Asistente inteligente para la conducción autónoma

Alberto González Isorna

Universidad Internacional de la Rioja, Logroño (España)

17/02/2020

Resumen

En la actualidad, la conducción autónoma está revolucionando el sector automovilístico de una forma drástica. Por un lado, creando nuevos paradigmas de conducción y por otro lado implicando una gran inversión de tecnología. Uno de los problemas del coche autónomo es la pérdida de control en las decisiones del vehículo por parte de los pasajeros.

El objetivo de este presente proyecto es crear una interfaz hombre-máquina que sea capaz de comunicarse mediante técnicas de inteligencia artificial y modificar la conducta del coche autónomo conforme a ello.

Para ello abordaremos las tareas de reconocimiento facial y de gestos, a continuación, la construcción de una interfaz gráfica y por último la integración de esto a un entorno de simulación de conducción autónoma. Veremos que, podemos hacer que el vehículo se comunique con la interfaz gráfica y sobre todo que con los gestos o caras del usuario podemos modificar satisfactoriamente la conducta del coche sin necesidad de ninguna interacción física.

Palabras Clave

Vehículo autónomo, interfaz hombre-máquina, reconocimiento facial, reconocimiento de gestos, simulador de conducción.

I. Introducción

En la década a venir, los vehículos autónomos circularán por nuestras calles y van a cambiar por completo la movilidad y el transporte que conocemos hoy día: perderíamos el sentido de la figura del conductor, la propiedad de un coche tendería a ser algo más comunitario en vez de personal y se reduciría el tráfico y los accidentes [1].

En este contexto la interacción entre el humano y el vehículo y entre el vehículo y el entorno es fundamental. Necesitamos que el vehículo sea capaz de reconocernos, adaptarse y aprender de nosotros. Actualmente no hay interfaces hombre-maquina (HMI) que sean adaptables a las necesidades del futuro de la conducción autónoma.

El riesgo de separar totalmente conducción de pasajeros se traduciría en una pérdida total de la libertad del pasajero, ya que el pasajero se convertiría en un mero sirviente del vehículo, no podría cambiar de ruta o manejar que es lo que quiere hacer con su coche. Igualmente, en cuanto a confort ya que si es el coche el que toma todas las decisiones decidiría por él mismo lo que es mejor sin tener en cuenta a los pasajeros. Es por ello por lo que una interfaz que responda en estas situaciones se hace necesaria.

El objetivo de este proyecto es proporcionar un sistema de asistencia a la conducción autónoma. Una interfaz hombre-maquina dedicada a preservar el confort y la seguridad del pasajero de un vehículo autónomo.

Lo que se propone es realizar un sistema que mediante diferentes vías de percepción sea capaz de interactuar con el pasajero, influyendo en la seguridad y confort de este.

El planteamiento del presente proyecto va a ser bidireccional:

- Por un lado, construir las vías de comunicación inteligentes con la interfaz y que puedan comunicarse con el entorno de simulación. Aquí entraran en juego el reconocimiento de gestos y el reconocimiento facial.
- Por otro lado, construir el SW del simulador que sea capaz de reaccionar adecuadamente a las peticiones del usuario y comunicar a la interfaz gráfica.

Por lo tanto nuestro proyecto se basará en el desarrollo tanto del Software de inteligencia artificial como el simulador y la interfaz gráfica como desarrollaremos a continuación.

En este artículo abordaremos primero un estado del arte para conocer el contexto y que están haciendo los constructores del automóvil actualmente y expondremos nuestra solución. Tras esto, entraremos en el desarrollo de la propuesta y discutiremos los resultados.

II. Estado del Arte

En primer lugar hablaremos del contexto y estado del arte del vehículo autónomo para entender cuales son los contribuyentes y sus soluciones y en segundo lugar hablaremos de que sistemas proponen que sean similares a lo que se trata en el presente proyecto.

Cuando pensamos en vehículos autónomos rápidamente se nos vienen a la cabeza nombres de empresas punteras en automoción como Tesla o BMW, pero la realidad es un poco distinta. Actualmente estas empresas y el resto que las siguen (PSA, Renault, etc.) tienen su foco puesto en la ayuda a la conducción (nivel de autonomía 3-4). Son las empresas de Software e Inteligencia artificial las que están más avanzadas en dicho campo. El objetivo es que por un lado dichas empresas proporcionen un equipo integrado y conectado de vehículo autónomo y las constructoras de automóviles hagan de soporte a dichos medios.

Las empresas que lideran actualmente el sector de conducción autónoma son las siguientes [2]:

- **Waymo:** antes conocida como ‘Google self-driving car project’, es una empresa desarrolladora de vehículos autónomos perteneciente al conglomerado Alphabet Inc. Lleva trabajando en vehículos autónomos desde 2009. Su proyecto se centra en taxis autónomos y han realizado ya varios experimentos con pasajeros.

- **GM CRUISE:** es una compañía estadounidense de automóviles autónomos con sede en San Francisco, California. Cuenta con una flota de 180 vehículos autónomos en pruebas en circulación.
- **Argo AI:** empresa independiente fundada en 2017. Tiene el soporte de constructoras como Ford o Wolkswagen. Está especializada en crear sistemas y soluciones de conducción autónoma adaptables a diferentes vehículos.

Las personas contributivas al desarrollo de esta tecnología son numerosas, pero cabe destacar una mención especial a las dos personas siguientes:

- **Chris Urmson** es un ingeniero conocido por su trabajo pionero en la tecnología de la conducción autónoma [3]. Trabajó con Alphabet en su proyecto de vehículo autónomo Waymo y es CEO de la empresa emergente Aurora Innovation.
- **Sebastian Thrun,** (nacido el 14 de mayo de 1967) es un empresario, educador e informático alemán. Es CEO de Kitty Hawk Corporation, y presidente y cofundador de Udacity. Antes de eso, adjunto de Google, profesor de ciencias de la computación en la Universidad de Stanford, y antes de eso en la Universidad Carnegie Mellon. En Google, fundó Google X y el equipo de coches autónomos de Google. También es profesor adjunto en la Universidad de Stanford y en Georgia Tech. Se le considera uno de los mayores exponentes de divulgación de inteligencia artificial y vehículo autónomo [4].

Una vez visto el estado del arte de las tecnologías de conducción autónoma, nos centraremos en las tecnologías propuestas que sean similares a la del presente proyecto. Hablaremos separándolo por empresas, ya que en cuestión de fechas es algo relativamente nuevo y sobre todo que el presente proyecto está centrado en un entorno que todavía no existe como tal hoy en día.

Ford: Ford ha realizado numerosos experimentos de interacción del coche autónomo con el usuario, pero centrándose en cómo se puede comunicar el coche con el exterior. Un escenario típico es un coche autónomo que llega a un paso de cebra. El peatón, que busca la mirada del conductor no la encuentra y puede generar desconfianza. Es por ello por lo que están probando una interfaz externa simple para indicar al peatón que ha sido visto y que puede pasar [7].

Tesla y Waymo: para los dos gigantes del vehículo autónomo se centran más en la practicidad actual de la interfaz. Dado que sus vehículos ya circulan por las calles, la interfaz se ajusta a la disposición de un coche actual y las interfaces habituales. Para ambos constructores, toda la interacción se realiza por forma de comandos táctiles, algunos comandos de voz y desde el dispositivo móvil [8].

Hyundai: la marca coreana es la que más se está introduciendo en el mundo de las interfaces para vehículos autónomos. Han hecho en los años 2018 y 2019 dos propuestas sobre este tema.

- Por un lado, en 2018, presentaron su sistema *Hyundai Intelligent Personal Cockpit*, que integra sistemas de reconocimiento facial y gestos para ofrecer al usuario una interacción más personalizada, aunque todavía en el contexto de conducción manual [9].
- Por otro lado, en 2019, presentaron su sistema *Hyundai Virtual Touch*. Este sistema consiste en manejar toda la interfaz usuario del coche a través de gestos, sin necesidad alguna de contacto físico [10].

Eyesight: cabe mencionar la empresa eyesight que se presentó en el CES 2018 igualmente, son expertos en electrónica embebida y reconocimiento facial, pero no integran interacción con los gestos.

Apple: por último, Apple, que en principio no tiene relación con el mundo automovilístico, tiene una patente para desbloquear el coche con reconocimiento facial [11]. Algo que es curioso, ya que Apple explota este sistema de reconocimiento facial para el desbloqueo de sus móviles y no es descabellado el acercamiento a otros sectores. Además, cabe mencionar que en el futuro, smartphone y vehículo estarán totalmente conectados.

Como hemos visto, la mayoría de las empresas se acercan por una vertiente u otra al mundo de las interfaces gráficas para vehículos autónomos. Sin embargo, no hemos visto ninguna que integre reconocimiento facial y de gestos al mismo tiempo y se desarrolle en un entorno de conducción autónoma, ya sea real o simulado. Por eso, el objeto del presente proyecto es acercar un poco más si cabe el mundo de las interfaces gráficas a un entorno de conducción autónoma.

III. Objetivos y Metodología

El objetivo general del presente proyecto es el siguiente: **"Diseñar el software de los sistemas de percepción y actuación para una interfaz hombre-máquina de un coche autónomo de nivel 4 y 5."**

Los objetivos específicos de dicho sistema son los siguientes:

1. El sistema debe ser capaz de incluir un análisis de los gestos del usuario y el reconocimiento facial a tiempo real, siguiendo el movimiento tanto de manos como cara.

2. El sistema debe ser capaz de reconocer a los pasajeros del vehículo y modificar su comportamiento acorde a ello.

3. El sistema debe anticipar en la medida de lo posible los modos de conducción que desea el conductor, tanto de desactivación y activación.

4. El sistema debe señalar el modo de conducción actual.

5. El conductor podrá tomar ciertas decisiones sobre la dinámica del vehículo, ya sea por motivos de propia voluntad del pasajero o emergencia, para ello se apoyará en los gestos predefinidos por el usuario.

6. El sistema deberá incorporar una interfaz gráfica que informe a tiempo real el estado del vehículo y las predicciones tanto facial como gestuales.

Para la metodología hemos utilizado dos técnicas:

- A principio de proyecto, un diagrama de Gantt general para ver todos los objetivos a abordar en función del tiempo. Lo hemos utilizado como base para tener una visión global y rápida de cómo nos vamos a organizar.

- A lo largo del proyecto, hemos elegido la metodología Scrum-Agile para organizar los plazos y entregas intermedias del desarrollo de SW.

Dividiremos el proyecto en cuatro etapas importantes:

1. Reconocimiento facial y elección del simulador
2. Reconocimiento de gestos
3. Implementación en tiempo real
4. Interfaz gráfica y simulador de conducción

A continuación abordaremos como se ha resuelto las distintas etapas y cuales han sido las distintas aportaciones de cada una.

IV. CONTRIBUCIÓN

Reconocimiento facial y elección del entorno de simulación

La primera tarea que abordar es el reconocimiento facial. Este permitirá cumplir parte de nuestros objetivos para el sistema de ayuda a la conducción, ya que permite reconocer al ocupante del vehículo y adaptar la interfaz y la conducción al pasajero en cuestión

La implementación del reconocimiento facial en el presente proyecto se realiza mediante técnicas de inteligencia artificial. En concreto nos basamos en técnicas de redes convolucionales.

A continuación, mostramos que herramientas se han utilizado para cada etapa de la tarea, desde el procesado de imagen a su clasificación:

- Para el procesado de imágenes, se ha elegido la herramienta **OpenCV**.
- Para la extracción de características en primer lugar se utiliza una CNN propia y posteriormente se ha utilizado el modelo **FaceNet**.
- Para la clasificación de imágenes se ha elegido un modelo de **VNN**, es decir, redes neuronales totalmente conectadas.

Acto seguido se detalla cada uno de los puntos citados anteriormente.

OpenCV es una biblioteca de código libre lanzada por Intel en el año 2000 cuya función es la extracción y el procesamiento de datos a través de imágenes. Las funciones principales de dicha herramienta que nos servimos en nuestro Software son las siguientes:

- Conversión de color: blanco a negro, por ejemplo.
- Ajuste de tamaño de la imagen: para ajustar a los modelos creados.
- Detección de regiones: localización del rostro.
- Lectura de imágenes o escritura en imágenes: para leer la imagen o escribir texto o formas en ella.

FaceNet es un modelo desarrollado por Google en 2015 con gran éxito en la tarea de reconocimiento facial y en concreto extracción de características.

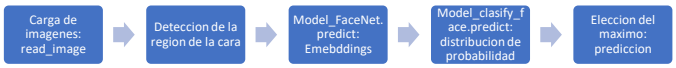
FaceNet proporciona una integración unificada para tareas de reconocimiento facial, verificación y agrupación. Mapea cada imagen de la cara en un espacio euclidiano de modo que las distancias en ese espacio corresponden a la similitud de la cara, es decir, una imagen de la persona “A” se colocará más cerca de todas las otras imágenes de la persona “A” en comparación con las imágenes de cualquier otra persona presente en el conjunto de datos. El tamaño de dicho espacio es de 128 dimensiones, es decir 128 características de una cara dada una imagen.

Una vez obtenido el vector de 128 características se entrena una red neuronal para la clasificación. Hemos elegido una arquitectura totalmente conectada, Vainilla Neuronal Network o **VNN**.

Nuestro modelo tiene las siguientes características:

- Dataset de entrenamiento: Obtenido desde la base de datos Keagle [15].
- Entrada: Vector de 128 características obtenidas de FaceNet.
- Capa1: capa densa de 300 neuronas y dropout a la salida
- Capa2: capa densa de 200 neuronas y dropout a la salida
- Capa3: capa densa de 100 neuronas y dropout a la salida
- Capa de salida: 6 neuronas, capa softmax.
- Función de activación: ReLu

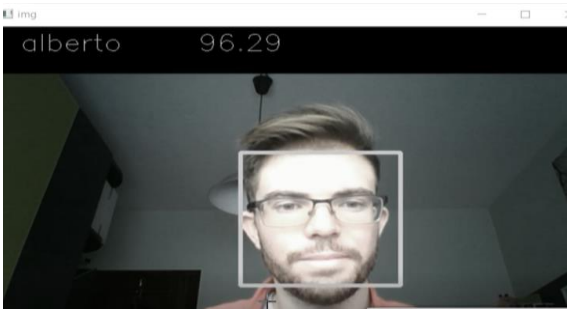
Una vez tenemos todas las herramientas seguimos el siguiente procedimiento para el análisis de rostros:



Primeramente, cargamos y procesamos la imagen, detectamos donde se encuentra la cara, aplicamos el modelo FaceNet para extraer las características y aplicamos por último nuestro modelo entrenado para extraer las categorías. Las categorías son las siguientes:

['alberto', 'ben_afflek', 'elton_john', 'jerry_seinfeld', 'madonna', 'mindy_kaling']

A continuación, vemos una imagen del resultado de la predicción para la clase Alberto:



Vemos como predice la clase adecuadamente y con una alta precisión.

Elección del simulador

Para la elección del simulador tenemos que tener en cuenta que sea un modelo de simulador de dinamica del vehiculo y que pueda integrarse a nuestro proyecto, por lo que debe estar escrito en Python. Observando los requisitos expuestos en la memoria el más apropiado para el presente proyecto era el simulador **Carla**. Está programado en Python, es de código abierto y hay bastante documentación y videos explicativos.

CARLA (siglas del inglés de: Car Learning to Act, o Coche que Aprende a Actuar, en español) es el producto del trabajo del investigador de Intel Labs Alexey Dosovitskiy y varios investigadores del Toyota Research Institute y del Centro de Visión por Computador de Barcelona (España) [23].

CARLA consiste principalmente en dos módulos:

- Carla_simulator: El simulador realiza la mayor parte del trabajo pesado, controla la lógica, la física y la representación de todos los actores y sensores en la escena.
- Carla_Python_API: es un módulo que puede importar scripts Python, proporciona una interfaz para controlar el simulador y recuperar datos.

Reconocimiento de gestos

Este apartado se focaliza en el desarrollo de Software de reconocimiento de gestos.

Los métodos mas habituales para el reconocimiento de gestos se separan en métodos basados en 3D y en 2D. Dado que nos servimos de una sola cámara nos serviremos de los modelos en 2D.

Dentro de los modelos en 2D hemos utilizado en concreto los Modelos basados en la silueta binaria: Son modelos basados en las propiedades geométricas de la silueta del objeto. Es decir, partimos de la imagen binaria de la mano analizamos su forma.

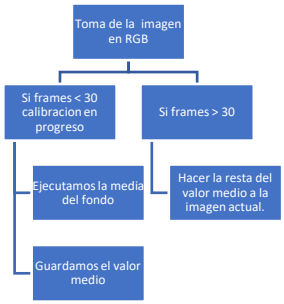
Las clases que hemos aprendido son las presentadas en el dataset [28]:

Para analizar y clasificar los gestos empleamos una Red Neuronal Convolucional o **CNN.** La estructura elegida tras varias pruebas de entrenamiento es la siguiente:

- Capa de entrada: imagen de 224x224x3, imagen binaria que corresponde con la silueta de la mano a clasificar.
- Capas de filtros convolucionales:
  - Convolución 2D, 32 filtros de tamaño 5X5
  - Convolución 2D, 64 filtros de tamaño 3X3
  - Convolución 2D, 64 filtros de tamaño 3X3
- Capas de maxpooling: entre cada capa convolucional se introduce una capa de maxpooling, que reduce la imagen a la mitad.
- VNN: por último, capas totalmente conectadas para la clasificación: 3 capas densas, de 128, 128 y 64 respectivamente.
- Capa de salida: 6 salidas con función sotmax (las correspondientes a cada clase).

Una vez definido el modelo es turno de la fase de entrenamiento, tras varios ajustes y entrenos, obtennos una precisión del 90% con los datos de test.

Uno de los principales problemas cuando analizamos las imágenes reales obtenidas por la cámara es que dichas imágenes no son binarias, por lo que hay que realizar un proceso de binarización. El proceso de binarización consiste en lo siguiente:



Lo que hacemos es realizar una calibración del fondo, obteniendo su valor medio por región y restándolo a la imagen tras la calibración. Esto conlleva como limitación que no podemos poner la mano en la región de interés mientras se realiza la calibración, pero por otro lado los resultados son bastante efectivos.

Los resultados obtenidos son los siguientes:

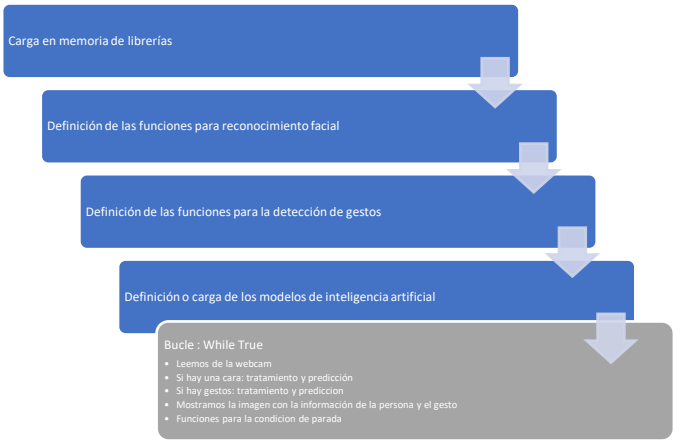
Realizamos la binarización y la predicción de forma correcta.

Implementación en tiempo real

Una vez que ya tenemos la parte de inteligencia artificial pura integrada y hemos exportado los modelos y encapsulado las funciones en clases tanto para la clase de reconocedor facial como de reconocedor de gestos, queda integrarlo todo en tiempo real.

En primer lugar, la solución por la que se opta es acoplar todas las funciones de reconocimiento tanto facial como de gestos que tenemos anteriormente en un programa principal, el cual en un momento dado va a entrar en un bucle infinito en el que va captando imágenes, reconociendo y actualizando.

El esquema de esta primera implementación sería el siguiente:

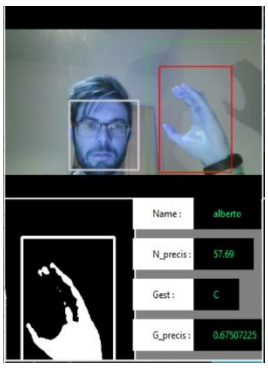


Este método pese a parecer efectivo tiene algunas desventajas:

- No se pueden realizar operaciones en paralelo: una vez que el programa entra en el bucle while, estará ejecutando las mismas instrucciones una y otra vez. Para el propósito de este apartado es suficiente, pero si queremos hacer que interactúe con una interfaz gráfica y el simulador, no es posible.

- La extensión es muy larga: el hecho de integrar todas las funciones y definiciones en el mismo código en el que se ejecuta el bucle hace el programa demasiado largo y no entendible. En esta primera versión se llegan a las 500 líneas de código.

Los resultados obtenidos con este método son los siguientes:



Como podemos ver en la imagen de la izquierda el sistema responde adecuadamente en tiempo real.

Por un lado, vemos como el cuadrado blanco está ubicado exactamente dónde está nuestra cara. La clase detectada la podemos ver más abajo, es Alberto. Tenemos una precisión del 60% debido en este caso a condiciones de iluminación.

Por otro lado, en la imagen superior en la parte derecha, vemos como la mano ha sido satisfactoriamente detectada (recuadro rojo) dentro de la región de interés, el recuadro verde.

En este caso tenemos la detección de la clase C lo que es correcto si nos fijamos en la forma de la mano y una precisión de entrono el 70%.

Interfaz gráfica

Para la interfaz gráficas se efectua una selección de cuales son las mas adecuadas para este pryecto, debido a su prototipado rápido y su base nativa en Python, se decide utilizar Tkinter.

Lo que hacemos es realizar una pantalla principal o window y dividirla en diferentes partes o frames, cada una de ellas tendrá un cometido dentro de la interfaz.

Frame 1: información de bienvenida, detección de rostro y de gestos. Texto sobre la detección de gestos e información sobre el modo de conducción actual.

Frame 2: navegación y estado del vehículo. En esta parte se incluyen descriptivos de la ruta y el estado del vehículo.

Frame 3: menú e información extra. Será el panel con información extra, como la música, el tiempo, etc.

Para llevar a cabo la implementación se crea una clase propia que va a ser como el lienzo donde pongamos todas las imágenes, botones y funciones.

Tanto la ventana principal como las subventanas o frames están incluidas en la inicialización de esta clase gráfica, denominada **mbackground**(). Tras montar los distintos elementos el resultado es el siguiente:

En la columna de la izquierda vemos efectivamente el primer frame diferenciado, con el mensaje de bienvenida, el texto para el nombre, la precisión los gestos y su precisión y debajo una columna con los modos de conducción. Los huecos que quedan en gris se deben a que todavía no se ha integrado todo el módulo de inteligencia artificial en tiempo real. A continuación, en el frame central vemos el entorno de navegación, en la parte superior la ruta y la parte inferior el modo de conducción. En el frame de la derecha podemos ver el menú y otros datos interesantes del vehículo.

Implementación del simulador

Para el simulador hemos partido de un ejemplo prácticamente vacío que simplemente hace una llamada a la librería PyGame para representar lo que se ve en el mundo de CARLA.

A continuación, partimos de ese mapa y colocamos un vehículo en una posición determinada, le asignamos una cámara flotante para observarlo y hacemos que el vehículo se mueva.

Por último, ponemos una dinámica al vehículo y lo ponemos a andar. Las imágenes que ve la cámara, que son las del coche, se pasan el módulo PyGame en otra ventana. El resultado es el presentado a la derecha.

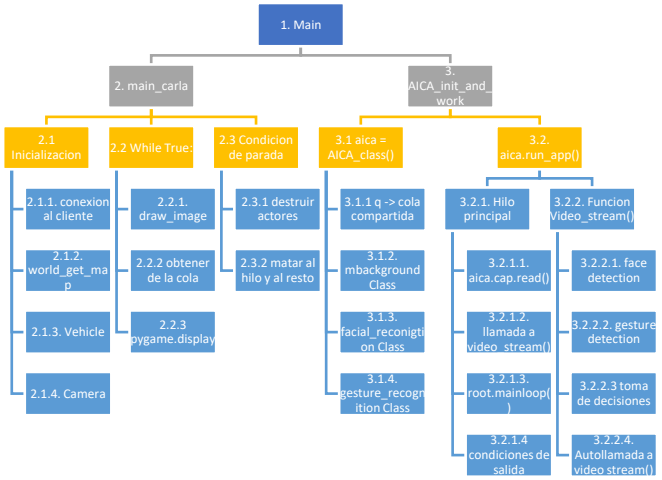
La clase vehicle state

Esta clase contiene la información de la dinámica y estado del vehículo. Es el objeto que hace de nexo de unión tanto entre el simulador y la interfaz gráfica con los módulos de inteligencia artificial.

Implementación final

Para la implementación final lo que se hace es crear dos hilos, que van a compartir la clase vehicle_state y se van a ejecutar en paralelo. El primero de ellos va a integrar la interfaz gráfica con los módulos de inteligencia artificial y el segundo va a integrar el simulador.

A continuación, podemos ver el gráfico de la implementación final, cuantos más niveles bajamos significa que son funciones, métodos o clases que están dentro de la clase superior. Las distintas ramas indican ejecución en paralelo.



Si observamos el gráfico vemos como el programa principal se subdivide en dos hilos: main_carla y AICA.

Main_carla va a ejecutar el simulador. En un primer lugar va a inicializar todas las variables (2.1), en segundo lugar, va a entrar en el bucle infinito (2.2). En este, lo que hacemos es obtener información del mundo actual, modificar comportamientos y plasmarlos de nuevo en la interfaz de simulador, es la actuación de un agente inteligente. Por último, va a tener también condiciones de parada segura.

AICA va a ejecutar toda la interfaz grafica y el proceso de inteligencia y comunicación.

Por un lado, se carga la inicialización de la clase (3.1) con la cola compartida, la interfaz y los módulos de reconocimiento.

Por otro lado se va ejecutando las tareas de reconocimiento y toma de decisiones con el método de la clase run_app() (3.2).

A su vez el método run_app tiene un hilo principal (3.2.1) en el que se lee de la webcam, se llama a la función de inteligencia y se actualiza la interfaz con los resultados y por otro lado la función video_stream() (3.2.2) que realiza las funciones de reconocimiento y toma de decisiones del vehículo.

A continuación, el resultado realizando un gesto en concreto “C” que activa el modo de conducción autónoma:

Si realizamos el gesto adecuado, vemos como se ilumina la interfaz de verde y se pone el nivel de conducción en nivel 5 indicando que hemos activado el modo autónomo.

V. Resultados

En esta sección se analiza si la herramienta desarrollada ha cumplido satisfactoriamente los requisitos. A continuación, expondremos una lista de los requisitos logrados y al final comentaremos los resultados.

Requisitos funcionales:

- El sistema ha sido capaz de reconocer a los ocupantes del vehículo, dado un conjunto de datos (RF_01)
- El sistema ha conseguido hacer una transición de modo de conducción únicamente con la utilización de gestos (RF_02).
- Se ha construido satisfactoriamente una interfaz en la que se muestran los modos de conducción (RF_03) y además los datos tanto del vehículo como del pasajero se actualizan en tiempo real en la interfaz (RF_04, RF_05).
- Los módulos de reconocimiento facial y de gestos se han integrado satisfactoriamente con la interfaz gráfica (RF_06).
- El sistema es capaz de comunicarse con el estado del vehículo (RF_07).

Requisitos no funcionales:

- El sistema tiene bien definidas las condiciones de parada y finalización de los hilos como explicado en los apartados 6.3 y 6.4 (RNF_01).
- Podemos ejecutar todo el sistema simplemente arrancando el simulador y ejecutando el programa principal (RNF_02).
- El entorno de simulación contiene al vehículo en primera persona (RNF_03).
- El sistema es capaz de ejecutar todos los hilos a tiempo real, con un ciclo de 20 ms cada hilo, menor que los 30ms establecidos como límite (RNF_04).
- El sistema está compuesto en distintas clases diferenciadas y explicadas tanto en el desarrollo como en el diseño de SW (RNF_05). Además, las clases están estructuradas por función como se ha desarrollado en la arquitectura (RNF_06).
- Dado a la estructura de clases y la flexibilidad de vehicle state, podemos aumentar fácilmente el sistema si queremos introducir varios reconocedores o nuevas funcionalidades (RNF_07, RNF_08).

En resumen, tanto los requisitos funcionales como no funcionales han sido logrados y descritos en la presente memoria.

A continuación, discutiremos sobre los resultados en relación con los objetivos generales y específicos.

VI. Discusión o Análisis de Resultados

Primeramente, recordemos el objetivo general del proyecto:

"Diseñar el software de los sistemas de percepción y actuación para una interfaz hombre-máquina de un coche autónomo de nivel 4 y 5."

Como hemos visto a lo largo de la memoria, hemos creado en los primeros apartados los sistemas de percepción y actuación, a continuación, la interfaz hombre-máquina y por último el entorno de simulación autónoma. Por lo tanto, el objetivo general se considera cumplido satisfactoriamente.

Seguidamente, se aborda si el presente proyecto ha cumplido con los objetivos específicos:

1. El sistema es capaz de incluir un análisis de los gestos del usuario y el reconocimiento facial a tiempo real, siguiendo el movimiento tanto de manos como cara.

2. El sistema es capaz de reconocer a los pasajeros del vehículo y modificar su comportamiento acorde a ello, hemos visto cómo nos indica la persona a bordo según la cara detectada.

3. El sistema anticipa en la medida de lo posible los modos de conducción que desea el conductor, tanto de desactivación y activación, hemos visto que con el gesto "c" podemos activar o desactivar el piloto automático.

4. El sistema señala el modo de conducción actual, hemos visto en el apartado 6.4 como cambia la interfaz según el modo de conducción.

5. El conductor puede tomar ciertas decisiones sobre la dinámica del vehículo, ya sea por motivos de propia voluntad del pasajero o emergencia, para ello se apoyará en los gestos predefinidos por el usuario.

6. El sistema incorporar una interfaz gráfica que informe a tiempo real el estado del vehículo y las predicciones tanto facial como gestuales.

El Software ha cumplido satisfactoriamente con todos los objetivos propuesto al inicio del proyecto satisfactoriamente.

A continuación, hablaremos de las conclusiones del proyecto y a raíz de algunas limitaciones o posibles mejoras hemos construido los posibles trabajos futuros que veremos al final.

VII. Conclusiones

En este trabajo hemos visto y abordados temas muy distintos en el mundo de la ingeniería. Por un lado la inteligencia artificial, por otro las interfaces de usuario, y por otro el sector automovilístico y la conducción autónoma. El mercado actual del automóvil se está sofisticando tanto que en el futuro la tecnología o las capacidades de cómputo de un coche autónomo superen con creces los ordenadores que tenemos hoy en día.

Hemos visto como el reconocimiento facial es un ámbito bastante desarrollado y hemos aportado un sistema de seguimiento, reconocimiento y clasificación fácilmente encapsulable en una clase, aportando mucha flexibilidad a un reconocedor facial.

Se ha observado igualmente que la misma tarea para el reconocimiento de gestos no es tan evidente, ya que por un lado no está en el foco de la inteligencia artificial, y los gestos son muchos mas dinámicos que las expresiones faciales. A pesar de todo hemos conseguido discernir entre una clase predefinida de gestos.

Igualmente cabe mencionar que la integración en tiempo real ha aportado numerosos contratiempos y problemas, ya que no era posible ejecutar códigos distintos en un bucle infinito. Esto ha llevado a implementar una solución bastante avanzada en computación con la creación de los hilos y la compartición por colas.

Por último, cabe mencionar la integración del simulador y como hemos podido comprobar el objeto principal del proyecto, que no es ni más ni menos que ser capaz de interactuar con un coche autónomo, sin ningún tipo de contacto físico.

Trabajos futuros

En cuanto a trabajos futuros o líneas de mejora, cabe destacar los siguientes puntos:

•Widgets dinámicos: una posible mejora sería la integración de iconos dinámicos que se modificaran según la información del vehículo, por ejemplo, la barra de batería o combustible restante.

•Control por voz: para una experiencia completa de control a distancia, se puede añadir módulos de reconocimiento y control por voz, tal y como incorporan algunos coches actualmente.

•Adición de nuevas caras o gestos: una característica interesante a incluir es la posibilidad de añadir nuevos gestos o caras de forma dinámica.

REFERENCIAS

[1] Daimler, «Safety First for Automated Driving,» 2019.

[2] S.A.E, «sae.org,» 2018. [En línea]. Available: https://www.sae.org/news/press-room/2018/12/sae-international-releases-updated-visual-chart-for-its-%E2%80%9Clevels-of-driving-automation%E2%80%9D-standard-for-self-driving-vehicles.

[3] Zdnet, «Zdnet,» Noviembre 2019. [En línea]. Available: https://www.zdnet.com/article/the-top-3-companies-in-autonomous-vehicles-and-self-driving-cars/.

[4] Forbes, «Chris Urmson Reflects On Challenges,» 2017. [En línea]. Available: https://www.forbes.com/.

[5] «"4 - Sebastian Thrun". The FP Top 100 Global Thinkers,» 2012. [En línea]. Available: http://www.foreignpolicy.com/.

[6] National Geographic, «National Geographic,» 2018, [En línea]. Available: https://www.nationalgeographic.com.es/ciencia/actualidad/futuro-coches-autonomos-y-conectados_13619.

[7] Ford, «FORD MEDIA CENTER,» 2019. [En línea]. Available: https://media.ford.com/content/fordmedia/feu/de/en/news/2019/02/06/ford-tests-light-based-visual-language-that-could-help-autonomou.html.

[8] T. Stevens, «CNET,» 2017. [En línea]. Available: https://www.cnet.com/roadshow/news/how-waymo-is-defining-the-ui-of-autonomy/.

[9] Hyundai Tech, 2018. [En línea]. Available: https://tech.hyundaimotorgroup.com/video/tech-film-_-v-touch-ces-2019-kia/.

[10] T. H. Jr., «www.cnbc.com,» 2019. [En línea]. Available: https://www.cnbc.com/2019/01/10/at-ces-2019-self-driving-cars-with-vr-and-facial-recognition.html.

[11] D. Robitzski, «futurism.com,» 2019. [En línea]. Available: https://futurism.com/apple-facial-recognition-car.

[12] P. C. e. al., Documenting Software Architectures – Views and Beyond, Addison-Wesley.

[13] A. Ollero, Robótica. Manipuladores y robots móviles, Marcombo-Boixareu, 2001.

[14] 2020 Lucid Software Inc., «lucidchart.com,» [En línea]. Available: https://www.lucidchart.com/pages/fr/quest-ce-que-le-langage-de-modelisation-unifie.

[15] R. G. Hernández, «Estudio de técnicas de reconocimiento facial,» Barcelona, 2010

[16] U. I. d. L. R. (UNIR), «Convolutional Neural Networks (CNN),» de Sistemas Cognitivos Artificiales, UNIR, 2019.

[17] S. O. Krzysztof SIWEK, «Deep neural networks and classical approach to face recognition – comparative analysis,» Warsaw University of Technology, Warsaw , 2018.

[18] Kaggle inc, «www.kaggle.com,» 2017. [En línea]. Available: https://www.kaggle.com/c/facial-keypoints-detection/data.

[19] D. Karunakaran, «Medium,» 2018. [En línea]. Available: https://medium.com/intro-to-artificial-intelligence/one-shot-learning-explained-using-facenet-dff5ad52bd38.

[20] Google Inc., «Going deeper with convolutions,» Cornell University, 2014.

[21] Kaggle, «www.kaggle.com,» [En línea]. Available: https://www.kaggle.com/dansbecker/5-celebrity-faces-dataset.

[22] F. Rosique, «A Systematic Review of Perception System and Simulators for Autonomous Vehicles Research,» Universidad Politécnica de Cartagena, 2019.

[23] Emerging Technology From The Arxiv, «MIT Technology Review,» 28 Noviembre 2017. [En línea]. Available: https://www.technologyreview.es/s/9770/gratis-y-con-sello-espanol-asi-es-carla-el-simulador-de-conduccion-autonoma-mas-completo.

[24] K. S. a. J. Sutherland, «The Scrum Guide,» Creative Commons, 2016.

[25] L. F. Ronchetti, «Reconocimiento de gestos dinámicos,» UNIVERSIDAD NACIONAL DE LA PLATA, 2016.

[26] F. J. Andrade, «Un enfoque inteligente para el reconocimiento de gestos manuales,» Universidad Nacional del Centro de la Provincia de Buenos Aires, Buenos Aires, 2016.

[27] O. B. J.-E. V. e. D. C. Sébastien Marcel, «www-prima.inrialpes.fr,» FG'2000 Conference on Automatic Face and Gesture Recognition, 2000. [En línea]. Available: https://www-prima.inrialpes.fr/FGnet/data/10-Gesture/gestures/main.html.

[28] B. Heintz, «github.com,» Diciembre 2018. [En línea]. Available: https://github.com/athena15/project_kojak.

[29] Dpto. Ingeniería Electrica, Electrónica y Control - UNED, «Sistemas en Tiempo Real,» de Comunicaciones industriales y en Tiempo Real, UNED, 2016, pp. 3-10.

[30] Universidad Carlos III de Madrid, «Hilos y Procesos,» de SISTEMAS OPERATIVOS, Madrid, p. 58.

[31] DEV Community, «dev.to,» Agosto 2019. [En línea]. Available: https://dev.to/amigosmaker/pyqt-vs-tkinter-spanish-2n4k.