies," *IEEE Access*, vol. 8, pp. 58443–58469, 2020, doi: 10.1109/ACCESS.2020.2983149.

[5] M. R. Bachute and J. M. Subhedar, "Autonomous Driving Architectures: Insights of Machine Learning and Deep Learning Algorithms," *Mach. Learn. with Appl.*, vol. 6, no. March, p. 100164, 2021, doi: 10.1016/j.mlwa.2021.100164.

[6] C. for S. and T. S. CWTS, "VOSviewer Visualizing Scientific Landscapes," *VOSviewer*, 2022. https://www.vosviewer.com/

[7] D. A. Pomerleau, "ALVINN," *Source*, 1989.

[8] B. Ulmer, "VITA II," *Power*, pp. 1–6, 1996.

[9] SAE International, "Taxonomy and definitions for terms related to driv- ing automation systems for on-road motor vehicles," *SAE Int.*, vol. 4970, no. 724, pp. 1–35, 2018.

[10] National Highway Traffic Safety Administration, "AUTOMATED DRIVING SYSTEMS 2.0: A VISION FOR SAFETY," Washington, D.C., 2017.

[11] H. Chae, C. M. Kang, B. Kim, J. Kim, C. C. Chung, and J. W. Choi, "Autonomous braking system via deep reinforcement learning," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, Oct. 2017, vol. 2018-March, pp. 1–6. doi: 10.1109/ITSC.2017.8317839.

[12] F. Codevilla, M. Miiller, A. Lopez, V. Koltun, and A. Dosovitskiy, "End-to-End Driving Via Conditional Imitation Learning," *Proc. - IEEE Int. Conf. Robot. Autom.*, pp. 4693–4700, 2018, doi: 10.1109/ICRA.2018.8460487.

[13] K. Min, H. Kim, and K. Huh, “Deep Q Learning Based High Level Driving Policy Determination,” *IEEE Intell. Veh. Symp. Proc.*, vol. 2018-June, no. Iv, pp. 226–231, 2018, doi: 10.1109/IVS.2018.8500645.

[14] Y. Chen, D. Zhao, L. Lv, and Q. Zhang, “Multi-task learning for dangerous object detection in autonomous driving,” *Inf. Sci. (Ny).*, vol. 432, pp. 559–571, 2018, doi: 10.1016/j.ins.2017.08.035.

[15] A. Amini, G. Rosman, S. Karaman, and D. Rus, “Variational end-to-end navigation and localization,” *Proc. - IEEE Int. Conf. Robot. Autom.*, vol. 2019-May, pp. 8958–8964, 2019, doi: 10.1109/ICRA.2019.8793579.

[16] A. Kendall *et al.*, “Learning to drive in a day,” *Proc. - IEEE Int. Conf. Robot. Autom.*, vol. 2019-May, pp. 8248–8254, 2019, doi: 10.1109/ICRA.2019.8793742.

[17] C. Katrakazas, M. Quddus, and W. H. Chen, “A new integrated collision risk assessment methodology for autonomous vehicles,” *Accid. Anal. Prev.*, vol. 127, no. September 2018, pp. 61–79, 2019, doi: 10.1016/j.aap.2019.01.029.

[18] W. Yang, Z. Li, C. Wang, and J. Li, “A multi-task Faster R-CNN method for 3D vehicle detection based on a single image,” *Appl. Soft Comput. J.*, vol. 95, p. 106533, 2020, doi: 10.1016/j.asoc.2020.106533.

[19] A. Zaarane, I. Slimani, W. Al Okaishi, I. Atouf, and A. Hamdoun, “Distance measurement system for autonomous vehicles using stereo camera,” *Array*, vol. 5, no. May 2019, p. 100016, 2020, doi: 10.1016/j.array.2020.100016.

[20] S. Chang *et al.*, “Spatial attention fusion for obstacle detection using mmwave radar and vision sensor,” *Sensors (Switzerland)*, vol. 20, no. 4, pp. 1–21, 2020, doi: 10.3390/s20040956.

[21] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, and J. Garcia-Rodriguez, “A Review on Deep Learning Techniques Applied to Semantic Segmentation,” pp. 1–23, 2017.

[22] N. Zou, Z. Xiang, Y. Chen, S. Chen, and C. Qiao, “Simultaneous semantic segmentation and depth completion with constraint of boundary,” *Sensors (Switzerland)*, vol. 20, no. 3, pp. 1–15, 2020, doi: 10.3390/s20030635.

[23] I. S. Weon, S. G. Lee, and J. K. Ryu, “Object Recognition Based Interpolation with 3D LIDAR and Vision for Autonomous Driving of an Intelligent Vehicle,” *IEEE Access*, vol. 8, pp. 65599–65608, 2020, doi: 10.1109/ACCESS.2020.2982681.

[24] Y. Lee and S. Park, “A deep learning-based perception algorithm using 3d lidar for autonomous driving: Simultaneous segmentation and detection network (SSADNet),” *Appl.*

Sci., vol. 10, no. 13, 2020, doi: 10.3390/app10134486.

[25] W. Song, S. Zou, Y. Tian, S. Fong, and K. Cho, "Classifying 3D objects in LiDAR point clouds with a back-propagation neural network," *Human-centric Comput. Inf. Sci.*, vol. 8, no. 1, 2018, doi: 10.1186/s13673-018-0152-7.

[26] W. Lu, Y. Zhou, G. Wan, S. Hou, and S. Song, "L3-Net: Towards Learning Based Lidar Localization for Autonomous Driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 6382–6391.

[27] B. Wu, A. Wan, X. Yue, and K. Keutzer, "SqueezeSeg: Convolutional Neural Nets with Recurrent CRF for Real-Time Road-Object Segmentation from 3D LiDAR Point Cloud," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2018, pp. 1887–1893. doi: 10.1109/ICRA.2018.8462926.

[28] D. Yang, L. Li, K. Redmill, and U. Ozguner, "Top-view trajectories: A pedestrian dataset of vehicle-crowd interaction from controlled experiments and crowded campus," in *IEEE Intelligent Vehicles Symposium, Proceedings*, 2019, vol. 2019-June, no. Iv, pp. 899–904. doi: 10.1109/IVS.2019.8814092.

[29] S. M. Azimi, P. Fischer, M. Korner, and P. Reinartz, "Aerial LaneNet: Lane-Marking Semantic Segmentation in Aerial Imagery Using Wavelet-Enhanced Cost-Sensitive Symmetric Fully Convolutional Neural Networks," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 5, pp. 2920–2938, 2019, doi: 10.1109/TGRS.2018.2878510.

[30] C. Pek, S. Manzinger, M. Koschi, and M. Althoff, "Using online verification to prevent autonomous vehicles from causing accidents," *Nat. Mach. Intell.*, vol. 2, no. 9, pp. 518–528, 2020, doi: 10.1038/s42256-020-0225-y.

[31] "The KITTI Vision Benchmark Suite," *Available online:*, 2013. https://www.cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=bev

[32] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3354–3361.

[33] J. I. Ortega-gomez, L. A. Morales-hernandez, and I. A. Cruz-albarran, "A Specialized Database for Autonomous Vehicles Based on the KITTI Vision Benchmark," *Electronics*, vol. 12, no. 14, 2023, doi: https://doi.org/10.3390/electronics12143165.

[34] "TensorFlow." https://www.tensorflow.org/?hl=es-419

[35] M. Hewitt, "Make Sense," *Victorian Studies*, 2006. https://www.makesense.ai/

[36] M. F. Caltagirone, L.; Scheidegger, S.; Svensson, L.; Wahde, "Fast LIDAR-based road detection using fully convolutional neural networks," in *IEEE Intelligent Vehicles*

Symposium, Proceedings, 2017, no. Iv, pp. 1019–1024. doi: 10.1109/IVS.2017.7995848.

[37] M. A. ACEVES FERNÁNDEZ, *INTELIGENCIA ARTIFICIAL PARA PROGRAMADORES CON PRISA*, 1ra ed. Querétaro, Querétaro, México, 2021.

[38] "Keras." https://keras.io/

[39] E. Seneta, "A tricentenary history of the law of large numbers," *Bernoulli*, vol. 19, no. 4, pp. 1088–1121, 2013, doi: 10.3150/12-BEJSP12.

[40] "scikit-learn_ machine learning in Python." https://scikit-learn.org/stable/

[41] "HDF5 for Python -- h5py.alfven.org." https://docs.h5py.org/en/stable/#

[42] J. S. Schwarz, C. Chapman, and E. McDonnell Feit, "Welcome to Python," *Python for Marketing Research and Analytics*, 2020. https://www.python.org/

[43] "The KITTI Vision Benchmark Suite - Road/Lane Detection." http://www.cvlibs.net/datasets/kitti/eval_road.php

[44] G. L. Oliveira, W. Burgard, and T. Brox, "Efficient deep models for monocular road segmentation," in *IEEE International Conference on Intelligent Robots and Systems*, 2016, vol. 2016-Novem, pp. 4885–4891. doi: 10.1109/IROS.2016.7759717.

[45] R. Mohan, "Deep Deconvolutional Networks for Scene Parsing," 2014.

[46] A. Laddha, M. K. Kocamaz, L. E. Navarro-Serment, and M. Hebert, "Map-supervised road detection," in *IEEE Intelligent Vehicles Symposium, Proceedings*, 2016, vol. 2016-Augus, no. Iv, pp. 118–123. doi: 10.1109/IVS.2016.7535374.

[47] C.-C.-T. Mendes, V. Frémont, and D.-F. Wolf, "Exploiting fully convolutional neural networks for fast road detection," in *2016 IEEE International Conference on Robotics and Automation*, 2016, pp. 3174–3179.

[48] and M. H. D. Munoz, J. A. Bagnell, "Stacked Hierarchical Labeling," in *Computer Vision—Eccv 2010, Pt Vi*, vol. 6316, D. Kostas, M. Petros, and P. Nikos, Eds. Berlin/Heidelberg, Germany: Springer, 2010, pp. 57–70.

[49] X. Chen *et al.*, "3D object proposals for accurate object class detection," *Adv. Neural Inf. Process. Syst.*, vol. 2015-Janua, pp. 424–432, 2015.

[50] D. F. W. and C. S. Patrick Y. Shinzato, "Road Terrain Detection: Avoiding Common Obstacle Detection Assumptions Using Sensor Fusion," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*, 2014, no. Iv, pp. 687–692.

[51] C. Han, Y. Duan, X. Tao, and J. Lu, "Fully Convolutional Networks for Semantic Segmentation," *IEEE Access*, vol. 7, pp. 43369–43382, 2019, doi: 10.1109/ACCESS.2019.2908685.

[52] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal Loss for Dense Object

Detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–327, 2020, doi: 10.1109/TPAMI.2018.2858826.

[53] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," 2018.

[54] "Materials | An Open Access Journal from MDPI." https://www.mdpi.com/journal/materials

[55] NVIDIA, "Jetson Nano." https://www.nvidia.com/es-la/autonomous-machines/embedded-systems/jetson-nano/product-development/ (accessed Jun. 22, 2022).

electronics

MDPI

Article

A Specialized Database for Autonomous Vehicles Based on the KITTI Vision Benchmark

Juan I. Ortega-Gomez *, Luis A. Morales-Hernandez * and Irving A. Cruz-Albarran

Faculty of Engineering, San Juan del Rio Campus, Autonomous University of Querétaro, San Juan del Rio 76807, Querétaro, Mexico; irving.cruz@uaq.mx
* Correspondence: jortega02@alumnos.uaq.mx (J.I.O.-G.); luis.morales@uaq.mx (L.A.M.-H.)

Abstract: Autonomous driving systems have emerged with the promise of preventing accidents. The first critical aspect of these systems is perception, where the regular practice is the use of top-view point clouds as the input; however, the existing databases in this area only present scenes with 3D point clouds and their respective labels. This generates an opportunity, and the objective of this work is to present a database with scenes directly in the top-view and their labels in the respective plane, as well as adding a segmentation map for each scene as a label for segmentation work. The method used during the creation of the proposed database is presented; this covers how to transform 3D to 2D top-view image point clouds, how the detection labels in the plane are generated, and how to implement a neural network for the generated segmentation maps of each scene. Using this method, a database was developed with 7481 scenes, each with its corresponding top-view image, label file, and segmentation map, where the road segmentation metrics are as follows: F1, 95.77; AP, 92.54; ACC, 97.53; PRE, 94.34; and REC, 97.25. This article presents the development of a database for segmentation and detection assignments, highlighting its particular use for environmental perception works.

Keywords: autonomous driving; driverless vehicle; environment perception; LiDAR; point cloud; top view; database

Citation: Ortega-Gomez, J.I.; Morales-Hernandez, L.A.; Cruz-Albarran, I.A. A Specialized Database for Autonomous Vehicles Based on the KITTI Vision Benchmark. *Electronics* **2023**, *12*, 3165. https://doi.org/10.3390/electronics12143165

Academic Editor: Felipe Jiménez

Received: 30 May 2023
Revised: 20 June 2023
Accepted: 29 June 2023
Published: 21 July 2023

1. Introduction

Guided by statistics, autonomous driving systems have emerged with the promise of preventing accidents, leading to research that is concerned, in the first instance, with integrating known techniques and methods [1]. Currently, the Society for Automotive Engineering (SAE) recommends dividing the types of autonomous cars into six levels, ranging from no automation to complete automation [2]. These levels are best presented by the US Department of Transportation [3].

The creation of a level-six car involves the fulfillment of multiple tasks. The first critical aspect is perception, which is responsible for perceiving the surroundings of the autonomous vehicle to detect each object of interest and the vehicle's three corresponding measurements of the center of the object in three-dimensional space, along with its dimensions and the class of object to which it belongs [4]. In pursuit of this critical task, research has relied on artificial intelligence methods, such as neural networks, to create accurate models to identify both vehicle movement spaces [5,6] and vehicles and pedestrians on the streets [7].

A regular practice in these works is the use of a LiDAR sensor as the primary means of input data acquisition for the neural networks and the basis of the representation of the results [8–10]. One of the main ways to present results in these investigations is to obtain a segmentation map of the analyzed scene, seeking to classify each pixel or scene datum within the classes of interest. This results in works where neural networks are proposed that aim to improve the metrics of the segmentation results [11–13].

Figura A.1 Primera hoja del artículo [33] referenciado en la sección 3.2, donde se encuentra información más detallada del proceso descrito en ese mismo apartado.