ARQUITECTURA SOFTWARE Y DE NAVEGACIÓN PARA VEHÍCULO AUTÓNOMO

MIGUEL ÁNGEL DE MIGUEL PARAÍSO

Tesis depositada en cumplimiento parcial de los requisitos para el grado de Doctor en

Ingeniería Electrónica, Eléctrica y Automática

Universidad Carlos III de Madrid

Directores:

José María Armingol Moreno

Fernando García Fernández

Tutor:

José María Armingol Moreno

Marzo 2022

"People are so bad at driving cars
that computers don't have to be that good
to be much better."

— Marc Andreessen

AGRADECIMIENTOS

Por fin llega el momento de escribir estas líneas, que significan el cierre de una larga etapa de la que me llevo mucho, académica y personalmente.

En primer lugar, quiero agradecer a mi director Fernando por su apoyo en todo este proceso, por guiarme durante todos estos años y por sacar siempre tiempo para revisar los papers, la tesis o comentar los avances.

No puedo dejar de mencionar también a mi otro director, José María, por su apoyo y confianza durante estos años, así como a David y Arturo.

Tengo que agradecer también a todos los compañeros del laboratorio, los que he visto pasar, los que me han acompañado desde el principio y los nuevos que se van incorporando, todos gente estupenda que han creado un ambiente de trabajo fantástico.

De todos ellos, tengo que hacer una mención especial a aquellos con los que más he trabajado: principalmente a los del equipo Timanfaya, con los que he aprendido y he disfrutado mucho y por supuesto al equipo ATLAS (Martín, Fran, Sergio, Ángel, Fernando y Abdulla) con quienes he compartido numerosas horas tanto en reuniones, como en los "cómodos y espaciosos" asientos del coche y junto a los que puedo decir que he construido un coche autónomo y forman, por lo tanto, una parte muy importante de esta tesis.

A todos mis amigos, por todos los momentos compartidos durante estos años y en especial a Javi, quien además de un gran amigo, también ha contribuido en parte al conocimiento que hay detrás de esta tesis con cursos, charlas y proyectos de redes.

Por último, la mención más especial es para mi familia, principalmente a mis padres por estar siempre ahí para lo que he necesitado y por apoyarme en todo y a Belén por acompañarme durante toda esta etapa, por todas las aventuras, viajes y emociones juntos y por todo lo que he aprendido de medicina, que casi me da para hacer otra tesis.

RESUMEN

La importancia de los vehículos autónomos en el sector del transporte durante las próximas décadas es ya un hecho. La implementación a gran escala de estos vehículos supondrá una serie de ventajas entre las que destacan una conducción más segura y por lo tanto una disminución de los accidentes de tráfico, una reducción de las emisiones y del consumo energético y un acortamiento de los tiempos de trayecto.

Sin embargo, existen todavía numerosos problemas por resolver de cara a una conducción completamente autónoma y generalizada. Todavía es necesario investigar en distintas tecnologías como percepción, control o navegación. Esta última área, es especialmente crítica ya que el correcto movimiento del vehículo depende de una localización y planificación de trayectorias robustas y fiables, entre otras tareas de navegación. Además, también es necesario estudiar la relación y el funcionamiento conjunto de todos los módulos de estas áreas junto con el hardware y entre ellas, relaciones definidas por la arquitectura.

El objetivo de esta tesis es: Por una parte, desarrollar una plataforma de investigación constituida por un vehículo autónomo completamente funcional, en la que se puedan probar distintos algoritmos relacionados con la conducción autónoma. Se investigarán las distintas arquitecturas posibles y se describirá la incorporada al vehículo desarrollado. Por otra parte, esta tesis presenta los avances realizados en el área de la navegación para mejorar la localización del vehículo en entornos mixtos donde métodos convencionales basados en GNSS o la correlación entre un mapa y las lecturas del LiDAR no obtienen resultados precisos, así como los avances en predicción del movimiento de otros vehículos, necesarios para una buena planificación de trayectorias. Además se investigará acerca de la interacción entre peatones y vehículos autónomos, y cómo mejorarla haciendo uso de distintas interfaces de comunicación.

Los resultados de los algoritmos desarrollados en localización y predicción de trayectorias han sido obtenidos con bases de datos públicas y comparados con métodos del estado del arte a los que superan en precisión, mientras que los resultados relativos a la interacción entre peatones y vehículos autónomos se ha evaluado mediante experimentos reales. Además, la arquitectura completa del vehículo ha sido probada en distintos experimentos que certifican su correcto funcionamiento.

PALABRAS CLAVE: Vehículo autónomo; Localización; predicción de trayectorias; Factores humanos; Arquitectura.

ABSTRACT

The importance of autonomous vehicles in the transportation sector over the next decades is already a fact. The large-scale implementation of these vehicles will bring several advantages, including safer driving and therefore a decrease of traffic fatalities, lower emissions and energy consumption, and shorter journey times.

However, there are still many issues to be solved for fully autonomous and widespread driving. A deeper research is still needed in different technologies such as perception, control and navigation. This last area is especially critical since the correct movement of the vehicle depends on precise localization and a robust and reliable path planning, among other navigation tasks. In addition, it is also necessary to study the relationships and the joint operation of all the modules of these areas together with the hardware and between them, relationships defined by the architecture.

The objective of this thesis is: On the one hand, to develop a research platform consisting of a fully functional autonomous vehicle, on which different algorithms related to autonomous driving can be tested. The different possible architectures will be investigated and the one incorporated in the developed vehicle will be described. On the other hand, this thesis presents the advances made in the area of navigation to improve vehicle localization in mixed environments where conventional methods based on GNSS or the correlation between a map and LiDAR readings do not obtain accurate results, as well as advances in predicting the movement of other vehicles, necessary for good trajectory planning. In addition, the interaction between pedestrians and autonomous vehicles will be studied, and how to improve it using different communication interfaces.

The results of the developed algorithms in localization and trajectory prediction have been obtained with public databases and compared with state-of-the-art methods which are outperformed in termos of accuracy, while the results related to the interaction between pedestrians and autonomous vehicles have been evaluated by means of real experiments. In addition, the complete vehicle architecture has been tested in different experiments certifying its correct operation.

KEYWORDS: Autonomous vehicle; Localization; Trajectory prediction; Human factors; Architecture.

PUBLICACIONES

Algunas ideas, tablas y figuras utilizadas han aparecido previamente en las siguientes publicaciones[1]:

CONTENIDO PUBLICADO

Artículos en revista

- **M.A. de Miguel**, F. Garcia y J.M. Armingol (2020). "Improved LiDAR Probabilistic Localization for Autonomous Vehicles Using GNSS." *Sensors*, vol. 20, no. 11, pgs 3145, DOI: 10.3390/s20113145 ([1]).

  Este artículo ha sido incluido parcialmente en esta tesis en los capítulos 2 y 4. La inclusión en la tesis del material de esta fuente se especifica en una nota de pie de página de cada capítulo en el que se incluye. El material de esta fuente en esta tesis no está señalado con medios tipográficos y referencias.

- **M.A. de Miguel**, F.M. Moreno, P. Marín-Plaza, A. Al-Kaff, M. Palos, R. Encinar y F. García (2020). "A Research Platform for Autonomous Vehicles Technologies Research in the Insurance Sector." *Applied Sciences*, vol. 10, num. 16, pgs 5655, 10.3390/app10165655 ([2]).

  Este artículo ha sido incluido parcialmente en esta tesis en los capítulos 2 y 3. La inclusión en la tesis del material de esta fuente se especifica en una nota de pie de página de cada capítulo en el que se incluye. El material de esta fuente en esta tesis no está señalado con medios tipográficos y referencias.

- E. Marti, **M.A. de Miguel**, y F. Garcia (2019). "A review of sensor technologies for perception in automated driving." *IEEE Intelligent Transportation Systems Magazine*, vol. 11, no. 4, pgs 94-108, DOI: 10.1109/MITS.2019.2907630 ([3]).

  Este artículo ha sido incluido parcialmente en esta tesis en los capítulos 2 y 3. La inclusión en la tesis del material de esta fuente se especifica en una nota de pie de página de cada capítulo en el que se incluye. El material de esta fuente en esta tesis no está señalado con medios tipográficos y referencias.

1 Los autores de los artículos se enumeran por orden de contribución relativa. El nombre del autor de esta tesis está marcado en negrita para describir fácilmente su participación.

- P. Marín-Plaza, D. Yagüe, F. Royo, **M.A. de Miguel**, F.M. Moreno, A. Ruiz-de-la-Cuadra, F. Viadero J. García, J.L. San-Román y J.M. Armingol (2021). "Project ARES: Driverless Transportation System. Challenges and Approaches in an Unstructured Road." *Electronics*, vol. 10, num. 15, pgs 1753, 10.3390/electronics10151753 ([4]).

  Este artículo ha sido incluido parcialmente en esta tesis en los capítulos 2 y 3. La inclusión en la tesis del material de esta fuente se especifica en una nota de pie de página de cada capítulo en el que se incluye. El material de esta fuente en esta tesis no está señalado con medios tipográficos y referencias.

Artículos publicados en congreso

- **M.A. de Miguel**, F. García, J.M. Armingol y R. Encinar (2019). "Autonomous vehicle architecture for high automation." *17th International Conference on Computer Aided Systems Theory EUROCAST 2019*, pgs 182-183 ([5]).

  Este artículo ha sido incluido parcialmente en esta tesis en los capítulos 2 y 3. La inclusión en la tesis del material de esta fuente se especifica en una nota de pie de página de cada capítulo en el que se incluye. El material de esta fuente en esta tesis no está señalado con medios tipográficos y referencias.

- **M.A. de Miguel**, D. Fuchshuber, A.Hussein y C. Olaverri-Monreal (2019). "Perceived pedestrian safety: Public interaction with driverless vehicles." *2019 IEEE Intelligent Vehicles Symposium (IV)*, pgs 90-95, DOI: 10.1109/IVS.2019.8814145 ([6]).

  Este artículo ha sido incluido parcialmente en esta tesis en los capítulos 2 y 6. La inclusión en la tesis del material de esta fuente se especifica en una nota de pie de página de cada capítulo en el que se incluye. El material de esta fuente en esta tesis no está señalado con medios tipográficos y referencias.

- W.M. Álvarez, **M.A. de Miguel**, F. García y C. Olaverri-Monreal (2019). "Response of vulnerable road users to visual information from autonomous vehicles in shared spaces." *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pgs 3714-3719, DOI: 10.1109/ITSC.2019.8917501 ([7]).

  Este artículo ha sido incluido parcialmente en esta tesis en los capítulos 2 y 6. La inclusión en la tesis del material de esta fuente se especifica en una nota de pie de página de cada capítulo en el que se incluye. El material de esta fuente en esta tesis no está señalado con medios tipográficos y referencias.

Capítulos de Libro

- **M.A. de Miguel**, F.M. Moreno, F. García, J.M. Armingol y R. Encinar (2019). "Autonomous vehicle architecture for high automation." *International Conference on Computer Aided Systems Theory*, Springer, pgs 145-152, ISBN: 978-3-030-45096-0 ([8]).

  Este artículo ha sido incluido parcialmente en esta tesis en los capítulos 2 y 3. La inclusión en la tesis del material de esta fuente se especifica en una nota de pie de página de cada capítulo en el que se incluye. El material de esta fuente en esta tesis no está señalado con medios tipográficos y referencias.

MATERIAL DE TERCEROS

Material NO escrito o co-escrito por el autor de la tesis.

- Las figuras 2.2 (©2007 IEEE) de [9], 2.12 (©2018 IEEE) [10], 2.10 (©2018 IEEE) [11], 2.11d (©2020 IEEE) de [12] se han incluido con el permiso del IEEE, titular de los derechos de autor.
- La figura 2.11b de [13] no requieren un permiso especial para reutilizar todo o parte del artículo publicado por MDPI. En el caso de los artículos publicados bajo una licencia Creative Common CC BY 4.0 de acceso abierto, cualquier parte del artículo puede ser reutilizada sin permiso siempre que el artículo original sea claramente citado. La reutilización de un artículo no implica el respaldo de los autores o de MDPI.

Material escrito o co-escrito por el autor de la tesis

- Las figuras 4.1, 4.2 y 4.3 de [1] y 2.15 y 2.16 de [2] no requieren un permiso especial para reutilizar todo o parte del artículo publicado por MDPI. En el caso de los artículos publicados bajo una licencia Creative Common CC BY 4.0 de acceso abierto, cualquier parte del artículo puede ser reutilizada sin permiso siempre que el artículo original sea claramente citado. La reutilización de un artículo no implica el respaldo de los autores o de MDPI.
- Las figuras 2.4 (©2019 IEEE) de [3] y 6.1 y 6.2 (©2019 IEEE) de [6] se han incluido con el permiso del IEEE, titular de los derechos de autor.

OTROS MÉRITOS DE INVESTIGACIÓN

Artículos de congreso

- J. Carmona, F. García, **M.A. de Miguel**, A. de la Escalera y J.M. Armingol (2016). "Analysis of Aggressive Driver Behaviour us-

ing Data Fusion." *VEHITS 2016*, pgs 85-90, 10.5220/0005805700850090 ([14]).

- J. Carmona, **M.A. de Miguel**, D. Martín, F. García y A. de la Escalera (2016). "Embedded system for driver behavior analysis based on GMM." *2016 IEEE Intelligent Vehicles Symposium (IV)*, pgs 61-65, 10.1109/IVS.2016.7535365 ([15]).

Becas

- Beca del ministerio de universidades de España en el programa de Formación de Profesorado Universitario (FPU) desde octubre de 2017 hasta marzo de 2022.
- Beca Colaboración predoctoral concedida por el Ministerio de Educación Cultura y Deporte de España de Octubre 2014 a Junio 2015

Participación en congresos

- Participación en la demostración de vehículos autónomos organizada en el congreso ICVES 18 (del 12 al 14 de septiembre de 2018).
- Participación en la demostración de vehículos autónomos organizada en el congreso IROS 18 (del 1 al 5 de octubre de 2018).
- Asistencia y presentación en el congreso EUROCAST 19 (del 17 al 22 de febrero de 2019).

Trabajos de fin de grado tutorizados

- J. Rueda, "Generación de entorno de simulación para vehículo autónomo", Trabajo Fin de Grado (TFG), Grado en Ingeniería Electrónica Industrial y automática, Universidad Carlos III de Madrid, Jul. 2020.
- A. Siam, "Autonomous Vehicle Control", Trabajo Fin de Grado (TFG), Bachelor of Science in Mechatronics Engineering Degree, German University in Cairo, Jul. 2020.
- G. Hidalgo, "Aprendizaje automático para vehículos autónomos", Trabajo Fin de Grado (TFG), Grado en Ingeniería Electrónica Industrial y automática, Universidad Carlos III de Madrid, Sep. 2020.
- S. Muñoz, "Diseño de algoritmos de control de nivel bajo en vehículo inteligente", Trabajo Fin de Grado (TFG), Grado en Ingeniería Electrónica Industrial y automática, Universidad Carlos

III de Madrid, Oct. 2019. [Online]. Disponible: https://e-archivo.uc3m.es/handle/10016/30243

- A. Nashaat, "Adaptive Lateral Geometric Control of Autonomous Vehicles", Trabajo Fin de Grado (TFG), Bachelor of Science in Mechatronics Engineering Degree, German University in Cairo, Jul. 2019.
- M. Khedr, "Autonomous vehicle speed control using PID and Fuzzy logic controllers", Trabajo Fin de Grado (TFG), Bachelor of Science in Mechatronics Engineering Degree, German University in Cairo, Jul. 2019.
- M. Abdeen, "Trajectory Velocity Planning and Control for Automated Vehicles", Trabajo Fin de Grado (TFG), Bachelor of Science in Mechatronics Engineering Degree, German University in Cairo, Jul. 2018.

ÍNDICE DE CONTENIDOS

1 INTRODUCCIÓN 1
1.1 Motivación . . . 1
1.1.1 Accidentes de tráfico . . . 3
1.1.2 Contaminación . . . 4
1.1.3 Atascos . . . 5
1.1.4 Vehículos autónomos . . . 5
1.2 Objetivos . . . 8
1.3 Estructura . . . 9
2 ESTADO DEL ARTE 11
2.1 Historia de plataformas de conducción autónoma . . . 11
2.1.1 Primeros trabajos (1980-2000) . . . 11
2.1.2 Pruebas de viabilidad (2000-2010) . . . 12
2.1.3 Desarrollo de productos comerciales (2010-2022) 13
2.2 Arquitecturas para vehículos autónomos . . . 17
2.2.1 Arquitectura hardware . . . 17
2.2.2 Arquitectura software . . . 19
2.3 Bases de datos . . . 23
2.3.1 Bases de datos de localización . . . 24
2.3.2 Bases de datos de predicción de trayectorias . . . 24
2.4 Sensores . . . 25
2.4.1 Cámara . . . 25
2.4.2 Radar . . . 28
2.4.3 LiDAR . . . 29
2.4.4 Uso de sensores para percepción . . . 31
2.5 Navegación . . . 33
2.5.1 Localización . . . 33
2.5.2 Seguimiento de trayectoria . . . 38
2.5.3 Predicción de trayectorias . . . 41
2.5.4 Planificación . . . 44
2.6 Interfaz Hombre Máquina . . . 47
3 ARQUITECTURA 49
3.1 Arquitectura hardware . . . 49
3.1.1 Plataforma . . . 49
3.1.2 Procesamiento . . . 50
3.1.3 Comunicación . . . 52
3.1.4 Potencia . . . 52
3.2 Sensores de percepción . . . 53
3.2.1 Elección de sensores . . . 53
3.2.2 Disposición y características de los sensores . . . 54
3.2.3 Calibración de sensores . . . 57
3.3 Arquitectura software . . . 58
3.3.1 Adquisición de datos . . . 60

3.3.2 Localización . . . 62
3.3.3 Percepción . . . 62
3.3.4 Control y planificación . . . 64
3.3.5 Interfaz de usuario gráfica . . . 68
3.3.6 Interfaz hombre máquina . . . 69
3.3.7 ROS . . . 69
3.3.8 Seguridad . . . 69
3.3.9 Simulación . . . 70
3.4 Resultados y caso de usos . . . 71
3.4.1 Casos de uso . . . 71
3.4.2 Carga computacional . . . 72
3.4.3 Ancho de banda . . . 73
3.4.4 Cobertura de los sensores . . . 74
3.4.5 Control y funcionamiento . . . 79
4 LOCALIZACIÓN 83
4.1 Localización basada en GNSS . . . 83
4.1.1 Filtrado de la localización GNSS . . . 84
4.2 Localization basada en LiDAR y mapa . . . 86
4.2.1 Generación del mapa georreferenciado . . . 86
4.2.2 Configuración de los parámetros del AMCL . . . 87
4.3 Combinación de LiDAR y GNSS . . . 89
4.3.1 Probabilidad del LiDAR . . . 89
4.3.2 Estimación de probabilidad del sensor GNSS . . . 91
4.3.3 Cálculo del peso de cada partícula . . . 91
4.3.4 Generación de nuevas partículas . . . 92
4.4 Resultados . . . 93
4.4.1 Base de datos y metodología . . . 93
4.4.2 Mapa vacío . . . 94
4.4.3 Escenarios con dificultades para la localización con GNSS . . . 95
4.4.4 Escenarios mixtos . . . 96
5 PREDICCIÓN DE TRAYECTORIAS 101
5.1 Preprocesamiento de datos . . . 102
5.1.1 Preprocesamiento y normalización . . . 102
5.1.2 Clasificación de las maniobras . . . 102
5.1.3 División de la base de datos . . . 103
5.2 Arquitectura del modelo . . . 103
5.2.1 Modelo VAE . . . 104
5.2.2 Modelo de predicción . . . 105
5.2.3 Entrenamiento del modelo completo . . . 106
5.2.4 Análisis de las variables latentes . . . 107
5.3 Resultados . . . 109
5.3.1 Métricas de evaluación . . . 110
5.3.2 Comparación con otros modelos . . . 110
5.3.3 Obtención y evaluación de los resultados . . . 112
6 INTERACCIÓN ENTRE PEATÓN Y VEHÍCULO AUTÓNOMO 115

6.1 Estudio de distintas interfaces hombre máquina 115
6.1.1 Plataformas 116
6.2 Experimentos 117
6.2.1 Toma de datos 118
6.3 Resultados 119
6.3.1 Datos subjetivos cualitativos 119
6.3.2 Datos objetivos de análisis de secuencias 122
6.3.3 Discusión 125
7 CONCLUSIONES Y TRABAJOS FUTUROS 127
7.1 Conclusiones 127
7.2 Trabajos futuros 128

BIBLIOGRAFÍA 131

ÍNDICE DE FIGURAS

Figura 1.1 Evolución del número de vehículos de pasajeros en uso a nivel mundial desde 2005 hasta 2015 [16]. 2
Figura 1.2 Comparación de la ratio de accidentes de tráfico por cada 100.000 habitantes de distintas regiones en 2013 y 2016. 3
Figura 1.3 Comparación del parque automovilístico con respecto al número de accidentes para cada región definida por la OMS [16]. 4
Figura 1.4 Evolución de la proporción de vehículos con ADAS entre 2018 y 2023. 6
Figura 2.1 Interior y exterior del vehículo VaMoRs. . . . 12
Figura 2.2 Vehículo Stanley del equipo de la universidad Stanford. [9] 13
Figura 2.3 Vehículo autónomo de la empresa Waymo (Fuente: Waymo). 15
Figura 2.4 Cronología de las demostraciones más relevantes junto a la arquitectura sensorial del Vehículo autónomo (VA) presentado. 16
Figura 2.5 Ejemplo de arquitectura centralizada en un VA. 18
Figura 2.6 Ejemplo de arquitectura descentralizada en un VA. 18
Figura 2.7 Ejemplo de arquitectura distribuida en un VA. 19
Figura 2.8 Ejemplo de arquitectura centralizada en un VA. 20
Figura 2.9 Ejemplo de arquitectura descentralizada en un VA. 21
Figura 2.10 Comparación entre las detecciones de la base de datos NGSIM (arriba) y highd (abajo)[11]. . 25
Figura 2.11 Ejemplos de distintos tipos de cámaras con información tridimensional o temporal. 27
Figura 2.12 Imagen de la escena (izquierda) y vista de pájaro de la información del radar (derecha). Los vehículos detectados aparecen rodeados en ambas imágenes [10]. 28
Figura 2.13 Mapa del entorno generado con un sensor LiDAR. 30
Figura 2.14 Esquema del controlador Pure Pursuit. La trayectoria a seguir en verde, el arco de circunferencia en azul. 39

Figura 2.15 Planificación global (rojo) sobre un grafo generado con OSM desde la localización (verde) hasta el objetivo (azul). 45
Figura 2.16 Planificación local en simulación. 46
Figura 3.1 Plataforma de VA basada en un Mitsubishi iMiev. 50
Figura 3.2 Esquema de las conexiones entre los distintos dispositivos desde la batería hasta los sensores y ordenadores. 53
Figura 3.3 Planta y alzado del modelo virtual del vehículo donde se puede apreciar la colocación de los sensores. 57
Figura 3.4 Esquema de las relaciones entre los distintos módulos de software incluidos en la arquitectura, así como el tipo de datos de entrada y salida. 60
Figura 3.5 Captura de la simulación en la que se comprueba el rango de cada sensor en función de si detecta o no el bloque. 74
Figura 3.6 Área de detección de cada LiDAR para la configuración con los LiDAR laterales sólo inclinados en el eje *roll*. 76
Figura 3.7 Área de detección de cada LiDAR para la configuración con los LiDAR laterales inclinados en el eje *roll* y *yaw*. 77
Figura 3.8 Visualización de la detección de peatones con la cámara y medición de su distancia máxima y mínima en la nube de puntos. 78
Figura 3.9 Evolución de la distancia a la trayectoria a seguir determinada por centro del carril a lo largo de la ruta. 80
Figura 3.10 Señales de control del volante grabadas (verde) y generadas por el VA (rojo). 80
Figura 4.1 Comparación entre el método propuesto sin mapa con la localización con GNSS filtrada. (GNSS está representado como constante para una mejor visualización). 95
Figura 4.2 Error de localización de los tres métodos descritos utilizando la secuencia 36 de KITTI. (El error del método GNSS se representa como una línea constante del valor de la media para una mejor visualización. 97
Figura 4.3 Error de localización de los tres métodos descritos utilizando la secuencia 34 de KITTI. (El error del método GNSS se representa como una línea constante del valor de la media para una mejor visualización. 97

Figura 5.1 Representación de las zonas alrededor del vehículo analizado (azul). Como ejemplo de cómo se obtiene d_i, se han colocado dos vehículos alrededor (amarillo). 103
Figura 5.2 Arquitectura completa del modelo completo. . 104
Figura 5.3 Reconstrucción del modelo VAE_{surr} (en rojo) de algunas variables de entrada (en verde). El eje X representa el tiempo, mientras que el eje Y es el valor normalizado de la variable. 108
Figura 5.4 Representación de las dos dimensiones más relevantes del hiperespacio latente para VAE y AE. Los colores muestran la maniobra futura: mantener el carril (rojo), cambiar al carril izquierdo (verde) y cambiar al carril derecho (azul). 109
Figura 5.5 Reconstrucción del modelo VAE (línea verde) para distintos pares de valores del espacio latente (Dimensión Latente 0, Dimensión Latente 1). Los límites del carril están representados por líneas grises. 110
Figura 5.6 La predicción del modelo (rojo), *ground truth* de la trayectoria pasada (azul rayado) y de la trayectoria futura (azul oscuro con puntos separados cada segundo) para todos los vehículos de la escena. Todas las trayectorias generadas por el modelo aparecen con un valor de transparencia alto para visualizar mejor la varianza del método y comparar las varianzas de ambas figuras. Los valores de la trayectoria media de la predicción aparecen con puntos rojo oscuro separados cada segundo. 113
Figura 6.1 Imágenes propuestas para mostrar detección o no detección del peatón. 116
Figura 6.2 Interacción entre un peatón y el VA junto con la detección de la posición del peatón. 118
Figura 6.3 Porcentaje de dudas al cruzar con y sin interfaz en la escala Likert. 121
Figura 6.4 Resultados del contacto visual al cruzar en la escala de Likert. 122

ÍNDICE DE TABLAS

Tabla 1.1 Resumen de las capacidades de cada nivel de automatización según SAE [32]. 8
Tabla 2.1 Tipos de información en el dominio de un VA. 31
Tabla 2.2 Idoneidad de los sensores para distintas tareas de percepción. 32
Tabla 2.3 Robustez de los sensores ante distintos factores ambientales. 33
Tabla 3.1 Coordenadas teóricas de traslación y rotación de los sensores cámara y LiDAR laterales con respecto al LiDAR de 32 planos. 58
Tabla 3.2 Resumen del procesamiento requerido de los algoritmos con mayor carga. 73
Tabla 3.3 Resumen del ancho de banda de los sensores y módulos de software más relevantes. 74
Tabla 4.1 Valores utilizados de todos los parámetros de AMCL. 90
Tabla 4.2 Comparación entre el método propuesto y AMCL en ausencia de GNSS. 96
Tabla 4.3 Evaluación de los errores del método propuesto para distintos grados de precisión de GNSS en la secuencia 34. 98
Tabla 5.1 Valores de EAM y RECM longitudinal y lateral para los distintos métodos. 111
Tabla 6.1 Resumen de los resultados de las encuestas. . 120
Tabla 6.2 Valores estadísticos del test chi cuadrado (χ^2, 1 grado de libertad) para el comportamiento del peatón en función de la imagen mostrada. . . 123
Tabla 6.3 Distancia entre el peatón y el vehículo y TDC al cruzar para distintas imágenes mostradas. . 124
Tabla 6.4 Efecto del contacto visual en la interacción con el VA. 124
Tabla 6.5 Resultados del análisis de los vídeos grabados. 125

ACRÓNIMOS

ACP Análisis de Componentes Principales

ADAS Sistemas Avanzados de Asistencia a la Conducción o *Advanced Driver Assistance Systems*

AE *Auto Encoder*

AMCL *Adaptive Monte Carlo Localization*

CNN *Convolutional Neural Networks*

CPU Unidad Central de Procesamiento o *Central Processing Unit)*

cVAE VAE condicional

DE Desviación Estándar

DL *Deep Learning*

DRDO Detección y Respuesta ante la Detección de un Objeto

EAM Error Absoluto Medio

ECM Error Cuadrático Medio

FMCW Onda Continua Modulada en Frecuencia o *Frequency-Modulated Continuous Wave*

GAN *Generative Adversarial Network*

GNSS Sistema Global de Navegación por Satélite o *Global Navigation Satellite System*

GPU Unidad de Procesamiento Gráfico o *Graphics Processing Unit)*

GUI Interfaz Gráfica de Usuario o *Graphics User Interface*

HMI Interfaz Hombre Máquina o *Human-Machine Interface*

IEA *International Energy Agency*

IMU Unidad de Medida Inercial o *Inertial Measurement Unit*

IR Infrarrojo

KLD Distancia Kullback-Leibler

LiDAR *Light Detection and Ranging*

LOAM *Lidar Odometry and Mapping*

LSTM *Long Short-Term Memory*

MCL *Monte Carlo Localization*

MPC Control Predictivo por Modelo o *Model Predictive Control*

NAHSC *National Automated Highway Systems Consortium*

NGSIM *Next Generation Simulation*

NHTSA *National Highway Traffic Safety Administration*

NTRIP *Networked Transport of RTCM via Internet Protocol*

OMS Organización Mundial de la Salud

OPA *Optical Phased Array*

OSM *Open Street Maps*

PIB Producto Interior Bruto

RECM Raíz del Error Cuadrático Medio

RNN *Recurrent Neural Networks*

ROS *Robot Operating System*

RTK *Real Time Kinematic*

SAE *Society of Automotive Engineers*

SAI Sistema de Alimentación Ininterrumpida

SLAM Localización y mapeado simultáneo o *Simultaneous Localization And Mapping*

TCD Tarea de conducción dinámica

TTC Tiempo hasta la colisión

UKF *Unscented Kalman Filter*

VA Vehículo autónomo

VAE *Variational Autoencoder*

1 INTRODUCCIÓN

Hoy en día, el transporte privado por carretera forma parte de nuestras vidas, siendo necesario por numerosas personas casi a diario, desde un uso por necesidad para desplazarse al lugar de trabajo o a adquirir bienes, hasta como una parte del ocio.

Las personas constituyen todavía una parte esencial en el control de estos vehículos, sin embargo, en los últimos años, los avances de la tecnología están llegando a los automóviles, desarrollando cada vez Sistemas Avanzados de Asistencia a la Conducción o *Advanced Driver Assistance Systems* (ADAS) más avanzados que automatizan o facilitan cada vez más las tareas del conductor.

Sin embargo, todavía existen numerosos problemas que deben resolverse de cara al desarrollo de un vehículo completamente autónomo. La conducción de un vehículo es una tarea compleja que requiere de una habilidad adquirida con el tiempo y la práctica que permite adaptarse a distintos tipos de entorno. Las características en las que un vehículo circula son muy diversas, incluyendo distintas condiciones climatológicas (sol en el horizonte, niebla, etc.) distintas localizaciones (desde autopistas hasta carreteras sin asfaltar), lo que puede dificultar las tareas de percepción del entorno.

Por otro lado, aunque existen normas de circulación bien definidas, su aplicación a la conducción real no es trivial y requiere de una interpretación y una capacidad de decisión ante situaciones nunca antes vistas, que pueden resultar complejas para un VA (puede ser necesario seguir indicaciones de un guardia de tráfico y para ello infringir alguna norma de circulación, o pueden aparecer en la vía elementos nunca antes vistos ante los que reaccionar adecuadamente).

El contenido de esta tesis pretende avanzar en la investigación sobre los vehículos autónomos, enfocándose en la arquitectura que hace posible que un vehículo de este tipo sea capaz de navegar de forma autónoma. Esta investigación describe los aspectos a tener en cuenta a la hora de diseñar un VA, además de aportar nuevos métodos que mejoren algunas de las partes más críticas de la navegación del vehículo.

1.1 MOTIVACIÓN

Se entiende por transporte la actividad de desplazar bienes o personas entre dos localizaciones mediante el uso de un vehículo que circula por una infraestructura.

En los últimos siglos, la industrialización y el aumento de los desplazamientos humanos han hecho que esta actividad se haya convertido en una de las que más crecimiento ha experimentado. Según avanza la sociedad, las necesidades de transporte van creciendo, siendo necesario alcanzar lugares más remotos y disminuir el tiempo que se requiere para llegar al destino deseado.

Es un hecho que los medios de transporte forman parte de nuestra actividad diaria y están integrados completamente en la sociedad hasta tal punto que, hoy en día en la mayoría de los núcleos de población, las infraestructuras de transporte (carreteras, vías de tren, etc.) están completamente integradas con las viviendas y otros espacios (comercios, oficinas, etc.).

Entre los distintos tipos de transporte, el privado por carretera es uno de los que más ha crecido considerablemente en los últimos años [16, 17]. La Figura 1.1 muestra la tendencia en aumento del número de vehículos de pasajeros a nivel mundial, lo que hace necesario no sólo destacar las ventajas, si no también analizar los posibles problemas existentes, que se puedan agravar por un aumento del número de vehículos en circulación, así como los que puedan surgir en un futuro por esta causa.

Entre los problemas actuales más preocupantes se encuentran: los accidentes de tráfico, la contaminación y los atascos, los cuales se desarrollarán a continuación.

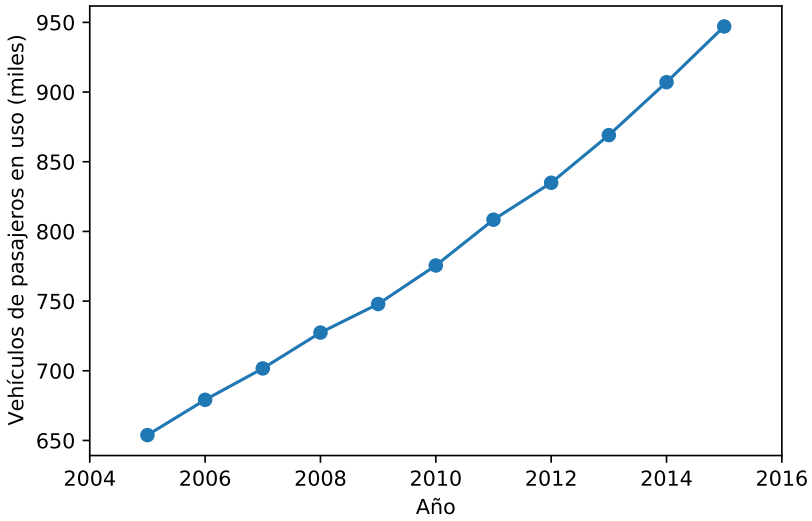


Figura 1.1: Evolución del número de vehículos de pasajeros en uso a nivel mundial desde 2005 hasta 2015 [16].

1.1.1 *Accidentes de tráfico*

Uno de los problemas más importantes derivados de los vehículos son los accidentes de tráfico. Se estima que cada año pierden la vida en torno a 1.3 millones de personas y, entre 20 y 50 millones más sufren daños permanentes a causa de estos accidentes.

El correcto desempeño de la conducción, libre de accidentes, depende de diversos factores, tales como el estado del conductor (nivel de concentración, afectaciones por sustancias, somnolencia, etc.), el estado del entorno (carretera bien asfaltada, curvas peligrosas, situaciones con mucho tráfico), o del clima (Lluvia, niebla, hielo, etc.).

Sin embargo, de todas estas causas, un error humano en la conducción es la más común. Según un informe de *National Highway Traffic Safety Administration* (NHTSA), esta causa está involucrada en entre el 94% y el 96% de todos los accidentes ocurridos en EEUU [18, 19].

Por otra parte, el problema de los accidentes de tráfico está marcado por el distinto desarrollo económico de cada sociedad. Mientras que la media de muertes por accidentes de tráfico es de 18.2 por cada 100.000 habitantes en todo el mundo, como se puede apreciar en la Figura 1.2, existe una variación significativa entre distintas regiones. Mientras que Europa y América cuentan con las ratios más bajas (9.3 y 15.6), en regiones más pobres como África o el Sudeste Asiático se encuentran valores mucho más elevados (26.6 y 20.7).

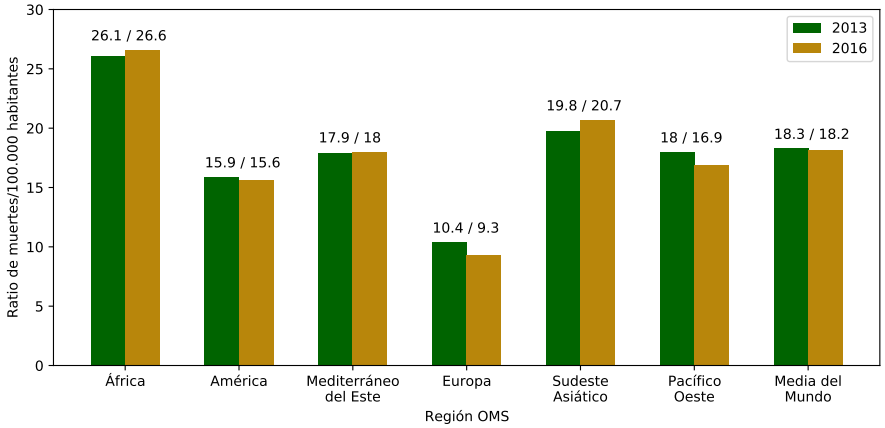


Figura 1.2: Comparación de la ratio de accidentes de tráfico por cada 100.000 habitantes de distintas regiones en 2013 y 2016.

Estos datos chocan con los del parque automovilístico en uso en estas regiones, ya que las que menor ratio de accidentes tienen, son a la vez las que cuentan con un mayor número de vehículos por habitante, tal y como se puede observar en la Figura 1.3. Este contraste puede explicarse por las malas condiciones de las carreteras en estas zonas, así como por los vehículos, más antiguos y que no cumplen los estándares de seguridad.

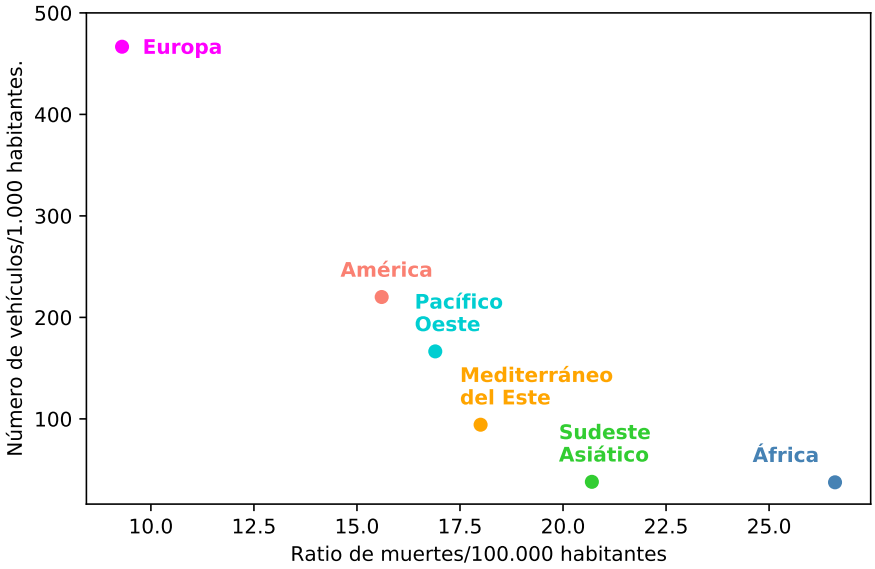


Figura 1.3: Comparación del parque automovilístico con respecto al número de accidentes para cada región definida por la OMS [16].

Además del enorme coste social y humano que provocan los accidentes de tráfico, también hay que mencionar el coste económico que suponen. Estos accidentes causan unas pérdidas considerables tanto a las personas afectadas, como a sus familias y a los gobiernos. Estos costes surgen del tratamiento requerido y el descenso de productividad de las personas afectadas, así como de los familiares que deben invertir tiempo del trabajo o estudios en su cuidado. Se estima este coste para la mayoría de los países en el 3% de su Producto Interior Bruto (PIB) [20].

1.1.2 *Contaminación*

Otro preocupante problema para la sociedad es la polución derivada del extensivo uso de estos vehículos.

Los vehículos de combustión, en especial los que tienen motores diésel, expulsan en el proceso de combustión una gran cantidad de partículas perjudiciales para la salud, además de gases de efecto invernadero.

Los vehículos eléctricos por otra parte, aunque reducen significativamente la emisión de contaminantes, no la eliminan por completo, ya que parte de la energía que utilizan proviene de fuentes no renovables. Además, el proceso de fabricación de cualquier vehículo va ligado a la utilización de una gran cantidad de energía y su consiguiente contaminación emitida.

La *International Energy Agency* (IEA) estima que el transporte es el responsable de un 24% de las emisiones globales de gases de

efecto invernadero y esta cantidad va en aumento a un ritmo de aproximadamente el 1.9% desde el año 2000 [21].

Este problema se acentúa en los núcleos urbanos donde existe una mayor concentración de vehículos. Según la Organización Mundial de la Salud (OMS), el 99% de la población global respira aire que supera los límites de contaminantes establecidos por esta organización, lo que provoca un aumento en la mortalidad por enfermedades cardio-respiratorias y en el riesgo de enfermedades respiratorias no alérgicas [22]. Esta organización estima que 4,2 millones de muertes al año están relacionadas con la contaminación ambiental, reduciendo la esperanza de vida en un año en muchas ciudades [23, 24].

1.1.3 *Atascos*

Los atascos o congestión de tráfico son otro problema relevante que sufren numerosas ciudades de todo el mundo. Entre los inconvenientes que provocan se encuentran el aumento del uso de combustible y por lo tanto de las emisiones, el encarecimiento del transporte, y la pérdida de tiempo personal entre otros. En las grandes ciudades, este problema empeora, ya que en ciudades como Londres, INRIX estima que cada conductor perdió de media en 2021, 148 horas en atascos, generando un coste individual de 1.211£ al año [25].

1.1.4 *Vehículos autónomos*

Como se ha explicado en la Sección 1.1.1, las personas no son idóneas para realizar algunos tipos de tareas necesarias en la conducción como son:

- Tareas rutinarias: Como viajes largos por carreteras sin apenas curvas
- Estimación de parámetros del entorno: Como distancias o velocidad relativa de otros vehículos.
- Visión en condiciones adversas: Con poca luz, baja visibilidad por niebla o polvo, etc.

Para minimizar el riesgo producido por estos factores, en los últimos años, se han desarrollado numerosos sistemas para aumentar la seguridad de estos vehículos. Dichos sistemas han ido evolucionando y aumentando su complejidad desde los primeros sistemas diseñados para reducir riesgos ante una situación peligrosa (como el ABS y el ESP, que aumentan la seguridad ante una frenada o un giro busco), hasta los sistemas desarrollados en la actualidad, orientados en prevenir una situación peligrosa (como el sistema de mantenimiento de carril) o en facilitar las tareas de conducción (detección de señales de tráfico o sistemas de ayuda al estacionamiento).

Estos sistemas de seguridad han demostrado ser útiles, hecho que se puede comprobar en estudios como el realizado por LexisNexis, donde los vehículos equipados con ADAS mostraron una reducción del 27% en los daños personales y del 19% en daños materiales [26]. Otro ejemplo es el estudio presentado en [27], donde el sistema que monitoriza el punto muerto de un vehículo consigue reducir el número de accidentes un 14%, esto quiere de decir que, si en cada coche de EEUU posterior a 2015 se hubiese equipado este sistema, se habrían evitado alrededor de 50.000 accidentes.

Tales son los beneficios de este tipo de sistemas que según [28], se estima que la proporción de vehículos con algunos de estos sistemas (como cámaras traseras o detectores de colisión frontal) va a aumentar en más de un 20% en un periodo de 5 años tal y como se observa en la Figura 1.4.

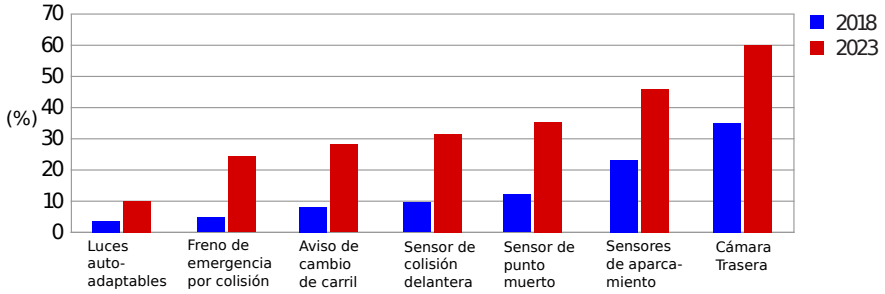


Figura 1.4: Evolución de la proporción de vehículos con ADAS entre 2018 y 2023.

Todos estos hechos apuntan a que el futuro de la conducción va a estar basado en sistemas de ayuda a la conducción que facilitarán y automatizarán cada vez más esta tarea. La tendencia por lo tanto apunta a descargar cada vez de más tareas al conductor hasta que este sea completamente prescindible del bucle de control de un vehículo.

Los vehículos autónomos presentan una serie de ventajas, muchas de las cuales mitigan o solucionan completamente algunos de los problemas planteados en los apartados anteriores. Algunas de estas ventajas son:

- Aumento de la seguridad del transporte por carretera al reducir significativamente el número de accidentes, muchos de ellos producidos actualmente por errores humanos.
- Reducción del estrés y aumento de la productividad. Estos vehículos permitirán a los pasajeros descansar o trabajar mientras viajan, en lugar de conducir.
- Aumento de la movilidad de personas que no pueden conducir, ya sea por no tener la licencia necesaria o por algún problema médico o de salud.

- Reducción de atascos y contaminación aprovechando las posibilidades que ofrece la comunicación entre vehículos (V2V) o entre vehículos e infraestructura (V2I).

- Reducción de costes con la posibilidad de implantar flotas de taxis autónomos compartidos que aumenten la eficiencia del transporte y no haga necesario en muchos casos la adquisición de un vehículo privado.

Sin embargo, la implantación por completo del VA en la sociedad es un proceso lento en el que actualmente apenas se pueden encontrar unos pocos modelos de vehículos comerciales que exhiben algunas de las características de un VA, pero que todavía distan mucho de uno.

Existen distintas organizaciones que han estudiado los distintos grados de automatización que puede tener un vehículo autónomo, clasificándolos en niveles (OICA [29], NHTSA [30] o BASt [31]) Sin embargo, actualmente la clasificación más utilizada, adoptada incluso por algunas de las organizaciones mencionadas anteriormente es la realizada por la *Society of Automotive Engineers* (SAE) [32]. Esta organización realiza una clasificación de seis niveles que comienzan en el 0 (vehículo sin ningún tipo de automatización), hasta el 5 (vehículo completamente autónomo). La Tabla 1.1 muestra un resumen de estos niveles, e indica para cada uno, quien realiza cada tarea de conducción (el vehículo o el conductor), y si el dominio de entornos o situaciones en los que pueden funcionar las tareas automatizadas (Dominio) son limitados o por el contrario funciona en cualquier entorno (Todos). Las tareas de conducción que aparecen en la tabla se definen de la siguiente manera:

- Tarea de conducción dinámica (TCD) lateral y longitudinal (Lat/-Long): La TCD realiza el control lateral y longitudinal del vehículo (niveles del 2 al 5). El sistema puede requerir que el conductor colabore en una de las TCD, lateral o longitudinal (nivel 1) o que las realice por completo la TCD (nivel 0).

- Detección y Respuesta ante la Detección de un Objeto (DRDO): Engloba las tareas de monitorización del entorno (detectar, reconocer y clasificar objetos y eventos, preparando la respuesta necesaria). Esta tarea puede ser realizada por el vehículo (niveles del 3 al y 5) o por el conductor (niveles del 0 al 2).

- Recuperación ante un fallo de la TCD (TCD Rec.): Ante un fallo en el que la TCD no sea capaz de mantener el control del vehículo, pude ser necesario que el conductor tome el control, por lo que debe supervisar la TCD en todo momento (niveles del 0 al 3). Para los niveles 4 y 5, la TCD es suficientemente robusta para que no sea necesario un conductor alerta para tomar el control en caso de fallo.

Nivel	Definición	TCD Lat/Long	DRDO	TCD Rec.	Dominio
0	Sin automatización				-
1	Asistencia al conductor				Limitado
2	Conducción autónoma parcial				Limitado
3	Conducción autónoma condicional				Limitado
4	Alta automatización de la conducción				Limitado
5	Conducción autónoma completa				Todos

Tabla 1.1: Resumen de las capacidades de cada nivel de automatización según SAE [32].

1.2 OBJETIVOS

Esta tesis está englobada en el contexto de la arquitectura de un VA. Entre los distintos módulos que componen la arquitectura de los vehículos autónomos, está enfocada principalmente en la elección de sensores y en los módulos de navegación (localización, planificación y factores humanos). El principal objetivo es desarrollar una plataforma de investigación constituida por un VA completamente funcional, en la que se puedan probar distintos algoritmos relacionados con la conducción autónoma. Las características principales de la plataforma a desarrollar están determinadas por un comportamiento básico de un vehículo autónomo, capaz de circular entre dos puntos distintos sin intervención humana, realizando de forma automática todas las tareas de conducción. El vehículo debe circular a una velocidad de al menos $30 km/h$ por un entorno de carretera urbana conocido que puede haber sido previamente mapeado. Además, debe ser capaz de detectar otros agentes de la carretera (otros vehículos o peatones) y responder de forma adecuada en cada situación en función de su situación o velocidad.

Para ello, se definen las siguientes tareas:

- Estudiar qué configuración de sensores es la adecuada para un VA en base a sus características
- Definir la arquitectura de la plataforma en base al estudio de sensores y a las especificaciones del vehículo y determinar qué algoritmos utilizar para cada módulo del vehículo y qué algoritmos es necesario mejorar.
- Implementar y adaptar los algoritmos de localización para los entornos en los que circulará el vehículo, así como describir la configuración y funcionamiento del módulo de localización.
- Implementar y adaptar los algoritmos de predicción de trayectorias que se utilizarán en el módulo de planificación de trayectorias.
- Estudiar la interacción entre el VA y los peatones para desarrollar una interfaz de comunicación entre ambos.
- Analizar los resultados de la arquitectura propuesta junto con los módulos definidos y describir los casos de uso para el vehículo.

La consecución de estos objetivos dará como resultado una plataforma constituida por un vehículo autónomo donde poder investigar y avanzar en esta tecnología probando nuevos algoritmos de cualquiera de los módulos de los que está compuesta.

1.3 ESTRUCTURA

Este documento está estructurado en siete capítulos, empezando con este. Cada capítulo cuenta en su comienzo con una breve introducción al tema tratado en él. A continuación, se resume brevemente el contenido de cada capítulo:

- Capítulo 1 ha servido como introducción a la tesis, describiendo la problemática de los vehículos autónomos, la motivación para el desarrollo de vehículos autónomos y se han fijado los objetivos.
- Capítulo 2 comienza con una reseña histórica sobre la evolución de los vehículos autónomos haciendo hincapié en sus arquitecturas a lo largo de la historia, e incluye una revisión del estado del arte de cada parte de un VA.
- Capítulo 3 describe y la arquitectura hardware, software y de sensores del VA de esta tesis y motiva la elección de cada componente de ella.
- Capítulo 4 desarrolla el sistema de localización utilizado en el vehículo incluyendo una descripción de un nuevo algoritmo

de localización desarrollado para mejorar la precisión en la localización de un VA, así como los resultados de este método

- Capítulo 5 recoge los avances logrados en el módulo de predicción de trayectorias, describiendo un nuevo modelo de arquitectura que mejora estas predicciones.
- Capítulo 6 presenta un estudio acerca de la interacción entre los vehículos autónomos y el resto de los usuarios de la vía, centrándose en las distintas interfaces de comunicación que se pueden incorporar en un VA
- Capítulo 7 pone fin al documento presentando y discutiendo los logros alcanzados en el trabajo desarrollado en esta tesis, así como algunas de las líneas de investigación futuras que darán continuidad a este trabajo.

2 ESTADO DEL ARTE

Desde que comenzaron los primeros trabajos relacionados con vehículos autónomos, alrededor de los años 80, se han publicado numerosos trabajos de investigación sobre distintos aspectos de estos.

La amplia extensión de temas cubiertos en este documento requiere una revisión del estado del arte de distintos temas y líneas de investigación relacionadas con la conducción autónoma.

En este capítulo se revisarán los trabajos más relevantes relacionados con los principales temas de esta tesis, comenzando con una breve historia de los vehículos autónomos, de las posibles arquitecturas y de los sensores que se pueden utilizar. También se describirán los algoritmos utilizados para la navegación de un VA, así como las distintas interfaces de comunicación entre VA y humano.

2.1 HISTORIA DE PLATAFORMAS DE CONDUCCIÓN AUTÓNOMA

En esta sección se describen las competiciones, demostraciones y plataformas comerciales más relevantes relacionadas con la conducción autónoma de forma cronológica desde los años 80 hasta el presente. Esta descripción se realizará siguiendo un enfoque distinto al habitual, en el que se prestará especial atención a la arquitectura hardware de sensores utilizada en cada vehículo a lo largo de la historia. De esta manera, se podrá apreciar la tendencia en el uso de sensores según se ha ido desarrollando el VA, siendo este aspecto de gran utilidad a la hora de diseñar su arquitectura. Para ayudar a visualizar esto, en la Figura 2.4 aparecen las plataformas más relevantes junto con los sensores utilizados, colocadas en una línea temporal que permite distinguir distintas épocas en el desarrollo de la tecnología para la conducción autónoma.

2.1.1 *Primeros trabajos (1980-2000)*

Los primeros trabajos en vehículos autónomos empiezan a mediados de los años 80, y estaban basados principalmente en algoritmos de visión, los cuales representaban una gran carga computacional para los ordenadores embebidos de la época. Representando esta primera época destacan el vehículo VaMoRs [33] (Figura 2.1), considerado uno de los primeros vehículos autónomos desarrollado, o el vehículo VaMP [34], de la universidad Bundeswehr de Munich.

Este capítulo incluye contenido de [1], [2], [3], [4], [6], [7] y [8].

Figura 2.1: Interior y exterior del vehículo VaMoRs.

La universidad de Parma empezó su proyecto ARGO en 1996. Este vehículo completó más de 2000 km de conducción autónoma en carreteras públicas [35], utilizando un sistema basado en dos cámaras para seguir la carretera y evitar obstáculos.

En los años 90 nace el concepto *Cybercar* [36] como un vehículo urbano sin pedales ni volante. En 1997, se instala un prototipo en el aeropuerto de Schiphol para transportar pasajeros entre la terminal y el parking [37]. Este vehículo utilizaba tanto cámaras como *Light Detection and Ranging* (LiDAR) para conducir de forma autónoma por un carril dedicado con semáforos y pasos de cebra.

Además, en 1997 el *National Automated Highway Systems Consortium* (NAHSC), presentó una demostración de las funcionalidades de un VA [38] a modo de prueba de la viabilidad de esta tecnología. La demostración mostró el seguimiento autónomo de la carretera basado en una cámara, mantenimiento de distancia utilizando LiDAR y seguimiento de otro vehículo con un radar, entre otras funcionalidades, como maniobras en entornos con más vehículos.

2.1.2 *Pruebas de viabilidad (2000-2010)*

En el año 2004, DARPA lanzó una serie de desafíos para fomentar el desarrollo de tecnologías de conducción autónoma. Los logros durante las tres primeras ediciones supusieron un gran avance además de llamar la atención de grandes empresas. Los dos primeros desafíos (2004 y 2005), consistieron en cubrir una ruta de carreteras de tierra no asfaltadas en la que las tareas más importantes eran la navegación y el control. En la primera edición, ningún vehículo consiguió completar la ruta, mientras que en la segunda, el equipo ganador fué el de la univerdidad Stanford. Su vehículo, llamado Stanley (Figura 2.2), logró completar el recorrido y llegar en primera posición utilizando 5 LiDAR, una cámara frontal 2 radares, un Sistema Global de Navegación por Satélite o *Global Navigation Satellite System* (GNSS) y una Unidad de Medida Inercial o *Inertial Measurement Unit* (IMU) [39]. En la edición de 2007, orientada a entornos urbanos, los vehículos tenían que interaccionar con otros vehículos, peatones y obedecer regulaciones de tráfico complejas. El equipo de la universidad Carnegie

Mellon acabó en primera posición con su vehículo Boss [40, 41], que incorporaba un sistema de percepción con 5 radares y, 13 LIDAR.

Figura 2.2: Vehículo Stanley del equipo de la universidad Stanford. [9]

Estos eventos llamaron la atención de Google. La empresa contrató alrededor de 15 ingenieros de la competición DARPA, incluyendo a los ganadores de 2005 y 2007 [42, 43] para impulsar su propia investigación en este campo. La división de vehículos autónomos de Google (más tarde conocida como Waymo) apostó por la tecnología LIDAR, siendo este el principal sensor de percepción que montan sus vehículos [44].

Por otro lado, la universidad de Parma creó el laboratorio VisLab en 2009. Al contrario que Google, apostaban por la visión artificial como principal componente de los sistemas de percepción. En 2010 completaron el Desafío Autónomo Intercontinental VIAC (del inglés *VisLab Intercontinental Autonomous Challenge*), donde 4 caravanas autónomas condujeron desde Italia hasta China por carreteras públicas que incluían tramos de tierra o zonas no mapeadas [45]. El vehículo principal, situado en primera posición, realizaba las tareas de percepción (con cámaras y LIDAR), toma de decisiones y control con alguna intervención humana para elegir la ruta y gestionar situaciones críticas [46]. En 2013 realizaron una prueba llamada PROUD, donde pusieron a un vehículo sin conductor a circular por las carreteras de Parma en tráfico real [47].

2.1.3 *Desarrollo de productos comerciales (2010-2022)*

En la última década, las actividades de los vehículos autónomos han estado dominadas por iniciativas privadas que prevén desarrollar sistemas con un nivel de automatización de 4 o 5 en los próximos años.

Esta estrategia ha dado lugar a la aparición de distintas empresas dedicadas a este fin, la mayoría de ellas formadas por ingenieros participantes en DARPA [44]. Ejemplos de estas empresas incluyen nuTonomy (cofundad por el líder del equipo del MIT en la edición DARPA 2017, Cruise (fundada por otro miembro del mismo equipo), Otto (fundad por un participante de las dos primeras ediciones), Uber (quien contrató a 50 personas del laboratorio CMU), Zoox robotaxi (cofundada con un miembro del equipo de Stanford), y Aurora (que incluyó ingenieros de Uber, MIT y Waymo).

Los fabricantes de coches, sin embargo, han tardado más en reaccionar. Algunos de ellos empezaron líneas de investigación independientes, como el caso de BMW, que ha estado probando prototipos en carreteras desde 2011 [48], o el proyecto de Mercedes-Benz llamado Bertha [49] que circuló en una ruta de 103 km de forma autónoma utilizando sensores comerciales (8 radares y 3 cámaras), pero en la mayoría de los casos, los fabricantes han establecido alianzas con las empresas emergentes mencionadas.

La empresa Mobileye empezó a trabajar en una solución basada sólo en visión desde 2015. Después de probar sus algoritmos en condiciones reales [50], presentaron en 2018 una demostración de un vehículo de la marca Ford automatizado con 12 pequeñas cámaras que le permitían circular de forma autónoma [51].

Tesla entró en la conducción autónoma en 2014. Todos sus vehículos estaban equipados con una cámara (basada en el sistema Mobileye) y un radar de automoción que habilitaba una función de automatización de nivel 2 o 3. A partir de 2017, todos sus vehículos fueron equipados con un sistema de percepción más avanzado (la versión 2), compuesta por un radar frontal, 12 sónares y 8 cámaras. La compañía asegura que esta configuración es suficiente para alcanzar el nivel 5 de autonomía [52], que se podrá adquirir a través de una actualización de software.

La empresa Delphi Automotive completó en 2015 un viaje autónomo entre SanFrancisco y Nueva York utilizando un Audi Q5 equipado con 10 radares, 6 LIDAR y 3 cámaras. En 2017 adquirieron la empresa nuTonomy (la primera en desarrollar un taxi autónomo) y crearon Aptiv. Aptiv presentó otro servicio de taxi autónomo para la conferencia CES en 2018, como parte de una flota de 20 vehículos que habían estado en funcionamiento en Las Vegas durante varios meses. Estos taxis cuentan con 10 radares y 9 LIDAR integrados en el chasis además de una cámara.

Por otro lado, Waymo ha reunido una flota de vehículos Chrysler Pacifica que ha automatizado y habían conducido más de 32 millones de km de forma autónoma. Sus desarrollos han conseguido recortar el precio de los sensores LIDAR más de 10 veces en unos pocos años, habiendo llegado incluso a crear sus propios modelos de LIDAR [53]. Esta compañía, al igual que Cruise, consiguieron en 2021 la aprobación

por parte del gobierno de California para utilizar de forma comercial sus vehículos autónomos en algunas áreas de San Francisco [54].

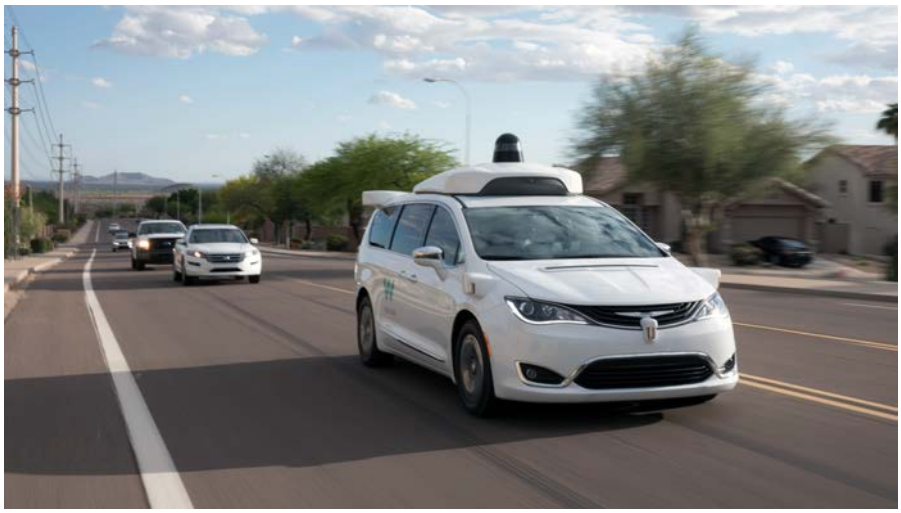


Figura 2.3: Vehículo autónomo de la empresa Waymo (Fuente: Waymo).

El último avance en automatización por parte de un fabricante lo presenta Mercedes-Benz en diciembre de 2021, consiguiendo los requerimientos legales para incorporar un sistema de automatización de nivel 3. Este sistema permitirá conducir de forma autónoma a velocidades de hasta $60 km/h$ en situaciones de tráfico denso o en tramos de autopista determinados [55].

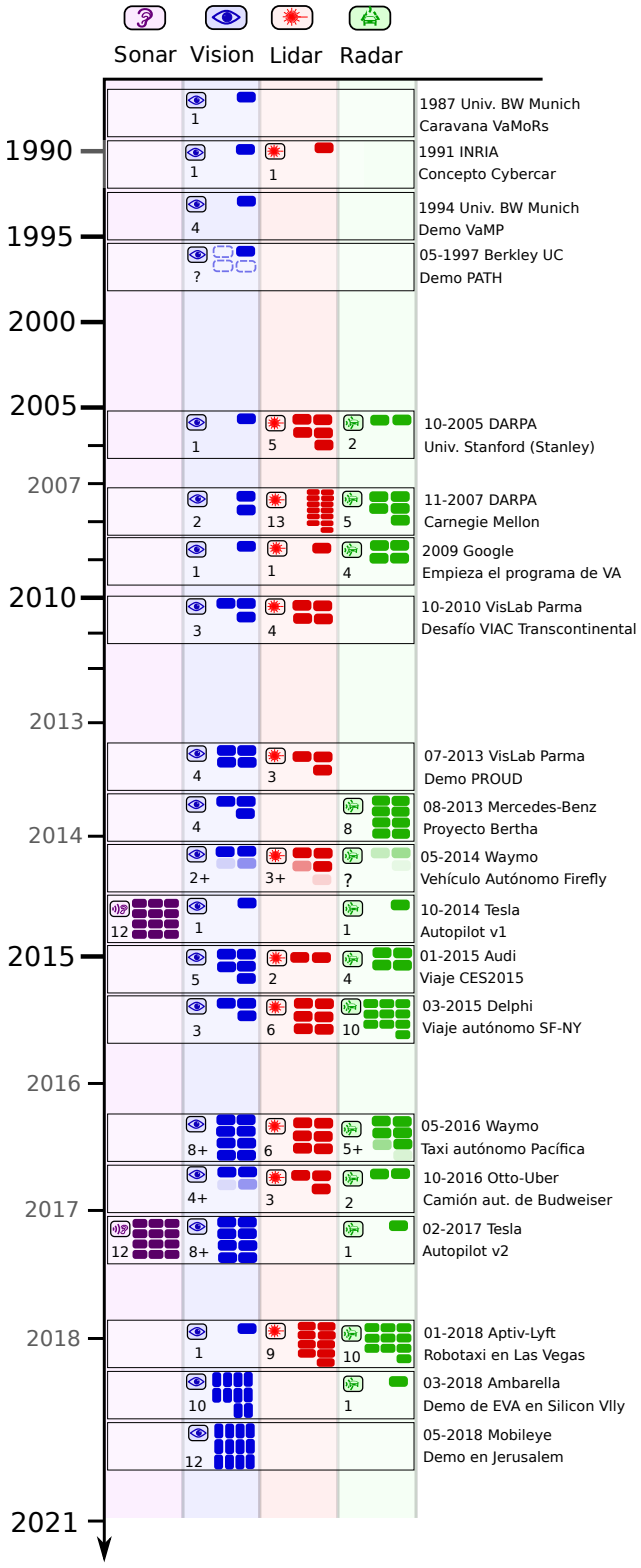


Figura 2.4: Cronología de las demostraciones más relevantes junto a la arquitectura sensorial del VA presentado.

2.2 ARQUITECTURAS PARA VEHÍCULOS AUTÓNOMOS

La arquitectura de un VA tiene una gran importancia en su proceso de diseño, ya que es la base sobre la que todos los algoritmos se ejecutarán y permitirá que el vehículo funcione de forma correcta. Su diseño requiere de un exhaustivo estudio y una definición concisa de las características que son necesarias para el VA, así como una previsión de las que serán necesarias en un futuro.

Comúnmente, la arquitectura de un VA se puede dividir en arquitectura software y arquitectura hardware en función de si los elementos que incluye son físicos (sensores, actuadores, etc.) o digitales (software, configuraciones de red, etc.), aunque ambas están estrechamente relacionadas, y hay que tener en cuenta una para diseñar la otra.

2.2.1 *Arquitectura hardware*

La arquitectura hardware se refiere a todos los dispositivos físicos que hacen posible que el VA funcione. Esta arquitectura engloba todos los componentes físicos, ya sean electrónicos o mecánicos, prestando especial atención a la relación e interacción entre ellos.

Atendiendo a la forma que pueden estar conectados los elementos hardware, y por lo tanto, la manera en la que se comunican, se pueden distinguir tres principales tipos de arquitecturas [56]:

- Centralizada: En este tipo de arquitectura, la información fluye directamente a una unidad de procesamiento principal que debe evaluar y procesar los datos recibidos para generar unas señales de control adecuadas. Los sensores y actuadores se conectan directamente a la unidad principal, de tal forma que las conexiones siempre se dirigen desde los sensores hacia el computador principal, y de este hacia los actuadores. Esta configuración en la más sencilla y rápida de implementar, y puede ser la más adecuada para proyectos sencillos que no requieran un gran procesamiento y flujo de datos. Sin embargo, al estar todo el procesamiento centrado en una única unidad, un fallo aquí puede suponer un compromiso en la seguridad del vehículo.

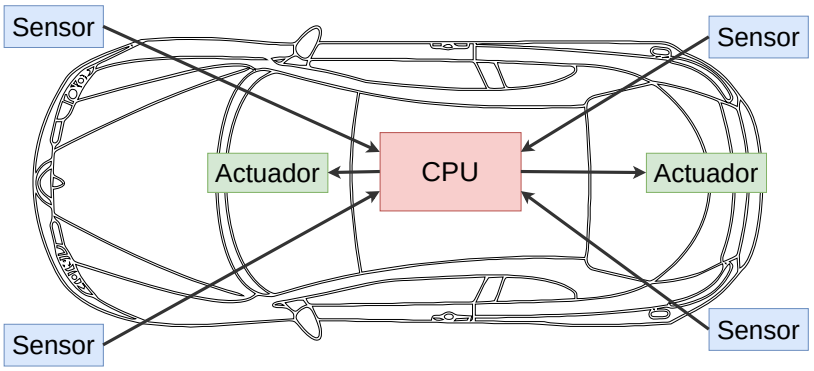


Figura 2.5: Ejemplo de arquitectura centralizada en un VA.

- Descentralizada: En esta arquitectura existen múltiples unidades de procesamiento encargadas, cada una de tareas distintas. Al contrario que en la centralizada, las computadoras sólo tienen acceso y procesan un conjunto de datos determinado, y solo tienen acceso a las lecturas de algunos de los sensores o computadores. Estas unidades centrales comparten información entre ellas, datos ya procesados, que se utilizarán para generar las señales de control.

  Aunque en este tipo de arquitecturas, un fallo puede tener graves consecuencias para el funcionamiento del vehículo, es más tolerante a fallos que la arquitectura centralizada. Su desarrollo, por el contrario, es más elaborado y requiere de una fase de diseño más tediosa, así como de un mayor coste de mantenimiento.

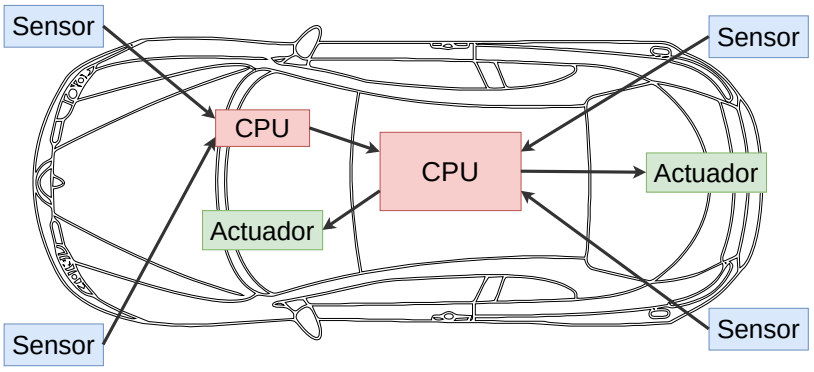


Figura 2.6: Ejemplo de arquitectura descentralizada en un VA.

- Distribuida: Esta última arquitectura, plantea un sistema con múltiples unidades de procesamiento en el que todos los elementos están interconectados, es decir, todas las computadoras tienen acceso a todos los sensores y los datos procesados del resto de computadoras. Es el tipo de arquitectura más tolerante a fallos, ya que cada unidad puede asumir las funciones de otra que presente un fallo. Por otra parte, es la más compleja y, además, tiene un límite de ancho de banda, ya que todos los datos fluyen por la misma red.

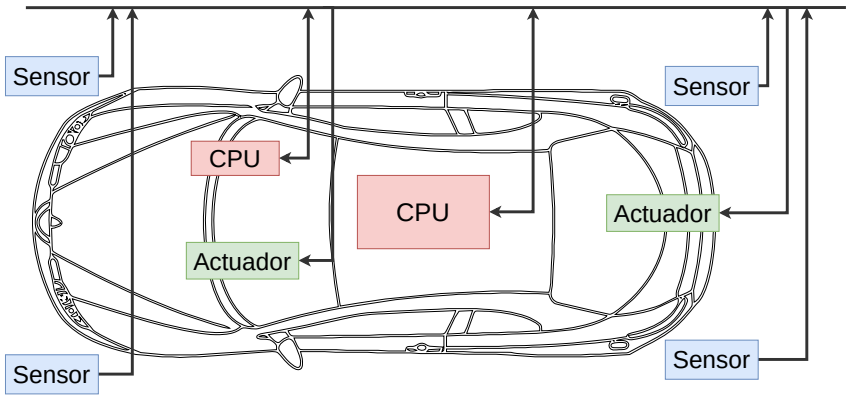


Figura 2.7: Ejemplo de arquitectura distribuida en un VA.

2.2.2 *Arquitectura software*

La arquitectura software de un VA no solo comprende todos los módulos de software y algoritmos incluidos en el vehículo que son necesarios para que este funcione de forma correcta, sino que además, analiza la estructura que organiza y relaciona todos estos módulos, teniendo en cuenta aspectos como funcionalidad, rendimiento, flexibilidad o reutilización entre otros.

En los siguientes apartados se describirán los distintos tipos de arquitecturas atendiendo a distintos niveles de abstracción, así como aspectos de seguridad y ejemplos de arquitecturas diseñadas para vehículos similares

2.2.2.1 *Tipos de arquitecturas de alto nivel*

A continuación, se describen los componentes software principales, así como las interacciones y el flujo de datos entre ellos. De forma similar a la arquitectura hardware ya descrita, se pueden definir distintos tipos de arquitecturas software, considerando los nodos de la red los elementos software (drivers de sensores, algoritmos de procesamiento de datos, etc.) en lugar de unidades de computación. Los sistemas de gran tamaño y complejidad pueden considerar como

nodos módulos que comprendan a su vez otra arquitectura software en su interior dada su complejidad. Existen principalmente dos tipos principales de arquitecturas [56]:

- Centralizada: Como se muestra en la Figura 2.8, la conexión entre sensores y actuadores se centraliza en un único nodo que procesa la información para generar las señales de control de los actuadores. Esta arquitectura es sencilla de desarrollar para proyectos pequeños y simples pues facilita la comprensión del flujo de la información ya que no se comparte entre procesos. Es necesario calcular y asegurar que el tiempo de ciclo de ejecución cumple los plazos establecidos. Este tipo de arquitectura puede resultar menos adecuada para algunos proyectos de vehículos autónomos debido a la gran cantidad de datos que es necesario procesar.

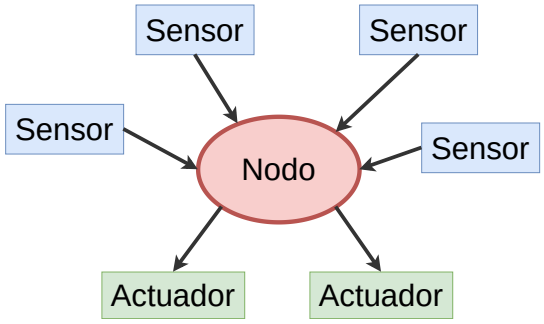


Figura 2.8: Ejemplo de arquitectura centralizada en un VA.

- Descentralizada: La Figura 2.9 muestra la configuración de nodos donde los sensores y actuadores se encuentran fuera del núcleo de la arquitectura dotando al sistema de una capa de abstracción de hardware. Esta capa tiene la ventaja de aislar los dispositivos físicos de la arquitectura, haciendo posible cambiar un sensor o un actuador sin afectar a los nodos de la arquitectura. Estos nodos están interconectados entre ellos, procesando e incorporando información de otros nodos hasta generar las señales de control. Este tipo de arquitecturas es útil en proyectos de investigación en constante evolución donde la configuración de sensores y actuadores no es definitiva.

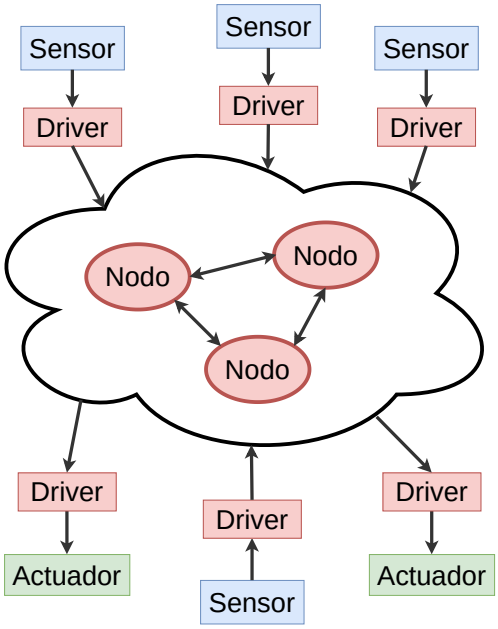


Figura 2.9: Ejemplo de arquitectura descentralizada en un VA.

No existe ningún diseño válido superior en el desarrollo de un VA, cualquier arquitectura diseñada puede ser válida siempre que se cumplan unos requisitos de tiempo de ciclo de computación y ancho de banda, así como otros factores distintos que se enumeran a continuación [56]:

- Mantenimiento: Este es un elemento crucial para proyectos con una larga duración. Con el tiempo, el sistema necesitará actualizaciones tanto de software como de hardware, así como realizar pruebas que aseguren que el sistema sigue funcionando correctamente. Los sistemas centralizados por lo general presentan un mantenimiento más rápido y sencillo.
- Detección de Puntos de Fallo: Dentro de la red de conexiones en la que distintos módulos comparten datos, es más sencillo localizar dónde se ha producido un fallo en la arquitectura centralizada, ya que solamente existe un módulo donde se concentra todo el procesamiento. Los sistemas distribuidos, de forma contraria, tienen múltiples puntos de fallo, por lo que detectar el punto de fallo puede ser menos intuitivo.
- Tolerancia a fallos o Estabilidad: Se refiere a la habilidad del sistema para continuar funcionando cuando se produce un fallo. Para sistemas centralizados, un error en el nodo central puede suponer una pérdida completa del sistema, haciendo que el vehículo pueda no ser seguro. Por el contrario, los sistemas distribuidos, si un nodo falla, otro puede ejecutarse en paralelo para solucionar el problema.

- Escalabilidad: Es un factor de gran importancia en el diseño de una arquitectura, ya que la plataforma puede requerir modificaciones en las que se añadan más elementos (como sensores o algoritmos que requieran computación) al sistema. Incorporar estos nuevos elementos a la arquitectura pude resultar más sencillo en sistemas distribuidos, ya que en los centralizados, puede ser necesario rediseñar la lógica de funcionamiento.

- Retrocompatibilidad: Es la característica que permite a la arquitectura funcionar con versiones anteriores de algunos de sus módulos o partes. En ocasiones, puede ser necesario probar algoritmos o sensores anteriores a los que el vehículo incorpora, bien para compararlos con los actuales o para recuperar alguna funcionalidad perdida. El diseño concreto de la arquitectura puede facilitar esta capacidad.

2.2.2.2 *Tipos de arquitecturas middleware*

El *Middleware* se puede definir como una capa entre el sistema operativo y las aplicaciones que gestiona las comunicaciones entre estas en sistemas distribuidos, es decir, funciona como un "Pegamento" de aplicaciones [57].

Existen distintos entornos de trabajo que proporcionan un *Middleware* orientado a la robótica, y por lo tanto, al desarrollo de vehículos autónomos. A continuación, se exponen los más relevantes:

- *Robot Operating System* (ROS) [58]: Sus siglas significan Sistema Operativo Robótico. Es un entorno de trabajo de Código Abierto para desarrollar software para robots. Desde su lanzamiento en 2007, ha pretendido simplificar la tarea de programar robots complejos y fomentar el uso de software colaborativo. Para ello, ROS proporciona una capa de comunicación entre nodos que permite enviar mensajes de forma asíncrona (*Topics*), llamadas a procesos de forma síncrona (Servicios) y un sistema de parámetros distribuidos que facilitan la configuración de los nodos. Además, permite grabar secuencias de datos que facilitan la realización de pruebas sin el vehículo real. ROS incluye numerosas librerías de software que incluyen desde drivers para un amplio catálogo de sensores, hasta complejo software de procesamiento de datos que permiten entre otras cosas, que un robot sea capaz de navegar de forma autónoma apenas dedicando trabajo. Además, numerosos simuladores y herramientas cuentan con soporte para ROS, como Gazebo [59] o CARLA [60]. Por último, ROS cuenta con una gran cantidad de herramientas que permiten depurar el código, visualizar datos y obtener información de utilidad para el desarrollo del sistema.

- ROS2 [61, 62]: Nace como una nueva versión de ROS, que mantiene muchas de las funcionalidades de la primera versión, pero trata de mejorar algunos aspectos críticos como la seguridad o la estructura. Entre los cambios más relevantes se encuentran: la utilización de DDS como middleware en lugar de uno propio (TCPROS), el cambio a un sistema distribuido en lugar de centralizado (desaparece el "Máster") y la compatibilidad para otros sistemas operativos como Windows.

- YARP [63]: Su sigla viene de "Otra Plataforma Robótica Más" (*Yet Another Robot Platform*). Se trata de un *Middleware* orientado a robots, en especial humanoides, que requieren un control de los motores además de realizar tareas de percepción en tiempo real. YARP sigue una filosofía que intenta facilitar la creación y mantenimiento de proyectos a largo plazo, simplificando la reutilización de código y proporcionando compatibilidad con otros tipos de *Middleware*.

- MRTP: Su sigla viene de "Herramientas para Programar Robots Móviles" (*Mobile Robot Programming Toolkit*). Proporciona una serie herramientas y algoritmos utilizados comúnmente en la investigación de robots móviles. Estas herramientas, entre las que se encuentran algoritmos de planificación, visión por computador y localización, cuentan con una gran fiabilidad ya que han sido probadas y evaluadas con numerosas pruebas.

2.3 BASES DE DATOS

Para probar y validar los algoritmos utilizados en un VA, es imprescindible ponerlos a prueba en escenarios reales, o lo más parecidos a la realidad. Las pruebas en entornos reales son las que mejor pueden validar este tipo de algoritmos, sin embargo, requieren de la inversión de una gran cantidad de tiempo y recursos, además de la necesidad de disponer de una plataforma de pruebas, que según el algoritmo que se pruebe, tendrá que ser parcial o totalmente funcional. Como alternativa, la simulación de un VA, si está correctamente configurada, puede arrojar resultados muy similares a los obtenidos con la plataforma real. Sin embargo, a la hora de comparar algoritmos, es necesario ponerlos a prueba bajo las mismas condiciones, lo que implicaría implementar y adaptar dichos algoritmos a la plataforma, ya sea simulada o real. Además, ciertos tipos de algoritmos pueden requerir de una gran cantidad de datos, como es el caso de los que utilizan técnicas de aprendizaje automático. Por estos motivos, la mayoría de los trabajos analizan sus resultados utilizando bases de datos públicas, con las que resulta más sencillo evaluar y comparar los resultados obtenidos, con los de trabajos similares.

Los algoritmos desarrollados en esta tesis requieren de dos tipos distintos de datos: de localización, y de trayectorias de vehículos, por lo que solamente se analizaran este tipo de bases de datos.

2.3.1 *Bases de datos de localización*

En el caso de localización, la base de datos debe incluir entre sus variables información de sensores de percepción de un vehículo que se puedan utilizar para obtener la localización del vehículo (cámaras, LIDAR, IMU, etc.), así como un *groundtruth* de la posición real, que permita evaluar los resultados del algoritmo. Existen numerosas bases de datos ([64–66]), sin embargo, la publicada por KITTI [65], es una de las más completas, incluyendo distintos tipos de entornos (ciudad, residencial, extrarradio), y una de las más utilizadas para evaluar la precisión de los algoritmos desarrollados, incluyendo una clasificación de los mejores métodos.

2.3.2 *Bases de datos de predicción de trayectorias*

Para la predicción de trayectorias, es necesario un tipo de base de datos completamente distinto, ya que no se necesita la información de los sensores de percepción, pero si es imprescindible que las secuencias incluyan información de la localización de todos los vehículos que circulan por la carretera, así como otra serie de datos que puedan resultar útiles para la predicción de la futura trayectoria (velocidades, aceleraciones, dimensiones y tipo de vehículo, etc.). Una de las bases de datos más utilizas en los estudios que se pueden encontrar en la literatura es la que base de datos del proyecto *Next Generation Simulation* (NGSIM) [67], la cual incluye varias horas de datos de conducción en dos autovías distintas de Estados Unidos. Sin embargo, tal y como muestra el estudio [68], esta base de datos presenta algunos problemas en los datos en crudo, que no pueden ser solventados mediante su filtrado. Estos problemas incluyen trayectorias erróneas que colisionan con otras, o magnitudes de algunas variables mucho mayores de lo esperado.

Por esta razón, en los últimos años han surgido nuevas bases de datos (como *Waymo Open Dataset* [69] o *Argoverse* [70]), algunas de las cuales están orientadas a nuevos escenarios distintos a las autovías (rotondas en *rounD* [71] o intersecciones en *inD* [72]). Una de las bases de datos más completas en el ámbito de las autopistas es la recientemente publicada *highway drone* (highD) [11], que no presenta los problemas mencionados con la base de datos NGSIM, proporcionando datos más precisos, y un mayor número de secuencias.

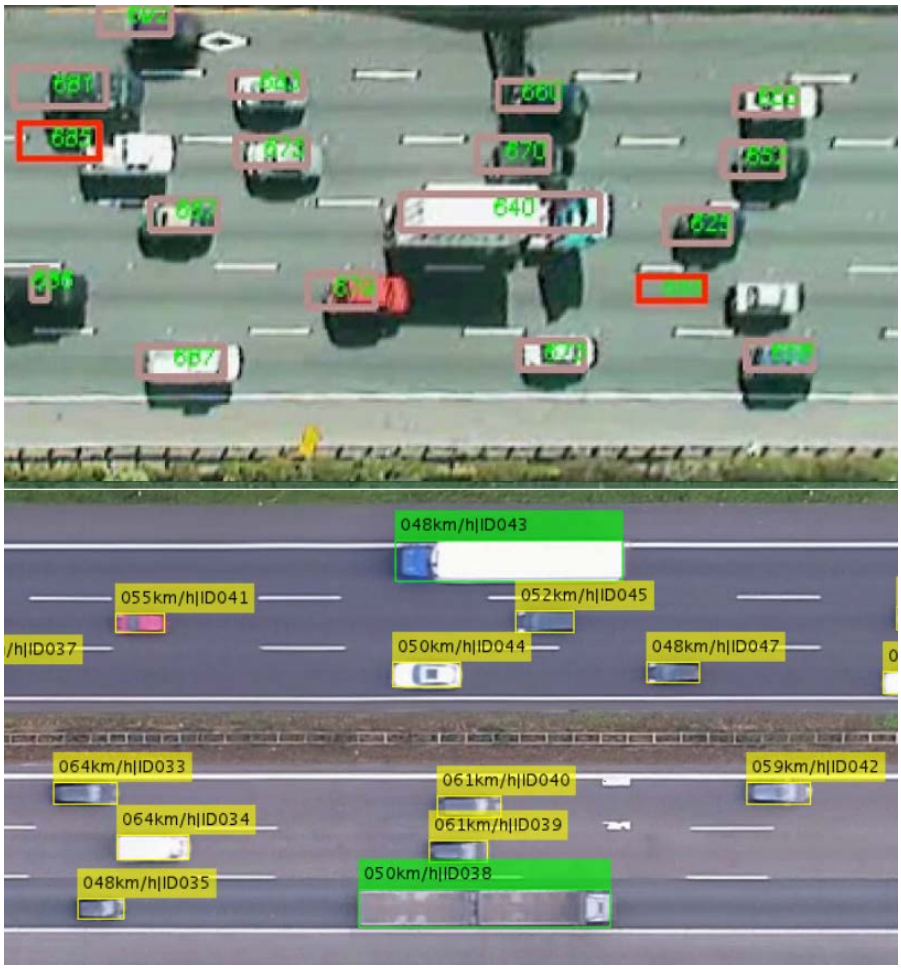


Figura 2.10: Comparación entre las detecciones de la base de datos NGSIM (arriba) y highd (abajo)[11].

2.4 SENSORES

Esta sección describe los sensores más utilizados en la conducción autónoma detallando sus principales aplicaciones y ventajas, pero también los inconvenientes o situaciones en las que no son adecuados, además de mencionar los últimos avances de cada tecnología.

El último punto de esta sección analiza la utilidad de cada sensor para obtener distintos tipos de información necesarios para un VA, así como su rendimiento en distintas condiciones atmosféricas.

2.4.1 *Cámara*

La visión artificial es una tecnología que se lleva utilizando durante décadas en algunas disciplinas como la robótica, vigilancia o inspección industrial. Las principales ventajas de esta tecnología residen en un bajo coste de los sensores (en comparación con otras tecnologías como LIDAR o radar), y la capacidad de obtener información del

entorno de distinto tipo como espacial (forma, tamaño, distancia), dinámica (movimiento analizando fotogramas consecutivos) o semántica (analizando la forma y textura).

Sin embargo, este sensor presenta algunos inconvenientes, en especial en aplicaciones de conducción autónoma. El principal de ellos está relacionado con las variaciones en la iluminación y visibilidad. La conducción puede llevarse a cabo de día, de noche, o con el sol en el horizonte. Además, pueden aparecer en la imagen zonas con sombra, reflejos u otros efectos que dificulten el desempeño de los algoritmos de visión. Una de las soluciones más populares consiste en extender el rango de espectro que capturan los sensores: El Infrarrojo (IR) lejano puede ser útil para detectar personas o animales en situaciones de mala iluminación [73, 74], y la inclusión del espectro IR cercano puede aumentar el contraste de las imágenes capturadas mejorando la visibilidad por la noche.

La visión por computador tradicional aporta información únicamente en dos dimensiones, sin embargo, existen otros tipos de sensores que pueden percibir información de profundidad. Estas tecnologías son:

Visión estéreo: La profundidad se calcula a partir del desplazamiento relativo entre imágenes capturadas por dos cámaras monoculares que tienen la misma orientación, pero están separadas una distancia conocida [75]. Una de las mayores ventajas de esta tecnología es la capacidad de proporcionar mapas de profundidad densos (Figura 2.11a), al contrario que con los sensores LiDAR o radar, que generan una menor cantidad de puntos. Las desventajas de este tipo de sensores son los problemas con colores planos y sin texturas que pueden dificultar la correspondencia entre imágenes

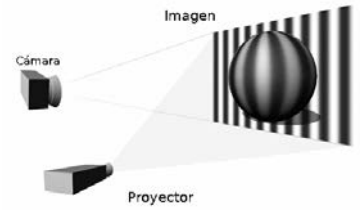
Luz estructurada: Consiste en una cámara monocular junto a un dispositivo que ilumina la escena con un patrón conocido de luz IR. Las superficies irregulares producen una deformación de dicho patrón a partir del cual se puede calcular el mapa de profundidad, tal y como se puede observar en la Figura 2.11b. Esta tecnología supera las limitaciones de la visión estéreo ya que no necesita superficies con texturas y tiene un coste computacional más bajo, sin embargo, requieren de una calibración precisa y tienen un rango de operación limitado por la fuente de luz (normalmente menor a 20 metros).

Tiempo de vuelo: Es un sensor activo basado en el mismo principio de operación del LiDAR (Sección 2.4.3). Un emisor de IR ilumina la escena con luz modulada que es capturada de vuelta con un sensor tras reflejarse en los elementos de la escena. El tiempo de vuelta puede ser calculado para cada pixel en base al desfase entre la luz capturada y la emitida, que se transforma en

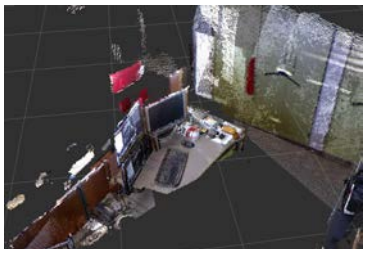
distancia. Como principal ventaja, este sensor es capaz de producir un mapa de profundidad denso a una alta frecuencia (más de 50Hz), sin embargo, suele tener un rango de operación corto (menor a 20 metros) y presenta problemas ante escenas con una luz ambiente muy intensa. En la Figura 2.11c se puede observar una imágen captada con este sensor en la que cada pixel, además de los valores de color (habitualmente RGB), tienen asociados una posición en el esepacio.

Visión basada en eventos: En este tipo de sensores, emergentes en los últimos años, los píxeles se activan de forma asíncrona e independiente cuando detectan un cambio de intensidad de luz (un evento). El sensor produce una serie de eventos que pueden ser agrupados en ventanas temporales para obtener una imagen. La independencia de los elementos del sensor aumenta el rango dinámico hasta 120dB, permitiendo aplicaciones de alta velocidad en condiciones de baja visibilidad. Los eventos pueden ser la entrada de algoritmos de odometría visual o Localización y mapeado simultáneo o *Simultaneous Localization And Mapping* (SLAM) reduciendo la carga computacional de operaciones con imágenes en crudo. La Figura 2.11d muestra los eventos sucedidos en una ventana de tiempo, donde el color representa el cambio en iluminación de cada píxel.

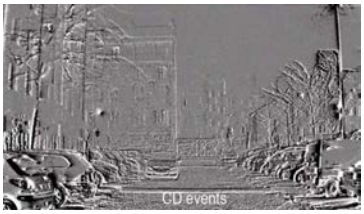
(a) Mapa de disparidad de una cámara estéreo.



(b) Principio de funcionamiento de la cámara de luz estructurada [13].



(c) Ejemplo de imagen de una cámara de tiempo de vuelo.



(d) Imagen de eventos de una cámara de eventos [12].

Figura 2.11: Ejemplos de distintos tipos de cámaras con información tridimensional o temporal.

2.4.2 *Radar*

La tecnología radar utiliza ondas electromagnéticas de alta frecuencia para medir distancias a los objetos. El proceso de medición se basa en el tiempo que tarda la onda en rebotar en el objeto y volver al sensor.

La mayoría de los radares modernos utilizados en automoción conocidos como radar de Onda Continua Modulada en Frecuencia o *Frequency-Modulated Continuous Wave* (FMCW), emiten una señal con una frecuencia conocida que es modulada con otra señal continua que aumenta y disminuye su frecuencia.

La distancia se determina mediante la comparación entre las frecuencias de la señal emitida y la observada. Los radares también aprovechan el efecto Doppler para obtener una observación de la velocidad relativa del objeto con respecto al sensor. La figura 2.12 muestra un ejemplo de detección de dos vehículos en movimiento mediante un sensor radar de alta resolución.

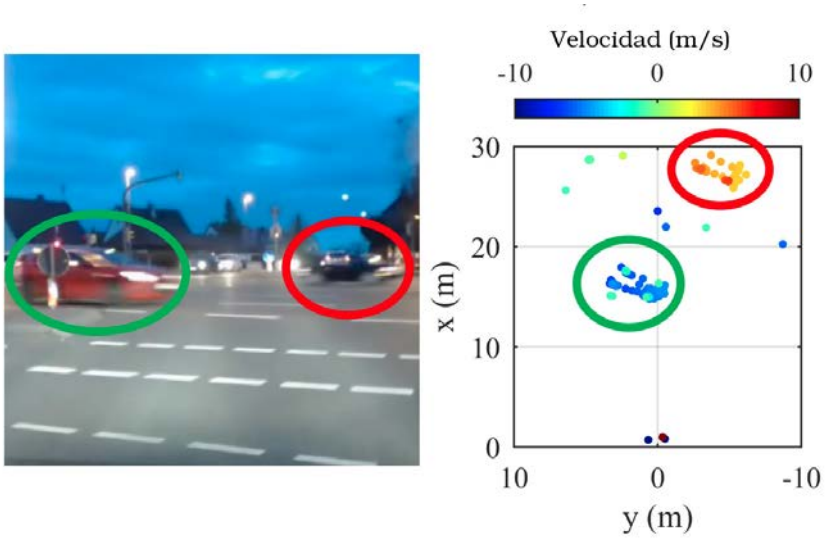


Figura 2.12: Imagen de la escena (izquierda) y vista de pájaro de la información del radar (derecha). Los vehículos detectados aparecen rodeados en ambas imágenes [10].

Una de las principales razones para incluir un sensor radar en vehículos autónomos es su buen rendimiento ante condiciones de luz y atmosféricas adversas. Este sensor funciona en la oscuridad, y las detecciones son similares con nieve, lluvia, niebla o polvo [76]. Además, los radares de largo alcance pueden detectar objetos hasta una distancia de 250m incluso en condiciones adversas, al contrario que el resto de los sensores.

Sin embargo, esta tecnología presenta algunos inconvenientes. El principal de ellos es la alta sensibilidad a la reflectividad del objeto. Procesar la señal de un radar es complicado debido a la diferente

reflectividad que tiene cada material. Los metales pueden amplificar la señal, produciendo un objeto aparentemente mayor de lo que en realidad es (una lata metálica en la carretera puede parecer un objeto mucho más grande), mientras que otros materiales como la madera pueden ser indetectables para el radar. Esto hace que las detecciones de este sensor tengan tanto falsos positivos (detecta objetos que en realidad no existen) como falsos negativos (no detecta objetos potencialmente peligrosos).

En cuanto a la resolución y precisión, los radares son muy precisos midiendo distancia y velocidad, sin embargo, la resolución horizontal depende de las características del haz emitido. Comúnmente, esta resolución varía entre 2 y 5 grados, lo que hace difícil separar dos objetos próximos entre ellos que se encuentren cerca del vehículo (un peatón y un vehículo próximos a menos de 30m pueden parecer un mismo obstáculo, o dos vehículos a 100m en carriles distintos pueden ser detectados como uno sólo).

En los últimos años, la investigación de estos sensores orientados a automóviles se ha focalizado principalmente en el incremento de la resolución [77, 78]. Además de ser capaces de rastrear objetos separados, una alta resolución puede producir información más rica que permita clasificar o incluso mapear el entorno [76].

2.4.3 *LiDAR*

Un LiDAR es una tecnología que calcula la distancia a objetos midiendo el tiempo de vuelo de un pulso láser.

Los sensores de este tipo utilizados en robótica y automoción utilizan un láser de baja potencia en el rango del IR cercano (900-1050 nm) que es invisible y seguro para el ojo. Los haces láser tienen una baja dispersión para reducir la pérdida de potencia con la distancia, permitiendo medir distancias de hasta 200 m bajo la luz del sol.

Normalmente, en este tipo de sensores existe un espejo rotatorio que se utiliza para cambiar la dirección del haz, alcanzando una cobertura horizontal de 360 grados. Algunas soluciones comerciales utilizan un conjunto de emisores que producen varios planos verticales (comúnmente entre 4 y 128). Todo esto, genera una nube de puntos 3D que representa el entorno y con la que es posible detectar objetos junto con su posición y forma e incluso clasificarlos.

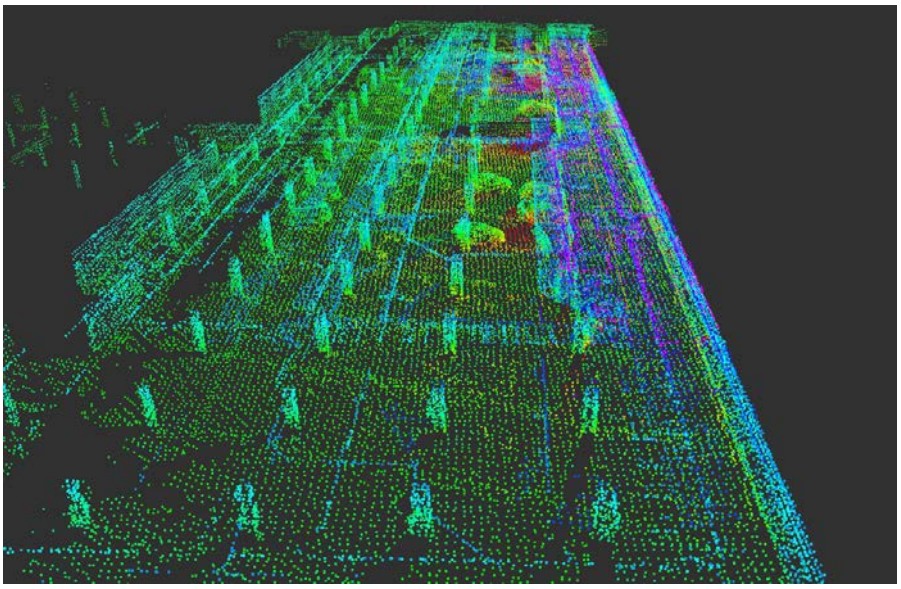


Figura 2.13: Mapa del entorno generado con un sensor LiDAR.

Sin embargo, presentan distintos inconvenientes que se deben considerar. El primero es una resolución vertical baja. Los modelos de bajo coste habitualmente tienen menos de 16 planos lo que produce una resolución vertical (separación entre dos planos consecutivos) de hasta 2 grados. A 100m de distancia, esto se traduce en una distancia vertical de 1.7m. Los modelos con más planos pueden reducir esta distancia hasta 0.2m, pero con un coste muy elevado.

Otro factor a tener en cuenta es la dificultad para detectar objetos muy oscuros. Algunos vehículos u objetos muy oscuros y con una superficie no reflectante, pueden parecer invisibles para este sensor, ya que los haces láser pueden ser absorbidos por la superficie y no regresar al sensor. Además, la luz del IR cercano puedo ser afectada por la niebla o la lluvia ya que las gotas de agua dispersan la luz, reduciendo el rango de operación o produciendo falsos positivos. El sensor LiDAR en entornos adversos obtiene peores resultados que un radar, pero mejores que una cámara [79].

Los últimos avances en esta tecnología incluyen los FMCW (del inglés *Frequency Modulated Continuous Wave*) LiDAR [80], que emite luz de forma continua para medir la velocidad del objeto mediante el efecto Doppler de forma similar al radar. Por otra parte, también están ganando relevancia los avances en LiDAR de estado sólido. Esta denominación se puede referir tanto a microespejos que oscilan y desvían los haces láser en distintas direcciones para cubrir un área determinada, como a la tecnología *Optical Phased Array* (OPA) [81], que permite controlar la dirección del haz sin utilizar partes móviles.

2.4.4 *Uso de sensores para percepción*

La tarea de un sistema de percepción consiste en enlazar los datos en crudo de los sensores, con la información que necesitan los algoritmos de toma de decisión. Los términos datos e información, aunque son similares, hacen referencia a dos conceptos distintos. El primero se refiere a los hechos en crudo y no organizados que necesitan ser procesados, mientras que el segundo incluye los datos ya procesados, organizados, estructurados y presentados en un contexto apropiado.

La Tabla 2.1 presenta una clasificación de los distintos tipos de información relacionados con la percepción en vehículos autónomos, que permitirá más adelante discutir la idoneidad de los sensores para cada tarea de percepción. Los elementos marcados con un asterisco representan información derivada que se puede inducir a partir de los datos de los sensores, pero no puede ser directamente observada; está relacionada con el estado interno de agentes externos, como las intenciones de las personas o animales del entorno.

Categoría	**#**	**Tipo de información**
Propio vehículo	1	Cinemática/dinámica (incluye la posición)
	2	Propioceptiva (estado de los componentes)
Ocupantes	3	Atención/capacidades del conductor
	4	* Intenciones del conductor (modelo mental)
	5	Estado de pasajeros (necesidades, factores de riesgo)
Entorno	6	Información del entorno: situación, tamaño, forma, características propias
	7	Identificación: clase, tipo, identidad
	8	Características semánticas: señales, marcas viales
	9	Factores contextuales: clima, situación especial (p. ej. retenciones, tramos sin asfaltar, emergencias)
Actores externos	10	Características espaciales: localización, tamaño, forma, características propias
	11	Cinemática/dinámica: posición, movimiento, intención
	12	Identificación: clase, tipo, identidad
	13	Características semánticas: luces del vehículo, vestimenta del peatón (policía, obras, etc.), gestos
	14	* Tipo de comportamiento: colaborativo/consciente (adultos, otros vehículos) o no colaborador/inconsciente (animales, niños)

Tabla 2.1: Tipos de información en el dominio de un VA.

La selección y disposición de los sensores es uno de los aspectos más importantes en el diseño de la percepción de un VA. Es necesario

elegir una configuración suficiente para las características del vehículo que se quiera desarrollar sin excederse en las especificaciones, ya que el coste de los sensores puede en algunos casos ser varias veces mayor que el del propio vehículo. A continuación, se presentan los dos aspectos con mayor importancia de los sensores: el tipo de información que obtienen, y el impacto que tienen sobre ellos distintos factores ambientales. Para un análisis de la cobertura y rango de los sensores se puede consultar [82].

Las características de cada tecnología de percepción determinan su idoneidad para adquirir ciertos tipos de información, y restringe sus condiciones de operación. La Tabla 2.2 relaciona las principales tecnologías utilizadas actualmente en la conducción autónoma con los tipos de información identificados en la Tabla 2.1.

Tipo de información		Visión (Mono)	Visión (estéreo)	Radar	LiDAR 2D	LiDAR 3D
Configuración espacial (6,10)	Localización	M	R	B	B	B
	Tamaño	B	B	R	B	B
	Forma	B	B	M	R	B
	Características propias	B	B		M	R
Cinemática (11)	Velocidad, aceleraciones	M	R	B	B	B
Identificación (7,12)		B	B	R	M	B
Regulación o Semántica (8,13)	Señales de tráfico	B	B			M
	Marcas de la carretera	B	B			R
	Gestos (Personas)	B	B			R
	Vestimenta (Personas)	B	B			
	Luces de vehículos	B	B			
Contexto (9)	Clima	B	B			
	Situación de conducción	R	R			R

Tabla 2.2: Idoneidad de los sensores para distintas tareas de percepción.

El desempeño de un sensor adquiriendo un cierto tipo de información (o dicho de otro modo, la calidad esperada de la información capturada por una tecnología) se clasifica en tres niveles: Buena (verde), Media (amarillo) y Mala (rojo).

Por otra parte, los sensores y la percepción están pensados para trabajar ininterrumpidamente durante la actuación del vehículo. El tiempo atmosférico y otros factores pueden empeorar el funcionamiento de los sensores, pero cada tecnología es afectada de forma

distinta. La Tabla 2.3 resume el efecto de los factores externos más comunes en rendimiento de las tecnologías analizadas utilizando la misma notación que en la Tabla 2.2.

Tecnología	Baja Luminosidad	Luz del sol directa	Lluvia	Niebla/Polvo
Visión (Mono, luz visible)	M	B	R	M
Visión (Estéreo, luz visible)	M	B	R	M
Visión (IR cercano)	R	B	R	M
Visión (IR lejano)	B	B	R	M
Visión (Tiempo de vuelo)	B	B	R	M
LIDAR 2D	B	B	R	M
LIDAR 3D	B	B	R	M
Radar	B	B	B	B

Tabla 2.3: Robustez de los sensores ante distintos factores ambientales.

2.5 NAVEGACIÓN

La navegación engloba las distintas tareas que son necesarias para que un vehículo se desplace entre dos localizaciones.

Habitualmente, la navegación consta de un módulo de localización que proporciona información sobre la posición del vehículo en el espacio, un módulo de planificación que genera una trayectoria en base a la localización, la percepción y la predicción de trayectorias del resto de agentes cercanos, y un módulo de seguimiento de trayectorias que genera los comandos de control del vehículo.

A continuación, se describen todos estos módulos de navegación.

2.5.1 *Localización*

La localización de vehículos autónomos es el problema de estimar su posición, comúnmente determinada en vehículos terrestres por las coordenadas X e Y en un mapa y su orientación. Esta localización debe ser tan precisa como sea posible ya que numerosos módulos del vehículo, tales como el control o la planificación de trayectorias, dependen estrechamente de su precisión. Errores en la localización pueden causar que el vehículo tenga un comportamiento indeseado, o que sea incapaz de seguir la trayectoria.

Existen múltiples algoritmos de localización, de los cuales sólo se describirán los más relevantes para la localización en vehículos autónomos. Dichos algoritmos pueden dividirse en localización absoluta, o referida a un sistema de referencia global, tales como las localizaciones basadas en un receptor GNSS , o basadas en la comparación de un mapa con un sensor LiDAR, y odometrías relativas, que ofrecen una estimación de los incrementos de desplazamiento con respecto a la posición inmediatamente anterior, incluyendo siempre un pequeño error que se va sumando y va aumentando con el tiempo. En este tipo de localización se engloban la odometría basada en una cámara, (odometría visual), odometría basada en un LiDAR y odometría basada en los *encoders* de las ruedas del vehículo.

2.5.1.1 *Localización basada en un receptor GNSS*

El GNSS es comúnmente utilizado para obtener la localización de un vehículo. Este sistema proporciona una localización global sin deriva, que puede llegar a ser muy precisa. Los receptores que utilizan un GNSS de alta precisión que admiten correcciones diferenciales o *Real Time Kinematic* (RTK), obtienen un aumento considerable de la precisión, aunque comúnmente tienen un alto coste y no son capaces de solucionar todas las fuentes de error asociadas a esta tecnología.

Estos errores pueden estar generados por los propios satélites (p. ej. inexactitud en el reloj o dilución de la precisión), problemas en la recepción de las señales del satélite (p. ej. oclusión de satélites, interferencias en la señal) o errores de propagación de la señal. Este último tipo de error incluye inexactitudes producidas por distintas condiciones climatológicas en las capas terrestres ionosfera o troposfera, y por la propagación multicamino, causada por la reflexión de las ondas del satélite en objetos de gran tamaño (como edificios o árboles) que rodean al sensor [83, 84].

Algunos de estos problemas, como la propagación multicamino, son un tema recurrente de investigación para tratar de mitigarlo. Un ejemplo de ello es [85], donde se genera un mapa del entorno utilizando *Open Street Maps* (OSM) para prevenir este error o [86], donde el multicamino es estimado y mitigado utilizando un filtro de partículas. Sin embargo, estos trabajos no proporcionan suficiente precisión y dependen de la información de OSM, que no está disponible en todas partes, o puede no ser suficientemente precisa.

2.5.1.2 *Localización basada en un mapa y un* LiDAR

Los métodos probabilísticos de localización basados en un sensor LiDAR, pueden comparar un mapa generado previamente con las lecturas láser del sensor para generar una posición y orientación precisa del vehículo, como en la localización Monte Carlo o *Monte Carlo Localization* (MCL) [87].

MCL funciona como un filtro de partículas que utiliza la correlación entre las lecturas del LIDAR y el mapa generado como una característica para cada partícula que determina la probabilidad de supervivencia en la siguiente iteración. Como mejora del MCL, la localización *Adaptive Monte Carlo Localization* (AMCL), supera al clásico MCL [88], ya que utiliza un muestreo con la Distancia Kullback-Leibler (KLD) para conseguir que el filtro converja de forma más rápida. Este tipo de métodos de localización pueden ser utilizados en vehículos autónomos como en [89], que dispone de un LIDAR 2D y un mapa con las características de la carretera para localizarse.

El problema presentado con este tipo de algoritmos es el contrario que el comentado en el apartado anterior con el receptor GNSS : en espacios abiertos con menos objetos en el mapa la precisión disminuye considerablemente. En este caso, las partículas del filtro se dispersan generando distintos núcleos de partículas lejos de la localización real. Este suceso está relacionado con el *kidnapped robot problem* o problema del secuestro del robot, donde un robot es colocado en otro punto distinto a su localización real (es secuestrado), y debe reconocer esta nueva localización [90]. Este problema es relativamente común en los métodos de localización probabilística como MCL [91].

2.5.1.3 *Odometría basada en encoders*

Este tipo de odometría es comúnmente utilizada en todo tipo de vehículos o robots que se desplazan apoyándose en una superficie. El principio de funcionamiento consiste en calcular el desplazamiento del vehículo, sabiendo que siempre hay al menos una parte de este en contacto con el suelo, y utilizando ecuaciones de cinemática. Para el caso de un vehículo como el desarrollado en esta tesis, con sólo dos grados de libertad (movimiento de las ruedas y del volante), el cálculo del desplazamiento del vehículo se puede simplificar en gran medida si se desprecian las componentes dinámicas del movimiento y sólo se consideran las cinemáticas. En este caso, con la información de un *encoder* situado en las ruedas que aporte información sobre el desplazamiento del vehículo y otro colocado en el volante para conocer el giro de las ruedas delanteras, se puede estimar el desplazamiento del vehículo en cada instante de tiempo.

2.5.1.4 *Odometría visual*

La odometría visual estima la posición y orientación de un vehículo analizando las variaciones producidas por el movimiento una o múltiples cámaras en una secuencia de imágenes [92].

La principal ventaja de la odometría visual con respecto a la odometría de las ruedas consiste en que la primera no está afectada por el posible deslizamiento de las ruedas en terrenos irregulares o que ofrezcan poca fricción. Comparando ambas fuentes de odometría, la visual

produce trayectorias más precisas con un menor error en la posición relativa. Esto hace de la odometría visual un suplemento a considerar para mejorar otros sistemas de navegación como los mencionados en este apartado [93].

Para un correcto funcionamiento de la odometría visual, se deben cumplir una serie de características: El entorno debe estar suficientemente iluminado y la escena debe ser lo más estática posible y con suficiente textura para obtener puntos característicos con los que poder calcular el movimiento. Además, los fotogramas consecutivos que se capturan deben tener en común suficiente área, por lo que el algoritmo queda limitado por la velocidad de movimiento o por la frecuencia máxima que la computadora puede procesar fotogramas consecutivos. Asimismo, las oclusiones de los vehículos de alrededor, que además no están estáticos, pueden reducir la precisión de este tipo de odometría [93].

Los principales problemas de este tipo de localización son el alto coste computacional comparado con otros algoritmos con resultados similares y la acumulación de deriva, ya que el error del desplazamiento calculado entre fotogramas se va acumulando, aunque este último problema se puede solucionar aplicando técnicas de SLAM .

2.5.1.5 *Odometría basada en un* LiDAR

Este tipo de odometría utiliza las nubes de puntos generadas por el LiDAR para estimar el desplazamiento del sensor, y por lo tanto el vehículo. Existen distintos algoritmos que utilizan este principio para calcular la odometría, sin embargo, uno de los más utilizados y con mejores resultados en la clasificación KITTI es *Lidar Odometry and Mapping* (LOAM) [94].

El funcionamiento en líneas generales de dicho algoritmo consiste en extraer puntos característicos de la nube de puntos, seguido de establecer una correlación entre los puntos de dos lecturas consecutivas y estimar el movimiento del sensor. Además de calcular la odometría, LOAM también genera un mapa 3D del entorno, que utiliza para realizar una nueva correlación de toda la reconstrucción con la nube de puntos actual, consiguiendo así eliminar la deriva acumulada que se puede generar teniendo en cuenta sólo el desplazamiento producido entre dos lecturas de LiDAR consecutivas.

A pesar de que esta última característica le otorga la capacidad de generar una localización absoluta (sin deriva acumulada), en la práctica la localización genera saltos erróneos debido a una mala correlación de la lectura actual con la nube acumulada que lo hacen inviable como única localización (Aunque si en combinación con otros algoritmos).

Además, la calidad la odometría de este método depende en gran medida del entorno y la cantidad y tamaño de los objetos que haya alrededor, ya que son necesarios para una mejor detección de puntos

característicos. Por otro lado, las deformaciones que pueden surgir en las nubes de puntos a altas velocidades, así como la frecuencia máxima de ejecución de este algoritmo, le hace adecuado en vehículos que operen a velocidades reducidas.

2.5.1.6 *Fusión de distintos tipos de localización*

Una práctica común a la hora de mejorar la localización consiste en combinar dos o más algoritmos de distintos modos.

El método de fusión más comúnmente utilizado es el Filtro de Kalman [95], un algoritmo que es capaz de estimar un estado no observable (la localización del vehículo), a partir de una serie de medidas observadas a lo largo del tiempo (p. ej. las posiciones que genera el GNSS), las cuales tienen cierto error y una predicción del siguiente estado (p. ej. del sensor inercial o de la odometría de las ruedas).

El filtro de Kalman por lo tanto, se utiliza para fusionar una fuente de localización absoluta (la medida) con otra incremental (la predicción).

Otro tipo de fusión consiste en combinar dos fuentes de localización absoluta como la localización utilizando un receptor GNSS (Sección 2.5.1.1) o la basada en algoritmos similares a MCL (Sección 2.5.1.2), para lo cual es necesario aplicar otro tipo de métodos.

Para obtener lo mejor de cada algoritmo, existen distintos trabajos que exploran la idea de combinar ambas fuentes de información para mejorar la localización. La mayoría de los trabajos que se pueden encontrar en la literatura, mejoran el algoritmo MCL reiniciando la posición calculada cuando esta difiere demasiado con respecto a la obtenida por GNSS o, en otras palabras, el problema del secuestro del robot se soluciona una vez detectado. Las principales diferencias entre estos trabajos residen en el método de reinicio ([96] utiliza el método *Expansion resetting* mientras que en [97] utiliza la información del LIDAR) o en el sensor de localización (en [97] se utiliza el sensor GNSS , mientras que [98] se basa en la señal Wi-Fi).

Este tipo de métodos son eficaces en resolver el problema del secuestro del robot, sin embargo, durante el tiempo necesario hasta que se detecta y se resuelve, la localización tiene un gran error que puede ser inaceptable para algunas aplicaciones como la conducción autónoma de un vehículo, ya que puede llevar a una incorrecta señal de control que provoque que el vehículo se salga de su carril. Por esta razón, parece razonable intentar prevenir el problema del secuestro del robot en lugar de solucionarlo cuando sucede.

En el trabajo [99], se utiliza la información de localización GNSS para generar nuevas partículas en el filtro, eliminando las más distantes. Sin embargo, este método no considera la guiñada (Yaw), esencial para los algoritmos de control. Por otra parte, introducir partículas en base a las lecturas del receptor GNSS en todos los ciclos

del filtro, puede hacer que la correlación entre el mapa y la nube de puntos pierda importancia, aumentando el ruido de la localización. Además, no se presentan resultados cuantitativos en dicho trabajo. En [100], se muestra cómo la fusión entre ambos sensores basada en la modificación de los pesos de las partículas tiene mejores resultados que añadir directamente partículas. Sin embargo, en este trabajo no se presentan estrategias de recuperación cuando la posición del filtro y la obtenida con el sensor de localización difieren considerablemente.

2.5.2 *Seguimiento de trayectoria*

El seguimiento de trayectoria o control de un vehículo consiste en generar una serie de comandos para los actuadores que permitan al vehículo, a partir de los datos de localización obtenidos, seguir una trayectoria predefinida lo más cerca posible. Habitualmente se distinguen dos tipos de control en un vehículo: el control longitudinal, que genera las acciones de control del acelerador y freno para que el vehículo se mantenga a una velocidad objetivo, y el control lateral que genera las señales de control del volante para que el vehículo se mantenga lo más próximo posible a la trayectoria.

En la mayoría de los controladores clásicos, más sencillos, sólo se estudia el control lateral, dejando para el longitudinal un controlador sencillo tipo PID. Sin embargo, existen algunos controladores que generan ambas señales a la vez, ya que la aceleración longitudinal del vehículo puede afectar también a la posición lateral.

Existen diferentes estrategias y algoritmos de control, a continuación, se describen los más relevantes.

2.5.2.1 *Pure pursuit*

Se trata de una de las primeras estrategias de control desarrolladas. Apareció por primera vez en [101], y fue desarrollado en [102, 103].

Esta estrategia, junto con sus variaciones, han sido una herramienta indispensable de control, gracias a su fácil implementación y sus buenos resultados. Existen numerosas publicaciones que citan este controlador, incluyendo tres vehículos que participaron en el desafío DARPA.

La ley de control está basada en ajustar un arco de circunferencia de radio R, de tal forma que empiece en la posición y orientación actual del vehículo, y pase por un punto de la trayectoria de referencia que se encuentra siempre a una distancia L del vehículo, llamada *lookahead distance*, tal y como se muestra en la Figura 2.14.

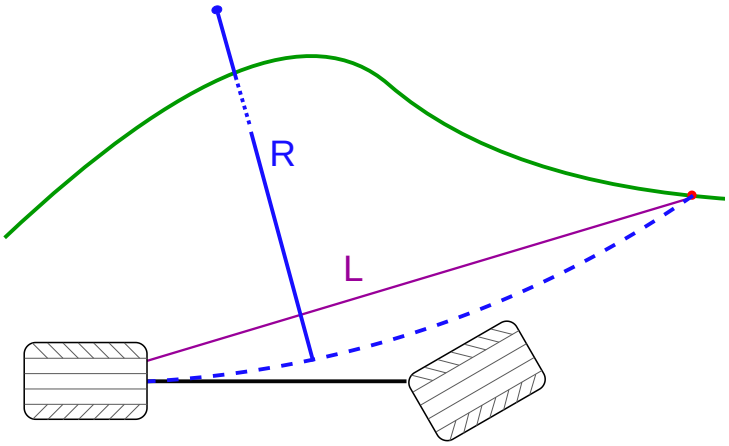


Figura 2.14: Esquema del controlador Pure Pursuit. La trayectoria a seguir en verde, el arco de circunferencia en azul.

Este algoritmo es capaz de controlar el vehículo si se asume que la trayectoria de referencia no tiene curvatura y la velocidad es constante. Sin embargo, para trayectorias con una curvatura constante, pure pursuit tiene un pequeño error constante, y para trayectorias en las que la curvatura cambia, el vehículo puede desviarse de la trayectoria. Además, el controlador no está definido para un estado en el que el vehículo se encuentre a una distancia de la trayectoria mayor que L.

2.5.2.2 *Stanley*

El origen del algoritmo de seguimiento de trayectorias Stanley está en la competición DARPA, donde el vehículo Stanley consiguió el primer puesto utilizando este método de control [104].

Este algoritmo realiza el cálculo de la señal de control del volante con una ecuación no lineal donde se considera la orientación de las ruedas delanteras en lugar de la orientación del vehículo como es habitual en otros controladores.

Este controlador demuestra estabilidad y consigue que el error de la trayectoria del vehículo se reduzca de forma exponencial siempre que la trayectoria sea continua y diferenciable.

Sin embargo, las principales limitaciones que presenta están en la velocidad máxima que puede adquirir el vehículo, ya que al no considerar dinámicas del vehículo (momento de inercia y masa), debe ser reducida. Además, este controlador no es estable para circular marcha atrás, por lo que no se puede utilizar para realizar maniobras como el aparcamiento.

Existen distintas versiones de este controlador, que añaden distintos términos a la función de control para mejorar el rendimiento de la forma original. La expresión más completa para el ángulo de giro del volante δ, según [9] viene dada por:

$$\delta = \Psi_e + arctan(\frac{k \cdot e}{k_{soft} + v}) + k_{curv} \cdot r_e + k_{steer} \cdot \delta_e \tag{2.1}$$

donde se incluye el error de ángulo de giro Ψ_e y error lateral e, además de la curvatura r_e, los posibles retrasos producidos por la inercia del volante o la comunicación y la velocidad actual del vehículo v. La influencia de estos términos en el resultado final, están ajustados por distintas constantes ($k, k_{soft}, k_{yaw} y k_{steer}$).

Para el control longitudinal de velocidad, se utiliza un controlador PI con el que se genera la señal del acelerador para conseguir la velocidad objetivo [9].

2.5.2.3 *Control predictivo por modelo (MPC)*

Las leyes de control definidas en los apartados anteriores son adecuadas para condiciones de conducción moderadas o suaves, es decir, bajas velocidades, trayectorias suaves sin cambios bruscos de dirección o aceleración, y en situaciones con buena adherencia a la carretera. Para situaciones que se salen de estos términos, es necesario aplicar un controlador más complejo con un modelo más exacto.

El Control Predictivo por Modelo o *Model Predictive Control* (MPC), es un método avanzado de control para procesos con un comportamiento complejo que deben cumplir una serie de restricciones. Este tipo de controlador es de gran utilidad en diversos campos, entre ellos la robótica y los vehículos autónomos. El MPC está basado en un proceso iterativo de optimización con un horizonte temporal finito de un modelo de planta. En el caso del control de un vehículo, la función de coste del modelo de planta puede incluir términos como la distancia a la trayectoria o la diferencia de la velocidad actual con la velocidad objetivo, que deben ser minimizados mediante el cálculo de una secuencia de valores para la señal de control que, en el caso estudiado, se tratan del volante y acelerador.

Dado que las restricciones del comportamiento del vehículo (aceleraciones máximas, giro máximo de las ruedas, ...) están bien definidas, la complejidad del algoritmo viene dada principalmente por la determinación precisa del modelo del vehículo elegido, así como la resolución en un tiempo aceptable de la optimización, que incluye expresiones no lineales.

La optimización sobre un modelo no lineal [105] puede incurrir en demasiado tiempo para obtener la solución, o la imposibilidad de encontrarla. El controlador presentado en [106], sugiere la linealización del sistema alrededor del punto de operación en cada instante de tiempo, dando como resultado una solución que, aunque es subóptima, la reducción del tiempo de cómputo puede mejorar la actuación del controlador. El modelo del vehículo está basado en el modelo de bicicleta [107], donde se consideran tanto las dinámicas del vehículo (la masa o momento de inercia), como el comportamiento de

las ruedas ante las distintas fuerzas ejercidas durante la conducción, definido por el modelo de Pacejka [108].

2.5.2.4 *Controladores basados en deep learning*

Los avances recientes en aprendizaje máquina y más concretamente en aprendizaje profundo o *Deep Learning* (DL), han hecho posible la aplicación de estas técnicas también a los algoritmos de control. Los métodos basados en DL, tienen habitualmente un mejor rendimiento en sistemas no lineales, así como una mejor capacidad de generalizar las reglas aprendidas a situaciones nunca vistas [109].

Los principales métodos de aprendizaje son el aprendizaje supervisado y el aprendizaje por refuerzo, los cuales pueden tener distintas arquitecturas de redes neuronales, entre las cuales, las más comunes son la densa, la convolucional o *Convolutional Neural Networks* (CNN), utilizada sobre todo para procesar las imágenes de entrada, y la recurrente o *Recurrent Neural Networks* (RNN), con una gran capacidad para procesar series temporales.

Este tipo de algoritmos permite en algunos casos, que la entrada del algoritmo de control sea la imagen del vehículo directamente, en lugar de la localización y la trayectoria a seguir, eliminando del bucle de control el sistema de localización. Aunque para algunas aplicaciones puede ser útil, utilizar la imagen como única entrada puede restar flexibilidad al sistema, ya que normalmente estos algoritmos están entrenados para circular por un tipo de carretera concreto o para realizar una acción únicamente (seguir el carril). Además, otro aspecto a tener en cuenta en este tipo de algoritmos es su no determinismo, que lleva a no conocer, cuando el sistema falla, los motivos o la causa de este fallo, dificultando su ajuste. Por este motivo, estos controladores requieren complejos procesos de validación y pruebas.

2.5.3 *Predicción de trayectorias*

La predicción de trayectorias, comúnmente aplicada a otros vehículos, pero también extendida a otros agentes como peatones u obstáculos, consiste en generar la trayectoria más probable que un agente va a seguir en los próximos instantes de tiempo. Este horizonte temporal suele abarcar unos pocos segundos en el futuro, que son los que más relevancia y menos incertidumbre tienen (cuanto más larga es la predicción, mayor incertidumbre se tienen, ya que numerosos factores pueden alterar la predicción inicial).

La predicción de trayectorias es esencial para un correcto funcionamiento del planificador de trayectorias. El módulo de planificación de trayectorias local, descrito en la Sección 2.5.4, necesita como entrada, la trayectoria futura que van a seguir los agentes de la escena para poder calcular la mejor trayectoria que debe seguir el VA.

De acuerdo con el estudio realizado en [110], los algoritmos de predicción de trayectorias se pueden clasificar en tres tipos: los basados en un modelo físico, en un modelo de maniobras o en un modelo de interacción.

Los modelos clásicos utilizan la definición del modelo físico del vehículo, que puede incluir tanto la geometría (modelo cinemático) como las fuerzas que afectan su movimiento (modelo dinámico). Este tipo de modelos son habitualmente utilizados para detectar colisiones, ya que el horizonte temporal máximo en el que pueden realizar predicciones no se puede extender demasiado sin aumentar considerablemente la incertidumbre. La mayor desventaja de este tipo de modelos está relacionada con la gran dependencia en las condiciones iniciales, normalmente estimadas con sensores que añaden ruido, y la limitada información del entorno (geometría de la carretera y vehículos en los alrededores). Por otra parte, su simplicidad hace que sean utilizados en numerosas aplicaciones que no requieren una gran precisión. Se pueden encontrar ejemplos de este tipo de modelos en [111, 112].

En los modelos basados en maniobras, el vehículo se considera independiente del resto de la carretera, y el modelo tiene que predecir, de una serie de posibles trayectorias, cuál es más probable que el vehículo siga. Aunque este tipo de modelos pueden considerar los obstáculos de los alrededores o la geometría de la carretera para descartar parte de las maniobras posibles, no consideran las interacciones con otros vehículos que pueden ocasionar un comportamiento no deseado. Un ejemplo representativo se puede encontrar en [113], donde la trayectoria generada es elegida de un conjunto real de trayectorias de acuerdo a la situación actual.

Por último, los modelos que tienen en cuenta las interacciones consideran que el movimiento del vehículo está influenciado por otros vehículos. Este tipo de modelos se basan en detectar e identificar las interacciones de forma precisa, pero modelar y considerar estas interacciones puede ser muy complejo.

Numerosos trabajos han utilizado Redes Bayesianas Dinámicas [114, 115] para resolver este problema, pero en los últimos años, el auge del aprendizaje automático por sus buenos resultados y flexibilidad ha generado una predilección por el uso de estas técnicas para realizar predicciones.

La mayoría de los últimos trabajos relevantes en este campo utilizan DL para predecir la trayectoria futura, donde la mayor diferencia entre los trabajos se encuentra en el preprocesamiento que se realiza a los datos o en el tipo de neuronas utilizadas en la red. En [116], los autores utilizan un tensor de 3 dimensiones para representar el campo potencial de la escena de tráfico en distintos instantes de tiempo, mientras que en [117] y [118], las interacciones entre vehículos se modelan utilizando una capa de atención o de interacción respectivamente en el modelo de la red neuronal. En [119], se utiliza una capa convolucional

de agrupación social (*convolutional social pooling layer*) para codificar el movimiento pasado de los vehículos vecinos y un decodificador basado en la capa *Long Short-Term Memory* (LSTM) para generar la trayectoria futura.

Analizando los modelos que están basados en DL, se pueden observar dos características principales: El uso de la capa LSTM, que mejora la capacidad de analizar las series de datos [119–123], y una arquitectura de codificador-decodificador, también conocido como secuencia a secuencia en los modelos que utilizan LSTM, que es muy utilizado para este tipo de tareas.

La arquitectura *Variational Autoencoder* (VAE) sigue la estructura codificador-decodificador utilizada en la mayoría de los trabajos revisados, está basada en estadística bayesianas y destaca por su capacidad para generar datos de cualquier tipo, así como por generar una representación latente de los datos codificados más regular.

La estructura VAE se ha utilizado en numerosas ocasiones en los últimos años para la generación de trayectorias para vehículos autónomos, tal y como se muestra en los trabajos que se describen a continuación.

El trabajo TrajVAE [124] utiliza un modelo basado en un VAE para generar trayectorias globales realistas (con un horizonte temporal que se extiende desde decenas de segundos hasta minutos) para crear una base de datos. Sin embargo, este tipo de trayectorias globales generadas por este modelo son de naturaleza distinta a las tratadas en este apartado, las cuales dependen de los vehículos de los alrededores y solamente predicen en un horizonte de hasta 5 segundos, aunque con mayor detalle y precisión.

El método propuesto en [125] llamado *Multi-Vehicle Trajectory Generation* (MTG) o Generación de trayectorias multi-vehículo, utiliza un VAE para generar trayectorias de pares de vehículos que conducen a través de una intersección. Este método demuestra la excelente capacidad de generar trayectorias realistas con este tipo de modelo, pero no se puede utilizar para predecir las trayectorias de los vehículos de alrededor ya que este método está orientado a generar escenarios de prueba fiables y no incluye información de los vehículos que se encuentran alrededor de los que se utilizan para generar las trayectorias.

Los dos métodos mencionados anteriormente ([124] y [125]) comparan el rendimiento del VAE con otros modelos generativos como *Generative Adversarial Network* (GAN), que pueden generar reconstrucciones de gran calidad, pero tienden a representar peor todo el domino de los datos [126], haciendo más idóneos para esta tarea los modelos basados en VAE que los que utilizan GAN.

El trabajo presentado en [127], genera predicciones de las trayectorias de los vehículos utilizando un VAE condicional (cVAE), el cual tiene una entrada adicional que corresponde a la intención del vehículo, predicha a su vez por otro modelo que funciona como un

clasificador. Esta arquitectura predice las trayectorias futuras de los vehículos mejorando los métodos con los que se compara. Sin embargo, este método está basado en una buena precisión del modelo clasificador que reconoce la intención del vehículo y hace que el cVAE genere una de las tres clases de maniobras (mantenerse en el carril, cambiar al carril izquierdo/derecho). Además, la base de datos utilizada en este trabajo (NGSIM) no incluye clases para el tipo de maniobra, por lo que han tenido que ser añadidos de forma manual siguiendo un criterio que no se detalla.

Como se puede apreciar en la revisión de la literatura acerca de la predicción de las trayectorias de los vehículos, tarea de gran importancia para el desarrollo de un VA, las últimas líneas de investigación se aprovechan de la gran capacidad para predecir y generalizar de los modelos basados en DL. En especial, los modelos bayesianos como el VAE mencionado en este apartado, parecen ser los más idóneos para resolver este problema y, por lo tanto, los que más investigación requieren para solventar los inconvenientes descritos.

2.5.4 *Planificación*

La planificación de trayectorias es la encargada de generar trayectorias que el vehículo pueda seguir hasta alcanzar una localización objetivo a la que se quiere llegar.

Debido al gran grado de dificultad que tendría generar una trayectoria precisa desde la posición actual del vehículo hasta la final, teniendo en cuenta todos los obstáculos del entorno (tanto estáticos como dinámicos), habitualmente la tarea de planificación se subdivide en dos procesos distintos: uno con mayor nivel de detalle, pero con un horizonte temporal limitado, llamado planificación local y otro que planifica la ruta completa pero con menor nivel de detalle, conocido como planificación global.

2.5.4.1 *Planificación global*

La planificación global (o planificación de la ruta) se encarga de generar una ruta completa desde la posición actual del vehículo hasta el origen a través de la red de carreteras (Figura 2.15). En general, esta red está representada como un grafo dirigido con carreteras, las cuales tienen asociado un coste (habitualmente determinado por el tiempo medio que se tarda en recorrer ese tramo) e intersecciones o uniones entre carreteras.

Este tipo de rutas, por lo tanto, no tienen en cuenta obstáculos dinámicos como otros vehículos ni proporcionan información detallada de la ruta como el carril por el que debe ir, o la trayectoria exacta para esquivar un obstáculo. En consecuencia, esta tarea se puede definir como la búsqueda de una trayectoria que minimice el coste en el grafo definido para la red de carreteras.

Algunos de los métodos más comúnmente utilizados en la búsqueda del camino más corto son el algoritmo de Dijkstra y A* [128], sin embargo, los grafos de algunas carreteras pueden contener millones de nodos, haciendo que estos algoritmos sean incapaces de ejecutarse en un tiempo razonable. La solución para esta situación consiste en utilizar una serie de algoritmos que realizan un preprocesamiento previo del grafo para conseguir reducir el tiempo de cómputo posterior al calcular una trayectoria, a costa de invertir una cantidad de tiempo previo y un espacio de almacenamiento para almacenar los datos de este preprocesamiento [129].

Figura 2.15: Planificación global (rojo) sobre un grafo generado con OSM desde la localización (verde) hasta el objetivo (azul).

2.5.4.2 *Planificación local*

La planificación local tiene como objetivo generar la trayectoria que seguirá el VA en los próximos instantes de tiempo, teniendo como partida, el estado actual del vehículo, y como origen un punto intermedio de la ruta generado por el planificador global. Esta ruta (similar a la de la Figura 2.16) puede incluir maniobras como cambio de carril, giro en intersección, o mantenerse en el carril entre otras. Las trayectorias generadas deben cumplir una serie de requerimientos (curvatura, aceleración linear o aceleración angular máxima) específicas para cada tipo de vehículo, que hagan que la trayectoria pueda ser seguida por el controlador de bajo nivel (Sección 2.5.2). Además, esta trayectoria debe tener en cuenta parámetros de confort para el pasajero (evitar aceleraciones altas y sacudidas), así como evitar los obstáculos detectados por los sistemas de percepción, incluyendo, si estos obstáculos son vehículos, una predicción de su trayectoria futura, generada por los algoritmos de predicción de trayectorias descritos en la Sección 2.5.3. Para ello, el planificador local minimiza una función de coste en la que, además del tiempo de trayecto, penaliza movimientos bruscos que puedan ser incómodos para los pasajeros. La solución exacta a este problema, es decir, la trayectoria óptima, en general es

imposible de obtener (requeriría un tiempo demasiado elevado). Por esta razón, en la mayoría de los casos, se utilizan métodos numéricos de aproximación. Los métodos numéricos para la planificación se pueden dividir en tres categorías [130]:

- Métodos variacionales: Son los métodos más comúnmente utilizados. Plantean el problema como una optimización no lineal de la trayectoria sujeta a una función de coste específica y una serie de restricciones. El principal problema de estos métodos es la imposibilidad de saber si la trayectoria converge a un mínimo global.
- Métodos de búsqueda en grafo: Al contrario que los métodos variacionales, este tipo de planificadores buscan en todo el espacio para obtener el mínimo global. Para ello discretizan el espacio de estados del vehículo y buscan la trayectoria utilizando métodos de búsqueda en grafo como Dijkstra o A*. Como desventaja, este tipo de métodos sólo son capaces de buscar en las trayectorias que se pueden construir sobre sobre el espacio discretizado, pudiendo dar como resultado una solución no óptima o incluso no encontrando ninguna.
- Métodos de búsqueda incremental: Estos métodos se basan en crear un árbol de posibles estados partiendo del estado inicial del vehículo para después analizarlos y seleccionar el mejor de ellos. Estos árboles se van expandiendo y refinando en cada iteración. La principal desventaja está en la incertidumbre del tiempo de ejecución necesario, el cual no se puede acotar, y depende de cada situación.

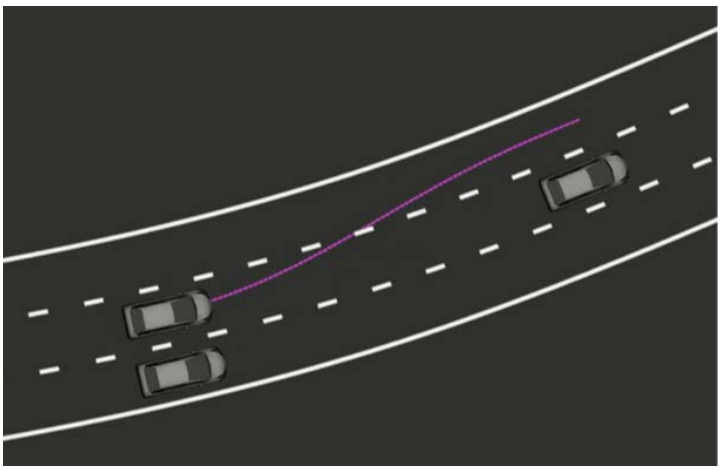


Figura 2.16: Planificación local en simulación.

2.6 INTERFAZ HOMBRE MÁQUINA

La mayoría de los vehículos autónomos que se pueden encontrar en las carretas son prototipos [131], y la mayor parte de los que se pueden encontrar en funcionamiento, operan en entornos industriales. Estos últimos son utilizados para transportar bienes o personas en el interior del recinto en el que operan, reduciendo así tiempo y costes. En este tipo de entornos, las interacciones humano máquina ocurren de forma controlada, ya que los operadores o usuarios de estos vehículos están familiarizados con ellos y conocen su funcionamiento.

Sin embargo, la existencia de vehículos autónomos en las carreteras, pude llegar a sorprender al resto de usuarios de esta, dando cabida a la aparición de situaciones y reacciones inesperadas por parte de los conductores. Por este motivo, es necesario desarrollar algún paradigma de comunicación que pueda ser entendido por ambas partes.

Para desarrollar esta comunicación, es necesario antes estudiar tanto el comportamiento de los peatones en las situaciones en las que interactúan con un VA, así como estimar la confianza que existe actualmente en este tipo de vehículos. Para esto último, se han recogido datos cualitativos en distintos estudios para calcular el grado de confianza actual en los vehículos autónomos, como en [132] , donde a través de una serie de cuestionarios, se determina que una gran parte da la población actual no confía en los vehículos autónomos. La confianza en los robots es definida y analizada en [133], que sirve también de guía para la interacción entre humano y máquina.

Con respecto al estudio e interpretación del comportamiento de los peatones interactuando con vehículos, existe una amplia colección de literatura dedicada a este tema. Como ejemplo de esto, los autores de [134], identificaron los parámetros que afectan en los cruces para tratar de predecir sus intenciones, enfocando esta acción desde dos perspectivas distintas: la del peatón y la del vehículo. La conclusión de este estudio fue que las interacciones estaban basadas en mayor medida por la distancia entre ambos que por el Tiempo hasta la colisión (TTC), corroborando los resultados presentados en [135].

Además, el trabajo [136] muestra que existen distintas localizaciones físicas en la calle, donde los peatones analizan la situación para confirmar la proximidad de un vehículo y garantizar su seguridad. Un indicador de esto se puede encontrar en la frecuencia con la que giran la cabeza, la cual es mayor en los extremos, que en el centro del paso de peatones. Aunque estos estudios establecen los principales parámetros que influyen en la toma de decisiones al cruzar, están basados en situaciones de tráfico convencional con tráfico de vehículos que no son autónomos.

En recientes estudios que se centran en la interacción entre vehículos autónomos y peatones, se ha mostrado que la gente se siente más cómoda cruzando la carretera cuando existe algún tipo de respuesta

por el lado del vehículo, de forma similar a la interacción que se produce con un vehículo conducido de forma manual [137, 138]. En esta misma línea de investigación, un estudio de la Liga Americana de Ciclistas determinó que la imposibilidad por parte de los peatones o ciclistas de comunicarse y establecer contacto visual con un VA, aumentaba sensación de riesgo [139].

Los autores de [140] analizaron la importancia de utilizar interfaces de comunicación entre los peatones y el VA. Para ello, utilizaron un carro de golf manejado por control remoto con una pantalla LED que muestra palabras indicando específicamente a los peatones si debían o no cruzar por delante y desarrollaron un simulador para probar el comportamiento humano en esta situación concreta. Los resultados obtenidos, basados en métodos cualitativos de obtención de datos mostraron que la confianza en la tecnología depende del conocimiento previo sobre las tecnologías de conducción autónoma y en la distancia entre el peatón y el vehículo.

El estudio realizado en [141] evalúa distintos conceptos de interfaces en desarrollo y [142] analiza interfaces de comunicación más avanzadas que utilizan elementos como imágenes que siguen al peatón [143] o formas de comunicación implícitas que incluyen patrones de movimiento del vehículo como frenar [144].

A pesar de que los estudios hasta ahora descritos muestran cómo añadir un monitor o una pantalla a un VA puede ayudar a los peatones a contar con mayor información con la que identificar mejor la intención del vehículo y mejorar la toma de decisiones, existen estudios como [145], que muestran cómo los patrones analizados en los estudios anteriores no son decisivos en la definición de un comportamiento de los peatones. Además, en distintos trabajos [146, 147], los autores concluyeron que las reacciones y el comportamiento de las personas estaba determinado en mayor medida por la distancia y velocidad del vehículo que por la interfaz del VA [148]. Todos los estudios anteriores se centran en definir los factores principales que influyen en el comportamiento de cruzar la carretera de los peatones. Sin embargo, la mayor parte de ellos se basan en datos cualitativos y los estudios con datos cuantitativos se utiliza un paradigma tipo "Experimento Mago de Oz" que trata de imitar el comportamiento de un VA que realmente está controlado por un ser humano.

3 ARQUITECTURA

Diseñar un VA desde cero es un proceso complejo, que requiere un exhaustivo estudio. En primer lugar, es necesario definir de forma clara los objetivos, características y capacidades que el vehículo necesita tener. El vehículo puede estar destinado a transportar sólo personas o sólo bienes, puede ser necesario que circule por autopistas a alta velocidad, por ciudad junto con otros vehículos o por entornos sin carreteras definidas o puede ser necesario que funcione en situaciones adversas (mala visibilidad, suelo muy deslizante, etc.). Todas estas características (entre otras muchas más), deben ser consideradas a la hora de diseñar el vehículo, ya que determinarán elementos clave como la elección de sensores o de software específicos para funcionar en las condiciones requeridas.

A esta elección tanto de hardware como de software de acuerdo a unas condiciones, así como la definición de las relaciones entre los distintos elementos, y la comunicación entre ellos se le conoce como arquitectura.

El diseño de la arquitectura de un VA es una de las primeras fases, y una de las más importantes, ya que los siguientes pasos en el desarrollo del vehículo estarán condicionados por esta arquitectura.

En este capítulo se detalla y justifica la arquitectura del VA desarrollado, describiendo todos los elementos necesarios, tanto de software como de hardware, analizando en profundidad la elección de sensores y su disposición.

3.1 ARQUITECTURA HARDWARE

En esta sección se describen los elementos de hardware que no son sensores (los cuales son descritos en las Secciones 3.2), pero son necesarios para el funcionamiento del vehículo.

3.1.1 *Plataforma*

La parte principal de un VA es el propio vehículo. El VA desarrollado en esta tesis debe ser capaz de circular por carreteras tanto de ciudad como autopistas. Por este motivo, el vehículo del que se parte es un coche convencional no autónomo, al que se le realizarán las modificaciones pertinentes para automatizarlo.

Este capítulo incluye contenido de [2], [3], [4] y [8].

El modelo o tipo concreto de vehículo no influye en ninguno de los apartados de este documento, ya que la arquitectura está pensada para ser flexible y adaptarse a cualquier tipo de vehículo.

El vehículo que se ha utilizado para realizar los ensayos y, por tanto, el vehículo utilizado como plataforma ha sido un Mitsubishi iMiev (Figura 3.1). Se trata de un vehículo eléctrico que puede ser conducido por todo tipo de carreteras hasta una velocidad máxima de 130 km/h y cuenta con una autonomía de 160km. Este vehículo ha sido modificado para que su manejo sea posible mediante comandos enviados por los módulos de control. Los detalles de estas modificaciones se describen en [149].

Figura 3.1: Plataforma de VA basada en un Mitsubishi iMiev.

3.1.2 *Procesamiento*

Cada sensor produce una gran cantidad de datos que necesitan ser procesados, y estas unidades de procesamiento pueden ser también una de las partes con mayor coste del vehículo. Por esta razón, es necesario establecer la capacidad de procesamiento mínima necesaria tanto de Unidad Central de Procesamiento o *Central Processing Unit)* (CPU) como de Unidad de Procesamiento Gráfico o *Graphics Processing Unit)* (GPU) que necesitan los algoritmos ejecutados, para poder ajustar el coste lo máximo posible. Para esta plataforma, se ha dividido la computación entre tres computadoras distintas, permi-

tiendo la ejecución de todos los algoritmos en tiempo real en paralelo, distribuyendo la carga entre las tres computadoras. Cada una de ellas tiene unas características distintas, ya que cada una está enfocada a ejecutar algoritmos de distinto tipo.

Ordenador de control: Este ordenador maneja principalmente la comunicación con el vehículo a través de la interfaz de control basada en el Bus CAN. Además, está encargado de ejecutar las tareas de control y planificación que permiten al vehículo moverse de forma autónoma. La principal característica de este ordenador es una CPU (de 4 núcleos) con una elevada frecuencia de reloj donde pueda ejecutar todas estas tareas críticas en tiempo real.

Ordenador de localización y mapeo: Este ordenador es el responsable de procesar la gran cantidad de datos que producen los tres sensores LiDAR. Entre la información que se puede extraer de las nubes de puntos están la odometría basada en LiDAR y detección de obstáculos mediante distintas operaciones. Además, se encarga de otras tareas de localización que incluyen los sensores GNSS e IMU como el filtrado de la localización con distintos algoritmos. Debido a que la mayoría de los algoritmos de nubes de puntos se ejecutan en CPU y no cuentan con aceleración GPU, este ordenador sólo necesita un procesador suficientemente potente (con una elevada frecuencia de reloj y al menos 4 núcleos) y una memoria RAM de gran capacidad, necesarios para procesar todos los datos en tiempo real.

Ordenador de percepción: La tarea principal de este ordenador es ejecutar todos los algoritmos de DL utilizados para la clasificación de obstáculos. Por este motivo, las características de este ordenador son más específicas, por lo que cuenta con una tarjeta gráfica de alto rendimiento NVIDIA Titan-XP GPU con 12GB de memoria gráfica, que permite al ordenador ejecutar todos los algoritmos necesarios, además de una CPU de 8 núcleos. Por último, este ordenador está conectado a la pantalla del interior del vehículo, por lo que sirve también como interfaz con el usuario de dentro.

Cabe destacar que la elección de estos equipos está motivada por la carga computacional que requieren los algoritmos que se ejecutan en el momento de la escritura de esta tesis. Sin embargo, la flexibilidad de la arquitectura permite aumentar o disminuir el número de equipos o modificar sus características en función de las necesidades en el futuro.

3.1.3 *Comunicación*

Todos los datos producidos por los sensores, así como los datos procesados generados por los distintos algoritmos, necesitan ser compartidos. Los datos de un sensor, al igual que los datos procesados, pueden ser necesitados por distintos algoritmos ejecutados en ordenadores diferentes.

Para compartir toda esta información, se ha utilizado una topología en estrella, donde todos los dispositivos están conectados a un conmutador o *switch* de alta velocidad (al menos 1000 Mbps) que gestiona la distribución de los datos. La principal ventaja de esta topología está en la posibilidad de extenderla de forma sencilla añadiendo más sensores u ordenadores a la red simplemente conectándolos a este conmutador.

Para comprobar que un *switch* de estas características es capaz de gestionar toda la información de los sensores y toda la información generada por los procesos intermedios, se ha realizado un estudio del ancho de banda de cada elemento de la red. La Sección 3.4.3 muestra los resultados de la medición del ancho de banda utilizado por el VA en pleno funcionamiento, el cual es ampliamente inferior a los 1000 Mbps que puede proporcionar el *switch*. De esta manera, se deja margen para poder incluir en el futuro más sensores que cubran distintas zonas, o sensores con mayor densidad de datos.

Por otro lado, además de las comunicaciones entre los sistemas del vehículo, es necesario tener acceso a internet principalmente para obtener las correcciones diferenciales del sensor GNSS, descargar mapas digitales o conectarse a redes de comunicación V2X. Para ello, se conecta un *router* 4G al *switch* proporcionando conexión a internet a todos los ordenadores y al receptor GNSS.

3.1.4 *Potencia*

La última parte por describir del hardware es la potencia. Todos los sensores y ordenadores necesitan ser alimentados. Usar la potencia eléctrica del vehículo eléctrico puede ser peligroso debido al alto voltaje (más de 300 Voltios), y dicha batería puede ser dañada, ya que está diseñada para alimentar únicamente al vehículo. La imposibilidad de utilizar esta batería interna, y el interés de aumentar la flexibilidad de la arquitectura propuesta motivan la incorporación de una etapa de potencia independiente. La principal fuente de potencia es una batería de plomo de 12V y 240Ah con la que se puede obtener una autonomía de los ordenadores y sensores de más de 3 horas según las pruebas realizadas. La batería está conectada a un inversor que convierte los 12V de corriente continua en 230V de corriente alterna para alimentar todos los equipos. La salida del inversor pasa por un Sistema de Alimentación Ininterrumpida (SAI) que permite interrupciones

momentáneas del suministro principal, bien para poder cambiar la batería principal en caliente por otra cargada, o para conectar todos los equipos a la red eléctrica sin necesidad de apagar todo. La Figura 3.2 muestra un esquema de las conexiones.

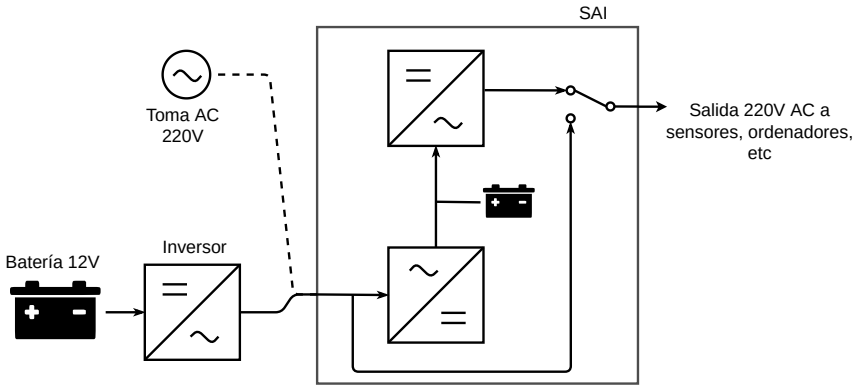


Figura 3.2: Esquema de las conexiones entre los distintos dispositivos desde la batería hasta los sensores y ordenadores.

3.2 SENSORES DE PERCEPCIÓN

En esta sección se describe la configuración de sensores elegida para el VA desarrollado, motivando tanto la elección de las características de cada sensor, como su colocación en el chasis del vehículo.

3.2.1 *Elección de sensores*

De acuerdo con lo descrito en la Sección 2.4.4, la elección de los sensores está estrechamente ligada a competencias de percepción requeridas o, dicho de otro modo, a qué es necesario que el VA sea capaz de detectar, y en qué grado de detalle y precisión.

Para el VA desarrollado en esta tesis, pensado para circular por carreteras con tráfico real y tránsito de peatones, las necesidades de percepción requeridas son las siguientes:

- Detección de obstáculos determinando además su posición con respecto al vehículo y su tamaño.
- Clasificación de obstáculos entre al menos las clases: peatones, vehículos u otros obstáculos.
- Detección de carriles, límites de la carretera o zona navegable, señales de tráfico y cálculo de su posición con respecto al vehículo.

Para la primera competencia (detección de obstáculos), se pueden utilizar distintos sensores, tales como LIDAR, cámara estéreo o radar.

Todos ellos son capaces de detectar, mediante distintos algoritmos, la posición de los obstáculos y estimar su tamaño. Sin embargo, tal y como se muestra en la Tabla 2.2, el sensor que obtiene mayor precisión a la hora de estimar la posición y la forma o tamaño del obstáculo es el LiDAR. La cámara estéreo, a pesar de ser capaz de proporcionar información sobre la distancia del obstáculo, depende en gran medida del algoritmo que calcula la disparidad, pudiendo dar problemas en situaciones de baja visibilidad como la noche o momentos con visión directa del sol. Por otra parte, el sensor radar es capaz de proporcionar, no sólo información de la posición del obstáculo, sino que además calcula la velocidad a la que se desplaza bajo casi cualquier condición meteorológica (poca luz, niebla, lluvia). Sin embargo, el nivel de detalle del obstáculo es bajo comparado con el que se puede obtener con un LiDAR, que genera una nube de puntos más densa y es capaz de funcionar bajo condiciones climatológicas adversas como falta de luz o lluvia y niebla suaves. Como conclusión, se elige el sensor LiDAR para realizar la primera tarea de detección de obstáculos. El número de sensores y sus características se discutirán en el siguiente apartado.

La clasificación de obstáculos es una tarea esencial para el correcto funcionamiento del VA. El comportamiento de un vehículo es distinto si tiene ante él a otro vehículo que, según la circunstancia, puede ser necesario seguirlo o adelantarlo, o si tiene ante él un peatón, en cuyo caso habitualmente el vehículo debe detenerse. Aunque los sensores LiDAR 3D, son capaces de clasificar los obstáculos según los últimos trabajos de la literatura, la utilización de una cámara proporciona un nivel más alto de detalle, además de obtener mejores resultados en la clasificación de obstáculos.

Por último, el caso de la detección de elementos de la carretera tales como líneas o señales de tráfico es similar al de la clasificación de obstáculos. La cámara es el sensor más adecuado ya que es el que obtiene mejores resultados en los trabajos encontrados revisando el estado del arte.

3.2.2 *Disposición y características de los sensores*

La colocación de los sensores en el vehículo es de gran importancia, ya que debe permitir que los sensores capten la información del entorno con suficiente perspectiva para poder detectar la mayor parte posible de los agentes del entorno, y además deben permitir el correcto funcionamiento de los algoritmos de percepción.

3.2.2.1 *Soporte*

Por una parte, la mayoría de los algoritmos de percepción, en especial los basados en DL, han sido entrenados con bases de datos en las que, en la mayoría, los sensores están colocados en horizontal

en el techo del vehículo. Esto dificulta la posibilidad de diseñar configuraciones en las que los sensores que se van a utilizar (cámara y LIDAR) están colocados en posiciones más bajas por debajo de la luna delantera.

Por otra parte, la normativa de circulación impide que existan objetos que sobresalgan de la proyección en planta del vehículo [150], por lo que cualquier sensor que se quiera colocar por debajo del techo del vehículo, requerirá una integración en el chasis que puede resultar inviable para un prototipo, además de dificultar el cambio de la posición del sensor una vez colocado para realizar distintas pruebas.

La solución adoptada, por lo tanto, consiste en una baca colocada en el techo del vehículo mediante unos soportes para baca estándar sobre la que se situarán los sensores. Dicha baca cuenta con diversos perfiles de extrusión de aluminio que permiten colocar y mover los sensores de forma sencilla. Además, el hecho de utilizar este tipo de soporte para los sensores permite, si fuera necesario, trasladar todos los sensores a otro vehículo sin afectar a la calibración de extrínsecos realizada entre todos los sensores.

3.2.2.2 *Cámara*

Para ser capaz de clasificar los obstáculos de forma precisa, es necesario incorporar una cámara. Esta cámara se encuentra colocada en el soporte descrito en el apartado anterior y orientada hacia el frente del vehículo ya que, tanto los obstáculos que interesa clasificar, como los elementos de la carretera relevantes se encuentran principalmente en esta dirección, que es hacia donde tendría visión un conductor humano.

Para elegir la óptica que montará la cámara, hay que realizar un estudio y encontrar un balance entre una óptica con una distancia focal pequeña, que permitirá detectar obstáculos también en los laterales debido a la apertura del campo de visión, y una óptica con una gran distancia focal, que permitirá detectar obstáculos más lejanos.

Para el VA desarrollado, se ha optado por una óptica de 4.5mm, que permite tener visibilidad de parte de los laterales, pero sin renunciar a la detección de vehículos a corta y media distancia.

El uso de una única cámara con una distancia focal media está motivado por unas especificaciones en las que el vehículo no va a circular a altas velocidades y nunca marcha atrás (pues las maniobras de aparcamiento no son objeto de estudio en esta tesis). En el caso de circular a velocidades mayores, sería necesario colocar una serie de cámaras con una distancia focal mayor y con distintas orientaciones para cubrir un área más amplia a mayor distancia.

El modelo de cámara seleccionado cuenta, como características principales, con un sensor de tipo *global shutter*, una resolución de $2048x1536px$ y un tamaño de sensor de $7.1x5.3mm$. El sensor de tipo *global shutter*, a diferencia del tipo *rolling shutter*, permite exponer a

todos los píxeles del sensor al mismo tiempo en la imagen, evitando distintos efectos de distorsión producidos por el método de obtención de las imágenes de los sensores de tipo *rolling shutter*, que escanea todos los píxeles de forma progresiva. La resolución elegida permite obtener imágenes con suficiente detalle para detectar obstáculos en distancias lejanas

3.2.2.3 LiDAR

La principal función de este sensor consiste en proporcionar información sobre la posición de los obstáculos y detectar cualquier elemento cercano al vehículo que pueda ocasionar una colisión. Por ello, este sensor, además de cubrir la misma sección que la cámara para detectar la posición de los obstáculos detectados por esta, debe cubrir también las partes delanteras y laterales cercanas al vehículo donde es necesario comprobar que no hay situado ningún obstáculo que pueda ocasionar una colisión.

Debido a la amplitud del área a cubrir por este sensor, es necesario incorporar más de un LiDAR en la configuración de sensores, obteniendo lo siguiente:

- LiDAR central: Este LiDAR, al igual que ocurre con la cámara, está situado en lo alto del vehículo de forma totalmente horizontal para facilitar la incorporación de algoritmos de detección que han sido entrenados con una configuración similar. Además, esta colocación permite que varios planos coincidan con el ángulo de visión de la cámara, lo cual es necesario para determinar la posición de los obstáculos. Por último, este sensor ha sido elevado con un soporte a $2,05m$ del suelo ($42cm$ del techo del vehículo) para evitar que los primeros planos, que apuntan con un ángulo más negativo choquen con el chasis del vehículo, y así aprovechar el mayor número de planos.
- LiDAR lateral: para cubrir las zonas laterales y la zona frontal cercana al vehículo, que no puede ser cubierta por el LiDAR central debido a la amplitud de los planos, se colocan dos sensores en la parte delantera lateral izquierda y derecha que cubrirán estas zonas. Estos sensores, al no estar destinados a la clasificación de obstáculos, sino a la detección por seguridad de elementos cercanos al vehículo, pueden ser inclinados para aumentar la cobertura.

La colocación de todos los sensores, así como las posiciones relativas teóricas de cada sensor, referidas a la posición del sensor LiDAR de 32 planos se resumen en la Tabla 3.1 y en la Figura 3.3.

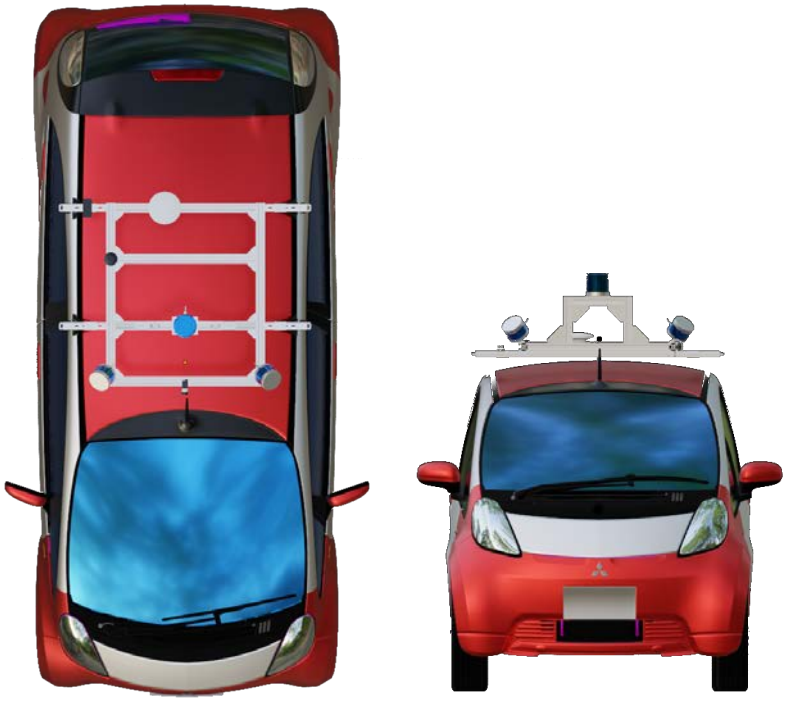


Figura 3.3: Planta y alzado del modelo virtual del vehículo donde se puede apreciar la colocación de los sensores.

3.2.3 *Calibración de sensores*

La calibración de los sensores tiene un papel esencial en funcionamiento de los algoritmos de percepción. Todos los obstáculos, detectados por distintos sensores, deben transformarse a un sistema de referencia común del vehículo para que puedan ser evaluados correctamente. Por otra parte, los algoritmos de detección que fusionan información de distintos sensores necesitan conocer la posición exacta de estos para poder correlacionar los datos de todos los sensores. Este tipo de calibración en el que se determina la posición y orientación de cada sensor con respecto a un sistema de referencia común se conoce como calibración de parámetros extrínsecos.

Para la configuración de sensores del VA desarrollado, es necesario obtener los parámetros intrínsecos de 4 sensores, o lo que es lo mismo, calcular la posición de los 4 sensores con respecto al chasis del vehículo. Las coordenadas de los sensores LIDAR y cámara frontal se pueden obtener con un método de calibración mediante un patrón como el presentado en [151]. Sin embargo, para la calibración de los LIDAR laterales, al no compartir suficientes puntos con el resto de sensores se ha utilizado un método de calibración distinto [152] que permite la calibración entre LIDAR aunque no compartan ningún plano.

Además, algunos sensores como las cámaras necesitan un proceso de calibración adicional. Los sensores como LIDAR o radar propor-

Coordenada	LiDAR 16 izquierdo	LiDAR 16 derecho	Cámara
X (m)	0.23	0.23	0.25
Y (m)	0.38	-0.38	0.0
Z (m)	-0.23	-0.23	-0.27
Roll (º)	0.0	0.0	0.0
Pitch (º)	45.0	45.0	0.0
Yaw (º)	45.0	-45.0	0.0

Tabla 3.1: Coordenadas teóricas de traslación y rotación de los sensores cámara y LiDAR laterales con respecto al LiDAR de 32 planos.

cionan información 3D de cada punto u obstáculo que detectan, sin embargo, relacionar un píxel de la imagen con un punto o una región del espacio no es inmediato. La imagen obtenida por la cámara es distorsionada por su óptica, dificultando esta correlación entre el mundo real y la imagen. Para corregir esto, se realizan una serie de transformaciones a la imagen para que se adapte al modelo *pinhole*, que simula una cámara sin lente que genera una imagen sin distorsión. Dichas transformaciones dependen de una serie de parámetros que se obtienen mediante lo que se conoce como calibración de parámetros intrínsecos [153].

La calibración de intrínsecos de la cámara se ha realizado mediante un método común de calibración basado en la detección de la distorsión de un patrón de tablero de ajedrez. Con este método de calibración, se detectan los bordes de un tablero de ajedrez con número y tamaño de cuadrados conocido en distintas imágenes en las que el tablero se coloca y se orienta de distintas formas. Los parámetros de calibración intrínsecos se pueden obtener con estos datos obtenidos utilizando alguno de los distintos métodos que minimizan el error de proyección de los puntos característicos del patrón.

3.3 ARQUITECTURA SOFTWARE

La arquitectura software describe todos los módulos de software incluidos en el VA desarrollado, así como el flujo de información que discurre entre ellos.

En la Figura 3.4 se muestra un diagrama simplificado de todos los módulos de software necesarios para hacer que el vehículo se mueva de forma autónoma. El proceso de conducción autónoma comienza con un usuario indicando que quiere trasladarse a una localización determinada a través de la Interfaz Gráfica de Usuario o *Graphics User Interface* (GUI) (morado). Utilizando la información de los sensores (gris), se obtienen la localización del vehículo con los módulos de

localización (azul) y las detecciones de los carriles y los obstáculos de la carretera con los módulos de percepción (rojo). Por último, el software de control y planificación (verde) generará los comandos de control que se envían al vehículo (amarillo). Estos comandos, junto con los mensajes mostrados en la Interfaz Hombre Máquina o *Human-Machine Interface* (HMI), modificarán el entorno (la posición, del vehículo, el comportamiento del resto de vehículos y peatones, etc.), generando unas nuevas características del entorno, que son de nuevo percibidas por los sensores, cerrando así el bucle de control del VA.

A continuación, se describen con más detalle el funcionamiento y características de todos estos módulos.

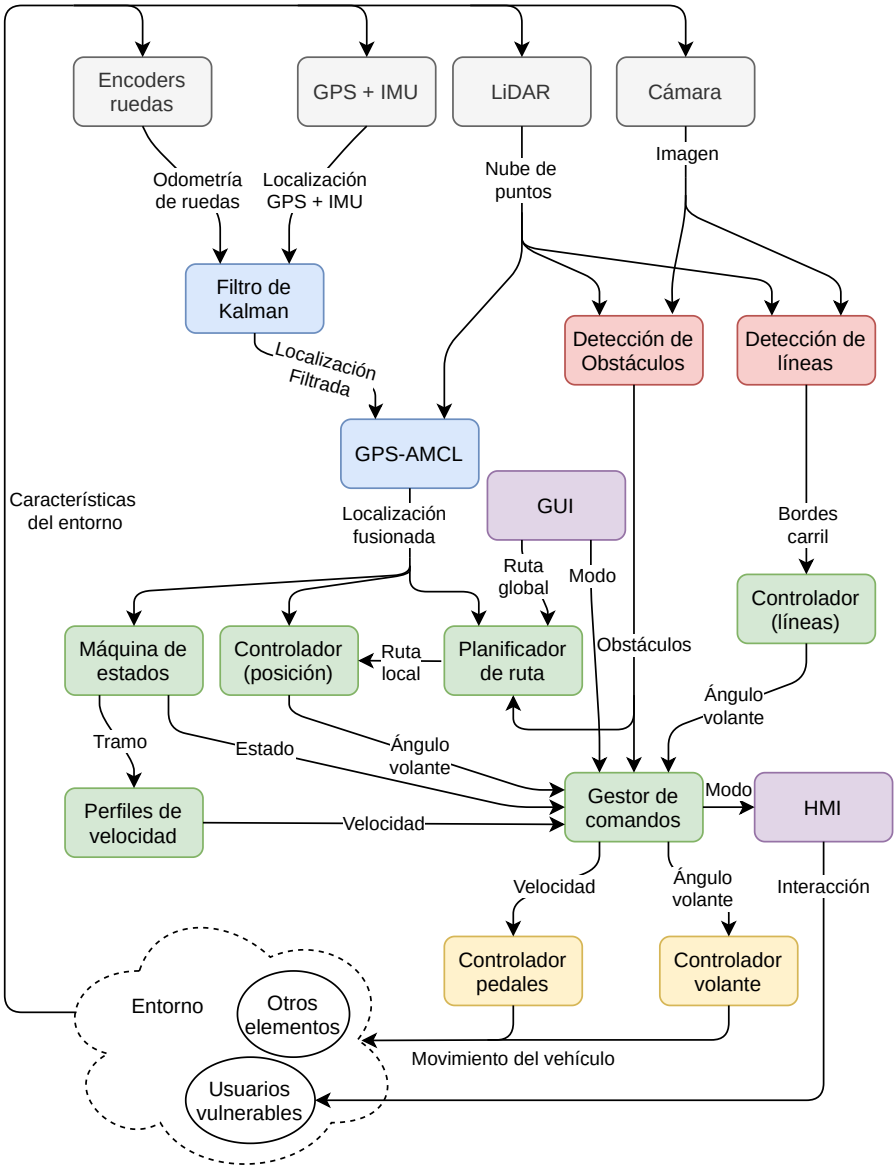


Figura 3.4: Esquema de las relaciones entre los distintos módulos de software incluidos en la arquitectura, así como el tipo de datos de entrada y salida.

3.3.1 *Adquisición de datos*

Los módulos de adquisición de datos están compuestos principalmente de distintos controladores o *drivers* que se encargan de leer los datos en crudo de los sensores. Una vez estos datos son recibidos, la tarea de estos módulos tiene dos partes: En primer lugar, los datos de entrada son transformados a valores de magnitudes que queremos medir (distancia, posición, etc.) y son preprocesados. Este preprocesamiento puede consistir en un algoritmo de filtrado para reducir el ruido de la medida, o cualquier otro tipo de algoritmo que genere

información más compleja de los datos recibidos. En segundo lugar, estos *drivers* deben convertir los datos del sensor a un formato de mensaje de ROS, para que sea posible compartir la información del sensor con los distintos módulos que la necesiten. De este modo, la información recibida por los sensores es transformada a un formato estándar consiguiendo que el resto de los componentes de la arquitectura sea capaz de funcionar independientemente del sensor que le esté enviando la información.

Un ejemplo de las ventajas de esta estandarización de los mensajes de los sensores se puede observar en un módulo del vehículo que necesite una nube de puntos 3D como entrada para detectar obstáculos o para localizarse. Esta nube de puntos, siempre que esté en el formato de un mensaje estándar de ROS, puede llegar, no sólo de distintos tipos de sensores LiDAR, con distintos números de planos o distintas configuraciones de estos planos, si no que se puede obtener de sensores totalmente distintos como cámaras de tiempo de vuelo o cámaras estéreo sin afectar al algoritmo que recibe esta información (salvo algoritmos optimizados para un sensor con unas características concretas).

El vehículo estudiado en esta tesis obtiene datos del entorno principalmente a través de cuatro tipos de sensores distintos, los cuales son descritos a continuación:

3.3.1.1 *Encoders de las ruedas y volante*

El vehículo cuenta con dos tipos de *encoder* (o codificador rotatorio). El primero se utiliza para medir la velocidad de giro de las ruedas sobre su eje, y por lo tanto estimar la velocidad del vehículo. El segundo tipo proporciona el ángulo de giro sobre el eje de la rueda perpendicular al plano del suelo, indicando la dirección del vehículo. Este último giro de las ruedas no es el mismo para ambas ruedas directrices, por lo que habitualmente se indica el ángulo de giro del volante, a partir del cual, conociendo la geometría del vehículo se puede estimar el de ambas ruedas.

Además, este módulo utiliza estos dos ángulos de giro para generar una odometría del vehículo a partir de las ecuaciones de movimiento del vehículo (Sección 2.5.1.3).

3.3.1.2 *GNSS e IMU*

Estos dos sensores, se suelen encontrar combinados e integrados en un solo dispositivo que proporciona una localización global que, según las circunstancias, puede llegar a ser muy precisa. Estos datos de localización se pueden enviar en distintos formatos, siendo NMEA [154] el más común y estándar. Por lo tanto, es necesario disponer de un *driver* que convierta esta trama de datos a un mensaje de ROS estándar.

3.3.1.3 LiDAR

Los sensores de este tipo generan una nube de puntos con las mediciones realizadas a los objetos de su alrededor. Aunque existen distintos fabricantes que producen este tipo de sensores, la mayoría son muy similares en cuanto a características e incluyen un *driver* que transforma estos puntos a un mensaje de ROS de tipo *PointCloud2*. Si bien es cierto que todas las nubes de puntos se pueden publicar bajo ese tipo de mensaje, además de los campos habituales de una nube de puntos (coordenadas x, y, z, e intensidad), cada modelo concreto puede incluir otro tipo de campos (como ruido de la medición o número de plano), por lo que es necesario prestar atención a este punto para asegurarla compatibilidad de distintos sensores con los módulos que utilizan la información de este sensor.

3.3.1.4 *Cámara*

Al igual que con el sensor LiDAR, existen numerosos fabricantes de cámaras, la mayoría de los cuales incluyen un *driver* que transforma la imagen a un mensaje estándar. Además, es necesario calibrar los parámetros intrínsecos de la cámara para generar un mensaje de tipo *CameraInfo*, necesario por los módulos que utilicen la imagen de la cámara.

3.3.2 *Localización*

Los módulos incluidos en la parte de localización tratan de combinar todas las posibles fuentes de localización y odometría para generar una localización fusionada que sea más precisa y robusta a distintos entornos.

A grandes rasgos, la localización consta de dos módulos principales: El primero, es un filtro de Kalman de tipo *Unscented* que filtra la localización obtenida por los sensores GNSS e IMU con la odometría de las ruedas, generando una localización más estable y con menos ruido. El segundo módulo combina esta localización con la generada comparando un mapa de los alrededores del vehículo con las lecturas del sensor LiDAR.

Al ser la localización una de las partes principales de esta tesis, se le ha dedicado un capítulo completo en el que se describirá con más detalle el algoritmo utilizado y su funcionamiento (Capítulo 4).

3.3.3 *Percepción*

Los módulos de percepción integrados en el vehículo son los responsables de analizar los datos en crudo de los sensores y extraer información relevante sobre la escena. Estos módulos se pueden clasificar en dos grupos:

3.3.3.1 *Detección y clasificación de obstáculos*

Un VA debe ser capaz de identificar al resto de agentes que se encuentran en la carretera, como vehículos, peatones o ciclistas. Esta tarea puede ser realizada utilizando distintas técnicas, sin embargo, la tendencia en los últimos años se inclina hacia el uso de métodos de DL para detectar y clasificar los distintos obstáculos alrededor del vehículo.

Para realizar esta detección y clasificación de obstáculos, este módulo consta de un algoritmo de DL que detecta en cada imagen capturada, los obstáculos de las clases vehículo, peatón y ciclista que se encuentran en la carretera indicando además su posición en la imagen, su tamaño y orientación [155]. Aprovechando la calibración de parámetros extrínsecos entre la cámara y el LiDAR es posible, utilizando para la cámara un modelo *pin-hole*, proyectar los puntos correspondientes a la detección de los obstáculos y obtener la correspondencia entre los píxeles del obstáculo y los puntos tridimensionales de la nube de puntos del LiDAR, obteniendo una posición tridimensional para cada obstáculo más precisa.

Además, la nube de puntos de todos los sensores LiDAR es analizada para detectar obstáculos que superan cierta altura del suelo. Esto permite evitar colisiones con obstáculos que no han sido detectados por la cámara (por malas condiciones de iluminación o por no pertenecer a las clases a detectar entre otros motivos) o que se encuentran fuera de su rango de visión (laterales o parte frontal muy cercana).

3.3.3.2 *Detección de elementos de la carretera*

Además de los obstáculos, un VA debe ser capaz de reconocer los distintos elementos que se pueden encontrar en la carretera que aportan información sobre cómo se debe circular en cada tramo. Además de las señales de tráfico, que indican qué maniobras están permitidas o los límites de velocidad, uno de los elementos más importantes son los límites de la carretera o de cada carril, determinados por líneas. La correcta detección de las líneas de la carretera es una parte crítica de la arquitectura, ya que el vehículo tiene la funcionalidad de circular de forma autónoma siguiendo el carril en el que se encuentra. El procedimiento seguido por el módulo de detección de líneas es el siguiente: En primer lugar, se utiliza un algoritmo de DL para detectar en la imagen de la cámara los píxeles que corresponden a las líneas de la carretera [156]. De todas las detecciones, se seleccionan únicamente los píxeles correspondientes a las dos líneas del carril en el que se encuentra el vehículo. Estos píxeles son proyectados al espacio tridimensional utilizando la información del LiDAR de forma similar a la proyección de las detecciones de obstáculos, de tal forma que se

puedan obtener las coordenadas tridimensionales de los carriles de forma más precisa.

3.3.4 *Control y planificación*

Este módulo del vehículo depende en gran medida de los módulos de localización y percepción. Una vez que la localización del vehículo es adquirida con precisión, y los elementos del entorno correctamente identificados, el planificador de trayectorias genera una serie de puntos que el vehículo debe seguir mediante un módulo de control que genera los comandos para el volante y los pedales de acelerador y freno. Se han evaluado y probado en el vehículo los distintos tipos de controladores revisados en la Sección 2.5.2. Dadas las condiciones operativas del VA desarrollado definidas en la Sección 1.2, se han descartado todos aquellos controladores que requieren un modelo preciso del vehículo (como MPC o controladores basados en modelos de DL) debido a que las principales ventajas se aprecian a partir de velocidades muy superiores a las que va a funcionar el VA y a la dificultad de obtener un modelo físico preciso del vehículo o entrenar una red neuronal. En su lugar, se ha optado por el controlador Stanley (Sección 2.5.2.2), un controlador más sencillo y robusto que permite generar los comandos de control necesarios para seguir trayectorias a bajas velocidades. El ajuste de este controlador se realiza de forma experimental con un método de prueba y error para determinar los valores de las constantes de la función matemática que calcula la posición del volante. El ajuste de estas constantes debe conseguir hacer que el vehículo alcance un equilibrio entre seguir los puntos de la trayectoria lo más cerca posible y no sobreoscilar, es decir, evitar correcciones de volante demasiado bruscas. Alcanzar este equilibrio requiere numerosas pruebas, realizadas en un circuito cerrado, así como un cuidadoso ajuste ya que en general, las combinaciones de parámetros que consiguen un seguimiento muy cercano de la trayectoria aumentan las posibilidades de sobreoscilación y viceversa.

3.3.4.1 *Planificación local de trayectorias*

El módulo de planificación local es el encargado de generar una trayectoria local, es decir, de unos pocos segundos de recorrido y que incluya información y precisión suficiente para que el vehículo pueda realizar maniobras complejas como giros o adelantamientos. Esta planificación necesita la información de localización del vehículo, la ruta global que debe seguir y los obstáculos del entorno detectados por los módulos de percepción.

Para realizar una correcta planificación de la trayectoria local del vehículo es deseable, no sólo conocer la posición de los obstáculos de alrededor, sino ser capaz de predecir su futura trayectoria. Por este motivo, este módulo de planificación incluye un algoritmo de

predicción de trayectorias que, al ser objeto de un exhaustivo estudio en esta tesis, cuenta con un capítulo completo dedicado en el que se describe de forma más detallada (Capítulo 5).

Una vez obtenidas las predicciones de movimiento de los obstáculos del entorno, se procede a calcular la trayectoria local que debe seguir el vehículo. Para ello, se ha utilizado el algoritmo descrito en [157], el cual ha sido modificado para añadir la capacidad de considerar las predicciones del movimiento de los obstáculos, así como la función de coste para incluir comportamientos que no se incluyen en el algoritmo original (como la preferencia de circular por el carril derecho).

La combinación de estos dos algoritmos permite dotar al VA de la capacidad de adaptarse a las características del entorno actual de la carretera, generando comportamientos más complejos como adelantamientos o cambios de carril.

3.3.4.2 *Máquina de estados*

El comportamiento de un vehículo no es siempre igual, ya que depende de las características del entorno por el que circula: no es lo mismo circular por una autopista, a una velocidad elevada, que hacerlo por ciudad más despacio o en las inmediaciones de un paso de cebra, donde es necesario parar en presencia de peatones.

A falta de un módulo de percepción que distinga con gran precisión todas las señales y características de la carretera de forma fiable, y dado que el entorno por el que circulará el vehículo es siempre conocido (según se ha definido en los objetivos), este módulo suple al de detección de señales y además permite incorporar distintos estados personalizados que no dependen de la señalización vial, como lugares de parada para recoger pasajeros.

Para ello, este módulo se encarga de monitorizar la posición del vehículo (obtenida del módulo de localización) dentro de la ruta global y publicar el estado o modo en el que debe funcionar (búsqueda de peatones en paso de cebra, seguimiento de carril, etc.), así como un identificador del tramo de la ruta en el que se encuentra, variables que serán utilizadas por los módulos de gestión de comandos y perfiles de velocidad respectivamente. Estos estados y tramos deben ser añadidos de forma manual a una lista en la que se indica la localización a partir de la cual se activa cada estado y comienza cada tramo.

3.3.4.3 *Controlador basado en líneas*

El control lateral del vehículo basado en las líneas de la carretera utiliza como entrada la detección, en forma de puntos tridimensionales del módulo de detección de carriles descrito en la Sección 3.3.3.2.

Con estos puntos, se realiza una regresión polinómica de segundo grado con las componentes x e y de los puntos de los carriles, de tal manera que se obtienen, para cada carril, los valores a, b y c

que minimizan el error entre estos puntos y la ecuación de la curva $y = a \cdot x^2 + b \cdot x + c$.

Para calcular la señal de control, es necesario obtener, a partir de las líneas detectadas tres valores que indicarán la posición del vehículo y las características del carril. Para mejorar la estabilidad del controlador, estos valores se calculan para un punto que se encuentre a una distancia de l metros por delante del vehículo, simulando la percepción de un conductor humano, que mantiene el vehículo en el centro del carril mirando siempre la zona de las líneas de la carretera que se encuentran por delante del vehículo a una distancia determinada conocida como *look-ahead distance* [158]. Estos tres valores se calculan de la siguiente manera:

- Error transversal: Se trata de la distancia al centro del carril, que se puede obtener a partir de la siguiente expresión, que calcula la distancia transversal a cada carril dt:

$$dt = a \cdot l^2 + b \cdot l + c. \tag{3.1}$$

  El error transversal e, se calcula a partir de la distancia transversal al carril derecho (negativa) dt_d e izquierdo (positiva) dt_i utilizando la expresión

$$e = \frac{dt_i + dt_d}{2} - dt_i. \tag{3.2}$$

- Error angular (Ψ_e): Mide el error de orientación entre el vehículo y los carriles. Para ello, se utiliza la expresión

$$\Psi_e = arctan(b + 2 \cdot a \cdot l) \tag{3.3}$$

  que calcula el error angular para cada carril, obteniendo el valor final de error angular como el valor medio del error en ambos carriles.

- Curvatura de la carretera: El último valor necesario para el controlador es la curvatura r_e de la carretera, definida como la inversa del radio de curvatura, definida para cada carril como:

$$r_e = \frac{((b + 2 \cdot a \cdot l)^2 + 1)^{\frac{3}{2}}}{a}. \tag{3.4}$$

  El valor final de la curvatura se calcula como la inversa de la media de las inversas de la curvatura de cada carril.

Con estos datos, se puede obtener el ángulo de giro del volante del vehículo para mantener su posición centrada entre los dos carriles. Para el vehículo desarrollado, se ha utilizado una adaptación del

controlador Stanley [9] descrito en la Sección 2.5.2.2. El ángulo de giro δ se calcula con la expresión

$$\delta = k_{ang} \cdot \Psi_e + arctan(\frac{k \cdot e}{k_{soft} + v}) + k_{curv} \cdot r_e \tag{3.5}$$

donde v es la velocidad del vehículo y k_{ang}, k, k_{soft} y k_{curv} son las constantes que indican el peso de cada componente y deben ser ajustadas de forma experimental.

3.3.4.4 *Controlador basado en localización*

Este controlador, también calcula el ángulo de giro del volante necesario para que el vehículo siga una trayectoria. A diferencia del controlador descrito en el apartado anterior, el objetivo de este controlador radica en hacer circular al vehículo lo más cerca posible de una trayectoria definida por una serie de posiciones en el espacio. Además, en lugar de obtener los valores de las variables de error transversal, angular y curvatura a partir de las detecciones de los carriles de la carretera, estos valores son calculados utilizando la posición global que devuelve el módulo de localización, y la trayectoria generada por el módulo de planificación de trayectorias de la siguiente manera:

- Error transversal: Es la distancia entre el punto del vehículo a evaluar y la recta definida por el punto de la trayectoria más cercano y su orientación. Este valor debe ser positivo si el vehículo se encuentra a la izquierda de la trayectoria y negativo si está a la derecha.
- Error angular: Este valor se obtiene directamente de la diferencia entre el ángulo del punto del vehículo a evaluar, y el ángulo del punto más cercano de la trayectoria.
- Curvatura de la trayectoria: Este valor se puede obtener directamente de la trayectoria generada por el planificador. El valor de la curvatura r_e se obtiene utilizando dos puntos consecutivos ($P1$ y $P2$) del tramo de trayectoria que se está analizando utilizando la expresión

  $$r_e = 2 \cdot d(P1, P2) \cdot sen(\frac{P1_\psi - P2_\psi}{2}), \tag{3.6}$$

  donde $d(P1, P2)$ determina la distancia entre los dos puntos y $P1_\psi$ y $P2_\psi$ la orientación de cada uno de ellos.

Con estos valores, el ángulo final de giro del volante se obtiene de forma similar al caso anterior utilizando la expresión 3.5, donde los valores de las constantes deben ser ajustados de nuevo para este controlador.

3.3.4.5 *Perfiles de velocidad*

El módulo de Perfiles de velocidad tiene almacenada una lista con las velocidades deseadas en cada tramo de la ruta, así como la aceleración o deceleración máxima permitida para alcanzar esta velocidad objetivo. Para cada identificador de tramo recibido de la máquina de estados, este módulo envía una velocidad y aceleración objetivo al módulo Gestor de comandos.

3.3.4.6 *Gestor de comandos*

Este último módulo recibe toda la información acerca del control del vehículo (longitudinal y lateral), estado o situación de circulación, modo de funcionamiento y presencia de obstáculos. Actúa como un sistema de toma de decisiones que, analizando toda esta información, decide qué comandos de control enviar al controlador de bajo nivel del vehículo.

De forma simplificada, la lógica de funcionamiento de este módulo es la siguiente: La velocidad final será la publicada por el módulo de Perfiles de velocidad salvo en el caso que desde la GUI el pasajero active el modo parada, o estando en el estado que indica cercanía a un paso de cebra, se detecte un peatón, en cuyos casos, la velocidad publicada será cero o de parada. Para los comandos de control del volante, en función del estado definido en cada tramo, se publicará la señal de control del controlador basado en las líneas de la carretera o del controlador basado en la localización.

3.3.5 *Interfaz de usuario gráfica*

La GUI, es la encargada de gestionar la interacción entre los usuarios del VA y el propio VA.

Los pasajeros, a través de esta interfaz pueden indicar, señalando en un mapa, la localización a la que desean desplazarse. Este módulo, generará, en base a la elección del usuario, una ruta global que será procesada por el módulo planificador de ruta. Esta ruta se puede generar utilizando un algoritmo de búsqueda como A* en un mapa de la red de carreteras representada como grafos, el cual se puede obtener directamente de OSM. El usuario también puede controlar funciones básicas del VA, como indicarle que se detenga, o que continúe su marcha.

Por otro lado, la GUI muestra a los pasajeros información acerca de la ruta como velocidad, localización en el mapa o detección de obstáculos.

3.3.6 *Interfaz hombre máquina*

La HMI gestiona la interacción entre el VA y los peatones, a diferencia de la GUI que únicamente interactúa con los pasajeros de dentro del vehículo.

La principal función de la HMI, es influir en el entorno, en concreto en los peatones o usuarios vulnerables de la carretera a favor del VA, es decir, tratando de generar una situación más segura y un comportamiento de los peatones más predecible.

Para ello, la HMI muestra distintos mensajes en función de la situación en la que el vehículo se encuentra (deteniéndose por un obstáculo, reanudando la marcha, etc.) con la intención de ser percibidos por los peatones para que conozcan las intenciones del VA. De esta manera, se pueden evitar situaciones potencialmente peligrosas que involucren a los peatones.

El estudio de estas interfaces ha sido analizado con gran profundidad en esta tesis, por lo que se le ha dedicado un capítulo completo en el que se analizan y estudian distintas opciones (Capítulo 6).

3.3.7 *ROS*

La arquitectura software del vehículo está basada en ROS [58] (véase Sección 2.2.2.2). ROS es un entorno de trabajo para desarrollar software de robots que permite lanzar múltiples algoritmos en paralelo (nodos) e implementa la comunicación entre ellos con mensajes (*topics*) o servicios. Utilizar ROS aporta flexibilidad al sistema haciendo posible reemplazar un algoritmo por otro similar o un sensor por otro siempre que utilicen el mismo tipo de mensajes. También ayuda a balancear la carga computacional ya que los recursos empleados por cada algoritmo se pueden configurar. Otra gran ventaja del entorno ROS es la gran librería de software con la que cuenta. Incluye desde controladores para múltiples sensores (LIDAR, cámara GNSS, etc.) hasta distintos algoritmos muy útiles para robótica o vehículos autónomos (planificación de trayectorias, SLAM, percepción, etc.).

3.3.8 *Seguridad*

Uno de los puntos más críticos en los vehículos autónomos es la seguridad. El vehículo debe ser capaz de detectar cualquier anomalía o problema para reaccionar acorde de forma segura o ceder el control al conductor que supervisa. Para ello, la arquitectura propuesta incluye un sistema para evaluar si las condiciones necesarias para que el vehículo pueda circular de forma segura en modo autónomo se cumplen. Estas condiciones están evaluadas en base a los siguientes factores:

- **Conciencia propia**. Este indicador proviene del módulo que comprueba el estado de los sensores y el *heartbit* de cada proceso de software que se está ejecutando en el sistema.
- **Precisión de la localización**. Si el vehículo no es capaz de localizarse en el mundo con suficiente precisión para navegar de forma autónoma, los comandos de control pueden resultar en un comportamiento inadecuado. Por lo tanto, este indicador representa el estado de la localización, haciendo que el vehículo utilice otro algoritmo de control distinto o, en último caso pare.
- **Trayectoria inalcanzable**. Este indicador proviene del módulo de planificación. El vehículo se detendrá si el módulo de planificación no es capaz de generar una trayectoria hacia el destino, o dicha trayectoria no puede ser seguida porque transcurre por fuera del área navegable o presenta un obstáculo.

La detección de un fallo relacionado con alguno de estos factores provocaría un comportamiento de recuperación o, en el peor de los casos, la parada del vehículo.

3.3.9 *Simulación*

La simulación, aunque no forma parte del bucle de control del vehículo, es de gran importancia para el desarrollo de un VA debido a su capacidad de simplificar el proceso de probar nuevas características software o de obtener datos para realizar pruebas, sin ser necesario el uso de la plataforma real.

En la simulación la parte del entorno real representado en la Figura 3.4, es sustituida por un simulador configurado de tal forma que tanto el vehículo como el entorno sean lo más parecido posible a la realidad. El único elemento que no se simula es el módulo HMI y la interacción con los peatones, ya que el comportamiento humano no puede ser simulado de forma precisa.

Para la plataforma desarrollada se ha utilizado el simulador CARLA para crear un "gemelo digital" de la plataforma dentro de un entorno de simulación, consiguiendo una forma rápida de diseñar y probar un VA, siendo esta última parte esencial, dado que la legislación española no permite circular a un VA en carreteras en condiciones de tráfico normal.

CARLA es un simulador de código abierto para vehículos autónomos especializado en entornos urbanos [60]. Está diseñado sobre *Unreal Engine 4*, un motor gráfico diseñado para videojuegos que permite simular gráficos visuales, físicas y entornos muy realistas. Además, CARLA incluye los sensores más comunes en un VA con un alto grado de personalización, permitiendo simular el comportamiento de cualquier sensor de los descritos en la Sección 3.2, si se

parametrizan de forma correcta. Por otro lado, CARLA es completamente compatible con ROS gracias a un módulo de software oficial llamado *CARLA ROS bridge* [159], haciendo que este simulador sea apropiado para el tipo de arquitectura propuesta para la plataforma desarrollada.

El software completo de la plataforma, a excepción de la HMI, es replicado dentro del vehículo simulado en CARLA. La estandarización de los mensajes de ROS, hace posible sustituir los sensores reales por los simulados, permitiendo al gemelo digital percibir el entorno simulado como si fuera real. Por consiguiente, la cámara y los sensores LiDAR, GNSS e IMU han sido replicados en el simulador ajustando los parámetros de CARLA con los datos de las hojas de características de los sensores para obtener la mayor correlación posible entre realidad y simulación.

Una vez añadidos, las lecturas simuladas son publicadas en mensajes de ROS, haciendo posible que acceda a ellos cualquier módulo de la arquitectura. De forma similar, los comandos de control generados por el software pueden ser directamente leídos por el vehículo simulado, cerrando el bucle de control de la simulación. De esta forma, es posible obtener información de todo tipo del simulador y probar o realizar ajustes a los algoritmos utilizados en la arquitectura para validarlos sin realizar pruebas reales. Cabe destacar que tanto el software utilizado para la simulación como el del vehículo real es exactamente el mismo, haciendo que ambas plataformas se comporten de manera similar.

3.4 RESULTADOS Y CASO DE USOS

Esta última sección presenta los resultados de los experimentos realizados con el VA. El principal objetivo de estos experimentos es demostrar que la arquitectura es completamente funcional, y los requerimientos y objetivos propuestos en el capítulo 1 han sido logrados. Para probarlo, se han realizado varios experimentos que evalúan distintos aspectos de la arquitectura y cuyos resultados se presentan a continuación.

3.4.1 *Casos de uso*

Las pruebas finales del vehículo se han realizado en dos tipos de escenarios distintos. El primero consiste en una pista destinada a realizar pruebas en la que se han podido ajustar todos los elementos y parámetros del vehículo. El segundo escenario es un tramo de carretera real, en el que con el tráfico cortado (ya que la legislación actual no permite la circulación de un VA), se ha demostrado el funcionamiento del vehículo.

El funcionamiento completo empieza con un pasajero que espera en un punto de recogida. Una vez dentro del vehículo, el pasajero selecciona en la GUI el destino al que se quiere dirigir y pulsa el botón de iniciar la marcha. En ese momento, el vehículo comienza a circular de forma autónoma hacia el destino seleccionado. Durante este trayecto, el vehículo es capaz de circular por el carril correspondiente a una velocidad variable que se adecua a la morfología de la carretera y a los elementos de esta (decelerando en curvas cerradas y en las proximidades de los pasos de cebra, y acelerando en las rectas). Además, ante la presencia de un peatón esperando en un paso de cebra, el vehículo se detiene y le indica a través de la HMI, que ha sido detectado para que cruce la carretera con seguridad. De igual manera, ante un vehículo que circule por delante, el VA se ajusta a su velocidad si es más lenta, deteniéndose si fuera necesario. Una vez llegado al destino, el vehículo se detiene para que el pasajero se baje. Desde este punto, el vehículo puede dirigirse a otra localización distinta en la que recoger a otro pasajero, repitiendo el proceso las veces que sean necesarias.

Cabe destacar que, aunque el pasajero no debe montar en el puesto del conductor (puede ocupar cualquiera de los otros tres espacios), ya que el vehículo se controla de forma autónoma, el puesto del conductor está siempre ocupado por una persona con conocimientos del funcionamiento del vehículo. Esta persona, aunque no actúa en ningún momento sobre los controles del vehículo, por seguridad y por legislación debe estar siempre supervisando el funcionamiento para, ante un comportamiento anómalo, actuar de forma acorde.

3.4.2 *Carga computacional*

El objetivo de la arquitectura de hardware elegida, en la que el procesamiento se divide entre distintas computadoras, consiste en permitir que todos los algoritmos se ejecuten correctamente, de tal forma que el desempeño de los algoritmos no se vea afectado por una CPU sobrecargada o falta de memoria.

Para comprobar que la arquitectura seleccionada es adecuada y suficiente para ejecutar todos los algoritmos necesarios, se ha realizado un análisis, para cada ordenador, la carga computacional que necesita cada algoritmo ejecutado.

La Tabla 3.2 resume los datos obtenidos en los que se muestran los valores de consumo de CPU y de memoria RAM de los algoritmos más pesados computacionalmente.

Con estos datos, se puede comprobar que el hardware elegido es suficiente para ejecutar de forma correcta toda la arquitectura software, sabiendo las características concretas de los ordenadores elegidos, descritos en la Sección 3.1.2, que superan holgadamente los requerimientos de memoria RAM y CPU necesarios.

Algoritmo	Núcleos (%)	Memoria (Mb)	Unidad de procesamiento
Clasif. obs.	80	250	Percepción
Localización	18	170	Percepción
Obs. LiDAR	73.3	195	Loc. y map.
Driver LiDAR	63.6	70	Loc. y map.
Detección carril	147.6	2861	Percepción
Control	46.4	248	Control
Visualización	170	595	Percepción
Bus CAN	44.2	136	Control

Tabla 3.2: Resumen del procesamiento requerido de los algoritmos con mayor carga.

3.4.3 *Ancho de banda*

La arquitectura elegida, en la que la computación está distribuida entre distintas unidades de computación, conectadas entre sí y a los sensores y actuadores a través de una red cuyo elemento central es un *switch*, sólo es posible siempre que dicha red no se sature por el flujo de datos.

Por este motivo, es necesario analizar el ancho de banda total consumido por todos los elementos de la arquitectura, para comprobar que la arquitectura es estable. Para ello, durante el funcionamiento normal del VA, se ha medido el ancho de banda consumido por cada elemento de la arquitectura.

La Tabla 3.3 resume el ancho de banda para cada sensor y para los módulos de software que más ancho de banda consumen. Teniendo en cuenta estos datos, con la configuración del VA propuesta, se obtiene un ancho de banda de 349.3 Mbps, inferior a los 1000 Mbps que puede proporcionar el *switch*. De esta manera, se deja margen para poder incluir en el futuro más sensores que cubran distintas zonas, o sensores con mayor densidad de datos.

En caso de que el ancho de banda de la arquitectura hubiera sido superior al que puede proporcionar el *switch*, sería necesario estudiar la implementación de otro tipo de arquitectura de las definidas en la Sección 2.2.1, sin embargo, al contar con un holgado margen, no se ha visto necesario su estudio para este vehículo.

Sensor	Ancho de banda (Mbps)
LiDAR 16	11.2
LiDAR 32	22.3
Cámara	191.9
Visualización	99.5
Otros	13.2
Total	349.3

Tabla 3.3: Resumen del ancho de banda de los sensores y módulos de software más relevantes.

3.4.4 *Cobertura de los sensores*

Los algoritmos de percepción deben ser capaz de detectar los obstáculos del entorno de tal forma que se pueda tomar una acción acorde. Para ello, los sensores deben cubrir todas las zonas en las que es necesario detectar obstáculos, siendo las zonas más críticas, la delantera y las laterales, ya que el movimiento futuro del vehículo puede discurrir por ellas.

Para definir la posición final de todos los sensores, se ha realizado un estudio en simulación donde, para distintas configuraciones de los sensores, se evalúa si estos son capaces de detectar un objeto con un tamaño concreto. Este objeto se desplaza de forma automática por todo el espacio alrededor del vehículo, generando un mapa en el que se indican las zonas en las que dicho objeto es detectado. La Figura 3.5 muestra un instante de esta simulación.

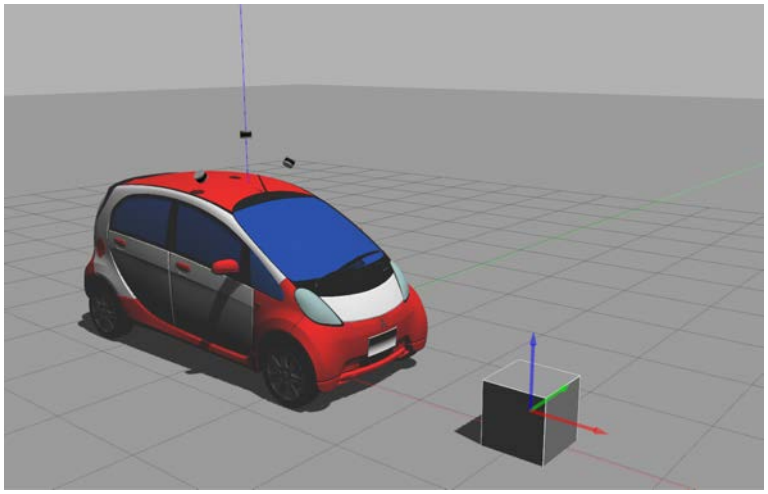


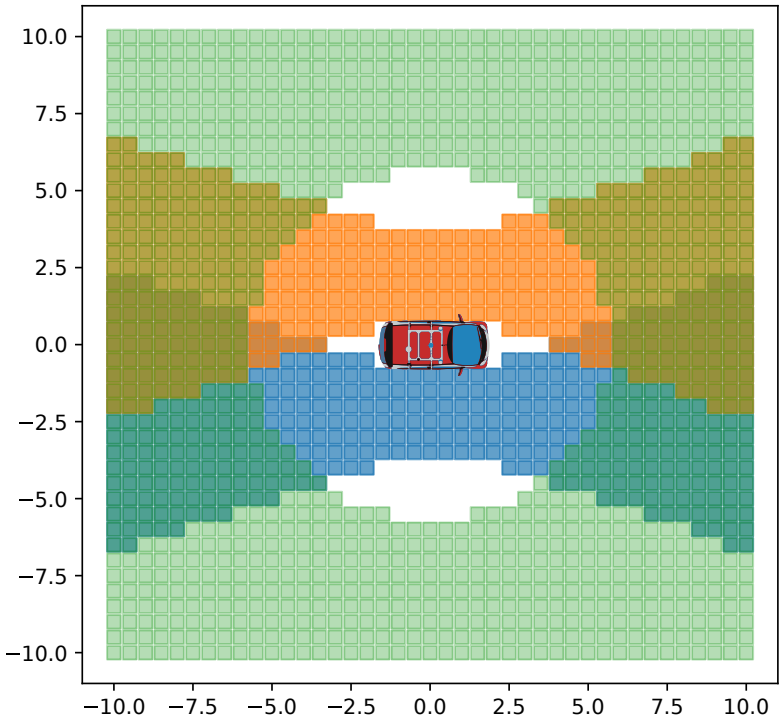
Figura 3.5: Captura de la simulación en la que se comprueba el rango de cada sensor en función de si detecta o no el bloque.

Este experimento se ha realizado para cada sensor en distintas configuraciones y con obstáculos de distintas alturas, lo que permite determinar qué tipo objetos se pueden detectar en cada zona alrededor del vehículo.

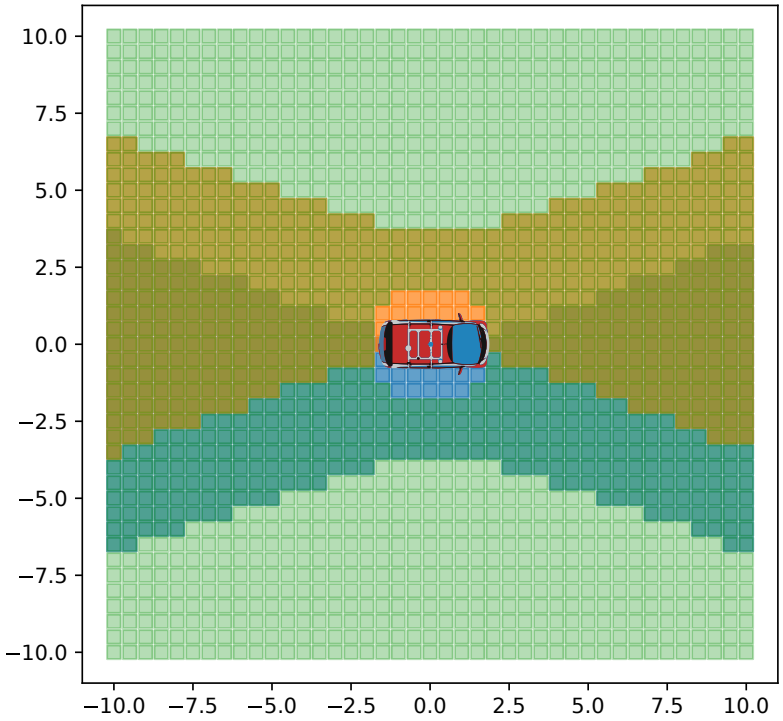
A continuación, se muestran los resultados obtenidos para las dos configuraciones más apropiadas de los sensores LiDAR laterales de entre las consideradas:

La primera configuración analizada, posiciona los sensores LiDAR en los laterales con una inclinación en el eje *roll* de 45º, con el objetivo de cubrir la zona de los laterales del vehículo, ya que con el sensor LiDAR central, existen zonas en las que no se pueden detectar obstáculos de 1.5m de alto, lo cual puede generar una amenaza para peatones que se encuentren en esta zona. Como se puede observar en la Figura 3.6, los LiDAR laterales cubren estas zonas laterales próximas al vehículo para obstáculos de 0.5m y 1.5m. Sin embargo, esta configuración deja tres áreas sin cubrir en la detección de obstáculos de 0.5m, como muestra la Figura 3.6a. La más comprometida se encuentra justo en el frontal del vehículo, donde ni LiDAR ni cámara podrían detectar obstáculos pequeños (de 0.5m de altura), como bolardos, cajas o incluso animales pequeños, poniendo en peligro la circulación.

Para solventar el problema de rango de la configuración anterior, se propone una modificación en la que los sensores LiDAR laterales, además de inclinarlos en el eje *roll*, también se rotan en el eje *yaw* para cubrir la zona delantera y las laterales. Como se puede observar en la Figura 3.7, esta configuración cubre adecuadamente estas zonas que no tenían visibilidad en la anterior configuración. A cambio, esta configuración deja sin cubrir la zona trasera y lateral trasera cercana vehículo, como se aprecia en la Figura 3.7a. Sin embargo, debido a que el vehículo siempre se va a desplazar hacia delante, estas zonas que no cubre esta configuración son menos relevantes y no tienen por qué comprometer la seguridad del VA. Por otra parte, esta configuración permite obtener más información de la parte delantera cercana al vehículo, ya que inciden un mayor número de planos, y por lo tanto se obtiene una nube de puntos más densa en esta parte más crítica.

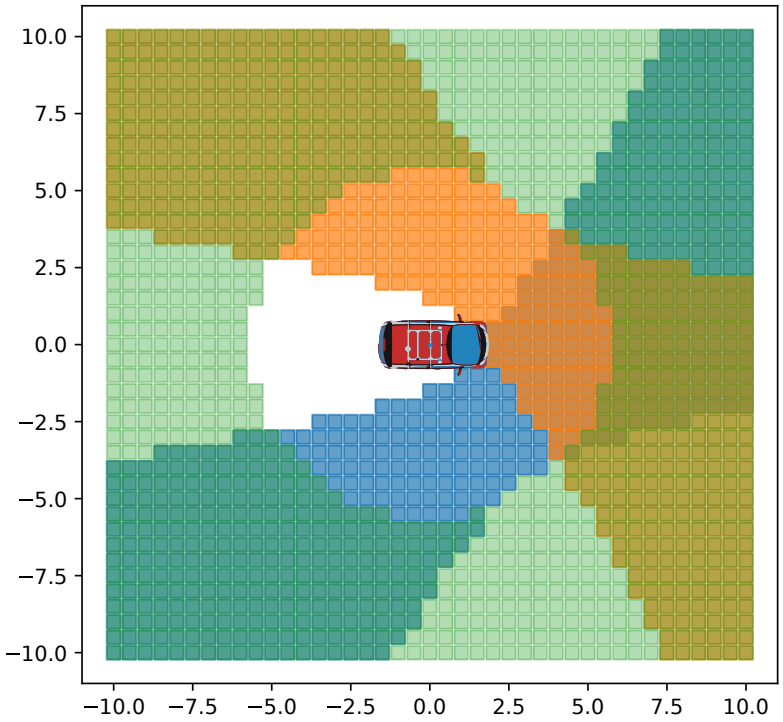


(a) Obstáculo de altura 0.5m

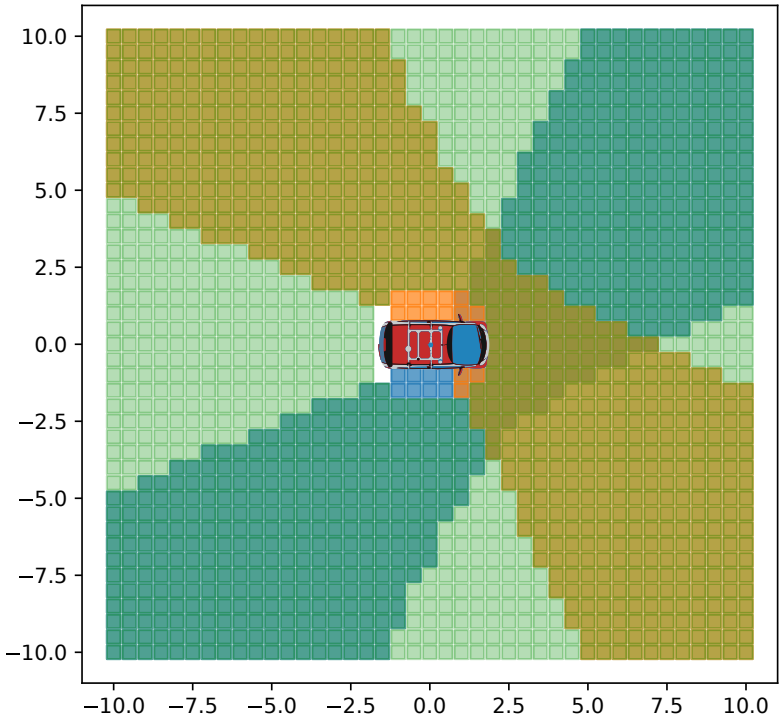


(b) Obstáculo de altura 1.5m

Figura 3.6: Área de detección de cada LiDAR para la configuración con los LiDAR laterales sólo inclinados en el eje *roll*.



(a) Obstáculo de altura 0.5m

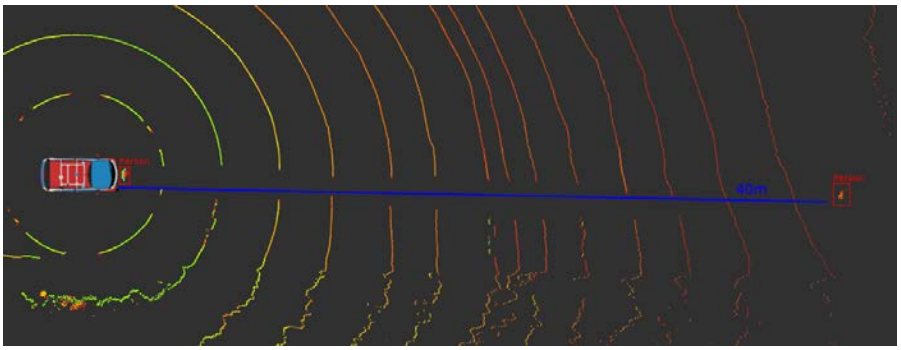


(b) Obstáculo de altura 1.5m

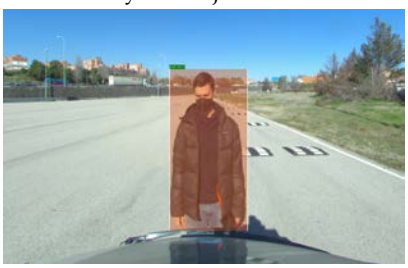
Figura 3.7: Área de detección de cada LiDAR para la configuración con los LiDAR laterales inclinados en el eje *roll* y *yaw*.

Para analizar la capacidad de la cámara de detectar obstáculos del entorno, y determinar el área en el que dichos obstáculos pueden ser detectados, se ha realizado un análisis con los datos de la cámara, la óptica y el algoritmo de percepción para determinar el rango en el que el vehículo es capaz de detectar y clasificar los obstáculos con la cámara.

Con la combinación de cámara y óptica elegidos, se han evaluado las limitaciones a la hora de detectar obstáculos, fijadas principalmente por el tamaño en píxeles del obstáculo. Para ello, se ha determinado de forma experimental el rango en el que el vehículo es capaz de detectar y clasificar correctamente el obstáculo más pequeño de las clases definidas (un peatón). La Figura 3.8 resume los resultados obtenidos donde un peatón se sitúa en los puntos más cercano (Figura 3.8b) y más lejano (Figura 3.8c) en los que es detectado correctamente. Estas distancias son medidas mediante la nube de puntos del LiDAR, obteniendo un rango de 1.7m para la distancia más cercana (corresponde al borde delantero del vehículo) y 40m para la distancia más lejana.



(a) Visualización del peatón en la nube de puntos del LiDAR en la posición más cercana y más lejana.



(b) Visualización del peatón en cámara en la posición más cercana.

(c) Visualización del peatón en cámara en la posición más lejana.

Figura 3.8: Visualización de la detección de peatones con la cámara y medición de su distancia máxima y mínima en la nube de puntos.

La capacidad de esta configuración de cámara y óptica para detectar obstáculos hasta una distancia de 40m, permite al vehículo detenerse en caso de emergencia con una deceleración de hasta $0.87 m/s^2$ (partiendo de la velocidad máxima del vehículo de 30km/h). Como

comparativa, la deceleración habitual en una situación de emergencia varía entre los 4 y $7 m/s^2$ [160], valores muy superiores.

3.4.5 *Control y funcionamiento*

Este apartado muestra los resultados del funcionamiento del vehículo, de tal forma que se pueda adquirir una visión global de su comportamiento y de cómo se adecúa a los objetivos definidos.

Para ello, se han realizado pruebas en un entorno real en el que se ha puesto a funcionar el vehículo, haciéndole circular de forma autónoma entre dos localizaciones en repetidas ocasiones. Durante el trayecto, se han grabado todos los mensajes de datos que genera cada módulo de la arquitectura para analizarlos posteriormente.

De los datos recogidos durante estas pruebas finales, se han analizado los siguientes cuatro aspectos principales.

3.4.5.1 *Señales de control con lineas*

Durante el modo de funcionamiento en el que el control del vehículo es generado a partir de las detecciones de las líneas de la carretera, se han extraído de las grabaciones los datos de localización y las señales de control enviadas al vehículo para analizarlas.

Los datos de localización, obtenidos del módulo GPS-AMCL, determinan la ruta que ha seguido el vehículo mientras seguía las líneas de la carretera. Sin embargo, estos datos no se pueden utilizar para obtener el error de seguimiento de trayectoria, ya que no existe ningún *groundtruth* que determine el centro exacto de la carretera y, aun teniendo esa información, la conducción real de un vehículo no siempre discurre por el centro exacto del carril, sino que con las curvas, cada conductor en cada momento las puede tomar de forma distinta (aproximándose más a la línea interior o exterior). Por lo tanto, la manera propuesta para evaluar el funcionamiento del control consiste en comparar cualitativamente la trayectoria seguida superpuesta con los límites de la carretera.

La figura 3.9 muestra, a partir de los datos de las detecciones, la distancia a lo largo del recorrido a ambos carriles, así como al centro de la carretera. Esta última distancia está acotada por un valor máximo de $78cm$ por lo que, teniendo en cuenta que la distancia media entre el vehículo y el borde de los carriles es de $100cm$, se puede asegurar que el vehículo en ningún momento se ha salido de la carretera.

Por último, cabe destacar, que estas pruebas se han realizado a la velocidad de 30km/h, establecida como la velocidad máxima para el control utilizando las líneas de la carretera

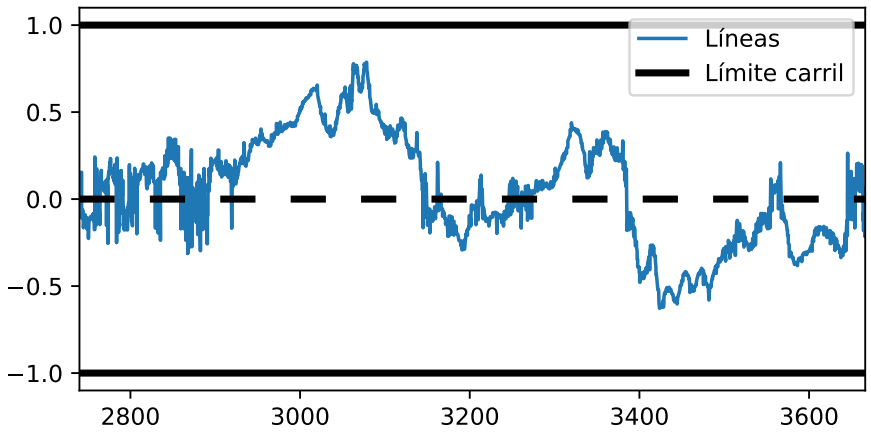


Figura 3.9: Evolución de la distancia a la trayectoria a seguir determinada por centro del carril a lo largo de la ruta.

3.4.5.2 *Señales de control con GPS-AMCL*

En este modo de funcionamiento, el control está basado en la localización generada por el módulo GPS-AMCL y en una ruta que el vehículo debe seguir, que ha sido grabada previamente con el propio vehículo, pero conducido de forma manual.

Para este modo de funcionamiento, se ha realizado una comparación entre la señal de control para el volante generada por el VA, y la aplicada durante la grabación de la ruta. La Figura 3.10, muestra un tramo de ambas señales de unos segundos (para apreciar mejor la similitud). Analizando todos los datos de las pruebas, se obtiene una diferencia media entre ambas señales de 2.5^{o} lo cual, teniendo en cuenta que el valor máximo del giro del volante es de 540^{o}, supone una señal de control muy parecida a la generada de forma manual

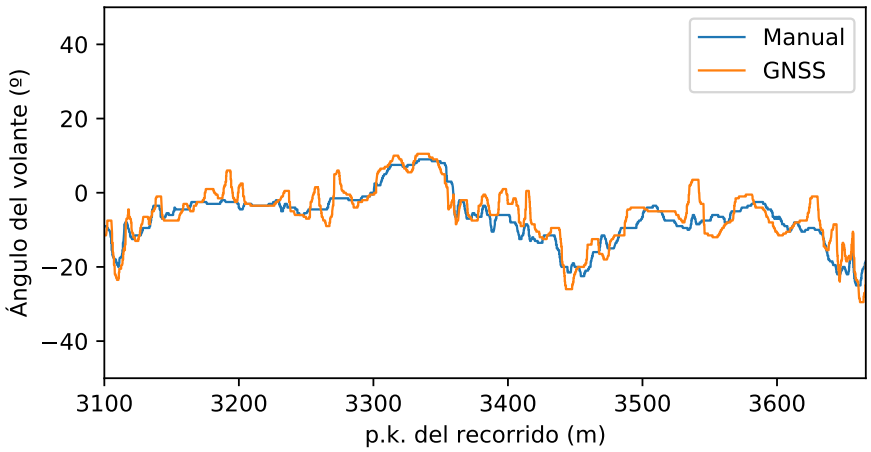


Figura 3.10: Señales de control del volante grabadas (verde) y generadas por el VA (rojo).

Por último, al igual que con el control basado en las líneas de la carretera, estas pruebas se han realizado a la velocidad de 30km/h, establecida como la velocidad máxima.

4

LOCALIZACIÓN

Cualquier robot móvil y, en especial, los vehículos autónomos, requieren conocer de forma precisa su localización para ser capaces de navegar hasta su destino. La localización de un vehículo está determinada por un conjunto de coordenadas espaciales de posición y orientación, es decir, para un espacio tridimensional, la localización de un vehículo está determinada por las coordenadas de posición x, y, z y las coordenadas de orientación en los tres ejes de giro *roll, pitch* y *yaw*. Sin embargo, para algunas aplicaciones como los vehículos terrestres que van a circular por carreteras relativamente planas, estas coordenadas se pueden reducir a las de posición y orientación en dos dimensiones x, y y *yaw*.

Todos los puntos del espacio relevantes como la localización del vehículo, la de destino o la trayectoria a seguir, deben estar referenciados con respecto a un eje de coordenadas común. Dicho origen de coordenadas puede estar situado en algún punto de un mapa digital, o puede corresponderse con una coordenada de latitud y longitud de la superficie de la Tierra.

Para obtener estas coordenadas de localización referenciadas a un punto en común (localización absoluta), existen distintos métodos, de los cuales, los más relevantes han sido descritos en la Sección 2.5.1. De entre los más utilizados, cabe destacar los sistemas de localización basados en GNSS y los que utilizan la correlación entre el mapa percibido y un mapa digital generado previamente. Cada uno de estos algoritmos funciona mejor un entorno con unas características específicas.

Motivado por los problemas de localización del vehículo encontrados en entornos mixtos, en los que parte del recorrido transcurre por carreteras urbanasy parte por amplias carreteras, surge la necesidad de desarrollar un algoritmo que los solvente.

El algoritmo desarrollado expuesto en este capítulo mejora la localización utilizando una versión modificada del método AMCL [88] en la que el peso de las partículas es modificado añadiendo información del sensor GNSS.

4.1 LOCALIZACIÓN BASADA EN GNSS

Los sistemas de localización basados principalmente en un sensor GNSS, son especialmente útiles en espacios al aire libre con pocos o

Este capítulo incluye contenido de [1].

ningún obstáculo de gran tamaño que oculte parte del cielo, o pueda introducir interferencias a las señales de los satélites.

Para un VA que navegue únicamente por carreteras amplias, lejos de los núcleos urbanos, tales como autovías o autopistas, este tipo de localización es el más adecuado.

Aunque el coste del sensor puede ser elevado, la simplicidad, robustez y precisión de este tipo de sistemas, hacen que sea el más utilizado.

Sin embargo, para obtener una alta precisión, es necesario procesar los datos en crudo de localización que nos devuelve el sensor.

En primer lugar, para mejorar la precisión de la localización, muchos receptores GNSS incorporan la posibilidad de recibir correcciones de distintos tipos (satélite, radio, internet, etc.), que pueden conseguir mejorar la precisión hasta valores de 1cm. En el caso del VA desarrollado en esta tesis, la solución adoptada para recibir estas correcciones es *Networked Transport of RTCM via Internet Protocol* (NTRIP) [161]. A través de NTRIP, se envían correcciones RTK [162] de una o varias estaciones base cercanas a la posición del vehículo, que reducen considerablemente el error de posición de este sensor. Las ventajas de este tipo de correcciones, y la motivación para su uso en este proyecto, radica en un coste nulo, o muy reducido en comparación con sistemas de corrección por satélite tipo *TerraStar* [163], con un radio de cobertura muy similar (el área con cobertura para conexión a internet en España es muy amplio). Además, la posibilidad de utilizar estaciones base de correcciones distribuidas por todo el territorio, evita el trabajo de instalar y calibrar una estación base propia que envíe correcciones por radio en cada zona en la que opera el vehículo.

A pesar de la gran precisión que se puede conseguir utilizando los sensores GNSS con las correcciones descritas en este apartado, las posiciones que devuelve el sensor contienen ruido de distintos tipos, que hace que la posición tenga fluctuaciones incluso con el vehículo completamente parado. Para reducir este ruido y obtener una localización más estable, es necesario filtrar las posiciones y fusionar información de distintas fuentes de odometría.

4.1.1 *Filtrado de la localización GNSS*

Uno de los métodos más utilizados para filtrar la localización GNSS es el Filtro de Kalman, el cual es capaz de fusionar distintas fuentes de localización para obtener como resultado una más precisa. Existen distintas variaciones del Filtro de Kalman original en la literatura, sin embargo, para este proyecto se ha utilizado el *Unscented Kalman Filter* (UKF), ya que es capaz de manejar no linealidades del modelo en el proceso de filtrado [164].

El software utilizado en este proyecto es una implementación en ROS del UKF [165], que es capaz de filtrar un número arbitrario de fuentes de odometría.

Además de posición, una orientación fiable es esencial para un correcto control del vehículo. El uso de una IMU, no sólo puede mejorar la fusión del filtro UKF incluyendo velocidades de rotación y aceleraciones lineales, sino que algunos modelos de IMU incluyen además un sensor magnético que puede proporcionar información de la orientación midiendo el campo magnético de la Tierra. Sin embargo, este tipo de sensores requieren de una correcta calibración para poder proporcionar medidas de orientación precisas. Dicha calibración, comúnmente se realiza girando el vehículo y el sensor en distintos ejes, ya que el propio vehículo puede afectar a la medida del campo magnético. La dificultad para realizar este proceso de calibración con un vehículo de varios cientos de kilos de peso hace que esta calibración no se pueda realizar con éxito y, por lo tanto, la orientación en el eje Z (Yaw) no sea precisa. Por otro lado, las orientaciones en los ejes X e Y (Roll y Pitch), si son fiables, ya que no dependen del campo magnético y su calibración se puede realizar sin el vehículo.

Debido a los problemas mencionados en la obtención del ángulo *Yaw* por el sensor IMU, y a la gran precisión obtenida con el sensor GNSS, la orientación es calculada utilizando las últimas posiciones recibidas por este sensor. El ángulo *Yaw* ψ es calculado como:

$$\psi_t = \arctan\left(\frac{\Delta Y}{\Delta X}\right) + \frac{d \cdot \dot{\psi_{t-1}}}{v_t} \tag{4.1}$$

donde ΔX y ΔY son los incrementos de las coordenadas X e Y en dos posiciones consecutivas separadas d metros, v_t es la velocidad actual, ψ_{t-1} y ψ_t, son el actual y pasado ángulo Yaw y $\dot{\psi_t}$ es la velocidad angular en Yaw. La covarianza $\sigma^2_{\psi_t}$, utilizada por el Filtro UKF se puede aproximar con la siguiente expresión:

$$\begin{aligned}\sigma^2_{\psi_t} = \frac{1}{(\Delta Y^2 + \Delta X^2)^2}\Big[\Delta X^2(\sigma^2_{t_{-1}Y} + \sigma^2_{tY}) + \Delta Y^2(\sigma^2_{t_{-1}X} + \sigma^2_{tX}) + \\ + \Delta Y \Delta X(\sigma_{t_{-1}XY} + \sigma_{t_{-1}XY})\Big] + \left(\frac{d}{v}\right)^2 \sigma^2_{\psi_{t-1}} + \left(\frac{d\psi_{t-1}}{v^2}\right)^2 \sigma^2_v + \\ + \frac{1}{\Delta Y^2 + \Delta X^2}\left(\frac{\psi_{t-1}}{v_t}\right)^2 \Big[\Delta Y^2(\sigma^2_{t_{-1}X} + \sigma^2_{tX}) + \\ + \Delta X^2(\sigma^2_{t_{-1}Y} + \sigma^2_{tY}) + \Delta X \Delta Y(\sigma_{t_{-1}XY} + \sigma_{tXY})\Big]\end{aligned} \tag{4.2}$$

donde σ^2_{tX}, σ^2_{tY} y σ^2_{tXY} son las actuales covarianzas de las posiciones X/Y, y $\sigma^2_{t_{-1}X}$, $\sigma^2_{t_{-1}Y}$ y $\sigma^2_{t_{-1}XY}$ las covarianzas de la posición anterior.

El método propuesto para calcular la orientación, una vez filtrado con el UKF, proporciona mejores resultados que el ángulo del sensor IMU, que tiende a tener una deriva no estable con respecto a la orientación real.

La última fuente de odometría incluida en el Filtro de Kalman es la odometría de las ruedas. Esta odometría está basada en la velocidad de las ruedas y el ángulo del volante en cada momento, obtenido a través del BUS CAN del vehículo. Estos datos son utilizados para estimar el desplazamiento del vehículo utilizando un modelo cinemático de bicicleta [107], donde los incrementos de las coordenadas X e Y ($\Delta X and \Delta Y$) y Yaw ($\Delta\psi$) son proporcionados por:

$$\Delta X = v \cdot \cos\left(\psi + \arctan\left(k_{cg} \cdot \tan\delta\right)\right) \cdot \Delta T \tag{4.3}$$

$$\Delta Y = v \cdot \sin\left(\psi + \arctan\left(k_{cg} \cdot \tan\delta\right)\right) \cdot \Delta T \tag{4.4}$$

$$\Delta\psi = v \cdot \cos\left(\arctan\left(k_{cg} \cdot \tan\delta\right)\right) \cdot \frac{\tan\delta}{w\_b} \cdot \Delta T \tag{4.5}$$

donde δ es el ángulo del volante, k_{cg} determina la posición del centro de gravedad (0 si está en el eje delantero y 1 si está en el trasero) y w_b es la distancia entre los ejes del vehículo. La odometría calculada acumula deriva con el tiempo, por lo que sólo los incrementos de posición y velocidad son considerados en el filtro. La incorporación de esta información al UKF da como resultado una localización más estable y con una orientación más precisa

4.2 LOCALIZATION BASADA EN LIDAR Y MAPA

Este apartado se centra en describir los detalles de la implementación del algoritmo AMCL para el vehículo desarrollado. Este algoritmo es adecuado para entornos con numerosos objetos estáticos alrededor, que proporcionen un mapa con múltiples características donde el LIDAR pueda realizar medidas para compararlas con el mapa y estimar su localización. A continuación, se describe el proceso para generar dicho mapa, así como la configuración de los parámetros del AMCL, adaptados para funcionar en el VA desarrollado.

4.2.1 *Generación del mapa georreferenciado*

La generación del mapa es un proceso con gran importancia, ya que la localización dependerá directamente de su calidad y precisión.

Para un robot de interior, o que navegue por un espacio reducido, el proceso de generación de este mapa no plantea una gran dificultad, y se puede obtener aplicando un algoritmo de SLAM como [166] o [167]. En este caso no es imprescindible que el mapa represente de forma precisa la realidad, es decir, el mapa puede sufrir transformaciones fruto de la deriva acumulada del algoritmo de SLAM, pero dichas transformaciones no afectarán a la precisión de la localización, ya que únicamente va a estar referenciadas con respecto al mapa generado.

El caso de un VA es totalmente distinto, ya que este cubre áreas mayores con carreteras largas, en las que sí es importante que no existan deformaciones, ya que las carreteras están interconectadas y una pequeña deriva en la generación del mapa, puede provocar que en las intersecciones las carreteras no coincidan entre ellas.

Esta característica del entorno del vehículo desarrollado hace que la generación del mapa sea distinta a los métodos habituales basados únicamente en SLAM. El método utilizado consta de tres fases:

- **Acumulación de puntos utilizando GNSS:** En primer lugar, se utiliza un receptor GNSS de alta precisión, así como el LiDAR del vehículo para acumular la nube de puntos y generar un mapa únicamente de las zonas en las que la localización es estable, es decir, cuando la covarianza del sensor GNSS es inferior a un umbral de aproximadamente 2m en X e Y, o en las zonas en las que visualmente la generación del mapa está dando buenos resultados. El resto de las zonas se dejan sin generar mapa, como zona desconocida.
- **SLAM:** Para estas zonas en las que la cobertura del receptor GNSS ha hecho imposible la generación del mapa, se generan pequeños mapas locales de estas zonas por separado con cualquier algoritmo de SLAM.
- **Fusión de mapas:** El último proceso consiste en fusionar todos los mapas, colocando los pequeños mapas generados por SLAM en el mapa global georreferenciado, completando las zonas sin mapa. Al ser los mapas una imagen en 2 dimensiones, este proceso se puede realizar de forma manual con cualquier editor de imágenes disponible.

La principal ventaja de este método radica en que cada punto del mapa está georreferenciado y es posible conocer su ubicación en coordenadas globales

4.2.2 *Configuración de los parámetros del AMCL*

Para conseguir un correcto funcionamiento del algoritmo AMCL, es necesario ajustar sus distintos parámetros de tal forma que el algoritmo se adapte a cada aplicación concreta en la que se va a utilizar, en este caso un VA, y obtenga los mejores resultados posibles de localización.

A continuación, se describen los parámetros más relevantes, indicando y justificando el valor elegido:

- *global_frame_id*, *odom_frame_id* y *base_frame_id*: Estos parámetros indican el nombre de los sistemas de coordenadas (*frames*) que se utilizarán. Habitualmente en los proyectos de ROS existen tres

frames principales: el global *map*, el del vehículo o robot *base_link* y el origen de la odometría *odom*. La transformada entre el vehículo y el origen de la odometría viene determinado por el algoritmo de odometría utilizado basado en el movimiento de las ruedas y el volante (Sección 2.5.1.3). Como este algoritmo suele tener un error acumulado, es necesario desplazar gradualmente el *frame odom* para que la posición del vehículo con respecto al sistema de referencia global *map* sea correcto. Esta transformación la proporciona el algoritmo AMCL. Los valores de estos parámetros se eligen siguiendo la convención de nombres de *frames* más utilzada de ROS: *map*, *odom* y *base_link*.

- *odom_model_type*: Este parámetro indica el tipo de movimiento del vehículo y que influirá en la manera de muestrear las posiciones. En la implementación utilizada de AMCL solo existen dos posibles valores: *diff*, que corresponde al modelo definido en [168] de un robot con geometría diferencial, y *omni*, que describe el movimiento de un robot omnidireccional. Aunque ningún modelo se adapta completamente al movimiento de un automóvil, el valor escogido es *diff* ya que restringe más los movimientos que puede realizar el vehículo y se asemeja más al utilzado en esta tesis.

- *laser_max_beams* y *laser_max_range*: Representan respectivamente el número máximo de haces láser en el plano que se considerarán en cada iteración, y la distancia máxima a partir de la cual, los puntos detectados por el sensor no serán tomados en cuenta. Esta última variable debe ser elegida en función de la distancia máxima a la que se espera encontrar objetos relevantes del mapa. Para el número máximo de haces láser, cuanto más alto sea este valor, mayor será la precisión, pero también la carga computacional. Su valor debe ser elegido en consonancia con el rango máximo elegido, de tal forma que se disponga de suficiente resolución angular y la carga computacional no sea excesiva. Para el caso estudiado en este documento, los valores elegidos, de forma experimental, son de 360 para *laser_max_beams* y 60m para *laser_max_range*.

- *min_particles* y *max_particles*: Estos parámetros determinan el rango del número de partículas que existirán en el filtro. Valores elevados consiguen una mejor representación de las distintas funciones de probabilidad asociadas a las posibles localizaciones, aumentando por otro lado la carga computacional. Los valores finales deben buscar un equilibrio entre un número que represente las distribuciones con suficiente precisión y una carga computacional no demasiado alta. Los valores han sido establecidos de forma experimental en 500 para *min_particles* y 2000 para *max_particles*.

- *laser_model_type*: Se pueden elegir dos tipos de modelos para el sensor LiDAR, *beam* y *likelihood field*. El modelo *beam* define una función de probabilidad que depende de la distancia medida y considera distintos tipos de error. Sin embargo, este modelo presenta varios inconvenientes. En primer lugar, para entornos con numerosos objetos pequeños, la distribución puede no ser suficientemente suave. Además, evaluar este modelo para cada medida puede resultar muy costoso computacionalmente. Como solución se presenta el modelo *likelihood field*, que soluciona estos problemas utilizando un mapa con probabilidades establecidas para cada celda [168]. Por estos motivos, el modelo utilizado en la configuración es *likelihood field*.

En la Tabla 4.1, se muestra un resumen con la configuración de todos los parámetros del algoritmo AMCL, cuyos valores, escogidos principalmente de forma experimental, también han sido utilizados en las pruebas del algoritmo propuesto en este capítulo.

4.3 COMBINACIÓN DE LIDAR Y GNSS

Los dos métodos de localización descritos en los apartados anteriores pueden llegar a alcanzar una gran precisión en condiciones ideales. Sin embargo, cada método de localización tiene un tipo de escenario en el que no funciona correctamente o su precisión disminuye considerablemente. Estos escenarios problemáticos de ambos tipos de localización son complementarios, es decir, el escenario ideal para la localización utilizando GNSS, un espacio abierto, es el peor para el otro método (AMCL) y viceversa, un lugar cubierto con múltiples obstáculos favorecería este último tipo de localización, pero impediría el uso de un sensor GNSS.

Este apartado describe el algoritmo desarrollado para combinar ambos métodos, consiguiendo una localización precisa en cualquier tipo de escenario. Dicho algoritmo consiste en una serie de modificaciones al algoritmo original AMCL que integran la información GNSS en el bucle. Los apartados anteriores describen cómo se han configurado cada uno de los métodos, por lo que este apartado se centrará únicamente en describir el algoritmo desarrollado, ya que la configuración de ambos métodos de localización será la misma que ya se ha descrito. El nuevo algoritmo se puede observar en el Algoritmo 1.

4.3.1 *Probabilidad del LiDAR*

La solución propuesta calcula un peso para cada partícula de AMCL comparando las lecturas del LiDAR transformadas al eje de coordenadas de cada partícula, con el mapa generado. Además de este peso, se calcula una puntuación que expresa cómo de precisas son

Parámetro	**Valor**
global_frame_id	map
base_frame_id	base_link
odom_frame_id	odom
odom_model_type	diff
odom_alpha5	0.1
gui_publish_rate	1.0
laser_max_beams	360
laser_max_range	60.0
min_particles	500
max_particles	2000
kld_err	0.05
kld_z	0.99
odom_alpha1	0.2
odom_alpha2	0.2
odom_alpha3	0.2
odom_alpha4	0.2
laser_z_hit	1.0
laser_z_short	0.05
laser_z_max	0.05
laser_z_rand	0.0
laser_sigma_hit	0.2
laser_lambda_short	0.1
laser_model_type	likelihood_field
laser_likelihood_max_dist	60.0
update_min_d	0.25
update_min_a	0.05
resample_interval	1
transform_tolerance	1.0
recovery_alpha_slow	0.001
recovery_alpha_fast	0.1

Tabla 4.1: Valores utilizados de todos los parámetros de AMCL.

estas lecturas del láser desde la partícula con respecto al mapa. Esta puntuación s_i es calculado utilizando un modelo Gausiano [168, 169] para el LIDAR, siguiendo la expresión:

$$s_i = \frac{1}{N} \cdot \sum_{n=1}^{N} \frac{1}{\sigma_{hit} \cdot \sqrt{2 \cdot \pi}} \cdot e^{\frac{-z_n^2}{2 * \sigma_{hit}^2}} \tag{4.6}$$

donde N es el número de lecturas láser, z es la distancia de la lectura del láser al obstáculo más cercano del mapa y σ_{hit} es la desviación estándar de las lecturas del LIDAR obtenida de su hoja de características. Con esta expresión, se puede evaluar cómo de buenas son las medidas del sensor láser en cada partícula con respecto al mapa, información que se usará para modificar la probabilidad de existir de cada partícula.

4.3.2 *Estimación de probabilidad del sensor GNSS*

Además del peso calculado utilizando las lecturas del LIDAR, se añade un segundo peso basado en la posición estimada con el sensor GNSS. Tal y como en el caso anterior, se utiliza un modelo Gaussiano para estimar este nuevo peso de cada partícula d_i, pero en este caso, debido a que la posición de cada partícula es tridimensional (x, y y ψ para la orientación), se utiliza el modelo n dimensional,

$$d_i = \frac{1}{(2 \cdot \pi)^{\frac{3}{2}} \cdot |\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x_i - \mu_k)^T \Sigma_k^{-1} (x_i - \mu_k)\right) \tag{4.7}$$

donde la posición recibida es $\mu_k = (x_k, y_k, \psi_k)$ con matriz de covarianza Σ_k y la posición y orientación de cada partícula está definida con $x_i = (x_i, y_i, \psi_i)$.

Utilizando este modelo, el error de orientación es también considerado a la hora de calcular el peso basado en el sensor GNSS. Por otra parte, la multiplicación en la parte exponencial por la inversa de la matriz de covarianza hace que los errores de posición y orientación sean equivalentes en magnitud. Es importante señalar que el uso de la covarianza de la localización utilizando GNSS, reduce el valor del peso cuando la precisión de la posición es baja.

4.3.3 *Cálculo del peso de cada partícula*

Para modificar el peso de cada partícula, incorporando la información del sensor GNSS al filtro, el nuevo peso de cada partícula (w_{i-new}) es calculado utilizando (4.6), (4.7) y la siguiente ecuación:

$$w_{i-new} = s_i \cdot k_l + d_i \tag{4.8}$$

donde el peso calculado utilizando el algoritmo original de AMCL es modificado para incorporar la información obtenida con el sensor

GNSS. La constante k_i es añadida para balancear la importancia de cada fuente de información. En los experimentos realizados ha sido ajustada empíricamente a un valor de 200, donde se han encontrado los mejores resultados en todos los entornos en los que se ha probado. Después de esta modificación en los pesos, es necesario normalizarlos de tal forma que la suma de todos ellos sea igual a 1.

4.3.4 *Generación de nuevas partículas*

A medida que el método original AMCL elimina partículas de acuerdo a su probabilidad, se añaden nuevas partículas aleatoriamente distribuidas en el mapa de acuerdo a dos parámetros que especifican cómo de frecuente es necesario añadir estas nuevas partículas. Para el algoritmo propuesto, ha sido definida una nueva función que genera nuevas partículas X_{new} cercanas a la posición estimada con GNSS y determina la probabilidad de generación de estas nuevas partículas. La siguiente expresión define cómo las nuevas partículas son generadas para seguir una distribución normal centrada en μ_k con covarianza Σ_k.

$$\begin{pmatrix} x_{new} & y_{new} & \psi_{new} \end{pmatrix}^T = \lambda_k^{\frac{1}{2}} \phi_k \cdot R + \mu_k \tag{4.9}$$

donde R es un vector aleatorio con distribución $N[0,1]$ y λ, ϕ son la matriz diagonal de valor propio y vector propio de la matriz de covarianza Σ_k respectivamente. Además, la probabilidad p de generar una nueva partícula basada en la información del sensro GNSS en cada ciclo es determinada como:

$$p(X_{new}|x,\mu) = \begin{cases} d_{mean}, & \text{si } d_{mean} > 0 \\ 0, & \text{si no es así} \end{cases} \tag{4.10}$$

con

$$d_{mean} = p_{max} - \frac{1}{N} \cdot \sum_{n=1}^{N} d_i \tag{4.11}$$

donde p_{max} es la máxima probabilidad permitida de generar nuevas partículas en cada ciclo del filtro y es ajustada experimentalmente con un valor de 0.01.

La incorporación de estas nuevas partículas no aumenta el ruido en la localización final del filtro de partículas, ya que solamente son añadidas cuando la posición y orientación de las partículas difieren considerablemente de la localización basada en GNSS, y el número de estas nuevas partículas nunca supera un 1% del total.

Algoritmo 1: Nuevos pesos y generación de partículas del filtro.

Entrada: x, μ_k, w, z
Salida: x_{nuevo}
for $i = 0$ **to** N **do**
  $s_i = probabilidadLidar(z_i)$;
  $d_i = probabilidadGNSS(x_i, \mu_{ki})$;
  $w_{i-nuevo} = w_i \cdot s_i \cdot k_l + d_i$;
end
$w_{nuevo} = normalizarPesos(w_{nuevo})$;
for $i = 0$ **to** N **do**
  $p(X_{nuevo}|x, \mu) =$
   $probabilidadNuevaParticula(d_i)$;
  if $rand() < p(X_{nuevo}|x, \mu)$ **then**
    $x_{i-nuevo} =$
     $generarNuevaPoseParticula(\mu_k)$;
  else
    $x_{i-nuevo} =$
     $muestrearPartcula(x, w_{nuevo})$;
     `// Función original de AMCL`
  end
end

4.4 RESULTADOS

En esta sección se presentan los resultados de los métodos discutidos en este capítulo, evaluados en distintos escenarios, utilizando los datos de las secuencias de odometría de KITTI para poder obtener resultados cuantitativos.

4.4.1 *Base de datos y metodología*

La base de datos de odometría de KITTI (descrita en la Sección 2.3.1), es utilizada habitualmente para probar distintos algoritmos de localización, evaluarlos y compararlos entre ellos, ya que incluye un *ground truth* de la localización.

Los experimentos realizados utilizan esta base de datos para comparar el *ground truth* con los siguientes métodos:

- Filtro de Kalman de los sensores GNSS e IMU. Debido a que la covarianza de este método es fija, en las figuras se representa el valor medio para una mejor visualización.
- Implementación original de AMCL.

- Método descrito en esta sección, con la misma fuente de odometría y la misma configuración de parámetros que la implementación original de AMCL.

Para cada posición y orientación, se calculan la distancia euclídea y error de orientación con respecto a las más próximas en el tiempo del *ground truth* (interpolando si fuera necesario). Las secuencias de KITTI proporcionan datos de distintos escenarios. Para estos experimentos, se han seleccionado las secuencias en entornos residenciales, ya que combinan calles estrechas con carreteras amplias.

Para cada secuencia, en primer lugar, se genera un mapa utilizando la localización del *ground truth* y el LIDAR tal y como se ha descrito en la Sección 4.2.1. Los experimentos están diseñados para comparar el método propuesto con los otros dos considerando los peores escenarios para el primero donde al menos, debe conseguir un resultado similar al mejor de los otros dos métodos en ese escenario. Por último, se evalúan los tres métodos en una situación mixta más real en la que coexistan los dos tipos de escenario en una misma secuencia para cuantificar la mejora del método propuesto.

Las tres siguientes secciones describen los resultados en cada tipo de escenario probado.

4.4.2 *Mapa vacío*

Este escenario simula las situaciones con pocos obstáculos y por lo tanto falta de referencias con las que correlacionar las lecturas del LIDAR. Este escenario es el más adverso para el algoritmo AMCL. Sin embargo, el sistema GNSS consigue tener mayor precisión cuando no está rodeado por obstáculos que interfieran con las observaciones de los satélites. En este apartado se comparan los errores de la localización GNSS filtrada, con el método propuesto. Tal y como se muestra en la Figura 4.1, ambos errores de localización son muy similares, obteniendo valores parecidos tanto para el error de posición como para el de orientación. Este resultado es el esperado ya que, aunque el método propuesto no puede mejorar la localización sin un mapa en el que casar las lecturas del LIDAR, prueba que en un entorno vacío funcionará de forma similar a la localización utilizando GNSS.

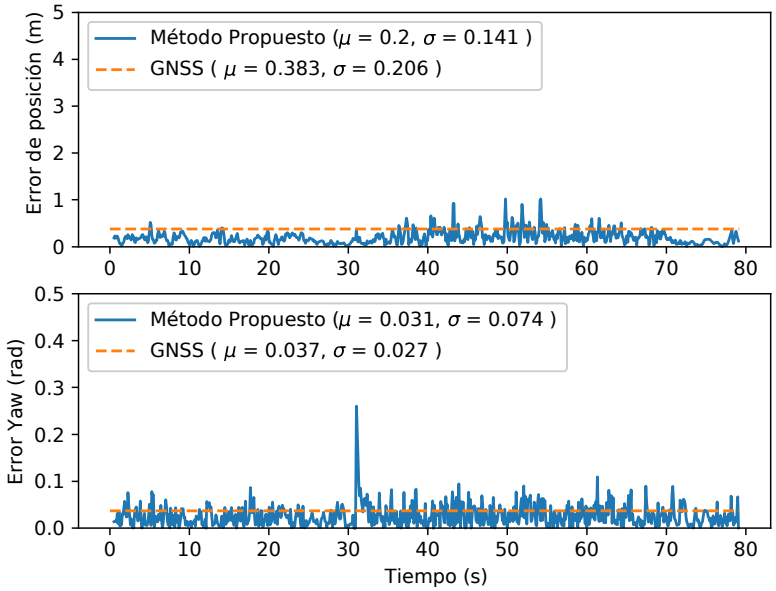


Figura 4.1: Comparación entre el método propuesto sin mapa con la localización con GNSS filtrada. (GNSS está representado como constante para una mejor visualización).

4.4.3 *Escenarios con dificultades para la localización con GNSS*

Este tipo de escenarios representan el problema opuesto. En un entorno lleno de objetos o entornos de ciudad con cañones urbanos, la localización con GNSS no funcionará correctamente dando medidas poco precisas con un valor de covarianza elevado. En este caso, el algoritmo AMCL proporciona mejores resultados debido a que el mapa tendrá numerosos puntos característicos con los que correlacionar las lecturas del LiDAR. En este apartado se comparan los errores de localización del método propuesto con los del AMCL. Como se puede comprobar en la Tabla 4.2, la cual muestra los resultados de Desviación Estándar (DE) y Error Absoluto Medio (EAM), los errores de localización son similares. Cuando no se recibe localización de GNSS o la covarianza es demasiado elevada, el método propuesto tiene un comportamiento similar a la implementación original de AMCL.

Método	EAM Posición (m)	DE Posición (m)	EAM Yaw (rad.)	DE Yaw (rad.)
Propuesto	0.513	0.856	0.033	0.025
AMCL	0.566	0.742	0.023	0.023

Tabla 4.2: Comparación entre el método propuesto y AMCL en ausencia de GNSS.

4.4.4 *Escenarios mixtos*

El último escenario probado es uno intermedio a los dos escenarios planteados en los apartados anteriores y, por lo tanto, más similar a los lugares por los que suele navegar un VA. En este escenario se recibe señal GNSS con algo de ruido ya que existe algún obstáculo que puede influir en la recepción, y el mapa contiene obstáculos y puntos característicos, aunque no están repartidos por el entorno de forma uniforme (existen zonas con más objetos y zonas más vacías).

Los tres algoritmos son comparados en este escenario utilizando las secuencias de odometría de KITTI, que incluyen más de 45 minutos de datos en un entorno residencial, con las características descritas.

4.4.4.1 *AMCL*

Cuando se evalúa el funcionamiento de AMCL, se pueden identificar dos problemas de localización. El primero, tal y como se puede observar en la Figura 4.2, presenta un incremento en el error debido a la existencia de múltiples núcleos de partículas cercanos que el filtro identifica como los mejores, haciendo que la localización salte entre ellos aumentando el error de localización hasta que el filtro vuelve a converger a la posición real. El segundo problema puede observarse en la Figura 4.3, donde aparece el problema del secuestro del robot alrededor del segundo 180, momento en el que la localización salta a un lugar similar del que no se vuelve a la posición real.

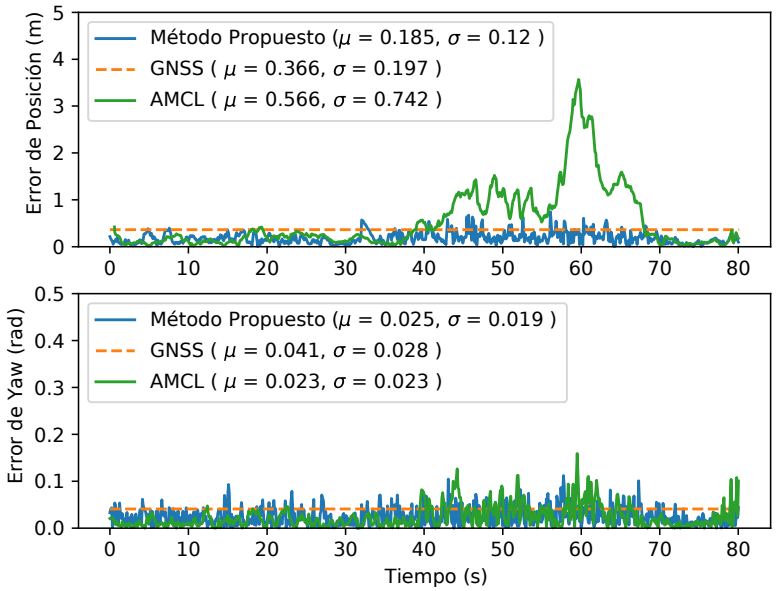


Figura 4.2: Error de localización de los tres métodos descritos utilizando la secuencia 36 de KITTI. (El error del método GNSS se representa como una línea constante del valor de la media para una mejor visualización.

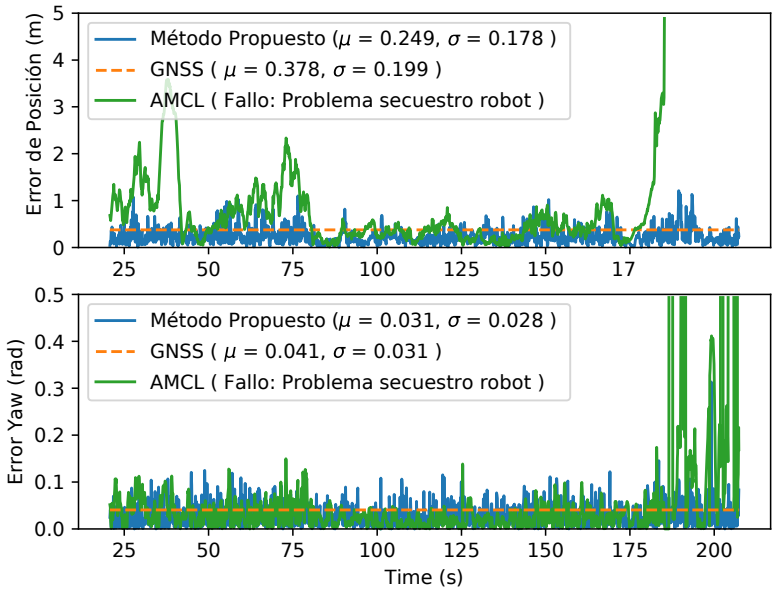


Figura 4.3: Error de localización de los tres métodos descritos utilizando la secuencia 34 de KITTI. (El error del método GNSS se representa como una línea constante del valor de la media para una mejor visualización.

4.4.4.2 *GNSS*

Se simula a partir del *groundtruth* de localización al que se le añade ruido Gaussiano con una covarianza configurable, el cual se puede asumir como un modelo de ruido razonablemente realista para receptores con un ancho de banda estrecho [170]. Por esta razón, el error medio de localización de este método en cada experimento se mantiene constante.

4.4.4.3 *Método propuesto*

El método propuesto obtiene un mejor resultado en toda la secuencia, mejorando el error medio de localización comparado con AMCL y GNSS. Mejora los otros dos métodos en términos de precisión y estabilidad, ya que tiene un error de localización menor o igual al de GNSS, y además no presenta el problema del secuestro del robot.

Las Figuras 4.2 y 4.3 presentan el error de localización para dos de las secuencias probadas (las secuencias 36 y 34), mostrando cómo el método propuesto mejora el algoritmo original AMCL evitando el problema del secuestro del robot mientras mantiene el error siempre por debajo del de GNSS.

EAM GNSS (m)	**EAM Posición (m)**	**DE Posición(m)**	**EAM Yaw**	**DE Yaw**
0.127	0.141	0.137	0.015	0.016
0.374	0.186	0.11	0.028	0.024
1.256	0.367	1.767	0.029	0.129
6.256	0.496	1.963	0.023	0.116
12.600	0.554	2.115	0.026	0.151
37.581	0.593	2.495	0.032	0.196

Tabla 4.3: Evaluación de los errores del método propuesto para distintos grados de precisión de GNSS en la secuencia 34.

Por otra parte, la Tabla 4.3 compara el error del método propuesto con el sensor GNSS al que se le ha añadido distintas magnitudes de ruido, simulando situaciones en las que este sensor es más y menos preciso. Como se puede observar, para una precisión alta de GNSS el error de ambos métodos es parecido, pero a medida que esta precisión disminuye, el método propuesto es capaz de reducir este error utilizando la correlación entre el sensor LiDAR y el mapa, demostrando un mejor funcionamiento para todas las situaciones posibles comparado con los otros dos métodos.

Por último, el código fuente de este trabajo se encuentra publicado y se puede encontrar en el siguiente enlace para que aquellos interesados

en este método puedan utilizarlo y replicar los resultados https://github.com/midemig/gps_amcl.

5

PREDICCIÓN DE TRAYECTORIAS

En este capítulo se describe un análisis de la implementación y funcionamiento de un modelo de aprendizaje automático basado en capas recurrentes y un modelo de tipo VAE para la predicción de la maniobra y trayectoria que va a seguir un vehículo en los próximos segundos.

La Sección 2.5.3 revisa los distintos métodos que se pueden encontrar en la literatura para realizar esta tarea, dando mayor importancia a los últimos avances utilizando técnicas de aprendizaje automático. La principal conclusión que se puede extraer de esta revisión del estado del arte es que la tendencia actual a la hora de resolver este tipo de problemas es el uso de modelos de aprendizaje automático, habiendo ganado gran importancia en los últimos años los modelos generativos basados en estructuras de tipo GAN o VAE.

Existen distintos trabajos que utilizan una arquitectura basada en un modelo VAE para predecir trayectorias, sin embargo, difieren en distintos aspectos con respecto al modelo descrito en este capítulo

Algunos de estos trabajos (ya descritos en la Sección 2.5.3) no están orientados a la generación de trayectorias en tiempo real, sino que se centran en generar trayectorias realistas para crear una base de datos. Es el caso de MTG [125] o de TrajVAE [124], que además genera trayectorias globales, con características distintas a las estudiadas en este capítulo. Por otra parte, el trabajo presentado en [127], sí está orientado a predecir trayectorias locales en tráfico real utilizando un modelo cVAE (una variación del VAE original), sin embargo, está basado en un modelo que predice el tipo de maniobra futura. Este trabajo por lo tanto necesita añadir a la base de datos una variable que, para cada trayectoria, indique a qué tipo de maniobra corresponde, lo cual no es trivial al tratar de clasificar un concepto abstracto y con distintas interpretaciones ya que no está bien definido cuándo empieza un cambio de carril, ni que todos los cambios de carril sean iguales (es distinto un cambio lento, que uno rápido por una situación de emergencia).

En los siguientes apartados se describirá el procesamiento y la obtención de los datos utilizados para entrenar y evaluar el modelo, seguido de una descripción del modelo y un análisis que muestra la utilidad de los modelos VAE aplicados a este tipo de problema. Por último, se analizan los resultados obtenidos con el modelo propuesto.

5.1 PREPROCESAMIENTO DE DATOS

Para entrenar el modelo, es necesario obtener datos reales de vehículos circulando por una carretera que incluyan variables como su posición y velocidad en todo momento. La Sección 2.3 analiza las distintas bases de datos de este tipo que existen y, como se indica, aunque la base de datos más utilizada en la mayoría de los trabajos es NGSIM [67], los problemas detectados en sus datos, y la aparición de nuevas bases de datos más precisas y con una mayor cantidad de secuencias, hacen más apropiadas bases de datos como highD [11], que es la que se utilizará para entrenar y analizar el modelo propuesto.

5.1.1 *Preprocesamiento y normalización*

La base de datos highD incluye información dinámica como posición, velocidad y aceleración para cada vehículo en la carretera en cada instante de tiempo. Combinando distintos campos y aplicando un filtro paso bajo, se pueden extraer, para cada trayectoria, todos los vehículos que se encuentran alrededor, definiendo ocho zonas que pueden estar ocupadas por otro vehículo. Como se puede observar en la Figura 5.1, en cada zona se considera el vehículo más cercano y se genera una métrica m_i que representa su distancia. La expresión de esta métrica para todas las zonas excepto la 2 y la 7 viene dada por:

$$m_i = \begin{cases} \sqrt[3]{d_{max} - d_i}, & \text{si } d_i < d_{max} \\ 0, & \text{si no} \end{cases}, \tag{1}$$

donde d_{max} es el valor de la distancia máxima considerada (determinada en $200m$ de forma experimental). La distancia real d_i se transforma a una métrica m_i que representa de forma más coherente la ausencia de vehículo en esa zona y los valores de las distancias tengan una distribución más uniforme.

Para las zonas 2 y 7, el valor de m_i se fija en 1 si el vehículo o parte de él está en la zona y 0 en el resto de los casos.

El resto de los valores son normalizados para ajustarse a una distribución normal, con una particularidad para las coordenadas de posición. Para la longitudinal, se les resta a todos los valores de cada trayectoria su primer valor y se normalizan de tal manera que el valor máximo de todas las secuencias sea 1. Para la coordenada lateral, en cada carretera se transforma a un sistema común en el que los centros de los carriles se encuentran en los valores 0.25, 0.75 y 1.25.

5.1.2 *Clasificación de las maniobras*

Para realizar el análisis del modelo en la Sección 5.2.4 y analizar la utilidad de la arquitectura basada en VAE, es necesario añadir

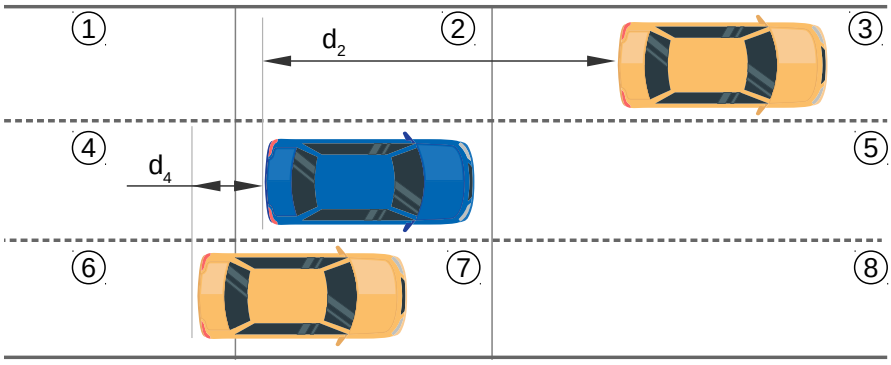


Figura 5.1: Representación de las zonas alrededor del vehículo analizado (azul). Como ejemplo de cómo se obtiene d_i, se han colocado dos vehículos alrededor (amarillo).

una nueva variable a la base de datos. Esta variable clasifica cada trayectoria en una de las siguientes clases: mantener carril, cambio al carril izquierdo y cambio al carril derecho. La pertenencia a alguna de estas clases se determina comprobando si en algún momento de la trayectoria, al menos una parte de ella forma parte de un cambio de carril o no. Esta variable generada, no se podrá utilizar para el entrenamiento, ya que incluye información del futuro (en una situación real, no sabes si la trayectoria de un vehículo cambiará o no de carril), por lo que sólo será utilizada con propósitos de análisis.

5.1.3 *División de la base de datos*

La base de datos consiste en 60 secuencias grabadas en distintas carreteras. Para evaluar de forma correcta los resultados del modelo propuesto, la base de datos se divide en 45 secuencias con datos para entrenar y 15 para evaluar (75-25%). De esta forma, los resultados se evalúan con datos de secuencias que nunca se han visto en el proceso de entrenamiento. Además, de la parte de entrenamiento (45 secuencias), el 30% se utiliza para validación y ajuste de los hiperparámetros de la red.

5.2 ARQUITECTURA DEL MODELO

Esta sección describe la arquitectura del modelo propuesto: En primer lugar, se describe el uso del modelo VAE, a continuación, el modelo utilizado para predecir las trayectorias futuras, así como una descripción de cómo se entrena el modelo completo. Por último, se analiza el espacio latente generado por el codificador y la precisión del modelo VAE.

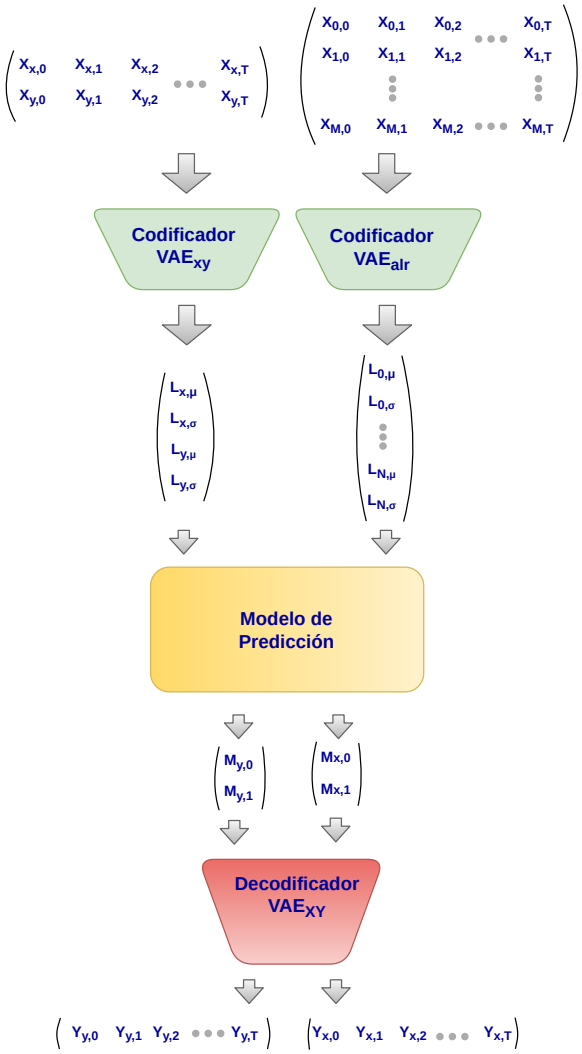


Figura 5.2: Arquitectura completa del modelo completo.

5.2.1 *Modelo VAE*

Un modelo VAE, definido por primera vez en [171], es muy similar en cuanto a la forma, a un *Auto Encoder* (AE) estándar, el cual utiliza un decodificador para reconstruir los datos de entrada, que han sido previamente codificados en una variable con un número menor de dimensiones que estos datos de entrada (variable latente). En el caso del VAE, el codificador en lugar de mapear los datos de entrada en un único punto multidimensional, genera una distribución normal sobre este punto, definida por su media y varianza. Para ello, la función de coste incluye un término adicional para forzar que el espacio latente sea similar a este tipo de distribución [126].

La naturaleza temporal de este problema hace que el uso de las neuronas de tipo recurrente sea el más adecuado para extraer características temporales, así como para generar series de datos en el tiempo.

La arquitectura propuesta para el modelo VAE utiliza capas de tipo LSTM, ampliamente utilizadas para una gran variedad de problemas complejos de aprendizaje automático en los que hay involucradas secuencias temporales [172]. Estas capas son combinadas con otras de tipo Densa o *fully-Connected* y Densa distribuida en el tiempo o *time-distributed fully-Connected*.

La arquitectura propuesta utiliza dos modelos VAE, que son entrenados de forma independiente. El primer modelo es entrenado con M características, correspondientes a los datos de todos los vehículos de los alrededores (distancias y velocidades) para los T instantes de tiempo previos. En el modelo final ensamblado, sólo se utilizará la parte del codificador de VAE_{alr} para generar variables latentes, aunque el modelo completo se analizará en la Sección 5.2.4 para probar que todas las características relevantes están representadas en el espacio latente. El segundo modelo (VAE_{xy}) sólo se entrena con posiciones longitudinales y laterales de trayectorias con una duración de T instantes de tiempo. El decodificador de este modelo se utilizará para generar trayectorias con apariencia real, mientras que la parte del codificador generará variables latentes de la trayectoria previa del vehículo, que serán combinadas con las generadas por el codificador del modelo VAE_{alr}.

La función de coste utilizada combina el coste de la divergencia de Kullback–Leibler (D_{KL}) (que mide cómo de iguales son dos distribuciones de probabilidad),

$$D_{KL} = -\frac{1}{2} \cdot (1 + \log(\sigma^2) - \mu^2 - \sigma^2) \tag{2}$$

donde μ y σ^2 son la media y varianza estimadas. Este coste se combina con el Error Cuadrático Medio (ECM) entre la secuencia reconstruida $\hat{y}_i$ y el *ground truth* y_i durante T pasos de tiempo, que se calcula con

$$ECM = \frac{1}{T} \cdot \sum_{i=1}^{T} (y_i - \hat{y}_i)^2 \tag{3}.$$

Ambos costes se combinan en una suma ponderada

$$coste = \beta \cdot D_{KL} + ECM \tag{4}$$

donde β es un peso que puede ser tratado como un hiperparámetro ajustado de forma experimental que aumenta la importancia de una reconstrucción precisa (valor bajo) o regular en el espacio latente (valor alto) tal y como se muestra en [173].

5.2.2 *Modelo de predicción*

El modelo de predicción toma como entrada la situación de la escena de conducción previamente codificada, es decir, toma la representación latente de la situación de tráfico de los últimos T instantes de tiempo

($L_{n,\mu}$ y $L_{n,\sigma}$) donde n es el componente de la variable latente. Estas variables son generadas por los dos codificadores de la arquitectura, las cuales son concatenadas y son la entrada del modelo de predicción, que genera una representación en el espacio latente de la trayectoria futura del vehículo estimada $M_{x,m}$ y $M_{y,m}$, donde m es el componente de la variable latente de salida.

El modelo de predicción consta de dos partes. La primera se encarga de muestrear las variables latentes de entrada, ya que el codificador de un modelo VAE genera dos valores por cada variable, que corresponden a la media ($L_{n,\mu}$) y varianza ($L_{n,\sigma}$) de una distribución de Gauss. El muestreo se realiza generando un valor aleatorio para cada variable siguiendo su distribución normal. Este proceso se realiza tanto para generar una predicción, como a la hora de entrenar el modelo, haciendo que el modelo sea más robusto que si se usase siempre solamente el valor medio y descartase la varianza. Además, el muestreo de los datos de entrada al modelo permite estimar la varianza de las trayectorias generadas utilizando el método de Monte Carlo.

Los datos muestreados son introducidos a un modelo basado en redes neuronales que utiliza una arquitectura relativamente sencilla basada en capas Densas. Teniendo en cuenta que los datos de entrada han sido ya procesados por el codificador, y que los datos de salida serán también procesados por el decodificador, el modelo no requiere una complejidad muy alta.

5.2.3 *Entrenamiento del modelo completo*

El modelo completo se entrena en dos fases: En primer lugar, se entrenan los modelos VAE. El modelo VAE_{alr} se encargará de codificar la escena de la carretera, por lo que es entrenado con los datos de los vehículos que se encuentran alrededor del vehículo estudiado. Por otra parte, VAE_{xy} se entrena únicamente con trayectorias de la base de datos. El objetivo de estos dos modelos es codificar los datos de entrada en una variable con un número de dimensiones menor, para luego decodificarlos y generar unos datos lo más parecidos posible a los originales. Por esta razón, para el entrenamiento se utilizan los mismos datos para alimentar el modelo, y para calcular el coste, utilizando el ECM entre la salida del VAE y los datos de entrada. Una vez entrenados estos dos modelos, es necesario ensamblar el modelo completo para entrenar el de predicción. Este ensamblado, consiste en utilizar los codificadores ya entrenados de los modelos VAE_{alr} y VAE_{xy} que codificarán tanto la situación de tráfico como la trayectoria seguida por el vehículo en los últimos instantes de tiempo (las variables $X_{m,t}$, donde m puede ser el índice de la variable de entrada al modelo VAE_{alr} o x/y para indicar trayectoria longitudinal o lateral respectivamente y t es el instante de tiempo). Los datos

generados por estos codificadores concatenados ($L_{n,\sigma}$ y $L_{n,\mu}$) serán la entrada del modelo de predicción, previamente muestreados como se indica en la Sección 5.2.2. La salida del modelo de predicción ($M_{x/y,0}$ y $M_{x/y,1}$) será la entrada del decodificador que generará la predicción de la trayectoria futura $Y_{x/y,t}$. El coste utilizado al entrenar se calcula utilizando el ECM de la trayectoria generada y la real, el cual se propaga hacia atrás a través del decodificador, cuyos pesos, junto a los de los dos codificadores, están fijos durante el proceso de entrenamiento del modelo de predicción.

La Figura 5.2 muestra un esquema general del modelo completo ensamblado.

5.2.4 *Análisis de las variables latentes*

En primer lugar, es necesario elegir el número de variables latentes que son necesarias a la hora de definir los modelos VAE. Este valor tiene que ser suficientemente grande para representar correctamente todas las variables de entrada, pero a su vez lo más bajo posible para que se puedan extraer componentes relevantes que representen características de mayor nivel.

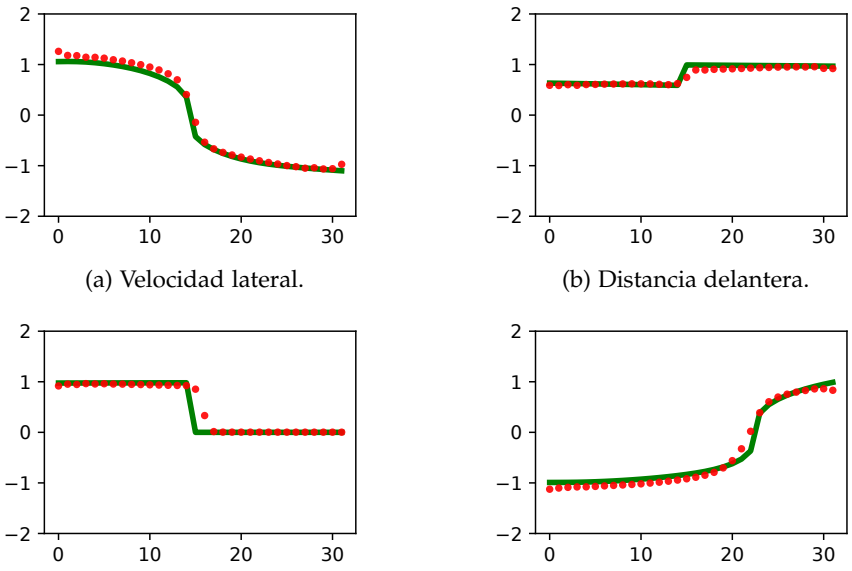
Para elegir este parámetro, se ha realizado un Análisis de Componentes Principales (ACP) utilizando la implementación LAPACK de la descomposición en valores singulares tal y como se muestra en [174].

En primer lugar, se eligen un número de dimensiones lo suficientemente grande para entrenar el VAE. Después, se analizan los resultados del ACP y, utilizando la suma acumulada de la varianza de cada componente principal, se elige el menor número de dimensiones que representa al menos el 95% de los datos que, para el caso del modelo VAE_{surr} es 7 y para el modelo VAE_{xy} es 2.

Una vez determinado el número de dimensiones de cada espacio latente, se realiza un estudio de este espacio, generado por los codificadores del VAE. Este estudio trata de corroborar dos hipótesis:

- El espacio latente representa correctamente los datos de entrada.
- Situaciones similares se encuentran cerca en el espacio latente y situaciones distintas lejos.

Para comprobar la primera hipótesis, se calcula el ECM de los datos reconstruidos por el decodificador y los originales, dando un resultado cuantitativo de 0.7m para la trayectoria lateral y 0.04 para la longitudinal. Además, se realiza un análisis visual de distintas muestras de los datos donde se aprecia de forma cualitativa una buena reconstrucción como se puede observar en la Figura 5.3, donde los datos reconstruidos son similares a los originales y se captura la evolución en el tiempo de estos datos, mostrando que los modelos VAE son capaces de representar todos los datos relevantes en las variables latentes.



(a) Velocidad lateral.

(b) Distancia delantera.

(c) Distancia trasera.

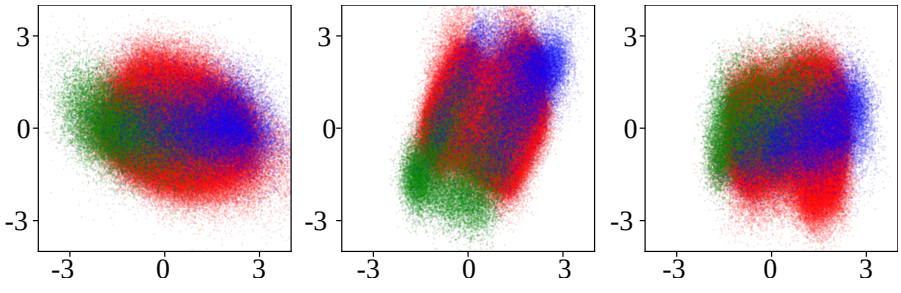
(d) Aceleración longitudinal.

Figura 5.3: Reconstrucción del modelo VAE_{surr} (en rojo) de algunas variables de entrada (en verde). El eje X representa el tiempo, mientras que el eje Y es el valor normalizado de la variable.

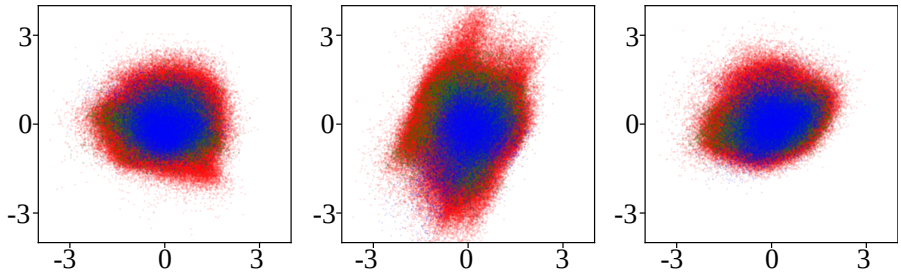
Para la segunda hipótesis, los datos en el espacio latente que representan situaciones en las que se realiza o se va a realizar en un futuro cercano un cambio de carril, se agrupan en tres grupos que representan: cambios al carril izquierdo, al carril derecho o ningún cambio.

La figura 5.4 compara las dos variables más representativas del hiperespacio latente para un modelo VAE y un AE normal. Las maniobras codificadas en el espacio latente para VAE son claramente distinguibles y forman tres grupos distintos (Figura5.4a), mientras que para un AE normal, las distintas maniobras no se pueden distinguir y se representan en la misma región (Figura 5.4b). Este análisis muestra además que, únicamente utilizando información de los vehículos de los alrededores, es posible predecir la próxima maniobra, y el VAE es capaz de distinguir y separar estas situaciones de forma no supervisada.

Además, se ha realizado otro análisis para corroborar la segunda hipótesis. La Figura 5.5,muestra las trayectorias generadas por un decodificador VAE para distintas combinaciones de dos variables del espacio latente. El modelo VAE ha aprendido una representación regular de todas las posibles trayectorias en el espacio latente de tal forma que, en este caso concreto, la dimensión latente 0 representa en su mayor parte la posición lateral en la carretera, mientras que la dimensión latente 1 indica el tipo de maniobra o desplazamiento lateral. Esta representación regular de todas las trayectorias posibles en el espacio latente mejorará la precisión del modelo completo, ya



(a) VAE



(b) AE

Figura 5.4: Representación de las dos dimensiones más relevantes del hiperespacio latente para VAE y AE. Los colores muestran la maniobra futura: mantener el carril (rojo), cambiar al carril izquierdo (verde) y cambiar al carril derecho (azul).

que un pequeño error en la predicción de la trayectoria futura en las variables del espacio latente tendrá como resultado una trayectoria muy similar.

Asimismo, la representación en un espacio dimensional más reducido, donde las distintas maniobras están agrupadas, facilita el trabajo del modelo de predicción para generar predicciones correctas y reduce la complejidad requerida de dicho modelo.

5.3 RESULTADOS

El modelo propuesto ha sido evaluado utilizando secuencias de una base de datos de trayectorias para evaluar sus características y precisión. En esta sección se describe el proceso de obtención de las métricas necesarias para determinar la precisión del modelo, se comparan los resultados con modelos del estado del arte y por último se evalúan los resultados obtenidos.

El modelo descrito en este capítulo ha sido evaluado utilizando distintas secuencias de datos obtenidas de la base de datos highD.

En los siguientes aparatados se describe el proceso de evaluación del modelo y se presentan los datos obtenidos, comparándolos con otros métodos del estado del arte.

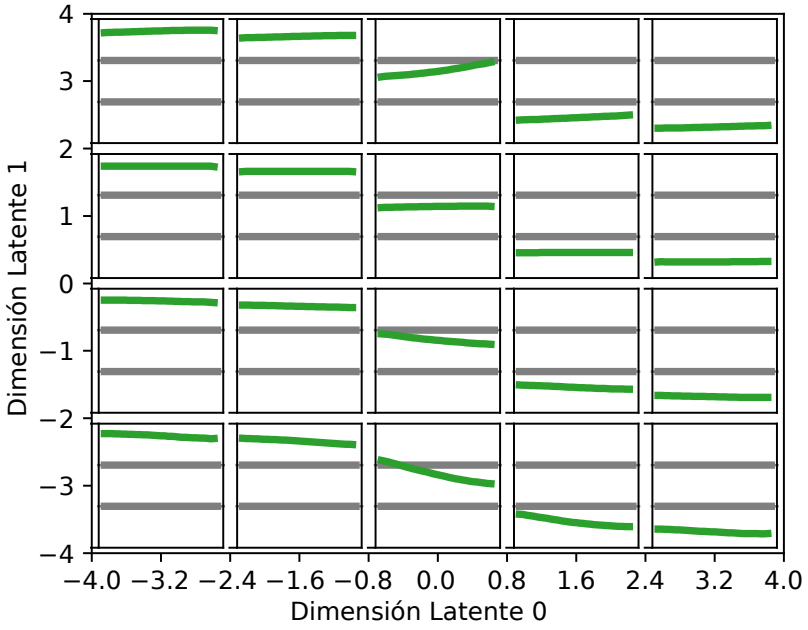


Figura 5.5: Reconstrucción del modelo VAE (línea verde) para distintos pares de valores del espacio latente (Dimensión Latente 0, Dimensión Latente 1). Los límites del carril están representados por líneas grises.

5.3.1 *Métricas de evaluación*

Existen trabajos similares para la predicción de trayectorias que evalúan su precisión utilizando bien la Raíz del Error Cuadrático Medio (RECM) para la trayectoria completa o bien el EAM de forma separada para las predicciones longitudinales y laterales para evaluar la precisión del modelo. Por este motivo, se utilizarán estas dos métricas para evaluar el modelo propuesto y poder compararlo con trabajos similares.

Los resultados presentados son evaluados en la parte de pruebas de la base de datos, de tal forma que el modelo nunca ha visto estos datos en la etapa de entrenamiento.

5.3.2 *Comparación con otros modelos*

La Tabla 5.1 compara la precisión de las trayectorias generadas con método propuesto para distintos horizontes de tiempo con los siguientes modelos:

1. 3D CNN-LSTM [116]: Se trata de un modelo que es consciente de las interacciones utilizando un modelo basado en un tensor espaciotemporal que representa los alrededores del entorno. Este

método aporta mejores resultados que un modelo LSTM simple o un modelo basado en *convolutional social pooling* LSTM [119].

2. Multi-Headd Att. [117]: Es un modelo basado en una capa llamada *multi-head attention*. El trabajo original demuestra que el método propuesto obtiene mejores resultados que un modelo lineal, un modelo basado en capas LSTM y un modelo codificador-decodificador con capas LSTM. Este trabajo sólo aporta resultados con un horizonte temporal máximo de 3 segundos.

3. AE simple: Es una copia exacta del modelo propuesto, pero los modelos VAE son entrenados como un AE normal. Este modelo sirve para comprobar si el uso de un modelo VAE aporta las mejoras previstas con respecto a un AE simple.

Para asegurar que las comparaciones son equitativas, se han considerado los siguientes aspectos: El modelo propuesto ha utilizado la misma base de datos para entrenar que todos los métodos elegidos con los comparar (highD). Además, como estos métodos utilizan distintas métricas (EAM y RECM), se han calculado ambos valores para el método propuesto, de tal manera que la comparación con el resto de los métodos se pueda realizar con la misma métrica.

	Time Hor. (s)	**3D CNN-LSTM EAM (m)**	**Multi-Head Att. RECM (m)**	**Simple AE EAM (m)**	**Método propuesto EAM/ RECM (m)**
Longitudinal	1	0.23	0.43	0.25	**0.21/0.30**
	2	0.59	**0.47**	0.5	0.37/0.52
	3	1.12	0.89	0.74	**0.46/0.68**
	4	1.81	-	1.07	**0.70/0.98**
	5	2.63	-	1.39	**0.91/1.33**
Lateral	1	0.07	**0.04**	0.23	0.06/0.09
	2	0.16	**0.06**	0.23	0.13/0.18
	3	0.25	**0.11**	0.24	0.14/0.20
	4	0.32	-	0.27	**0.20/0.28**
	5	0.39	-	0.30	**0.24/0.36**

Tabla 5.1: Valores de EAM y RECM longitudinal y lateral para los distintos métodos.

5.3.3 *Obtención y evaluación de los resultados*

Los datos de los resultados se han obtenido evaluando las predicciones del modelo propuesto, que son comparadas con las trayectorias reales recogidas en la base de datos.

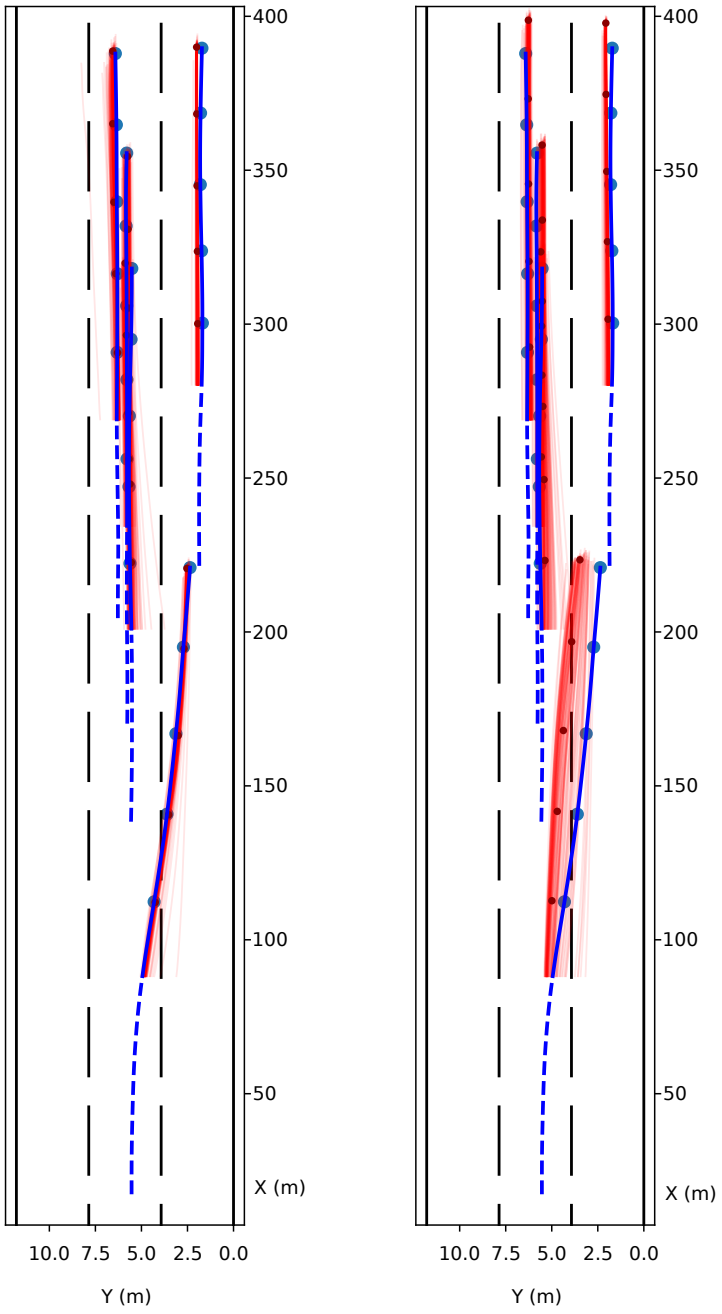
Como entrada al modelo, se han introducido distintas secuencias de 5 segundos de duración que incluyen todos los datos preprocesados que necesita el modelo. El modelo se ha evaluado y configurado cinco veces para generar predicciones de entre uno y cinco segundos. Los resultados obtenidos se comparan con las trayectorias reales que ha seguido el vehículo en cada secuencia, calculando distintas métricas que estiman la precisión del modelo.

La Tabla 5.1 muestra cómo el método propuesto captura mejor la situación del entorno y genera trayectorias realistas con mayor precisión que el resto de los métodos para horizontes temporales largos. A medida que este horizonte temporal se reduce, los resultados no muestran apenas mejoras.

Por otro lado, la Figura 5.6a muestra un ejemplo de la predicción de trayectoria de cada vehículo en la escena. Se puede observar que estas trayectorias son similares a la original lateral y longitudinalmente y, aunque algunas trayectorias aparezcan lejos del *ground truth*, la media de todas las trayectorias muestreadas sí es similar. De hecho, una de las principales ventajas de este método, es la posibilidad de obtener estadísticas de confianza de la trayectoria generada. Esta confianza viene determinada por la varianza, la cual es baja para situaciones conocidas, como se puede observar en la Figura 5.6a, sin embargo, para situaciones muy distintas a los datos de entrenamiento o con mucho ruido, el error de la predicción será mayor, pero también lo será el valor de la varianza, tal y como muestra la Figura 5.6b, donde se ha añadido una pequeña cantidad de ruido a los datos de entrada al modelo.

Además, el método propuesto es capaz de funcionar suficientemente rápido para poder ejecutarse y funcionar en un vehículo autónomo. El muestreo de las predicciones se puede realizar en paralelo, reduciendo el tiempo de computación en comparación con otro tipo de modelos. Los datos de todos los vehículos se pueden concatenar y repetir varias veces para muestrear con el método de Monte Carlo en paralelo utilizando una GPU.

Las pruebas realizadas utilizando la base de datos de trayectorias prueba la viabilidad el método propuesto basado en un modelo VAE con capas LSTM, y los resultados muestran mejoras significativas con respecto a los métodos más relevantes y actuales que se pueden encontrar en la literatura. Además, la capacidad de estimar una varianza de las trayectorias generadas aporta una característica no siempre presente en los algoritmos de generación de trayectorias, pero



(a) Predicción utilizando datos reales. (b) Predicción utilizando datos con errores añadidos de forma intencional.

Figura 5.6: La predicción del modelo (rojo), *ground truth* de la trayectoria pasada (azul rayado) y de la trayectoria futura (azul oscuro con puntos separados cada segundo) para todos los vehículos de la escena. Todas las trayectorias generadas por el modelo aparecen con un valor de transparencia alto para visualizar mejor la varianza del método y comparar las varianzas de ambas figuras. Los valores de la trayectoria media de la predicción aparecen con puntos rojo oscuro separados cada segundo.

muy útil para el módulo de planificación que puede necesitar esta estimación.

Por último, el código fuente utilizado en este trabajo ha sido publicado y se puede encontrar en el siguiente enlace para que el método pueda ser replicado https://github.com/midemig/traj_pred_vae.

6

INTERACCIÓN ENTRE PEATÓN Y VEHÍCULO AUTÓNOMO

Los vehículos totalmente autónomos de nivel V, los cuales no necesitan un conductor, aún no están muy extendidos, concentrándose su uso en zonas controladas como industrias o almacenes. Sin embargo, su rápida evolución en los últimos años, así como los avances tecnológicos en esta área, hacen que cada vez vayan a estar más presentes en entornos compartidos con otros vehículos no autónomos y peatones.

Actualmente existen numerosas situaciones en las que peatones y conductores interactúan, siendo los pasos de peatones una de las más habituales donde el peatón espera a ser visto por el conductor para cruzar. Con el VA, la ausencia del conductor puede llevar a dudas y a comportamientos no habituales. Por este motivo, entender y estudiar la interacción entre estos vehículos y los peatones es de gran interés, ya que su integración en este tipo de entornos depende de la aceptación por parte del resto de usuarios.

El trabajo realizado en esta tesis acerca de la interfaz hombre-máquina consta del estudio de dos aspectos distintos. En primer lugar, se analiza en detalle la interacción entre peatones y vehículo autónomo atendiendo a distintos aspectos (si lo han visto, reacciones al verlo, hábitos de comportamiento con vehículos no autónomos). Por otro lado, se han probado distintas HMI para analizar cuál obtiene mejores resultados.

6.1 ESTUDIO DE DISTINTAS INTERFACES HOMBRE MÁQUINA

El estudio de las distintas interfaces se realiza mediante pruebas reales en las que se expone a distintos peatones a un vehículo autónomo que incorpora una HMI.

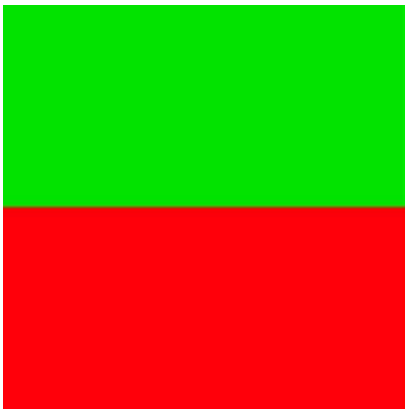
Dicha interfaz tiene dos estados posibles, uno para indicar que el vehículo ha detectado al peatón y por lo tanto se parará, y otro para indicar que no ha detectado a nadie y por lo tanto seguirá su camino. La elección de uno de los dos estados está determinada por un algoritmo de detección que genera una lista de todos los obstáculos que se encuentran en el área por el que va a pasar el vehículo; si esta lista contiene algún obstáculo, se activa el primer estado y si por el contrario no contienen nada, el segundo.

Este capítulo incluye contenido de [6] y [7].

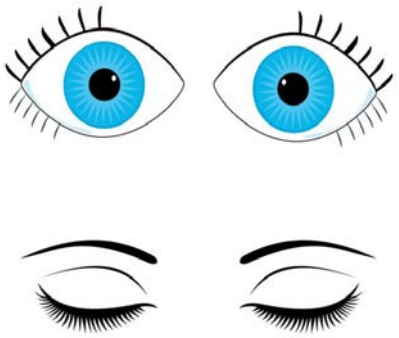
Esta interfaz está integrada en el vehículo de tal manera que puede leer los mensajes que publica el módulo de detección de obstáculos y mostrar la imagen correspondiente al estado.

Las distintas interfaces que se analizan tienen como variante la imagen que muestran en cada estado y están descritas a continuación:

- Referencia: En primer lugar, se realiza una prueba sin ninguna interfaz para poder determinar más tarde si es importante disponer de una comunicación entre el vehículo y el peatón.
- Colores: Se han elegido los colores rojo y verde para transmitir al peatón el mensaje de si es o no seguro cruzar por delante del vehículo (Figura 6.1a). Este paradigma está inspirado en el código de colores estándar de los semáforos, conocido por todos los peatones
- Ojos: Inspirado por las situaciones de la vida real en las que establecer contacto visual con el conductor del vehículo que se aproxima es esencial para los peatones en los pasos de cebra. Se ha simulado esta interacción con una imagen de dos ojos abiertos (te veo) y una imagen de un par de ojos cerrados (no te he visto), tal y como se muestra en la Figura 6.1b. Esta idea también ha sido adoptada por la industria en algunos prototipos [175].



(a) Colores.



(b) Ojos.

Figura 6.1: Imágenes propuestas para mostrar detección o no detección del peatón.

6.1.1 *Plataformas*

Para realizar los experimentos, se han utilizado dos plataformas distintas correspondientes al vehículo desarrollado en esta tesis (descrito en el capítulo 3) y al VA iCab [176].

El VA iCab es una plataforma de investigación completamente funcional construida a partir de un carro de golf modificado que puede navegar de forma autónoma. El vehículo está equipado con *encoders* en las ruedas, cámara de visión estéreo, LIDAR, IMU y GNSS, lo cual le permite tanto navegar de forma autónoma como detectar obstáculos. Este vehículo se configura para realizar la ruta programada de forma autónoma, deteniéndose en caso de detectar algún obstáculo a menos de 3 metros en frente del vehículo. La utilización de esta plataforma permite recoger más datos de interacciones en entornos peatonales en los que no es posible circular con un automóvil convencional y así poder validar la interfaz de comunicación en entornos más variados.

Para ambos vehículos, además de todo lo descrito en este documento, se les incorpora una pantalla en el frontal orientada hacia delante, dónde se muestran las imágenes de la interfaz. Esta pantalla consta de 23 pulgadas de diagonal y un brillo de $250 cd/m^2$.

6.2 EXPERIMENTOS

Para obtener resultados sobre la validación y la mejora de las interfaces de comunicación propuestas, se realizan una serie de pruebas reales con peatones. El objetivo es estudiar las reacciones de los peatones al vehículo que se les aproxima en función de los distintos mensajes que aparecen en la interfaz.

Para obtener estos datos, se ha definido un escenario que se encuentra en el campus de Leganés de la Universidad Carlos III de Madrid. Se trata de un entorno con zonas verdes y peatonales en las que el tráfico está restringido sólo a vehículos autorizados y de suministro a baja velocidad. Dicho escenario conecta dos partes de la ciudad, por lo que, además de los estudiantes de la universidad, los habitantes de la ciudad también transitan por estas calles.

Para los experimentos se seleccionan distintos recorridos en los que el tránsito de peatones es perpendicular al recorrido programado del VA, creando múltiples oportunidades de interacción entre este y los peatones. El vehículo autónomo circula de forma constante por el recorrido especificado de forma autónoma y, a lo largo de la ruta, distintos peatones se cruzan con él e interactúan, sin saber que están formando parte de un experimento. Además, el vehículo en todo momento circula con el asiento del conductor vacío, haciendo evidente que se trata de un vehículo autónomo. Por otra parte, las cámaras y los sensores de grabación están a siempre a la vista, de tal forma que los peatones sepan que pueden estar siendo grabados.

La Figura 6.2 muestra un momento de interacción entre un peatón y el VA, donde se establece contacto visual.

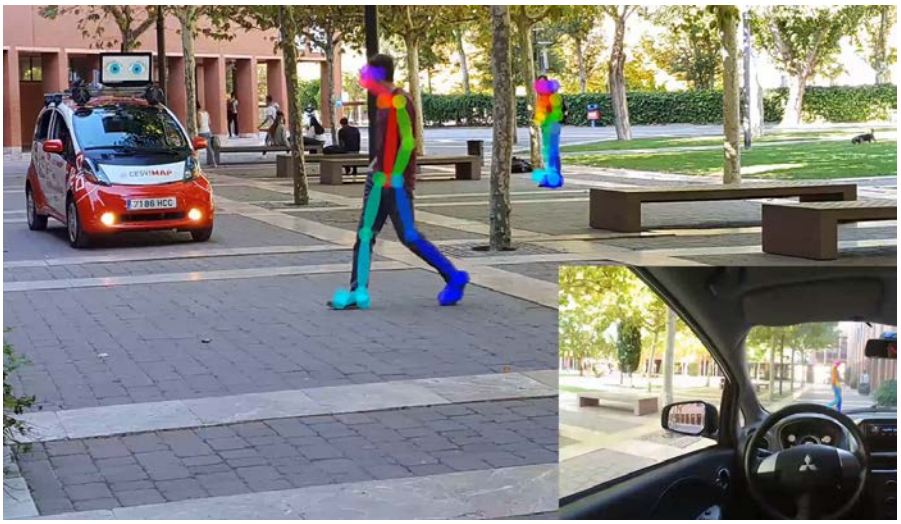


Figura 6.2: Interacción entre un peatón y el VA junto con la detección de la posición del peatón.

6.2.1 *Toma de datos*

Los experimentos se llevaron a cabo durante varios días, en los que los peatones entraban en contacto tanto con ambos vehículos autónomos.

Durante los experimentos, se realizaron grabaciones de datos para postprocesar obteniendo un total de 36 videos y 135 interacciones de peatones con el VA iCab y 22 videos y 49 interacciones de peatones con el VA desarrollado en esta tesis. Después de cada interacción, a los peatones que accedieron, se les pidió responder a una encuesta con distintas preguntas acerca de su experiencia con el vehículo y la interfaz, así como cuestiones relacionadas con sus hábitos en los cruces de peatones.

Para conseguir un entorno lo más parecido a la realidad, los peatones no eran conscientes de que se estaba llevando a cabo un experimento. Además, existía un control remoto capaz de enviar una señal de parada de emergencia al vehículo en caso de que ocurriera un fallo para aumentar la seguridad del experimento.

Para determinar la respuesta a los mensajes mostrados por la interfaz, se tuvieron en cuenta las siguientes variables:

- Distancia entre el peatón y el VA
- Velocidad del vehículo en su recorrido
- Posición del cuerpo y cabeza del peatón

Utilizando los datos mencionados, se pueden obtener patrones de comportamiento de los peatones, así como datos derivados como TTC. Estos datos fueron analizados teniendo en cuenta qué imagen se está mostrando en la interfaz en cada instante.

6.3 RESULTADOS

En este apartado se muestran los resultados obtenidos en los distintos experimentos realizados sobre la interacción humano-máquina. Estos experimentos han proporcionado datos subjetivos de la interacción con un VA a través de cuestionarios respondidos por los participantes en los experimentos (Sección 6.3.1). Además, se han obtenido datos objetivos en la Sección 6.3.2 a partir de las secuencias grabadas de los experimentos. Estos datos se consideran objetivos ya que en todas las secuencias analizadas se aplican los mismos criterios a la hora de determinar las variables de interés, al contrario que con los datos subjetivos, basados en la percepción propia de cada participante. Ambos experimentos proporcionan una valiosa información acerca de la interacción, pero de distinto carácter.

6.3.1 *Datos subjetivos cualitativos*

Estos datos han sido obtenidos mediante el análisis de las respuestas a los cuestionarios que han sido respondidos por los participantes en los experimentos de interacción con un VA.

La Tabla 6.1 resume todos los resultados obtenidos en estos experimentos, que son detallados con mayor profundidad en los siguientes apartados.

6.3.1.1 *Resultados generales y sin interfaz*

Ninguno de los participantes en el experimento había interactuado anteriormente con un VA, sin embargo, alguno había visto el vehículo con anterioridad circulando por el área de pruebas. Todos los participantes se dieron cuenta de que el vehículo era autónomo ya que no había nadie en el puesto del conductor. En cuanto al comportamiento en presencia del VA, sólo el 9% de los participantes esperó hasta que se detuviera completamente el vehículo, mientras que el resto cruzó mientras el vehículo decceleraba. Las dudas al cruzar fueron evaluadas en la escala de Likert del 1 al 5, donde 1 significa ninguna duda y 5 muchas dudas. En la prueba de referencia sin interfaz, la media de las dudas fue 1.29 donde el 27% de los participantes dijo no tener ninguna duda al cruzar.

Entre las posibles razones de incertidumbre están la falta de conocimiento sobre si el vehículo le ha detectado, si el vehículo se iba a detener o desconfianza de un fallo de los sensores.

6.3.1.2 *Pruebas de las interfaces*

Los resultados de los experimentos realizados muestran que la mayoría de los participantes (62%) vieron la imagen del monitor. De aquellos que la vieron, en el experimento utilizando la interfaz de los

Referencia sin interfaz	
Pregunta	**Resultado**
Aprecia que es autónomo	100% (si)
Espera a que se detenga totalmente el VA antes de cruzar	9% (si)
Valor medio de dudas al cruzar (1-5)	1.91
Ninguna duda al cruzar	27%
Código de colores / Imagen de ojos abiertos o cerrados	
Pregunta	**Resultado**
Aprecia que es autónomo	100% (si)
Espera a que se detenga totalmente el VA antes de cruzar	9% (si)
Valor medio de dudas al cruzar (1-5)	1.8
Ninguna duda al cruzar	35%
Vio la imagen del monitor	62% (si)
La imagen de los ojos se entiende	50% (si)
El código de colores se entiende	33% (si)
Preferencia de la imagen de ojos frente a colores	70% (ojos)
Hábitos al cruzar	
Pregunta	**Resultado**
Frecuencia media de contacto visual al cruzar (1-5)	4.1
Espera hasta que el vehículo se ha detenido completamente	50% (si)
Aumentaría la sensación de seguridad si el VA indica que te ha detectado	100% (si)

Tabla 6.1: Resumen de los resultados de las encuestas.

ojos abiertos o cerrados, la mitad consideraron el mensaje claro, el 25% no entendió el mensaje y el 25% restante no supo contestar.

Para la interfaz con código de colores, únicamente el 33% de los encuestados dijo haber entendido si debían o no cruzar.

Comparando las dos imágenes propuestas para la interfaz, al preguntar a los participantes en el experimento cuál indica mejor el mensaje, el 70% eligió la imagen de los ojos. Estos resultados no indicaron una preferencia significativa ($\chi^2(1, N = 21) = 7.54, p = 0.06$).

Utilizando las interfaces propuestas, tal y como se puede observar en la Figura 6.3 las dudas al cruzar en una escala de Likert del 1 al 5, fueron reducidas a un valor medio de 1.8 (DE=1.34) en contraste con 1.9 (DE=1.37) sin utilizar ninguna interfaz. Este efecto no es significativo (t(41) = 0.69, p = 0.24).

El porcentaje de gente que no dudó nada al cruzar (1 en la escala de Likert) aumentó del 27% sin interfaz al 38%, mostrando una tendencia de los peatones a sentirse más seguros con interfaz. Este efecto sin embargo no es significativo ($\chi^2(1, N = 42) = 0.29, p = 0.58$).

Por último, cabe destacar que ninguno respondió con altos niveles de dudas (4 o 5) y que la proporción de personas que pararon completamente antes de cruzar fue del 9% tanto con interfaz como sin ella.

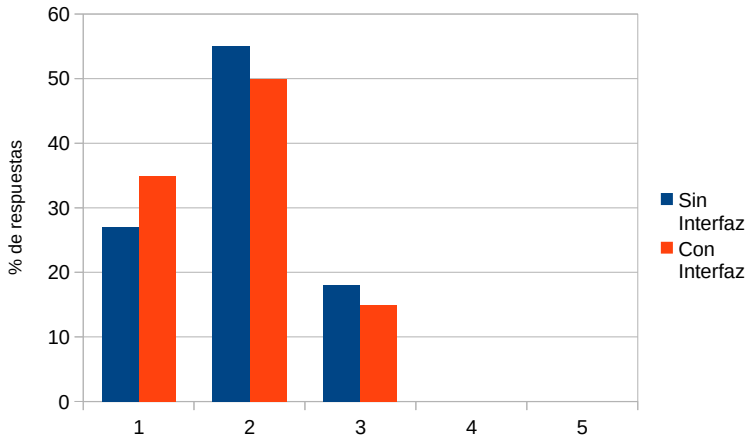


Figura 6.3: Porcentaje de dudas al cruzar con y sin interfaz en la escala Likert.

6.3.1.3 *Hábitos de cruce*

Todos los cuestionarios realizados, contaban con un apartado relativo a los hábitos de cada peatón en el momento de cruzar un paso de peatones.

Los resultados muestran que la mitad de los encuestados esperan hasta que el vehículo se detiene completamente para cruzar, mientras que la otra mitad no esperan en su día a día. En la pregunta sobre la frecuencia con la que se establece contacto visual con el conductor que se aproxima en el paso de peatones, en la escala de Likert del 1 al 5 donde 1 significa nunca y 5 siempre, el valor medio es 4.1 (DE=2.53), lo que significa que los peatones habitualmente miran a los ojos del conductor para saber si pueden o no cruzar (Figura 6.4).

Además del contacto visual para asegurarse de que se puede cruzar, los participantes en el experimento también consideraron como factor determinante la deceleración del vehículo.

Otro resultado notable es el hecho de que algunos peatones son afectados por el comportamiento de los vehículos y peatones de los alrededores para decidir si cruzar o no.

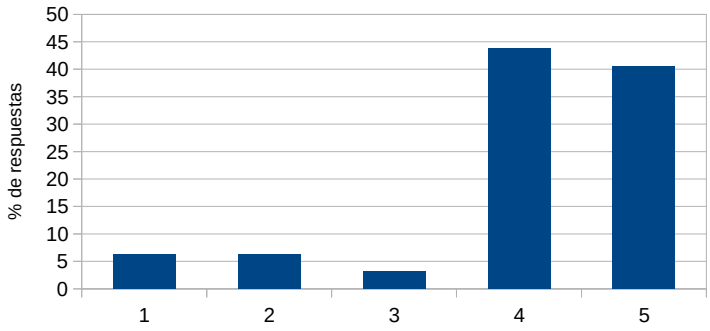


Figura 6.4: Resultados del contacto visual al cruzar en la escala de Likert.

En la pregunta relativa a la utilidad de la interfaz propuesta, todos los participantes coincidieron en que mostrar una imagen con información sobre si el vehículo autónomo le ha detectado, les haría sentir más seguros.

Estas respuestas confirmaron la necesidad de una interfaz que sea capaz de establecer una comunicación entre los peatones y el VA para aumentar la seguridad en los espacios públicos con un VA.

6.3.2 *Datos objetivos de análisis de secuencias*

Además de los datos obtenidos de las encuestas, todos los experimentos han sido grabados, respetando siempre la privacidad de los participantes de acuerdo con la Ley, para ser posteriormente analizados. En los siguientes apartados se analizan la interacción de los peatones con distintas interfaces, así como las reacciones generales de éstos al encontrarse con un VA.

6.3.2.1 *Resultados de la interfaz con el vehículo autónomo desarrollado*

Analizando los datos extraídos de las grabaciones de los experimentos realizados con el VA desarrollado en esta tesis, se pueden extraer las siguientes conclusiones:

Se puede determinar que 92 peatones (68.14%) miraron a la pantalla que sirve de interfaz mientras que 43 (31.86%) no se percataron del vehículo. De los peatones que miraron a la pantalla, se puede observar un mayor porcentaje de peatones que cruzaron frente al VA, independientemente del tipo de mensaje mostrado.

Analizando los resultados de la Tabla 6.2, se puede comprobar que las distribuciones entre variables categóricas difieren entre ellas, siendo estas diferencias en proporción de peatones que andan o paran cuando la pantalla muestra el color rojo o los ojos cerrados no estadísticamente significativa. Por lo tanto, no se puede rechazar la hipótesis nula.

Comparación	n	$\chi^2(\alpha = 0.05)$	valor p
Sin interfaz contra Color verde	31	1.99	0.158
Sin interfaz contra Color rojo	46	1.77	0.183
Sin interfaz contra Ojos abiertos	32	0.49	0.484
Sin interfaz contra Ojos cerrados	43	0.41	0.522
Color verde contra Color rojo	27	2.13	0.144
Color verde contra Ojos abiertos	23	0.49	0.484
Color verde contra Ojos cerrados	34	0.65	0.420
Color rojo contra Ojos abiertos	38	0.33	0.570
Color rojo contra Ojos cerrados	49	0.50	0.480
Ojos abiertos contra Ojos cerrados	35	0.01	0.974

Tabla 6.2: Valores estadísticos del test chi cuadrado (χ^2, 1 grado de libertad) para el comportamiento del peatón en función de la imagen mostrada.

Para los datos relativos a la distancia y TTC, los resultados del poder estadístico para muestras independientes de Test-T varían desde el 73% hasta el 97%, indicando que existe una baja probabilidad de tener un error de tipo II y de aceptar la hipótesis nula probando los parámetros. La Tabla 6.3 muestra los valores obtenidos. El análisis indica que la distancia del vehículo en el momento de cruzar es menor en la condición de control.

También se muestra que el TTC es menor cuando los mensajes con código de colores (rojo/verde) o las imágenes de ojos (abiertos/cerrados) se muestran. Sin embargo, estos valores no difieren de forma significativa entre participantes.

En cuanto a las personas que no vieron el vehículo, los resultados de TTC y distancia al vehículo mostraron que el efecto en la seguridad en la carretera de la falta de contacto visual en intersecciones no señalizadas no es significativo (Tabla 6.4).

	Métrica	t(92)	p
Sin interfaz	Distancia	1.27	0.20
contra Colores	TDC	1.51	0.13
Sin interfaz	Distancia	0.85	0.39
contra Ojos	TDC	1.07	0.28
Colores	Distancia	0.67	0.55
contra Ojos	TDC	0.90	0.12

Tabla 6.3: Distancia entre el peatón y el vehículo y TDC al cruzar para distintas imágenes mostradas.

Métrica	Sin contacto visual		Con contacto visual		Test t (a = 0.05)	
	Media	DE	Media	DE	t(113)	p
Distancia (m)	6.93	3.28	7.81	3.56	1.41	0.1599
TDC (s)	5.87	6.71	8.93	12.22	1.37	0.1726

Tabla 6.4: Efecto del contacto visual en la interacción con el VA.

6.3.2.2 *Interacción entre peatón y vehículo autónomo*

Analizando los vídeos de los experimentos, se pueden apreciar una serie de reacciones hacia el iCab. Algunos de los peatones parecían no advertir que el vehículo que se aproxima era autónomo. Muchos peatones estaban tan involucrados en el uso de su teléfono móvil que no prestaban atención a sus alrededores, sorprendiéndose repentinamente por el vehículo. Algunas reacciones mostraron incertidumbre, ya que dudaban si podían o no cruzar por delante de él. Este caso en particular se producía con más frecuencia cuando el vehículo no circula en línea recta. Si el VA avanza recto, muchas personas se movían a la izquierda o derecha para evitar la confrontación

El VA pareció ser muy interesante para la mayoría de los peatones, ya que atrajo su curiosidad y mostraron interés. Muchos de ellos incluso tomaron fotos o vídeos. En muchos casos, sonrieron e incluso comprobaron si el vehículo se paraba al pasar. También se apreciaron las reacciones contrarias: algunos peatones intentaron evitar acercarse al vehículo.

Como resultado del análisis del vídeo, se pueden apreciar siete categorías diferentes. Estas categorías se pueden clasificar como positivas, neutrales o negativas, como se muestra en la Tabla 6.5.

Cuatro de estas categorías son clasificadas como positivas, y representan el 51% de todas las reacciones e incluyen:

- Sonrisa: todos los peatones que sonrieron después de ver el vehículo.

Reacción	Distribución	Grupo	Nº de Personas	%
Positiva	51.02%	Sonrisa	14	28.6
		Interés	6	12.2
		Prueba	2	4.1
		Foto/Video	3	6.1
Neutral	22,49%	Neutral	12	24.5
Negativa	24,249	Desvío	7	14.3
		Desconfianza	5	10.2

Tabla 6.5: Resultados del análisis de los vídeos grabados.

- Interés: todos aquellos que examinaron el vehículo después de aproximarse a ellos.
- Prueba: las personas que probaron varias veces a ponerse delante del VA para comprobar que se paraba.
- Foto / vídeo: todos los que capturaron el vehículo con una cámara.

Las reacciones de los peatones reacios a mirar el vehículo fueron clasificadas como neutrales. Alrededor del 24.5% de personas pertenecen a este grupo.

Por último, una reacción negativa hacia el vehículo fue mostrada por dos grupos que aparentaban esquivar el vehículo. Alrededor del 24.5% de las reacciones se engloban en este grupo, que incluye:

- Desvío: dar un paso hacia otro lado mientras el vehículo se aproxima o una expresión facial o corporal que denota aversión.
- Desconfianza: peatones que dejan pasar el vehículo y esperan sin moverse hasta que éste continúa y se aleja.

6.3.3 *Discusión*

La mayoría de los participantes encuestados en los experimentos coincidieron en que la existencia de un sistema de comunicación para interactuar con los vehículos sin conductor era positiva. Esto implica la necesidad de medidas para informar a los ciudadanos sobre la presencia de los vehículos autónomos por diferentes medios.

Sin embargo, los resultados presentados en este documento no mostraron diferencias estadísticamente diferencias estadísticamente significativas en la proporción de peatones que siguieron caminando y cruzaron delante del VA y los que se detuvieron al observar el mensaje de la pantalla. Además, en la mayoría de los casos se observó que

los peatones cruzaron incluso cuando el mensaje estaba en rojo o mostraba los ojos cerrados. Aparentemente, la detección del vehículo por parte de los peatones es suficiente para tomar la decisión de cruzar o no. Por lo tanto, no se ha podido determinar si es más probable que los peatones se detengan y se abstengan de cruzar delante de un VA si no se muestra un mensaje. El motivo de estos hechos puede estar relacionado con la velocidad del vehículo (la cual no supera nunca los $5m/s$), ya que es menos probable que los peatones respondan ante un VA que se mueve a baja velocidad y no les supone un peligro.

Como se describe en la Sección 2.6, los trabajos anteriores han demostrado la importancia del movimiento del vehículo (por ejemplo, la velocidad y la distancia) para los peatones. Sin embargo, estos trabajos están basados en su mayoría en simulaciones o en datos subjetivos, mientras que el trabajo de esta tesis describe los resultados cuantitativos de una prueba de campo realizada con un VA sin conductor.

Por otro lado, la relación entre la ausencia o presencia de contacto visual en parámetros relacionados con la seguridad vial, como la distancia y el TTC no ha sido significativa. Por lo tanto, no se puede confirmar o descartar si el contacto visual con un VA en el escenario de los experimentos, en el que el tráfico se basa más en las normas informales del tráfico a pie, sin semáforos, marcas viales o señales que indiquen el derecho de paso, facilitó la acción cooperativa.

Podemos concluir que, a partir de los resultados obtenidos, que la implementación de una interfaz de comunicación visual para interactuar con los peatones no es necesariamente indispensable para un espacio compartido en el que se aplican normas de tráfico informales. Es más probable que esta tecnología ayude en mayor medida cuando el vehículo y el peatón tienen conflictos potenciales que causan peligro. Estos resultados están en consonancia con los hallazgos de [145–147], los cuales afirman que la información mostrada en los monitores externos no es determinante para definir un comportamiento en los peatones, siendo la distancia y la velocidad del vehículo más decisivas [148].

7 CONCLUSIONES Y TRABAJOS FUTUROS

En este capítulo se resumen los logros alcanzados en el trabajo desarrollado en esta tesis. Además, también se exponen algunas de las líneas de investigación futuras que la darían continuidad.

7.1 CONCLUSIONES

El trabajo desarrollado en esta tesis ha permitido desarrollar un vehículo autónomo funcional, que cumple con los requisitos establecidos en la Sección 1.2 y le permite navegar entre dos puntos de forma autónoma.

La arquitectura desarrollada realiza de forma autónoma el control tanto lateral como longitudinal del vehículo. Es decir, tanto el giro del volante, como la aceleración y el freno, son controlados de forma autónoma y le permiten circular por la carretera sin intervención humana. Por otro lado, este vehículo también puede reaccionar ante la aparición de obstáculos, detectando qué clase de obstáculo es, su posición y actuando en consecuencia (parando o esquivándolo si fuera necesario). Además, a pesar de que por normativa siempre existe un conductor supervisando el funcionamiento del vehículo en las pruebas, el vehículo ha sido diseñado para que no sea necesaria su intervención en ninguna de las tareas de la conducción.

Sin embargo, el dominio de entornos o situaciones en las que el vehículo puede funcionar de forma autónoma son limitadas, ya que los entornos en los que puede operar están determinados por el área que ha sido mapeado previamente. Por lo tanto, el nivel de automatización de este vehículo, de acuerdo a los criterios establecidos en la SAE [32], corresponde con un nivel IV.

Las principales contribuciones aportadas en esta tesis se resumen a continuación:

- El diseño de una arquitectura, tanto software como hardware, que permite poner en marcha un VA totalmente funcional. Esta arquitectura dota a la plataforma de gran flexibilidad a la hora de cambiar tanto módulos de hardware (sensores u ordenadores), como módulos de software (algoritmos). Esta cualidad, la hace ideal para probar distintos algoritmos y evaluar su funcionamiento o compararlo con el de otros algoritmos del estado del arte.

- Un análisis de los tipos de sensores que existen actualmente, y su adecuación a cada tarea de percepción. Se han analizado las

características necesarias de la plataforma desarrollada, y se ha propuesto una configuración de sensores incluyendo sus características y colocación con respecto al vehículo. Se ha descrito el proceso seguido para validar la configuración de sensores y su posicionamiento.

- Mejora de la localización de un vehículo. Ante los problemas que se observan en algunos entornos con los algoritmos de localización más habituales (basados en GNSS y basados en un mapa y un sensor LIDAR), se ha propuesto un nuevo algoritmo que combina ambas localizaciones. Este sistema de localización mejora a los dos métodos que combina por separado y le permite navegar por entornos por los que antes presentaba dificultades.

- Se ha desarrollado un modelo de predicción para los vehículos en el entorno del VA. Este modelo genera, a partir de la detección de los vehículos de los alrededores, su trayectoria futura más probable. De esta manera, se puede mejorar notablemente la planificación local de trayectorias.

- Se ha realizado un estudio evaluando la capacidad de los modelos VAE tanto para analizar las distintas situaciones de tráfico que pueden aparecer, como para generar trayectorias futuras en base a cada situación. Se ha demostrado que los modelos VAE tienen una mayor capacidad para representar las situaciones de tráfico y para generar trayectorias realistas que los modelos AE.

- Desarrollo de una HMI que permite mejorar la interacción entre el VA y los peatones. Esta interfaz muestra mensajes a los peatones a través de una pantalla, en la que se indica la intención del vehículo (parar o continuar) y si ha detectado al peatón. Los mensajes mostrados buscan ser intuitivos para una persona que nunca los haya visto.

- Se ha realizado un estudio de la interacción entre un VA y los peatones que circulan a su alrededor. En este estudio, además de analizar la efectividad de los distintos tipos de HMI propuestas, se estudian otros aspectos relevantes del comportamiento de los peatones en presencia de un VA.

7.2 TRABAJOS FUTUROS

La investigación llevada a cabo en esta tesis ha hecho posible desarrollar un VA con un nivel IV de autonomía, capaz de funcionar bajo unas condiciones específicas en el entorno. Los trabajos futuros apuntan a la dirección de conseguir una automatización de nivel V, donde el vehículo es capaz de navegar por cualquier tipo de entorno, aunque no haya sido mapeado o analizado previamente. Entre todas

las tareas para conseguir esto, partiendo del trabajo realizado en esta tesis, destacan los siguientes puntos:

- Una de las principales líneas de investigación futura relacionada con la arquitectura propuesta se centra en dotar al vehículo de la capacidad de circular, junto con otros vehículos, en entornos más complejos como intersecciones, rotondas, incorporaciones, etc. Todavía esta es un área en desarrollo que requiere un gran esfuerzo de investigación por sí solo que engloba además distintos módulos de la arquitectura (localización, percepción, etc.) de los que depende.

- Otro punto importante con el que continuar la investigación y mejora de la arquitectura, consiste en dotar de una mayor autonomía y prescindir del mapeado de todas las señales de tráfico y zonas relevantes. Esta línea de investigación estudiará la manera más efectiva de incorporar la información de los algoritmos de detección de señales de tráfico de todo tipo (pintadas, verticales o agentes de seguridad) a la arquitectura del vehículo para que funcione y se adapte a ellas.

- En el área de la localización, a pesar de que los resultados muestran que el algoritmo propuesto genera una localización del vehículo precisa y robusta ante distintos tipos de entornos, la dependencia de un mapa precomputado limita en gran medida el área en el que puede circular el VA. Las líneas de investigación futuras en este tema apuntan a mejorar el algoritmo de localización haciendo prescindible este mapa precomputado que bien puede ser generado mientras el vehículo se desplaza (con técnicas basadas en SLAM) o adquirido a través de otros vehículos que hayan circulado previamente por esa área, extendiendo la investigación hacia la localización cooperativa.

- En el área de la predicción de la trayectoria futura de los vehículos de los alrededores, los trabajos futuros principales se enfocan en adaptar la arquitectura del modelo propuesto a distintos escenarios dentro de las ciudades como las intersecciones o las rotondas, ya que el trabajo realizado en esta tesis se limita a carreteras continuas de uno o más carriles. Además, se pueden mejorar estas predicciones incluyendo datos de los sensores en crudo (imágenes o nubes de puntos) directamente que generen una mejor descripción del entorno aportando más información que las detecciones de estos vehículos.

- En cuanto la investigación acerca de las HMI, al no estar muy extendidos los vehículos autónomos, requiere todavía un gran trabajo de investigación que permita determinar su utilidad y eficacia. Entre los trabajos necesarios para continuar la investigación realizada en este campo, destaca la mejora de la interfaz

de comunicación hacia una más clara e intuitiva. Además de probar distintos códigos de imágenes u otros tipos de información visual y estudiar su efectividad, también se pueden analizar otras formas de comunicación distintas basadas en señales sonoras o patrones de comportamiento (mediante las formas concretas de decelerar o patrones de movimiento del vehículo) que faciliten el reconocimiento de la intención del VA, o bien una combinación de todas las mencionadas.

Existen numerosos indicios para pensar que la investigación relacionada con los vehículos autónomos va a ser un tema con gran repercusión en los próximos años. Este trabajo debe ser visto como una contribución en el gran esfuerzo de investigación que es necesario para el objetivo final de conseguir una conducción totalmente autónoma, segura y eficiente.

BIBLIOGRAFÍA

[1] Miguel Angel de Miguel, Fernando Garcia, and Jose Maria Armingol. «Improved LiDAR Probabilistic Localization for Autonomous Vehicles Using GNSS.» In: *Sensors* 20.11 (2020), p. 3145.

[2] Miguel Angel de Miguel, Francisco Miguel Moreno, Pablo Marin-Plaza, Abdulla Al-Kaff, Martin Palos, David Martin, Rodrigo Encinar-Martin, and Fernando Garcia. «A Research Platform for Autonomous Vehicles Technologies Research in the Insurance Sector.» In: *Applied Sciences* 10.16 (2020), p. 5655.

[3] Enrique Marti, Miguel Angel de Miguel, Fernando Garcia, and Joshue Perez. «A review of sensor technologies for perception in automated driving.» In: *IEEE Intelligent Transportation Systems Magazine* 11.4 (2019), pp. 94–108.

[4] Pablo Marin-Plaza, David Yagüe, Francisco Royo, Miguel Ángel de Miguel, Francisco Miguel Moreno, Alejandro Ruiz-de-la Cuadra, Fernando Viadero-Monasterio, Javier Garcia, José Luis San Roman, and José María Armingol. «Project ARES: Driverless Transportation System. Challenges and Approaches in an Unstructured Road.» In: *Electronics* 10.15 (2021), p. 1753.

[5] Francisco Alexis Quesada Arencibia, José Carlos Rodríguez Rodríguez, Roberto Moreno Díaz, R Moreno Diaz, Gabriele Salvatore De Blasio, and Carmelo Rubén García Rodríguez. «Computer Aided Systems Theory, Eurocast 2019. Extended Abstracts. EUROCAST 2019.» In: *Lecture Notes in Computer Science* (2019).

[6] Miguel Angel de Miguel, Daniel Fuchshuber, Ahmed Hussein, and Cristina Olaverri-Monreal. «Perceived pedestrian safety: Public interaction with driverless vehicles.» In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2019, pp. 90–95.

[7] Walter Morales Alvarez, Miguel Angel de Miguel, Fernando Garcia, and Cristina Olaverri-Monreal. «Response of vulnerable road users to visual information from autonomous vehicles in shared spaces.» In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE. 2019, pp. 3714–3719.

[8] Miguel Angel de Miguel, Francisco Miguel Moreno, Fernando Garcia, Jose Maria Armingol, and Rodrigo Encinar Martin. «Autonomous vehicle architecture for high automation.» In: *International Conference on Computer Aided Systems Theory*. Springer. 2019, pp. 145–152.

[9] Gabriel M Hoffmann, Claire J Tomlin, Michael Montemerlo, and Sebastian Thrun. «Autonomous automobile trajectory tracking for off-road driving: Controller design, experimental validation and racing.» In: *2007 American control conference*. IEEE. 2007, pp. 2296–2301.

[10] Mingkang Li, Martin Stolz, Zhaofei Feng, Martin Kunert, Roman Henze, and Ferit Küçükay. «An adaptive 3D grid-based clustering algorithm for automotive high resolution radar sensor.» In: *2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*. IEEE. 2018, pp. 1–7.

[11] Robert Krajewski, Julian Bock, Laurent Kloeker, and Lutz Eckstein. «The highD Dataset: A Drone Dataset of Naturalistic Vehicle Trajectories on German Highways for Validation of Highly Automated Driving Systems.» In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. 2018, pp. 2118–2125. DOI: 10.1109/ITSC.2018.8569552.

[12] Thomas Finateu et al. «5.10 A 1280×720 Back-Illuminated Stacked Temporal Contrast Event-Based Vision Sensor with 4.86µm Pixels, 1.066GEPS Readout, Programmable Event-Rate Controller and Compressive Data-Formatting Pipeline.» In: *2020 IEEE International Solid- State Circuits Conference - (ISSCC)*. 2020, pp. 112–114. DOI: 10.1109/ISSCC19947.2020.9063149.

[13] Abhipray Paturkar, Gourab Sen Gupta, and Donald Bailey. «Making Use of 3D Models for Plant Physiognomic Analysis: A Review.» In: *Remote Sensing* 13.11 (2021), p. 2232.

[14] Juan Carmona, Fernando Garcia, Miguel Angel de Miguel, Arturo de la Escalera, and José Maria Armingol. «Analysis of Aggressive Driver Behaviour using Data Fusion.» In: *VEHITS*. 2016, pp. 85–90.

[15] Juan Carmona, Miguel Angel de Miguel, David Martin, Fernando Garcia, and Arturo de la Escalera. «Embedded system for driver behavior analysis based on GMM.» In: *2016 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2016, pp. 61–65.

[16] International Organization of Motor Vehicle Manufacturers. *World vehicles in use*. 2015. URL: https://www.oica.net/wp-content/uploads/Total_in-use-All-Vehicles.pdf (visited on 01/20/2022).

[17] International Organization of Motor Vehicle Manufacturers. *World vehicles in use*. 2015. URL: https://www.oica.net/wp-content/uploads//PC_Vehicles-in-use.pdf (visited on 01/20/2022).

[18] NHTSA. *Federal Automated Vehicles Policy*. Tech. rep. September. NHTSA, 2016, pp. 1–116. URL: https://www.transportation.gov/AV{\%}0Apapers3://publication/uuid/2E47DA05-79E3-4DA5-BE3B-B172900149A1.

[19] National Highway Traffic Safety Administration (NHTSA). *2016 Fatal Motor Vehicle Crashes: Overview*. 2016. URL: https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812456 (visited on 01/20/2022).

[20] World Health Organization. *Road traffic injuries*. 2021. URL: https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries (visited on 02/15/2022).

[21] Jacob Teter, Jacopo Tattini, and Apostolos Petropoulos. *Tracking Transport 2020*. 2020. URL: https://www.iea.org/reports/tracking-transport-2020 (visited on 12/09/2021).

[22] World Health Organization. *Air pollution*. 2021. URL: https://www.who.int/health-topics/air-pollution (visited on 12/09/2021).

[23] World Health Organization. *What are the effects on health of transport-related air pollution?* 2020. URL: https://www.euro.who.int/en/data-and-evidence/evidence-informed-policy-making/publications/hen-summaries-of-network-members-reports/what-are-the-effects-on-health-of-transport-related-air-pollution (visited on 12/09/2021).

[24] Michał Krzyżanowski, Birgit Kuna-Dibbert, and Jürgen Schneider. *Health effects of transport-related air pollution*. WHO Regional Office Europe, 2005.

[25] Bob Pishue. «Inrix global traffic scorecard.» In: (2021).

[26] LexisNexis. *LexisNexis Risk Solutions Launches Vehicle Build Product to Offer U.S. Insurers Much-Needed ADAS Details for Better Premium Evaluation and Pricing Segmentation*. 2020. URL: https://risk.lexisnexis.com/about-us/press-room/press-release/20200618-vehicle-build (visited on 02/15/2022).

[27] Jessica B Cicchino. «Effects of blind spot monitoring systems on police-reported lane-change crashes.» In: *Traffic injury prevention* 19.6 (2018), pp. 615–622.

[28] Highway Loss Data Institute [HLDI]. «Predicted availability and fitment of safety features on registered vehicles—a 2019 update.» In: *HLDI Bull* 35.27 (2018), p. 9.

[29] Organisation Internationale des Constructeurs d'Automobiles (OICA). *Automated Driving, Definition for Levels of Automation*. 2014. URL: https://www.oica.net/wp-content/uploads//WP-29-162-20-OICA-automated-driving.pdf (visited on 02/15/2022).

[30] U.S. Department of Transportation. *U.S. Department of Transportation Releases Policy on Automated Vehicle Development*. 2019. URL: https://www.transportation.gov/briefing-room/us-department-transportation-releases-policy-automated-vehicle-development (visited on 02/15/2022).

[31] Tom M Gasser and Daniel Westhoff. «BASt-study: Definitions of automation and legal issues in Germany.» In: *Proceedings of the 2012 road vehicle automation workshop*. Bergisch Gladbach: German Federal Highway Research Institute. 2012, pp. 1–20.

[32] SAE International. *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*. 2021. URL: https://www.sae.org/standards/content/j3016_202104/ (visited on 12/09/2021).

[33] E.D. Dickmanns and A. Zapp. «Autonomous High Speed Road Vehicle Guidance by Computer Vision 1.» In: *IFAC Proceedings Volumes* 20.5 (June 1987), pp. 221–226. ISSN: 14746670. DOI: 10.1016/S1474-6670(17)55320-3. URL: https://www.sciencedirect.com/science/article/pii/S1474667017553203.

[34] Rudolf Gregor, M. Lützeler, M. Pellkofer, K. H. Siedersberger, and Ernst Dieter Dickmanns. «EMS-Vision: A Perceptual System for Autonomous Vehicles.» In: *IEEE Transactions on Intelligent Transportation Systems* 3.1 (Mar. 2002), pp. 48–59. ISSN: 15249050. DOI: 10.1109/6979.994795. URL: http://ieeexplore.ieee.org/document/994795/.

[35] Alberto. Broggi, Massimo Bertozzi, Alessandra Fascioli, and Gianni Conte. *Automatic Vehicle Guidance : the Experience of the ARGO Autonomous Vehicle*. World Scientific, 1999, p. 242. ISBN: 9810237200. URL: www.ce.unipr.it/people/broggi/publications/argo.pdf.

[36] M. Parent and P. Daviet. «Automatic Driving For Small Public Urban Vehicles.» In: *Proceedings of the Intelligent Vehicles '93 Symposium*. IEEE, 1993, pp. 402–407. ISBN: 0-7803-1370-4. DOI: 10.1109/IVS.1993.697360. URL: http://ieeexplore.ieee.org/document/697360/.

[37] Nicola Bernini, Massimo Bertozzi, Luca Castangia, Marco Patander, and Mario Sabbatelli. «Real-time obstacle detection using stereo vision for autonomous ground vehicles: A survey.» In: *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*. IEEE. 2014, pp. 873–878.

[38] C. Thorpe, T. Jochem, and D. Pomerleau. «The 1997 automated highway free agent demonstration.» In: *Proceedings of Conference on Intelligent Transportation Systems*. IEEE, 1997, pp. 496–501. ISBN: 0-7803-4269-0. DOI: 10.1109/ITSC.1997.660524. URL: http://ieeexplore.ieee.org/document/660524/.

[39] Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, and Al. Et. «The robot that won the DARPA Grand Challenge.» In: *Journal of Field Robotics* 23-9.9 (2006), pp. 661–692. ISSN: 1556-4967. DOI: 10.1002/rob.20147. URL: www.interscience.wiley.comhttp://doi.wiley.com/10.1002/rob.20147.

[40] Tartan Racing. *Boss at a glance*. 2005. URL: http://www.tartanracing.org/press/boss-glance.pdf.

[41] Chris Urmson. *DARPA Urban Challenge Final Report for Tartan Racing*. Tech. rep. 2007. URL: https://pdfs.semanticscholar.org/3116/38e299acef3cbd3423649b77ef73c2a94fc1.pdf.

[42] Michael Montemerlo et al. «Junior: the Stanford entry in the Urban Challenge.» In: *Journal of Field Robotics* 25.9 (Sept. 2008), pp. 569–597. ISSN: 15564959. DOI: 10.1002/rob.20258. URL: http://doi.wiley.com/10.1002/rob.20258.

[43] Jesse Levinson et al. «Towards fully autonomous driving: Systems and algorithms.» In: *IEEE Intelligent Vehicles Symposium, Proceedings*. IEEE, June 2011, pp. 163–168. ISBN: 9781457708909. DOI: 10.1109/IVS.2011.5940562. arXiv: 1702.06827. URL: http://ieeexplore.ieee.org/document/5940562/.

[44] Lindsay Chapell. *The Big Bang of autonomous driving*. 2016. URL: http://www.autonews.com/article/20161219/OEM06/312199908/the-big-bang-of-autonomous-driving (visited on 10/08/2018).

[45] Massimo Bertozzi, Alberto Broggi, Elena Cardarelli, Rean Isabella Fedriga, Luca Mazzei, and Pier Paolo Porta. «VIAC expedition toward autonomous mobility.» In: *IEEE Robotics and Automation Magazine*. Vol. 18. 3. Sept. 2011, pp. 120–124. DOI: 10.1109/MRA.2011.942490. URL: http://ieeexplore.ieee.org/document/6016589/.

[46] A. Broggi, P. Medici, P. Zani, A. Coati, and M. Panciroli. «Autonomous vehicles control in the VisLab Intercontinental Autonomous Challenge.» In: *Annual Reviews in Control* 36.1 (Apr. 2012), pp. 161–171. ISSN: 13675788. DOI: 10.1016/j.arcontrol.2012.03.012. URL: https://www.sciencedirect.com/science/article/pii/S1367578812000132.

[47] Alberto Broggi, Stefano Cattani, Paolo Medici, and Paolo Zani. «Applications of computer vision to vehicles: An extreme test.» In: *Studies in Computational Intelligence* 411 (2013), pp. 215–250. ISSN: 1860949X. DOI: 10.1007/978-3-642-28661-2_9. URL: https://link.springer.com/content/pdf/10.1007/978-3-642-28661-2{_}9.pdf.

[48] Michael Aeberhard, Sebastian Rauch, Mohammad Bahram, Georg Tanzmeister, Julian Thomas, Yves Pilat, Florian Homm, Werner Huber, and Nico Kaempchen. «Experience, results and lessons learned from automated driving on Germany's highways.» In: *IEEE Intelligent Transportation Systems Magazine* 7.1 (2015), pp. 42–57. ISSN: 15249050. DOI: 10.1109/MITS.2014.2360306. URL: http://ieeexplore.ieee.org/document/7014396/.

[49] Julius Ziegler et al. «Making bertha drive-an autonomous journey on a historic route.» In: *IEEE Intelligent Transportation Systems Magazine* 6.2 (2014), pp. 8–20. ISSN: 19391390. DOI: 10.1109/MITS.2014.2306552. URL: http://ieeexplore.ieee.org/document/6803933/.

[50] Stephen Edelstein. *Intel/Mobileye Self-Driving Cars Begin Testing in Jerusalem - The Drive*. 2018. URL: http://www.thedrive.com/tech/20919/intel-mobileye-self-driving-cars-begin-testing-in-jerusalem (visited on 10/06/2018).

[51] Steven Scheer. *Exclusive: Intel's Mobileye gets self-driving tech deal for 8 million cars*. 2018. URL: https://www.reuters.com/article/us-israel-tech-intel-mobileye-exclusive/exclusive-intels-mobileye-gets-self-driving-tech-deal-for-8-million-cars-idUSKCN1II0K7 (visited on 10/06/2018).

[52] Andrew Hawkins. *Tesla has been working on a backup plan in case its self-driving promises fail*. 2017. URL: https://www.theverge.com/2017/8/9/16119746/tesla-self-driving-hardware-upgrade-hw-2-5 (visited on 11/13/2018).

[53] Waymo team. *Introducing Waymo's suite of custom-built, self-driving hardware*. 2017. URL: https://medium.com/waymo/introducing-waymos-suite-of-custom-built-self-driving-hardware-c47d1714563 (visited on 10/06/2018).

[54] Department of Motor Vehicles State of California. *DMV approves Cruise and Waymo to use autonomous vehicles for commercial service in designated parts of bay area*. 2021. URL: https://www.dmv.ca.gov/portal/news-and-media/117199-2/ (visited on 02/15/2022).

[55] Mercedes-Benz. *First internationally valid system approval for conditionally automated driving*. 2021. URL: https://group.mercedes-benz.com/innovation/product-innovation/autonomous-driving/system-approval-for-conditionally-automated-driving.html (visited on 02/15/2022).

[56] Pablo Marín Plaza. «Software architecture for self-driving navigation.» In: *Universidad Carlos III de Madrid* (2018).

[57] Toni A Bishop and Ramesh K Karne. «A Survey of Middleware.» In: *Computers and Their Applications*. 2003, pp. 254–258.

[58] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, Andrew Y Ng, et al. «ROS: an open-source Robot Operating System.» In: *ICRA workshop on open source software*. Vol. 3. 3.2. Kobe, Japan. 2009, p. 5.

[59] Nathan Koenig and Andrew Howard. «Design and use paradigms for gazebo, an open-source multi-robot simulator.» In: *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*. Vol. 3. IEEE, pp. 2149–2154.

[60] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. «CARLA: An Open Urban Driving Simulator.» In: *Proceedings of the 1st Annual Conference on Robot Learning*. 2017, pp. 1–16.

[61] Dirk Thomas, William Woodall, and Esteve Fernandez. «Next-generation ROS: Building on DDS.» In: *ROSCon Chicago 2014*. Mountain View, CA: Open Robotics, Sept. 2014. DOI: 10.36288/ROSCon2014-900183. URL: https://vimeo.com/106992622.

[62] Yuya Maruyama, Shinpei Kato, and Takuya Azumi. «Exploring the performance of ROS2.» In: *Proceedings of the 13th ACM SIGBED International Conference on Embedded Software (EMSOFT)*. 2016, pp. 1–10.

[63] Giorgio Metta, Paul Fitzpatrick, and Lorenzo Natale. «YARP: yet another robot platform.» In: *International Journal of Advanced Robotic Systems* 3.1 (2006), p. 8.

[64] Abhinav Valada, Noha Radwan, and Wolfram Burgard. «Deep Auxiliary Learning for Visual Localization and Odometry.» In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. May 2018.

[65] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. «Vision meets Robotics: The KITTI Dataset.» In: *International Journal of Robotics Research (IJRR)* (2013).

[66] Nicholas Carlevaris-Bianco, Arash K. Ushani, and Ryan M. Eustice. «University of Michigan North Campus long-term vision and lidar dataset.» In: *International Journal of Robotics Research* 35.9 (2015), pp. 1023–1035.

[67] Vassili Alexiadis. «Video-Based Vehicle Trajectory Data Collection.» In: *Transportation Research Board 86th Annual Meeting*. Citeseer. 2006.

[68] Benjamin Coifman and Lizhe Li. «A critical evaluation of the Next Generation Simulation (NGSIM) vehicle trajectory dataset.» In: *Transportation Research Part B: Methodological* 105 (2017), pp. 362–377.

[69] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. «Scalability in perception for autonomous driving: Waymo open dataset.» In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 2446–2454.

[70] Ming-Fang Chang et al. «Argoverse: 3D Tracking and Forecasting with Rich Maps.» In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.

[71] Robert Krajewski, Tobias Moers, Julian Bock, Lennart Vater, and Lutz Eckstein. «The rounD Dataset: A Drone Dataset of Road User Trajectories at Roundabouts in Germany.» submitted.

[72] Julian Bock, Robert Krajewski, Tobias Moers, Steffen Runde, Lennart Vater, and Lutz Eckstein. «The inD Dataset: A Drone Dataset of Naturalistic Road User Trajectories at German Intersections.» In: 2019.

[73] R. O'Malley, Martin Glavin, and E. Jones. «A review of automotive infrared pedestrian detection techniques.» In: *Signals and Systems Conference, 208.(ISSC 2008). IET Irish* (2008), pp. 168–173. DOI: 10.1049/cp:20080657. URL: http://digital-library.theiet.org/content/conferences/10.1049/cp{_}20080657papers2://publication/uuid/FFD556E9-8AF3-48F2-9C9B-3B57247056B7{\%}5Cnhttp://ieeexplore.ieee.org/xpls/abs{_}all.jsp?arnumber=4780948{\%}5Cnpapers2://publication/uuid/FC2526CD-D332-4959-B833.

[74] Bassem Besbes, Alexandrina Rogozan, Adela Maria Rus, Abdelaziz Bensrhair, and Alberto Broggi. «Pedestrian detection in far-infrared daytime images using a hierarchical codebook of SURF.» In: *Sensors (Switzerland)* 15.4 (Apr. 2015), pp. 8570–8594. ISSN: 14248220. DOI: 10.3390/s150408570. URL: http://www.mdpi.com/1424-8220/15/4/8570/.

[75] Rostam Affendi Hamzah and Haidi Ibrahim. «Literature survey on stereo vision disparity map algorithms.» In: *Journal of Sensors* 2016 (Dec. 2016), pp. 1–23. ISSN: 16877268. DOI: 10.1155/2016/8742920. URL: http://www.hindawi.com/journals/js/2016/8742920/.

[76] Giulio Reina, David Johnson, and James Underwood. «Radar sensing for intelligent vehicles in urban environments.» In: *Sensors (Switzerland)* 15.6 (June 2015), pp. 14661–14678. ISSN: 14248220. DOI: 10.3390/s150614661. URL: http://www.mdpi.com/1424-8220/15/6/14661/.

[77] Eli Brookner. «Metamaterial Advances for Radar and Communications.» In: *IEEE International Symposium on Phased Array Systems and Technology (PAST)*. IEEE, Oct. 2016, pp. 1–9. ISBN:

978-1-5090-1447-7. DOI: 10.1109/ARRAY.2016.7832577. URL: http://ieeexplore.ieee.org/document/7832577/.

[78] Timothy Sleasman, Michael Boyarsky, Laura Pulido-Mancera, Thomas Fromenteze, Mohammadreza F. Imani, Matthew S. Reynolds, and David R. Smith. «Experimental Synthetic Aperture Radar with Dynamic Metasurfaces.» In: *IEEE Transactions on Antennas and Propagation* 65.12 (Feb. 2017), pp. 6864–6877. ISSN: 0018926X. DOI: 10.1109/TAP.2017.2758797. arXiv: 1703.00072. URL: http://arxiv.org/abs/1703.00072http://dx.doi.org/10.1109/TAP.2017.2758797.

[79] Tyson Govan Phillips, Nicky Guenther, and Peter Ross McAree. «When the Dust Settles: The Four Behaviors of LiDAR in the Presence of Fine Airborne Particulates.» In: *Journal of Field Robotics* 34.5 (Aug. 2017), pp. 985–1009. ISSN: 15564967. DOI: 10.1002/rob.21701. arXiv: 10.1.1.91.5767. URL: http://doi.wiley.com/10.1002/rob.21701.

[80] Daniel Nordin. «Optical Frequency Modulated Continuous Wave (FMCW) Range and Velocity Measurements.» In: *Thesis* (2004), p. 110. URL: https://www.diva-portal.org/smash/get/diva2:999065/FULLTEXT01.pdfhttp://epubl.luth.se/1402-1544/2004/43/LTU-DT-0443-SE.pdf.

[81] Paul F. Mcmanamon et al. «Optical phased array technology.» In: *Proceedings of the IEEE* 84.2 (Feb. 1996), pp. 268–298. ISSN: 00189219. DOI: 10.1109/5.482231. URL: http://ieeexplore.ieee.org/document/482231/.

[82] Brandon Schoettle. *Sensor Fusion: A Comparison of Sensing Capabilities of Human Drivers and Highly Automated Vehicles*. Tech. rep. SWT-2017-12. University of Michigan, 2017, pp. 1–42. URL: http://www.umich.edu/{~}umtriswt.

[83] Malek Karaim, Mohamed Elsheikh, and Aboelmagd Noureldin. «GNSS Error Sources.» In: *Multifunctional Operation and Application of GPS*. IntechOpen, 2018.

[84] DF Huang, YL Xiong, and LG Yuan. «Global positioning system (GPS)-theory and practice.» In: *Xi'an Jiaotong University Press, Chengdu* (2006), pp. 71–73.

[85] Marcus Obst, Sven Bauer, Pierre Reisdorf, and Gerd Wanielik. «Multipath detection with 3D digital maps for robust multi-constellation GNSS/INS vehicle localization in urban areas.» In: *2012 IEEE Intelligent Vehicles Symposium*. IEEE. 2012, pp. 184–190.

[86] Honglei Qin, Xia Xue, and Qian Yang. «GNSS multipath estimation and mitigation based on particle filter.» In: *IET Radar, Sonar & Navigation* 13.9 (2019), pp. 1588–1596.

[87] Dieter Fox, Wolfram Burgard, Frank Dellaert, and Sebastian Thrun. «Monte carlo localization: Efficient position estimation for mobile robots.» In: *AAAI/IAAI* 1999.343-349 (1999), pp. 2–2.

[88] Patrick Pfaff, Wolfram Burgard, and Dieter Fox. «Robust monte-carlo localization using adaptive likelihood models.» In: *European robotics symposium 2006*. Springer. 2006, pp. 181–194.

[89] KyungJae Ahn and Yeonsik Kang. «A Particle Filter Localization Method Using 2D Laser Sensor Measurements and Road Features for Autonomous Vehicle.» In: *Journal of Advanced Transportation* 2019 (2019).

[90] J-S Gutmann and Dieter Fox. «An experimental comparison of localization methods continued.» In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. Vol. 1. IEEE. 2002, pp. 454–459.

[91] Andras Majdik, Mircea Popa, Levente Tamas, Istvan Szoke, and Gheorghe Lazea. «New approach in solving the kidnapped robot problem.» In: *ISR 2010 (41st International Symposium on Robotics) and ROBOTIK 2010 (6th German Conference on Robotics)*. VDE. 2010, pp. 1–6.

[92] Sherif AS Mohamed, Mohammad-Hashem Haghbayan, Tomi Westerlund, Jukka Heikkonen, Hannu Tenhunen, and Juha Plosila. «A survey on odometry for autonomous navigation systems.» In: *IEEE Access* 7 (2019), pp. 97466–97486.

[93] Davide Scaramuzza and Friedrich Fraundorfer. «Visual odometry [tutorial].» In: *IEEE robotics & automation magazine* 18.4 (2011), pp. 80–92.

[94] Ji Zhang and Sanjiv Singh. «LOAM: Lidar Odometry and Mapping in Real-time.» In: *Robotics: Science and Systems*. Vol. 2. 9. 2014.

[95] Greg Welch, Gary Bishop, et al. «An introduction to the Kalman filter.» In: (1995).

[96] Ryuichi Ueda, Tamio Arai, Kohei Sakamoto, Toshifumi Kikuchi, and Shogo Kamiya. «Expansion resetting for recovery from fatal error in monte carlo localization-comparison with sensor resetting methods.» In: *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*. Vol. 3. IEEE. 2004, pp. 2481–2486.

[97] Hiroki Goto, Ryuichi Ueda, and Yasuo Hayashibara. «Resetting Method using GNSS in LIDAR-based Probabilistic Self-localization.» In: *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE. 2018, pp. 1113–1118.

[98] Yiploon Seow, Renato Miyagusuku, Atsushi Yamashita, and Hajime Asama. «Detecting and solving the kidnapped robot problem using laser range finder and wifi signal.» In: *2017 IEEE international conference on real-time computing and robotics (RCAR)*. IEEE. 2017, pp. 303–308.

[99] Matthias Hentschel, Oliver Wulf, and Bernardo Wagner. «A GPS and laser-based localization for urban and non-urban outdoor environments.» In: *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2008, pp. 149–154.

[100] Daniel Perea, Antonio Morell, Jonay Toledo, and Leopoldo Acosta. «GNSS integration in the localization system of an autonomous vehicle based on Particle Weighting.» In: *IEEE Sensors Journal* (2019).

[101] Richard S Wallace, Anthony Stentz, Charles E Thorpe, Hans P Moravec, William Whittaker, and Takeo Kanade. «First Results in Robot Road-Following.» In: *IJCAI*. 1985, pp. 1089–1095.

[102] Omead Amidi and Chuck E Thorpe. «Integrated mobile robot control.» In: *Mobile Robots V*. Vol. 1388. International Society for Optics and Photonics. 1991, pp. 504–523.

[103] R Craig Coulter. *Implementation of the pure pursuit path tracking algorithm*. Tech. rep. Carnegie-Mellon UNIV Pittsburgh PA Robotics INST, 1992.

[104] Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, et al. «Stanley: The robot that won the DARPA Grand Challenge.» In: *Journal of field Robotics* 23.9 (2006), pp. 661–692.

[105] Francesco Borrelli, Paolo Falcone, Tamas Keviczky, Jahan Asgari, and Davor Hrovat. «MPC-based approach to active steering for autonomous vehicle systems.» In: *International journal of vehicle autonomous systems* 3.2-4 (2005), pp. 265–291.

[106] Paolo Falcone, Francesco Borrelli, Jahan Asgari, Hongtei Eric Tseng, and Davor Hrovat. «Predictive active steering control for autonomous vehicle systems.» In: *IEEE Transactions on control systems technology* 15.3 (2007), pp. 566–580.

[107] Jason Kong, Mark Pfeiffer, Georg Schildbach, and Francesco Borrelli. «Kinematic and dynamic vehicle models for autonomous driving control design.» In: *2015 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2015, pp. 1094–1099.

[108] Egbert Bakker, Lars Nyborg, and Hans B Pacejka. «Tyre modelling for use in vehicle dynamics studies.» In: *SAE Transactions* (1987), pp. 190–204.

[109] Sampo Kuutti, Richard Bowden, Yaochu Jin, Phil Barber, and Saber Fallah. «A survey of deep learning applications to autonomous vehicle control.» In: *IEEE Transactions on Intelligent Transportation Systems* (2020).

[110] Stéphanie Lefèvre, Dizan Vasquez, and Christian Laugier. «A survey on motion prediction and risk assessment for intelligent vehicles.» In: *ROBOMECH journal* 1.1 (2014), pp. 1–14.

[111] Aris Polychronopoulos, Manolis Tsogas, Angelos J Amditis, and Luisa Andreone. «Sensor fusion for predicting vehicles' path for collision avoidance systems.» In: *IEEE Transactions on Intelligent Transportation Systems* 8.3 (2007), pp. 549–562.

[112] Adam Houenou, Philippe Bonnifait, Véronique Cherfaoui, and Wen Yao. «Vehicle trajectory prediction based on motion model and maneuver recognition.» In: *2013 IEEE/RSJ international conference on intelligent robots and systems*. IEEE. 2013, pp. 4363–4369.

[113] Wen Yao, Huijing Zhao, Philippe Bonnifait, and Hongbin Zha. «Lane change trajectory prediction by using recorded human driving data.» In: *2013 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2013, pp. 430–436.

[114] Gabriel Agamennoni, Juan I Nieto, and Eduardo M Nebot. «A Bayesian approach for driving behavior inference.» In: *2011 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2011, pp. 595–600.

[115] Gabriel Agamennoni, Juan I Nieto, and Eduardo M Nebot. «Estimation of multivehicle dynamics by considering contextual information.» In: *IEEE Transactions on Robotics* 28.4 (2012), pp. 855–870.

[116] Martin Krüger, Anne Stockem Novo, Till Nattermann, and Torsten Bertram. «Interaction-Aware Trajectory Prediction based on a 3D Spatio-Temporal Tensor Representation using Convolutional–Recurrent Neural Networks.» In: *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2020, pp. 1122–1127.

[117] Hayoung Kim, Dongchan Kim, Gihoon Kim, Jeongmin Cho, and Kunsoo Huh. «Multi-Head Attention based Probabilistic Vehicle Trajectory Prediction.» In: *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2020, pp. 1720–1725.

[118] Kyushik Min, Hayoung Kim, Jongwon Park, Dongchan Kim, and Kunsoo Huh. «Interaction Aware Trajectory Prediction of Surrounding Vehicles with Interaction Network and Deep Ensemble.» In: *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2020, pp. 1714–1719.

[119] Nachiket Deo and Mohan M Trivedi. «Convolutional social pooling for vehicle trajectory prediction.» In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2018, pp. 1468–1476.

[120] Florent Altché and Arnaud de La Fortelle. «An LSTM network for highway trajectory prediction.» In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2017, pp. 353–359.

[121] Seong Hyeon Park, ByeongDo Kim, Chang Mook Kang, Chung Choo Chung, and Jun Won Choi. «Sequence-to-sequence prediction of vehicle trajectory via LSTM encoder-decoder architecture.» In: *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2018, pp. 1672–1678.

[122] Shengzhe Dai, Li Li, and Zhiheng Li. «Modeling vehicle interactions via modified LSTM models for trajectory prediction.» In: *IEEE Access* 7 (2019), pp. 38287–38296.

[123] Guo Xie, Anqi Shangguan, Rong Fei, Wenjiang Ji, Weigang Ma, and Xinhong Hei. «Motion trajectory prediction based on a CNN-LSTM sequential model.» In: *Science China Information Sciences* 63.11 (2020), pp. 1–21.

[124] Xinyu Chen, Jiajie Xu, Rui Zhou, Wei Chen, Junhua Fang, and Chengfei Liu. «TrajVAE: A Variational AutoEncoder model for trajectory generation.» In: *Neurocomputing* (2020).

[125] Wenhao Ding, Wenshuo Wang, and Ding Zhao. «Multi-vehicle trajectories generation for vehicle-to-vehicle encounters.» In: *2019 IEEE International Conference on Robotics and Automation (ICRA)*. 2019.

[126] Diederik P Kingma and Max Welling. «An introduction to variational autoencoders.» In: *arXiv preprint arXiv:1906.02691* (2019).

[127] Xidong Feng, Zhepeng Cen, Jianming Hu, and Yi Zhang. «Vehicle trajectory prediction using intention-based conditional variational autoencoder.» In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE. 2019, pp. 3514–3519.

[128] Ade Candra, Mohammad Andri Budiman, and Kevin Hartanto. «Dijkstra's and A-Star in Finding the Shortest Path: a Tutorial.» In: *2020 International Conference on Data Science, Artificial Intelligence, and Business Analytics (DATABIA)*. IEEE. 2020, pp. 28–32.

[129] Hannah Bast, Daniel Delling, Andrew Goldberg, Matthias Müller-Hannemann, Thomas Pajor, Peter Sanders, Dorothea Wagner, and Renato F Werneck. «Route planning in transportation networks.» In: *Algorithm engineering*. Springer, 2016, pp. 19–80.

[130] Brian Paden, Michal Čáp, Sze Zheng Yong, Dmitry Yershov, and Emilio Frazzoli. «A survey of motion planning and control techniques for self-driving urban vehicles.» In: *IEEE Transactions on intelligent vehicles* 1.1 (2016), pp. 33–55.

[131] Eckard Minx and Rainer Dietrich. *Autonomes Fahren*. Piper ebooks, 2015.

[132] Kornélia Lazányi and Gréta Maráczi. «Dispositional trust—Do we trust autonomous cars?» In: *Intelligent Systems and Informatics (SISY), 2017 IEEE 15th International Symposium on*. IEEE. 2017, pp. 000135–000140.

[133] Peter A Hancock, Deborah R Billings, and Kristen E Schaefer. «Can you trust your robot?» In: *Ergonomics in Design* 19.3 (2011), pp. 24–29.

[134] Sarah Schmidt and Berthold Faerber. «Pedestrians at the kerb–Recognising the action intentions of humans.» In: *Transportation research part F: traffic psychology and behaviour* 12.4 (2009), pp. 300–310.

[135] Jennifer A Oxley, Elfriede Ihsen, Brian N Fildes, Judith L Charlton, and Ross H Day. «Crossing roads safely: an experimental study of age differences in gap selection by pedestrians.» In: *Accident Analysis & Prevention* 37.5 (2005), pp. 962–971.

[136] Hidekatsu Hamaoka, Toru Hagiwara, Masahiro Tada, and Kazunori Munehiro. «A study on the behavior of pedestrians when confirming approach of right/left-turning vehicle while crossing a crosswalk.» In: *Journal of the Eastern Asia Society for Transportation Studies* 10 (2013), pp. 2109–2122.

[137] Tobias Lagström and Victor Malmsten Lundgren. «AVIP-Autonomous vehicles' interaction with pedestrians-An investigation of pedestrian-driver communication and development of a vehicle external interface.» MA thesis. 2016.

[138] Su Yang. *Driver behavior impact on pedestrians' crossing experience in the conditionally autonomous driving context (Doctoral dissertation)*. 2017.

[139] League of American Bicyclists. *Autonomous and Connected Vehicles: Implications for Bicyclists and Pedestrians*. https://bikeleague.org/sites/default/files/Bike_Ped_Connected_Vehicles.pdf. 2014.

[140] Milecia Matthews, Girish Chowdhary, and Emily Kieson. «Intent communication between autonomous vehicles and pedestrians.» In: *arXiv preprint arXiv:1708.07123* (2017).

[141] Lex Fridman, Bruce Mehler, Lei Xia, Yangyang Yang, Laura Yvonne Facusse, and Bryan Reimer. «To walk or not to walk: Crowdsourced assessment of external vehicle-to-pedestrian displays.» In: *arXiv preprint arXiv:1707.02698* (2017).

[142] Karthik Mahadevan, Sowmya Somanath, and Ehud Sharlin. «Communicating awareness and intent in autonomous vehicle-pedestrian interaction.» In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. 2018, pp. 1–12.

[143] Chia-Ming Chang, Koki Toda, Daisuke Sakamoto, and Takeo Igarashi. «Eyes on a Car: an Interface Design for Communication between an Autonomous Car and a Pedestrian.» In: *Proceedings of the 9th international conference on automotive user interfaces and interactive vehicular applications*. 2017, pp. 65–73.

[144] Matthias Beggiato, Claudia Witzlack, Sabine Springer, and Josef Krems. «The right moment for braking as informal communication signal between automated vehicles and pedestrians in crossing situations.» In: *International Conference on Applied Human Factors and Ergonomics*. Springer. 2017, pp. 1072–1081.

[145] Dirk Rothenbücher, Jamy Li, David Sirkin, Brian Mok, and Wendy Ju. «Ghost driver: A field study investigating the interaction between pedestrians and driverless vehicles.» In: *2016 25th IEEE international symposium on robot and human interactive communication (RO-MAN)*. IEEE. 2016, pp. 795–802.

[146] Michael Clamann, Miles Aubert, and Mary L Cummings. *Evaluation of vehicle-to-pedestrian communication displays for autonomous vehicles*. Tech. rep. 2017.

[147] Anantha Pillai et al. «Virtual reality based study to analyse pedestrian attitude towards autonomous vehicles.» In: *Aalto University* (2017).

[148] Amir Rasouli and John K Tsotsos. «Autonomous vehicles that interact with pedestrians: A survey of theory and practice.» In: *IEEE transactions on intelligent transportation systems* 21.3 (2019), pp. 900–918.

[149] Miguel Angel de Miguel, Francisco Miguel Moreno, Fernando Garcia, and Jose Maria Armingol. «Autonomous vehicle architecture for high automation.» In: *EUROCAST 2019, Las Palmas de Gran Canarias*. EUROCAST 2019. Feb. 2019.

[150] Dirección General de Tráfico. *Código de Tráfico y Seguridad Vial*. 2021. URL: https://www.boe.es/biblioteca_juridica/codigos/abrir_pdf.php?fich=020_Codigo_de_Trafico_y_Seguridad_Vial.pdf (visited on 12/09/2021).

[151] Jorge Beltrán, Carlos Guindel, and Fernando García. «Automatic extrinsic calibration method for lidar and camera sensor setups.» In: *arXiv preprint arXiv:2101.04431* (2021).

[152] Zheng Gong, Chenglu Wen, Cheng Wang, and Jonathan Li. «A target-free automatic self-calibration approach for multibeam laser scanners.» In: *IEEE Transactions on Instrumentation and Measurement* 67.1 (2017), pp. 238–240.

[153] Paolo Medici. «Pin-Hole Camera Reference Frame and Calibration Techniques.» In: (2011).

[154] Lee A Luft, Larry Anderson, and Frank Cassidy. «Nmea 2000 a digital interface for the 21st century.» In: *Proceedings of the 2002 National Technical Meeting of The Institute of Navigation*. 2002, pp. 796–807.

[155] Carlos Guindel, David Martin, and Jose Maria Armingol. «Fast joint object detection and viewpoint estimation for traffic scene understanding.» In: *IEEE Intelligent Transportation Systems Magazine* 10.4 (2018), pp. 74–86.

[156] Zequn Qin, Huanyu Wang, and Xi Li. «Ultra Fast Structure-aware Deep Lane Detection.» In: *The European Conference on Computer Vision (ECCV)*. 2020.

[157] Moritz Werling, Julius Ziegler, Sören Kammel, and Sebastian Thrun. «Optimal trajectory generation for dynamic street scenarios in a frenet frame.» In: *2010 IEEE International Conference on Robotics and Automation*. IEEE. 2010, pp. 987–993.

[158] Umit Ozguner, KA Unyelioglu, and C Hatipoglu. «An analytical study of vehicle steering control.» In: *Proceedings of International Conference on Control Applications*. IEEE. 1995, pp. 125–130.

[159] CARLA. *ROS/ROS2 bridge for CARLA simulator*. 2021. URL: https://github.com/carla-simulator/ros-bridge (visited on 01/20/2022).

[160] Harwin Saptoadi. «Suitable deceleration rates for environmental friendly city driving.» In: *International Journal of Research in Chemical, Metallurgical and Civil Engineering* 4.1 (2017), pp. 2–5.

[161] Georg Weber, D Dettmering, and H Gebhard. «Networked transport of RTCM via internet protocol (NTRIP).» In: *A Window on the Future of Geodesy*. Springer, 2005, pp. 60–64.

[162] Richard B Langley. «Rtk gps.» In: *Gps World* 9.9 (1998), pp. 70–76.

[163] K Sheridan, P Toor, D Russell, C Rocken, and L Mervart. «TerraStar-C: A Global GNSS Service for cm Level Precise Point Positioning with Ambiguity Resolution.» In: *European Navigation Conference*. 2015.

[164] Eric A Wan and Rudolph Van Der Merwe. «The unscented Kalman filter for nonlinear estimation.» In: *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373)*. Ieee. 2000, pp. 153–158.

[165] T. Moore and D. Stouch. «A Generalized Extended Kalman Filter Implementation for the Robot Operating System.» In: *Proceedings of the 13th International Conference on Intelligent Autonomous Systems (IAS-13)*. Springer, July 2014.

[166] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. «Improved techniques for grid mapping with rao-blackwellized particle filters.» In: *IEEE transactions on Robotics* 23.1 (2007), pp. 34–46.

[167] Wolfgang Hess, Damon Kohler, Holger Rapp, and Daniel Andor. «Real-time loop closure in 2D LIDAR SLAM.» In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2016, pp. 1271–1278.

[168] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. Vol. 1. MIT press Cambridge, 2000.

[169] Wolfram Burgard, Cyrill Stachniss, Maren Bennewitz, Giorgio Grisetti, and Kai Arras. «Introduction to mobile robotics.» In: *Probalistic Sensor Models. University of Freiburg* (2011), pp. 2–3.

[170] MS Milošević and MČ Stefanović. «Performance Loss Due to Atmospheric Noise and Noisy Carrier Reference Signal in Qpsk Communication Systems.» In: *Elektronika ir Elektrotechnika* 58.2 (2005).

[171] Diederik P Kingma and Max Welling. «Auto-encoding variational bayes.» In: *arXiv preprint arXiv:1312.6114* (2013).

[172] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. «LSTM: A search space odyssey.» In: *IEEE transactions on neural networks and learning systems* 28.10 (2016), pp. 2222–2232.

[173] Chaochao Yan, Sheng Wang, Jinyu Yang, Tingyang Xu, and Junzhou Huang. «Re-balancing variational autoencoder loss for molecule sequence generation.» In: *Proceedings of the 11th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics*. 2020, pp. 1–7.

[174] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. «Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions.» In: *URL http://arxiv. org/abs/0909.4061. oai: arXiv. org* 909 (2009).

[175] D. Tripolone. *Land Rover develops virtual eyes to get others to trust driverless cars*. https://www.news.com.au/technology/innovation/motoring/hitech/land-rover-develops-virtual-eyes-to-get-others-to-trust-driverless-cars/news-story/5c86ce7b1c5edd04be85eb7a6af7c7fe. 2018.

[176] D Gomez, P Marin-Plaza, Ahmed Hussein, A Escalera, and JM Armingol. «Ros-based architecture for autonomous intelligent campus automobile (icab).» In: *UNED Plasencia Revista de Investigacion Universitaria* 12 (2016), pp. 257–272.