UNIVERSIDAD NACIONAL DE SAN ANTONIO ABAD DEL CUSCO

FACULTAD DE INGENIERÍA ELÉCTRICA, ELECTRÓNICA, INFORMÁTICA Y MECÁNICA

ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA

TESIS

DESARROLLO E IMPLEMENTACIÓN EN UN SISTEMA EMBEBIDO DE UN SISTEMA DETECTOR DE LA DISTRACCIÓN EN CONDUCTORES MEDIANTE LA ESTIMACIÓN DE LA POSTURA DE LA CABEZA

PRESENTADO POR:

Br. DIEGO ARMANDO UMAN FLORES

PARA OPTAR AL TÍTULO PROFESIONAL DE INGENIERO ELECTRÓNICO

ASESOR:

Dr. Ing. FACUNDO PALOMINO QUISPE

CUSCO - PERÚ

2024

INFORME DE ORIGINALIDAD

(Aprobado por Resolución Nro.CU-303-2020-UNSAAC)

El que suscribe, **Asesor** del trabajo deinvestigación/tesis titulada:.....
DESARROLLO E IMPLEMENTACIÓN EN UN SISTEMA EMBEBIDO DE UN SISTEMA DETECTOR DE LA DISTRACCIÓN EN CONDUCTORES MEDIANTE LA ESTIMACIÓN DE LA POSTURA DE LA CABEZA

presentado por: DIEGO ARMANDO UMAN FLORES con DNI Nro.: 77145233 presentado por: ..... con DNI Nro.: ..... para optar el título profesional/grado académico de INGENIERO ELECTRÓNICO

.....

Informo que el trabajo de investigación ha sido sometido a revisión por 03 veces, mediante el Software Antiplagio, conforme al Art. 6° del ***Reglamento para Uso de Sistema Antiplagio de la UNSAAC*** y de la evaluación de originalidad se tiene un porcentaje de 8 **%.**

Evaluación y acciones del reporte de coincidencia para trabajos de investigación conducentes a grado académico o título profesional, tesis

Porcentaje	Evaluación y Acciones	Marque con una (X)
Del 1 al 10%	No se considera plagio.	X
Del 11 al 30 %	Devolver al usuario para las correcciones.	
Mayor a 31%	El responsable de la revisión del documento emite un informe al inmediato jerárquico, quien a su vez eleva el informe a la autoridad académica para que tome las acciones correspondientes. Sin perjuicio de las sanciones administrativas que correspondan de acuerdo a Ley.	

Por tanto, en mi condición de asesor, firmo el presente informe en señal de conformidad y **adjunto** la primera página del reporte del Sistema Antiplagio.

Cusco, 07 de ENERO de 2025

.....
Firma

Post firma FACUNDO PALOMINO QUISPE

Nro. de DNI 00435194

ORCID del Asesor 0000-0002-5947-6682

Se adjunta:

1. Reporte generado por el Sistema Antiplagio.
2. Enlace del Reporte Generado por el Sistema Antiplagio: **oid:** 27259 : 419538509

Diego Armando Uman Flores

DiegoArmandoUmanFlores_tesis_v1_8_5.pdf

Universidad Nacional San Antonio Abad del Cusco

Detalles del documento

Identificador de la entrega
trn:oid:::27259:419538509

Fecha de entrega
6 ene 2025, 9:21 p.m. GMT-5

Fecha de descarga
6 ene 2025, 9:28 p.m. GMT-5

Nombre de archivo
DiegoArmandoUmanFlores_tesis_v1_8_5.pdf

Tamaño de archivo
3.3 MB

188 Páginas

37,998 Palabras

189,561 Caracteres

8% Similitud general

El total combinado de todas las coincidencias, incluidas las fuentes superpuestas, para ca...

Filtrado desde el informe

- Bibliografía
- Texto citado
- Texto mencionado
- Coincidencias menores (menos de 8 palabras)

Exclusiones

- N.º de coincidencias excluidas

Fuentes principales

6% Fuentes de Internet
2% Publicaciones
5% Trabajos entregados (trabajos del estudiante)

Marcas de integridad

N.º de alerta de integridad para revisión

Caracteres reemplazados
142 caracteres sospechosos en N.º de páginas
Las letras son intercambiadas por caracteres similares de otro alfabeto.

Los algoritmos de nuestro sistema analizan un documento en profundidad para buscar inconsistencias que permitirían distinguirlo de una entrega normal. Si advertimos algo extraño, lo marcamos como una alerta para que pueda revisarlo.

Una marca de alerta no es necesariamente un indicador de problemas. Sin embargo, recomendamos que preste atención y la revise.

Dedicatoria

A mi madre Esperanza y hermanas, por su apoyo incondicional. A mis amigos y asesor, por brindarme motivación y compañía durante esta travesía académica, y a todos los que me apoyaron directa o indirectamente en este desafío.

Agradecimientos

A Dios y a la virgen María, por guiar mi camino y por brindarme fortaleza ante las adversidades que se me presentaron y por haberme rodeado de grandiosas personas durante este desafío.

A mi familia, especialmente a mi madre, gracias por creer en mí y apoyarme en cada etapa de mi vida académica.

A mi asesor Dr. Ing. Facundo Palomino Quispe y a su esposa la Dra. Ana Beatriz, por su orientación, paciencia y apoyo constante durante el desarrollo de esta tesis.

A mis compañeros de investigación y amigos del laboratorio LIECAR, gracias por el apoyo mutuo y por crear un ambiente amigable y estimulante. Sin ustedes, el camino hubiera sido mucho más complicado.

Índice general

Índice de tablas

Índice de figuras

Resumen

La presente investigación tiene como objetivo desarrollar e implementar en un sistema embebido un detector de distracción en conductores de la ciudad del Cusco, basado en la estimación de la postura de la cabeza. La distracción de los conductores en un factor crítico en los accidentes de tránsito, por lo que este sistema aborda esta problemática al detectar cambios en la atención mediante el análisis de la postura de la cabeza. Para ello, se desarrolló un prototipo que utiliza una única cámara y fue implementado en un Orange Pi 5, aprovechando su NPU para ejecutar en tiempo real modelos de redes neuronales convolucionales (CNN), estos estiman los ángulos de rotación de la cabeza (yaw, pitch y roll). Se compararon 12 modelos, seleccionando MNASNet1-0 por su desempeño destacado: un MAE de 3,31°, 119.66 FPS, y un consumo de 0.69 A con el conjunto de datos BIWI, y un MAE de 4,78°, 48.17 FPS y 0.46 A con AFLW2000. Para determinar la distracción, se evaluaron distintas técnicas de detección, se partió de un método basado en umbralización, al cual se le aplicaron modificaciones y adicionalmente se propuso un enfoque alternativo basado en lógica difusa, estas técnicas fueron comparadas para identificar el método con mejores resultados en la detección de distracciones en conductores. En pruebas experimentales realizadas a conductores voluntarios en un entorno real controlado, el prototipo alcanzó una exactitud promedio de 93.58 %, una precisión promedio de 90.05 %, una sensibilidad promedio de 96.5 % y una puntuación F1 promedio de 93.09 %. Estos resultados muestran resultados prometedores en la capacidad del sistema para detectar distracciones en tiempo real.

Palabras clave: Sistema detector de distracción, postura de la cabeza, redes neuronales convolucionales (CNN), técnicas de detección, lógica difusa, Orange Pi 5.

Abstract

This research aims to develop and implement a distraction detection system for drivers in the city of Cusco, based on head posture estimation, in an embedded system. Driver distraction is a critical factor in traffic accidents, and this system addresses this issue by detecting attention changes through the analysis of head posture. A prototype was developed using a single camera and implemented on an Orange Pi 5, taking advantage of its NPU to run convolutional neural network (CNN) models in real-time, which estimate the head's rotation angles (yaw, pitch, and roll). Twelve models were compared, with MNASNet1-0 being selected due to its outstanding performance: a MAE of 3,31°, 119.66 FPS, and a consumption of 0.69 A with the BIWI dataset, and a MAE of 4,78°, 48.17 FPS, and 0.46 A with the AFLW2000 dataset. To determine distraction, various detection techniques were evaluated. The approach began with a thresholding-based method, which was modified, and an alternative fuzzy logic-based approach was proposed. These techniques were compared to identify the one with the best results for detecting driver distractions. In experimental trials with volunteer drivers in a controlled real-world environment, the prototype achieved an average accuracy of 93.58 %, an average precision of 90.05 %, an average sensitivity of 96.5 %, and an average F1 score of 93.09 %. These results show promising potential for the system's ability to detect distractions in real-time.

Keywords: Distraction detection system, head pose estimation, convolutional neural networks (CNN), detection techniques, fuzzy logic, Orange Pi 5.

Introducción

En la actualidad, la distracción al volante es una de las causas principales de accidentes de tránsito atribuidas al factor humano, tanto en el Perú como a nivel global. Este problema contribuye significativamente a la ocurrencia de lesiones y muertes, afectando de manera negativa la seguridad vial. Ante esta situación, surge la oportunidad de desarrollar soluciones que ayuden a mitigar el riesgo y promuevan una mayor atención por parte de los conductores. Si bien es difícil evitar por completo los accidentes, es posible reducir su incidencia a través de estas soluciones. En este contexto, se planteó la necesidad de un sistema de detección que sea robusto y capaz de alertar al conductor ante posibles señales de distracción.

La presente investigación tiene como objetivo desarrollar e implementar en un sistema embebido un detector de distracción en conductores de la ciudad del Cusco, basado en la estimación de la postura de la cabeza. Este sistema tiene el potencial de contribuir a la mejora de la seguridad vial al identificar indicios de distracción al analizar los movimientos de la cabeza del conductor, los cuales reflejan cambios en su atención.

Este trabajo está estructurado de la siguiente manera:

- El capítulo 1 introduce las generalidades de la investigación, incluyendo la formulación del problema, la justificación, los objetivos, variables, así como los alcances y limitaciones del estudio, destacando la relevancia de la detección de distracción en conductores para la seguridad vial.

- El capítulo 2 ofrece el marco teórico necesario para entender los conceptos fundamentales relacionados con el método de estimación de la postura de la cabeza, las redes neuronales

convolucionales (CNN), y las técnicas de detección de distracción.

- El capítulo 3 describe el diseño del sistema detector de distracción, incluyendo los requerimientos de hardware y software para su desarrollo, así como los requisitos para las pruebas en un entorno real. También se detalla la lógica del sistema y se presenta el diagrama de flujo.

- El capítulo 4 detalla la implementación del sistema, abarcando los componentes utilizados, el entrenamiento de las redes neuronales convolucionales (CNN) y su conversión al formato RKNN, así como el proceso de detección de distracción. Se incluyen además las configuraciones realizadas en el Orange Pi 5, y el proceso para el uso de su NPU, así como la implementación de la alarma y el indicador visual. También se describe la fuente de alimentación, el diagrama de conexiones y la instalación del sistema en un vehículo.

- El capítulo 5 presenta las pruebas y resultados de las comparaciones entre diferentes modelos de redes neuronales convolucionales, tanto en una PC como en el Orange Pi 5, con el fin de elegir el más adecuado. Además, se muestran los resultados de la comparación entre distintas técnicas de detección de distracción: el método original basado en umbralización, su versión modificada, y un enfoque alternativo basado en lógica difusa. Se selecciona la técnica de mejor desempeño, cuyos resultados de las pruebas en un entorno real también son mostrados.

- El capítulo 6 ofrece la discusión de los resultados, analizando el rendimiento de los modelos, técnicas y el prototipo del sistema en entorno real.

- Finalmente, se presentan las conclusiones de la investigación y se dan recomendaciones de posibles mejoras, además de sugerir posibles líneas de trabajo futuro.

Capítulo 1

Generalidades

1.1. Formulación del problema

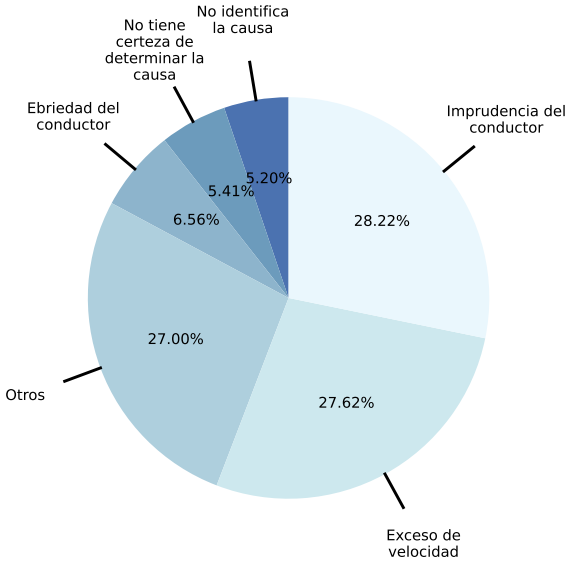
Cada año, aproximadamente 1.3 millones de personas mueren debido a un accidente de tránsito, y millones más sufren lesiones no mortales, que muchas veces provocan una discapacidad (OMS, 2022). A nivel mundial, los accidentes de tránsito representan una de las principales causas de muerte, siendo los niños y jóvenes de entre 5 y 29 años los más vulnerables. Este problema se agrava más en países con rentas bajas y medias, y factores como la velocidad, el consumo de alcohol, la falta de dispositivos de seguridad y la distracción representan un riesgo (OMS, 2022).

Son muchos los factores humanos que influyen en la probabilidad de sufrir algún accidente de tránsito, pero entre ellos, la distracción al momento de conducir es uno de los principales causantes de lesiones y muertes (Michelaraki et al., 2023; Papantoniou et al., 2017). En concreto, la distracción en los conductores impacta de manera negativa en la seguridad vial, y eso crea las oportunidades para desarrollar soluciones personalizadas que reduzcan el riesgo y mejoren el rendimiento de los conductores (Michelaraki et al., 2023).

En el Perú durante el año 2021 se produjeron 74624 casos de accidentes de tránsito relacionados con diversos motivos, donde una de las principales causas fue por imprudencia del conductor (PNP, 2021) (en la Figura 1.1 se visualiza los porcentajes de accidentes de tránsito,

Figura 1.1

Distribución porcentual de causas de accidentes de tránsito.



Nota: Extraído de (PNP, 2021).

donde el 28.22 % se le atribuye a esta causa). Durante el trascurso de ese año, 52 551 personas se vieron afectadas por los accidentes de tránsito, lo que genero un impacto negativo en ámbito social (PNP, 2021).

En el departamento del Cusco se registraron 917 de un total de 2996 casos de accidentes relacionados con la imprudencia de los conductores, ocupando la sexta posición a nivel nacional (PNP, 2021). Y como se definió antes, esta problemática está intrínsecamente ligada a la distracción al volante, que indican que los accidentes de tránsito derivados de la distracción representan un desafío en cuanto a la prevención de accidentes.

Este problema se ve acentuado al existir la carencia de sistemas eficaces de detección de distracción en tiempo real que ayuden a mitigar este problema, especialmente en contextos urbanos como el de la ciudad del Cusco. La falta de estos sistemas no solo limita la prevención de accidentes, sino que también resalta la necesidad de implementar soluciones tecnológicas para mejorar la seguridad vial en el área urbana de Cusco.

Aunque es difícil evitar por completo los accidentes vehiculares, sí es posible reducir significativamente la probabilidad de que ocurran. Es por este motivo que surge la necesidad de

desarrollar un sistema inteligente sólido y fácil, capaz de identificar y alertar a los conductores ante indicios prematuros de distracción con el propósito de prevenir accidentes viales, empleando técnicas de redes neuronales convolucionales y visión computacional en tiempo real, con la idea de no generar incomodidades ni interferir con la visibilidad del conductor.

1.1.1. Problema general

¿Cómo desarrollar e implementar en un sistema embebido un sistema detector de la distracción en conductores mediante la estimación de la postura de la cabeza?

1.1.2. Problemas específicos

- ¿Cómo desarrollar un sistema inteligente que analice la distracción mediante la estimación de la postura de la cabeza con una sola cámara?
- ¿Cómo determinar los ángulos de rotación o de Euler para estimar la postura de la cabeza mediante el uso de una CNN?
- ¿Cómo implementar el sistema de detección de distracción mediante estimación de la postura de la cabeza en un sistema embebido?
- ¿Cómo evaluar el desempeño del sistema inteligente experimental en un entorno real con conductores voluntarios?

1.2. Justificación

1.2.1. Justificación social

Al desarrollar un sistema inteligente basado en redes neuronales convolucionales para detectar la distracción, se busca reducir la tasa de accidentes de tránsito y, por consiguiente,

preservar la vida y la integridad física y mental de los conductores, pasajeros y peatones. Esto tiene un impacto directo en la ciudadanía, al promover un entorno vial más seguro y mejorar la calidad de vida de la sociedad.

1.2.2. Justificación tecnológica

Viendo que la tecnología crece a pasos agigantados, es necesario buscar una solución para las causas de accidentes viales por lo que diseñar e implementar un sistema inteligente basado en redes neuronales convolucionales para detectar la distracción de los conductores representa un avance tecnológico significativo en la ciudad del Cusco.

1.2.3. Justificación académica

La presente investigación en el campo de la detección de distracción en conductores utilizando redes neuronales convolucionales mediante la estimación de la postura de la cabeza representa un aporte académico valioso , ya que, al desarrollar y explorar nuevas técnicas y metodologías para abordar este problema, se contribuye al avance del conocimiento científico y tecnológico. Por lo que, los resultados obtenidos pueden servir como base para investigaciones futuras y para mejorar los sistemas existentes en el ámbito de la seguridad vial.

1.2.4. Justificación ambiental

Los accidentes de tránsito ocasionados por la distracción de los conductores pueden tener un impacto negativo en el medio ambiente, ya que, estos pueden resultar en la emisión de gases contaminantes, la generación de residuos y la degradación de los ecosistemas cercanos a un accidente de tránsito. Al prevenir estos accidentes a través de la detección de la distracción, se contribuye con la reducción de la huella ambiental y la preservación del entorno natural.

1.2.5. Justificación económica

Desde el punto de vista económico, los accidentes de tránsito que son causados por la distracción de los conductores generan costos significativos para la comunidad y la economía en general. Estos costos incluyen reparaciones de vehículos, congestión en las vías de tránsito, gastos médicos, pérdida de productividad de las personas, entre otros. Al implementar un sistema eficiente de detección de distracción, se pueden reducir los accidentes vehiculares y, en consecuencia, los costos asociados a este. Además, se mejora la productividad de conductores y empresas.

1.3. Objetivos

1.3.1. Objetivo general

Desarrollar e implementar en un sistema embebido un sistema detector de la distracción en conductores mediante la estimación de la postura de la cabeza en la ciudad del Cusco.

1.3.2. Objetivos específicos

- Desarrollar un prototipo del sistema inteligente que analice la distracción mediante la estimación de la postura de la cabeza con una sola cámara.
- Determinar los ángulos de rotación o de Euler para estimar la postura de la cabeza mediante el uso de una CNN.
- Implementar el sistema de detección de distracción mediante estimación de la postura de la cabeza en un sistema embebido.
- Evaluar el sistema inteligente experimental en un entorno real a un grupo de conductores voluntarios, para determinar su desempeño.

1.4. Variables

1.4.1. Variable independiente

Estado del conductor (alerta o distraído).

1.4.2. Variable dependiente

Salida del sistema detector de distracción (probabilidad de distracción).

1.5. Alcances y limitaciones

1.5.1. Alcances

Los alcances de este estudio pueden resumirse en:

- Desarrollo de un sistema de detección en tiempo real. Este sistema será capaz de analizar en tiempo real la distracción de los conductores utilizando una única cámara RGB.
- Utilización de un conjunto de datos existentes. Se utilizará un conjunto de datos existente para entrenar y validar el sistema de detección, se aprovechará un conjunto de datos existente que contenga imágenes relevantes para la estimación de la postura de cabeza.
- Implementación en un sistema embebido. La ejecución del sistema se llevará a cabo en un sistema embebido debidamente seleccionado. Este sistema embebido se caracterizará por su robustez, eficiencia energética y capacidad de cálculo aceptable, lo que garantizará un rendimiento confiable en condiciones de tiempo real.
- Evaluación de múltiples modelos de CNN livianos. Se llevará a cabo una evaluación de múltiples modelos de CNN livianos. Estos modelos se entrenarán y compararán para

determinar cuál de ellos ofrece el mejor rendimiento a la hora de detectar la distracción de los conductores.

1.5.2. Limitaciones

Algunas limitaciones de este estudio pueden resumirse en:

- La precisión de la detección de distracción puede verse afectada por condiciones adversas de iluminación o por elementos que obstruyan parcialmente la visión de la cámara, como gorras, lentes oscuros o sombras, así como también verse afectada por la noche.
- El estudio se centrará principalmente en la detección de la distracción a través de la estimación de la postura de la cabeza mediante los ángulos de rotación. Se excluirá el análisis de otras partes faciales como los ojos, la nariz y la boca, así como la parte trasera de la cabeza.
- El desarrollo de esta tesis se enfocará exclusivamente en la detección de la distracción en los conductores y no se realizará un análisis exhaustivo de otras partes de los vehículos, como el comportamiento de los pasajeros o las condiciones dentro del vehículo. Esto implica que ciertos factores de distracción relacionados con el entorno del conductor pueden quedar fuera del alcance de esta investigación.
- En la implementación del sistema detector de distracción, se utilizará hardware con un consumo máximo de 20 W, basado en arquitectura ARM o aarch64. Esta limitación de consumo puede implicar una reducción en el rendimiento del sistema, lo que podría afectar la velocidad de detección y procesamiento de los datos.

1.6. Delimitación del estudio

El estudio se centra en el desarrollo de un sistema inteligente basado en algoritmos de visión computacional para la detección de la distracción en conductores mediante la estimación

de la postura de la cabeza. El laboratorio donde se desarrolló y probó el sistema se encuentra en la ciudad del Cusco, Perú. El desarrollo y prueba del sistema se llevó a cabo durante un periodo de 12 meses, desde enero a diciembre del 2024, y durante este periodo, el prototipo experimental salió del laboratorio y se sometió a pruebas reales en la ciudad del Cusco durante un solo día.

Capítulo 2

Marco teórico

2.1. Antecedentes de estudio

En el artículo publicado por (Zhao et al., 2020), titulado **Driver Distraction Detection Method Based on Continuous Head Pose Estimation**, se hizo uso de una red neuronal convolucional (CNN) llamada HPE_ResNet50. El conjunto de datos utilizado para el entrenamiento fue 300W_LP, mientras que para la evaluación cuantitativa y cualitativa de la postura de la cabeza se emplearon AFLW y AFLW2000. Para el análisis de distracción se utilizó SF3D, con el cual se generaron los ángulos de rotación de giro, cabeceo y balanceo a través de la predicción del modelo CNN. Estos ángulos se usaron para determinar una zona de conducción segura mediante el cálculo de la distancia espacial y un umbral. Posteriormente, se utilizó un conjunto de datos llamado Driver_Imgs para la evaluación, donde se obtuvieron resultados prometedores en la detección de distracción. Sin embargo, los resultados no fueron convincentes en la detección de no distracción, especialmente en giros seguros hacia la derecha e izquierda.

En el artículo de conferencia publicado por (Sri Mounika et al., 2022), titulado **Driver Drowsiness Detection Using Eye Aspect Ratio (EAR), Mouth Aspect Ratio (MAR), and Driver Distraction Using Head Pose Estimation**, se aborda el problema de la distracción controlando la posición de la cabeza del conductor respecto a la vía. El análisis comienza con la detección de rostros mediante el algoritmo de Haar, seguido de la identificación de puntos de interés utilizando la herramienta Dlib (puntos de referencia en las esquinas exteriores de los

ojos, nariz, comisuras de los labios y mandíbula), que son transformados a sus coordenadas 3D para calcular los ángulos de rotación. La distracción se detecta basándose en los movimientos de la cabeza del conductor y los umbrales de distracción fueron propuestos por el autor.

En la tesis de grado presentada por (Florez Zela, 2024), titulada **Diseño e implementación de un sistema detector de somnolencia en tiempo real mediante visión computacional usando redes neuronales convolucionales aplicado a conductores**, se aborda el problema de somnolencia en conductores. Para ello, el autor utiliza un NVIDIA Jetson Nano y una cámara de infrarrojo cercano (NIR). El método que propone emplea Mouth Aspect Ratio (MAR) para la detección de bostezos y redes neuronales convolucionales (CNN) para la detección de somnolencia visual. El autor propone 2 redes neuronales, DD-AI y DD-AI-G, esta última obtuvo una precisión promedio del 91.48 % y una sensibilidad del 86.28 % al realizar pruebas en conducción real.

2.2. Bases teóricas

2.2.1. Distracción en la conducción

La distracción es la pérdida de concentración provocada por algún factor externo o interno (RAE, 2023). Esta pérdida de atención puede llevar a que los conductores no realicen adecuadamente su tarea de conducir, aumentando el riesgo de accidentes. A diferencia de factores como el alcohol o las drogas, que son ampliamente reconocidos y aceptados como graves riesgos para la seguridad vial, la distracción a menudo no se percibe con la misma gravedad. Muchos conductores subestiman el impacto que la distracción puede tener en su capacidad para operar un vehículo de manera segura. Este fenómeno se debe a que la distracción puede ser menos evidente y más difícil de identificar en comparación con los efectos de las sustancias mencionadas (Ministerio del Interior de España & Dirección General de Tráfico, 2014).

Causas principales

Las causas principales de la distracción pueden ser visuales, cognitivas, físicas y auditivas (Fundación Carlos Slim, 2016). Las distracciones visuales suelen ocurrir al usar o manipular celulares, GPS, leer publicidad o buscar objetos en la guantera. Las distracciones cognitivas se producen cuando la persona pierde la atención mental, ya sea por problemas personales o al mantener conversaciones con los pasajeros. Por otro lado, las distracciones físicas se presentan al realizar acciones como comer, beber, fumar, maquillarse o manipular objetos mientras se conduce. Además, las distracciones auditivas generalmente ocurren al escuchar música o contestar llamadas telefónicas (Fundación Carlos Slim, 2016; Ministerio del Interior de España & Dirección General de Tráfico, 2014).

2.2.2. Sistemas avanzados de ayuda a la conducción (ADAS)

Los sistemas ADAS (por sus siglas en inglés, Advanced Driver Assistance Systems) son una combinación de componentes activos y pasivos diseñados para reducir los accidentes de vehículos provocados por el factor humano. Estos sistemas brindan apoyo a los conductores y también pueden evaluar la eficacia de sus acciones. Utilizando una variedad de sensores, reciben datos en tiempo real del entorno en el que se encuentra el vehículo y reaccionan de manera adecuada ante situaciones específicas (DEWESoft, 2021).

Entre todas las tecnologías, la detección del estado del conductor (DMS, por sus siglas en inglés) es la más relevante, ya que se enfoca en monitorear el estado del conductor para identificar signos de distracción o somnolencia. Estos sistemas trabajan para aliviar la carga del conductor, fortalecer su percepción y alertar ante posibles errores (Khan & Lee, 2019). Mediante la integración de mediciones del comportamiento biológico, especialmente métricas oculares (cara, ojos, cabeza), con el análisis del desempeño al volante, estos sistemas tienen la capacidad de determinar el estado del conductor en tiempo real (Hayley et al., 2021).

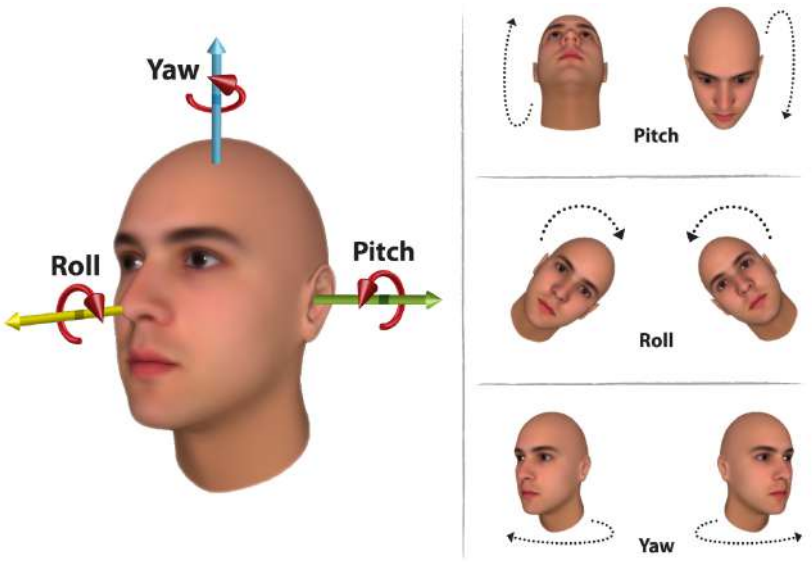
2.2.3. Métodos de detección de distracción

Postura de la cabeza

Una de las formas de detectar la distracción se realiza mediante la estimación de la postura de la cabeza, un enfoque que admite múltiples interpretaciones. Una estrategia para lograr esto implica el uso de múltiples grados de libertad (DOF), tal como lo señala (Arcoverde Neto et al., 2014). Desde esta perspectiva, incluso un movimiento básico de izquierda a derecha puede considerarse una indicación de la postura de la cabeza. Sin embargo, en enfoques más avanzados, la estimación puede alcanzar una mayor complejidad, permitiendo determinar la postura tridimensional con base en los ángulos de rotación conocidos como yaw (guiñada o giro), pitch (cabeceo) y roll (balanceo) (Arcoverde Neto et al., 2014). En la figura 2.1 se pueden apreciar los ángulos de rotación y sus respectivos movimientos.

Figura 2.1

Representación de los ángulos de rotación tridimensionales de la cabeza.



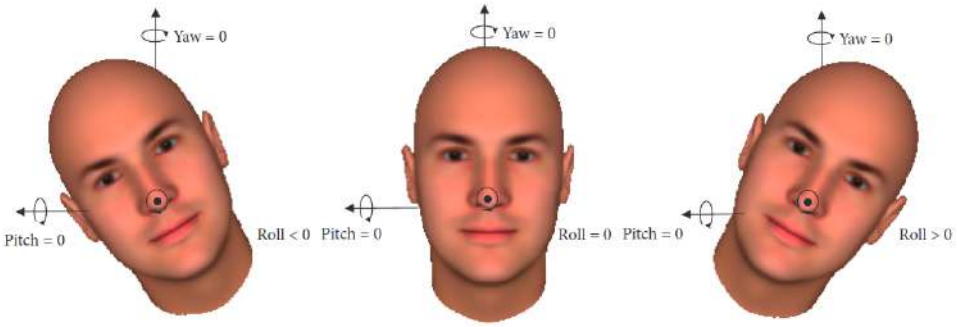
Nota: Extraído de (Arcoverde Neto et al., 2014).

Una cuestión a tener en cuenta al estimar la postura de la cabeza es el marco de referencia de la cabeza con respecto a la cámara. La figura 2.2 ilustra que cuando la cámara está orientada hacia el frente, los ángulos de rotación son iguales a cero, y cualquier cambio en el ángulo de

la cámara no afecta la rotación real de la cabeza en relación con el cuerpo humano. En otras palabras, si el ángulo de la imagen cambia, pero la cabeza no se rota, los resultados de la pose de la cabeza aún cambiarán, pero no debido a un cambio real en la postura de la cabeza.

Para observar una explicación gráfica del marco de referencia de la cámara y la postura de la cabeza, se presenta la figura 2.3, que muestra el sistema de coordenadas de la cámara y la postura de la cabeza, además del plano de la imagen y sus coordenadas intrínsecas.

Figura 2.2
Imagen de la cara con postura de la cabeza.



Nota: Extraído de (Zhao et al., 2020).

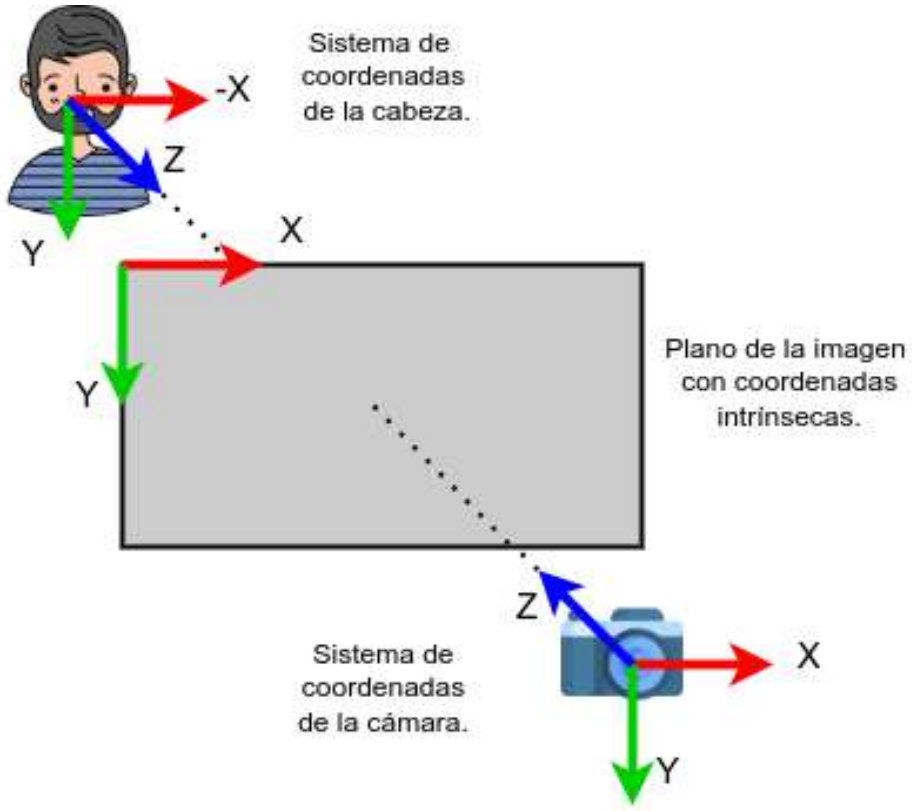
Umbralización basada en distancia espacial para distracción mediante ángulos de la postura de la cabeza

Este método de detección de distracción, basado en los ángulos de rotación extraídos como resultado de realizar predicciones en la estimación de la postura de la cabeza, fue propuesto por Zhao et al. (2020). Consiste en determinar una zona de conducción segura. Para ello, es importante considerar la perspectiva del ángulo de la cámara con el que se estima la postura de la cabeza.

Para determinar el rango y umbral de la zona de conducción segura, se siguen los pasos específicos detallados en (Zhao et al., 2020).

1. Calcular los ángulos de rotación.

Figura 2.3
Sistema de referencia de la cabeza respecto a la cámara.



- Para una secuencia continua de cuadros de conducción segura con n cuadros, se calcula el ángulo de rotación para cada cuadro, lo que da como resultado un conjunto de ángulos de rotación de giro, cabeceo y balanceo: $(yaw_i, pitch_i, roll_i)$ para cada cuadro i.

- Se halla el valor promedio de los ángulos de rotación para estos n cuadros, que se denota como $(\overline{yaw_n}, \overline{pitch_n}, \overline{roll_n})$. Este valor promedio se utiliza como punto de referencia base para b_j (donde $j = 0, ..., m$).

2. Repetir y establecer el punto de referencia seguro.

- Se repite el paso anterior un total de m veces para obtener un conjunto de puntos base seguros $B = \{b_1, b_2, ..., b_m\}$.

- Se calcula el valor promedio del conjunto B, que se denota como el punto base de conducción segura $\beta = (\overline{yaw}, \overline{pitch}, \overline{roll})$.

- Se realiza un análisis estadístico de los puntos $m \times n$ para eliminar los puntos extremos y se expande el rango a un espacio específico Ω, que representa el rango de conducción segura.

3. Calcular la distancia espacial.

   - Se calcula la distancia espacial d_i de cada punto en el grupo de n puntos desde el punto base β usando la expresión 2.1.

$$d_i(i = 1, 2, 3, ..., n) = \sqrt{(\overline{yaw} - yaw_i)^2 + (\overline{pitch} - pitch_i)^2 + (\overline{roll} - roll_i)^2} \tag{2.1}$$

   - Para establecer el umbral de seguridad, se toma el valor de d_i que contenga al menos el 90 % de los datos de conducción segura y está denotado con D.

4. Probar k cuadros continuos y compararlos con el espacio Ω. Se prueban cuadros continuos de video, calculando la distancia espacial d_i con respecto al punto base de conducción segura β.

A continuación se describen las métricas propuestas por (Zhao et al., 2020) para evaluar la distracción del conductor.

- Número total de valores que superan el umbral (OT). Esta métrica se obtiene utilizando la expresión 2.2, suponiendo secuencias de video de k cuadros. Donde $1(d_i > D)$ indica que vale 1 si $d_i > D$ y 0 en caso contrario. Este indicador ayuda a identificar poses de la cabeza que pueden indicar distracción.

$$OT = \sum_{i=1}^{k} 1(d_i > D) \tag{2.2}$$

- Número máximo de valores continuos que superan el umbral (MCOT). Es el número máximo de cuadros consecutivos que superan el umbral D. Este indicador ayuda a identificar patrones de distracción continua y sostenida.

- Con base a los puntos anteriores se puede hallar en forma porcentual el valor de OT mediante la expresión.

$$OT\,\% = \frac{OT}{k} * 100\,\% \tag{2.3}$$

- Para mejorar la estabilidad del sistema de detección de distracción y reducir el impacto de valores atípicos, Zhao et al. (2020) sugiere que, si se superan cinco cuadros consecutivos equivalentes a 0.25 s y con 20 cuadros por segundo (FPS), se incrementa un conteo de distracción en 1.

2.2.4. Procesamiento de imágenes digitales

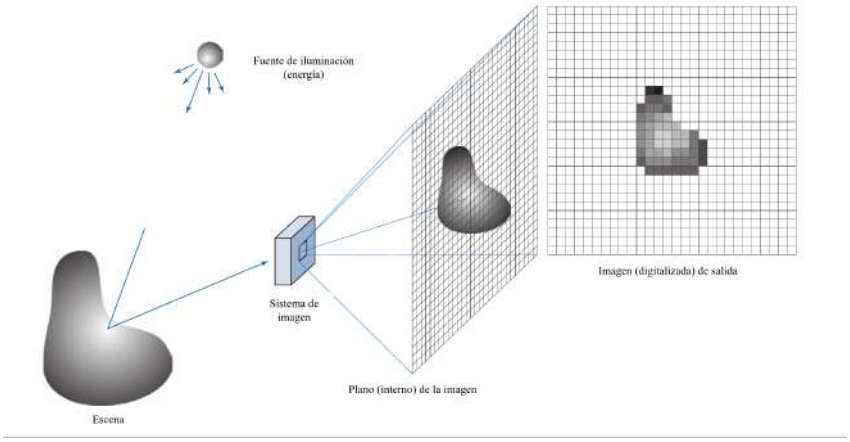
El procesamiento de imágenes digitales es una disciplina que se enfoca en estudiar los procesos en las que los datos de entrada y de salida son imágenes digitales. A pesar de que los seres humanos estamos limitados por el espectro visible del espectro electromagnético, es importante destacar que el procesamiento de imágenes digitales permite trabajar con imágenes que pueden representar información en otras partes del espectro electromagnético, como el infrarrojo, rayos-x, ondas de radio, entre otros, lo que permite que se amplíe significativamente las capacidades de percepción y de análisis (Gonzalez & Woods, 2008).

Formación de una imagen digital

Una imagen puede ser definida como una función de dos dimensiones $f(x, y)$, donde (x, y) son las coordenadas en el plano, y donde la función f es la intensidad de gris de la imagen en dicho punto, cada uno de estos puntos se llama pixel, que es la unidad básica de una imagen digital (Gonzalez & Woods, 2008).

El proceso de formación de una imagen digital hace referencia al proceso mediante el cual se captura una escena del mundo real y se convierte en una representación digital que puede ser almacenada, procesada y visualizada en un dispositivo electrónico. Este proceso se puede

Figura 2.4
Ejemplo de adquisición y formación de una imagen digital.



Nota: Extraído de (Gonzalez & Woods, 2008).

visualizar de manera gráfica en la Figura 2.4.

Representación de una imagen digital

Las imágenes digitales se representan matemáticamente mediante una matriz numérica donde cada elemento es un pixel de la matriz. Esta matriz numérica se utiliza en el procesamiento computacional de imágenes y permite representar una imagen de manera cuantitativa (Gonzalez & Woods, 2008). En (Gonzalez & Woods, 2008) se define la expresión 2.4 de una imagen digital de tamaño MxN.

$$f(x,y) = \begin{bmatrix} f(0,0) & \dots & f(0,N-1) \\ \vdots & \ddots & \vdots \\ f(M-1,0) & \dots & f(M-1,N-1) \end{bmatrix} \tag{2.4}$$

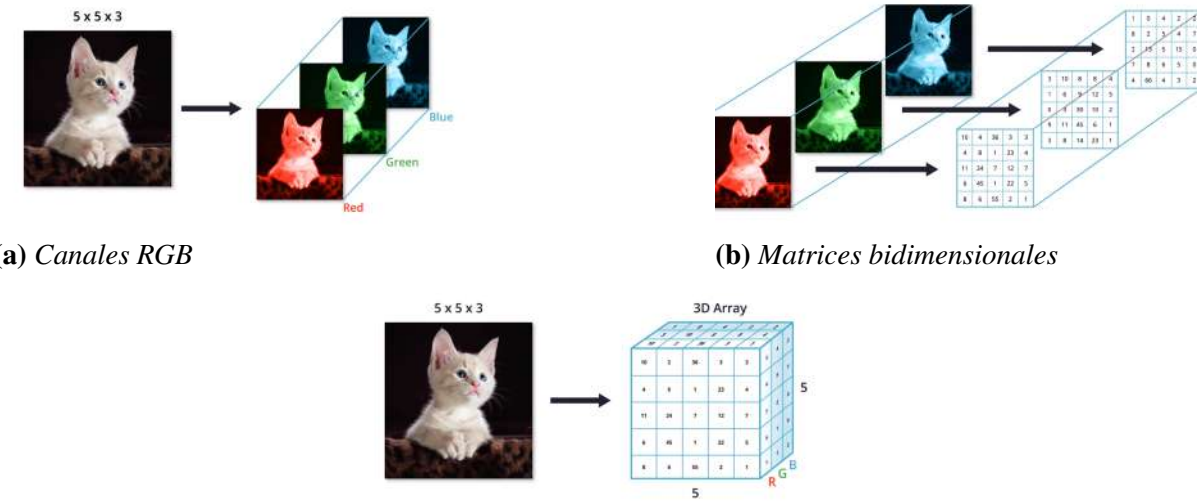
Imágenes digitales de múltiples canales

Como se vio anteriormente, las imágenes están compuestas por una matriz bidimensional de MxN píxeles en escala de grises. De la misma manera, las imágenes pueden tener matrices

tridimensionales que estarán formadas por su altura, ancho y una profundidad que será determinada por el número de canales de color que la imagen contenga. Las imágenes que contienen 3 canales son llamadas RGB, que en conjunto forman una sola imagen de color, y cabe resaltar que cada canal cuenta con su propia matriz bidimensional. En el caso de RGB se tienen 3 canales que al final se traducen en 3 matrices bidimensionales (Udacity, 2019). En la figura 2.5 se puede apreciar de forma gráfica la explicación dada.

Figura 2.5

Representación de una imagen RGB.



(a) *Canales RGB*

(b) *Matrices bidimensionales*

(c) *Matriz tridimensional*

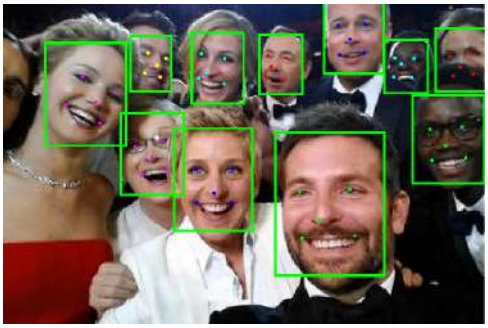
Nota: Extraído de (Udacity, 2019).

2.2.5. Detección de rostros y puntos faciales

La detección de rostros es la técnica biométrica que se utiliza para identificar rostros humanos en imágenes o videos (ver figura 2.6a). Esta técnica es utilizada para tareas de identificación como el reconocimiento facial y la seguridad (Amazon Web Services, Inc., 2023; Shetty et al., 2021). Los puntos faciales permiten detectar puntos de referencia de un rostro humano, estos se usan para aplicar filtros, crear avatares virtuales y efectos faciales (Mediapipe, 2023) (ver figura 2.6b). A continuación se mencionan las técnicas de detección de rostros y puntos de referencia faciales más comunes. Entre las técnicas de detección de rostros y puntos de referencia faciales más comunes se encuentran Haar Cascade, que es un método de detec-

ción muy eficaz y puede detectar imágenes en diferentes entornos (Shetty et al., 2021), Dlib es otra técnica que detecta rostros y además es capaz de extraer 68 puntos faciales del rostro de una persona (Dlib, 2015), Mediapipe se distingue por incorporar soluciones de aprendizaje automático de manera eficiente y destaca por ser rápido y consumir bajos recursos (Mediapipe, 2023), y MTCNN que se enfoca en realizar detección y alineamiento de rostros en entornos no controlados (Zhang et al., 2016).

Figura 2.6
Detección de rostros y puntos faciales.



(a) *Detección de rostros con MTCNN*

(b) *detección de puntos faciales con Mediapipe face landmarks*

Nota: Extraído de (Mediapipe, 2023; Zhang et al., 2016).

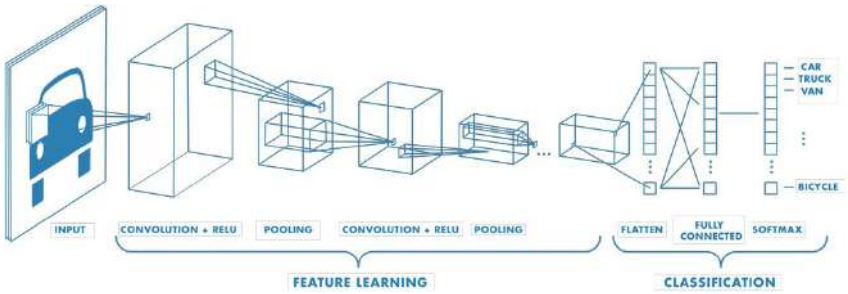
2.2.6. Redes neuronales convolucionales (CNN)

Esta tecnología forma parte del campo de la inteligencia artificial y del aprendizaje profundo, fueron diseñadas tomando de inspiración las redes neuronales biológicas, similares a las que componen el cerebro humano (Olabe, 1998). Las redes neuronales convolucionales (CNN), están compuestas de manera general por una capa de entrada, capas ocultas y una capa de salida. Las capas ocultas constan de tres tipos principales de capas: capa convolucional, capa de agrupación (o de ”pooling“) y capa totalmente conectada (o ”fully connected“), que a su vez está dividido en etapas: extracción de características y clasificación (IBM, 2021).

Las redes neuronales convolucionales son muy eficaces para identificar patrones que contienen las imágenes (reconocer objetos, clases, etc.), también resultan útiles procesando

otro tipo de datos como los de audio y series temporales (MathWorks,). Para que estas redes funciones de forma eficiente necesitan una gran cantidad de datos de entrenamiento para ajustar sus pesos y sesgos. Una vez se ha finalizado el proceso de entrenamiento, estas redes se convierten en poderosas herramientas (IBM, 2021). En la figura 2.7 se muestra un ejemplo de la representación de una CNN.

Figura 2.7
Representación de una CNN.



Nota: Extraído de (MathWorks,).

Capa convolucional

Las redes neuronales convolucionales (CNN) se basan en capas convolucionales como su bloque principal que operan mediante el desplazamiento de filtros sobre la entrada de datos, con el fin de detectar características relevantes. Cada filtro es una matriz de pesos que se aplica a regiones específicas de la entrada. Este proceso, llamado convolución, utiliza la convolución de matrices y produce mapas de características que resaltan las características importantes de la entrada (IBM, 2021). La convolución de matrices se expresa de la siguiente manera.

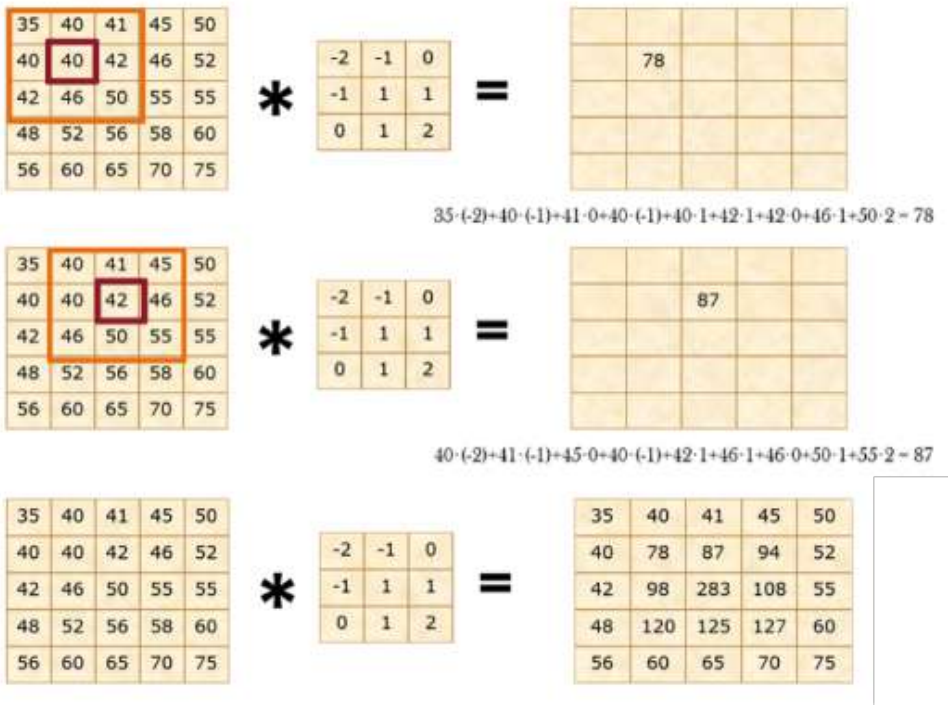
Sea una matriz A_{mxn} que representa una imagen digital en escala a grises, y una matriz $C_{(2N+1)x(2N+1)}$ con $2N + 1 < m, n$ que representa el filtro, núcleo o kernel de la convolución. Se define la convolución de las matrices A y C como una nueva matriz $D = A * C$ (Palomares et al., 2016), y definida por la expresión 2.5:

$$d_{ij} = \frac{1}{c} \sum_{r=1}^{2N+1} \sum_{s=1}^{2N+1} a_{i-N+r-1,j-N+s-1} c_{r,s} \tag{2.5}$$

Donde $c = \sum_{i,j=1}^{2N+1} c_{i,j}$ (si $c = 0$ entonces se toma $c = 1$), teniendo en cuenta que $d_{i,j}$ solamente está definido para valores de $i = N + 1, ..., m - N - 1$ y $j = N + 1, ..., n - N - 1$.

En la figura 2.8 se aprecia de manera gráfica el proceso de la convolución de matrices, donde el resultado de realizar dicha operación se obtiene una nueva matriz.

Figura 2.8
Proceso de convolución de matrices.



Nota: Extraído de (Palomares et al., 2016).

Después de realizar los cálculos de convolución, se suele aplicar una función de activación (ReLU, sigmoide, tangente hiperbólica, entre otros), con la finalidad de introducir no linealidad. Las CNN pueden tener múltiples capas convolucionales, que permite crear jerarquías de características complejas, resultando en la capacidad de identificar objetos o patrones en las imágenes a diferentes niveles de abstracción. Por ejemplo, una CNN puede detectar partes individuales de un objeto y luego combinarlas para reconocer el objeto completo (IBM, 2021).

Capa de agrupación

Es la capa que permite reducir el tamaño de imagen, y de igual forma que la capa convolucional barre toda la imagen con un filtro, pero la diferencia es que este no tiene ningún peso (IBM, 2021). Según (IBM, 2021) los dos tipos de agrupación principales son:

- Agrupación máxima: En un grupo de píxeles, se selecciona el mayor valor que se envía a la matriz de salida.
- Agrupación promedio: convierte los valores de un grupo de píxeles en uno solo tomando el promedio para enviarlo a la matriz de salida.

Capa densa o totalmente conectada

Esta capa tiene cada nodo de una capa conectada a cada nodo en la capa previa. Realiza la clasificación o regresión en función de las características que fueron extraídas de las capas previas (IBM, 2021), y devuelve a la salida un vector con las características relevantes, en caso de clasificación representan a las clases.

Transferencia de aprendizaje

Transferencia de aprendizaje es una técnica que consiste en utilizar una red neuronal guardada que fue entrenada previamente utilizando un conjunto de datos de gran tamaño, con el propósito de realizar la tarea de clasificar a gran escala. La idea clave detrás de la transferencia de aprendizaje es que un modelo preentrenado es capaz de capturar características generales del mundo visual, lo que permite aprovechar sus mapas de características aprendidos sin necesidad de entrenar un modelo desde cero. El modelo preentrenado puede ser usado directamente o puede ser personalizado dependiendo de la tarea especifica que se quiera realizar (TensorFlow, 2022).

Optimización de los parámetros de modelos

1. Parámetros e hiperparámetros: los parámetros de una CNN son los valores dados en la creación de cada capa convolucional, capa de agrupación o totalmente conectada, como por ejemplo el tamaño del filtro o el tamaño de los canales de entrada o de salida, que no se pueden modificar al momento de realizar el entrenamiento y el número de parámetros depende de estos valores. En cambio, los hiperparámetros son parámetros que se pueden ajustar y que ayudan a controlar el proceso de optimización, pudiendo afectar el entrenamiento y las tasas de convergencia de los modelos (PyTorch, 2024).

   Los hiperparámetros más utilizados son mencionados y definidos por (PyTorch, 2024):

   - Número de épocas: es el número de veces para iterar sobre el conjunto de datos.
   - Tasa de aprendizaje: es cuanto actualizar los parámetros del modelo en cada lote/época, donde los valores pequeños dan como resultado velocidades de aprendizaje lentas, mientras valores grandes pueden llevar a comportamientos impredecibles durante el entrenamiento.
   - Tamaño de lote: es el número de ejemplos de datos que se pasan a traves de la red neuronal antes que los parámetros se actualicen.

2. Función de pérdida: mide la desigualdad del resultado obtenido con el valor que se quiere obtener, y se busca minimizarlo durante el entrenamiento. Las funciones de pérdida más comunes son el error cuadrático medio (MSE) para la regresión y la entropía cruzada (Cross entropy) para clasificación, la que combina softmax y probabilidad logarítmica negativa (NLL) (PyTorch, 2024).

3. Optimizador: realiza ajustes a los parámetros de un modelo con el fin de reducir el error que pueda presentar dicho modelo en cada paso de entrenamiento. Los optimizadores más utilizados son el descenso de gradiente estocástico (SGD), ADAM y RMSProp (PyTorch, 2024).

Regresión y clasificación

Es el proceso de predecir una variable dependiente de otra independiente (regresión simple) o de múltiples variables independientes (regresión múltiple) (Field et al., 2012). Está representado por la siguiente ecuación general 2.6.

$$resultado_i = (modelo) + error_i \tag{2.6}$$

El modelo fue entrenado para predecir un determinado tipo de datos, y el resultado puede ser predicho por cualquier modelo entrenado con esos datos, aunque con algún margen de error (Field et al., 2012).

Mientras, la clasificación es el proceso mediante el cual se determina la clase o conjunto de características al que pertenece un elemento determinado. Por ejemplo, el conjunto de perros, gatos, vehículos, animales, entre otros.

2.2.7. Métricas de evaluación para la postura de la cabeza

La tarea de estimación de la postura de la cabeza es un problema de regresión, y las métricas son las siguientes:

Pérdida Geodésica

Esta es propuesta por (Hempel et al., 2022), en la que se interpreta a la distancia geodésica como el camino más corto entro entre dos rotaciones tridimensionales y está dada por la ecuación 2.7.

$$Lg = arccos(\frac{tr(R_p R_{gt}^T) - 1}{2}) \tag{2.7}$$

Donde R_p es la matriz estimada por el modelo y R_{gt}^T es la matriz transpuesta de la matriz de verdad fundamental, que pertenecen a un mismo grupo de rotación $SO3$.

Error absoluto promedio (MAE)

Es una métrica común usada generalmente en problemas de regresión, el cual mide el promedio de las diferencias absolutas entre los valores y los valores verdaderos (referencia), siempre es un valor positivo, es lineal y no diferenciable (Terven et al., 2023). MAE está definida por la ecuación 2.8

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{2.8}$$

Donde n es el número de muestras, y_i y $\hat{y}_i$ son el valor de verdad fundamental y predicho de la i-ésima muestra.

La expresión 2.8 se puede aplicar para hallar el MAE de los ángulos de rotación, los cuales se obtienen teniendo en cuenta la naturaleza cíclica de los ángulos. Es decir, considerando que, por ejemplo, $370° = 10°$ en el caso de considerar de $0°$ a $360°$ y $270° = -90°$ para el caso de considerar rangos de $-180°$ a $180°$.

2.2.8. Conjuntos de datos

Para estimación de la postura de la cabeza

Durante las últimas dos décadas, se ha realizado un desarrollo significativo de los conjuntos de datos relacionados con la estimación de la postura de la cabeza. En los últimos años, este enfoque se ha ido ampliando aún más hacia el contexto vehicular, como se refleja en la tabla 2.1, en esta tabla, los conjuntos de datos destacados en negrita se utilizan ampliamente y son aplicables a la estimación de la postura de la cabeza de manera general, mientras que los

demás conjuntos están específicamente desarrollados para un contexto vehicular.

Tabla 2.1
Conjuntos de datos disponibles para la estimación de la postura de la cabeza.

Conjunto de datos	Año	Personas	Número de imágenes	Giro	Cabeceo	Balanceo
BU	2000	5	200	Si	Si	Si
Pointing'04	2004	15	2970	Si	Si	No
AFLW	2011		25993	Si	Si	Si
BIWI Kinect	2011	20	15000	Si	Si	Si
AFW	2012	205	468	Si	Si	No
300W_LP	2015		122450	Si	Si	Si
AFLW2000-3D	2015		2000	Si	Si	Si
DriveAHead	2017	20	1M	Si	Si	Si
Pandora	2017	22	25000	Si	Si	Si
DD-Pose	2019	27	330k	Si	Si	Si
AutoPose	2020	20	1M	Si	Si	Si
MDM corpus	2021	9	12848	Si	Si	Si

Nota: Modificado de (Asperti & Filippini, 2023).

Para esta tesis, se tomarán en cuenta los conjuntos de datos más utilizados para la tarea de estimación de la postura de la cabeza.

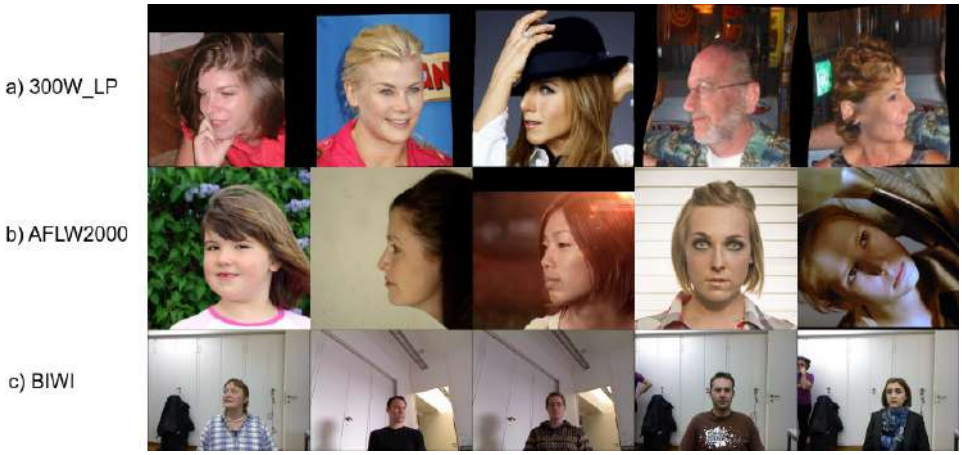
El primero de ellos es 300W_LP, que se utiliza generalmente para el entrenamiento de modelos CNN. Este conjunto de datos es una expansión del conjunto de datos sintéticos 300W y fue generado para aumentar el número de muestras con poses extremas. Este conjunto de datos cuenta con 122 453 imágenes, con un rango que va desde los $-90°$ hasta los $90°$ para los ángulos giro (yaw), cabeceo (pitch) y balanceo (roll) (Asperti & Filippini, 2023).

AFLW2000 es otro conjunto de datos, pero a diferencia de 300W_LP, se utiliza para realizar las pruebas a los modelos. Este conjunto cuenta con 2000 imágenes extraídas del conjunto de datos AFLW, las cuales tienen sus anotaciones basadas en 68 puntos de referencia utilizando un modelo 3D. Sus rangos varían desde $-120°$ hasta $120°$ en giro (yaw), y desde $-90°$ hasta $90°$ en cabeceo (pitch) y balanceo (roll) (Asperti & Filippini, 2023). Generalmente, los ángulos superiores a $90°$ e inferiores a $-90°$ se eliminan de este conjunto de datos antes de su uso para evaluación, quedando un total de 1968 imágenes.

De manera similar, BIWI es un conjunto de datos que se utiliza generalmente para realizar

pruebas. Sus anotaciones de ángulos fueron hechas con base en imágenes RGB-D de diferentes sujetos. Este conjunto de datos varía de $-75°$ a $75°$ en giro, $-60°$ a $60°$ en cabeceo, y $-50°$ a $50°$ en balanceo, con un total de 15,000 imágenes (Asperti & Filippini, 2023).

Figura 2.9
Ejemplos de los conjuntos de datos 300W_LP, AFLW2000 y BIWI.



La figura 2.9, muestra 5 ejemplos de cada conjunto de datos que fueron definidos anteriormente, 300W_LP, AFLW2000 y BIWI.

Para distracción

Para abordar el desafío de detectar la distracción en conductores, es crucial contar con conjuntos de datos que capturen una amplia variedad de comportamientos y condiciones de manejo. A continuación, se describen los conjuntos de datos más relevantes para esta tarea.

Uno de estos conjuntos de datos es el *State Farm Distracted Driver Detection* (SFDDD), empleado en el estudio de (Zhao et al., 2020). Este conjunto de datos fue lanzado por Kaggle como parte de una competencia con el objetivo de desarrollar modelos de aprendizaje profundo que identifiquen si un conductor está distraído basándose en imágenes. El conjunto de datos cuenta con 10 etiquetas que determinan el estado de la distracción, las cuales fueron registradas con una cámara que graba el cuerpo del conductor. Además, incluye más de 100,000 imágenes (Montoya et al., 2016).

Por otro lado, el conjunto de datos *Driver Monitoring Dataset* (DMD), propuesto por

Vicomtech en asociación con Intel, ofrece un conjunto de datos mucho más amplio y detallado. Este conjunto incluye información de videos en RGB, NIR y profundidad, con etiquetas realizadas cuadro por cuadro. A diferencia de SFDDD, este conjunto no solo está diseñado para detectar distracción, sino también para realizar un monitoreo completo de todas las actividades del conductor, como somnolencia, micro-sueños, dirección de la mirada, entre otros. Este conjunto recopila información de 3 cámaras RGB-D/NIR, que graban las manos, el rostro y el cuerpo del conductor, y proporciona datos de 37 personas en entornos de conducción real, conducción simulada y en un simulador de conducción (Ortega et al., 2020). Para más información, ver anexo E.

Una de las razones para elegir un conjunto de datos es que, a diferencia de DMD, SFDDD solo graba el cuerpo del conductor desde un costado. En cambio, DMD cuenta con videos grabados con una cámara frontal, lo cual facilita información de la cara del conductor, lo que lo hace más adecuado para su utilización en la presente tesis, ya que la estimación de la postura de la cabeza es mucho más sencilla.

Figura 2.10
Ejemplos de los conjuntos de datos de distracción SFDDD y DMD.

La figura 2.10 muestra ejemplos de los dos conjuntos de datos de distracción antes mencionados, teniendo que los ejemplos presentados de DMD pertenecen solo a la cámara frontal.

2.2.9. Representaciones de orientación

Matrices de rotación

Es una matriz que transforma los vectores por los cuales se multiplica rotándolos en un determinado ángulo θ (Soto, 2021), en 3D las rotaciones de los vectores se realiza con respecto a los ejes x, y o z, definidas por las matrices 2.9, 2.10 y 2.11 respectivamente.

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\theta & -sen\theta \\ 0 & sen\theta & cos\theta \end{bmatrix} \tag{2.9}$$

$$R_y(\theta) = \begin{bmatrix} cos\theta & 0 & sen\theta \\ 0 & 1 & 0 \\ -sen\theta & 0 & cos\theta \end{bmatrix} \tag{2.10}$$

$$R_z(\theta) = \begin{bmatrix} cos\theta & -sen\theta & 0 \\ sen\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.11}$$

Teniendo que en cada rotación alrededor de los ejes, la rotación se lleva a cabo en sentido antihorario, siguiendo la regla de la mano derecha.

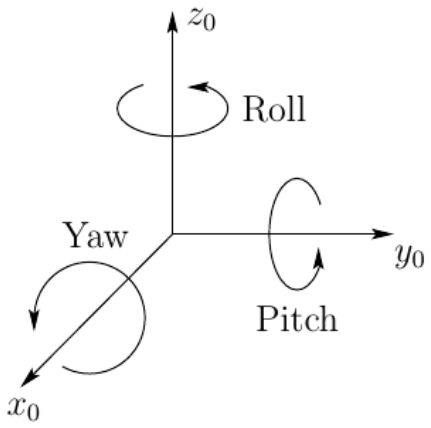
Para conseguir rotaciones de manera general, se deben multiplicar las matrices correspondientes. Una rotación siguiendo el orden x-y-z se representa con la expresión 2.12.

$$R = R_z(R_yR_x) \tag{2.12}$$

Giro, cabeceo y balanceo (yaw, pitch, roll)

Una matriz de rotación R puede describirse como un producto de rotaciones sobre los ejes principales de coordenadas x, y y z. Estas rotaciones definen los ángulos de giro (yaw), cabeceo (pitch) y balanceo (roll), que se denotan como ψ, θ y ϕ respectivamente, como se ilustra en la figura 2.11. El orden común para estas rotaciones es $X - Y - Z$, lo que implica una rotación de giro sobre el eje principal x_0, seguida de una rotación de cabeceo sobre el eje principal y_0 y, finalmente, una rotación de balanceo sobre el eje principal z_0 (Spong & Vidyasagar, 2008).

Figura 2.11
Representación de los ángulos de giro, cabeceo y balanceo.



Nota: Extraído de (Spong & Vidyasagar, 2008).

La matriz de rotación resultante de las múltiples rotaciones está dada por la expresión 2.13.

$$R = R_{z,\phi}R_{y,\theta}R_{x,\psi} = \begin{bmatrix} C_\phi & -S_\phi & 0 \\ S_\phi & C_\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C_\theta & 0 & S_\theta \\ 0 & 1 & 0 \\ -S_\theta & 0 & C_\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & C_\psi & -S_\psi \\ 0 & S_\psi & C_\psi \end{bmatrix} \tag{2.13}$$

El resultado de realizar las rotaciones es la matriz 2.14.

$$R = \begin{bmatrix} C_\phi C_\theta & -S_\phi C_\psi + C_\phi S_\theta S_\psi & S_\phi S_\psi + C_\phi S_\theta C_\psi \\ S_\phi C_\theta & C_\phi C_\psi + S_\phi S_\theta S_\psi & -C_\phi S_\psi + S_\phi S_\theta C_\psi \\ -S_\theta & C_\theta S_\psi & C_\theta C_\psi \end{bmatrix} \tag{2.14}$$

Haciendo uso de la matriz 2.15, da como resultado los ángulos de giro, cabeceo y balanceo que son representados por las ecuaciones 2.16, 2.17 y 2.18 respectivamente.

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{2.15}$$

$$\psi = \text{atan2}\left(r_{33}, r_{32}\right) \tag{2.16}$$

$$\theta = \text{atan2}\left(-\sqrt{r_{11}^2 + r_{21}^2}, -r_{31}\right) \tag{2.17}$$

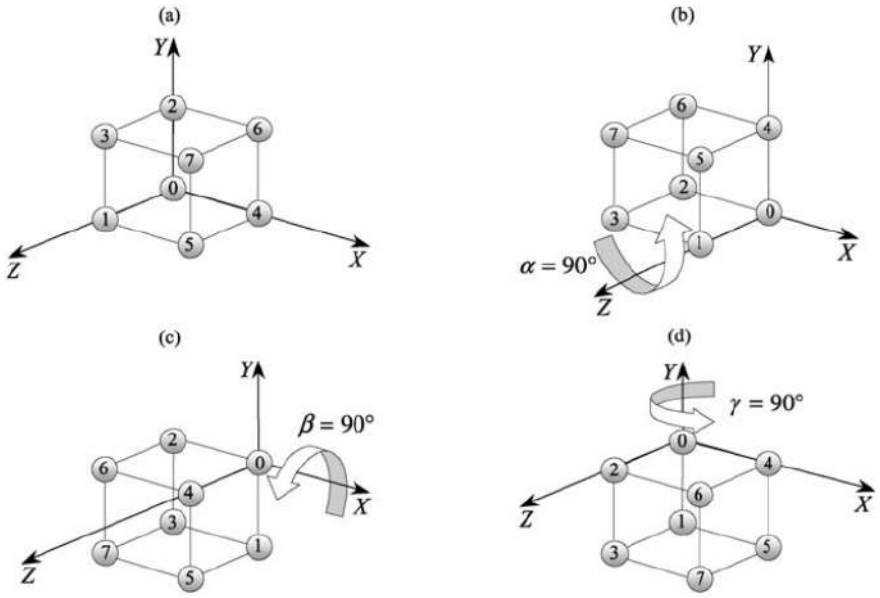
$$\phi = \text{atan2}\left(r_{11}, r_{21}\right) \tag{2.18}$$

Bloqueo de Cardán

El bloqueo de cardán (gimbal lock en inglés) es un fenómeno que ocurre en sistemas de rotación, como los descritos anteriormente. Sucede cuando dos de los tres ejes de rotación se alinean, lo que resulta en la pérdida de un grado de libertad en el movimiento, haciendo que un objeto que debería de girar libremente esté limitado por tras la pérdida de un grado de libertad. La alineación de los ejes ocurre cuando se usan ciertas combinaciones de ángulos (Vince, 2011).

La figura 2.12, muestra un ejemplo de bloqueo de cardán, donde el cubo tiene las siguientes rotaciones:

Figura 2.12
Ejemplo de bloqueo de cardán.



Nota: Extraído de (Vince, 2011).

- Primero, 90° alrededor del eje z.
- Luego, 90° alrededor del eje x.
- Finalmente, se intenta rotar el cubo alrededor del eje y.

Debido a las rotaciones previas, el cubo de la figura 2.12 se encuentra en una posición donde los ejes de rotación se han alineado de tal forma que la rotación alrededor del eje y ya no es independiente. Esto es el bloqueo de cardán, donde una de las rotaciones ya no es posible debido a la alineación de los ejes (Vince, 2011).

Para manejar el bloqueo de cardán, se verifica si la magnitud de $-\sqrt{r_{11}^2 + r_{21}^2}$ de la matriz 2.15 es muy pequeña, y dependiendo de esta condición es posible utilizar el ”problema inverso“ (que encuentra los ángulos que producen dicha rotación) para encontrar un conjunto de ángulos de rotación, que resulten en una orientación específica de un objeto en el espacio tridimensional (Mecharithm, 2021).

Si $C_\theta = 0$ y $\theta = \pm 90°$, se tiene que la matriz R se reducirá a la matriz siguiente:

$$R = \begin{bmatrix} 0 & S_{\psi-\phi} & C_{\psi-\phi} \\ 0 & C_{\psi-\phi} & -S_{\psi-\phi} \\ \pm 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{2.19}$$

Si de la matriz 2.19, se toma $r_{31} = -1$ cuando $\theta = 90^0$, entonces se tiene como resultado la expresión 2.20.

$$\begin{cases} \psi - \phi &= \text{atan2}(r_{22}, r_{12}) \\ \psi - \phi &= \text{atan2}(r_{13}, -r_{23}) \end{cases} \tag{2.20}$$

Si en el sistema 2.20, $\phi = 0$, entonces se tiene la expresión 2.21.

$$\psi = \text{atan2}\,(r_{22}, -r_{23}) = \text{atan2}\,(r_{22}, r_{12}) = \text{atan2}\,(r_{13}, -r_{23}) \tag{2.21}$$

2.2.10. Lógica difusa

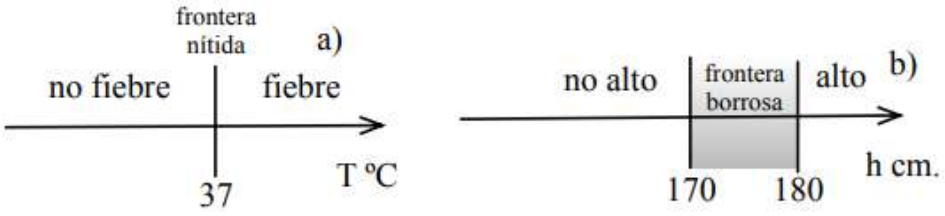
La lógica difusa es una rama de la inteligencia artificial que proporciona un lenguaje formal para representar el conocimiento, realizar inferencias y razonamientos. Está basada en las matemáticas, especialmente en la lógica (Manrique Gamo & Suárez de Figueroa Baonza, 2017). Permite llegar a conclusiones razonadas a partir de información imprecisa y se adapta de manera robusta a situaciones reales, a diferencia de la lógica clásica. Esto se debe a que el ser humano no toma decisiones basadas en la lógica clásica, sino que su comportamiento es, en muchos casos, más bien difuso (UNAM, 2021).

Conjuntos difusos

Una diferencia con la teoría clásica de conjuntos, donde los elementos pertenecen o no a un determinado conjunto y donde las fronteras son nítidas y precisas, es que la teoría de

conjuntos difusos introduce la idea de pertenencia vaga generando una frontera imprecisa y difusa (Manrique Gamo & Suárez de Figueroa Baonza, 2017). Un ejemplo de fronteras nítidas e imprecisas se muestra en la figura 2.13.

Figura 2.13
Fronteras nítidas e imprecisas.



Nota: Extraído de (Manrique Gamo & Suárez de Figueroa Baonza, 2017).

Sea el conjunto difuso A, definido en la expresión 2.22.

$$A = \{(X, \mu_A(X)) | X \in U\} \tag{2.22}$$

Donde X representa los elementos del universo U, y $\mu_A(X)$ es la función de pertenencia que asigna a cada elemento X un grado de pertenencia al conjunto difuso A. Este grado de pertenencia es un valor en el intervalo $[0, 1]$, indicando el grado en el que el elemento X pertenece al conjunto difuso.

Funciones de pertenencia

Una función de pertenencia es una curva que define cómo cada punto de un espacio de entrada es mapeado a un valor de pertenencia entre 0 y 1. El espacio de entrada es usualmente referido como el universo de discurso (MathWorks,).

Las siguientes funciones de pertenencia que se mencionan a continuación, son las más utilizadas en la creación de modelos y conjuntos difusos.

1. Función triangular, definida por 2.23.

$$\mu(x) = \begin{cases} 0, & x \leq a \\ \frac{x-a}{m-a}, & a < x \leq m \\ \frac{b-x}{b-m}, & m < x \leq b \\ 0, & b < x \end{cases} \tag{2.23}$$

2. Función trapezoidal, definida por 2.24.

$$\mu(x) = \begin{cases} 0, & x \leq a \\ \frac{x-a}{b-a}, & a < x \leq b \\ 1, & b < x \leq c \\ \frac{d-x}{b-c}, & c < x \leq d \\ 0, & d < x \end{cases} \tag{2.24}$$

3. Función de saturación lineal en forma de S, definida por 2.25.

$$\mu(x) = \begin{cases} 0, & x \leq a \\ \frac{x-a}{m-a}, & a < x < m \\ 0, & m \leq x \end{cases} \tag{2.25}$$

4. Función de saturación lineal en forma de Z. Esta función se puede definir como la inversa de la función de saturación lineal en forma de Z.

5. Función sigmoide, definida por 2.26.

$$\mu_s(x) = \begin{cases} 0, & x \leq a \\ 2(\frac{x-a}{c-a})^2, & a \leq x \leq \frac{a+c}{2} \\ 1 - 2(\frac{x-a}{c-a}), & \frac{a+c}{2} \leq x \leq c \\ 0, & c \leq x \end{cases} \tag{2.26}$$

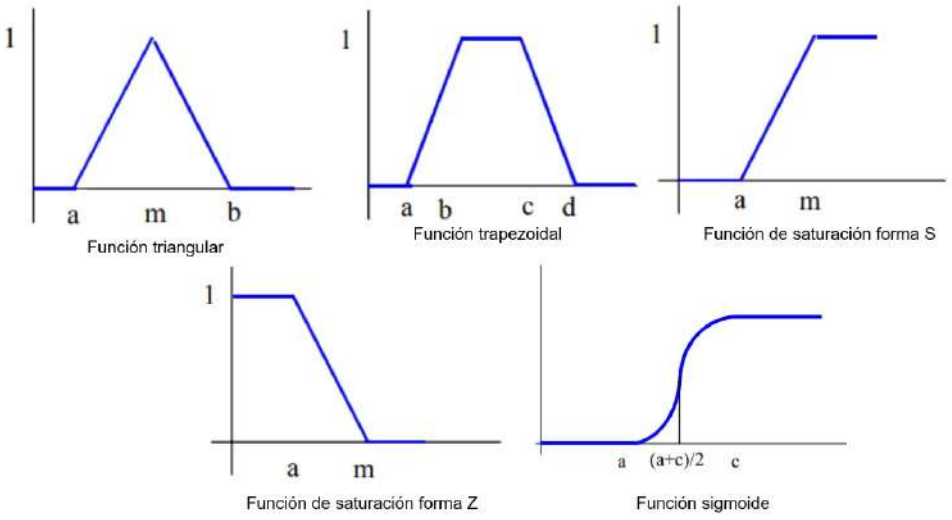
6. Función gaussiana, definida por 2.27.

$$\mu(x) = e^{\frac{-(x-c)^2}{2\sigma^2}} \tag{2.27}$$

Donde σ es la desviación estándar y c es el valor del punto medio en el dominio.

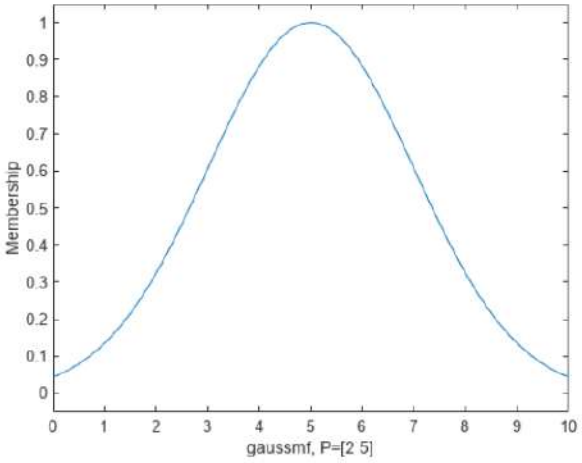
La figura 2.14 muestra las representaciones gráficas de las funciones de pertenencia triangular, trapezoidal, forma S, forma Z y sigmoide, mientras que la figura 2.15 muestra la representación gráfica de la función gaussiana.

Figura 2.14
Representación gráfica de las funciones de pertenencia.



Nota: Extraído de (UNAM, 2021).

Figura 2.15
Representación gráfica de la función gaussiana. Con $\sigma = 2$ y $c = 5$



Nota: Extraído de (MathWorks,).

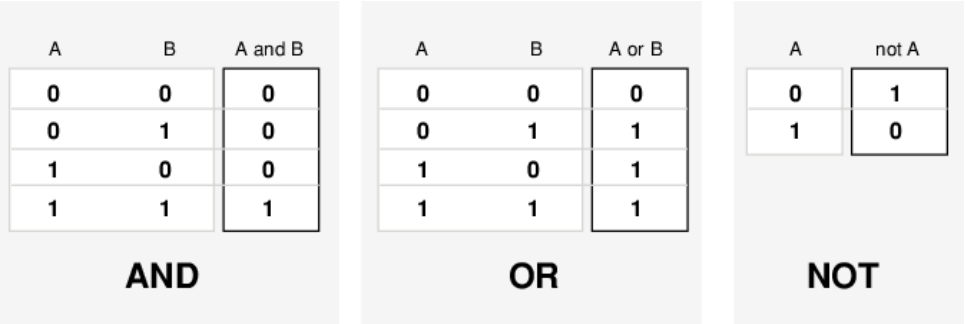
Operadores difusos

Una vez definidas las funciones de pertenencia, es posible realizar operaciones lógicas, como la intersección, la unión o el complemento, las cuales se efectúan a partir de la distribución definida y asociada para el conjunto difuso. Estas operaciones pueden asociarse a las operaciones lógicas de conjunción ($\wedge$), disyunción ($\vee$) y negación ($\neg$) (Manrique Gamo & Suárez de Figueroa Baonza, 2017).

Al ser la lógica difusa un superconjunto de la lógica booleana, se pueden considerar las siguientes tablas de verdad de la figura 2.16, tomando los valores más extremos, donde 1 representa lo completamente verdadero y 0 lo completamente falso. Sin embargo, considerando que en la lógica difusa existe un grado de pertenencia y que los valores pueden ser números reales en el rango de 0 a 1, surgen equivalentes con las operaciones más utilizadas de T-norma min y T-conorma max, para reemplazar las operaciones de Y y O, respectivamente, mientras que la operación de negación es reemplazada por $1 - A$ (ver figura 2.17) (MathWorks,).

Como ejemplo, consideremos dos funciones de pertenencia $\mu_p(x)$ y $\mu_q(x)$. A estas funciones se les pueden aplicar las operaciones lógicas de intersección, unión y complemento, como se ilustra en la figura 2.18.

Figura 2.16
Tabla de verdad para las operaciones lógicas Y, O, y la negación.



A	B	A and B
0	0	0
0	1	0
1	0	0
1	1	1

AND

A	B	A or B
0	0	0
0	1	1
1	0	1
1	1	1

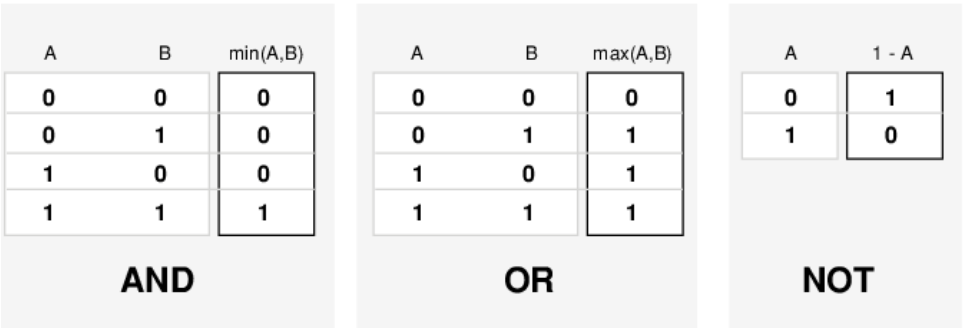
OR

A	not A
0	1
1	0

NOT

Nota: Extraído de (MathWorks,).

Figura 2.17
Tabla de verdad para las operaciones lógicas min, max, y la negación.



A	B	min(A,B)
0	0	0
0	1	0
1	0	0
1	1	1

AND

A	B	max(A,B)
0	0	0
0	1	1
1	0	1
1	1	1

OR

A	1 - A
0	1
1	0

NOT

Nota: Extraído de (MathWorks,).

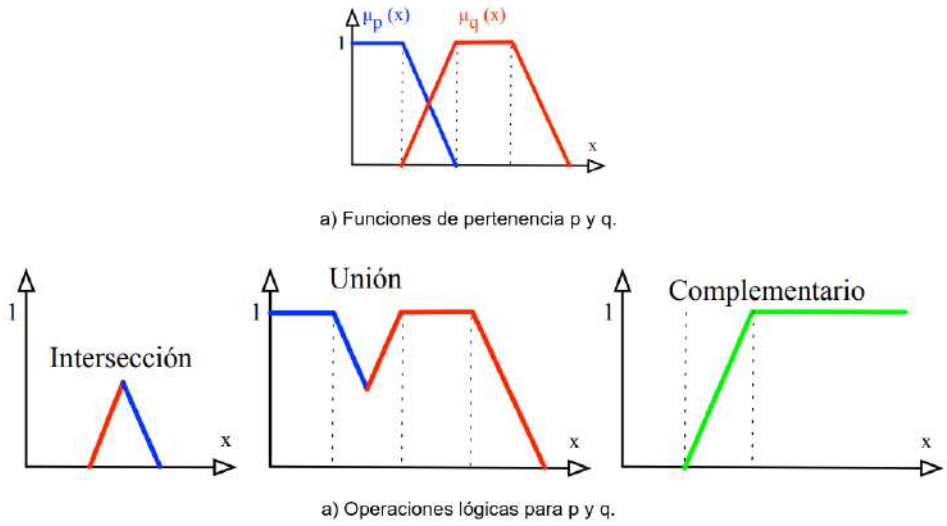
Reglas difusas

Las reglas difusas son un componente importante a la hora de modelar las decisiones y el comportamiento del sistema. Estas reglas se expresan en la forma $si - entonces$, comenzando con la pertenencia de una variable a un conjunto difuso. A partir de esta pertenencia, es posible construir proposiciones difusas utilizando los conectores lingüísticos Y, O y la negación NO (González, 2015), que, como se mencionó, son interpretados como la unión, intersección y complemento.

Las reglas difusas tienen la siguiente estructura simbólica 2.28.

Figura 2.18

Ejemplo de funciones de pertenencia y aplicación de los operadores difusos.



Nota: Extraído de (Manrique Gamo & Suárez de Figueroa Baonza, 2017).

$$Si\ ProposicionDifusa\ entonces\ ProposicionDifusa \tag{2.28}$$

Donde cada $ProposicionDifusa$ puede ser una proposición de pertenencia simple o compuesta. Por ejemplo, una regla simple podría ser la expresión 2.29.

$$p:\ Si\ X\ es\ A\ entonces\ Y\ es\ B \tag{2.29}$$

Donde el antecedente (Si) y el consecuente (Entonces) de una regla pueden incluir diversas o múltiples proposiciones difusas, lo que permite modelar relaciones más complejas entre variables (González, 2015).

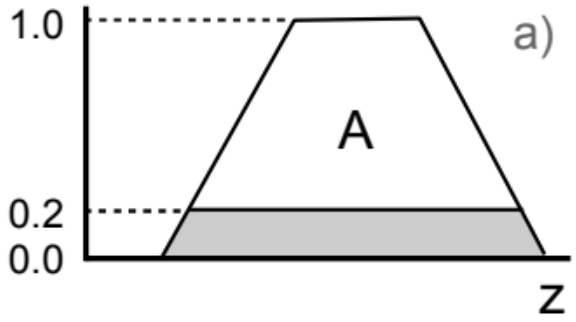
Inferencia difusa

La inferencia difusa es el proceso que permite obtener un valor de salida de un sistema difuso. El método más utilizado es el de Mamdani, el cual consiste en emplear cuatro pasos: fuzzificación de las variables de entrada, evaluación de las reglas, agregación de las salidas de

las reglas y defuzzificación (González, 2015).

La fuzzificación es un proceso que transforma valores nítidos en difusos. Se define un dominio específico (universo de discurso) para las variables a emplear y se establecen sus significados. Para cada variable, se determina el rango de valores numéricos que pueden tomar, así como sus unidades de medida. Además, se asignan etiquetas lingüísticas que representan los diferentes valores cualitativos que puede adoptar cada variable. Estas etiquetas se definen mediante las funciones de pertenencia, las cuales determinan el grado en que un valor pertenece a un conjunto difuso (Manrique Gamo & Suárez de Figueroa Baonza, 2017). Por ejemplo, para las variables x e y, se calculará el grado de pertenencia dentro de los universos X e Y, respectivamente. Esto permite que los valores precisos se conviertan en representaciones difusas que se pueden procesar por un sistema de lógica difusa (González, 2015).

Figura 2.19
Conjunto difuso recortado.



Nota: Extraído de (González, 2015).

Para llevar a cabo la evaluación de las reglas, primero es necesario definir un conjunto de reglas difusas que vinculen las variables entre sí utilizando operadores lógicos difusos. En el proceso de evaluación de estas reglas, se toman las entradas fuzzificadas y se aplican a los antecedentes de las reglas difusas. En el caso de que una regla tenga múltiples antecedentes, se emplean operadores lógicos como *Y* u *O* para combinar estos antecedentes en un único valor que represente el resultado de la evaluación. Este valor, conocido como el valor de verdad, se aplica al consecuente de la regla, realizando un recorte que lo ajusta dependiendo del valor de verdad del antecedente (ver figura 2.19) (González, 2015).

El proceso de agregación de las salidas une todas las reglas combinando las funciones de pertenencia de todos los consecuentes recortados previamente, con el fin de obtener un único conjunto difuso (González, 2015).

Defuzzificación

Para obtener el resultado final, generalmente este debe ser expresado como un valor nítido a partir del único conjunto difuso que se obtiene tras el proceso de agregación de salidas, en un proceso conocido como defuzzificación. Existen diversos métodos para realizar esta tarea, pero el más utilizado es el del centro de gravedad o centroide (González, 2015).

El método del centroide consiste en calcular el punto donde una línea vertical divide el conjunto en dos áreas con igual masa, dando como resultado un valor numérico (González, 2015). Dada la función de pertenencia genérica $\mu_p(x)$ y un conjunto de puntos de discretización de la función $\{x_1, x_2, \ldots, x_n\}$, elegidos de forma equidistante, el centro de gravedad se define con la expresión 2.30 (Manrique Gamo & Suárez de Figueroa Baonza, 2017).

$$Centroide = \frac{\sum_{i=1}^{n} \mu_p(x_i) x_i}{\sum_{i=1}^{n} \mu_p(x_i)} \tag{2.30}$$

2.2.11. Interpolación

La interpolación es una técnica de estimación utilizada en el análisis numérico para generar nuevos puntos de datos dentro del rango de un conjunto de puntos conocidos. En áreas como la ingeniería y la investigación, los datos obtenidos a partir de muestreos o experimentos representan los valores de una función. A menudo, es necesario estimar el valor de la función para un punto intermedio de la variable independiente, lo cual se logra mediante la interpolación (Sabry & Costa, 2024).

Interpolación lineal

Sea $f(x) = x$. Para un valor de x dentro del intervalo $[x_0, x_1]$, el valor de $f(x)$ a lo largo de la línea recta se obtiene mediante la ecuación 2.32, que resulta de despejar la expresión 2.31 (Sabry & Costa, 2024).

$$\frac{f(x) - f(x_1)}{x - x_0} = \frac{f(x_1) - f(x_0)}{x_1 - x_0} \tag{2.31}$$

$$f(x) = f(x_0) + (f(x_1) - f(x_0))\frac{x - x_0}{x_1 - x_0} \tag{2.32}$$

Interpolación bilineal

La interpolación bilineal es una técnica utilizada para aproximar funciones de dos variables, como x e y. Esta técnica consiste primero en realizar una interpolación en la dirección x y luego en y (Sabry & Costa, 2024).

Para determinar el valor de una función f en la posición (x, y), se asume que se conocen los valores de f en cuatro puntos: (x_0, y_0), (x_0, y_1), (x_1, y_0) y (x_1, y_1).

Sean x_d e $y_d$expresados por 2.33 y 2.34:

$$\begin{cases} 1 - x_d = \frac{x_1 - x}{x_1 - x_0}, \\ x_d = \frac{x - x_0}{x_1 - x_0}. \end{cases} \tag{2.33}$$

$$\begin{cases} 1 - y_d = \frac{y_1 - y}{y_1 - y_0}, \\ y_d = \frac{y - y_0}{y_1 - y_0}. \end{cases} \tag{2.34}$$

Realizando la interpolación lineal en el eje x en los puntos y_0 e y_1, se obtienen las

expresiones contenidas en 2.35.

$$\begin{cases} f(x, y_0) = f(x_0, y_0)(1 - x_d) + f(x_1, y_0)x_d, \\ f(x, y_1) = f(x_0, y_1)(1 - x_d) + f(x_1, y_1)x_d. \end{cases} \tag{2.35}$$

Finalmente, de manera análoga, se interpola a lo largo del eje y utilizando los resultados de 2.35, obteniendo el resultado con la expresión 2.36.

$$f(x, y) = f(x, y_0)(1 - y_d) + f(x, y_1)y_d. \tag{2.36}$$

Este proceso permite aproximar el valor de la función f en cualquier punto (x, y) dentro del área delimitada por los cuatro puntos conocidos.

Interpolación trilineal

La interpolación trilineal es una extensión de la interpolación bilineal para 3 dimensiones. Para esto, se tienen 8 valores de la función $f(x, y, z)$ en los puntos (x_i, y_i, z_i) con $i, j, k \in \{0, 1\}$. Estos valores representan los vértices de un prisma rectangular y permiten mapear el valor de interpolación suave de la función $f(x, y, z)$ (Buss, 2003).

De manera general, la interpolación trilineal se define con la expresión 2.37.

$$f(x, y, z) = \sum_{i,j,k=0,1} w_i(x_d)w_j(y_d)w_k(z_d)f(x_i, y_i, z_i) \tag{2.37}$$

Donde x_d y y_d están definidos por las expresiones 2.33 y 2.34 respectivamente, y z_d por la expresión 2.38.

$$z_d = \frac{z - z_0}{z_1 - z_0} \tag{2.38}$$

Además, los valores de $w_n(a)$ están definidos por la expresión 2.39.

$$w_n(a) = \begin{cases} 1 - a, & n = 0 \\ a, & n = 1 \end{cases} \tag{2.39}$$

Al igual que en la interpolación bilineal, primero se realiza la interpolación a lo largo del eje x, mediante la expresión 2.40.

$$\begin{cases} f(x, y_0, z_0) = f(x_0, y_0, z_0)(1 - x_d) + f(x_1, y_0, z_0)x_d \\ f(x, y_0, z_1) = f(x_0, y_0, z_1)(1 - x_d) + f(x_1, y_0, z_1)x_d \\ f(x, y_1, z_0) = f(x_0, y_1, z_0)(1 - x_d) + f(x_1, y_1, z_0)x_d \\ f(x, y_1, z_1) = f(x_0, y_1, z_1)(1 - x_d) + f(x_1, y_1, z_1)x_d \end{cases} \tag{2.40}$$

Posteriormente, se realiza la interpolación a lo largo del eje y, mediante la expresión 2.41.

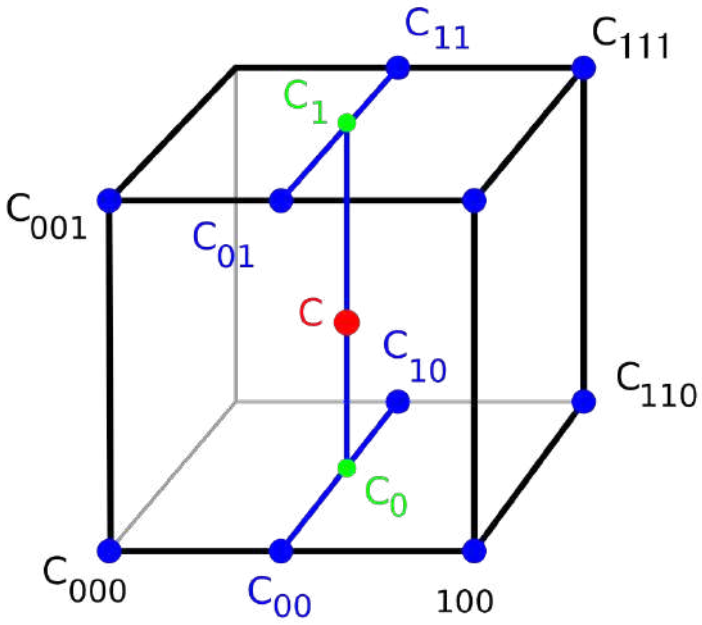
$$\begin{cases} f(x, y, z_0) = f(x, y_0, z_0)(1 - y_d) + f(x, y_1, z_0)y_d \\ f(x, y, z_1) = f(x, y_0, z_1)(1 - y_d) + f(x, y_1, z_1)y_d \end{cases} \tag{2.41}$$

Finalmente, se realiza la interpolación a lo largo del eje z, obteniendo la interpolación trilineal con la expresión 2.42.

$$f(x, y, z) = f(x, y, z_0)(1 - z_d) + f(x, y, z_1)z_d \tag{2.42}$$

En la figura 2.20 se muestra una representación de la interpolación trilineal, donde $c = f(x, y, z)$.

Figura 2.20
Representación gráfica de la interpolación trilineal. Se observan ocho puntos del prisma rectangular que rodean el punto de interpolación C.



Nota: Tomado de https://www.pngwing.com/es/free-png-nayjn/download.

2.2.12. Sistemas embebidos

(Aguirre & Giraldo, 2014) define un sistema embebido (SE) como una solución de computación especializada y diseñada para llevar a cabo funciones específicas en tiempo real. Estos SE están orientadas a cubrir necesidades determinadas, lo que lo diferencia de computadoras convencionales. Una característica clave de los SE es que tienen integrados todos sus componentes (procesador, memoria, interfaz de entrada/salida, entre otros) en una misma placa base.

Un ejemplo de SE son las computadoras de placa única (SBC, por sus siglas en inglés, Single Board Computer), que se volvieron populares gracias a los Raspberry Pi y que otros fabricantes han desarrollado, surgiendo así placas como Orange Pi, Banana Pi, Nvidia Jetson, entre otros. Estas placas generalmente están basadas en la arquitectura ARM 64 (por sus siglas en inglés, Advanced RISC Machine), también llamada aarch64. La figura 2.21 muestra un SBC perteneciente a la familia de Orange Pi.

Figura 2.21
Orange pi 5.

Nota: Extraído de (OrangePi, 2022).

2.2.13. Unidad de procesamiento neuronal (NPU)

Es una unidad de procesamiento que trabaja de forma similar al cerebro humano y tiene un diseño que se basa en los núcleos Tensor presentes en tarjetas gráficas de NVIDIA, para acelerar tareas relacionadas con la inteligencia artificial y el aprendizaje profundo. La NPU es capaz de realizar operaciones matriciales y procesamiento en paralelo de manera eficiente, siendo algunas de las aplicaciones como el reconocimiento de voz, detección de objetos, desenfoque de fondos en videollamadas, entre otros, donde la computación paralela de datos es fundamental para un rendimiento óptimo (Samsung, 2019; TechRadar, 2024). El rendimiento de estas unidades generalmente se mide en TOPs (Tera operaciones por segundo, 10^{12} de operaciones realizadas cada segundo).

2.2.14. Herramientas computacionales

A continuación se listan las principales herramientas computacionales empleadas en esta tesis:

- Python. Lenguaje de programación de alto nivel, muy usado en el campo de la inteligencia artificial.

- Mediapipe. Marco de desarrollo que facilita la detección de puntos faciales y rostro.
- OpenCV. Biblioteca para procesar imágenes y videos, disponible para multiplataforma.
- Pytorch. Biblioteca que proporciona un marco de desarrollo para de inteligencia artificial, especialmente el aprendizaje profundo y redes neuronales.
- Pandas. Biblioteca que permite interactuar con archivos tipo tabla como EXCEL, CSV, entre otros.
- PyGame. Biblioteca que permite reproducir audio.
- RKNN_toolkit2. Biblioteca de desarrollo para realizar conversiones, inferencia y rendimiento de modelos de inteligencia artificial en una PC o en una NPU de Rockchip, más detalles en el anexo G.
- RKNN_toolkit2_lite. Proporciona una interfaz de programación en python para las NPU de Rockchip, que despliega modelos RKNN y acelera la implementación de aplicaciones de IA, más detalles en la sección 4.4.
- RKNPU. Es el controlador de las NPU de Rockchip, que interactúa con el hardware NPU, más detalles en la sección 4.4.
- WiringOP. Una herramienta que permite controlar las interfaces de entrada y salida en computadoras de placa única (SBCs) como Raspberry Pi, Orange Pi, Banana Pi, entre otras.

Capítulo 3

Diseño del sistema detector de distracción

3.1. Requerimientos del sistema

Para lograr los objetivos establecidos en esta tesis, se han definido los siguientes requisitos.

3.1.1. Requisitos de hardware

- Cámara RGB para capturar continuamente la cabeza del conductor.
- Conjunto de parlantes con un consumo máximo de 3 W, utilizados para emitir alertas sonoras en caso de distracción del conductor.
- Indicadores visuales que muestren el estado de atención del conductor.
- Sistema embebido (SBC) con un una NPU dedicada para implementaciones de modelos de inteligencia artificial y con suficiente capacidad de procesamiento para ejecutar algoritmos de estimación de postura y detección de distracción en tiempo real, con un consumo máximo de 20 W.
- Fuente de alimentación que adapte el voltaje de la batería del vehículo a 5 V, y que sea capaz de proporcionar la corriente necesaria y entregar al menos 20 W de potencia para alimentar todo el sistema.

3.1.2. Requisitos de software

- Sistema operativo: Utilizar un sistema operativo compatible con todas las librerías necesarias para el procesamiento de imágenes, redes neuronales y manejo de hardware embebido (por ejemplo, Linux).

- Modelos de redes neuronales: Implementar modelos preentrenados (como redes neuronales convolucionales) para estimar los ángulos de rotación de la cabeza (yaw, pitch, roll) a partir de imágenes capturadas en tiempo real.

- Optimización de las CNN para NPU: Adaptar y transformar los modelos de redes neuronales para utilizar la Unidad de Procesamiento Neuronal (NPU) del sistema embebido, mejorando así la eficiencia y reduciendo el consumo de potencia.

- Detección de distracción: Desarrollar algoritmos optimizados para detectar distracción en el conductor, basados en la postura de la cabeza.

- Indicadores y alerta: Implementar indicadores visuales y alarma auditiva para notificar al conductor cuando se detecte una distracción.

- Integración de hardware: Implementar software que permita la integración fluida entre la cámara, los indicadores visuales, los parlantes y el sistema embebido, garantizando un funcionamiento cohesivo del sistema.

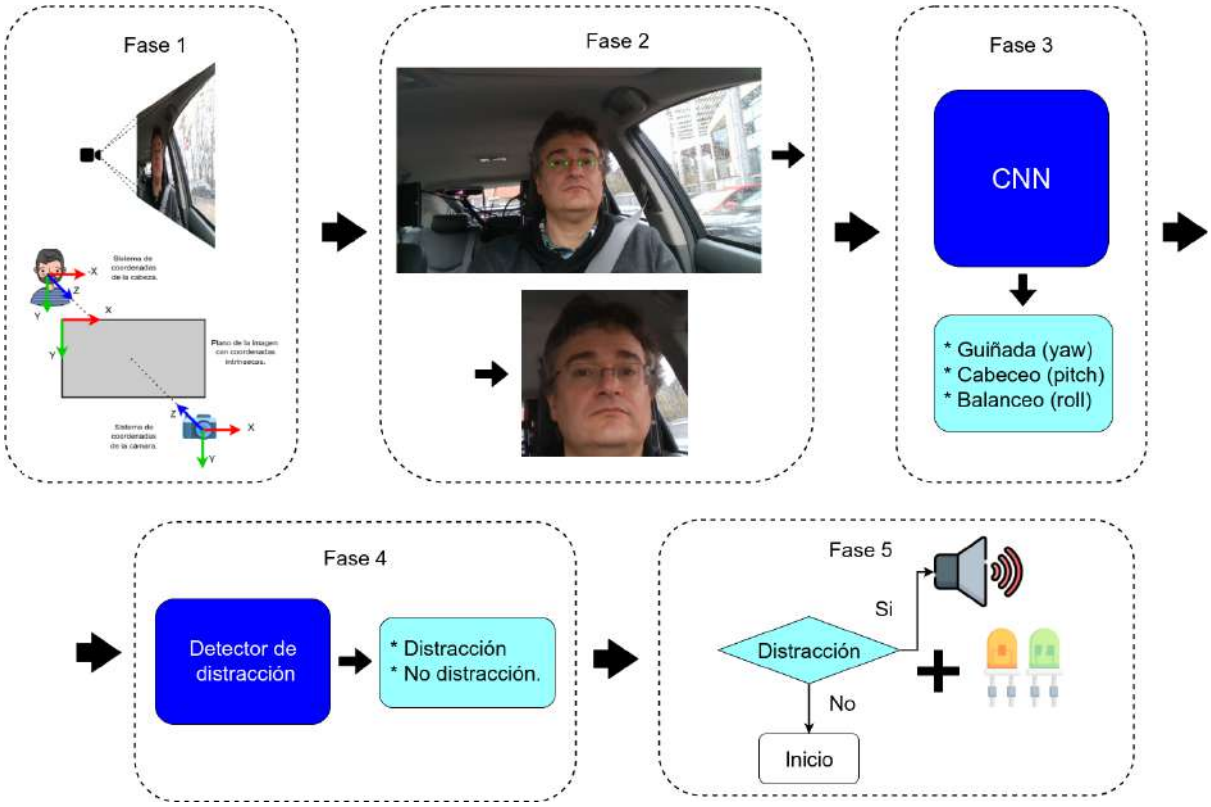
3.1.3. Requisitos de prueba y validación del prototipo

- Validación en un entorno real controlado: Probar el prototipo en un entorno real controlado para evaluar su capacidad de detección de distracción del conductor bajo condiciones controladas.

3.2. Lógica del sistema

El sistema de detección de distracción sigue una lógica secuencial que integra varios componentes. Estos componentes se dividen en 5 fases que describen el funcionamiento del sistema. La figura 3.1 muestra las fases y la lógica general del sistema detector de distracción.

Figura 3.1
Fases del sistema detector de distracción.



A continuación se describe el flujo lógico del sistema.

3.2.1. Fase 1: Adquisición de datos

La adquisición de las imágenes se realiza mediante una sola cámara RGB, que proporciona los datos de entrada y captura en todo momento los movimientos del conductor del vehículo.

Esta recolección de datos se lleva a cabo en tiempo real y se toma en cuenta el ángulo de la cámara, enfocándose principalmente en el conductor (ver figura 3.2). Además, está limitada

a 30 cuadros por segundo.

Figura 3.2
Adquisición de imágenes.

3.2.2. Fase 2: Detección de puntos faciales y extracción de región de interés (ROI)

Cada cuadro o fotograma obtenido en la fase previa se utiliza para extraer la región de interés, que corresponde a la cabeza del conductor. Este proceso comienza con la detección de puntos faciales mediante el uso de la librería Mediapipe, la cual identifica 468 puntos en el rostro (ver figura 3.3). De estos puntos, solo cuatro son de interés, ya que están ubicados en los extremos de los ojos. Estos puntos tienen las siguientes numeraciones: $p1 = 130$, $p2 = 243$, $p3 = 463$, y $p4 = 359$. Se extraen sus coordenadas, denotadas como $p_i = (x_i, y_i)$ para $i \in \mathbb{Z}$ y $1 \leq i \leq 4$, y se calcula el promedio de los puntos utilizando la ecuación 3.1, la cual representa el centro de la cabeza.

$$p_m = \frac{1}{4} \sum_{i=1}^{4} p_i \tag{3.1}$$

Donde $p_m = (x_m, y_m)$ representa el punto medio de la cabeza.

Figura 3.3
Mapa de puntos de referencia faciales de Mediapipe.

Nota: Extraído de (Mediapipe, 2023).

Una vez determinado el punto medio de la cabeza, este se utiliza para definir la región de interés (cabeza del conductor). Para ello, se calcula la distancia de los puntos p_i con respecto a p_m mediante la ecuación 3.2, y posteriormente se obtiene la distancia promedio de dichos puntos (ecuación 3.3). Esto permite obtener un valor no aleatorio que varía en función de los puntos ubicados en las esquinas de los ojos.

$$d_i = \sqrt{(x_i - x_m)^2 + (y_i - y_m)^2} \tag{3.2}$$

$$d_{prom} = \frac{1}{4} \sum_{i=1}^{4} d_i \tag{3.3}$$

Donde d_i es la distancia para un punto determinado con relación al punto medio p_m, d_{prom} es la distancia promedio.

A esta distancia promedio d_{prom}, se le aplica un factor multiplicativo k (este factor debe

ser ajustado para cada cámara, y su valor por defecto es de 6) para hallar el tamaño del recorte r, ecuación 3.4, con el que se puede realizar un ajuste en el tamaño de la zona de interés y obtener sus coordenadas.

$$r = \lfloor k d_{prom} \rfloor, \quad k \geq 1 \tag{3.4}$$

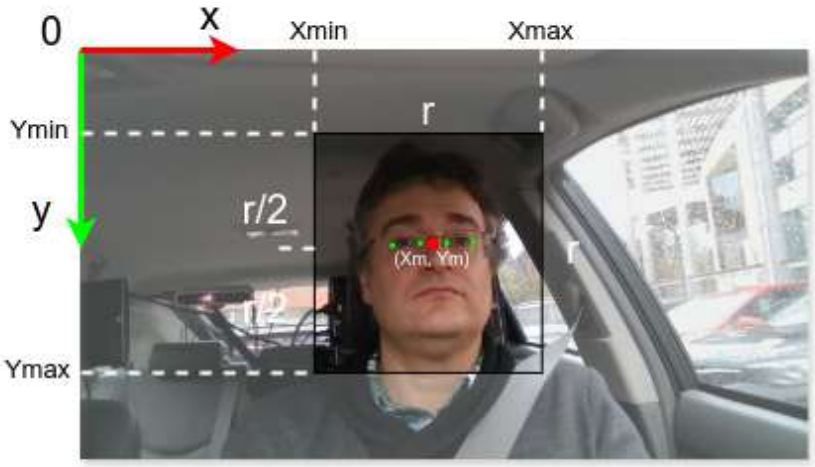
Por lo tanto, las coordenadas de la zona de interés se expresan mediante un punto mínimo $p_{min} = (x_{min}, y_{min})$ y un punto máximo $p_{max} = (x_{max}, y_{max})$, con las expresiones 3.5 y 3.6 respectivamente.

$$(x_{min}, y_{min}) = (x_m - \frac{r}{2}, y_m - \frac{r}{2}) \tag{3.5}$$

$$(x_{max}, y_{max}) = (x_{min} + r, y_{min} + r) \tag{3.6}$$

La figura 3.4 muestra de manera gráfica el proceso de extracción de la zona de interés.

Figura 3.4
Extracción de la zona de interés (ROI).

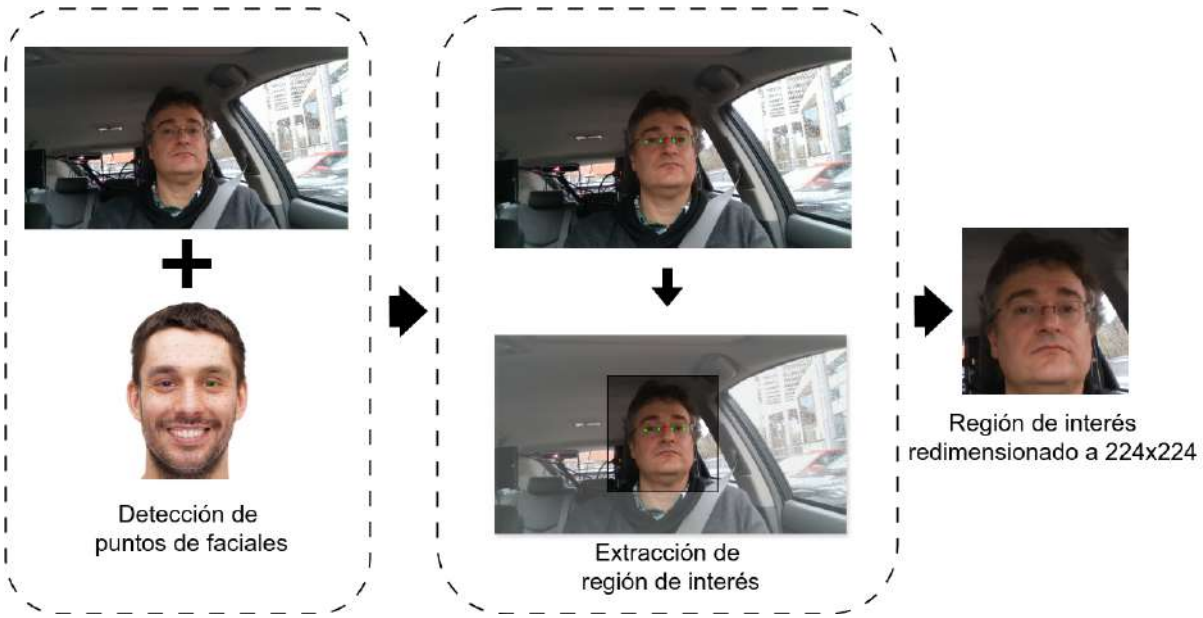


Al realizar el recorte final de la zona de interés, se deben considerar las posibles variaciones que puedan extender dicha zona más allá del tamaño de la imagen, causando que salga

parcial o totalmente del foco de la cámara. Esto podría generar errores y/o deformaciones en la representación de la cabeza del conductor. Para evitarlo, se establecen las siguientes condiciones: $(x_{min}, y_{min}) \geq (0, 0)$ y $(x_{max}, y_{max}) \leq (ancho, alto)$, las cuales garantizan que la zona de interés permanezca siempre dentro del tamaño de la imagen. Posteriormente, se redimensiona la zona de interés a un tamaño de 224x224 píxeles, finalizando así la fase 2.

El proceso correspondiente a la fase 2 se ilustra en la figura 3.5, ofreciendo una representación gráfica de lo descrito previamente.

Figura 3.5
Procedimiento llevado a cabo en la fase 2.



3.2.3. Fase 3: Estimación de la postura de la cabeza

En la estimación de la postura de la cabeza, existen varios métodos basados en redes neuronales convolucionales (CNNs) que pueden emplearse. Algunos de estos hacen uso de matrices de rotación, cuaterniones e incluso combinan enfoques de regresión y clasificación en un solo modelo. Sin embargo, en la presente tesis se opta por el enfoque propuesto por 6DRepNet (Hempel et al., 2022), ya que no está limitado a un rango específico de ángulos, lo que proporciona mayor flexibilidad y lo hace escalable en la tarea de estimación de la postura de la cabeza.

El método propuesto por (Hempel et al., 2022) consiste en obtener una matriz R de tamaño 3x3 que sea ortogonal, es decir, que cumpla la restricción $RR^T = I$, donde R^T es la transpuesta de R e I es la matriz identidad. La matriz R pertenece al grupo de rotación $SO(3)$. En el enfoque seguido por (Hempel et al., 2022), se impone la restricción ortogonal a partir de un vector de tamaño 1x6, donde los valores se ordenan en columnas de tres elementos, y se utiliza el proceso de Gram-Schmidt para mapear estos valores y obtener una matriz ortogonal.

Sin embargo, este paso puede simplificarse generando directamente un vector de tamaño 1x9, que luego se utiliza para formar la matriz R. Esta simplificación es posible debido a que uno de los componentes esenciales de la función de pérdida utilizada en el proceso de entrenamiento (ecuación 2.7) incluye la expresión $R_p R_{gt}^T$. Esta expresión evalúa la ortogonalidad de la matriz R predicha (R_p) en comparación con la matriz de rotación verdadera (R_{gt}), y el objetivo es que el producto resulte en la matriz identidad I.

Para la obtención del vector de tamaño 1x9, se toma un modelo CNN normalmente utilizado para clasificar objetos como base o columna vertebral, al que se le reemplaza la capa de clasificación por otra que proporcione los parámetros que representan a dicho vector, con los que se genera la matriz predicha R_p, representada por la matriz 3.9.

Sea el vector $\vec{v} = [v_1, v_2, v_3, ..., v_9]$, el cual es dividido en 3 vectores, representados en la expresión 3.7.

$$\vec{v}_1 = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \quad , \quad \vec{v}_2 = \begin{bmatrix} v_4 \\ v_5 \\ v_6 \end{bmatrix} \quad , \quad \vec{v}_3 = \begin{bmatrix} v_7 \\ v_8 \\ v_9 \end{bmatrix} \tag{3.7}$$

Los vectores se convierten en vectores unitarios utilizando la expresión 3.8, donde $i = 1, 2, 3$. Este proceso de normalización asegura que cada vector tenga una magnitud igual a 1, manteniendo la misma dirección que el vector original.

$$\hat{v}_i = \frac{\vec{v}_i}{\|\vec{v}_i\|} \tag{3.8}$$

Para obtener la matriz predicha R_p, estos vectores unitarios son colocados uno detrás de otro en forma de columnas, como se muestra en la matriz 3.9.

$$R_p = \begin{bmatrix} | & | & | \\ \hat{v}_1 & \hat{v}_2 & \hat{v}_3 \\ | & | & | \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \tag{3.9}$$

Para el proceso de entrenamiento, se tiene la matriz de rotación verdadera R_{gt}, que se genera a partir de los ángulos de rotación de giro (yaw) ψ, cabeceo (pitch) θ y balanceo (roll) ϕ verdaderos, expresados en radianes y proporcionados por los conjuntos de datos, mediante el uso de las expresiones 2.13 y 2.14, dando como resultado la matriz de rotación verdadera R_{gt} 3.10.

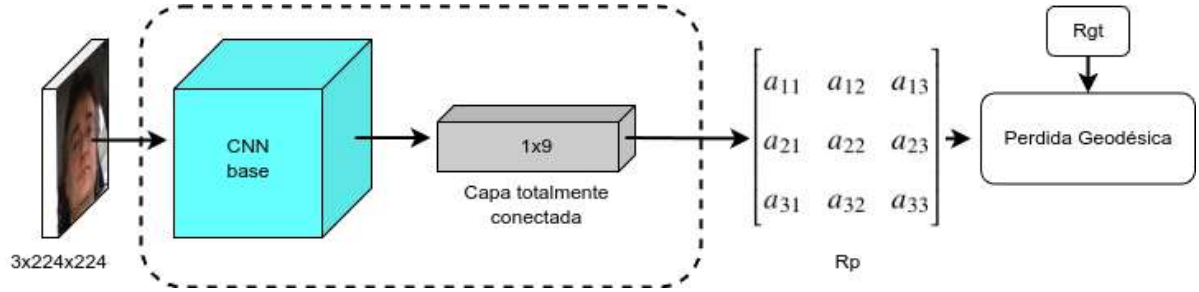
$$R_{gt} = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} \tag{3.10}$$

En el proceso de entrenamiento de la CNN, se hace uso de la pérdida geodésica (ecuación 2.7), la cual busca que L_g sea lo más pequeño posible, es decir, que tienda a cero.

Por otro lado, en el proceso de inferencia o predicción del modelo, la matriz R_p es convertida a ángulos de giro, cabeceo y balanceo mediante el uso de las expresiones 2.16, 2.17 y 2.18. Además, para evitar la ambigüedad en situaciones como el bloqueo de cardán, se utilizan las expresiones 2.20 y 2.21. Estos ángulos están expresados en radianes, por lo que se convierten a grados sexagesimales a través de la expresión 3.11. La figura 3.6 muestra una descripción general del método utilizado.

$$\theta^{\circ} = \frac{\theta rad * 180^{\circ}}{\pi rad} \tag{3.11}$$

Figura 3.6
Descripción del método utilizado: se emplea un modelo CNN como base, modificado en su última capa para generar como salida la matriz de rotación R_p.



3.2.4. Fase 4: Detección de distracción

En esta fase, el sistema se encarga de analizar en tiempo real el estado de atención del conductor para identificar posibles momentos de distracción. Se basa en los ángulos de rotación obtenidos a partir de la estimación de la postura de la cabeza, analizando los valores de estos ángulos para detectar cualquier desviación indicativa de distracción.

Para detectar la distracción con base en los ángulos de rotación, se ha tomado como referencia el método propuesto por (Zhao et al., 2020), descrito en la sección 2.2.3 Este método utiliza un punto base de conducción segura β, el cual se emplea para calcular la distancia espacial que permite determinar un umbral de conducción segura D, que abarca el 90 % de los datos de conducción segura. Posteriormente, se utiliza este umbral para determinar si existe o no distracción.

Con fines de comparación y mejora del enfoque de (Zhao et al., 2020), se propone una modificación que considera tres umbrales distintos de conducción segura: izquierda, centro y derecha. Esta propuesta busca identificar situaciones como la observación de los espejos retrovisores o los giros del conductor, que son acciones normales de conducción, pero que podrían parecer distracciones bajo el método original. Además, se plantea un enfoque basado en lógica difusa que permita integrar las acciones de distracción y conducción segura mediante

reglas difusas, con el fin de evaluar su viabilidad para detectar distracciones de manera más precisa.

Para implementar el método modificado y definir las reglas del modelo difuso, se requiere un conjunto de datos que permita llevar a cabo este propósito. Esto se puede lograr dividiendo el conjunto de datos de conducción segura, utilizando un umbral promedio D_{prom}, basado en la expresión 3.12 (ver expresión 2.1 para la distancia espacial d_i y donde n es el número total de cuadros en el conjunto de datos de conducción segura).

$$D_{prom} = \frac{1}{n} \sum_{i=1}^{n} d_i \tag{3.12}$$

Con este umbral promedio, los datos se dividen en 3 categorías: derecha, centro e izquierda, lo que facilita un análisis más detallado a la hora de detectar la distracción.

Método de umbralización modificado

Este enfoque toma como base el método de (Zhao et al., 2020), del cual se reutiliza el umbral de seguridad hallado D para la categoría de centro. El siguiente paso es calcular los umbrales de conducción segura para las categorías izquierda y derecha. Para ello, se siguen los pasos establecidos para detección de distracción en la sección 2.2.3, utilizando las categorías derivadas del conjunto de datos de conducción segura para obtener dos puntos de referencia: β_{izq} y β_{der}. Con estos puntos, se calculan las distancias espaciales para cada categoría, d_{izq_i} y d_{der_i}. Los umbrales D_{izq} y D_{der} se establecen usando el promedio de las distancias espaciales obtenidas mediante la expresión 3.13. Esto asegura que la modificación mejora la detección de los casos en los que se presenta la zona segura, sin afectar negativamente la detección de distracción.

$$D_{izq} = \frac{1}{m} \sum_{i=1}^{m} d_i, \quad D_{der} = \frac{1}{p} \sum_{i=1}^{p} d_i \tag{3.13}$$

Donde m y p son los números totales de cuadros para las categorías de izquierda y derecha respectivamente.

Método de lógica difusa

Al igual que en el enfoque anterior, se utilizan las tres categorías generadas a partir del conjunto de datos de conducción segura. Sin embargo, para este enfoque también se consideran las acciones del conductor que se clasifican como distracción. El proceso consta de los siguientes pasos:

1. Fuzzificación: El primer paso es realizar un análisis de los ángulos de rotación (yaw, pitch y roll) a través de gráficos de densidad, tanto para las zonas seguras como para las acciones de distracción. Estos gráficos permiten determinar los límites superiores e inferiores de cada ángulo, lo cual ayuda a definir el universo de discurso para cada uno de ellos. Adicionalmente, se determinan las zonas seguras y las acciones del conductor mediante intervalos. Para esto, son consideradas únicamente las acciones en las que existe movimientos significativos de la cabeza y no hay obstrucción de objetos. Además, se define el universo de discurso con el nombre de ”estado”, en el cual se incluyen las acciones del conductor clasificadas como distracción y las categorías de conducción segura.

    - Las funciones de pertenencia difusa para los universos de discurso de los ángulos de rotación, se establecen utilizando intervalos (por ejemplo, ”intervalo 1“, ”intervalo 2“, ”intervalo 3“, etc.).
    - Para el universo de ”estado“, se definen funciones de pertenencia: ”centro“, ”derecha e izquierda“, y ”distracción“.

2. Base de reglas difusas: Se crean reglas difusas que combinan las entradas mediante operadores lógicos difusos (como AND, OR), generando reglas que describen los estados del conductor. Ejemplos de reglas podrían ser:

- Si ”yaw“ está en ”intervalo 1“ Y ”pitch“ está en ”intervalo 2“ O ”roll“ está en ”intervalo 4“, ENTONCES el conductor está en ”estado“ de ”centro“.
- Si ”yaw“ está en ”intervalo 4“ Y ”pitch“ está en ”intervalo 1“ O ”roll“ está en ”intervalo 1“, ENTONCES el conductor está en ”estado“ de ”distracción“.

3. Inferencia difusa: Se utiliza el método de Mamdani para realizar la inferencia difusa. Este proceso toma las reglas difusas previamente definidas y combina las entradas para obtener una salida difusa que represente el estado del conductor.

4. Defuzzificación: El método de defuzzificación seleccionado es el método del centroide o centro de gravedad. Este toma las salidas difusas y genera un valor nítido.

5. Interpretación: Finalmente, el valor nítido obtenido, se compara con un umbral predefinido dentro del universo de discurso del estado del conductor que agrupa a las zonas seguras como no distracción y las acciones relacionadas con la distracción como distracción. Este umbral simplifica la interpretación del resultado: si el valor es mayor que el umbral, se clasifica al conductor en un estado de distracción; en caso contrario, se clasifica como no distracción. Este proceso convierte un valor continuo en una decisión binaria, lo cual es más práctico y fácil de interpretar en un contexto de detección de distracción en tiempo real.

3.2.5. Fase 5: Alarma e indicación visual

Esta fase es necesaria para alertar al conductor sobre su estado de distracción y ayudar a corregirlo en tiempo real sin generar nuevas distracciones. Consiste en utilizar tanto señales auditivas como visuales para notificar al conductor. La alarma se emite a través de un altavoz que genera una señal sonora en caso de distracción. Paralelamente, la indicación visual está compuesta por dos LEDs: uno rojo que se enciende en caso de distracción y otro verde que permanece activo durante la conducción segura (no distracción). Ambos LEDs proporcionan información visible sobre el estado del conductor. El funcionamiento de la fase 5 se muestra en la figura 3.1.

3.3. Diagrama de flujo del diseño del sistema de detección de distracción

El diagrama de flujo mostrado en la figura 3.7 describe el proceso lógico del diseño del sistema. A continuación, se ofrece una explicación de cada una de las etapas representadas en el diagrama:

1. Inicio: El proceso comienza.

2. Inicialización de dispositivos y controladores: Se inicializan la cámara, la NPU, los GPIO, el sonido y, además, se cargan los archivos y librerías requeridos.

3. Detección de puntos de interés: Se busca detectar correctamente los puntos de interés en el rostro del conductor (comisuras de los ojos). Si no se detectan, se intenta nuevamente con otro cuadro hasta obtener una imagen válida.

4. Cálculo del punto medio de la cabeza y distancia promedio: Se determina el punto medio de la cabeza y se calcula la distancia promedio entre los puntos de interés seleccionados.

5. Cálculo del recorte y sus coordenadas: A partir del punto medio y la distancia promedio, se calculan el tamaño y las coordenadas del recorte que contendrá la zona de interés.

6. Validación de coordenadas: Se verifica si las coordenadas del recorte están dentro de los límites de la imagen. Si están dentro de los límites, el sistema continúa; de lo contrario, se intenta detectar nuevamente los puntos de interés.

7. Recorte de la zona de interés: Si las coordenadas son válidas, se recorta la zona de interés correspondiente.

8. Preprocesamiento para la CNN: La imagen recortada se redimensiona y normaliza para cumplir con los requisitos de entrada de la CNN. Este paso garantiza que los datos sean compatibles con el modelo.

9. CNN (predicción de ángulos): La CNN procesa la imagen preprocesada y predice los ángulos de rotación de la cabeza (yaw, pitch y roll). Estos ángulos describen la postura de la cabeza del conductor.

10. Análisis de ángulos predichos: Se analiza el resultado para interpretar el estado de la cabeza del conductor.

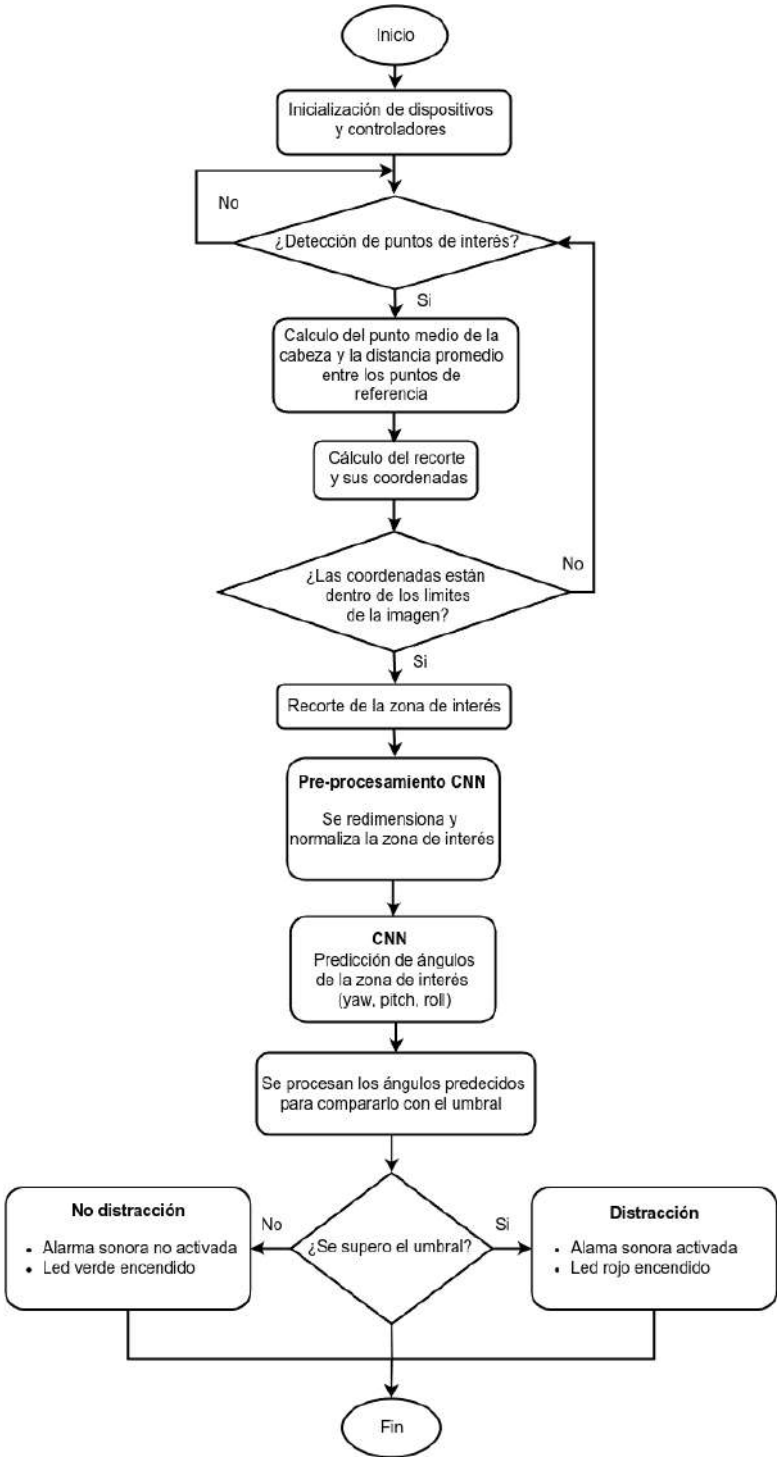
11. Decisión basada en el umbral: Si los valores de los ángulos superan un umbral predefinido, se identifica un posible caso de distracción. Si no se superan, se considera que el conductor está atento.

12. Indicadores y alarma:

    - Si no hay distracción: La alarma sonora no se activa y el LED verde se enciende para indicar que el conductor está atento.
    - Si hay distracción: Se activa una alarma sonora (entre 40 dB a 85 dB, que dependerá del entorno del vehículo, tipo de vehículo, la sensibilidad individual, entre otros.) para alertar al conductor y se enciende el LED rojo, indicando que está distraído.

13. Fin: El proceso concluye.

Figura 3.7
Diagrama de flujo del sistema de detección de distracción.



Capítulo 4

Implementación del sistema detector de distracción

4.1. Componentes del sistema

4.1.1. Cámara

Para la implementación del sistema, se ha seleccionado una cámara RGB genérica (ver figura 4.1) debido a sus especificaciones técnicas. Esta cámara ofrece una resolución máxima de 4k (3840x2160) a 30 FPS, junto con un amplio ángulo de visión panorámica de hasta 90° y una función de enfoque automático, lo cual la convierte en una opción adecuada para el sistema de detección de distracción.

Para una descripción más detallada de las características de la cámara, ver el anexo C.

4.1.2. Sistema embebido

Selección de un microprocesador

Para seleccionar un microprocesador, las características principales a tener en cuenta son que cuente con una NPU y que esté desarrollado con la arquitectura Aarch64 o ARM64, ya

Figura 4.1
Cámara.

que, en términos energéticos, este tipo de arquitectura resulta ser muy eficiente. Además, es importante que esté disponible en una gama de productos y que cuente con un marco de software para el desarrollo de aplicaciones.

En el mercado existen numerosos modelos de procesadores ARM64 que cuentan con una NPU dedicada, desarrollados por distintos fabricantes. Entre ellos se encuentran empresas como Apple, Qualcomm, Huawei, MediaTek, Rockchip, Samsung, Texas Instruments, entre otros, que han lanzado microprocesadores con capacidades avanzadas de inteligencia artificial.

Específicamente, los microprocesadores de Rockchip proporcionan una combinación de rendimiento y precios competitivos para muchas aplicaciones. Además, se destacan por ser eficientes energéticamente y por su buen desempeño en aplicaciones multimedia e inteligencia artificial. Rockchip también colabora de forma constante con fabricantes de dispositivos de consumo, lo que asegura una variedad de productos y un soporte técnico robusto.

Por lo descrito anteriormente, se opta por utilizar un microprocesador de Rockchip. A continuación se describen alguno de ellos.

1. **Rockchip RK3588/RK3588S**: Estos microprocesadores comparten gran parte de sus características, siendo que el RK3588S es una versión recortada del RK3588. Ambos cuentan con una NPU de 3 núcleos, que pueden ser empleados de forma individual o colectiva, una GPU Mali-G610 MP4 y 8 núcleos de CPU. Las características más relevantes de ambos se pueden observar en la tabla 4.1.

2. **Rockchip RK3566/RK3568**: La característica principal de estos microprocesadores es que comparten CPU, GPU y la NPU mononúcleo de 1 TOP (ver tabla 4.2)

3. **Rockchip RK3576**: Este es un microprocesador que fue lanzado recientemente por Rockchip, que cuenta con una NPU de doble núcleo de 6 TOPs capaz de trabajar de manera individual o colectiva (ver tabla 4.2).

Tabla 4.1
Comparación RK3588 y RK3588S.

	RK3588	**RK3588S**
CPU	4xCortex-A76 + 4xCortex-A55	4xCortex-A76 + 4xCortex-A55
GPU	ARM Mali-G610 MP4	ARM Mali-G610 MP4
NPU	6 TOPs, de 3 núcleos con soporte de aceleración para operaciones int4/int8/int16/FP16/BF16/TF32	6 TOPs, de 3 núcleos con soporte de aceleración para operaciones int4/int8/int16/FP16/BF16/TF32
Marcos de aprendizaje profundo soportados	TensorFlow, Caffe, Tflite, PyTorch, Onnx, etc.	TensorFlow, Caffe, Tflite, PyTorch, Onnx, etc.
Multimedia	Decodificador de video 8K@60fps y codificación de video 8K@30fps	Decodificador de video 8K@60fps y codificación de video 8K@30fps
USB	1xUSB3.0 compartido con PCIE2.0/SATA3.0, 2xUSB2.0, 2xUSB3.1 + 2xUSB2.0 por OTG (USB-C)	1xUSB3.0 compartido con PCIE2.0/SATA3.0, 2xUSB2.0, 1xUSB3.1 + 1xUSB2.0 por OTG (USB-C)
Pantalla	USB-C/DP1.4 compartido 2x4 carriles de DP1.4, 2xHDMI2.1/eDP1.4	USB-C/DP1.4 compartido 1x4 carriles de DP1.4, 1xHDMI2.1/eDP1.4
PCIE 3.0	2x2 carriles de PCIe 3.0	N/A
PCIe2.0/SATA3.0	3x1 carriles de PCIe2.0, 3xSATA3.0	2x1 carriles de PCIe2.0, 2xSATA3.0

Nota: Extraído de (Radxa, 2024; Rockchip, 2024).

Con base en las tablas 4.1 y 4.2, se encontró que el RK3588/RK3588S sobresale como la mejor alternativa para el caso de estudio, ya que posee un conjunto de características superior a los demás microprocesadores desarrollados por Rockchip y ofrece un sólido rendimiento general. Por lo tanto, se escogió un SBC basado en uno de estos dos microprocesadores.

Tabla 4.2
Comparación RK3566/RK3568 y RK3576.

	RK3566/RK3568	**RK3576**
CPU	4xCortex-A55	4xCortex-A72 + 4xCortex-A53
GPU	ARM G52-2EE	ARM Mali-G52 MC3
NPU	1 TOPs, con soporte de aceleración para operaciones int8/int16/FP16/BF16	6 TOPs, de 2 núcleos con soporte de aceleración para operaciones int4/int8/int16/FP16/BF16/TF32
Marcos de aprendizaje profundo soportados	TensorFlow, Caffe, Tflite, PyTorch, Onnx, etc.	TensorFlow, Caffe, Tflite, PyTorch, Onnx, etc.

Nota: Extraído de (Rockchip, 2024).

Selección de la placa SBC basada en RK3588/RK3588S

Existe una gran variedad de sistemas embebidos que han incorporado en su diseño, ya sea el RK3588 o su versión recortada, el RK3588S. Estos abarcan desde TV boxes, SBCs con factor de forma similar al Nvidia Jetson Nano o Raspberry Pi 4, enrutadores, laptops, entre otros. En la tabla 4.3 se listan 10 sistemas embebidos con un factor de forma similar al de la Raspberry Pi 4.

Tabla 4.3
Sistemas embebidos que llevan el microprocesador RK3588/RK3588S, la cantidad de memoria y su precio individual.

Modelo	**Memoria**	**CPU**	**Precio Referencial abril 2024 (8GB)**
ArmSoM Sige7	8GB/16GB/32GB LPDDR4X	RK3588	S/. 607.00
Indiedroid Nova	4GB/8GB/16GB LPDDR4X	RK3588S	S/. 662.00
Khadas Edge 2	8GB/16GB LPDDR4X	RK3588S	S/. 732.00
NanoPC T-6	4GB/8GB LPDDR4X	RK3588	S/. 511.00
NanoPi R6C	4GB/8GB LPDDR4X	RK3588S	S/. 423.00
Orange Pi 5	4GB/8GB/16GB/32GB LPDDR4X	RK3588S	S/. 324.00
Orange Pi 5 Plus	4GB/8GB/16GB LPDDR4X	RK3588	S/. 401.00
ROC-RK3588S-PC	4GB/8GB/16GB LPDDR4x	RK3588S	S/. 842.00
Rock 5A	4GB/8GB/16GB LPDDR4X	RK3588S	S/. 342.00
Rock 5B	4GB/8GB/16GB LPDDR4X	RK3588	S/. 564.00

Entre los SBCs enumerados en la tabla 4.3, el Orange Pi 5 ofrece una excelente relación calidad-precio, lo que convierte a este SBC en la opción más atractiva y adecuada para satisfacer las necesidades de este proyecto.

Orange Pi 5

El Orange Pi 5 (figura 2.21) es una potente y versátil placa de desarrollo SBC, fabricada por Shenzhen Xunlong Software Co., que ofrece un equilibrio entre calidad y precio. Cuenta con un amplio rango de interfaces, como HDMI, USB tipo C, interfaz de conexión M.2 a través de PCIe 2.0, puerto Gigabit Ethernet y 26 pines de expansión. Sin embargo, su principal característica es el microprocesador RK3588S, que puede ser empleado para aplicaciones de inteligencia artificial, visión artificial, computación en la nube, seguridad inteligente, casas inteligentes y otros campos (OrangePi, 2022).

En la tabla 4.4, se observan las características principales con las que cuenta el Orange Pi 5. Para ver de manera más detallada estas características ver el anexo B.

4.1.3. Altavoces o parlantes

Se utilizan parlantes de un tamaño reducido, con un consumo máximo de 3 W y que admite audio a través de un conector mini jack de 3.5 mm y se alimentan mediante USB 2.0 (ver figura 4.2).

Figura 4.2
Parlantes.

Tabla 4.4
Características principales Orange Pi 5.

Principales características Orange Pi 5	
Microprocesador	RK3588S
Salida de video	HDMI 2.1, hasta 8k@60fps, DP 1.4, 2xMIPI D-PHY TX 4 carriles
Cámara	1xMIPI CSI 4 carriles, 2xMIPI D-PHY RX 4 carriles
Almacenamiento soportados	16 MB QSPI NOR Flash, ranura MicroSD, ranura SSD M.2 PCIe 2.0
Ethernet	10/100/1000 Mbps
Audio	3.5 mm de entrada jack, micrófono incorporado, salida de audio a través de HDMI
Ranura M2	Soporte para PCIe WIFI 6 + BT 5.0 + BLE, soporta SSD (SATA o NVME 2.0)
Interfaz USB	1xUSB 3.0, 2xUSB 2.0 (uno es compartido con el USB tipo C), 1xUSB tipo C
26 pines de expansión	Para UART, PWM, I2C, SPI, CAN y GPIO
Puerto serial de depuración	3 pines de puerto serial de depuración
LED	Para la luz de encendido y estatus.
Botones	1 como switch, 1xMask ROM Key, 1xRecovery
Alimentación	5V/4A como máximo a través de USB tipo C
Tamaño del producto	100 mm x 62 mm
Peso	46 g

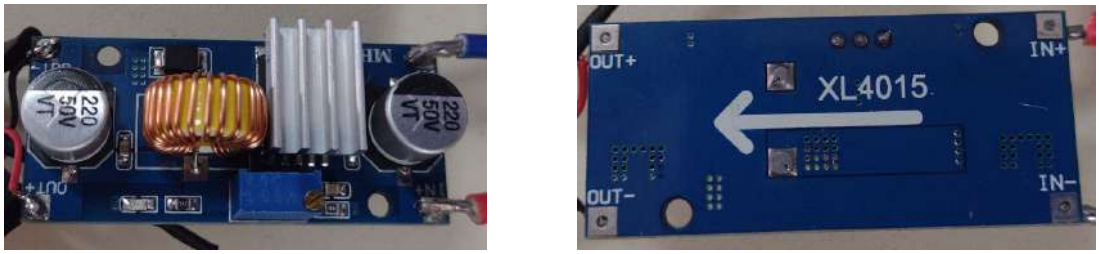
Nota: Extraído de (OrangePi, 2022).

4.1.4. Fuente de alimentación

Dadas las características eléctricas del Orange Pi 5, que requiere una alimentación de 5 V a un máximo de 4 A (ver tabla 4.4), es decir, una potencia máxima de 20 W, se necesita una fuente de alimentación capaz de entregar dicha potencia utilizando la batería del vehículo, que normalmente proporciona 12 V directos.

Para convertir la tensión de 12 V al voltaje de 5 V requerido por el Orange Pi 5, se utiliza un convertidor DC-DC buck (step-down). El convertidor que cumple con la entrega de un voltaje de 5 V y una corriente de 4 A es el XL4015 (ver figura 4.3).

Figura 4.3
Convertidor DC-DC Step-Down XL4015. Vista delantera a la izquierda y vista posterior a la derecha.



El XL4015 trabaja con una frecuencia de PWM de 180 kHz y es capaz de entregar una corriente de salida de hasta 5 A con alta eficiencia. Además, cuenta con una excelente regulación de línea y carga, y un bajo voltaje de rizado.

Se mencionan las características más importantes a continuación.

- Eficiencia: El XL4015 según su hoja de datos tiene una eficiencia mayor al 85 %, con una entrada de 12 V y una salida de 5 V/4 A, lo que permite que no se desperdicie demasiada energía y se genere menos calor en el propio módulo.
- Ajustable: Este convertidor cuenta con un potenciómetro de precisión que permite ajustar el voltaje de salida a 5 V.
- Protección: El XL4015 cuenta con una protección de temperaturas altas, además cuenta con una protección de corto circuito a la salida.

Para ver de manera más detallada las características del XL4015 ir al anexo F.

4.2. CNNs para la estimación de la postura de la cabeza

El primer paso fue seleccionar las redes neuronales convolucionales (CNNs). Esto se realizó basándose en el número de parámetros, ya que este debe ser pequeño para que pueda ejecutarse en el Orange Pi 5. El límite establecido para la selección de modelos fue de un máximo de 10 millones de parámetros. Los modelos que cumplen con esta especificación se

encuentran en la tabla 4.5, y además han sido preentrenados con la base de datos de IMAGENET versión 1.

Tabla 4.5
Modelos con números de parámetros inferior a 10 millones.

Modelo	# de parámetros iniciales	# de parámetros tras modificar
DenseNet121	8 M	6.96 M
EfficientNet_B0	5.3 M	4.02 M
EfficientNet_B1	7.8 M	6.52 M
EfficientNet_B2	9.1 M	7.71 M
GoogleNet	6.6 M	5.61 M
MNASNet0_5	2.2 M	0.95 M
MNASNet0_75	3.2 M	1.9 M
MNASNet1_0	4.4 M	3.11 M
MNASNet1_3	6.3 M	5.01 M
MobileNet_V2	3.5 M	2.23 M
MobileNet_V3_Large	5.5 M	4.21 M
MobileNet_V3_Small	2.5 M	1.53 M

Nota: Extraído de (Torchvision, 2023).

Tabla 4.6
División de los modelos en grupos con base en su número de parámetros.

Grupo	Modelo	# de parámetros tras modificar
1	MNASNet0_5	0.95 M
	MobileNet_V3_Small	1.53 M
	MNASNet0_75	1.9 M
	MobileNet_V2	2.23 M
2	MNASNet1_0	3.11 M
	EfficientNet_B0	4.02 M
	MobileNet_V3_Large	4.21 M
	MNASNet1_3	5.01 M
3	GoogleNet	5.61 M
	EfficientNet_B1	6.52 M
	DenseNet121	6.96 M
	EfficientNet_B2	7.71 M

Los modelos presentados en la tabla 4.5 fueron modificados en su última capa, siguiendo el método descrito en la fase 3 del capítulo de diseño (ver sección 3.2.3), reduciendo así su número de parámetros. Además, los modelos de la tabla 4.5 fueron clasificados en 3 grupos de acuerdo al número de parámetros tras la modificación (ver tabla 4.6): el primer grupo corresponde a los modelos ligeros, con un rango de 0 a 2.5 millones de parámetros; el segundo grupo incluye los

modelos intermedios, con un rango de 2.5 millones a 5.5 millones de parámetros; y el último grupo, denominado pesado, abarca modelos con un rango de 5.5 millones a 10 millones de parámetros.

4.2.1. Conjuntos de datos para entrenamiento y evaluación de las CNNs

Para entrenar los modelos se hace uso de los conjuntos de datos más utilizados para esta tarea: 300W_LP para el entrenamiento, y tanto BIWI como AFLW2000 para la evaluación de las CNNs, los cuales fueron descritos en la sección 2.2.8. Estos conjuntos de datos brindan la información de la postura de la cabeza en forma de anotaciones contenidas en archivos de metadatos.

Los metadatos proporcionados tanto para AFLW2000 como para 300W_LP incluyen puntos de referencia de la cara, así como los ángulos de rotación en radianes, y están en archivos con extensión .mat. Para usar dichos metadatos, se crea un archivo que lista los nombres de todas las imágenes de los conjuntos de datos 300W_LP y AFLW2000 llamado `filename.txt`, con la finalidad de tener las correspondencias en un solo archivo de texto para cada uno.

Por otro lado, el conjunto de datos BIWI, al contener imágenes completas, requiere que dichas imágenes sean preprocesadas utilizando un algoritmo provisto por (Yang, 2019), obteniendo únicamente los rostros. Al igual que en los otros conjuntos de datos, los ángulos de rotación de cada imagen están anotados en radianes. Este algoritmo genera como salida un archivo único llamado `BIWI_noTrack.npz`, que contiene toda la información de las imágenes procesadas, así como también los ángulos de rotación correspondientes.

4.2.2. Entrenamiento y evaluación de las CNNs

Hardware y configuraciones de entorno de entrenamiento y evaluación

En la tabla 4.7 se muestra información de la CPU y GPU utilizadas, así como también la versión de Python y el marco de desarrollo de inteligencia artificial empleados durante el entrenamiento de los modelos.

Tabla 4.7
Configuración de hardware y el entorno del software.

Hardware	RAM/VRAM	CPU	GPU
	32GB/5GB	Intel Xeon W-2123	NVIDIA Quadro P2000
Software	Python	CUDA	PyTorch
	3.10	12	2.1

Carga del conjunto de datos de entrenamiento

Para realizar el entrenamiento, se carga el conjunto de datos 300W_LP, el cual enlaza las imágenes con sus respectivos valores de ángulos de rotación, extraídos de los archivos .mat. Para ello, se utiliza el archivo de texto `filename.txt` que contiene los nombres de cada imagen y sus correspondientes metadatos generados anteriormente. Dado que las imágenes en el conjunto de datos varían en tamaño y cada píxel está representado con 8 bits, es necesario preprocesarlas. Este preprocesamiento comienza redimensionando todas las imágenes a un tamaño fijo de 224x224 píxeles.

Además, los valores de los píxeles se normalizan utilizando las medias [0,485, 0,456, 0,406] y las desviaciones estándar [0,229, 0,224, 0,225] para cada uno de los canales de color (rojo, verde y azul) recomendados para cada modelo por Pytorch. Esta normalización ajusta los valores de los píxeles en torno a cero, con valores cercanos dependiendo de cuánto se aleje cada píxel de la media, lo que ayuda a que los modelos CNN converjan más rápidamente y de manera más estable.

Por otro lado, los ángulos de rotación son convertidos en matrices de rotación de verdad fundamental R_{gt} (ver matriz 3.10).

Adicionalmente, tanto la imagen normalizada como la matriz de rotación R_{gt} son convertidas a tensores, ya que el marco de desarrollo PyTorch trabaja con este tipo de datos.

Proceso de entrenamiento

El siguiente paso es pasar los datos preprocesados a un generador provisto por PyTorch, el cual se encarga de manejar algunos hiperparámetros, como el tamaño de lote, la aleatoriedad de los datos, así como asignar el número de hilos del procesador que serán utilizados para entregar los lotes a la GPU, acelerando el proceso de generación de lotes del conjunto de datos. Los hiperparámetros utilizados en el entrenamiento son: el número de épocas, tamaño de lote, tasa de aprendizaje con una reducción a la mitad cada 10 épocas, optimizador de tipo Adam y el número de hilos de CPU para el generador. Estos valores de hiperparámetros están especificados en la sección de pruebas y resultados 5.1.

Al iniciar el proceso de entrenamiento, los lotes entregados por el generador se pasan a través del modelo CNN. Los valores predichos por el modelo se comparan con los valores reales (la verdad fundamental) utilizando la pérdida geodésica (ver expresión 2.7), la cual se usa para retroalimentar el entrenamiento mediante ”backpropagation“ (propagación hacia atrás) con la finalidad de optimizar los parámetros del modelo CNN. Es decir, se compara la matriz R_p con la matriz R_{gt} (matrices 3.9 y 3.10 respectivamente).

Proceso de evaluación

Durante cada época, una vez terminado el entrenamiento de un modelo, este es evaluado inmediatamente. Para ello, previamente, al igual que con 300W_LP, se realiza un preprocesamiento para los conjuntos de datos AFLW2000 y BIWI, utilizando un archivo de texto `filename.txt` que contiene los nombres de las imágenes y los metadatos .mat, así como

`BIWI_noTrack.npz` para cada conjunto, respectivamente. La diferencia con respecto al conjunto de datos de entrenamiento es que, para estos dos últimos, no se utiliza la matriz R_{gt}, sino que los ángulos de rotación son convertidos a grados sexagesimales mediante la expresión 3.11.

Para evaluar el modelo CNN en cada época, no se utiliza la pérdida geodésica, sino el error absoluto promedio (MAE, ver expresión 2.8).

Guardado de los modelos

El guardado de los modelos se realiza en función de los resultados obtenidos tanto en la evaluación con el conjunto de datos BIWI como en AFLW2000. Para ello, se calcula un promedio entre los valores de MAE obtenidos en cada evaluación, y los 5 mejores modelos se guardan automáticamente juntamente con los resultados obtenidos para cada evaluación, reemplazando en cada época aquellos que tienen un peor promedio.

En la figura 4.4, se describe de manera gráfica el flujo de entrenamiento de los modelos CNN.

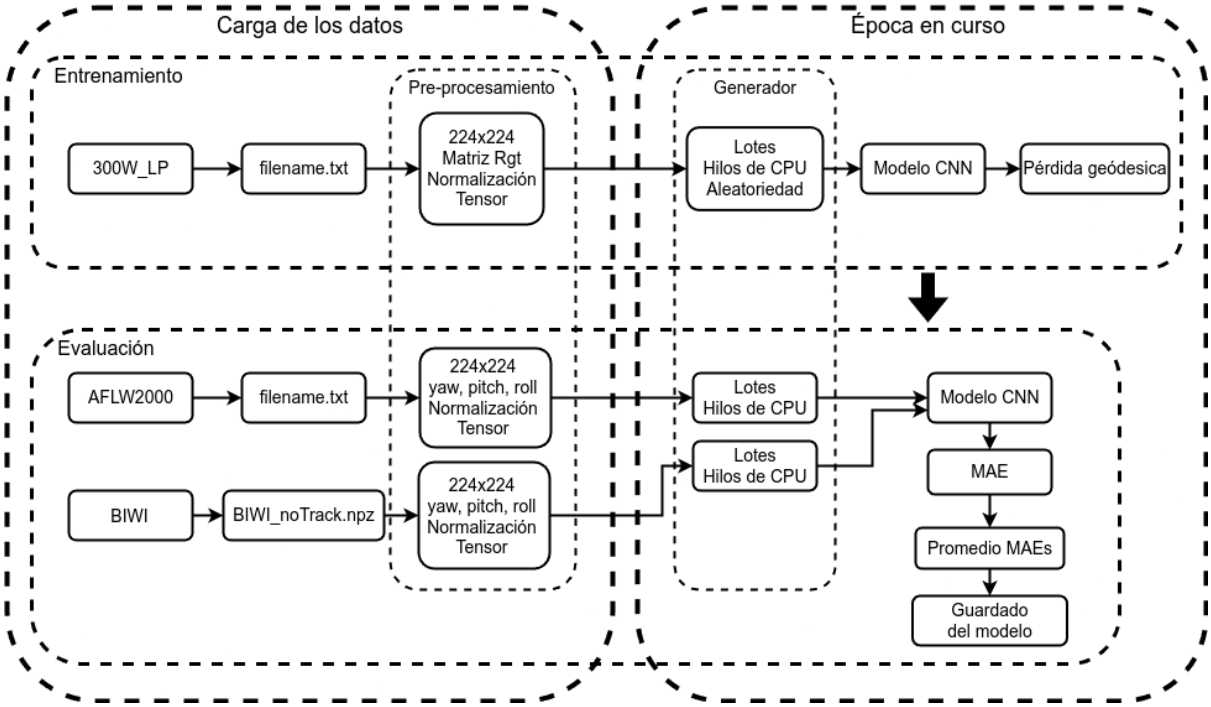
Cabe resaltar que el proceso de entrenamiento y evaluación se llevó a cabo haciendo uso de la GPU y para cada uno de los modelos CNNs de la tabla 4.5.

4.2.3. Conversión de los modelos CNN para la NPU del RK3588S

Para implementar los modelos CNN entrenados en el Orange Pi 5 y aprovechar la NPU del microprocesador RK3588S, es necesario convertir dichos modelos a un formato binario compatible. Esta conversión permite que la NPU ejecute los modelos de manera eficiente, optimizando el rendimiento en tareas de inferencia.

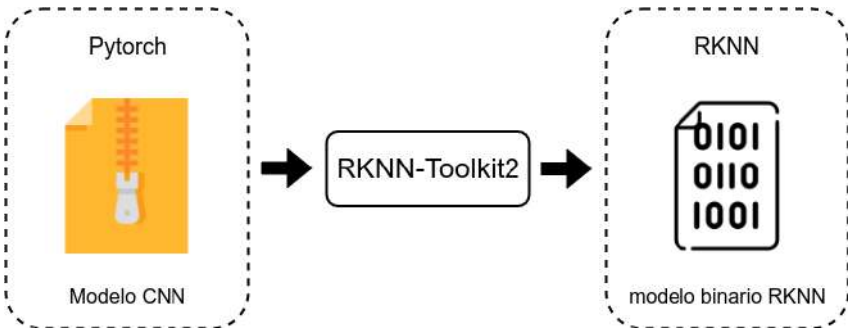
La conversión de los modelos se realizó utilizando la biblioteca RKNN-Toolkit2 (para ver el proceso de instalación, consulte el anexo G), diseñada específicamente para dar soporte a la NPU en plataformas basadas en Rockchip, como el Orange Pi 5. Este toolkit permite la

Figura 4.4
Proceso de entrenamiento y validación del modelo CNN



conversión de los modelos de PyTorch a un formato binario RKNN (ver figura 4.5).

Figura 4.5
Conversión del modelo CNN de PyTorch a RKNN.



El proceso de conversión sigue los siguientes pasos:

1. **Preparación del entorno**: La instalación de las herramientas necesarias para la conversión se realizó en un entorno de Python gestionado por Miniconda, garantizando la correcta configuración de dependencias.

2. **Preparación del modelo PyTorch**: Se optimiza el modelo entrenado para que pueda ejecutarse en diversos dispositivos, y luego se crea un archivo con el nombre del modelo

con la extensión ".pt".

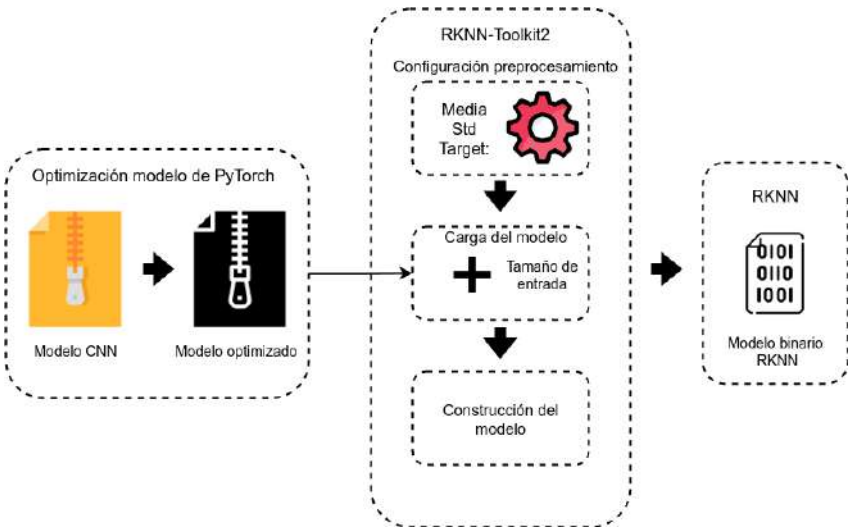
3. **Configuraciones de preprocesamiento**: Se deben definir las mismas configuraciones utilizadas en el entrenamiento de los modelos con PyTorch, las cuales son las siguientes:

   - $Media = [0{,}485 \times 255, 0{,}456 \times 255, 0{,}406 \times 255]$, se realiza una multiplicación por 255 ya que PyTorch, antes de normalizar en torno a cero, escala los valores a un intervalo de 0 a 1.
   - $DesviacionEstandar = [0{,}229 \times 255, 0{,}224 \times 255, 0{,}225 \times 255]$.
   - Plataforma de destino: $rk3588$.

4. **Carga del modelo optimizado**: Se carga el modelo optimizado y se define un tamaño de entrada, considerando el tipo de entrada que admite PyTorch, en este caso $[1, 3, 224, 224]$, representando el tamaño del lote, el número de canales, la altura y el ancho de la imagen, respectivamente.

5. **Construcción del modelo**: El último paso es construir el modelo, lo que finaliza con la creación del binario RKNN.

Figura 4.6
Proceso de creación del binario RKNN.



El proceso completo se ilustra gráficamente en la figura 4.6, que detalla los pasos para convertir los modelos de PyTorch a RKNN. La figura 4.7 muestra el flujo de creación del archivo

Figura 4.7
Salida del proceso de creación del binario RKNN.

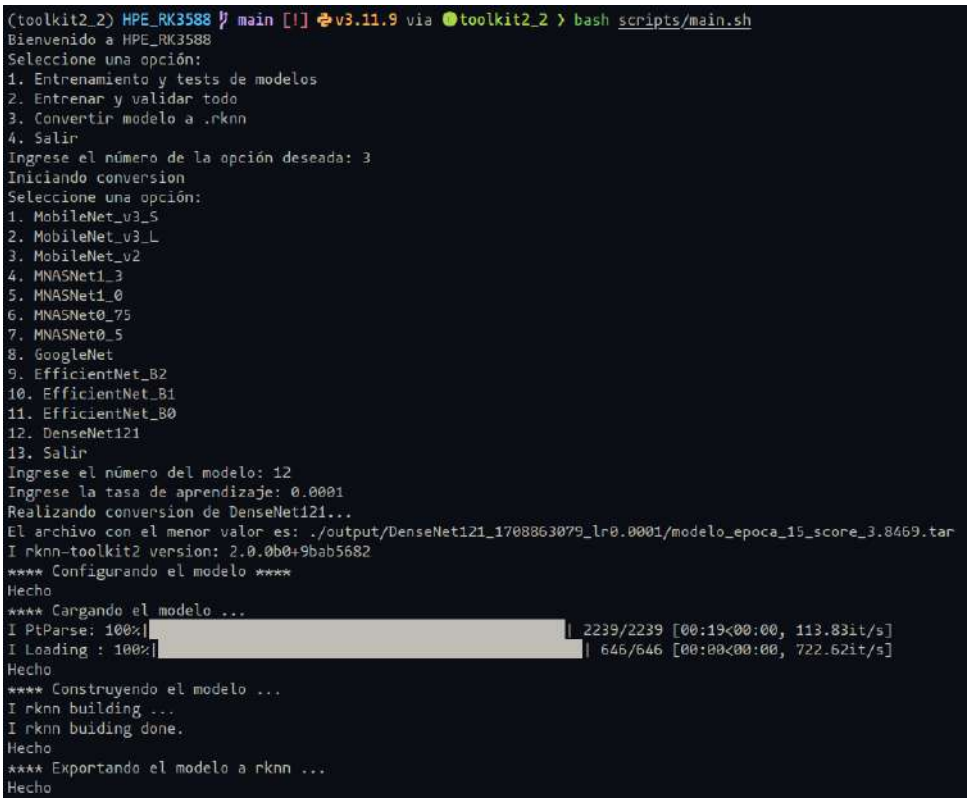


Figura 4.8
Archivos binarios RKNN de los modelos CNN.

binario, ejecutado mediante el script `main.sh` (para más detalles sobre `main.sh`, consulte el anexo H). Por último, la figura 4.8 presenta los archivos generados, incluyendo tanto los modelos `.pt` optimizados como los binarios resultantes listos para su uso en la NPU.

4.3. Proceso de detección de distracción

4.3.1. Conjunto de datos de distracción

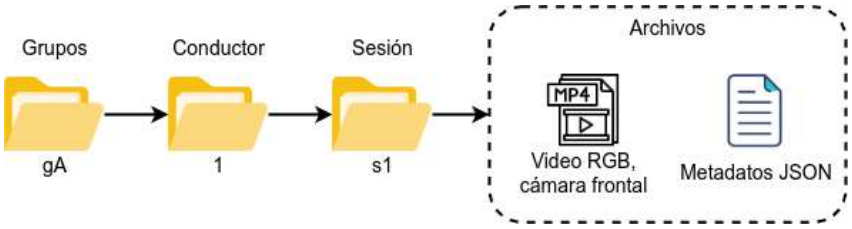
Para el desarrollo de esta sección, se opta por utilizar el conjunto de datos *Driver Monitoring Dataset* (DMD), ya que, a diferencia de SFDDD, este cuenta con una mayor cantidad de datos y fue etiquetado utilizando también una cámara que graba la cara. El conjunto de datos DMD está descrito en el anexo E.

De manera resumida, el conjunto de datos DMD separa sus datos en grupos con las siguientes denominaciones: gA, gB, gC, gE, gF y gZ. Estos contienen la información de 5 conductores cada uno, a excepción de gZ, que contiene la información de solo 4 conductores. Dentro de cada grupo, y a su vez dentro de la carpeta de cada conductor, existen sesiones enumeradas de la siguiente forma: $s1$, $s2$, $s3$ y $s4$. La sesión $s1$ fue tomada en un escenario real, $s2$ y $s3$ en un escenario real con el vehículo detenido, y $s4$ en un entorno de simulación de vehículo. Estas sesiones contienen videos en RGB, IR y de profundidad de 3 cámaras (una para la cara, otra para el cuerpo y otra para las manos) en formato MP4, además de un archivo de metadatos JSON para cada sesión.

Dado que los modelos de CNN fueron entrenados con el conjunto de datos 300W_LP, que cuenta con datos RGB, para este análisis se consideró únicamente la información disponible en RGB del conjunto de datos DMD. Además, se utilizó exclusivamente la información de la cámara que captura los movimientos de la cara y cabeza (cámara frontal) y las sesiones $s1$, $s2$ y $s4$, excluyendo a $s3$, ya que proporciona información sobre la búsqueda de objetos con la mano, la cual no aporta datos relevantes para este análisis. Un ejemplo del contenido para gA, el conductor número 1 y la sesión 1 se muestra en la figura 4.9, que es el contenido que tendrán todos los grupos.

Cada video de cada sesión perteneciente a un determinado conductor es procesado a través de un modelo CNN previamente entrenado. Para esta tarea se utilizó el modelo DenseNet121

Figura 4.9
Ejemplo del contenido utilizado del conjunto de datos DMD. Para gA, conductor 1 y sesión 1.



entrenado (ver tablas 5.1 y 5.3), con el cual se predicen los ángulos de rotación. Estos se registran en una tabla que contiene la siguiente información: ”frame” (cuadro), ”gaze_on_road”, ”driver_actions”, yaw, pitch y roll. Las etiquetas ”gaze_on_road” y ”driver_actions” corresponden a los niveles utilizados por el conjunto de datos DMD, se encuentran en los metadatos y proporcionan información para cada cuadro de video. Esta información se guarda en un archivo con la siguiente nomenclatura: ”nombregrupo_conductor_sesion.csv”; por ejemplo, si pertenece al grupo gA, el conductor número 1 y la sesión 1, el archivo será `gA_1_s1.csv`.

Las etiquetas contenidas en el nivel ”gaze_on_road” son ”looking_road” y ”not_looking_road”, las cuales indican si el conductor está prestando atención a la tarea de conducción a través de la mirada, basándose principalmente en la información proporcionada por la cámara frontal. Por otro lado, el nivel ”driver_actions” incluye las siguientes etiquetas: ”safe_drive”, ”reach_side”, ”radio”, ”drinking”, ”talking_to_passenger”, ”hair_and_makeup”, ”texting_left”, ”texting_right”, ”phonecall_left”, ”phonecall_right” y ”unclassified”, basándose principalmente en la cámara que apunta al cuerpo.

Al realizar el proceso de creación de tablas y archivos para cada grupo y sesión, se obtuvieron un total de 105 archivos CSV, con los cuales se procede a realizar el análisis de distracción. Para ello, primero se descarta el uso de las etiquetas de ”driver_actions” que no contribuyen ni aportan información relevante para determinar la distracción mediante la estimación de la postura de la cabeza, ya que están enfocadas en las manos, no implican movimientos significativos de la cabeza o presentan obstrucciones que impiden estimar correctamente la postura de la cabeza, o bien no están clasificadas. Las etiquetas que no se utilizan son: ”drinking”, ”phonecall_left”, ”phonecall_right” y ”unclassified”.

El siguiente paso consiste en cargar las tablas de datos, para ello se hace uso de la biblioteca Pandas de Python, que facilita la manipulación de datos en formato tabular y archivos del tipo CSV. Con esta biblioteca, las tablas se concatenan para unificar la información de los videos en un solo marco de datos (DataFrame). Luego, se filtra el DataFrame para eliminar cualquier registro que no contenga información válida, es decir, aquellos datos vacíos del tipo ”NaN”

Posteriormente, se procede a dividir el DataFrame en segmentos de información, los cuales representan los distintos intervalos de los videos que fueron etiquetados. Cada segmento recibe un identificador numérico único que lo distingue. Esta división se realiza bajo la siguiente lógica.

- Se comparan las etiquetas de la columna ”gaze_on_road” con su valor en la fila anterior. Si hay un cambio (por ejemplo de ”looking_road” a ”not_looking_road”), se marca el inicio de un nuevo segmento.
- De manera similar, si las etiquetas de la columna ”driver_actions” cambian, también se considera que ha comenzado un nuevo segmento.

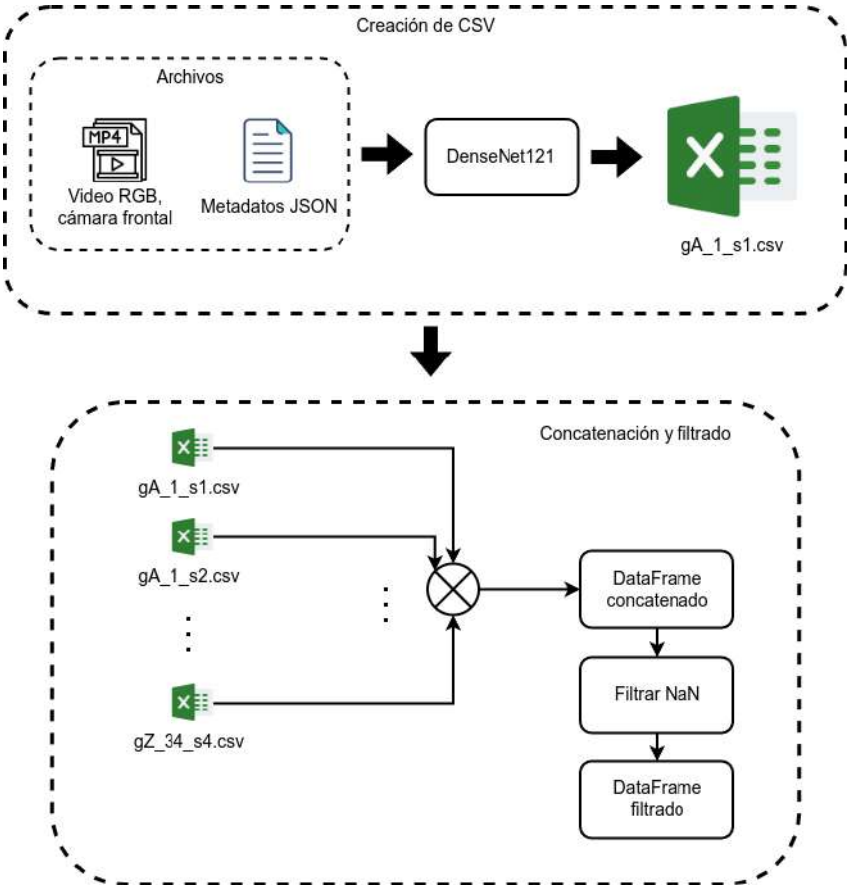
La figura 4.10 ilustra gráficamente el proceso descrito para la creación de los archivos CSV, la carga y concatenación de dichos archivos, y la eliminación de los datos vacíos (NaN).

4.3.2. Detectores de distracción

Umbralización

Siguiendo los pasos descritos para este método en la sección 2.2.3, es necesario contar con secuencias continuas de datos que representen conducción segura. Por ello, se filtran los segmentos del DataFrame resultante del análisis anterior, aplicando la siguiente condición: si la columna ”gaze_on_road” tiene el valor ”looking_road” y, simultáneamente, la columna ”driver_actions” tiene el valor ”safe_drive”, se seleccionan los datos de las secuencias continuas

Figura 4.10
Proceso de creación de archivos CSV y filtrado de datos vacíos NaN.



de cuadros. Como resultado, se identificaron 327 segmentos consecutivos que representan conducción segura.

De estos segmentos continuos de conducción segura, se extraen los valores de las columnas correspondientes a yaw, pitch y roll. Antes de proceder con el análisis, se eliminan los posibles datos atípicos presentes en cada uno de los segmentos. Para ello, se emplea el método estadístico del rango intercuartílico (IQR). Este método calcula la diferencia entre el primer cuartil (Q_1), que abarca el 25 % inferior de los datos, y el tercer cuartil (Q_3), que cubre el 25 % superior. A partir de este valor, se establecen límites inferior y superior utilizando 1,5 veces el IQR, tanto por debajo de Q_1 como por encima de Q_3, respectivamente (ver expresiones 4.1 y 4.2). Dicho proceso se realiza para los valores de yaw, pitch y roll de manera simultánea.

$$IQR = Q_3 - Q1 \tag{4.1}$$

$$limite_{sup} = Q3 + 1{,}5(IQR) \quad , \quad limite_{inf} = Q1 - 1{,}5(IQR) \tag{4.2}$$

Una vez eliminados los valores atípicos, se calcula el valor promedio de los ángulos yaw, pitch y roll en cada uno de los 327 segmentos, obteniendo un conjunto de puntos base seguros $B = \{b_1, b_2, \ldots, b_{327}\}$. Seguidamente, se obtiene el valor promedio de dicho conjunto B, que representa el punto base de conducción segura β.

Posteriormente, con base en β, se utiliza la expresión 2.1, donde se calcula la distancia espacial para los ángulos yaw, pitch y roll correspondientes, los cuales se comparan con β. Esto genera un conjunto de valores de distancia espacial d_i (para $i = 1, 2, 3, \ldots, 327$), con el cual se establece el umbral de seguridad D, que es el valor de distancia espacial que representa al menos el 90 % de los datos de todos los segmentos. En la tabla 4.8 se muestran los valores de β y del umbral D (columna "centro").

Tabla 4.8
Betas y umbrales.

	Derecha	Izquierda	Centro
Beta	(13,97°; 6,02°; −4,26°)	(−13,88°; 8,54°; −2,19°)	(0,52°; 7,06°; −3,43°)
Umbral	11.32	10.61	18.92

Umbralización modificada

Como se menciona en la sección 3.2.4, es necesario dividir el conjunto de datos de conducción segura, lo cual se logra calculando el umbral promedio utilizando el método de umbralización original. Para ello, se emplea la expresión 3.3, cuyo resultado es $D_{prom} = 10{,}97$. Este valor permite dividir el conjunto de conducción segura en 3 categorías: izquierda, centro y derecha, siguiendo los pasos descritos a continuación.

1. Si un valor de distancia espacial $d > D_{prom}$ y su respectivo ángulo $yaw < 0$, entonces se etiqueta como "izquierda".

2. Si un valor de distancia espacial $d > D_{prom}$ y su respectivo ángulo $yaw \geq 0$, entonces se

etiqueta como “derecha”.

3. Si no se cumple ninguna de las condiciones anteriores, se etiqueta como “centro”.

Para asignar estas etiquetas, se crea una nueva columna llamada “safe_drive_direction”. Se utiliza el ángulo yaw para determinar la dirección, ya que este es el ángulo que indica dicha orientación. La etiqueta se asigna cuadro por cuadro, dado que un segmento de conducción segura puede ser largo y contener subsegmentos más pequeños en los que se realizan movimientos tanto a la izquierda como a la derecha. Como resultado, se obtienen 218, 177 y 289 segmentos etiquetados como “derecha”, “izquierda” y “centro”, respectivamente, a los que se eliminan posibles datos atípicos a través del IQR.

Posteriormente, se hallan los umbrales izquierdo y derecho, tal como se explica en la sección 3.2.4. Haciendo uso de las expresiones contenidas en 3.2, se calculan los umbrales D_{der} y D_{izq}. Los valores de β para “derecha”, “izquierda” y “centro” se encuentran en la tabla 4.8.

Lógica difusa

Para este método, se hace uso de los datos de conducción segura divididos (izquierda, derecha y centro) obtenidos en la sección anterior. Además, se utilizan las etiquetas de la columna “driver actions”. El filtrado de cada grupo sigue los siguientes pasos:

1. Para datos de conducción segura: Se seleccionan los datos de la columna “safe drive direction”, filtrando aquellos segmentos donde los valores específicos sean: “derecha”, “izquierda” o “centro”.

2. Para datos de “driver actions”: Se seleccionan los datos de la columna “gaze on road” con el valor “not looking road” y los datos de la columna “driver actions”, filtrando aquellos segmentos que indican distracción: “reach side” (buscar al costado), “talking to passenger” (hablar con el pasajero), “hair and makeup” (arreglarse el cabello), “texting left” (texteo izquierda), “texting right” (texteo derecha), o “radio”.

Se utiliza únicamente el valor “not_looking_road” en la columna “gaze_on_road”, ya que, si un conductor está mirando hacia la carretera mientras realiza alguna acción, no se obtendrán movimientos significativos de la cabeza. Además, al estar observando la carretera, el sistema podría considerar dicha situación como no distracción, dado que solo se hace uso de una cámara frontal.

Los segmentos obtenidos para cada una de las acciones se muestran en la tabla 4.9. De igual manera, se aplica la eliminación de posibles datos atípicos en los segmentos mediante el uso del método IQR.

Tabla 4.9
Número de segmentos para acciones de distracción.

Buscar al costado	Hablar con el pasajero	Arreglarse el cabello	Texteo izquierda	Texteo derecha	radio
191	64	63	115	127	52

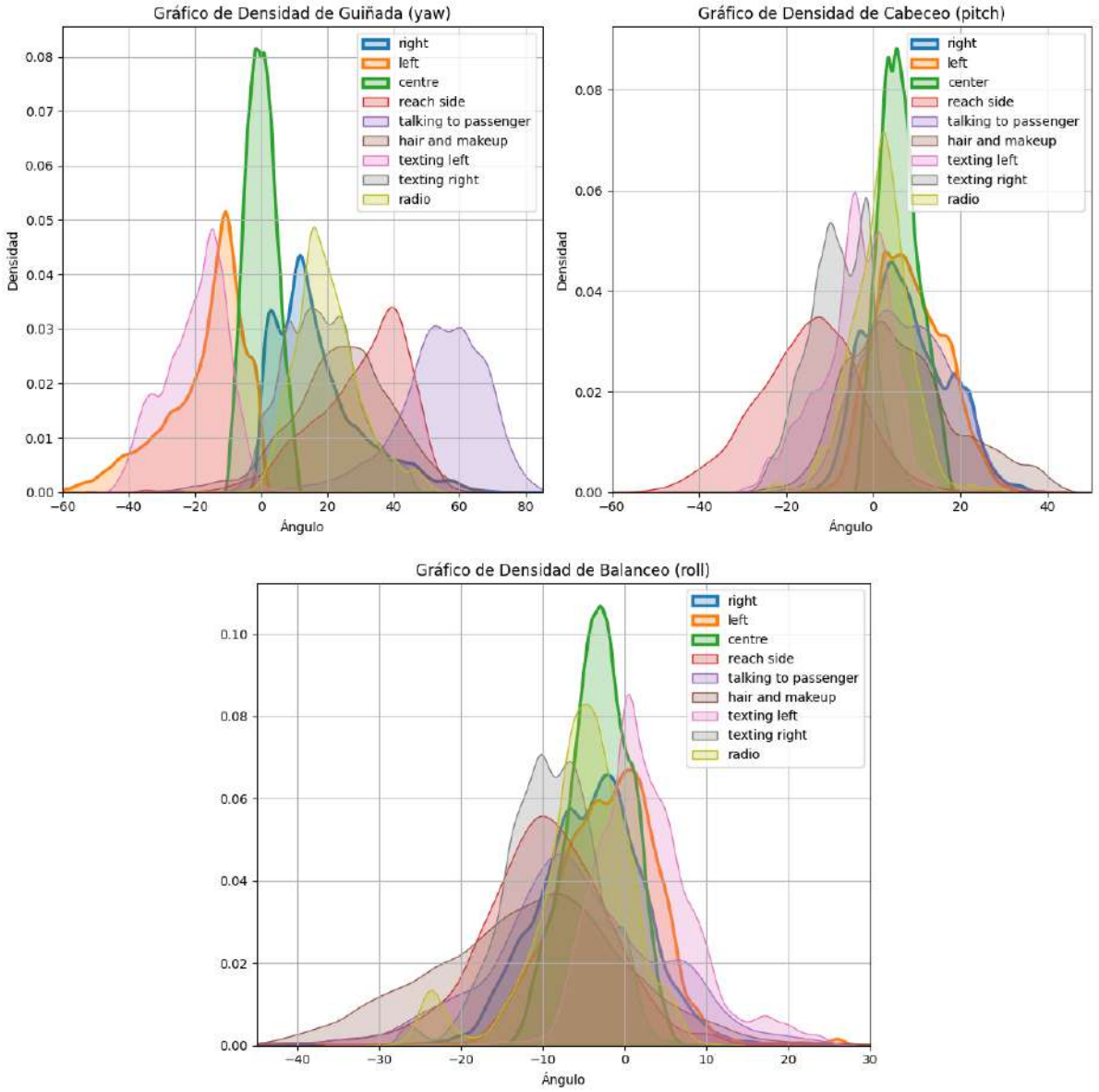
Con base en los segmentos de acciones del conductor y de conducción segura, se realizan gráficos de densidad para definir de manera visual los límites aproximados de los universos de discurso, así como también sus funciones de pertenencia. La figura 4.11 muestra las gráficas de densidad para cada uno de los ángulos de rotación (yaw, pitch, roll).

Los dominios aproximados para los universos de discurso “yaw”, “pitch” y “roll” se establecen a partir de la figura 4.11, siendo estos universos los antecedentes. De la misma manera, se establecen funciones de pertenencia con dominios aproximados, como se indica en la sección 3.2.4. Además, se define el dominio de discurso del “estado” y sus funciones de pertenencia. A continuación, se presentan los dominios de los universos y sus respectivas funciones de pertenencia:

1. Yaw (giro): El universo de discurso considerado para “yaw”, es el intervalo $[-65, 85]$, lo que implica que los valores relevantes están comprendidos dentro de este dominio. Las funciones de pertenencia se definen a continuación.

    - Función de pertenencia 1: “intervalo 1” del tipo triangular, tiene sus límites en $[-45, -1]$ y pertenencia máxima en -16.

Figura 4.11
Gráficos de densidad para yaw, pitch y roll, haciendo uso de los segmentos de acciones de conducción y etiquetas de conducción segura generadas a partir del conjunto de datos DMD.



- Función de pertenencia 2: “intervalo 2” del tipo triangular, tiene sus límites en $[-65, 0]$ y pertenencia máxima en -10.
- Función de pertenencia 3: “intervalo 3” del tipo gaussiano, con un valor de punto medio de 1 y una desviación estándar de 5.
- Función de pertenencia 4: “intervalo 4” del tipo triangular, tiene sus límites en $[-2, 50]$ y pertenencia máxima en 20.
- Función de pertenencia 5: “intervalo 5” del tipo triangular, tiene sus límites en

$[-10, 60]$ y pertenencia máxima en 40.

- Función de pertenencia 6: "intervalo 6" del tipo trapezoidal, tiene sus límites en $[20, 85]$ y sus pertenencia máximas en $[50, 70]$.

2. Pitch (cabeceo): El universo de discurso considerado para "pitch", es el intervalo $[-60, 50]$, lo que implica que los valores relevantes están comprendidos dentro de este dominio. Las funciones de pertenencia para "pitch" se definen a continuación.

   - Función de pertenencia 1: "intervalo 1" del tipo triangular, tiene sus límites en $[-60, 10]$ y pertenencia máxima en -12.
   - Función de pertenencia 2: "intervalo 2" del tipo triangular, tiene sus límites en $[-30, 12]$ y pertenencia máxima en -5.
   - Función de pertenencia 3: "intervalo 3" del tipo gaussiano, con valor de punto medio en 5 y una desviación estándar de 5.
   - Función de pertenencia 4: "intervalo 4" del tipo triangular, tiene sus límites en $[10, 35]$ y pertenencia máxima en 20.
   - Función de pertenencia 5: "intervalo 5" del tipo triangular, tiene sus límites en $[20, 50]$ y pertenencia máxima en 35.

3. Roll (balanceo): El universo de discurso considerado para "roll", es el intervalo $[-45, 30]$. Las funciones de pertenencia para "roll" se definen a continuación.

   - Función de pertenencia 1: "intervalo 1" del tipo triangular, tiene sus límites en $[-45, -18]$ y pertenencia máxima en -18.
   - Función de pertenencia 2: "intervalo 2" del tipo triangular, tiene sus límites en $[-25, 7]$ y pertenencia máxima en -10.
   - Función de pertenencia 3: "intervalo 3" del tipo gaussiano, con valor de punto medio en -2 y una desviación estándar de 5.
   - Función de pertenencia 4: "intervalo 4" del tipo triangular, tiene sus límites en $[-10, 13]$ y pertenencia máxima en 1.

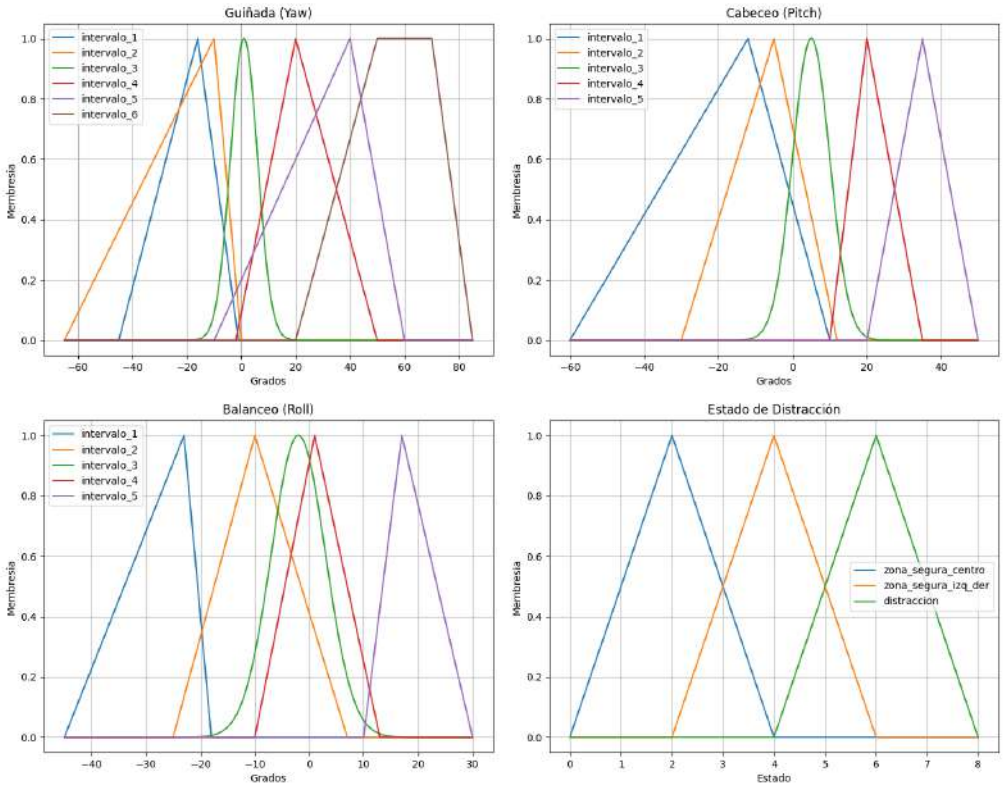
- Función de pertenencia 5: “intervalo 5” del tipo triangular, tiene sus límites en $[10, 30]$ y pertenencia máxima en 17.

4. Estado: El universo de discurso considerado para “estado”, es el intervalo $[0, 8]$. Las funciones de pertenencia para “estado” se definen a continuación.

   - Función de pertenencia 1: “zona segura centro” del tipo triangular, tiene sus límites en $[0, 4]$ y pertenencia máxima en 2.
   - Función de pertenencia 2: “zona segura izq der” del tipo triangular, tiene sus límites en $[2, 6]$ y pertenencia máxima en 4.
   - Función de pertenencia 3: “distracción” del tipo triangular, tiene sus límites en $[4, 8]$ y pertenencia máxima en 6.

La figura 4.12, muestra las funciones de pertenencia definidas para cada universo de discurso.

Figura 4.12
Funciones de pertenencia para cada universo.



Dado que los universos de “yaw”, “pitch” y “roll” cuentan con seis, cinco y cinco funciones de pertenencia respectivamente, el número máximo de reglas combinadas que deben definirse en el sistema difuso se obtiene multiplicando estos valores, lo que resulta en un total de 150 reglas.

Las siguientes reglas difusas definen el sistema, basándose en los antecedentes (yaw, pitch y roll) y el consecuente (estado), y se establecen a continuación:

- **Regla 1**: Si “yaw” es “intervalo 3” y “pitch” es “intervalo 3” y “roll” es “intervalo 3”, entonces “estado” es “safe drive center”.
- **Regla 2**: Si “yaw” es “intervalo 2” y “pitch” es “intervalo 3” y “roll” es “intervalo 3”, entonces “estado” es “safe drive izq_der”.
- **Regla 3**: Si “yaw” es “intervalo 2” y “pitch” es “intervalo 4” y “roll” es “intervalo 3”, entonces “estado” es “safe drive izq_der”.
- **Regla 4**: Si “yaw” es “intervalo 3” y “pitch” es “intervalo 4” y “roll” es “intervalo 3”, entonces “estado” es “safe drive izq_der”.
- **Regla 5**: Si “yaw” es “intervalo 3” y “pitch” es “intervalo 3” y “roll” es “intervalo 2”, entonces “estado” es “safe drive izq_der”.
- **Regla 6**: Si “yaw” es “intervalo 3” y “pitch” es “intervalo 4” y “roll” es “intervalo 2”, entonces “estado” es “safe drive izq_der”.
- **Regla 7**: Si “yaw” es “intervalo 4” y “pitch” es “intervalo 3” y “roll” es “intervalo 2”, entonces “estado” es “safe drive izq_der”.
- **Regla 8**: Si “yaw” es “intervalo 4” y “pitch” es “intervalo 3” y “roll” es “intervalo 3”, entonces “estado” es “safe drive izq_der”.
- **Regla 9**: Si “yaw” es “intervalo 4” y “pitch” es “intervalo 4” y “roll” es “intervalo 2”, entonces “estado” es “safe drive izq_der”.

- **Regla 10**: Si “yaw” es “intervalo 4” y “pitch” es “intervalo 4” y “roll” es “intervalo 3”, entonces “estado” es “safe drive izq_der”.
- **Regla 11**: Si ninguna de las combinaciones anteriores se cumple, es decir, se da cualquier otra combinación de las funciones de pertenencia de “yaw”, “pitch” y “roll”, entonces “estado” es “distracción”.

Como se explicó en la sección 3.2.4, para el proceso de inferencia se utiliza el método de Mamdani, y la defuzzificación se realiza mediante el método del centroide.

Finalmente, la interpretación de los valores nítidos defuzzificados se lleva a cabo mediante un umbral, que se establece en 5, ya que este valor separa los estados de conducción segura de los de distracción. Los valores menores o iguales a este umbral corresponden a situaciones de conducción segura, agrupadas bajo la categoría de “no_distracción”, mientras que los valores superiores indican un estado de distracción.

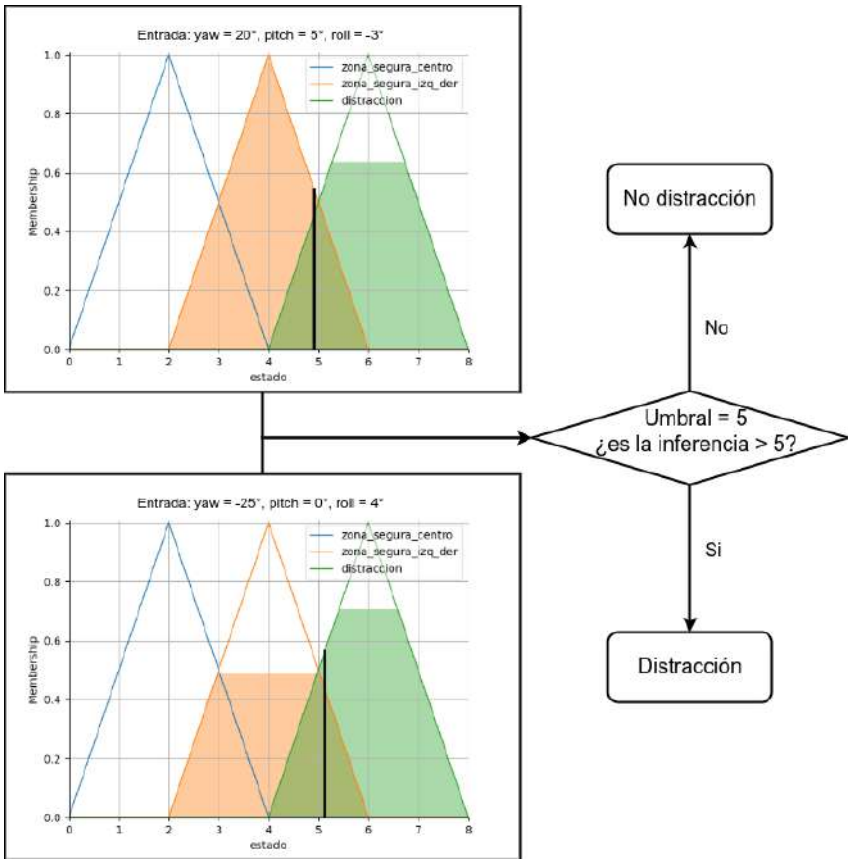
La interpretación de los valores nítidos obtenidos de la inferencia se ilustra en la figura 4.13.

Creación de tabla de decisión difusa

Para implementar el sistema difuso en el Orange Pi 5, realizar la inferencia de reglas en tiempo real podría representar un consumo computacional elevado, lo que afectaría el rendimiento del sistema al realizar detecciones en tiempo real. Una alternativa más eficiente es crear una tabla de decisión difusa, la cual consiste en calcular previamente los resultados de la inferencia para diferentes combinaciones de los valores de entrada (yaw, pitch y roll), basados en la lógica difusa definida previamente.

Esta tabla de decisión permite almacenar los valores de salida en una estructura tipo diccionario con la siguiente forma: “(valor_yaw, valor_pitch, valor_roll): valor_estado”, donde cada entrada corresponde a una combinación específica de los valores de yaw, pitch y roll, y su correspondiente inferencia difusa. Esta tabla se guarda en un archivo “.pkl”, ya que este formato

Figura 4.13
Interpretación de la inferencia difusa.



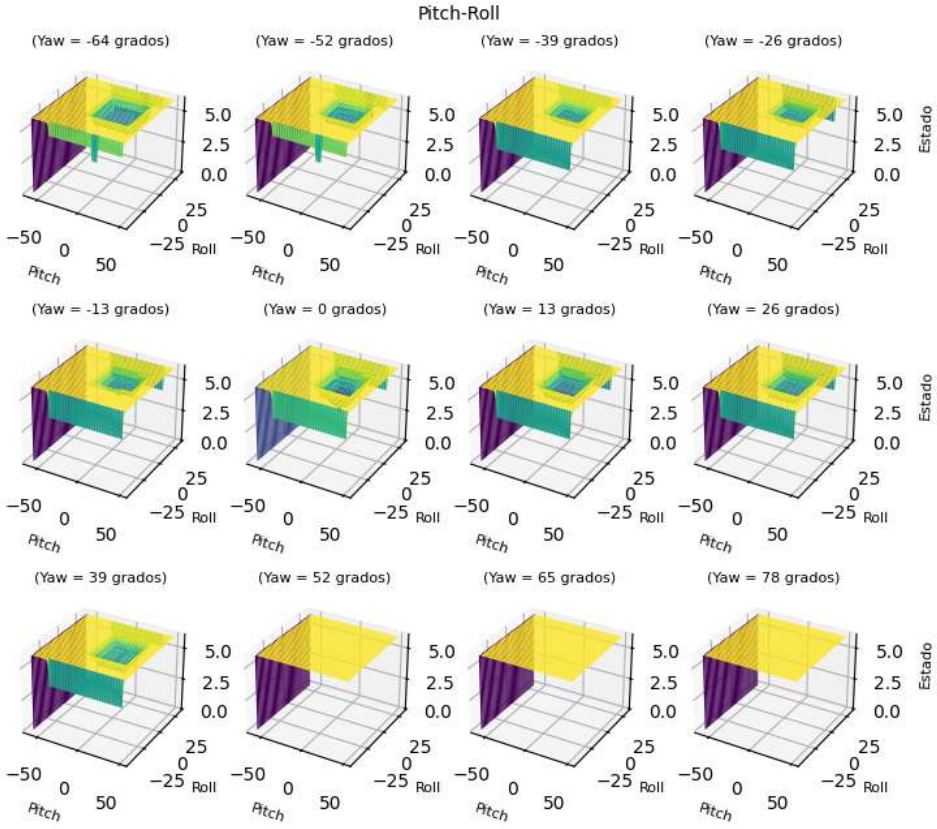
permite almacenar los diccionarios en su forma binaria, lo que facilita un acceso rápido.

Los rangos utilizados para la tabla son los mismos que definen los universos de yaw, pitch y roll, con una resolución de 1° en cada uno. Esto genera un total de 1 273 836 entradas en la tabla. La figura 4.14 muestra superficies en 3D que relacionan pitch, roll y el valor de estado, variando los valores de yaw, que son las superficies almacenadas en la tabla.

Dado que la tabla tiene una resolución de 1 y 3 variables independientes, se hace uso de la técnica de interpolación trilineal definida en la sección 2.2.11, la cual se encarga de aproximar los valores lo más cercano a una inferencia hecha por el modelo difuso original.

Sean yaw, $pitch$ y $roll$ las variables independientes, y $estado = f(yaw, pitch, roll)$. Siguiendo los pasos de la interpolación lineal, se definen yaw_d, $pitch_d$ y $roll_d$, que son iguales a las expresiones 4.3, 4.4 y 4.5 respectivamente.

Figura 4.14
Superficies 3D de Pitch-Roll de la inferencia difusa.



$$yaw_d = \frac{yaw - yaw_0}{yaw_1 - yaw_0} \tag{4.3}$$

$$pitch_d = \frac{pitch - pitch_0}{pitch_1 - pitch_0} \tag{4.4}$$

$$roll_d = \frac{roll - roll_0}{roll_1 - roll_0} \tag{4.5}$$

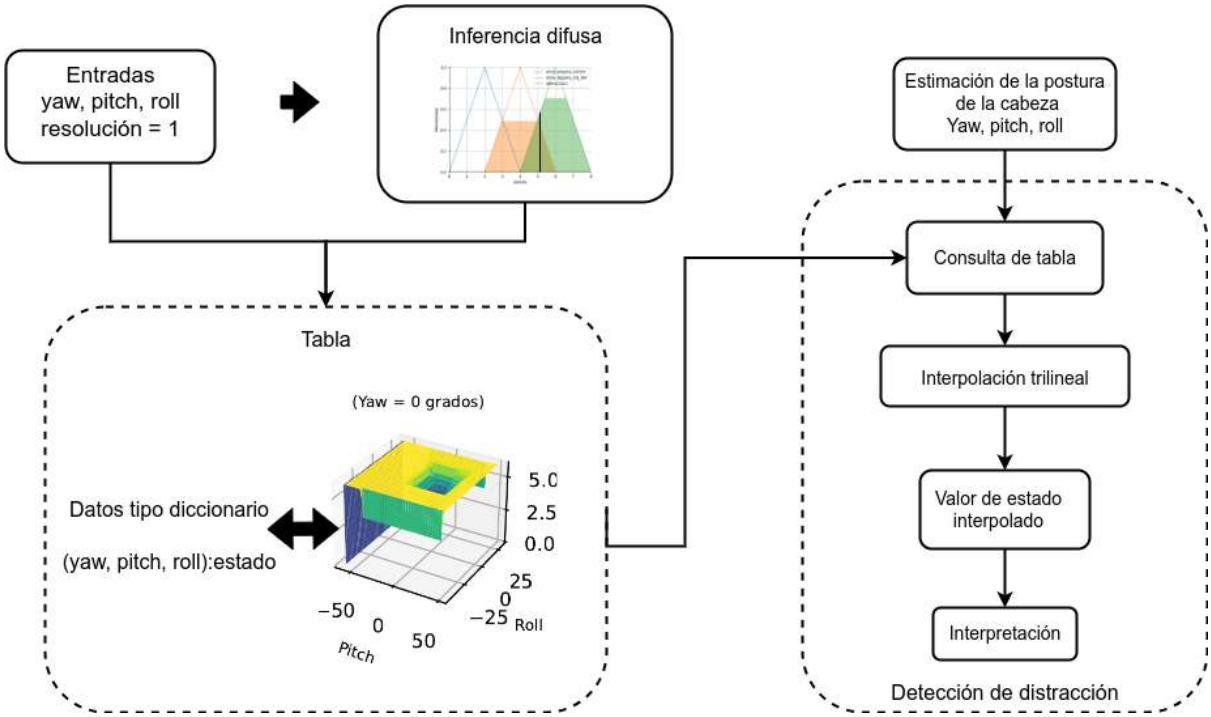
Donde $[yaw_0, yaw_1]$, $[pitch_0, pitch_1]$ y $[roll_0, roll_1]$ representan los intervalos en los que se encuentran las variables independientes. Con los valores extremos de los intervalos, se hallan los ocho puntos utilizando la función $f(yaw_i, pitch_i, roll_i)$ con $i \in \{0, 1\}$. Con dichos valores, se interpola primero en el eje de yaw, luego en el eje de $pitch$ y finalmente en el eje

de $roll$, obteniendo la expresión 4.6, que representa el valor interpolado.

$$f(yaw, pitch, roll) = f(yaw, pitch, roll_0)(1 - roll_d) + f(yaw, pitch, roll_1)roll_d \quad (4.6)$$

La figura 4.15 ilustra de forma gráfica el proceso de generación de la tabla difusa y su posterior aplicación, siendo la interpretación la misma que para el modelo difuso (ver figura 4.13).

Figura 4.15
Generación de la tabla difusa y su aplicación.



4.4. Configuración del Orange Pi 5

Para configurar el sistema, primero se necesita instalar un sistema operativo. El Orange Pi 5 es compatible con Android 12 y Linux, contando este último con una mayor variedad de distribuciones, como Ubuntu, Debian, ArchLinux ARM, entre otras. Existen versiones proporcionadas tanto por el fabricante como por la comunidad de código abierto. El proceso de

instalación del sistema operativo está descrito en el anexo D.

Una vez instalado el sistema operativo, es necesario instalar el controlador de la NPU del RK3588S, llamado RKNPU2, así como la librería que permite utilizarlo en Python, llamada RKNNToolkit2-lite. Para ello, se crea un entorno de Python 3.10 gestionado con `venv`, con el objetivo de evitar conflictos con algunas librerías de Python que pudiera estar utilizando el sistema operativo Linux. La creación del entorno se realiza mediante los siguientes comandos.

```
usuario@usuario-pc$ sudo apt install python3-venv
usuario@usuario-pc$ python3 -m venv ./Documentos/rknn2
```

Luego, se activa el entorno de Python utilizando el siguiente comando:

```
usuario@usuario-pc$ source Documentos/rknn2/bin/activate
```

Estando en el entorno, se procede a buscar el paquete a instalar. Para ello, es necesario dirigirse a la ruta donde se encuentra el paquete. En este caso, se encuentra en `./Descargas/rknn-toolkit2-2.0.0-beta0/rknn-toolkit-lite2/packages` (para ver el proceso de instalación de RKNN-Toolkit2, consulte el anexo G). Luego, se ejecuta el siguiente comando:

```
usuario@usuario-pc -> packages$ pip install \
rknn_toolkit2_lite2-2.0.0b0-cp310-linux-aarch64.whl
```

La figura 4.16 muestra el proceso de instalación del paquete en una terminal de Linux.

Por otro lado, para instalar el controlador RKNPU2, es necesario contar con una versión específica de "rknpu" que viene con el sistema operativo instalado en el Orange Pi 5. Esta debe ser mayor a la versión 0.9.2, lo cual se verifica utilizando el siguiente comando, que proporciona dicha información (ver figura 4.17 para el proceso de manera gráfica).

```
usuario@usuario-pc# cat /sys/kernel/debug/rknpu/version
```

Seguidamente, se accede a la ruta que contiene los archivos binarios de RKNPU2, los cuales deben ser copiados a los directorios del sistema. La ruta de los binarios es `./Descargas/rknn`

Figura 4.16
Instalación de rknn-toolkit-lite2.

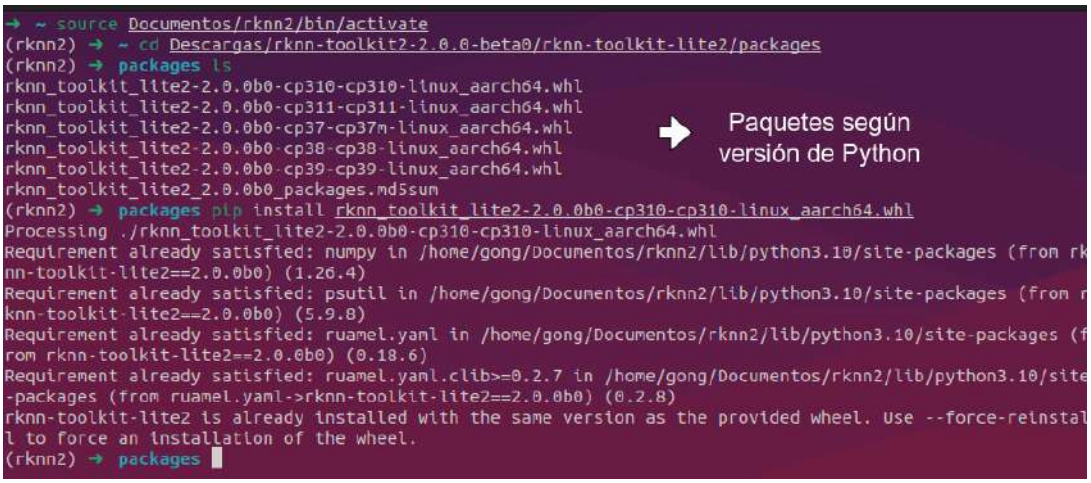


Figura 4.17
Verificación versión RKNPU.

`-toolkit2-2.0.0-beta0/ rknpu2/runtime/Linux/librknn_api/aarch64`, mientras que la ruta del sistema a utilizar es `/usr/lib`. El comando para copiar el binario es el siguiente:

```
usuario@usuario-pc$ sudo cp librknnrt.so /usr/lib
```

La figura 4.18 muestra la instalación y verificación de RKNPU2.

Figura 4.18
Instalación de RKNPU2.

Adicionalmente, en el entorno creado se instalan los paquetes necesarios para ejecutar el sistema de detección de distracción, como *MediaPipe*, *NumPy*, *OpenCV*, entre otros, mediante `pip`.

4.5. Alarma e indicación visual

Para la implementación de la alarma, se utilizó la biblioteca Pygame, la cual puede instalarse fácilmente mediante el comando “pip”. Se selecciona un archivo de sonido en formato MP3 que representa un tono de alarma.

Por otro lado, para el encendido de los LEDs que proporcionan la indicación visual, se utilizan las interfaces de entrada y salida (GPIO) del Orange Pi 5 (ver Anexo B). El control de estas interfaces se realiza mediante la biblioteca WiringOP, la cual permite gestionar los pines GPIO. Para verificar el estado de los pines GPIO y su configuración, se puede ejecutar el siguiente comando en la terminal y ver su salida en la figura 4.19.

```
usuario@usuario-pc$ sudo gpio readall
```

Figura 4.19
Verificación de las GPIO del Orange Pi 5.

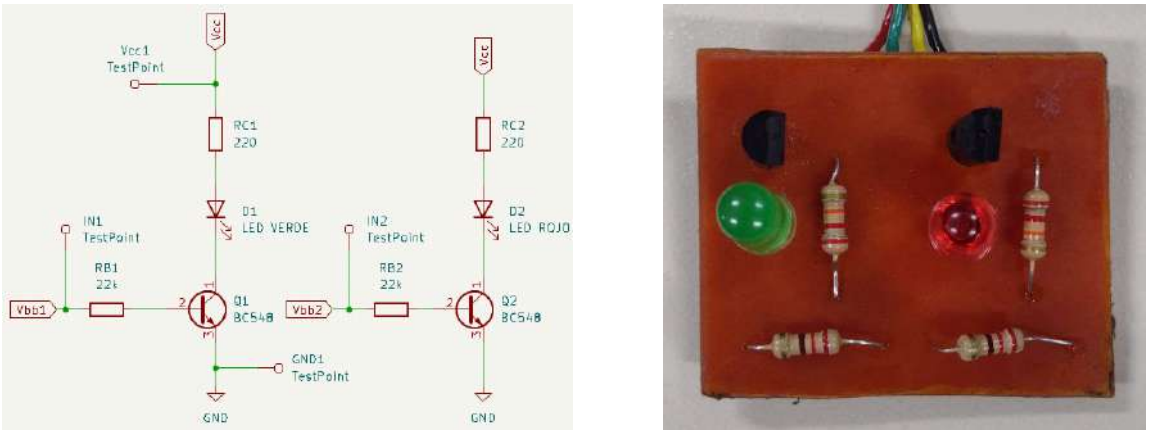
```
➜  ~ sudo gpio readall
[sudo] contraseña para gong:
 +------+-----+----------+--------+---+   OPI5   +---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |   47 |   0 |    SDA.5 |     IN | 1 |  3 || 4  |   |        | 5V       |     |      |
 |   46 |   1 |    SCL.5 |     IN | 1 |  5 || 6  |   |        | GND      |     |      |
 |   54 |   2 |    PWM15 |     IN | 1 |  7 || 8  | 0 | IN     | RXD.0    | 3   | 131  |
 |      |     |      GND |        |   |  9 || 10 | 0 | IN     | TXD.0    | 4   | 132  |
 |  138 |   5 |  CAN1_RX |     IN | 1 | 11 || 12 | 1 | IN     | CAN2_TX  | 6   | 29   |
 |  139 |   7 |  CAN1_TX |     IN | 1 | 13 || 14 |   |        | GND      |     |      |
 |   28 |   8 |  CAN2_RX |     IN | 1 | 15 || 16 | 1 | IN     | SDA.1    | 9   | 59   |
 |      |     |     3.3V |        |   | 17 || 18 | 1 | IN     | SCL.1    | 10  | 58   |
 |   49 |  11 | SPI4_TXD |     IN | 1 | 19 || 20 |   |        | GND      |     |      |
 |   48 |  12 | SPI4_RXD |     IN | 1 | 21 || 22 | 1 | IN     | GPIO2_D4 | 13  | 92   |
 |   50 |  14 | SPI4_CLK |     IN | 0 | 23 || 24 | 0 | IN     | SPI4_CS1 | 15  | 52   |
 |      |     |      GND |        |   | 25 || 26 | 1 | IN     | PWM1     | 16  | 35   |
 +------+-----+----------+--------+---+----++----+---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+   OPI5   +---+--------+----------+-----+------+
➜  ~
```

Cabe resaltar que los voltajes correspondientes al nivel lógico 1 de los GPIO son de 3.3 V. En la figura 4.19, se muestran los niveles lógicos de los GPIO (columna “V”) y su respectiva configuración de entrada o salida en la columna “Mode”. También se incluyen los pines de alimentación de 5 V, 3.3 V y GND, así como la numeración física en el Orange Pi 5 (columna “Physical”), además de su numeración en la librería wPi.

Estos GPIO no son capaces de suministrar suficiente energía a los LEDs de indicación

visual, lo que provoca que no proporcionen la iluminación adecuada. Para solucionar este inconveniente, se utiliza el pin físico 2 del Orange Pi 5, que proporciona 5 V, junto con un transistor BC548 para cada LED en configuración de interruptor. La figura 4.20 muestra, a la izquierda, el circuito utilizado para los LEDs, y a la derecha, su implementación.

Figura 4.20
Circuito de LEDs a la izquierda e implementación a la derecha.



Para la configuración del BC548 como interruptor se asigna un valor conocido de la resistencia de colector, con el fin de calcular el valor de la resistencia de base. El análisis para cada LED se muestra a continuación.

1. **Led Verde**

   Sea $V_{CC} = 5\,\text{V}$, $V_{BB1} = 3{,}3\,\text{V}$, $V_{LEDv} = 2{,}4\,\text{V}$, $V_{BE1} = 0{,}7\,\text{V}$ y $R_{C1} = 220\,\Omega$. Asumiendo que en saturación $V_{CE1} = 0\,\text{V}$, se halla el valor de I_{C1} mediante la siguiente expresión:

$$I_{C1} = \frac{V_{CC} - V_{CE1} - V_{LEDv}}{R_{C1}} = \frac{5\,\text{V} - 0\,\text{V} - 2{,}4\,\text{V}}{220\,\Omega} = 11{,}81\,\text{mA} \tag{4.7}$$

   Con el valor de I_{C1} se halla la corriente de base I_{B1}, como se muestra en la siguiente expresión:

$$I_{B1} = \frac{I_{C1}}{h_{fe1}} = \frac{11{,}81\,\text{mA}}{110} = 0{,}107\,\text{mA} \tag{4.8}$$

   Por lo tanto, se obtiene el valor de R_{B1} con la siguiente expresión:

$$R_{B1} = \frac{V_{BB1} - V_{BE1}}{I_{B1}} = \frac{3{,}3\,\text{V} - 0{,}7\,\text{V}}{0{,}107\,\text{mA}} = 24{,}3\,\text{k}\Omega \tag{4.9}$$

2. **Led Rojo**

   Sea $V_{CC} = 5\,\text{V}$, $V_{BB2} = 3{,}3\,\text{V}$, $V_{LEDr} = 1{,}6\,\text{V}$, $V_{BE2} = 0{,}7\,\text{V}$, $R_{C2} = 220\,\Omega$, y $h_{fe2} = 110$. Asumiendo que en saturación $V_{CE2} = 0\,\text{V}$, se halla el valor de I_{C2} mediante la siguiente expresión:

$$I_{C2} = \frac{V_{CC} - V_{CE2} - V_{LEDr}}{R_{C2}} = \frac{5\,\text{V} - 0\,\text{V} - 1{,}6\,\text{V}}{220\,\Omega} = 15{,}45\,\text{mA} \tag{4.10}$$

   Con el valor de I_{C2} se halla la corriente de base I_{B2}, como se muestra en la siguiente expresión:

$$I_{B2} = \frac{I_{C2}}{h_{fe2}} = \frac{15{,}45\,\text{mA}}{110} = 0{,}140\,\text{mA} \tag{4.11}$$

   Por lo tanto, se obtiene el valor de R_{B2} con la siguiente expresión:

$$R_{B2} = \frac{V_{BB2} - V_{BE2}}{I_{B2}} = \frac{3{,}3\,\text{V} - 0{,}7\,\text{V}}{0{,}140\,\text{mA}} = 18{,}57\,\text{k}\Omega \tag{4.12}$$

Los valores de R_{B1} como R_{B2} se aproximan a un valor de $22\,\text{k}\Omega$, ya que este es un valor comercial. Para ver el PCB del circuito, véase el anexo I.

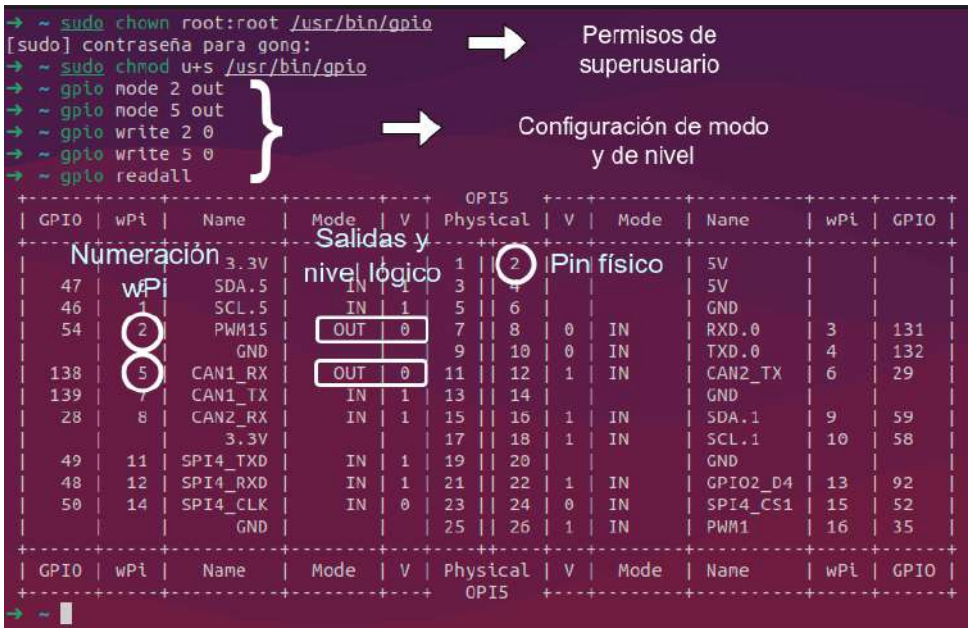
Posteriormente, los GPIO se gestionan utilizando la numeración de wPi, siendo 5 (pin físico 11) para el LED verde y 2 (pin físico 7) para el LED rojo. Estos se inicializan como salidas con un estado inicial de nivel bajo, como se muestra en la figura 4.21. Esta configuración se realiza automáticamente cada vez que se ejecuta el sistema de detección de distracción.

En cuanto a la implementación del código, todas las bibliotecas de audio presentan un problema de incompatibilidad al ejecutarse como superusuario, lo que provoca que el audio deje de funcionar. Dado que la versión de WiringOP para Python se ejecuta con permisos de superusuario, se opta por utilizar directamente el binario “gpio”, invocado desde la terminal de

Linux y que puede ser llamado a través de un script en bash. Para ello, se otorgan permisos de superusuario al programa “gpio” sin la necesidad de usar ‘sudo‘ o acceder como usuario raíz, mediante los siguientes comandos.

```
usuario@usuario-pc$ sudo chown root:root /usr/bin/gpio
usuario@usuario-pc$ sudo chmod u+s /usr/bin/gpio
```

Figura 4.21
Permisos de superusuario, configuración y nivel lógico de los GPIO.

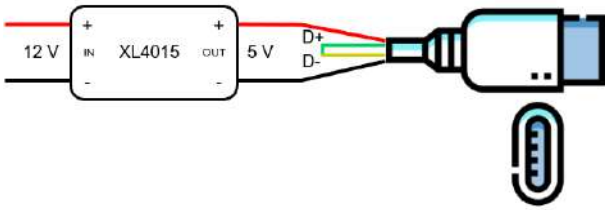


Finalmente, se sigue la lógica descrita en la fase 5 en la sección 3.2.5

4.6. Alimentación

En este proceso, se ajusta el voltaje de salida del convertidor DC-DC a 5.1 V mediante el potenciómetro, esto para dar un margen de caída de voltaje al utilizar el Orange Pi 5 a su máxima capacidad. Luego, dado que el Orange Pi 5 se alimenta a través de USB-C, se conecta la salida del módulo convertidor XL4015 a los cables de alimentación USB cortocircuitando los cables de datos $D+$ y $D-$, de modo que el Orange Pi 5 interprete que no está conectado a una salida USB estándar, sino a una fuente de alimentación, como se muestra en la figura 4.22.

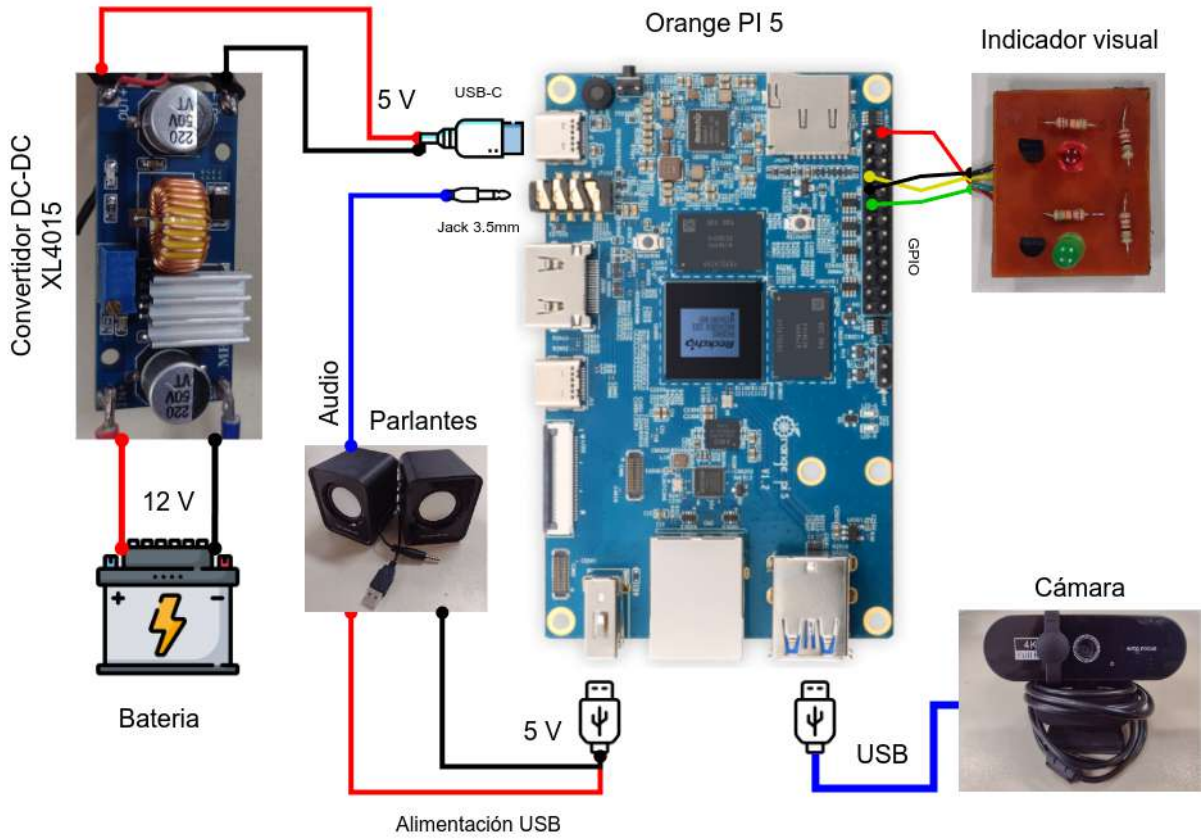
Figura 4.22
Alimentación XL4015 a USB-C.



4.7. Diagrama de conexiones

El diagrama de conexiones del sistema se muestra en la figura 4.23.

Figura 4.23
Diagrama de conexiones del sistema detector de distracción.



4.8. Instalación del sistema de detección de distracción en un vehículo

El montaje del sistema de detección de distracción se realizó colocando el sistema embebido debajo de la radio del vehículo, cerca de la toma de 12 V de la batería para facilitar su alimentación además de protegerlo del sol. El indicador visual se colocó al lado de la radio, en una posición visible para el conductor y pasajeros. Los parlantes se colocaron en la parte superior del tablero, separados para maximizar la cobertura del sonido.

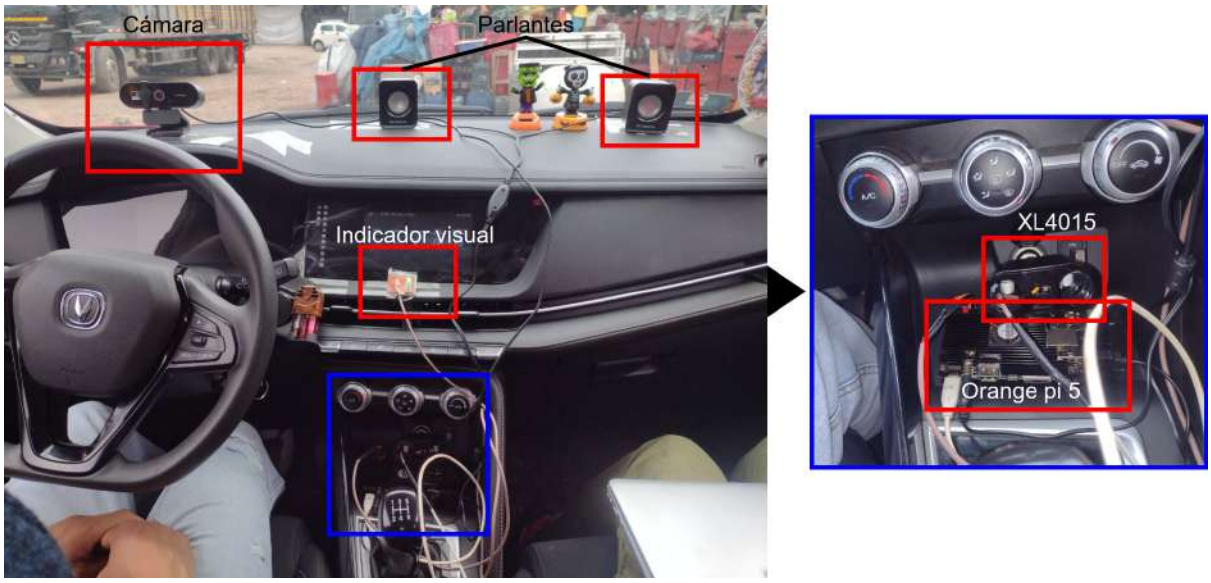
Debido a que el sistema depende de la ubicación de la cámara esta se debe colocar con un ángulo similar al utilizado por DMD para capturar la postura de la cabeza del conductor. Para esto se utiliza el punto de conducción segura $\beta = (0{,}52°, 7{,}06°, 3{,}43°)$, para esto se les pide a los conductores que miren al frente y la predicción obtenida de estimación de la postura de la cabeza es restada de β para obtener valores que sean cercanos a 0°.

Figura 4.24
Ajuste de ángulo de la cámara. (a) conductor 1, (b) conductor 2.

La figura 4.24 muestra el ajuste de la cámara a una posición óptima para el sistema, asegurando que su ángulo coincide con el conjunto de conducción segura, los cuales pertenecen a 2 conductores.

Figura 4.25

Instalación del sistema de detección de distracción en un vehículo.



La figura 4.25 muestra la instalación de los componentes en el vehículo. Para ver el código principal para la detección de distracción ver el anexo H.

Capítulo 5

Pruebas y resultados

5.1. Pruebas y resultados en las estimación de la postura de la cabeza

5.1.1. Pruebas y resultados de las CNNs en computadora

Cada modelo se entrenó 5 veces, comenzando con una tasa de aprendizaje (*learning rate*, lr) de 0.0001 hasta 0.0005. Cada entrenamiento constó de 30 épocas, donde la tasa de aprendizaje se redujo a la mitad cada 10 épocas, con un tamaño de lote de 32, optimizador Adam y 4 hilos de CPU para el generador.

Para evaluar el rendimiento de las redes neuronales convolucionales (CNNs), se utilizaron los conjuntos de datos BIWI y AFLW2000. El desempeño se midió empleando el error absoluto promedio (MAE). A continuación, se muestran los resultados obtenidos tras ejecutar las pruebas en una computadora.

Los resultados de las pruebas con el conjunto de datos BIWI se presentan en la Tabla 5.1. Mientras que, en la tabla 5.2 los modelos con mejores resultados de cada grupo se compararon con algunos modelos del estado del arte y sus resultados obtenidos en la evaluación con el conjunto de datos BIWI. Cabe resaltar que dichos modelos fueron entrenados utilizando el

conjunto de datos 300W-LP y las unidades de los ángulos están en grados sexagesimales.

Tabla 5.1
Resultados de MAE para el conjunto de datos BIWI.

Grupo	Modelo(lr)	Giro	Cabeceo	Balanceo	MAE
1	MNASNet0_5 (lr = 0.0004)	3.81	4.31	3.15	3.76
	MobileNet_V3_Small (lr = 0.0004)	4.26	4.48	3.03	3.92
	MNASNet0_75 (lr = 0.0001)	3.79	3.54	3.08	3.47
	MobileNet_V2 (lr = 0.0005)	3.23	4.25	3.18	3.55
2	MNASNet1_0 (lr = 0.0003)	3.37	3.94	2.63	3.31
	EfficientNet_B0 (lr = 0.0005)	3.71	3.62	2.67	3.33
	MobileNet_V3_Large (lr = 0.0004)	3.67	3.29	2.66	3.21
	MNASNet1_3 (lr = 0.0002)	3.47	3.95	2.68	3.37
3	GoogleNet (lr = 0.0002)	3.99	3.74	2.91	3.55
	EfficientNet_B1 (lr = 0.0005)	3.38	3.83	2.82	3.34
	DenseNet121 (lr = 0.0001)	3.54	3.99	2.66	3.40
	EfficientNet_B2 (lr = 0.0003)	3.89	3.84	2.80	3.51

Tabla 5.2
Comparación de los mejores de cada grupo con modelos del estado del arte para BIWI.

Modelo(lr)	Giro	Cabeceo	Balanceo	MAE
MNASNet0_75 (lr = 0.0001)	3.79	3.54	3.08	3.47
MobileNet_V3_Large (lr = 0.0004)	3.67	3.29	2.66	3.21
EfficientNet_B1 (0.0005)	3.38	3.83	2.82	3.34
HopeNet (α = 2)	5.17	6.98	3.39	5.18
HopeNet (α = 1)	4.81	6.61	3.27	4.90
QuatNet	2.94	5.49	4.01	4.15
WHENet-V	3.60	4.10	2.73	3.48
MHPNet	3.69	4.04	2.58	3.44
6DRepNet	3.24	4.48	2.68	3.47

Los resultados con el conjunto de datos AFLW2000 se muestran en la tabla 5.3 y de la misma manera que con el conjunto de datos anterior los resultados son comparados con modelos del estado del arte (ver tabla 5.4).

Las Figuras 5.1, 5.2 y 5.3 muestran ejemplos de las inferencias de los modelos de estimación de postura de la cabeza realizadas en una computadora para los grupos ligero, intermedio y pesado, respectivamente. Las filas superiores pertenecen al conjunto de datos BIWI, mientras que las inferiores corresponden a AFLW2000. La línea de color azul corresponde a la rotación en el eje Z, la línea roja a la rotación en el eje -X y la línea verde a la rotación en el eje Y, para ver los ejes ver figura 2.3.

Tabla 5.3
Resultados de MAE para el conjunto de datos AFLW2000.

Grupo	Modelo(lr)	Giro	Cabeceo	Balanceo	MAE
1	MNASNet0_5 (lr = 0.0005)	7.07	6.66	5.53	6.42
	MobileNet_V3_Small (lr = 0.0005)	6.87	6.09	4.98	5.98
	MNASNet0_75 (lr = 0.0001)	6.59	5.87	4.53	5.66
	MobileNet_V2 (lr = 0.0001)	5.51	5.60	4.14	5.08
2	MNASNet1_0 (lr = 0.0004)	4.74	5.57	4.04	4.78
	EfficientNet_B0 (lr = 0.0003)	4.85	5.55	4.24	4.88
	MobileNet_V3_Large (lr = 0.0002)	4.67	5.54	3.97	4.72
	MNASNet1_3 (lr = 0.0003)	5.00	5.98	4.53	5.17
3	GoogleNet (lr = 0.0001)	4.75	5.48	4.00	4.74
	EfficientNet_B1 (0.0003)	4.50	4.97	3.56	4.34
	DenseNet121 (lr = 0.0001)	4.13	5.00	3.54	4.22
	EfficientNet_B2 (lr = 0.0002)	4.70	5.13	4.06	4.63

Tabla 5.4
Comparación de los mejores de cada grupo con modelos del estado del arte para AFLW2000.

Modelo(lr)	Giro	Cabeceo	Balanceo	MAE
MobileNet_V2 (lr = 0.0001)	5.51	5.60	4.14	5.08
MobileNet_V3_Large (lr = 0.0002)	4.67	5.54	3.97	4.72
DenseNet121 (lr = 0.0001)	4.13	5.00	3.54	4.22
HopeNet ($\alpha = 2$)	6.47	6.56	5.44	6.16
HopeNet ($\alpha = 1$)	6.92	6.64	5.67	6.41
QuatNet	3.97	5.62	3.92	4.50
WHENet-V	4.44	5.75	4.31	4.83
MHPNet	3.25	5.18	3.81	4.08
6DRepNet	3.63	4.91	3.37	3.97

Figura 5.1
Algunos ejemplos para el grupo ligero.

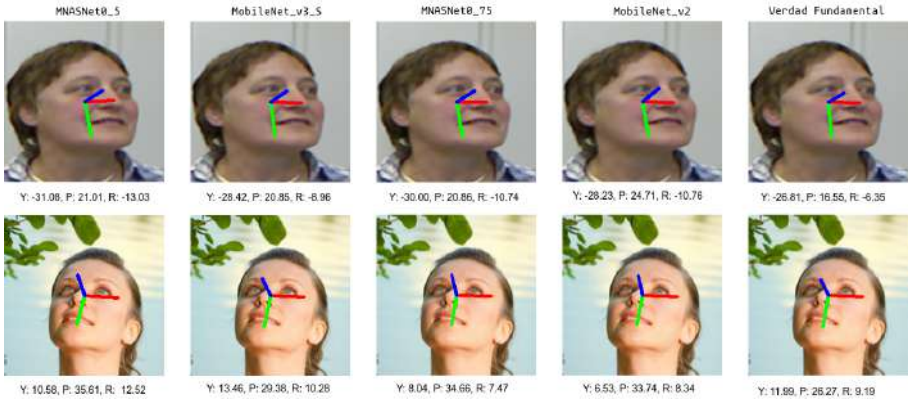


Figura 5.2
Algunos ejemplos para el grupo intermedio.

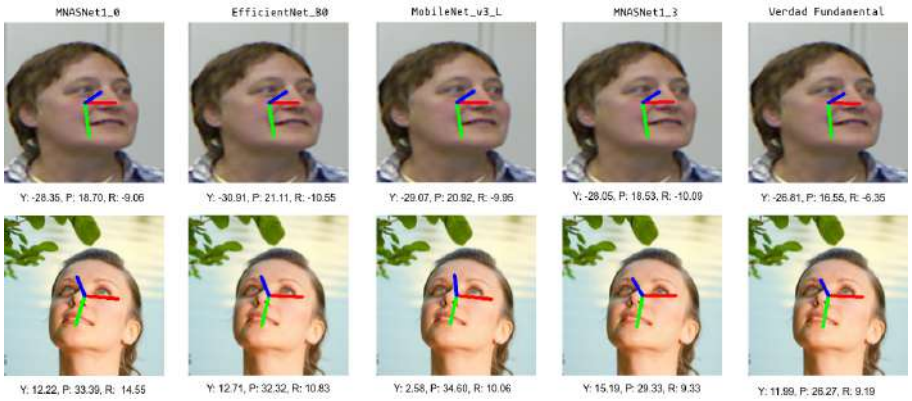
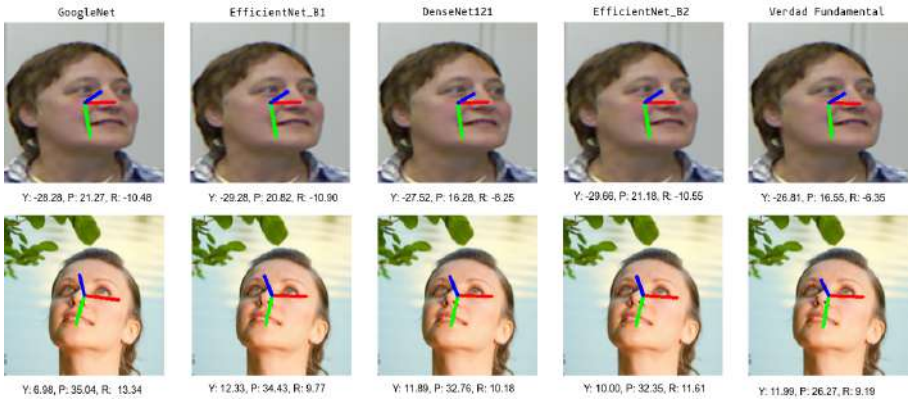


Figura 5.3
Algunos ejemplos para el grupo pesado.

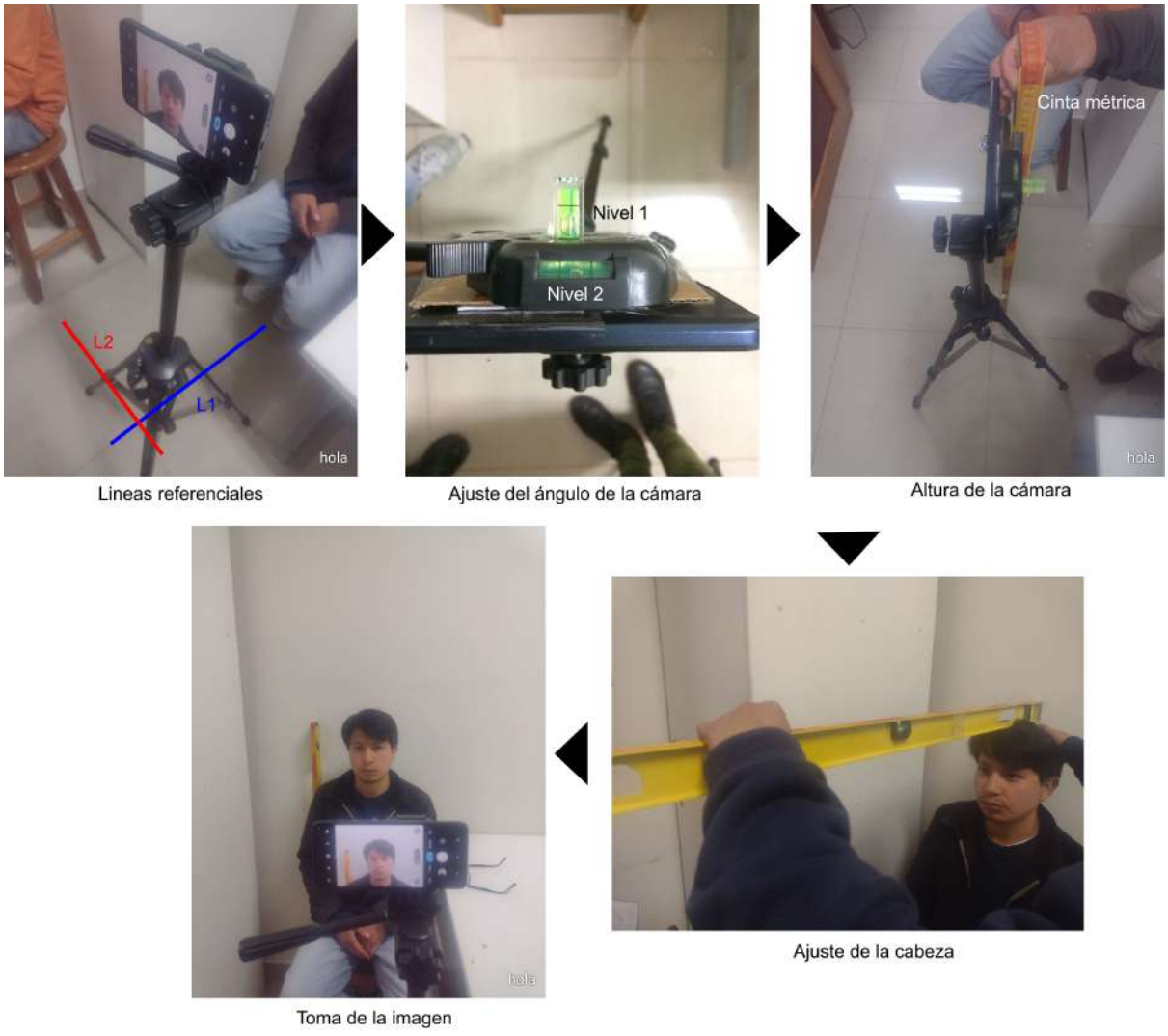


Para corroborar que las redes neuronales cumplen con los resultados obtenidos en las tablas anteriores, se realizó un experimento que consiste en determinar los ángulos de la postura de la cabeza, estableciendo una posición aproximada definida como $yaw = 0°$, $pitch = 0°$ y $roll = 0°$. Para garantizar que los ángulos sean lo más precisos posible, se emplearon dos niveles

para alinear la cámara, dos líneas de referencia tomadas del piso para guiar la colocación de la cabeza y una altura estimada que coincidiera con el centro de la cabeza. Seguidamente, se realizó el mismo procedimiento con la cabeza del voluntario, tratando que esté alineado correctamente con la cámara. Finalmente, se tomó una fotografía para obtener la posición aproximada. La figura 5.4 muestra el proceso de establecimiento de la posición aproximada.

Figura 5.4

Pasos del experimento para corroborar la estimación de la postura de la cabeza.



El siguiente paso del experimento es hacer pasar la imagen tomada a través de las CNNs, obteniendo las predicciones que se muestran en la figura 5.5. Mientras, los errores absolutos para cada uno de los modelos CNN, además de su MAE se muestran en la tabla 5.5.

Figura 5.5
Predicciones de los ángulos de la postura de la cabeza para cada modelo CNN.

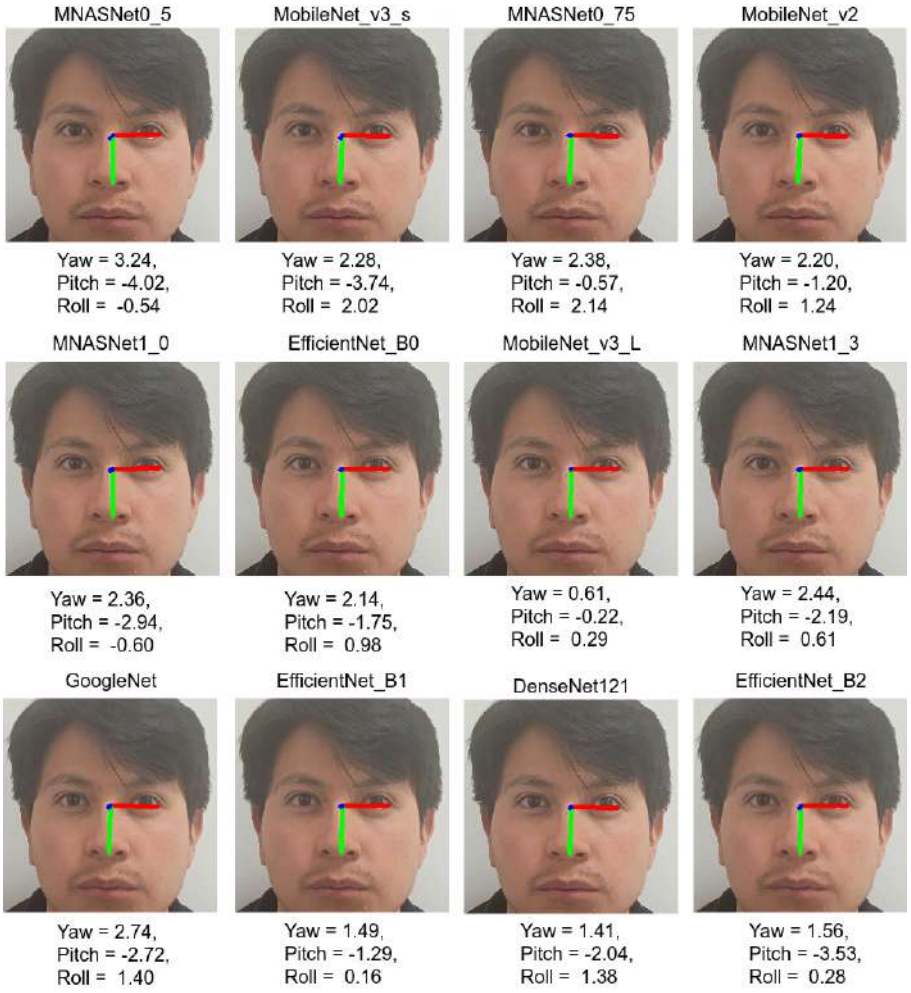


Tabla 5.5
Resultados de errores absolutos y MAE de los modelos CNN en el experimento.

Grupo	Modelo(lr)	Giro	Cabeceo	Balanceo	MAE
1	MNASNet0_5	3.24	4.02	0.54	2.60
	MobileNet_V3_Small	2.28	3.74	2.02	2.68
	MNASNet0_75	2.38	0.57	2.14	1.70
	MobileNet_V2	2.20	1.20	1.24	1.55
2	MNASNet1_0	2.36	2.94	0.60	1.97
	EfficientNet_B0	2.14	1.75	0.98	1.62
	MobileNet_V3_Large	0.61	0.22	0.29	0.37
	MNASNet1_3	2.44	2.19	0.61	1.75
3	GoogleNet	2.74	2.72	1.40	2.29
	EfficientNet_B1	1.49	1.29	0.16	0.98
	DenseNet121	1.41	2.04	1.38	1.61
	EfficientNet_B2	1.56	3.53	0.28	1.79

5.1.2. Pruebas y resultados de las CNNs en el Orange Pi 5

Los resultados de la evaluación de los modelos con el conjunto de datos BIWI, se muestran en la tabla 5.6, mientras los resultados de la evaluación con el conjunto de datos AFLW2000 se muestran en la tabla 5.7. Cabe señalar que estas pruebas fueron hechas en la NPU del Orange Pi 5.

Tabla 5.6
Resultados en el Orange Pi 5 con BIWI.

Grupo	Modelo	Giro	Cabeceo	Balanceo	MAE
1	MNASNet0_5	3.82	4.31	3.17	3.77
	MobileNet_V3_Small	4.24	4.49	**3.02**	3.92
	MNASNet0_75	3.79	**3.55**	3.08	**3.47**
	MobileNet_V2	**3.23**	4.25	3.18	3.55
2	MNASNet1_0	**3.37**	3.94	**2.63**	3.31
	EfficientNet_B0	3.75	3.63	2.70	3.36
	MobileNet_V3_Large	3.81	**3.32**	2.67	**3.27**
	MNASNet1_3	3.47	3.95	2.68	3.37
3	GoogleNet	3.99	**3.74**	2.91	3.55
	EfficientNet_B1	**3.40**	3.84	2.83	**3.35**
	DenseNet121	3.54	3.99	**2.67**	3.40
	EfficientNet_B2	3.92	3.85	2.81	3.53

Tabla 5.7
Resultados en el Orange Pi 5 con AFLW2000.

Grupo	Modelo	Giro	Cabeceo	Balanceo	MAE
1	MNASNet0_5	7.07	6.68	5.53	6.43
	MobileNet_V3_Small	6.93	6.11	4.99	6.01
	MNASNet0_75	6.59	5.88	4.53	5.67
	MobileNet_V2	**5.81**	**5.60**	**4.14**	**5.08**
2	MNASNet1_0	4.74	5.57	4.04	4.78
	EfficientNet_B0	4.89	5.65	4.36	4.97
	MobileNet_V3_Large	**4.68**	**5.53**	**3.98**	**4.73**
	MNASNet1_3	5.00	5.97	4.54	5.17
3	GoogleNet	4.75	5.48	4.00	4.74
	EfficientNet_B1	4.54	5.08	3.63	4.42
	DenseNet121	**4.16**	**5.00**	**3.54**	**4.22**
	EfficientNet_B2	4.75	5.22	4.18	4.72

Por otro lado, para analizar el rendimiento y el consumo energético de los modelos con los conjuntos de datos BIWI y AFLW2000, se presentan las Tablas 5.8 y 5.9 para cada conjunto, respectivamente. Siendo el consumo base del Orange Pi 5 de aproximadamente 0.45 A del cual

se restó y se obtuvo la información de consumo para cada modelo, esta información no es exacta , ya que varía de acuerdo a los periféricos conectados en el momento de la medición (teclado, ratón, pantalla, parlantes, módulos WiFi, etc.)

Tabla 5.8
Carga de los núcleos de la NPU, cuadros por segundo, tiempo de ejecución y consumo de corriente de los modelos en la prueba con BIWI.

Grupo	Modelo	N0 (%)	N1 (%)	N2 (%)	FPS	Tiempo	I (A)
1	MNASNet0_5	12	11	11	122.57	1m47s	0.58
	MobileNet_V3_Small	25	5	5	111.16	1m58s	0.61
	MNASNet0_75	17	16	16	120.79	1m49s	0.67
	MobileNet_V2	20	18	18	120.95	1m49s	0.68
2	MNASNet1_0	20	19	19	119.66	1m50s	0.69
	EfficientNet_B0	38	3	3	36.71	6m	0.5
	MobileNet_V3_Large	37	11	11	91.02	2m25s	0.63
	MNASNet1_3	27	26	26	117.5	1m52s	0.73
3	GoogleNet	38	24	24	74.03	2m58s	0.7
	EfficientNet_B1	42	4	4	28.33	7m46s	0.49
	DenseNet121	40	26	27	28.3	7m47s	0.49
	EfficientNet_B2	44	5	5	25.69	8m34s	0.5

Tabla 5.9
Carga de los núcleos de la NPU, cuadros por segundo, tiempo de ejecución y consumo de corriente de los modelos en la prueba con AFLW2000.

Grupo	Modelo	N0 (%)	N1 (%)	N2 (%)	FPS	Tiempo	I (A)
1	MNASNet0_5	4	4	4	47.98	41s	0.42
	MobileNet_V3_Small	10	2	2	43.19	45s	0.41
	MNASNet0_75	6	6	6	47.28	41s	0.44
	MobileNet_V2	8	7	7	48.26	40s	0.46
2	MNASNet1_0	8	7	7	48.17	40s	0.46
	EfficientNet_B0	46	4	4	43.52	45s	0.59
	MobileNet_V3_Large	19	5	5	46.62	42s	0.48
	MNASNet1_3	11	11	11	48.24	40s	0.48
3	GoogleNet	24	15	15	45.21	43s	0.61
	EfficientNet_B1	52	5	5	35.35	55s	0.53
	DenseNet121	52	34	35	37.87	51s	0.69
	EfficientNet_B2	47	6	6	28.48	1m09s	0.65

En la Figura 5.6 se muestra la carga de los núcleos en un diagrama de barras, mientras que en la Figura 5.7 se muestran los cuadros por segundo alcanzados en la prueba con BIWI.

En la Figura 5.8 se muestra la carga de cada uno de los núcleos de la NPU, mientras que en la Figura 5.9 se presentan los cuadros por segundo (FPS) obtenidos en la prueba con el

Figura 5.6
Carga de la NPU por núcleo con BIWI.

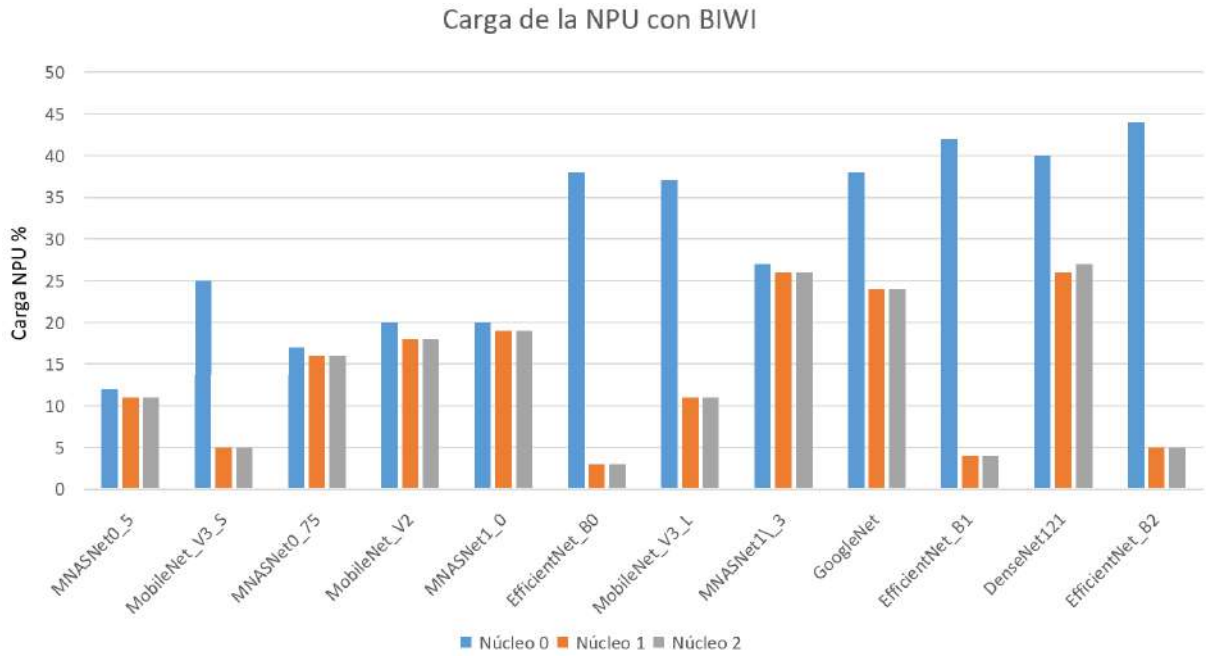
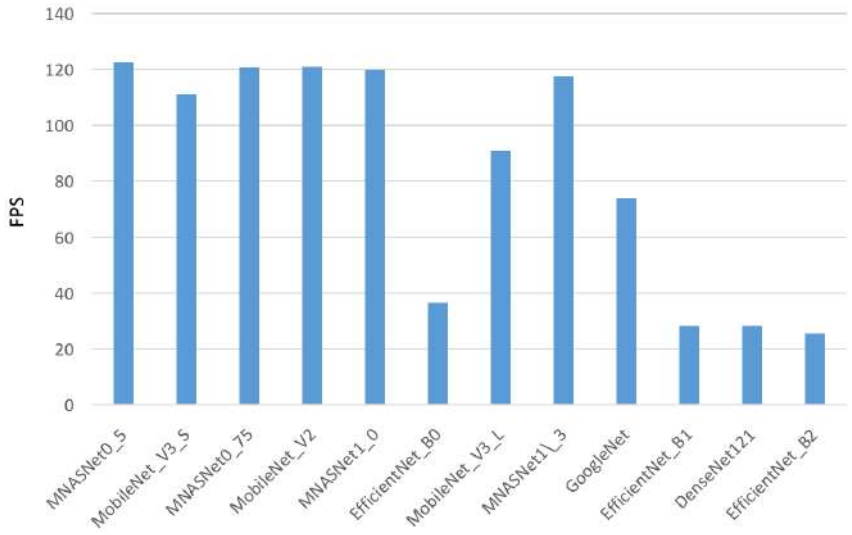


Figura 5.7
Cuadros por segundo alcanzados por los modelos con BIWI.



conjunto de datos AFLW2000.

Figura 5.8
Carga de la NPU por núcleo con AFLW2000.

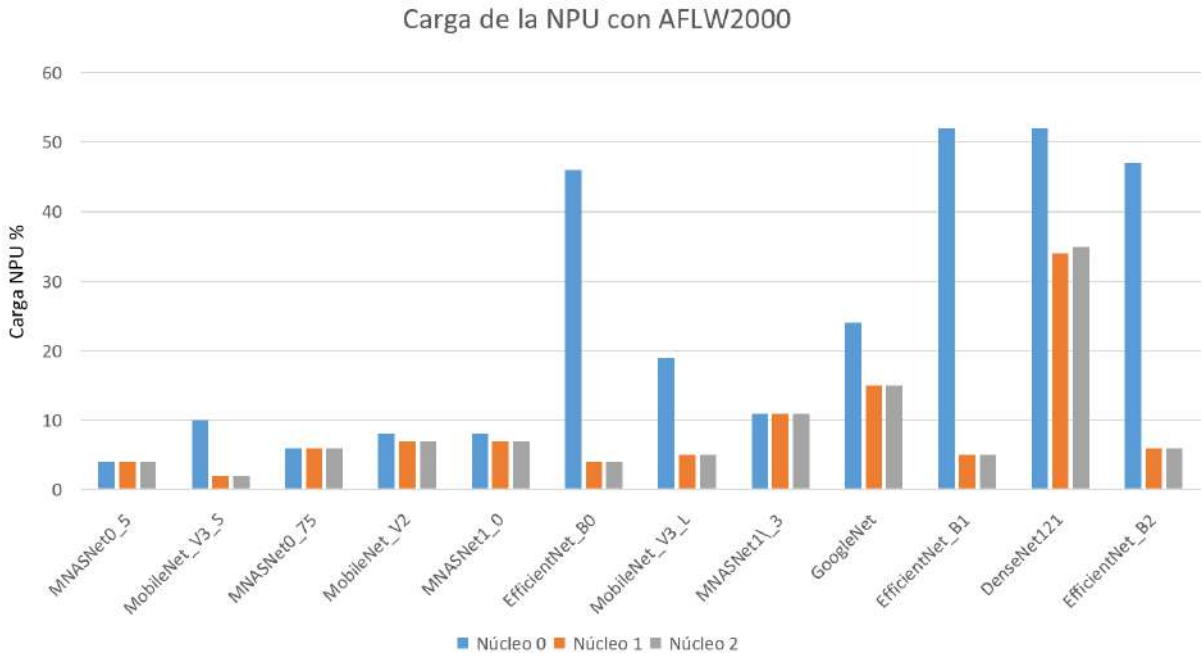
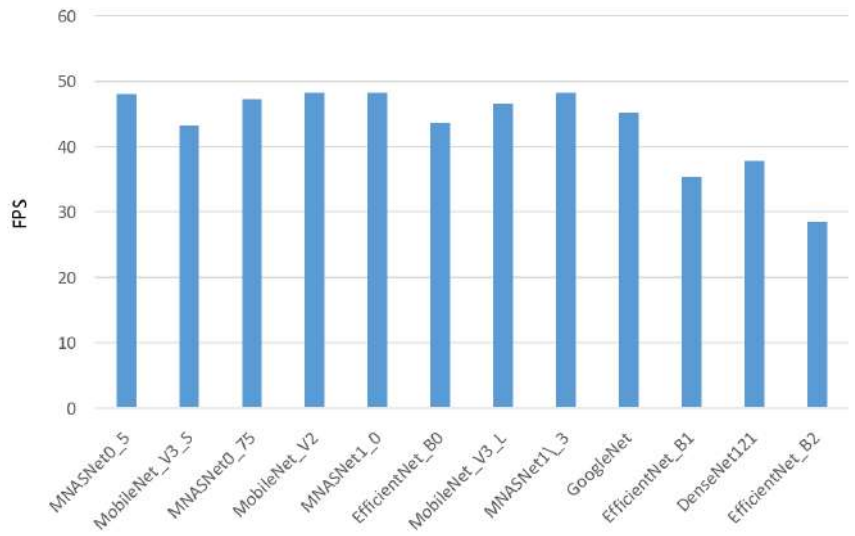


Figura 5.9
Cuadros por segundo alcanzados por los modelos con AFLW2000.



5.2. Pruebas y resultados de los algoritmos de distracción

Las pruebas fueron realizadas con los datos del grupo Z, concretamente con los conductores 31 al 34, que proporcionan datos de un total de 150 378 imágenes, siendo de conducción segura 46 858. En la sección 2.2.3 se definen las métricas de evaluación que son aplicadas a segmentos individuales. Para reflejar de mejor manera el comportamiento de cada segmento y su aporte en tamaño, se utiliza el promedio ponderado para la medición del número total de valores que superan el umbral (OT %), mientras que para las mediciones de OT, secuencias de video de k cuadros y el número máximo de valores continuos que superan el umbral (MCOT), se emplea el promedio simple. Además, se realiza una prueba para contar la cantidad de veces que se supera el umbral de distracción de manera continua. Específicamente, el conteo se incrementa en uno cada vez que se detecta que el número de cuadros consecutivos que superan el umbral excede un valor predeterminado.

(Zhao et al., 2020) utiliza un valor de 5 cuadros consecutivos para realizar el conteo, lo que representa 0.25 s en un total de 20 FPS. Para estas pruebas, se emplea el valor de 8 cuadros consecutivos, que representa 0.27 s, ya que, se tienen 30 FPS.

Los resultados mostrados en la tabla 5.10, se realizaron únicamente utilizando los datos de conducción segura sin dividirlos. Esto para tener una visión general del comportamiento de los métodos con el conjunto de conducción segura. Donde "Conteo" representa el número de cuadros que superan continuamente el umbral de distracción, "umbral" y "umbral 2" representan a los métodos de umbralización y umbralización modificado respectivamente.

Tabla 5.10
Resultados para la zona de conducción segura.

Método	OT %	OT	k	MCOT	Conteo (8)
Umbral	21.34	27.45	128.67	19.79	0.59
Umbral 2	16.29	20.96	128.67	14.35	0.52
Fuzzy	17.34	22.31	128.67	13.38	0.64
Tabla fuzzy	18.47	23.77	128.67	14.26	0.66

Por otro lado, los resultados si se usa el conjunto de conducción segura dividida en

centro, derecha e izquierda, además de las acciones del conductor, se tienen las tablas 5.11, 5.12, 5.13 y 5.14 para el método de umbralización, su modificación, lógica fuzzy y tabla difusa respectivamente.

Tabla 5.11
Resultados de las zonas seguras y los estados de distracción del método de umbrales.

Estado	OT %	OT	k	MCOT	Conteo (8)
Centro	0	0	77.08	0	0
Derecha	12.81	17.89	139.62	15.49	0.48
Izquierda	23.43	56.94	243.03	47.77	1.33
Buscar al costado	89.85	16.03	17.84	15.87	0.82
Hablar con el pasajero	92.94	74.48	80.14	74.02	3.34
Arreglarse el cabello	89.75	109.67	122.19	105.76	4.33
Texteo izquierda	86.35	81.77	94.7	79.49	3.12
Texteo derecha	79.65	90.1	113.13	87.83	3.46
Radio	79.18	245.52	310.1	239.67	9.02

Tabla 5.12
Resultados de las zonas seguras y los estados de distracción del método de umbrales 2.

Estado	OT %	OT	k	MCOT	Conteo (8)
Centro	0.00	0.00	77.08	0.00	0
Derecha	10.2	14.24	139.62	12.69	0.39
Izquierda	18.03	43.81	243.03	34.11	1.16
Buscar al costado	88.67	15.82	17.84	15.62	0.82
Hablar con el pasajero	91.91	73.66	80.14	72.14	3.38
Arreglarse el cabello	87.46	106.87	122.19	100.87	4.38
Texteo izquierda	82.36	77.99	94.7	74.18	3.08
Texteo derecha	76.59	86.64	113.13	82.89	3.42
Radio	75.08	232.83	310.1	223.4	8.77

Tabla 5.13
Resultados de las zonas seguras y los estados de distracción del método de lógica difusa.

Estado	OT %	OT	k	MCOT	Conteo (8)
Centro	0	0	77.08	0	0
Derecha	9.5	13.27	139.62	10.59	0.36
Izquierda	18.79	45.66	243.03	30.95	1.36
Buscar al costado	92.05	16.42	17.84	16.32	0.84
Hablar con el pasajero	93.49	74.92	80.14	73.55	3.41
Arreglarse el cabello	84.31	103.02	122.19	95.57	4.54
Texteo izquierda	82.08	77.72	94.7	72.29	3.3.30
Texteo derecha	79.25	89.65	113.13	84.09	3.70
Radio	77.49	240.31	310.1	226.46	9.48

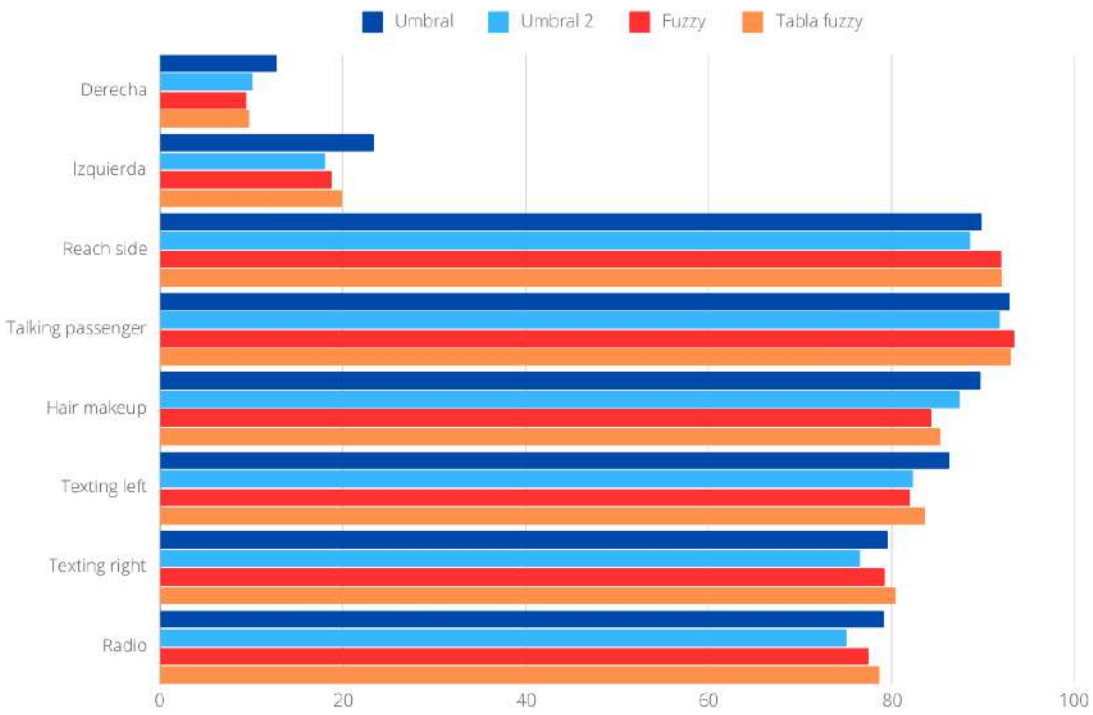
La Figura 5.10 muestra un gráfico de barras donde se observa el valor de OT % para cada

Tabla 5.14
Resultados de las zonas seguras y los estados de distracción del método de tabla de lógica difusa.

Estado	OT %	OT	k	MCOT	Conteo (8)
Centro	0	0	77.08	0	0
Derecha	9.76	13.62	139.62	11.01	0.36
Izquierda	19.98	48.56	243.03	33.36	1.42
Buscar al costado	92.08	16.43	17.84	16.30	0.84
Hablar con el pasajero	93.6	75.02	80.14	73.53	3.41
Arreglarse el cabello	85.37	104.32	122.19	96.86	4.52
Texteo izquierda	83.72	79.28	94.7	74.13	3.30
Texteo derecha	80.51	91.08	113.13	85.98	3.69
Radio	78.69	244.02	310.1	231.29	9.46

una de las categorías definidas en las tablas anteriores, sin contar con “centro”, dado que este es cero para todos los métodos. En dicha figura se destaca que, para las categorías de “derecha” e “izquierda”, las barras son pequeñas, mientras que para las acciones de distracción estas barras son altas.

Figura 5.10
Porcentaje de OT para conducción segura y acciones de distracción.



5.3. Pruebas y resultados del sistema en un entorno real

En esta sección, se presenta una evaluación del prototipo experimental del sistema de detección de distracción en conductores, que utiliza la postura de la cabeza para identificar posibles estados de distracción en tiempo real. Las pruebas se hicieron a 2 conductores voluntarios.

Se utilizó el modelo MNASNet1-0 para la estimación de la postura de la cabeza, mientras que para la detección se utilizó la tabla difusa. Cada acción fue realizada 50 veces por conductor y se evaluaron los eventos completos, incluyendo tanto la distracción como el retorno a la posición neutral del conductor. En total, las pruebas tuvieron una duración aproximada de 6 horas.

El sistema fue probado en situaciones que representan distracciones previamente estudiadas: “buscar al costado”, “hablar con el pasajero”, “arreglarse el cabello”, “texteo izquierda”, “texteo derecha”, así como ajustar la “radio”. Todas las pruebas comenzaron desde una posición de conducción segura central, representando el estado inicial de no distracción del conductor.

La figura 5.11 muestra las acciones realizadas durante las pruebas del desempeño del sistema en distracción.

Los resultados presentados a continuación se basan en el conteo de ocho cuadros consecutivos que superan el umbral y en la activación de la alarma para indicar una posible distracción. Las métricas de exactitud, precisión, sensibilidad (tasa de valores positivos) y puntuación F1 (representadas en las expresiones 5.1, 5.2, 5.3 y 5.4, respectivamente) fueron calculadas de manera individual para cada acción, considerando tanto los eventos de distracción como los de regreso al centro.

$$Exactitud = \frac{VN + VP}{VN + FP + VP + FN} \tag{5.1}$$

Figura 5.11
Acciones realizadas en las pruebas de distracción.

$$Precision = \frac{VP}{VP + FP} \tag{5.2}$$

$$Sensibilidad = \frac{VP}{VP + FN} \tag{5.3}$$

$$Puntuacion - F1 = 2 * \frac{Precision * Sensibilidad}{Precision + Sensibilidad} \tag{5.4}$$

Donde, VP representa los verdaderos positivos, es decir, los casos en los que el sistema identifica correctamente una distracción cuando esta realmente ocurre. VN representa los verdaderos negativos, que son los casos en los que el sistema identifica correctamente que no hay distracción cuando en realidad no la hay. FP representa los falsos positivos, o los casos en los que el sistema detecta una distracción cuando en realidad no la hay. FN representa los falsos negativos, o los casos en los que el sistema no detecta una distracción cuando en realidad sí la hay.

Además, la exactitud mide la proporción total de predicciones correctas (ya sean distracción o no). La precisión evalúa cuán efectivo es el sistema al identificar correctamente las distracciones entre todos los casos que predice como distracción. La sensibilidad mide la capacidad del sistema para detectar las distracciones cuando realmente ocurren y la puntuación F1 proporciona un balance entre precisión y sensibilidad.

1. **Conductor 1**

La figura 5.12 muestra algunas imágenes de las pruebas realizadas con el conductor 1.

Figura 5.12
Pruebas con el conductor 1.

La tabla 5.15 muestra los resultados obtenidos de las pruebas con el conductor 1.

2. **Conductor 2**

La figura 5.13 muestra algunas imágenes de las pruebas realizadas con el conductor 2.

La tabla 5.15 muestra los resultados obtenidos de las pruebas con el conductor 1.

Tabla 5.15
Resultados de las pruebas en entorno real para el conductor 1.

Acción	Exactitud %	Precisión %	Sensibilidad %	Puntuación F1 %
Buscar al costado	96	92.59	100	96.15
Hablar con el pasajero	92	86.21	100	92.59
Arreglarse el cabello	97	94.34	100	97.09
Texteo izquierda	89	89.80	88	88.89
Texteo derecha	95.05	94.12	96	95.05
Radio	96	96	96	96

Figura 5.13
Pruebas con el conductor 2.

Tabla 5.16
Resultados de las pruebas en entorno real para el conductor 2.

Acción	Exactitud %	Precisión %	Sensibilidad %	Puntuación F1 %
Buscar al costado	88	80.65	100	89.29
Hablar con el pasajero	94	89.29	100	94.34
Arreglarse el cabello	90	85.71	96	90.57
Texteo izquierda	96	92.59	100	96.15
Texteo derecha	93	93.88	92	92.93
Radio	88	86.54	90	88.24

Para determinar el resultado general del sistema, se consideró los resultados obtenidos para ambos conductores. La tabla 5.17 muestra los resultados generales obtenidos por el sistema de detección de distracción con ambos conductores.

Respecto a las actividades de conducción segura como girar a la izquierda y derecha (ver retrovisores, giros suaves y giros pronunciados), se pidió a los conductores que realicen estas acciones de manera espontánea. Se observó que el sistema respondió de forma satisfactoria al

Tabla 5.17
Resultados generales.

Acción	Exactitud %	Precisión %	Sensibilidad %	Puntuación F1 %
Buscar al costado	92	86.21	100	92.59
Hablar con el pasajero	93	87.72	100	93.46
Arreglarse el cabello	93.5	89.91	98	93.78
Texteo izquierda	92.5	91.26	94	92.61
Texteo derecha	94	94	94	94
Radio	92	91.18	93	92.08
Promedio	**93.58**	**90.05**	**96.50**	**93.09**

realizar giros suaves, sin embargo, al realizar giros pronunciados o mirar a los retrovisores, se generó inestabilidad al sistema.

Estos resultados se discuten de manera más detallada en la sección discusión (ver sección 6).

Capítulo 6

Discusión de resultados

Al analizar los resultados de la estimación de postura de la cabeza en una computadora para el conjunto de datos BIWI (ver tabla 5.1), se observa que el mejor resultado en términos de error absoluto promedio (MAE) fue alcanzado por MNASNet0-75 con un MAE de 3.55 en el grupo ligero, MobileNet-v3-Large con un MAE 3.21 en el grupo intermedio y EfficientNet B1 con un MAE de 3.34 en el grupo pesado. Es importante notar que otros modelos dentro del grupo pesado, como DenseNet121, también mostraron un rendimiento competitivo. Además, el grupo intermedio destacó por obtener resultados que en algunos casos superaron a los modelos de los otros grupos; en particular MNASNet1-0 logró el segundo mejor resultado global.

Para el conjunto de datos AFLW2000 (ver tabla 5.3) los modelos con mejores resultados fueron MobileNet-v2 con un MAE de 5.08 para el grupo ligero, MobileNet-v3-Large con 4.72 de MAE en el grupo intermedio y DenseNet121 con un MAE de 4.22 en el grupo pesado.

Estos modelos fueron comparados con modelos del estado del arte para la estimación de la postura de cabeza (ver tablas 5.2 y 5.4), Con BIWI, MobileNet-v3-Large obtuvo el mejor resultado, seguido por EfficientNet B1, ambos superando al modelo de referencia 6DRepNet, mientras que MNASNet0-75 mostró resultados comparables a 6DRepNet. Con AFLW2000, DenseNet121 del grupo pesado se ubicó en tercer lugar detrás de 6DRepNet y MHPNet, mientras los modelos del grupo ligero e intermedio presentaron aproximadamente un grado sexagesimal más de error.

La diferencia de rendimiento observada entre ambos conjuntos de datos podría estar relacionada con el mayor rango de ángulos y las características más complejas de AFLW2000, lo cual representa un desafío adicional para los modelos ligeros e intermedios. Sin embargo, estas diferencias no impactan significativamente en la detección de distracción, ya que esta tarea no requiere de rangos demasiado grandes (ver gráfico de densidad 4.11).

En cuanto a lo obtenido en el experimento realizado para corroborar los resultados obtenidos por los modelos con los conjuntos de datos (ver tabla 5.5), se observa que todos obtienen resultados dentro de lo esperado y rango de MAE obtenido tanto con BIWI y AFLW2000, siendo que en este experimento el mejor resultado lo obtiene MobileNet-v3-large con 0.37 de MAE y el peor lo obtiene MobileNet-v3-small con un MAE de 2.68.

Respecto a las pruebas realizadas en la NPU del Orange Pi 5 (ver tablas 5.6 y 5.7), se observa que no existen variaciones significativas en el MAE en comparación con los resultados obtenidos en computadora, sugiriendo que no hay pérdidas en la estimación de la postura de la cabeza. No obstante, se observan diferencias en la forma en la que los modelos interactúan con la NPU en términos de rendimiento (ver tablas 5.8 y 5.9). No todos los modelos logran aprovechar los tres núcles de la NPU (N0, N1 y N2) (ver figuras 5.6 y 5.8), entre ellos, MobileNet-v3-small del grupo ligero, EfficientNet B0 y MobileNet-v3-Large del grupo intermedio, y EfficientNet B1 y EfficientNet B2 del grupo pesado, además de modelos como GoogleNet y DenseNet121, que presentan un uso limitado. Esto podría deberse a la necesidad de optimizaciones adicionales o a la complejidad de las arquitecturas.

En términos de cuadros por segundo (FPS) (ver figuras 5.7 y 5.9), se observa que los modelos suelen tener un mejor rendimiento en BIWI en comparación con AFLW2000, esto podría estar relacionado con la forma en la que se cargan los datos de cada conjunto. Los modelos con mejor rendimiento en FPS con BIWI son MNASNet0-5 (122.57 FPS), MNASNet0-75 (120.79 FPS) y MobileNet-V2 (120.95 FPS) en el grupo ligero, MNASNet1-0 (119.66 FPS) y MNASNet1-3 (117.5 FPS) en el grupo intermedio y GoogleNet (74.03 FPS) en el grupo pesado. Con AFLW2000, los mejores modelos siguen siendo MNASNet0-5, MNASNet0-75 y MobileNet-v2 en el grupo ligero, MNASNet1-0 y MNASNet1-3 para el grupo intermedio,

GoogleNet en el grupo pesado.

En cuanto al consumo energético, se observa que los modelos tienen un consumo similar siendo MNASNet1-3 el modelo con mayor consumo (730 mA con BIWI) y EfficientNet B1 y DenseNet121 los de menor consumo. Con AFLW2000, DenseNet121 y MobileNet-v3-small presentan el mayor consumo. Los modelos que no usan todos los núcleos simetricamente tienden a tener un consumo energético más bajo en BIWI, mientras que en AFLW2000 sucede lo contrario. En general, mientras menos FPS procesa un modelo, menor es su consumo energético, lo que sugiere que ninguno de los modelos presentan problemas energéticos que afecten su rendimiento.

Después de analizar los modelos en computadora y en el Orange Pi 5, se tiene que MNASNet1-0 del grupo intermedio es el modelo con mejor rendimiento en términos de FPS, uso de núcleos, consumo energético y MAE, aunque otros modelos también podrían ser adecuados según se requieran.

En el análisis de los métodos de detección de distracción utilizando el conjunto de conducción segura sin divisiones (ver tablas 5.10), el método modificado de umbralización (umbral 2), fue más efectivo en la detección de no distracción, obteniendo un 16.29 % en el número total de valores que superan el umbral (OT %), mientras el método de lógica difusa obtuvo un promedio de 13.38 valores en el máximo número continuo de valores que superan el umbral (MCOT). Estos resultados, sin embargo, reflejan mayor enfoque en la no distracción.

Al dividir el conjunto de conducción segura en tres subgrupos y usar acciones de distracción (ver tablas 5.11, 5.12, 5.13 y 5.14), se observa un contexto más apropiado para la no distracción, en particular centro, OT % es de 0 % para todos, indicando que cuando el conductor mira al frente no se registra distracción. Por otro lado, para derecha, la lógica difusa muestra un resultado ligeramente superior a las demás con un OT % de 9.5 %, mientras que en MCOT, la tabla difusa de este método muestra el menor valor promedio con 10.59. Para izquierda, el resultado más bajo lo obtiene la umbralización modificada con 18.03 % para OT %, pero para MCOT lo obtiene la técnica de lógica difusa.

Para las acciones de distracción (ver figura 5.10), el método de lógica difusa obtuvo un OT % de 92.05 % y 93.49 % en“buscar al costado” y “hablar con el pasajero”, respectivamente. En “arreglarse el cabello”, “texteo izquierda” y “radio” la umbralización obtuvo mejores resultados (89.75 %, 86.35 % y 79.18 %). Para “texteo derecha”, la tabla difusa superó a los demás métodos con un OT % de 80.51 %. Si bien la tabla difusa presenta resultados ligeramente inferiores en no distracción, compensa esta pérdida en la detección de distracción, particularmente en algunas acciones como “arreglarse el cabello”, “texteo izquierda” y “radio”. La umbralización modificada, por otro lado, tiene un desempeño mejor en la detección de no distracción, pero no en distracción.

Al analizar el conteo de cuadros consecutivos que superan el umbral para las zonas seguras y las acciones de distracción, se observa que en cada segmento el conteo alcanza, por ejemplo, 8 conteos para la acción “radio” con umbralización y 9 conteos con lógica difusa. Este resultado es favorable para las detecciones de distracción, sin embargo, las actividades seguras se ven afectadas, lo cual es de esperarse, ya que tienen un MCOT (máximo número continuo de valores que superan el umbral) promedio superior a 8. Es importante destacar que estas actividades seguras, en comparación con mirar al frente (zona central), ocurren con menor frecuencia. Por esta razón, mirar al frente se considera el estado base por defecto de la no distracción en el análisis del sistema.

En general, la lógica difusa muestra mejores resultados y al ser expresado en tabla difusa mantiene ese rendimiento. Sin embargo, estos enfoques basados en la estimación de postura de la cabeza presentan limitaciones significativas, especialmente en su dependencia de los ángulos de la cámara, lo cual dificulta su aplicación. Además, estos métodos tienden a solapar los ángulos de la cabeza asociados a las acciones de distracción con los de no distracción, lo que implica que el sistema puede generar falsos positivos dependiendo cuál sea foco principal (distracción o no distracción).

Los resultados obtenidos del prototipo experimental en entorno real, muestran que el sistema es capaz de identificar las acciones de distracción estudiadas en conductores. Como se observa en la tabla 5.17, el rendimiento en cuanto a exactitud (93.58 %), precisión (90.05 %),

sensibilidad (96.50 %) y puntuación F1 (93.09 %) fue satisfactorio, lo que indica un desempeño robusto para las acciones de distracción analizadas. Los resultados obtenidos para ambos conductores (ver tablas 5.15 y 5.16) indican que el sistema tiene un desempeño relativamente alto en términos de sensibilidad en acciones como "buscar al costado" y "hablar con el pasajero", con una sensibilidad del 100 % en ambos casos. Este resultado sugiere que el sistema es eficaz identificando distracciones que implican un cambio visible en la postura de la cabeza.

Además, se observó una precisión alta en la mayoría de las acciones, especialmente en "arreglarse el cabello" (94.34 %) para el conductor 1 y "texteo izquierda" (93.88 %) para el conductor 2. Sin embargo, a pesar de estos buenos resultados, se observaron limitaciones en cuanto a la detección de no distracción, especialmente cuando los conductores miraron hacia los retrovisores o hicieron un giro pronunciado (izquierda y derecha). Estas situaciones generaron inestabilidades del sistema, indicando que el sistema puede confundirse ante posturas similares a las distracciones estudiadas.

Esta inestabilidad podría estar relacionada con el tiempo de conteo establecido por (Zhao et al., 2020) (0.25 s), que podría ser insuficiente para diferenciar las acciones. Extender este umbral podría mejorar la capacidad del sistema para distinguir entre movimientos seguros y acciones de distracción, aumentando su confiabilidad.

Conclusiones y recomendaciones

Conclusiones

1. Se logró desarrollar e implementar en un sistema embebido un sistema detector de distracción en conductores mediante la estimación de la postura de cabeza en la ciudad del Cusco. Capaz de realizar detecciones en tiempo real, cumpliendo con el objetivo general.

2. La determinación de los ángulos de rotación para estimar la postura de la cabeza mediante una red neuronal convolucional (CNN) fue eficaz, destacando el modelo MNASNet1-0 como el de mejor desempeño. En el Orange Pi 5, este modelo alcanzó un MAE de 3.31 utilizando el conjunto de datos BIWI y de 4.78 con AFLW2000 (ver tablas 5.6 y 5.7). MNASNet1-0 utilizó eficientemente los núcleos de la NPU, con un promedio de uso del 19 % y del 7 % al procesar BIWI y AFLW2000, respectivamente. Además, alcanzó una máxima tasa de 119.66 FPS con BIWI y 48.17 FPS con AFLW2000, manteniendo un consumo promedio de corriente de 0.69 A y 0.46 A en cada caso (ver tablas 5.8 y 5.9). Estos resultados destacan la eficiencia computacional y el bajo consumo energético del modelo MNASNet1-0 en un sistema embebido.

3. Se desarrolló un prototipo del sistema inteligente que utiliza una única cámara y que mediante la postura de la cabeza detecta la distracción en conductores. Se propuso una técnica basada en lógica difusa, la cual fue comparada con las técnicas de umbralización y su modificación. Los resultados demostraron que la técnica de lógica difusa ofrece mejoras, logrando un OT % de 92.05 % para la acción de buscar al costado y 93.49 % para

hablar con el pasajero. Además, al convertir esta técnica en una tabla difusa, se observaron mejoras en el rendimiento computacional manteniendo los valores de OT %, alcanzando 92.08 % para buscar al costado, 93.6 % para hablar con el pasajero, 80.51 % para textear a la derecha y 78.69 % para ajustar la radio (ver Figura 5.10). Estas características hacen que esta técnica sea apropiada para su implementación en sistemas embebidos, como el Orange Pi 5.

4. La implementación del sistema de detección de distracción en el Orange Pi 5 demostró ventajas significativas en portabilidad, eficiencia energética y potencia de cálculo. Al aprovechar su NPU dedicada, el sistema ejecutó modelos de estimación de postura de la cabeza de manera eficiente, con un uso promedio del 19 % de la NPU y un consumo energético de 0.69 A durante tareas intensivas con el modelo MNASNet1-0. Además, alcanzó una tasa máxima de procesamiento de 119.66 FPS y un consumo promedio de solo 3.45 W utilizando el conjunto de datos BIWI. La incorporación de una tabla difusa permitió reducir aún más el costo computacional, haciéndolo ideal para aplicaciones en tiempo real.

5. Las pruebas realizadas en un entorno real con un grupo de conductores voluntarios demostraron que el sistema puede detectar distracción de manera efectiva en tiempo real. El sistema alcanzó una exactitud general del 93.58 %, una precisión del 90.05 %, una sensibilidad del 96.50 % y una puntuación F1 del 93.09 % (ver tabla 5.17, cumpliendo con las expectativas establecidas.

Recomendaciones

1. Dado que el umbral de 0.25 s (8 cuadros) utilizado en esta tesis parece insuficiente para diferenciar de manera consistente entre acciones seguras y de distracción, futuros trabajos podrían investigar diferentes tiempos para el conteo. Ajustar este parámetro permitiría mejorar la precisión en la identificación de distracciones y evitar confusiones con movimientos de conducción normales.

2. Actualmente, el sistema está diseñado para detectar un conjunto específico de acciones de distracción. Futuros trabajos podrían ampliar estas categorías para incluir acciones más complejas y combinaciones de movimientos que se presentan en condiciones de conducción real, lo que haría el sistema más versátil y aplicable a una variedad de comportamientos de distracción.

3. Dadas las limitaciones impuestas por los ángulos de la cámara, se podrían explorar enfoques adicionales que complementen o sustituyan la estimación de la postura de la cabeza, como algoritmos de visión que analicen las posiciones de manos y otros gestos del conductor. Estos enfoques podrían proporcionar una visión más integral del estado de distracción.

4. Futuros estudios también podrían considerar el uso de cámaras o sensores adicionales que capturen el cuerpo completo del conductor y la presencia de objetos en el entorno del vehículo. Esto permitiría un análisis multimodal más detallado, mejorando la detección de distracciones y ayudando a clasificar diferentes niveles de este.

5. Los trabajos futuros podrían enfocarse en optimizar el sistema para sistemas embebidos con recursos más limitados que el Orange Pi 5, como por ejemplo para SBC basados en los chips RK3566/RK3568, centrándose en reducir tanto el tamaño y la latencia del modelo para asegurar su funcionamiento en tiempo real.

Bibliografía

Aguirre, Á. G., & Giraldo, P. J. R. (2014). Sistema embebido de bajo costo para visión artificial. *Scientia et technica*, *19*(2), 163-173.

Amazon Web Services, Inc. (2023). *¿Qué es el reconocimiento facial? - Explicación del software de reconocimiento facial y análisis facial - AWS*. https://aws.amazon.com/es/what-is/facial-recognition/

Arcoverde Neto, E. N., Duarte, R. M., Barreto, R. M., Magalhães, J. P., Bastos, C. C. M., Ren, T. I., & Cavalcanti, G. D. C. (2014). Enhanced real-time head pose estimation system for mobile device. *Integrated Computer-Aided Engineering*, *21*(3), 281-293. https://doi.org/10.3233/ICA-140462

Asperti, A., & Filippini, D. (2023). Deep Learning for Head Pose Estimation: A Survey. *SN Computer Science*, *4*(4), 349. https://doi.org/10.1007/s42979-023-01796-z

Buss, S. R. (2003). *3D computer graphics: a mathematical introduction with OpenGL*. Cambridge University Press.

DEWESoft. (2021). *What is ADAS (Advanced Driver Assistance Systems)*. https://dewesoft.com/blog/what-is-adas

Dlib. (2015). *dlib C++ Library - Introduction*. http://dlib.net/intro.html

Field, A., Field, Z., & Miles, J. (2012). Discovering statistics using R.

Florez Zela, R. D. (2024). Diseño e implementación de un sistema detector de somnolencia en tiempo real mediante visión computacional usando redes neuronales convolucionales aplicado a conductores. *Universidad Nacional de San Antonio Abad del Cusco*. http://hdl.handle.net/20.500.12918/8298

Fundación Carlos Slim. (2016). *Distracciones al conducir – Seguridad Vial*. https://fundacioncarlosslim.org/distracciones-al-conducir-seguridad-vial/

Gonzalez, R. C., & Woods, R. E. (2008). *Digital Image Processing*. Prentice Hall.

González, C. (2015). Lógica difusa una introducción práctica, técnicas de Soft Computing. *Recuperado de: https://www. esi. uclm. es/www/cglez/downloads/docencia/2011_Softcomputing/LogicaDifu pdf*.

Hayley, A. C., Shiferaw, B., Aitken, B., Vinckenbosch, F., Brown, T. L., & Downey, L. A. (2021). Driver monitoring systems (DMS): The future of impaired driving management? *Traffic Injury Prevention*. https://www.tandfonline.com/doi/abs/10.1080/15389588.2021.1899164

Hempel, T., Abdelrahman, A. A., & Al-Hamadi, A. (2022). 6d rotation representation for unconstrained head pose estimation. *2022 IEEE International Conference on Image Processing (ICIP)*, 2496-2500. https://doi.org/10.1109/ICIP46576.2022.9897219

IBM. (2021a). *¿Qué son las redes neuronales convolucionales?* https://www.ibm.com/es-es/topics/convolutional-neural-networks

IBM. (2021b). *¿Qué son las redes neuronales?* https://www.ibm.com/mx-es/topics/neural-networks

Khan, M. Q., & Lee, S. (2019). A Comprehensive Survey of Driving Monitoring and Assistance Systems. *Sensors*, *19*(11), 2574. https://doi.org/10.3390/s19112574

Manrique Gamo, D., & Suárez de Figueroa Baonza, M. d. C. (2017). Razonamiento con imprecisión: lógica borrosa. Apuntes y ejercicios.

MathWorks. (s.f.-a). *¿Qué es una red neuronal convolucional? | 3 cosas que debe saber* [Recuperado 25 de agosto de 2023]. https://es.mathworks.com/discovery/convolutional-neural-network-matlab.html

MathWorks. (s.f.-b). *Foundations of Fuzzy Logic* [Consultado el 2 de setiembre de 2024]. https://la.mathworks.com/help/fuzzy/foundations-of-fuzzy-logic.html

Mecharithm. (2021). *Other Explicit Representation for the Orientation in Robotics: Roll-Pitch-Yaw Angles* [Consultado el 10 de junio del 2024]. https://mecharithm.com/learning/lesson/explicit-representations-orientation-robotics-roll-pitch-yaw-angles-15

Mediapipe. (2023). *Face landmark detection guide*. Google for Developers. https://developers.google.com/mediapipe/solutions/vision/face_landmarker

Michelaraki, E., Katrakazas, C., Kaiser, S., Brijs, T., & Yannis, G. (2023). Real-time monitoring of driver distraction: State-of-the-art and future insights. *Accident Analysis & Prevention*, *192*, 107241. https://doi.org/10.1016/j.aap.2023.107241

Ministerio del Interior de España & Dirección General de Tráfico. (2014). *DGT - Distracciones al conducir*. https://www.dgt.es/muevete-con-seguridad/evita-conductas-de-riesgo/distracciones-al-conducir/

Montoya, A., Holman, D., SF_data_science, Smith, T., & Kan, W. (2016). *State Farm Distracted Driver Detection*. https://kaggle.com/competitions/state-farm-distracted-driver-detection

Olabe, X. B. (1998). *Redes neuronales artificiales y sus aplicaciones*. Publicaciones de la Escuela de Ingenieros.

OMS. (2022). Road Traffic Injuries. https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries

OrangePi. (2022). *Orange Pi 5*. http://www.orangepi.org/html/hardWare/computerAndMicrocontrollers/details/Orange-Pi-5.html

Ortega, J. D., Kose, N., Cañas, P., Chao, M.-A., Unnervik, A., Nieto, M., Otaegui, O., & Salgado, L. (2020). DMD: A Large-Scale Multi-modal Driver Monitoring Dataset for Attention and Alertness Analysis. En A. Bartoli & A. Fusiello (Eds.), *Computer Vision – ECCV 2020 Workshops* (pp. 387-405). Springer International Publishing. https://doi.org/10.1007/978-3-030-66823-5_23

Palomares, F. G., Monsoriu, J. A., & Alemany, E. (2016). Aplicación de la convolución de matrices al filtrado de imágenes. *Modelling in Science Education and Learning*, *9*(1), 97-108.

Papantoniou, P., Papadimitriou, E., & Yannis, G. (2017). Review of driving performance parameters critical for distracted driving research. *Transportation Research Procedia*, *25*, 1796-1805. https://doi.org/10.1016/j.trpro.2017.05.148

PNP. (2021). Anuario Estadístico Policial - Policía Nacional del Perú. https://web.policia.gob.pe/anuario_estadistico/documentos/ANUARIO%20PNP%202021.pdf

PyTorch. (2024). *Optimizing Model Parameters* [Consultado el 16 de mayo del 2024]. https://pytorch.org/tutorials/beginner/basics/optimization_tutorial.html

Radxa. (2024). *RK3588 vs RK3588S* [Consultado el 11 de abril, 2024]. https://wiki.radxa.com/Rock5/RK3588_vs_RK3588S

RAE. (2023). *distracción | Definición | Diccionario de la lengua española | RAE - ASALE*. https://dle.rae.es/distracci%C3%B3n

Rockchip. (2024). *RK3588/RK3588S* [Consultado el 11 de abril, 2024]. https://www.rockchips.com/a/en/

Sabry, F., & Costa, G. (2024). *Interpolación bilineal: Mejora de la resolución y claridad de la imagen mediante interpolación bilineal*. Mil Millones De Conocimientos [Spanish]. https://books.google.com.pe/books?id=EDYGEQAAQBAJ

Samsung. (2019). *The Neural Processing Unit (NPU): A Brainy Next-Generation Semiconductor* [Consultado el 10 de marzo de 2024]. https://semiconductor.samsung.com/support/tools-resources/dictionary/the-neural-processing-unit-npu-a-brainy-next-generation-semiconductor/

Shetty, A. B., Bhoomika, Deeksha, Rebeiro, J., & Ramyashree. (2021). Facial recognition using Haar cascade and LBP classifiers. *Global Transitions Proceedings*, *2*(2), 330-335. https://doi.org/10.1016/j.gltp.2021.08.044

Soto, E. (2021). *Glosario Ilustrado de Matemáticas Escolares* [Consultado el 24 de mayo del 2024]. https://books.google.es/books?id=_iRdEAAAQBAJ&pg=PA276#v=onepage&q&f=false

Spong, M. W., & Vidyasagar, M. (2008). *Robot dynamics and control*. John Wiley & Sons.

Sri Mounika, T. V. N. S. R., Phanindra, P. H., Sai Charan, N. V. V. N., Kranthi Kumar Reddy, Y., & Govindu, S. (2022). Driver Drowsiness Detection Using Eye Aspect Ratio (EAR), Mouth Aspect Ratio (MAR), and Driver Distraction Using Head Pose Estimation. En M. Tuba, S. Akashe & A. Joshi (Eds.), *ICT Systems and Sustainability* (pp. 619-627). Springer Nature Singapore.

TechRadar. (2024). *What is an NPU?* [Consultado el 10 de marzo de 2024]. https://www.techradar.com/computing/cpu/what-is-an-npu

TensorFlow. (2022). *Transferencia de aprendizaje y ajuste* [Consultado el 16 de mayo del 2024]. https://www.tensorflow.org/tutorials/images/transfer_learning?hl=es-419

Terven, J., Cordova-Esparza, D., Ramirez-Pedraza, A., & Chavez-Urbiola, E. (2023). Loss Functions and Metrics in Deep Learning. *arXiv preprint arXiv:2307.02694*. https://doi.org/https://doi.org/10.48550/arXiv.2307.02694

Torchvision. (2023). *Models and pre-trained weights — Torchvision 0.15 documentation*. https://pytorch.org/vision/stable/models.html

Udacity. (2019). *L5 040 Color Images Part 1 V2*. https://www.youtube.com/watch?v=pJNusvS8pho

UNAM. (2021). *Acervo para el mejoramiento del aprendizaje de alumnos de ingeniería, en inteligencia artificial: Lógica difusa* [Consultado el 30 de agosto del 2024]. https://virtual.cuautitlan.unam.mx/intar/?page_id=997

Vince, J. (2011). *Rotation transforms for computer graphics*. Springer Science & Business Media.

Yang, T.-Y. (2019). *FSA-Net* [Consultado el 24 de setiembre de 2024]. https://github.com/shamangary/FSA-Net/blob/master/data/TYY_create_db_biwi.py

Zhang, K., Zhang, Z., Li, Z., & Qiao, Y. (2016). Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks. *IEEE Signal Processing Letters*, *23*(10), 1499-1503. https://doi.org/10.1109/LSP.2016.2603342

Zhao, Z., Xia, S., Xu, X., Zhang, L., Yan, H., Xu, Y., & Zhang, Z. (2020). Driver distraction detection method based on continuous head pose estimation. *Computational Intelligence and Neuroscience*, *2020*, 1-10.

Anexos

A. Costos y presupuestos

Tabla 1
Presupuesto horas-hombre aproximado.

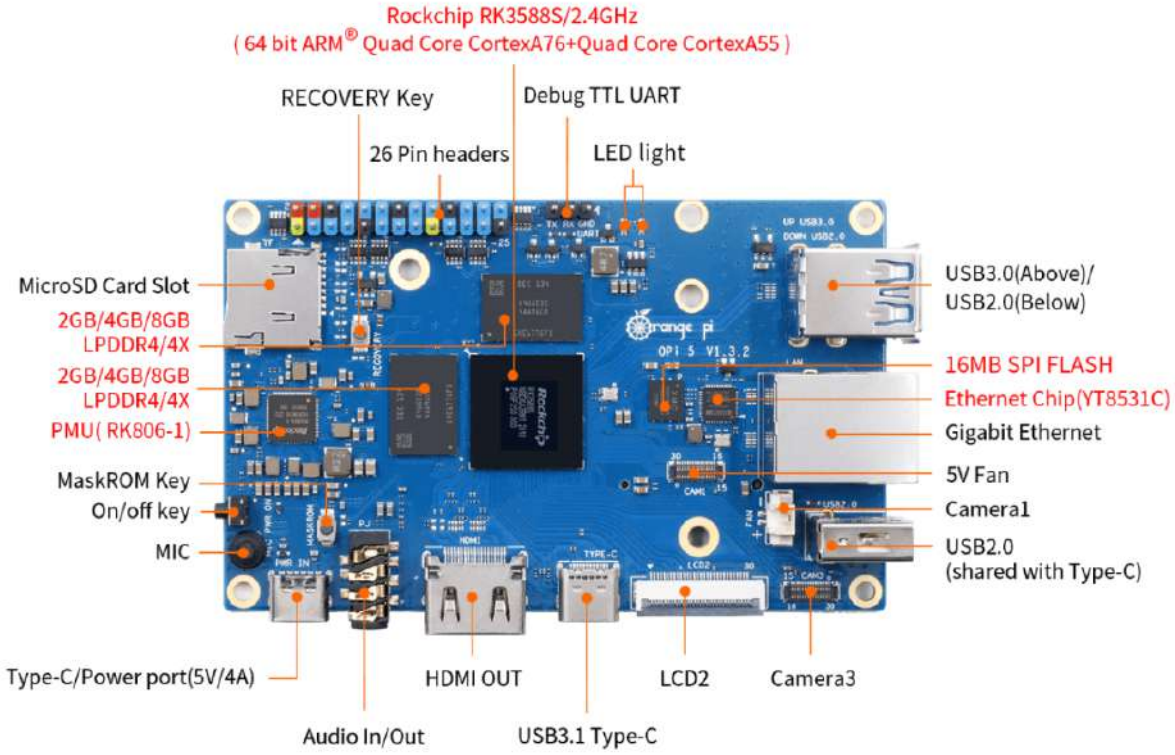
Tarea	Horas estimadas	Tarifa por hora (S/.)	Monto (S/.)
Revisión bibliográfica	120	8	960.00
Obtención de los datos	80	8	640.00
Procesamiento de datos	160	8	1280.00
Entrenamiento	160	8	1280.00
Implementación	120	8	960.00
Experimentos	160	8	1280.00
Redacción de tesis	400	8	3200.00
Total			**9600.00**

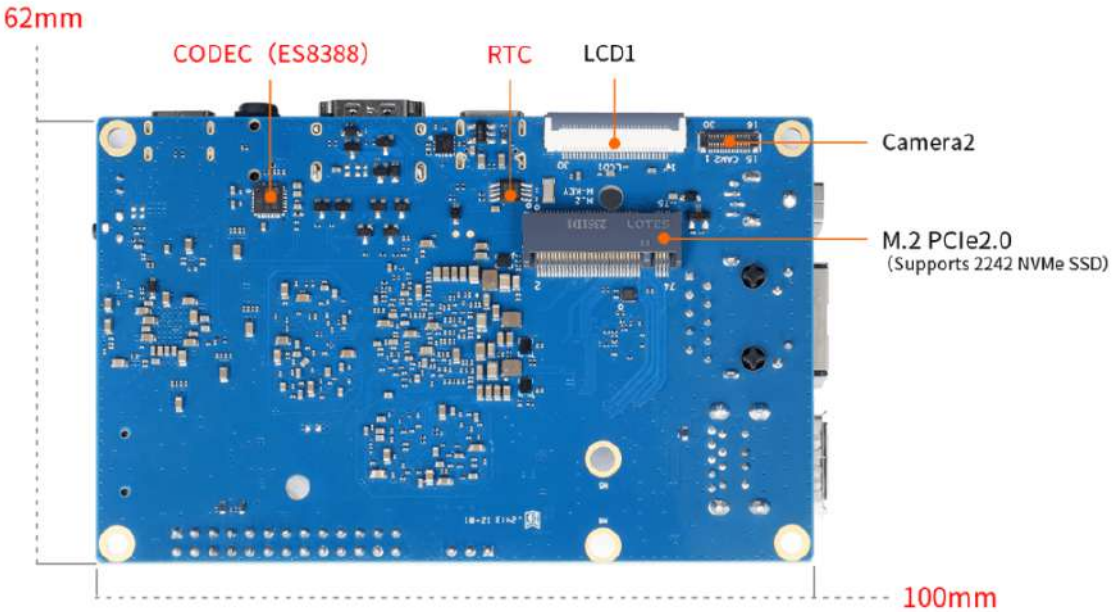
Tabla 2

Presupuesto de recursos para la elaboración de la tesis.

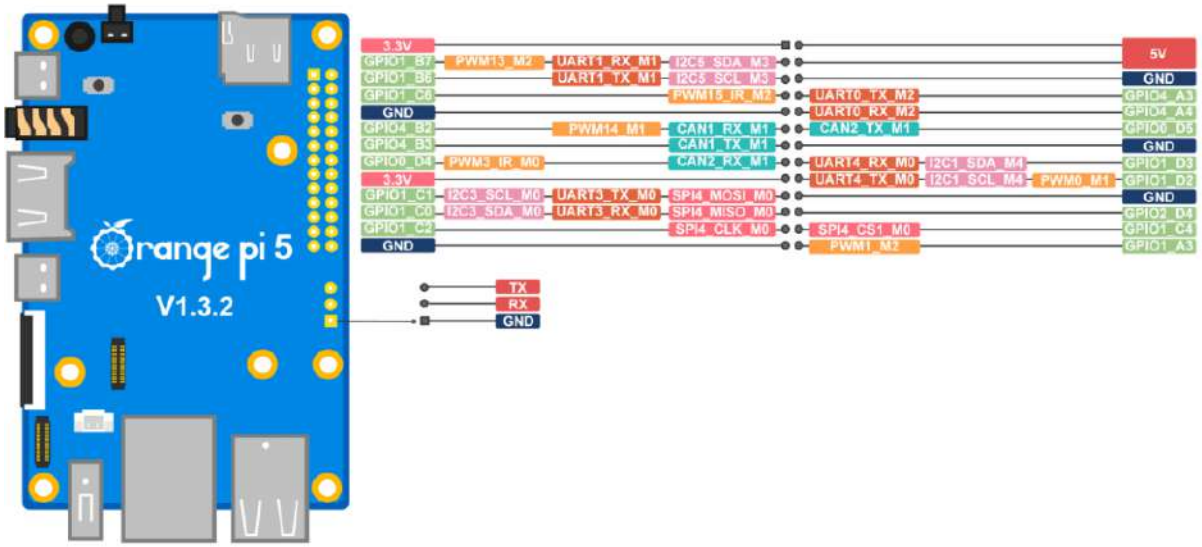
Categoría	Descripción	Fuente financiadora	Monto (S/.)
Equipo	Estación de trabajo HP, Intel Xeon W-2123 + Nvidia Quadro P2000, monitor, teclado y ratón.	LIECAR	0.00
Equipo	Orange Pi 5 8GB + cargador 5V/4A	Personal	418.80
Equipo	SSD NVME M.2 256 GB	Personal	70.00
Equipo	Carcasa de aluminio OP5 + ventilador	Personal	46.70
Equipo	Cámara	LIECAR	0.00
Equipo	Parlantes 3W	Personal	30.00
Equipo	Convertidor XL4015	Personal	12.00
Componentes	Leds, BC548, resistencias, entre otros.	Personal	20.00
Útiles de escritorio	Papelería, lapiceros, entre otros.	Personal	50.00
Software	Sistema operativo: Manjaro Linux y Ubuntu ARM	Licencia libre	0.00
Software	Python, Pytorch, RKNN, RKNN-toolkit2, Mediapipe, entre otros.	Licencia libre	0.00
Datos	300W-LP, BIWI, AFLW2000	Licencia libre	0.00
Datos	Driver monitoring dataset	Licencia restringida	0.00
Servicios	Internet por 6 meses	LIECAR	0.00
Servicios	Energía eléctrica por 6 meses	LIECAR	0.00
Servicios	Transporte	Personal	240.00
Total			**887.50**

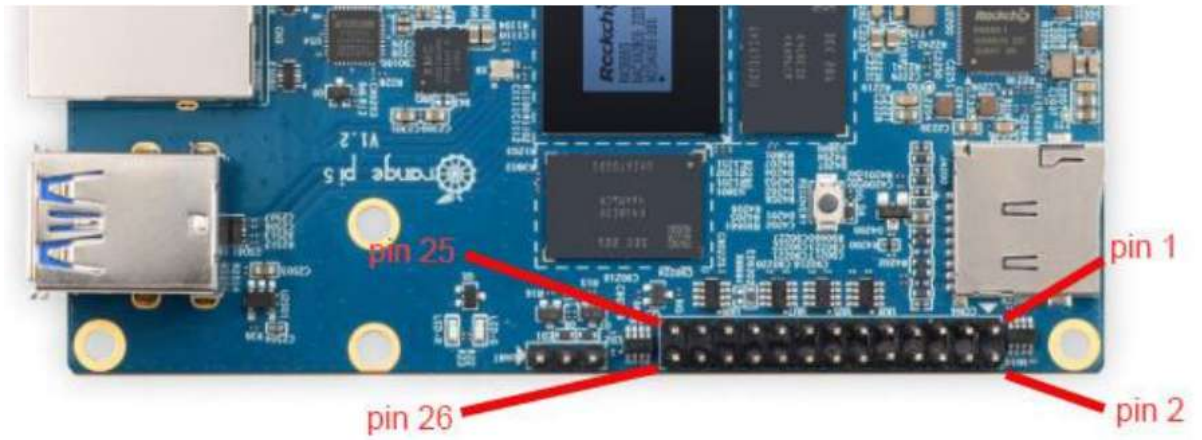
B. Hoja de datos Orange Pi 5





Las figuras siguientes muestran las funciones de los 26 pines del Orange Pi 5, teniendo un total de 17 GPIO que trabajan con 3.3 V.






```
→ ~ sudo gpio readall
[sudo] contraseña para gong:
 +------+-----+----------+--------+---+   OPI5   +---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---++----+----++---+--------+----------+-----+------+
 |      |     |     3.3V |        |   |  1 || 2  |   |        | 5V       |     |      |
 |   47 |   0 |    SDA.5 |     IN | 1 |  3 || 4  |   |        | 5V       |     |      |
 |   46 |   1 |    SCL.5 |     IN | 1 |  5 || 6  |   |        | GND      |     |      |
 |   54 |   2 |    PWM15 |     IN | 1 |  7 || 8  | 0 | IN     | RXD.0    | 3   | 131  |
 |      |     |      GND |        |   |  9 || 10 | 0 | IN     | TXD.0    | 4   | 132  |
 |  138 |   5 |  CAN1_RX |     IN | 1 | 11 || 12 | 1 | IN     | CAN2_TX  | 6   | 29   |
 |  139 |   7 |  CAN1_TX |     IN | 1 | 13 || 14 |   |        | GND      |     |      |
 |   28 |   8 |  CAN2_RX |     IN | 1 | 15 || 16 | 1 | IN     | SDA.1    | 9   | 59   |
 |      |     |     3.3V |        |   | 17 || 18 | 1 | IN     | SCL.1    | 10  | 58   |
 |   49 |  11 | SPI4_TXD |     IN | 1 | 19 || 20 |   |        | GND      |     |      |
 |   48 |  12 | SPI4_RXD |     IN | 1 | 21 || 22 | 1 | IN     | GPIO2_D4 | 13  | 92   |
 |   50 |  14 | SPI4_CLK |     IN | 0 | 23 || 24 | 0 | IN     | SPI4_CS1 | 15  | 52   |
 |      |     |      GND |        |   | 25 || 26 | 1 | IN     | PWM1     | 16  | 35   |
 +------+-----+----------+--------+---++----+----++---+--------+----------+-----+------+
 | GPIO | wPi |   Name   |  Mode  | V | Physical | V |  Mode  | Name     | wPi | GPIO |
 +------+-----+----------+--------+---+   OPI5   +---+--------+----------+-----+------+
→ ~
```

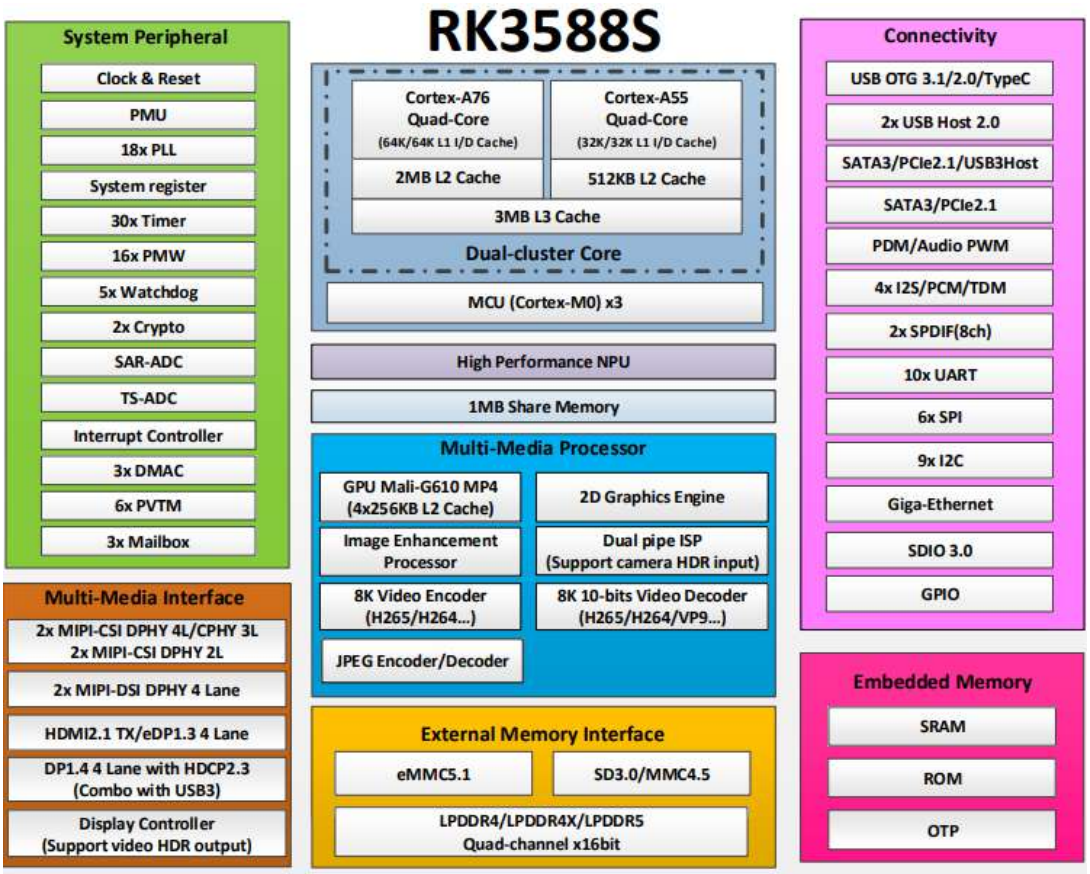
1. 3. Hardware features of Orange Pi 5

Introduction to hardware features	
CPU	• Rockchip RK3588S (8nm LP processor) • 8-core 64-bit processor • 4-core Cortex-A76 and 4-core Cortex-A55 core architecture • The main frequency of the large core is up to 2.4GHz, and the main frequency of the small core is up to 1.8GHz
GPU	•Integrated ARM Mali-G610 • OpenGL ES1.1/2.0/3.2, OpenCL 2.2 and Vulkan 1.2
NPU	• Built-in AI accelerator NPU with a computing power of up to 6 Tops • Support INT4/INT8/INT16 mixed operation
Video Output	• HDMI 2.1, up to 8K @60Hz • DP1.4（DisplayPort） • 2*MIPI D-PHY TX 4Lan
Memorry	4GB/8GB/16GB（LPDDR4/4x）
Camera	• 1 * MIPI CSI 4Lane • 2 * MIPI D-PHY RX 4Lane
PMU	RK806-1
Onboard Storage	• 16MB QSPI Nor FLASH • MicroSD (TF) Card slot • PCIe2.0x1 M.2 M-KEY (SSD) slot
Ethernet	10/100/1000Mbps ethernet (YT8531C)
Audio	• 3.5mm headphone jack audio in/out • Onboard MIC input • HDMI output
PCIe M.2 M-KEY	• Support PCIe WIFI6+BT5.0+BLE • Support SSD
USB Interface	1 * USB3.0 Interface 2 * USB2.0 Interface （ One of them is shared with the

	Type-C interface） 1 * USB3.0 Type-C Interface
26pin Extension Header	Used to expand UART, PWM, I2C, SPI, CAN and GPIO interfaces
Debug Serial Port	3pin debug serial port
LED Light	Power light and status light
Button	1 * Mask ROM key, 1 * RECOVERY, 1 * switch key
Power Supply	5V/4A Type-C power supply
Supported OS	Orange Pi OS（Droid）、Orange Pi OS（Arch）、Android12.1、Debian11、Ubuntu20.04 and Ubuntu22.04 operating systems
Introduction of Appearance Specifications	
Product Size	100mm*62mm
Weight	46g
range Pi™ is a registered trademark of Shenzhen Xunlong Software Co., Ltd	

1.2.6 Neural Process Unit

- Neural network acceleration engine with processing performance up to 6 TOPS
- Include triple NPU core, and support triple core co-work, dual core co-work, and work independently
- Support integer 4, integer 8, integer 16, float 16, Bfloat 16 and tf32 operation
- Embedded 384KBx3 internal buffer
- Multi-task, multi-scenario in parallel
- Support deep learning frameworks: TensorFlow, Caffe, Tflite, Pytorch, Onnx NN, Android NN, etc.
- One isolated voltage domain to support DVFS



- USB3.1 Gen1
  - Support USB3.1 Gen1,equal to USB3.2 Gen1 and USB3.0,up to 5Gbps datarate
  - Embedded 1 USB3.1 OTG interfaces which combo with DP TX (USB3OTG_0)
  - Embedded 1 USB3.1 Host interface which combo with Combo PIPE PHY2 (USB3OTG_2)
  - Compatible Specification
    - Universal Serial Bus 3.0 Specification, Revision 1.0
    - Universal Serial Bus Specification, Revision 2.0 (exclude USB3OTG_2)
    - eXtensible Host Controller Interface for Universal Serial Bus (xHCI), Revision 1.1
  - Support Control/Bulk (including stream)/Interrupt/Isochronous Transfer
  - Simultaneous IN and OUT transfer for USB3.1 Gen1
  - Descriptor caching and data pre-fetching used to improve system performance in high-latency systems
  - LPM protocol in USB 2.0 (exclude USB3OTG_2) and U0, U1, U2, and U3 states for USB3.1 Gen1
  - USB3.1 Gen1 Device Features
    - Up to 10 IN endpoints, including control endpoint 0
    - Up to 6 OUT endpoints, including control endpoint 0
    - Up to 16 endpoint transfer resources, each one for each endpoint
    - Flexible endpoint configuration for multiple applications/USB set-configuration modes
    - Hardware handles ERDY and burst
    - Stream-based bulk endpoints with controller automatically initiating data movement
    - Isochronous endpoints with isochronous data in data buffers
    - Flexible Descriptor with rich set of features to support buffer interrupt moderation, multiple transfers, isochronous, control, and scattered buffering support

- USB3.1 Gen1 xHCI Host Features
  - Support up to 64 devices
  - Support 1 interrupter
  - Support 1 USB2.0 port (exclude USB3OTG_2) and 1 Super-Speed port
  - Support standard or open-source xHCI and class driver
- USB3.1 Gen1 Dual-Role Device (DRD) Features
  - Static Device Operation
  - Static Host Operation
  - USB3.1/USB2.0 OTG A device and B device basing on ID, USB3OTG_2 only support USB3.1 Gen1
  - Not Support USB3.1/USB2.0 OTG session request protocol (SRP), host negotiation protocol (HNP) and Role Swap Protocol (RSP)
- Miscellaneous Features
  - USB2.0 PHY support Battery Charge detection
  - USB3OTG_0 support USB Type-C and DP Alt Mode
  - USB3OTG_2 PHY combos with PCIE and SATA

- USB 2.0 Host
  - Compatible with USB 2.0 specification
  - Support two USB 2.0 Host
  - Supports high-speed(480Mbps), full-speed(12Mbps) and low-speed(1.5Mbps) mode
  - Support Enhanced Host Controller Interface Specification (EHCI), Revision 1.0
  - Support Open Host Controller Interface Specification (OHCI), Revision 1.0a

C. Descripción de la cámara

• **Cámara de alta resolución:**Capture cada detalle con la cámara de alta resolución de la cámara web Mairuige 4K, que ofrece una resolución máxima de 3840x2160 para una calidad de video increíblemente clara.

• **Función de enfoque automático:**Nunca te preocupes por los videos borrosos de nuevo, ya que la función de enfoque automático de la cámara web garantiza que todo tu contenido esté enfocado, mejorando la experiencia visual general.

• **Micrófono integrado:**La cámara web viene con un micrófono integrado, proporcionando audio de alta calidad para complementar sus videos, por lo que es ideal para videoconferencias o clases en línea.

• **Interfaz USB:**Fácil conectividad con la interfaz USB de la cámara web, lo que garantiza una integración perfecta con su PC o portátil para un uso sin complicaciones.

• **Compacto y portátil:**Su diseño compacto y portátil hace que sea fácil de transportar, perfecto para aquellos que siempre están en el camino.

• **Producto certificado:**Garantía de calidad con la webcam certificada por la FCC, garantizando su seguridad y fiabilidad de uso.

Velocidad de fotogramas de vídeo: 1080P:30fps,2K 4K:30fps
Formato de salida: MJPG/YUY2
Micrófono: Micrófono digital
Interfaz: USB2.0
Longitud del cable: 140cm
Sistema de soporte: Windows XP, Win7,Win8,Win10M,Mac OS ,Vista, Linux

1080P		2K		4K	
Sensor pixel	2 Mega	Sensor pixel	3 Mega	Sensor pixel	8 Mega
Resolution	1920x1080/30fps	Resolution	2048x1536/30fps	Resolution	3840x2160/30fps
Autofocus	No	Autofocus	Yes	Autofocus	Yes
Microphone	Yes	Microphone	Yes	Microphone	Yes
Format	MJPEG, YUY2	Format	MJPEG, YUY2	Format	MJPEG, YUY2
Interface	USB 2.0	Interface	USB 2.0	Interface	USB 2.0

D. Instalación de sistema operativo en el Orange Pi 5

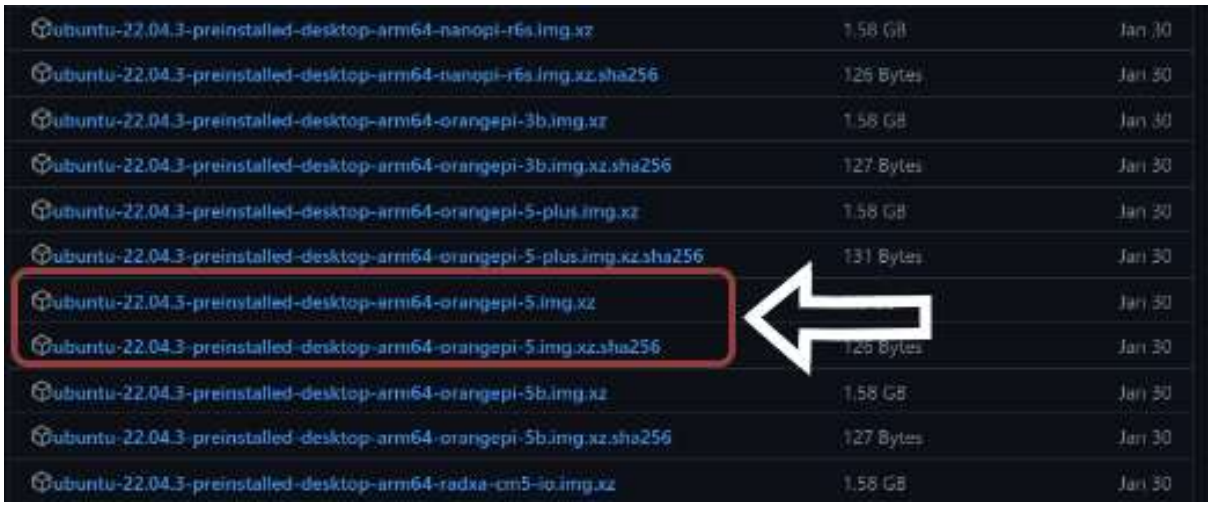
Para este caso se uso una versión de la comunidad de código abierto de Ubuntu Arm en su version 22.04 LTS 1.33 descargable en (https://github.com/Joshua-Riek/ubuntu-rockchip/releases), la cual cuenta con el kernel 5.15 BSP y los controladores de código abierto mesa-panfrost para el GPU Mali-G610.

La instalación cuenta con una serie de pasos que son válidos para una distribución de Linux (Se usó Manjaro Linux).

Descarga de la imagen

Para esto basta con dirigirse a dirección de descarga, seleccionar la imagen .img.xz desarrollada para el Orange Pi 5 juntamente con su archivo de comprobación, archivos que se pueden observar en la figura D.1.

Figura D.1
Archivos de Ubuntu ARM



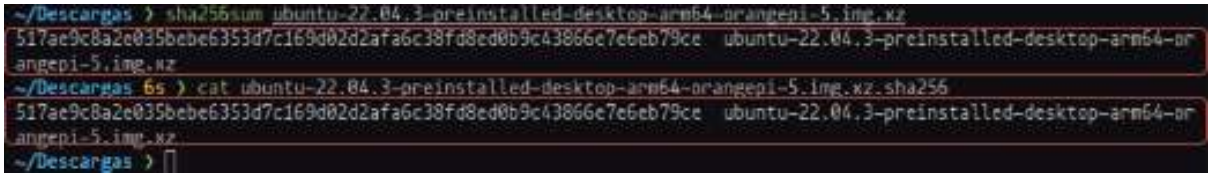
Una vez descargados ambos archivos, es momento de realizar una comprobación del estado de la imagen, haciendo uso de sha256.

Para esto primero se abre una terminal en Linux, y se debe ir hacia la dirección que contiene dichos archivos y ejecutar los siguientes comandos.

```
sha256sum nombre_de_la_imagen.img.xz
cat nombre_de_la_imagen.img.xz.sha256
```

Al ejecutar los comandos, se debe de verificar que ambos "hashçoincidan (ver figura D.2), si es así el archivo puede ser grabado en una microSD.

Figura D.2
Verificación de integridad de archivos.



Grabado de la imagen e instalación

Para grabar la imagen en un dispositivo de almacenamiento (microSD, SSD, USB, NVME, etc.), se emplea BalenaEtcher, una aplicación que es fácil de usar y multiplataforma. El proceso para grabar una imagen con BalenaEtcher es el siguiente:

- Descargar e instalar BalenaEtcher, está disponible en (https://etcher.balena.io/#download-etcher).
- Abrir la aplicación, que tiene una interfaz sencilla con tres pasos principales (ver figura D.3).
- Seleccionar la imagen haciendo clic sobre "Flash from filez se elige la imagen a grabar (ver figura D.4).
- Elegir la microSD. Antes conectar la microSD a través de un lector de memorias. En BalenaEtcher se selecciona el dispositivo haciendo clic en "Select target"(ver figura D.5).
- Para iniciar el proceso de grabado, basta con hacer clic en "Flash"(ver figura D.6), puede que se te pida la clave de superusuario para iniciarlo. Este proceso borrará toda la información que se encuentra en la microSD y grabará en su interior el sistema operativo.

Figura D.3
Interfaz BalenaEtcher.

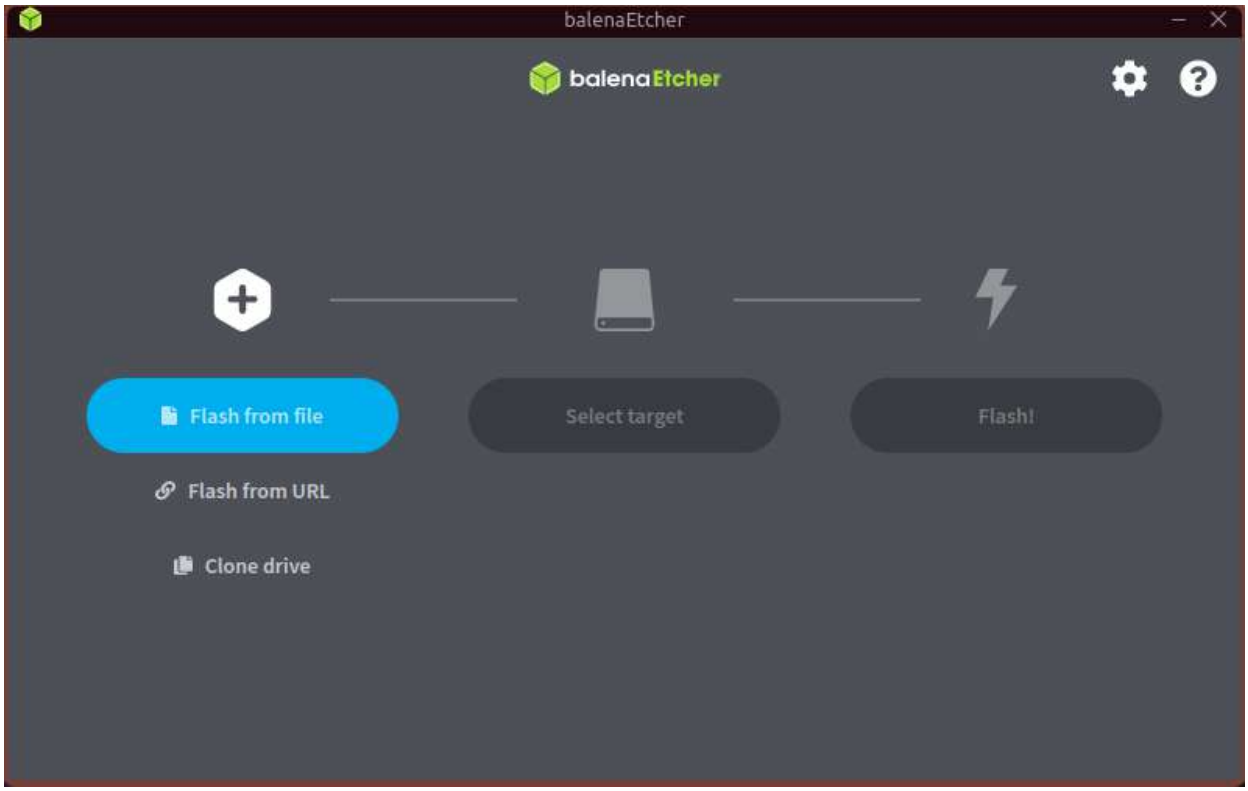


Figura D.4
Selección de la imagen.

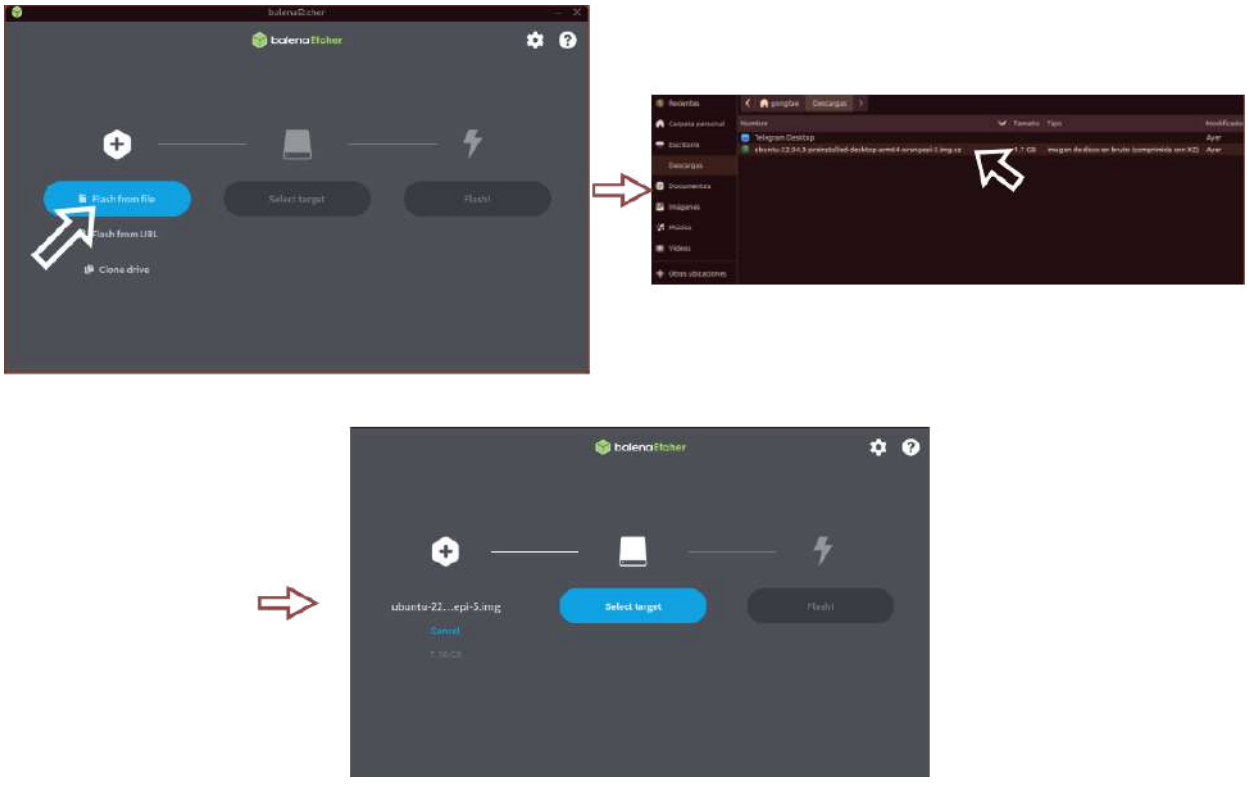


Figura D.5
Selección de la microSD.

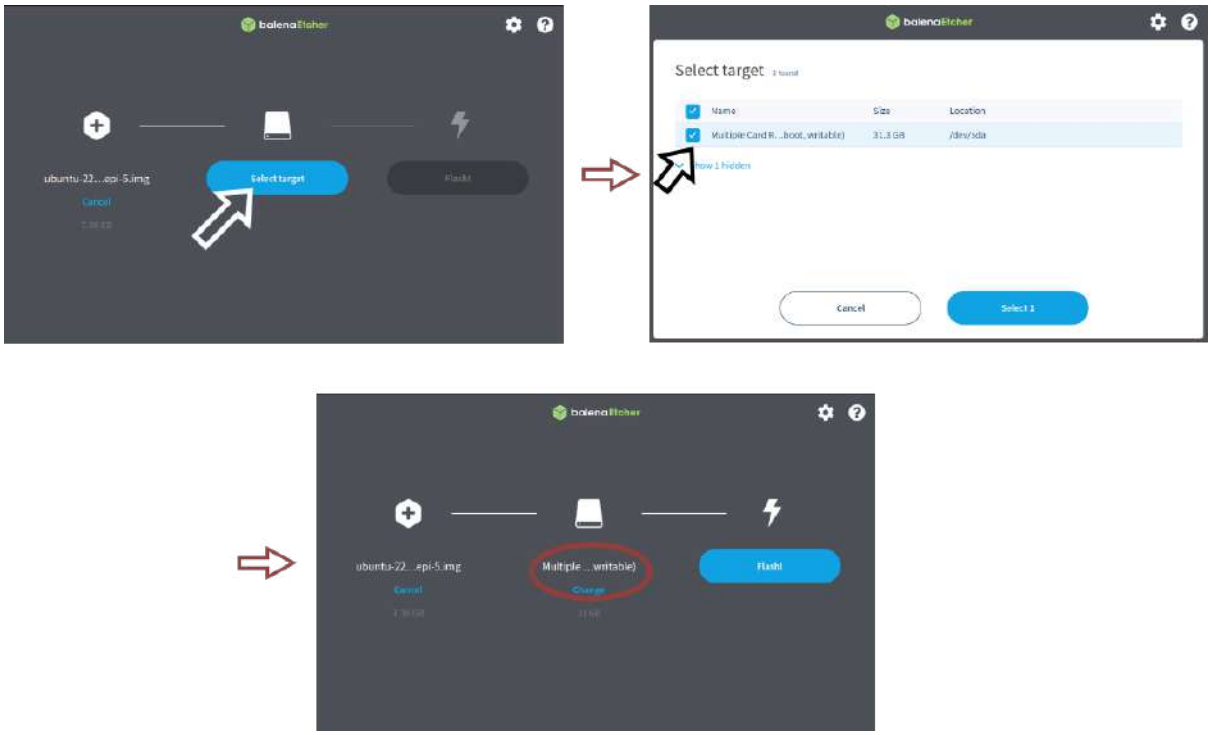
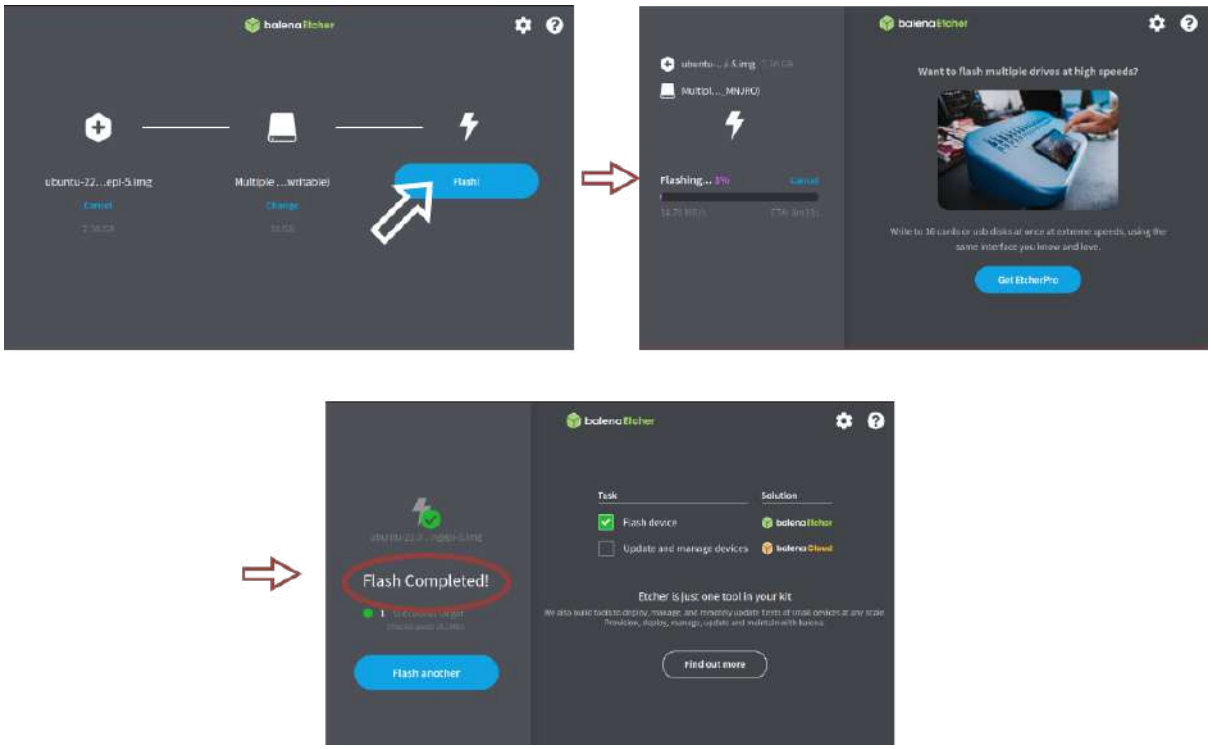


Figura D.6
Selección de la microSD.



- Una vez finalizado, se puede retirar el dispositivo y usarlo en el Orange Pi 5.

Una vez finalizado los pasos anteriores, se inserta la microSD y se arranca el Orange Pi 5 para completar la instalación del sistema operativo, los cuales se detallan a continuación.

Configuración Inicial: Una vez que el sistema haya arrancado, se presenta un asistente de configuración inicial. Se sigue las instrucciones para configurar elementos como el idioma, la zona horaria, el teclado y las credenciales del usuario, con el que se termina la instalación.

Actualización del Sistema: Después de completar la configuración inicial, es recomendable actualizar el sistema operativo para asegurarte de tener las últimas correcciones y parches de seguridad. Este proceso se realiza abriendo un terminal y ejecutando:

```
1 sudo apt update
2 sudo apt upgrade
```

E. Driver monitoring dataset (DMD)

Es un conjunto de datos que contiene eventos como la distracción, somnolencia, manos y mirada de 37 participantes en edades entre 22 a 47 en hombres y de 21 a 38 en mujeres, representando el 73 % y el 27 % respectivamente. Fue desarrollado por Vicomtech con colaboración de Intel bajo una licencia "Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License", la cual especifica que el conjunto de datos no se puede usar con fines comerciales y si este es modificado no debe ser distribuido.

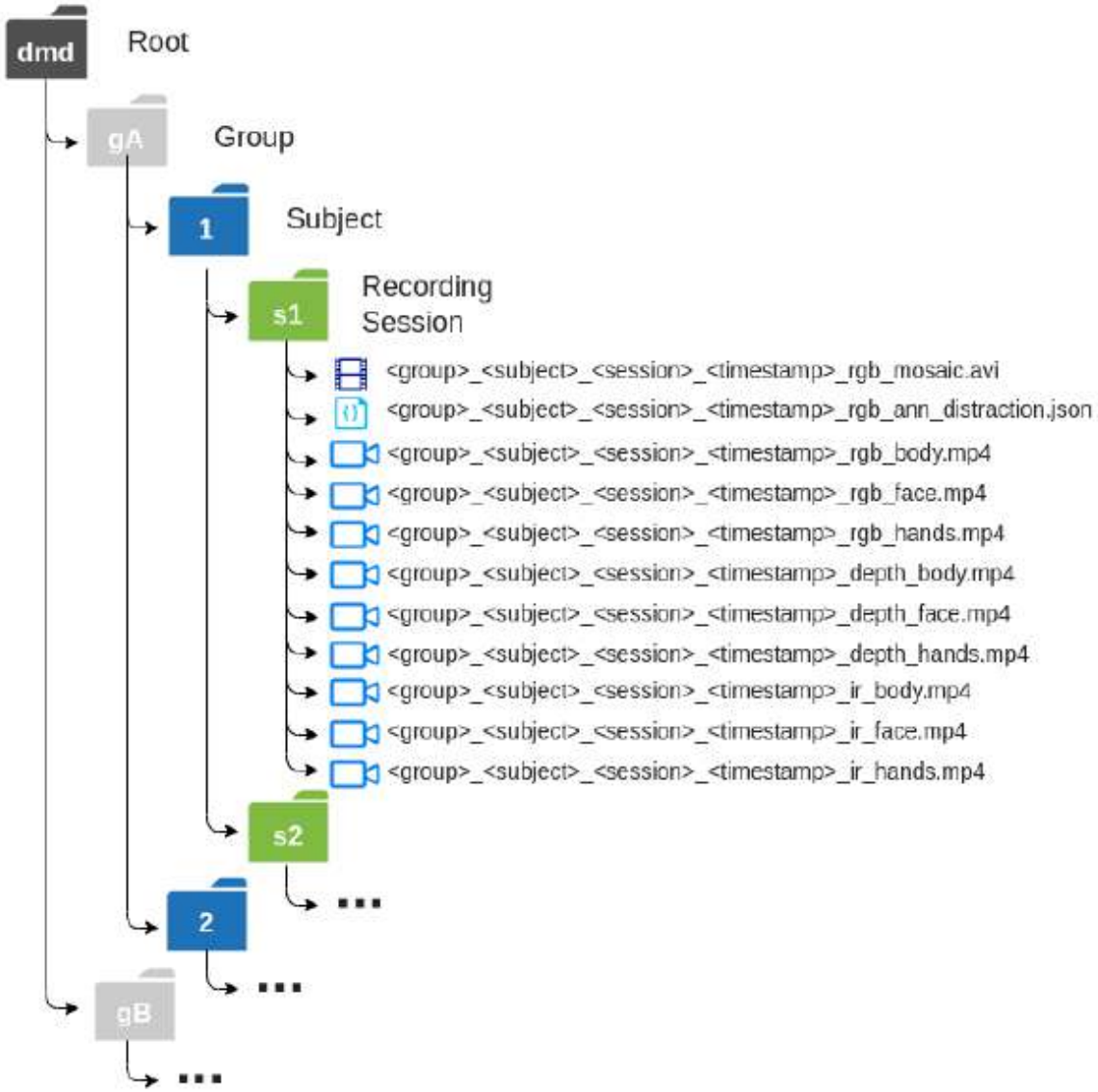
Para tener acceso a los datos, se tiene que realizar un registro en su página web https://dmd.vicomtech.org/, donde además se tiene que llenar un espacio en la que se tiene que indicar los motivos y donde se pretende usar dicho conjunto de datos.

Una vez con acceso a los datos, este cuenta con un tamaño total de 1.6 TB, con información de 3 cámaras en RGB, IR y de profundidad cada una. La organización de estos datos se muestra en la figura E.1, donde el material está dividido en grupos (gA, gB, etc.) con 5 carpetas de los participantes cada uno, cada carpeta de participantes contiene carpetas de las sesiones que se le hicieron (s1, s2, etc.), que contienen a su vez los datos de video en formato".mp4" y los metadatos en ".json".

De las sesiones, las que están enfocadas en distracción son de la 1 hasta la 4, las cuales están relacionadas con las siguientes actividades:

- s1, en entorno de conducción real: conducción segura, búsqueda al lado, manejar la radio, beber, hablar con el pasajero.
- s2, entorno real con vehículo detenido: conducción segura, búsqueda al lado, peinarse y maquillarse, hablar por el teléfono - derecha, hablar por el teléfono - izquierda, mensajear - derecha, mensajear - izquierda.
- s3, en vehículo detenido: buscar algo del asiento de atrás.
- s4, en simulador: las mismas actividades de s2.

Figura E.1
Organización del conjunto de datos DMD.



E.1. Anotaciones

El criterio que tomaron para realizar las anotaciones respecto a la distracción se encuentra disponible en https://github.com/Vicomtech/DMD-Driver-Monitoring-Dataset/wiki/DMD-distraction-related- donde se describe las acciones temporales definidos en 7 niveles, tendiendo que cada nivel tiene su propio conjunto de etiquetas, como se muestra en la tabla 3

De la tabla 3, los niveles que no necesitan tener anotaciones en todos los cuadros son:

Tabla 3
Niveles de anotaciones del conjunto DMD para distracción.

Id	Annotation levels for distraction:	# of Labels	Continuous
0	Occlusion in cameras	3	No
1	Gaze on Road	2	Yes
2	Driver is Talking	1	No
3	Hands using Wheel	4	Yes
4	Hand on gear	1	No
5	Objects in scene	3	No
6	Driver Actions	14	Yes

"Occlusion in cameras" (oclusión en las cámaras), "driver is talking" (el conductor está hablando), "hand on gear" (cambiar de marcha) y "objects in the scene" (objetos en la escena), mientras los niveles que si deben tener anotaciones en todos los cuadros son: "gaze on road" (mirada en la via), "hands using wheel" (manos en el volante) y "driver actions" (acciones del conductor).

E.2. Etiquetas de los niveles

Nivel 0: Occlusion in cameras (oclusión en las cámaras)

Se determina que si un objeto tapa el 50 a 60 % de la vista de alguna de las cámaras y la acción que realiza el conductor no se puede reconocer, entonces se trata de una oclusión. Para realizar las anotaciones en este nivel se toma en cuenta todas las cámaras.

Las etiquetas de este nivel son las siguientes:

- Face occlusion (oclusión de la cara), la cara del conductor se encuentra ocluida y no es posible reconocer la acción que realiza.
- Body occlusion (oclusión del cuerpo), el cuerpo de conductor se encuentra ocluido y no es posible reconocer la acción que realiza.

- Hands occlusion (oclusión de las manos), la cámara que graba las manos se encuentra ocluida.

Nivel 1: Gaze on road (mirada en la carretera)

En este nivel se determina si el conductor está poniendo toda su atención visual en la conducción. Esto se logra identificando si el conductor está mirando la carretera o zonas relacionadas con la conducción (espejos retrovisores, ventanas izquierda/derecha para verificar otros autos) o si no lo está haciendo (teléfono móvil, radio, regazo, volante, detrás).

Para este nivel se usa principalmente la cámara que apunta hacia la cabeza y como complemento se usa la cámara del cuerpo.

Las etiquetas son las siguientes:

- Looking at the road (mirando la carretera), cuando el conductor esta mirando a la carretera, espejos retrovisores, ventanas de la izquierda/derecha.
- Not looking at the road (No se esta mirando la carreta), cuando el conductor no está mirando nada relacionado con la carretera.

La consideración que se toma en las anotaciones de este nivel es que si se produce pestañeos pequeños deben de considerarse como "looking at the road".

Nivel 2: Hands using wheel (manos usando el volante)

En este nivel se realiza las anotaciones relacionadas con las manos de los conductores en la tarea de conducir. Se toma en cuenta principalmente la cámara de las manos.

Las etiquetas de este nivel son las siguientes:

- Both (ambas manos), ambas manos sostienen el volante.

- Only right (solo derecha), solo la mano derecha sostiene el volante.
- Only left (solo izquierda), solo la mano izquierda sostiene el volante.
- None (ninguno), no se esta sosteniendo el volante.

Nivel 4: Hand on gear (mano en la marcha)

En este nivel se hace anotaciones cuando el conductor tiene la mano en la marcha o palanca de cambios. Se usa principalmente la cámara de las manos aunque también se usa la cámara que graba el cuerpo.

Las etiquetas de este nivel son las siguientes:

- Hand on gear (mano en la marcha), si el conductor tiene su mano en la palanca de cambios.

Nivel 5: Objects in the scene (objetos en la escena)

En este nivel, se realizan anotaciones cuando un objeto es visible. Se utiliza la información capturada por las 3 cámaras.

Las etiquetas de este nivel son las siguientes:

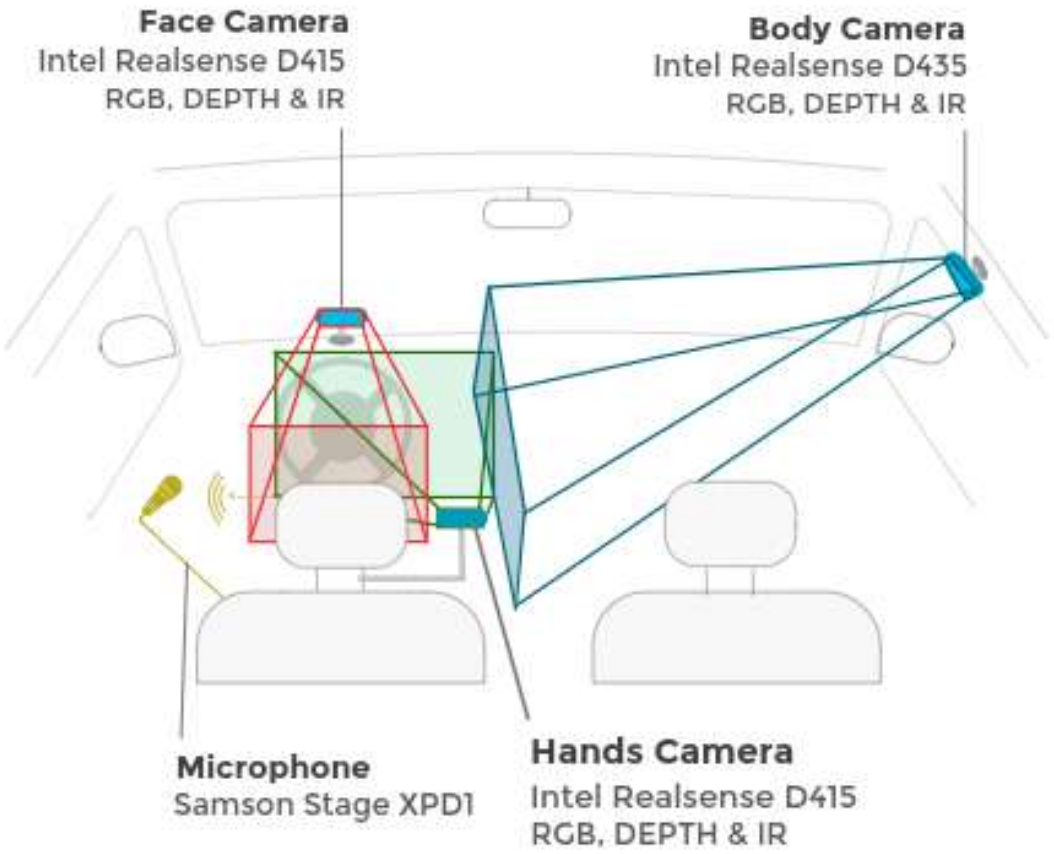
- Cellphone (celular), si un celular aparece en la escena.
- Hair comb (peine), si un peine aparece en la escena.
- Bottle (botella), si una botella aparece en la escena.

Nivel 6: Driver actions (acciones del conductor)

En este nivel, son 13 acciones que fueron anotadas. La información principal es tomada de la cámara del cuerpo, aunque se usan de complemento las cámaras de la cara y manos.

Las etiquetas de este nivel son las siguientes:

- Safe driving (conducción segura), cuando se realiza la tarea de conducir.
- Texting right (texteando derecha), cuando tiene el teléfono celular con la mano derecha y se empieza a escribir.
- Phone call right (llamada derecha), cuando se tiene el teléfono celular con la mano derecha y se realiza la acción de llamar.
- Texting left (texteando izquierda), cuando se tiene el teléfono celular con la mano izquierda y se empieza a escribir.
- Phone call left (llamada izquierda), cuando se tiene el teléfono celular con la mano izquierda y se realiza la acción de llamar.
- Radio, cuando se interactúa con la radio.
- Drinking (beber), cuando se sostiene una botella y se realiza la acción de beber.
- Reach side (buscar algo), cuando se intenta o se tiene la intensión de buscar algún objeto.
- Hair and makeup, (peinarse y maquillarse), cuando el conductor realiza la acción de peinarse o maquillarse.
- Talking to passenger (hablando con el pasajero), cuando se habla con el copiloto.
- Reach backseat (buscar asiento de atrás), cuando se busca algún objeto en el asiento trasero.
- Change gear (cambiar la marcha), cuando se tiene la intensión o se realiza la acción de cambiar la palanca de cambios.
- Stand still waiting (quedarse esperando), cuando el conductor no realiza ninguna acción.
- Unclassified (sin clasificar), cuando el conductor realiza cualquier otra acción que no pertenezca a los antes mencionados.



Toda la información de este anexo, incluidas las imágenes, fueron sacadas de la página web, guía y repositorio de Github del proyecto DMD, que se encuentran en https://dmd.vicomtech.org/ y https://github.com/Vicomtech/DMD-Driver-Monitoring-Dataset/wiki/DMD-distraction-related-action-annotation-criteria.

F. Hoja de datos XL4015

XLSEMI

Datasheet

5A 180KHz 36V Buck DC to DC Converter **XL4015**

Features

- Wide 8V to 36V Input Voltage Range
- Output Adjustable from 1.25V to 32V
- Maximum Duty Cycle 100%
- Minimum Drop Out 0.3V
- Fixed 180KHz Switching Frequency
- 5A Constant Output Current Capability
- Internal Optimize Power MOSFET
- High efficiency up to 96%
- Excellent line and load regulation
- Built in thermal shutdown function
- Built in current limit function
- Built in output short protection function
- Available in TO263-5L package

Applications

- LCD Monitor and LCD TV
- Portable instrument power supply
- Telecom / Networking Equipment

General Description

The XL4015 is a 180 KHz fixed frequency PWM buck (step-down) DC/DC converter, capable of driving a 5A load with high efficiency, low ripple and excellent line and load regulation. Requiring a minimum number of external components, the regulator is simple to use and include internal frequency compensation and a fixed-frequency oscillator.

The PWM control circuit is able to adjust the duty ratio linearly from 0 to 100%. An over current protection function is built inside. When short protection function happens, the operation frequency will be reduced from 180KHz to 48KHz. An internal compensation block is built in to minimize external component count.

TO263-5L

Figure1. Package Type of XL4015

Function Block

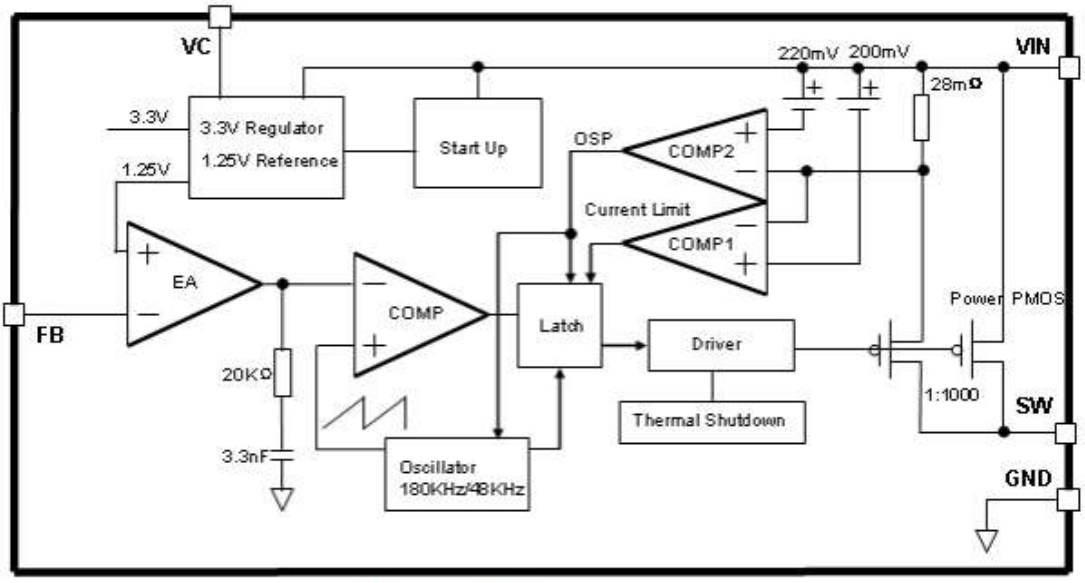


Figure3. Function Block Diagram of XL4015

Typical Application Circuit

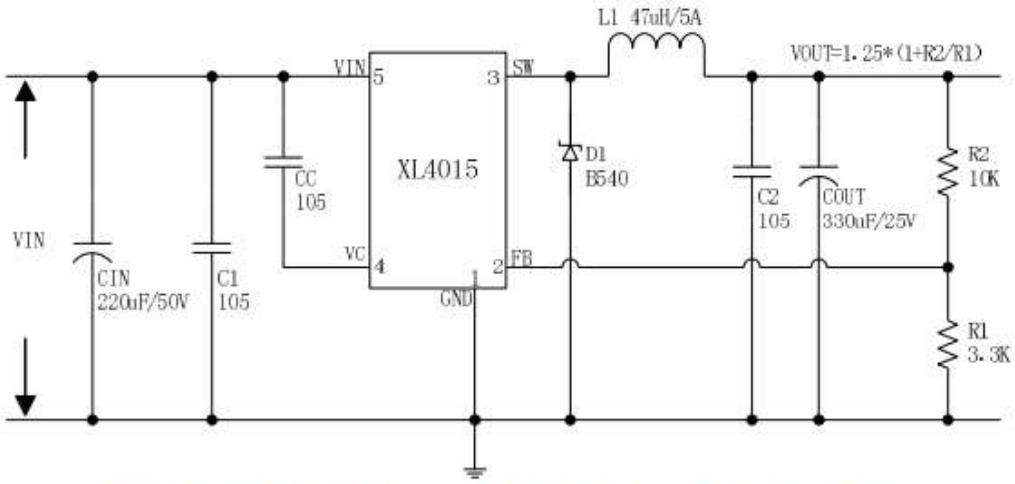


Figure4. XL4015 Typical Application Circuit (VIN=8V~36V, VOUT=5V/5A)

Typical System Application (VOUT=5V/5A)

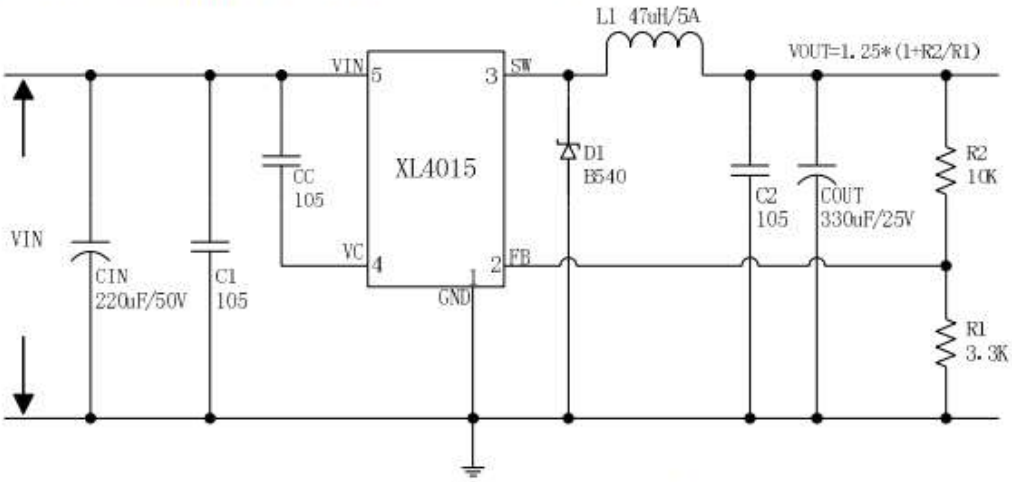


Figure5. XL4015 System Parameters Test Circuit (VIN=8V~36V, VOUT=5V/5A)

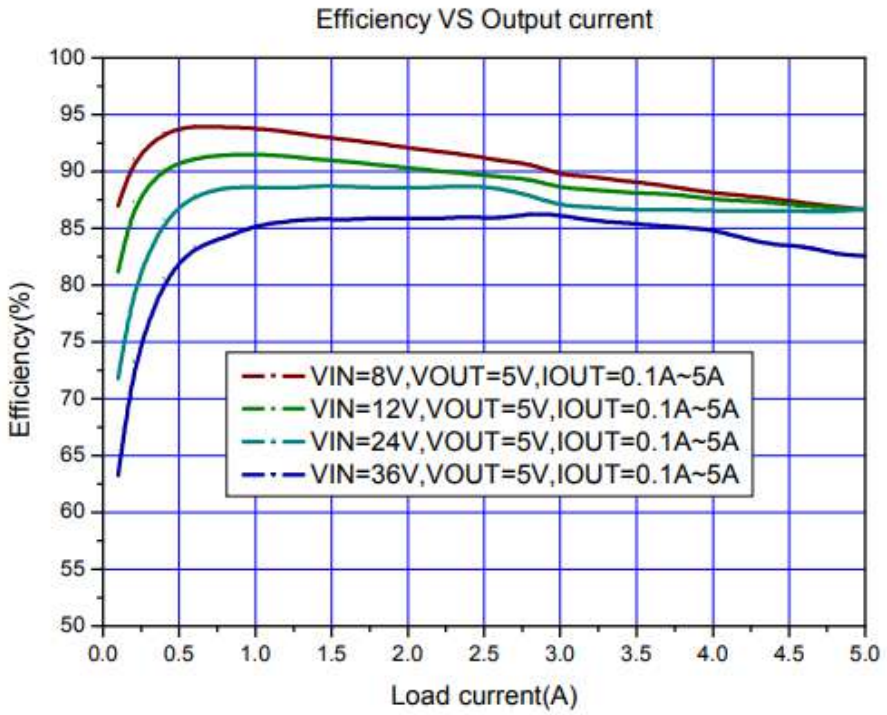


Figure6. XL4015 System Efficiency Curve

G. Instalación de RKNN-Toolkit2

La instalación de RKNN-Toolkit2 en un sistema operativo Linux (para este caso Manjaro Linux) requiere la creación de un entorno controlado mediante el uso de Miniconda para evitar conflictos con dependencias del sistema. A continuación, se detallan los paso realizados:

1. **Instalación de herramientas de desarrollo**: Para garantizar que el sistema tiene las herramientas necesarias para compilar, se ejecutó el siguiente comando en una terminal de Linux:

```
sudo pacman -Syu base base-devel
```

2. **Configuración del entorno Python**: Se utilizó Miniconda para crear un entorno de Python 3.11 donde se instaló RKNN-Toolkit2 y sus dependencias. El comando ejecutado para la creación y activación del entorno se muestran a continuación:

```
conda create -n rknn-toolkit2 python=3.11
conda activate rknn-toolkit2
```

3. **Descarga de RKNN-Toolkit2**: La descarga se realiza desde la plataforma de GitHub disponible en: https://github.com/airockchip/rknn-toolkit2/, se descomprime si fue descargado en ZIP y luego se accede a la carpeta. Suponiendo que se encuentra en descargas.

```
cd ./Descargas
cd ./rknn-toolkit2-2.0.0-beta0
```

4. **Instalación de dependencias**: Las dependencias fueron instaladas desde un archivo de requisitos específico para la versión de Python 3.11.

```
pip install -r \
rknn-toolkit2/packages/requirements_cp311.txt
```

5. **Instalación de RKNN-Toolkit2**: Finalmente, se instaló la herramienta con el siguiente comando, para este caso se usó la versión 2.0.0 beta0.

```
pip install \
rknn-toolkit2/packages/\
rknn_toolkit2-{version}+{commit}-cp311-cp311_linux_x86_64.whl
```

Para obtener información sobre las versiones y commits específicos, se recomienda consultar la documentación oficial en: https://github.com/airockchip/rknn-toolkit2/.

H. Código de detección de distracción

Código principal

```
import sys
import cv2
import mediapipe as mp
import time
import math
import numpy as np
import pickle
from rknnlite.api import RKNNLite
import utils_rk3588
import distraction_utils_rk3588
import pygame
import subprocess

mp_face_mesh = mp.solutions.face_mesh

# Cargar la tabla de consulta
with open("./model/lookup_table.pkl", "rb") as f:
    lookup_table = pickle.load(f)
yaw_values = sorted(set([key[0] for key in lookup_table.keys()]))
pitch_values = sorted(set(key[1] for key in lookup_table.keys()))
roll_values = sorted(set(key[2] for key in lookup_table.keys()))

# Cargar el sonido y la notificacion
pygame.mixer.init()
sound = pygame.mixer.Sound("./model/notificacion_1.mp3")

# Cargar archivo para controlar los GPIO
gpio_file_path = "./gpio_file.sh"
subprocess.run(["bash", gpio_file_path, "inicializar"])

# Iniciar estado de distraccion
distraccion_state = 1
ultimo_evento = None
no_rostro_evento = "El rostro salio de los limites de la camara"

# Definir tamano de imagenes
width = 800
height = 640

# Iniciar camara
cap = cv2.VideoCapture(0)
#cap = cv2.VideoCapture("./model/gA_1_s2.mp4")

```

```
# Configurar grabacion
cap.set(cv2.CAP_PROP_FRAME_WIDTH, width)
cap.set(cv2.CAP_PROP_FRAME_HEIGHT, height)
cap.set(cv2.CAP_PROP_FPS, 30)
cap.set(cv2.CAP_PROP_FOURCC, cv2.VideoWriter_fourcc(*'MJPG'))

DEVICE_COMPATIBLE_NODE = "/proc/device-tree/compatible"

limite_inferior = -1e10
limite_superior = 1e10

# Definir modelo
model_rknn = "./model/MNASNet1-0.rknn"

rknn_lite = RKNNLite()

utils_rk3588.evento("Inicio", "Cargando el modelo ...")

model = rknn_lite.load_rknn(model_rknn)

if model != 0:
    utils_rk3588.evento("Error", "Error al cargar el modelo")
    exit(model)

utils_rk3588.evento("Inicio", "Modelo cargado")
utils_rk3588.evento("Inicio", "Iniciando el runtime...")

host_name = utils_rk3588.get_host(DEVICE_COMPATIBLE_NODE)

utils_rk3588.evento("Inicio", f"Nombre del host: {host_name}")

# Validando procesador
if host_name == "RK3588":
    # Utilizando los 3 nucleos de la NPU
    ret = rknn_lite.init_runtime(core_mask=RKNNLite.NPU_CORE_0_1_2)
else:
    ret = rknn_lite.init_runtime()

if ret != 0:
    utils_rk3588.evento("Error", "Fallo el inicio del runtime")

utils_rk3588.evento("Inicio", "Hecho")

try:
    with mp_face_mesh.FaceMesh(
        static_image_mode=False,
        max_num_faces=1,
        min_detection_confidence=0.5) as face_mesh:

        frame_counter = 0
        start_time = time.time()
```

```
95          fps = 0
96
97          # Variables para umbral
98          OT = 0
99          valor_superado = 0
100         count = 0
101         fps_superados = 8
102
103         while True:
104
105             ret, frame = cap.read()
106
107             if not ret:
108                 break
109
110             frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
111             results = face_mesh.process(frame_rgb)
112
113             if results.multi_face_landmarks:
114                 face_landmarks = results.multi_face_landmarks[0]
115
116                 landmark_points = [(int(face_landmarks.landmark[idx
                        ].x * frame.shape[1]),
117                                     int(face_landmarks.landmark[idx
                                            ].y * frame.shape[0]))
118                                    for idx in [130, 243, 463, 359]]
119
120                 # Calcula el punto medio de la cabeza
121                 x_center = sum([p[0] for p in landmark_points]) //
                        len(landmark_points)
122                 y_center = sum([p[1] for p in landmark_points]) //
                        len(landmark_points)
123
124                 # Calcula la distancia promedio entre los puntos de
                        referencia
125                 distances = [math.sqrt((p[0] - x_center) ** 2 + (p
                        [1] - y_center) ** 2) for p in landmark_points]
126                 avg_distance = sum(distances) / len(distances)
127
128                 # Calcula el tamano del recorte en funcion de la
                        distancia promedio
129                 crop_size = int(6.0 * avg_distance)  # Ajustar el
                        factor multiplicativo.
130
131                 # Calcula las coordenadas del recorte
132                 x_min = x_center - crop_size // 2
133                 y_min = y_center - crop_size // 2
134                 x_max = x_min + crop_size
135                 y_max = y_min + crop_size
136
```

```
137                 # Verifica si las coordenadas de recorte estan
                        dentro de los limites de la imagen
138                 if x_min >= 0 and y_min >= 0 and x_max <= frame.
                        shape[1] and y_max <= frame.shape[0]:
139                     # Recorta la region de interes (cabeza)
140                     head_crop = frame[y_min:y_max, x_min:x_max]
141                     head_crop = cv2.resize(head_crop, (224, 224))
142                     head_crop1 = np.expand_dims(head_crop, 0)
143
144                     outputs = rknn_lite.inference(inputs=[head_crop1
                            ])
145
146                     if outputs is not None:
147                         outputs = np.clip(outputs, limite_inferior,
                                limite_superior)
148                     else:
149                         utils_rk3588.evento("Error", "Outputs␣es␣
                                None")
150
151                     outputs= outputs[0]
152
153                     euler = utils_rk3588.
                            compute_euler_angles_from_rotation_matrices(
                            outputs)*180/np.pi
154
155                     p , y , r= euler[:,0], euler[:,1], euler[:,2]
156
157                     resultado = distraction_utils_rk3588.
                            interpolar_tabla(y, p, r, lookup_table=
                            lookup_table, yaw_valores=yaw_values,
                            pitch_valores=pitch_values, roll_valores=
                            roll_values)
158
159                     if resultado > 5:
160                         OT +=1
161                         valor_superado += 1
162                     else:
163                         if valor_superado > fps_superados:
164                             count += 1
165                         valor_superado = 0
166
167                     distraccion = "Distraido" if valor_superado > 5
                            else "No␣distraido"
168
169                     #sonido y leds
170
171                     if distraccion == "Distraido" and not pygame.
                            mixer.get_busy():
172                         if distraccion_state == 0:
173                             sound.play()
```

```
                                subprocess.run(["bash", gpio_file_path,
                                    "distraccion"])
                                distraccion_state = 1
                                utils_rk3588.evento("Evento", f"
                                    Distraido / FPS = {fps:.2f}")
                                ultimo_evento = "Distraido"
                            else:
                                sound.play()
                                utils_rk3588.evento("Evento", f"
                                    Distraido / FPS = {fps:.2f}")
                                ultimo_evento = "Distraido"

                        if not pygame.mixer.get_busy() and distraccion
                            == "No distraido":
                            if distraccion_state == 1:
                                subprocess.run(["bash", gpio_file_path,
                                    "no_distraccion"])
                                distraccion_state = 0
                                utils_rk3588.evento("Evento", f"No
                                    distraido / FPS = {fps:.2f}")
                                ultimo_evento = "No distraido"

                    else:
                        # No se detecta rostro, pausa el programa
                        if no_rostro_evento != ultimo_evento:
                            subprocess.run(["bash", gpio_file_path, "
                                apagar"])
                            utils_rk3588.evento("Aviso", "El rostro
                                salio de los limites de la camara")
                            ultimo_evento = no_rostro_evento

                frame_counter += 1

                if time.time() - start_time >= 1:
                    fps = frame_counter / (time.time() - start_time)
                    frame_counter = 0
                    start_time = time.time()

    # liberar procesos y apagar indicadores
    cap.release()
    rknn_lite.release()
    subprocess.run(["bash", gpio_file_path, "apagar"])

except KeyboardInterrupt:
    # liberar procesos y apagar indicadores
    cap.release()
    rknn_lite.release()
    subprocess.run(["bash", gpio_file_path, "apagar"])
    utils_rk3588.evento("Fin", "Saliendo ...")
    utils_rk3588.evento_restante()
    sys.exit(0)
```

Utilidades

```
import numpy as np
import bisect
import platform

def get_host(DEVICE_COMPATIBLE_NODE):
    system = platform.system()
    machine = platform.machine()
    os_machine = system + "-" + machine
    if os_machine == "Linux-aarch64":
        try:
            with open(DEVICE_COMPATIBLE_NODE) as f:
                device_compatible_str = f.read()
                if "rk3588" in device_compatible_str:
                    host = "RK3588"
                elif "rk3562" in device_compatible_str:
                    host = "RK3562"
                else:
                    host = "RK3566_RK3568"
        except IOError:
            print("Leer el nodo del dispositivo {} fallo".format(
               DEVICE_COMPATIBLE_NODE))
            exit(-1)
    else:
        host = os_machine

    return host

def distancia_umbral(y, p, r):
    beta = [0.5201, 7.0561,-3.4258]
    umbral = 11.3665

    d = np.sqrt((y - beta[0])**2 + (p - beta[1])**2 + (r - beta[2])
       **2)

    return umbral, d

def distancia_umbral_2(y, p, r):
    umbrales = [11.31742, 10.60552, 10.97048]
    beta = [[13.968131692979687, 6.02086901968914,
       -4.25979024982431], [-13.881612977783512, 8.53865757182063,
       -2.185528786042723], [0.37379452192541723, 6.685269585595015,
        -3.4778315043685724]]

    beta1 = beta[0]
    beta2 = beta[1]
    beta3 = beta[2]

```

```
    d1 = np.sqrt(((y - beta1[0])**2 + (p - beta1[1])**2 + (r - beta1
        [2])**2))
    d2 = np.sqrt(((y - beta2[0])**2 + (p - beta2[1])**2 + (r - beta2
        [2])**2))
    d3 = np.sqrt(((y - beta3[0])**2 + (p - beta3[1])**2 + (r - beta3
        [2])**2))

    D1 = umbrales[0]
    D2 = umbrales[1]
    D3 = umbrales[2]

    return D1, D2, D3, d1, d2, d3

def encontrar_intervalos(valores, x):
    pos = bisect.bisect_left(valores, x)
    if pos == 0:
        return valores[0], valores[1]
    if pos == len(valores):
        return valores[-2], valores[-1]
    return valores[pos - 1], valores[pos]

def interpolacion_trilineal(x, x1, x2, y, y1, y2, z, z1, z2, valores
    ):
    def division_segura(numerador, denominador):
        return numerador / denominador if denominador != 0 else 0

    xd = division_segura(x - x1, x2 - x1)
    yd = division_segura(y - y1, y2 - y1)
    zd = division_segura(z - z1, z2 - z1)

    fx00 = valores[0] * (1 - xd) + valores[4] * xd
    fx01 = valores[1] * (1 - xd) + valores[5] * xd
    fx10 = valores[2] * (1 - xd) + valores[6] * xd
    fx11 = valores[3] * (1 - xd) + valores[7] * xd

    fxy0 = fx00 * (1 - yd) + fx10 * yd
    fxy1 = fx01 * (1 - yd) + fx11 * yd

    return fxy0 * (1 - zd) + fxy1 * zd

def interpolar_tabla(yaw, pitch, roll, lookup_table, yaw_valores,
    pitch_valores, roll_valores):
    yaw_0, yaw_1 = encontrar_intervalos(yaw_valores, yaw)
    pitch_0, pitch_1 = encontrar_intervalos(pitch_valores, pitch)
    roll_0, roll_1 = encontrar_intervalos(roll_valores, roll)

    f000 = lookup_table[(yaw_0, pitch_0, roll_0)]
    f001 = lookup_table[(yaw_0, pitch_0, roll_1)]
    f010 = lookup_table[(yaw_0, pitch_1, roll_0)]
    f011 = lookup_table[(yaw_0, pitch_1, roll_1)]
    f100 = lookup_table[(yaw_1, pitch_0, roll_0)]
```

```
    f101 = lookup_table[(yaw_1, pitch_0, roll_1)]
    f110 = lookup_table[(yaw_1, pitch_1, roll_0)]
    f111 = lookup_table[(yaw_1, pitch_1, roll_1)]

    return interpolacion_trilineal(yaw, yaw_0, yaw_1, pitch, pitch_0
       , pitch_1, roll, roll_0, roll_1, [f000, f001, f010, f011,
       f100, f101, f110, f111])
```

Conversión formato Pytorch a RKNN

```
from rknn.api import RKNN
import torch
import os
import sys
sys.path.append("../HPE_RK3588/")
import models

if len(sys.argv) > 1:
    name_model = sys.argv[1]
if len(sys.argv) > 2:
    dir_model = sys.argv[2]

# Cargando el modelo preentrenado en formato .tar y guardandolo en
   formato .pt
def export_pytorch_model():
    model_saved = None
    if not os.path.exists("./rknn/models"):
        os.makedirs("./rknn/models")
    try:
        modelo_clase = getattr(models, name_model)

        model = modelo_clase()

        print(f"Instancia creada para el modelo {name_model}")
    except AttributeError:
        print(f"No se pudo crear la instancia para el modelo {
           name_model}")

    model_saved = dir_model

    saved_state_dict = torch.load(model_saved, map_location="cpu")

    if "model_state_dict" in saved_state_dict:
        model.load_state_dict(saved_state_dict["model_state_dict"])
    else:
        model.load_state_dict(saved_state_dict)
    model.eval()
```

```
    trace_model = torch.jit.trace(model, torch.Tensor(1, 3, 224, 224))
    trace_model.save(f"./rknn/models/{name_model}.pt")

if __name__ == "__main__":
    # exporta el modelo y verifica si este existe previamente o no
    model = "./rknn/models/{}.pt".format(name_model)
    if not os.path.exists(model):
        export_pytorch_model()

    input_size_list = [[1, 3, 224, 224]]

    rknn = RKNN(verbose=False)

    # configuraciones de preprocesamiento
    print("**** Configurando el modelo ****")
    rknn.config(mean_values=[0.485*255, 0.456*255, 0.406*255],
                std_values=[0.229*255, 0.224*255, 0.225*255],
                target_platform='rk3588')
    print("Hecho")

    print("**** Cargando el modelo ...")
    ret = rknn.load_pytorch(model=model, input_size_list=input_size_list)
    if ret != 0:
        print("Fallo al cargar el modelo!")
        exit(ret)
    print("Hecho")

    # Construir el modelo
    print("**** Construyendo el modelo ...")
    #ret = rknn.build(do_quantization=True, dataset="./dataset/AFLW2000/dataset.txt")
    ret = rknn.build(do_quantization=False)
    if ret != 0:
        print("Fallo al construir el modelo!")
        exit(ret)
    print("Hecho")

    # Exportar el modelo
    print("**** Exportando el modelo a rknn ...")
    ret = rknn.export_rknn(f"./rknn/models/{name_model}.rknn")
    if ret != 0:
        print("Fallo la exportacion del modelo!")
        exit(ret)
    print("Hecho")

    rknn.release()
```

GPIO

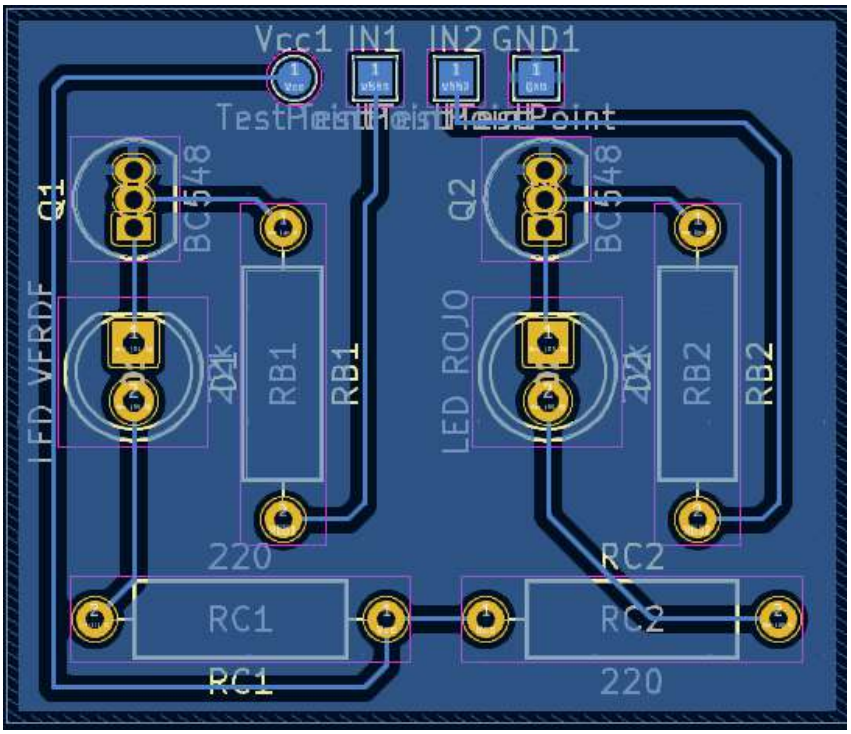
```
#!/bin/bash

inicializar_gpio() {
    gpio mode 5 out
    gpio mode 2 out
    apagar_led
}
encender_led_rojo() {
    gpio write 2 1
    gpio write 5 0

}
encender_led_verde() {
    gpio write 2 0
    gpio write 5 1

}
apagar_led() {
    gpio write 2 0
    gpio write 5 0

}
salir() {
    exit 1
}

# Comprobar si hay un argumento
if [ $# -eq 0 ]; then
    uso
fi
# Ejecutar la funcion correspondiente segun el argumento
case "$1" in
    inicializar)
        inicializar_gpio
        ;;
    distraccion)
        encender_led_rojo
        ;;
    no_distraccion)
        encender_led_verde
        ;;
    apagar)
        apagar_led
        ;;
    *)
        salir
        ;;
```

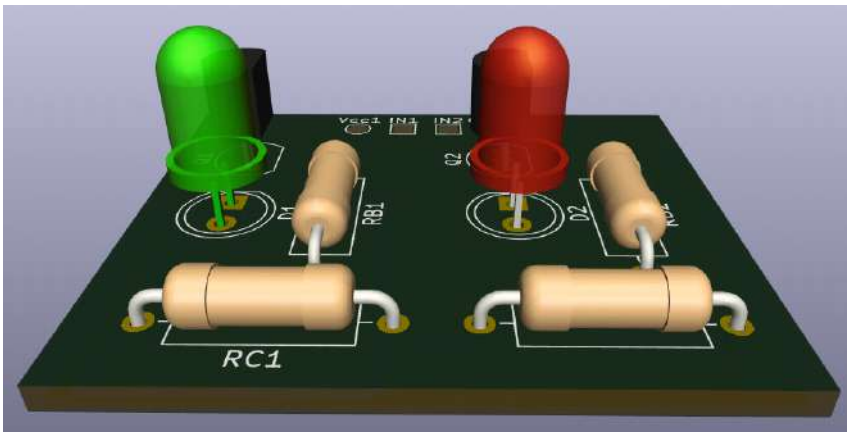
```
esac
```

Script main.sh

```
#!/bin/bash

echo "Bienvenido a HPE_RK3588"

while true; do
    echo "Seleccione una opcion:"
    echo "1. Entrenamiento y tests de modelos"
    echo "2. Entrenar y validar todo"
    echo "3. Convertir modelo a .rknn"
    echo "4. Salir"

    read -p "Ingrese el numero de la opcion deseada: " opcion

    case $opcion in
        1)
            echo "Escoger modelo para su entrenamiento y evaluacion ..."
            bash "$(pwd)/scripts/train.sh"
            ;;
        2)
            echo "Comenzando el entrenamiento de todos los modelos ..."
            bash "$(pwd)/scripts/train_all.sh"
            ;;
        3)
            echo "Iniciando conversion"
            bash "$(pwd)/scripts/conversion.sh"
            ;;

        4)
            echo "Saliendo del script."
            exit 0
            ;;
        *)
            echo "Opcion no valida. Por favor, ingrese un numero valido."
            ;;
    esac
done
```

I. PCB


Vcc1 IN1 IN2 GND1
Q1
BC548
Q2
BC548
LED VERDE
LED ROJO
RB1
RB1
RB2
RB2
220
RC2
RC1
RC2
RC1
220


RB1
RC1