UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE GUAYAQUIL

CARRERA DE ELECTRÓNICA Y AUTOMATIZACIÓN

DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO DE IDENTIFICACIÓN DE USO DE CASCO EN MOTOCICLISTAS.

Trabajo de titulación previo a la obtención del

Título de Ingeniero en Electrónica

AUTORES: DARWIN ENRIQUE GARCIA MUÑOZ

TUTOR: Ing. AVELINO VICENTE PEÑARANDA IDROVO, Msc.

Guayaquil – Ecuador

2024

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Yo, Darwin Enrique García Muñoz con documento de identificación N° 0951666429, manifiesto que:

Soy el autor y responsable del presente trabajo; y, autorizo a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación

Guayaquil, 02 de agosto del año 2024.

Atentamente,

Darwin García

Darwin Enrique García Muñoz
0951666429

CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

Yo, Darwin Enrique García Muñoz con documento de identificación N° 0951666429, manifiesto mi voluntad y por medio del presente documento cedo a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que soy autor del Proyecto Técnico: "Diseño e implementación de un Prototipo de Identificación de Uso de Casco en Motociclistas", el cual ha sido desarrollado para optar por el título de: Ingeniero en Electrónica, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribo este documento en el momento que hago la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Guayaquil, 02 de agosto del año 2024.

Atentamente,

Darwin García

Darwin Enrique García Muñoz
0951666429

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Vicente Avelino Peñaranda Idrovo, con documento de identificación N° 0916113426, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO DE IDENTIFICACIÓN DE USO DE CASCO EN MOTOCICLISTAS, realizado por Darwin Enrique García Muñoz con documento de identificación N° 0951666429, obteniendo como resultado final el trabajo de titulación bajo la opción de Proyecto Técnico que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Guayaquil, 02 de agosto del año 2024.

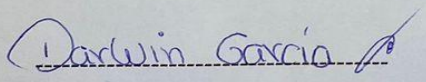
Atentamente,

Ing. Vicente Avelino Peñaranda Idrovo, Msc.
0916113426

DEDICATORIA

Mi título universitario de lo dedico a mi familia, en especial a mis padres que fueron mi principal apoyo a lo largo de esta larga carrera.

También quiero agradecerle a Dios que me permitió terminar esta etapa en mi educación.



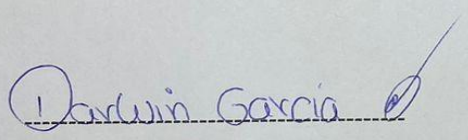
Darwin Enrique García Muñoz

AGRADECIMIENTO

Para mí, terminar esta carrera universitaria me llena de satisfacción y orgullo y le agradezco a los docentes que tuve por compartirme su conocimiento, a mis compañeros que fundamentaron en este camino y se convirtieron en amigos.

Y quiero agradecer muy profundamente a mi madre Liliana Muñoz quien ha cuidado de mí siempre y me ha llevado hasta aquí y seguramente me acompañara en todos los logros de mi vida.

También agradezco a mi enamorada Camila Lozano quien también ha sido mi principal apoyo para lograr esta meta, le agradezco por siempre incentivarme a superarme y nunca rendirme.



Darwin Enrique García Muñoz

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLA

RESUMEN

Una de las principales causas de millones de muertes al año a nivel mundial, sin duda alguna, son los accidentes de tránsito y cabe recalcar que los motociclistas son los usuarios más vulnerables. Aunque el casco de seguridad reduce la mortalidad, las cifras aumentan cada año. Específicamente en países subdesarrollados, los datos de defunciones son más elevados [1] Para revertir este problema, es importante implementar tecnologías innovadoras como la inteligencia artificial en prototipos diseñados para identificar a individuos que no porten el casco de seguridad y así corroborar con los agentes de seguridad vial para aplicar las respectivas sanciones y por ende mejorar la seguridad vial. Por lo tanto, este proyecto pretende diseñar e implementar un prototipo de identificación de cascos en motociclistas.

Este trabajo se basa en el entrenamiento de un modelo basado en el patrón de reconocimiento de imágenes mediante la herramienta tachable machine de Google, la implementación del modelo dentro de una computadora raspberry y la integración de una cámara. Esta tecnología permite la identificación del uso de cascos en motociclistas en tiempo real.

*Palabras Claves***:** detección, cascos, raspberry pi, Teachable machine, Python, inteligencia artificial.

ABSTRACT

One of the main causes of millions of deaths per year worldwide is undoubtedly traffic accidents, and motorcyclists are the most vulnerable users. Even though the use of safety helmets reduces mortality, the figures increase every year. Specifically in underdeveloped countries, the number of deaths is higher [1]. To reverse this problem, it is important to implement innovative technologies such as artificial intelligence in prototypes designed to identify individuals who do not wear helmets and thus corroborate with road safety agents to apply the respective sanctions and thus improve road safety. Therefore, the main objective of this project is to design and implement a prototype to identify the use of helmets by motorcyclists.

The project is based on the training of a model based on the image recognition pattern using Google's tactile machine tool, the implementation of the model inside a Raspberry computer and the integration of a camera. This technology allows identifying helmet use in motorcyclists in real time.

Keywords: sensing, helmets, raspberry pi, teachable machine, Python, artificial intelligence.

INTRODUCCIÓN

En los últimos años, las principales causas de muerte por accidentes de tránsito se han incrementado en un 90% en países subdesarrollado, además, que alrededor del planeta, más de 1 millón de personas fallecen a causa de lo expuesto anteriormente, según la Organización Panamericana de la Salud. Por otro lado, las cifras estadísticas apuntan que el 49% de defunciones por accidentes en las vías de tránsito son los usuarios motorizados, peatones y ciclistas **[1]**. Cabe recalcar que existen algunas razones por las cuales ocurren estos siniestros de tránsito, entre estas encontramos el exceso de velocidad, la imprudencia de no acatar y obedecer señales designadas por las instituciones de seguridad vial y sobre todo el desuso de equipos destinados para conducir como por ejemplo el casco de seguridad en motociclistas **[2]**. Para reducir las cifras mortales que cada año aumentan desenfrenadamente, se debe incluir la inteligencia artificial para detectar el uso de cascos en motociclistas y lograr que los usuarios de estos medios de transporte cumplan las normas asignadas por las autoridades viales.

El proyecto expuesto a continuación, se basa en el diseño e implementación de un prototipo para poder identificar el uso de cascos en motociclistas utilizando la herramienta Teachable Machine de Google, que está fundamentada en inteligencia artificial, como conexión para así poder realizar los respectivos acondicionamientos en algún modelo, así lograr prevenir y reducir lesiones craneoencefálicas y defunciones a causa del desuso de cascos de seguridad en motociclistas en accidentes de tránsito.

Por ende, el objetivo principal se centra en diseñar e implementar un prototipo de identificación de uso de cascos en motociclistas.

1. PLANTEAMIENTO DEL PROBLEMA

1.1.DEFINICIÓN DEL PROBLEMA

Los accidentes viales, provocan por lo menos una cifra de 1.19 millones de muertes a nivel mundial. Los usuarios de motos son el grupo más vulnerable a causa de imprudencias al conducir y no acatar las normas del uso de cascos de seguridad. Según la Organización Mundial de la Salud, en su "*THE GLOBAL STATUS REPORT ON ROAD SAFETY 2023",* menciona que, en 2018, los datos apuntan a que hubo un elevado índice de mortalidad en el grupo de poblacional que usa motos **[3]**, además afirma que los motociclistas, poseen cifras altas de lesiones craneoencefálicas y que en el año 2010 hubo un aumento del 15 % de defunciones y en el 2013 se elevó un 5% más que el año mencionado anteriormente **[4].**

Los accidentes craneoencefálicos, a nivel global y Latinoamérica, son una proporción significativa de las lesiones causadas por los siniestros viales. Pese a que existen normativas que obligan a utilizar en casco de seguridad en motociclistas, ciertos comportamientos en estos usuarios influyen en que no se lo use de manera correcta **[5]**.

En América Latina, la morbimortalidad por lesiones intracraneales tiene una similitud en cuanto a datos observados en el mundo. En el año 2024, las lesiones a causa de accidentes de tránsito superaron un 3.4% a la carga mundial de patología medida en años de vida ajustados por discapacidad o en sus siglas AVAD, con 2,2 de incidencia por cada 100.000 individuos **[6].**

En Ecuador, en el año 2023, se han registrado 32.687 siniestros de tránsito en los que vieron involucrados motociclistas. Al comparar con el año 2022, se observó un incremento de casi un 0.10% que representan a 32.660 accidentes reportados. Las causas de estos siniestros son debido a múltiples circunstancias, las cuales son por maniobras peligrosas, velocidad excesiva, incumplimiento de las normas de seguridad vial y principalmente por el desuso de equipos de protección como el casco de seguridad **[2].**

1.2.DELIMITACIÓN DEL PROBLEMA

1.2.1 Delimitación temporal

La implementación del prototipo cubre el periodo académico 64, durante el transcurso de este tiempo se acarrea el entrenamiento, desarrollo y puesta en marcha de un prototipo de identificación de uso de cascos en motociclistas, también se sirve efectuar pruebas en distintos espacios y ambientes para certificar la eficiencia del prototipo

1.2.2. Delimitación Espacial

Este prototipo se dirige a la ciudadanía y conductores motociclistas en los cuales se desea inculcar, normalizar y fomentar el uso correcto de los cascos de seguridad

1.2.3. Delimitación Académica

Este proyecto permite el incremento y la familiarización con la inteligencia artificial, el análisis de datos y los sistemas embebidos para atacar el problema de la verificación del correcto uso de cascos de seguridad en motociclistas, considerándose los aspectos prácticos y teóricos encadenados con estos fundamentos para la implementación de la solución planteada

2. OBJETIVOS

2.1. OBJETIVO GENERAL

- Diseñar e implementar un prototipo de identificación de uso de cascos en motociclistas.

2.2. OBJETIVOS ESPECÍFICOS

- Realizar la recolección de imágenes específicas para identificar motociclistas sin casco.
- Cargar y obtener el modelo utilizando el software Machine Learning para separar los motociclistas con casco de los que no utilizan casco.
- Implementar el modelo dentro de una computadora Raspberry y realizar pruebas del correcto funcionamiento del prototipo.

3. FUNDAMENTO TEÓRICO

3.1.INTELIGENCIA ARTIFICIAL

Se define a la inteligencia artificial a la rama que genera sistemas cualificados para llevar a cabo un sin número de tareas, específicamente informáticas, semejantes a las que realizaría la inteligencia humana. En la ingeniería, este campo de la computacion está revolucionando ya que permite realizar, resolver y optimizar múltiples procesos a la vez, provocando un cambio significativo en la humanidad **[7]**.

3.1.1. Uso de la inteligencia artificial en ingeniería

La ingeniería está experimentando una transformación sin precedentes gracias a la inteligencia artificial. Esta tecnología permite a los ingenieros tomar decisiones más informadas y automatizar tareas complejas, lo que a su vez optimiza los procesos y reduce costos. Sin embargo, hoy es crucial considerar las implicaciones éticas y sociales de la IA, como su impacto en el empleo y la necesidad de establecer marcos regulatorios adecuados **[7].**

3.1.2. Análisis y diagnóstico de datos

La capacidad de la inteligencia artificial para procesar grandes cantidades de datos y descubrir patrones complejos está transformando la manera en que los ingenieros abordan los problemas. Al aprovechar estas capacidades, los ingenieros pueden mejorar la precisión de sus análisis y tomar decisiones más informadas, lo que a su vez conduce a una mayor eficacia y productividad **[7].**

3.1.3. Control de procesos y optimización

La inteligencia artificial está transformando radicalmente la ingeniería, especialmente en lo que respecta a la optimización de procesos. Gracias a algoritmos de aprendizaje automático, ésta puede analizar numerosas cantidades de datos para identificar patrones y tendencias ocultas. Esta capacidad favorece a que los ingenieros puedan prevenir fallas en equipos, facilitar la planificación de proyectos, además de aumentar la eficacia y la productividad **[7] [8]**.

3.1.4. Diseño y fabricación

El aumento que hoy en día se ha observado sobre la importancia de la inteligencia artificial en el sector industrial, exige que los ingenieros estén capacitados para enfrentar los desafíos del mañana. Hoy es de suma importancia que los programas de ingeniería incorporen la inteligencia artificial en la hoja de vida, instruyendo a los estudiantes para crear y desarrollar soluciones innovadoras basadas en esta tecnología. Por otro lado, la industria aeroespacial, que en proyectos que han surgido como los Airbus, han demostrado que la inteligencia artificial puede transformar la fabricación incluso mejorar la competitividad **[7]**.

3.1.5. Predicción de fallos

La inteligencia artificial puede vigilar de manera constante el comportamiento de los trabajadores en el entorno laboral, identificando posibles riesgos como la omisión del equipo de protección personal hola ejecución incorrecta de tareas peligrosas. Además, la inteligencia artificial puede anticipar y prevenir fallas en las máquinas, disminuyendo el riesgo de accidente y mejorando la seguridad en los procesos productivos **[7].**

3.2.ROBÓTICA

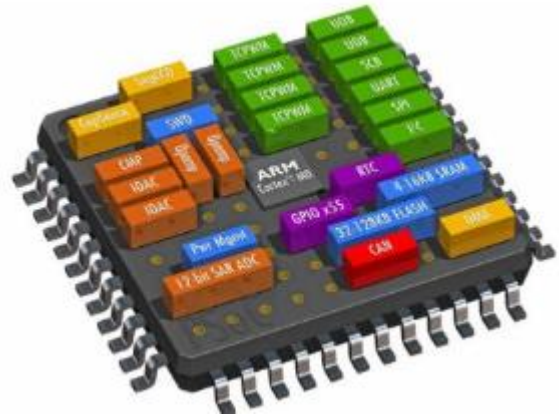
La robótica en el ámbito industrial se ha posicionado como un campo de estudio y desarrollo tecnológico de gran relevancia en la actualidad. En la actualidad, estos sistemas inteligentes trabajan en estrecha colaboración con los humanos, complementando sus habilidades y optimizando los procesos productivos **[9].**

3.3.REDES NEURONALES

Las redes neuronales están compuestas por unidades de procesamiento que se comunican entre sí. Tienen como función principal reconocer patrones en diferentes tipos de datos, como imágenes, textos manuscritos y series temporales, un ejemplo de esto son las fluctuaciones financieras. Cabe destacar que poseen la capacidad de adaptarse, aprender de manera autónoma y mejorar su funcionamiento a lo largo del tiempo **[10].**

3.4.SISTEMAS EMBEBIDOS

Se define sistemas embebidos a un computador especializado, concebido para llevar a cabo funciones concretas. Se encuentra integrado en sistemas más grandes y opera bajo restricciones de tiempo real, reaccionando a eventos de forma inmediata **[11].**



Figuras 1 Ejemplo de sistema embebido [12]

3.5.PYTHON

Python se define como un lenguaje de programación concebido por Guido Van Rossum en el año 1990. Este lenguaje promueve a la escritura de códigos legibles y sostenibles. Es un lenguaje que se puede interpretar, de tipado veloz y fuerte, este puede ser ejecutado en diversos sistemas operativos, además de sostener la programación orientada a objetos **[13].**

3.6.GOOGLE TEACHABLE MACHINE

Es una herramienta en línea que permite a los usuarios aprovechar esta tecnología para crear diversas aplicaciones personalizadas de clasificación. Esta plataforma de interfaz gráfica facilita el entrenamiento de modelos sin requerir conocimientos avanzados de programación **[14].**

3.7.CAMARA RASPBERRY

La camera Raspberry pi Es un dispositivo de solución integral ya que facilita la integración de capacidades de captura de imágenes y vídeos en distintos desarrollos. Se adapta una gran variedad de aplicaciones, los cuales son proyectos de visión artificial, sistema de control remoto y desarrollo de interfaces de usuario **[15].**

3.8.RASPBERRY PI

Se trata de un ordenador o computadora de bajo costo y con un diseño compacto desarrollado para facilitar el aprendizaje de la programación y la electrónica. La Raspberry Pi hoy es ampliamente utilizado en entornos educativos y principalmente para desarrollar proyectos de pequeña escala. Estos pueden vincularse a un monitor y emplearse con los periféricos habituales **[16]**

Figuras 2 Ejemplo de imagen Raspberry Pi [16].

3.8.1. Tipos de Raspberry Pi

La Raspberry Pi ofrece una amplia variedad de puertos de entrada y salida. Esto depende del modelo que se utilice, incluye de 1-4 puertos USB, un conector HDMI para video de alta definición, una salida de video compuesto para dispositivos con mayor antigüedad y un conector de sonido de 3.5 mm. Por otro lado, todos los modelos poseen de un conjunto de pines GPIO que se pueden configurar, estos permiten la conexión con una variedad de sensores y actuadores por medio de protocolos como I2C y SPI **[16].**

Entre los tipos de Raspberry Pi tenemos:

3.8.1.1.Raspberry Pi 1 modelo B

Creado en febrero del año 2012, posee un procesador Broadcom BCM2835 Soc A 700 MHz, una memoria RAM de 245 MB, un GPU de Coprocesador multimedia Dual core videocore 5 **[17].**

Figuras 3 Raspberry Pi 1 modelo B

3.8.1.2.Raspberry Pi 1 modelo A

Fue lanzada en hola febrero del año 2013, es un poco similar al hola modelo anterior. Se caracteriza principalmente por pesar 45 g y disponer de 26 GPIO **[17].**

Figuras 4 Raspberry Pi 1 modelo A

3.8.1.3.Raspberry Pi 1 modelo B+

Este modelo fue diseñado en el año 2014 en el mes de Julio. Dentro de sus características tenemos que su fuente de alimentación para 3.3 voltios y 1.8 voltios. Además, que posee cuatro puertos USB, 40 pone GPIO, tarjeta microSD, entre otras **[17].**

Figuras 5 Raspberry Pi 1 modelo B+

3.8.1.4.Raspberry Pi 1 modelo A+

Este modelo posee un procesador Broadcom BCM 2835 Soc Full HD, su memoria RAM es de 256 MB, se puede almacenar hoy cualquier tipo de documento que a través de tarjetas microSD y sólo posee un puerto USB **[17].**

Figuras 6 Raspberry Pi 1 modelo A+

3.8.1.5.Raspberry Pi 3 modelo B

Este modelo fue lanzado en el año 2016 en el mes de febrero, sus principales características son las conexiones inalámbricas, posee 1 gigabyte de RAM DDR2, además tiene 4 puertos USB. Por otro lado, se puede conectar a Wi-Fi y Bluetooth y también con HDMI **[17].**

Figuras 7 Raspberry Pi 3 modelo B

3.8.1.6.Raspberry Pi 5

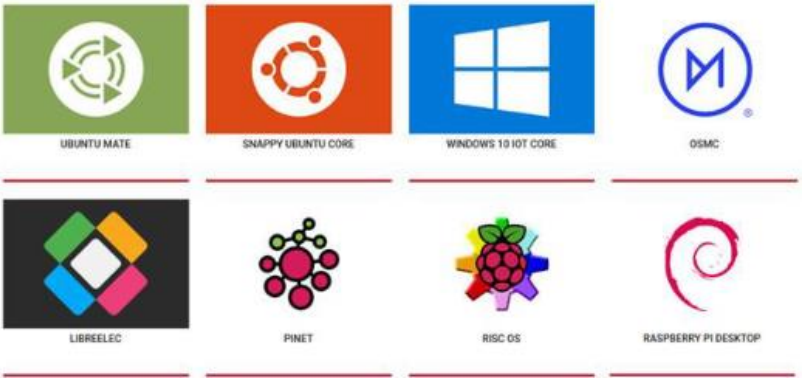
La Raspberry pi 5 representa la última generación de las computadoras populares de sólo una placa. Este modelo nuevo y novedoso posee un rendimiento de hasta 3 veces más veloz que la Raspberry pi 4. Otra de sus características es que nos ofrece un poder de procesamiento similar al de una PC de escritorio básica. Este modelo está equipado con 4 u 8 GB de RAM, hp además que es una herramienta ideal para aprender a programar, desarrollar proyectos de electrónica y automatización industrial, entre otros **[18].**

Figuras 8 Raspberry Pi 5

3.8.2. Sistemas operativos y sus principales orígenes

3.8.2.1.Raspbian

Este sistema operativo basado en Debian, la distribución Linux hoy ha sido optimizada para la Raspberry Pi y se puede descargar desde raspberry.org. Al igual que Debían, cuenta con una amplia comunidad de usuarios. Raspbian emplea LXDE como entorno gráfico y Midori como navegador web, e incorpora herramientas de desarrollo como IDLE para Python y Scratch, lo que la convierte en una excelente opción para aprender programación. **[17].**



Figuras 9 Raspbian

3.8.2.2.Ubuntu

Nacido del proyecto Debian, Ubuntu es una distribución Linux hoy de código abierto que ha ganado gran popularidad. Su nombre, de raíces africanas, refleja su filosofía de compartir y colaborar. Con un enfoque en la seguridad, la velocidad y la facilidad de uso, Ubuntu hp se ha posicionado como una alternativa confiable tanto para usuarios domésticos como para entornos empresariales **[17].**

Figuras 10 Ubuntu

3.8.2.3.OPENELEC Y OSMC

Es un sistema operativo versátil diseñado tanto para computadoras de placa reducida como Raspberry Pi, así también para ordenadores personales **[17].**

Figuras 11 OPENELEC Y OSMC

3.9.LIBRERÍAS

3.9.1. OpenCV:

Realiza captura y procesamiento de imágenes.

3.9.2. NumPy:

Tiene como función la manipulación de matrices.

3.9.3. TensorFlow:

Su principal funcionamiento es cargar y ejecutar el modelo de clasificación.

3.9.4. Sounddevice:

Permite la reproducción de audio.

3.9.5. RPi.GPIO:

Tiene como función el control de pines GPIO para encender el LED.

3.9.6. Pillow:

Esta librería puede realizar manipulación de imágenes.

3.9.7. scipy:

Funciones matemáticas y de procesamiento de imágenes adicionales.

4. MARCO METODOLÓGICO

4.1.COMPONENTES DEL PROTOTIPO

Para la implementación del este prototipo se optó por una Raspberry pi 5 conectada a una pantalla HDMI de 7 pulgadas, una cámara USB 3.0 con autoenfoque y un módulo de parlante para la reproducción de sonidos de alerta. Tal como se muestra en la figura 12.

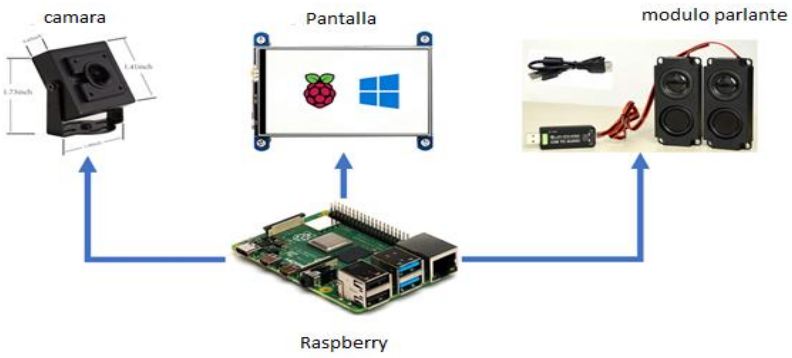


Figura 12 Componentes del prototipo

4.2.DIAGRAMAS DE CONEXIÓN DEL PROTOTIPO

En la figura 13 se explica las conexiones del prototipo desde el componente principal que es la computadora Raspberry Pi 5, la cual fue conectada vía USB a la tarjeta de sonido y a la cámara web, por otro lado, el puerto HDMI fue conectado a la pantalla de 7 pulgadas en cual podemos ejecutar el prototipo, todo esto siendo alimentado desde la Raspberry Pi 5 que se conecta a un cargador cana kit de 45w

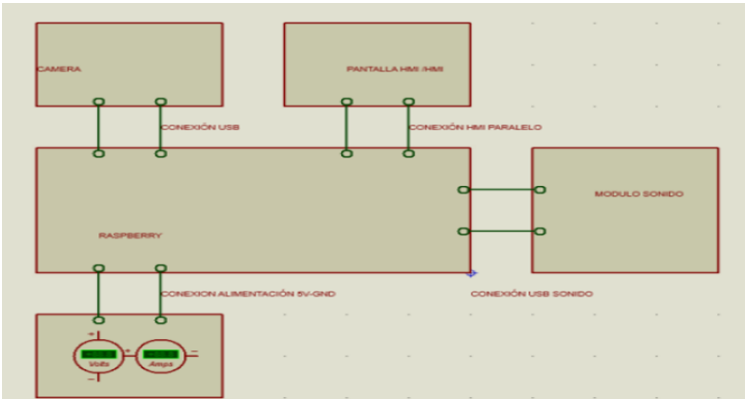


Figura 13 Diagramas de conexión del prototipo

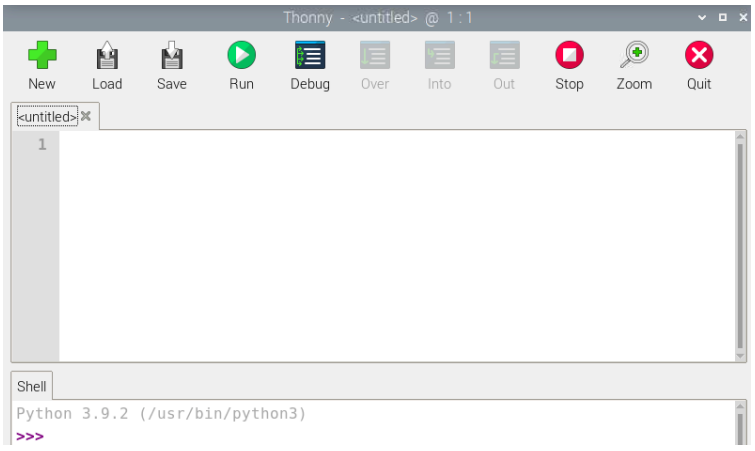
4.3.FORMATEO DE LA TARJETA E INSTALACIÓN DEL SISTEMA OPERATIVO

El paso inicial es instalar el sistema operativo mediante la herramienta Raspberry pi, tal como se muestra en la Figura 14.

Figuras 14 Raspberry pi imager

4.4.ENTORNO DE PROGRAMACIÓN

Para el procesamiento del prototipo se usó el entorno de programación Thonny Python, que es un componente ya instalado en el sistema raspbian con una versión de Python 3.9.2. tal como se muestra en la figura 15.



Figuras 15 Thonny Python

4.5.INSTALACIÓN DE DRIVER

Al empezar el sistema de la Raspberry pi es necesario instalar los componentes siguiendo los siguientes pasos:

4.5.1. Instalación de Pip para python3

El comando sudo apt-get install python3 python3-pip que se muestra en la figura 16 es utilizado en sistemas operativos basados en Linux, como Ubuntu, para instalar Python 3 y Pip para Python 3.

```
sudo apt-get install python3 python3-pip
```

Figuras 16 Pi imager

- **sudo:** Este prefijo se utiliza para otorgar privilegios de super-usuario al comando que sigue. Es necesario porque la instalación de software en la mayoría de las áreas del sistema operativo requiere permisos elevados.

- **apt-get:** Es una herramienta de línea de comandos utilizada para manejar paquetes en sistemas basados en Debian (como Ubuntu). Permite instalar, actualizar y eliminar paquetes.

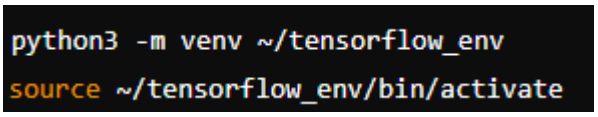
- **install**: Este es el comando que indica a apt-get que se desea instalar uno o más paquetes.

- **python3:** Este es el paquete de Python versión 3, instalar este paquete asegura que tener la versión más reciente de Python 3 disponible en los repositorios del sistema.

- **python3-pip:** Pip es el gestor de paquetes para Python, que facilita la instalación de bibliotecas y herramientas adicionales. El paquete python3-pip instala Pip específicamente para Python 3.

4.6.INSTALACIÓN ENTORNO VIRTUAL

El comando python3 -m venv ~/tensorflow_env y la secuencia de activación del entorno virtual source ~/tensorflow_env/bin/activate son partes esenciales para manejar proyectos de Python de forma aislada. tal como se muestra en la figura 17.



Figuras 17 Entorno virtual

- **python3 -m venv ~/tensorflow_env**

Este comando se utilizó para crear un entorno virtual en Python. Un entorno virtual es una instalación independiente y aislada de Python, lo cual permito trabajar en proyectos específicos sin afectar las instalaciones globales de Python y sin que otros proyectos interfieran entre sí.

- **python3: Especifica que estás utilizando la versión 3 de Python.**

-m venv: Le dice a Python que utilice el módulo venv, que es el módulo estándar para crear entornos virtuales.

~/tensorflow_env: Especifica la ubicación y el nombre del entorno virtual. En este caso tensorflow_env fue el nombre asignado del entorno virtual de este prototipo, y se crea en el directorio home del usuario (~) de la raspberry pi 5.

Al ejecutar este comando, se creó una carpeta llamada tensorflow_env en el directorio home. Esta carpeta contiene una copia independiente de Python y Pip, permitiendo instalar paquetes de manera aislada.

- **source ~/tensorflow_env/bin/activate**

Una vez creado el entorno virtual, se procedió a activarlo con el comando ~/tensorflow_env/bin/activate para poder utilizarlo.

Activar el entorno virtual ajusta las variables del entorno para que cualquier operación con Python y Pip se realice con las versiones contenidas en el entorno virtual.

4.7.INSTALACIÓN DE LIBRERÍA DE PROCESAMIENTO

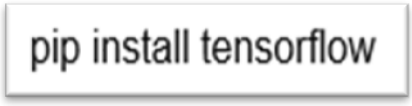
El comando sudo apt-get install build-essential libhdf5-dev libhdf5-serial-dev libhdf5-103 python3-dev gfortran que se muestra en la figura 18 en una Raspberry Pi o en cualquier sistema basado en Linux instala varios paquetes esenciales para la compilación y el desarrollo de software. build-essential incluye herramientas básicas de desarrollo como GCC (GNU Compiler Collection) y make. libhdf5-dev y libhdf5-serial-dev proporcionan bibliotecas de desarrollo para HDF5, un formato de datos utilizado para almacenar y organizar grandes cantidades de datos. libhdf5-103 es la biblioteca en sí. python3-dev instala los archivos de cabecera y una biblioteca necesaria para el desarrollo de módulos de Python. Finalmente, gfortran es el compilador de Fortran de GNU, utilizado para compilar programas escritos en el lenguaje Fortran. Estos paquetes son necesarios para compilar y desarrollar software, especialmente en proyectos que involucren el procesamiento y análisis de datos científicos o numéricos.

Figuras 18 Comandos para librerías de procesamiento

La figura 19 muestra el comando pip install numpy h5py que instala dos paquetes esenciales para Python: NumPy y h5py. NumPy proporciona soporte para arrays y matrices multidimensionales, junto con funciones matemáticas avanzadas, siendo fundamental para la computación científica y el análisis de datos. h5py permite interactuar con archivos HDF5, un formato utilizado para almacenar grandes volúmenes de datos, ofreciendo una interfaz eficiente para leer y escribir estos archivos desde Python. Ambos paquetes son cruciales para proyectos de análisis de datos, aprendizaje automático y aplicaciones científicas que requieren manejo y procesamiento de grandes datasets.

Figuras 19 Comandos para librerías de procesamiento

El comando pip install tensorflow como se muestra en la figura 20 instaló TensorFlow, que es una biblioteca de código abierto desarrollada por Google para el aprendizaje automático y la inteligencia artificial.



Figuras 20 Comandos para librerías de procesamiento

4.8.CAPTURA DE IMAGEN Y PROCESAMIENTO DE IMAGEN A ESCALA DE GRISES

El proceso de captura y procesamiento de imágenes a escala de grises en Teachable Machine resultó más preciso en ciertos contextos debido a algunas razones técnicas y prácticas las cuales son:

- **Reducción de la Complejidad del Modelo:**

Las imágenes en color contienen tres canales (rojo, verde y azul), lo que añade complejidad al modelo de aprendizaje automático. Al convertir las imágenes a escala de grises, se redujo el número de canales a uno, simplificando el modelo. Esta simplificación llevo a una mejora en la precisión del modelo, ya que hay menos parámetros a ajustar y menos información redundante a procesar.

- **Eliminación de Información Irrelevante:**

En muchos casos, el color puede no ser relevante para la tarea de clasificación. Por ejemplo, al clasificar formas geométricas o ciertas características estructurales, el color puede introducir ruido innecesario. Convertir las imágenes a escala de grises elimina esta información irrelevante, permitiendo que el modelo se enfoque en las características más importantes y distintivas de las imágenes.

- **Mejora en la Generalización:**

La escala de grises mejora la capacidad del modelo para generalizar nuevas imágenes que no ha visto antes. Los modelos entrenados con imágenes en color pueden ser más susceptibles a cambios en las condiciones de iluminación y variaciones de color, mientras que las imágenes en escala de grises son más robustas frente a estos cambios, ayudando al modelo a generalizar mejor.

- **Reducción del Ruido y Mejora de la Nitidez:**

Las imágenes en color pueden contener variaciones de color y sombras que no son relevantes para la tarea de clasificación. Convertir las imágenes a escala de grises ayuda a reducir este tipo de ruido, haciendo que las características importantes sean más prominentes y fáciles de detectar para el modelo.

- **Ahorro de Recursos Computacionales:**

El procesamiento de imágenes en escala de grises requiere menos recursos computacionales en comparación con las imágenes en color. Esto significa que los modelos se entrenan rápidamente y con menos consumo de memoria y poder de procesamiento, permitiendo experimentar con más configuraciones y ajustar los hiperparámetros de manera más eficiente.

4.9.PROGRAMACIÓN PARA CAPTURA DE IMÁGENES

Se detalla un script en la figura 21 en Python diseñado para la captura y procesamiento de imágenes en tiempo real utilizando la biblioteca OpenCV.

Este script permite interactuar con una cámara USB para capturar imágenes, mostrarlas en una ventana y guardar versiones redimensionadas y convertidas a escala de grises.

El script comienza con la importación de las bibliotecas esenciales: cv2 (OpenCV) para el manejo de imágenes y video para la gestión de marcas temporales.

Posteriormente, se inicializa la cámara USB mediante el comando cv2.VideoCapture(0), verificando su disponibilidad. Si la cámara no está accesible, el programa imprime un mensaje de error y finaliza su ejecución.

Una vez verificada la conexión con la cámara, se crea una ventana denominada "Frame" para visualizar las imágenes capturadas en tiempo real. El flujo principal del script se ejecuta dentro de un bucle while que captura imágenes continuamente y las muestra en la ventana creada. El bucle permite interactuar con el sistema mediante el teclado: presionando 'q' para salir del programa o 's' para iniciar la captura y almacenamiento de imágenes.

Cuando se activa el modo de guardado de imágenes (presionando 's'), el script redimensiona las imágenes capturadas a 96x96 píxeles y las convierte a escala de grises.

Ambas versiones de la imagen, tanto la original redimensionada como la versión en escala de grises, se guardan en el disco con nombres únicos generados a partir de la marca de tiempo actual, evitando así la sobre escritura de archivos anteriores. El script concluye liberando los recursos de

la cámara y cerrando todas las ventanas de OpenCV para asegurar una finalización limpia del proceso.

```python
import cv2
import time
# Inicializar la cámara USB
cap = cv2.VideoCapture(0)
if not cap.isOpened():
    print("Error: No se puede acceder a la cámara.")
    exit()
# Crear una ventana para mostrar la imagen
cv2.namedWindow("Frame")
saving = False
while True:
    # Capturar imagen en tiempo real
    ret, frame = cap.read()
    if not ret:
        print("Error: No se puede recibir la imagen (stream end?). Exiting ...")
        break
    # Mostrar la imagen en tiempo real
    cv2.imshow("Frame", frame)
    key = cv2.waitKey(1) & 0xFF
    # Si se presiona la tecla 'q', salir del bucle
    if key == ord('q'):
        break
    # Si se presiona la tecla 's', comenzar a guardar las imágenes
    if key == ord('s'):
        saving = True
        print("Comenzando a guardar imágenes...")
    # Si estamos en modo de guardar imágenes, procesar y guardar
    if saving:
        # Redimensionar la imagen a 96x96 píxeles
        resized_image = cv2.resize(frame, (96, 96))
        # Convertir la imagen redimensionada a escala de grises
        gray_image = cv2.cvtColor(resized_image, cv2.COLOR_BGR2GRAY)
        # Guardar las imágenes con un nombre basado en la hora actual para evitar sobrescritura
        timestamp = int(time.time() * 1000)
        cv2.imwrite(f"resized_image_{timestamp}.png", resized_image)
        cv2.imwrite(f"gray_image_{timestamp}.png", gray_image)
        print(f"Imágenes guardadas como resized_image_{timestamp}.png y gray_image_{timestamp}.png")
# Liberar los recursos
cap.release()
cv2.destroyAllWindows()
```

Figuras 21 Scrip de programación en phyton para captura de imágenes

4.10. CAPTURA DE IMÁGENES A COLOR Y EN ESCALA DE GRISES

En la Figura 22 se presentan las imágenes capturadas en color y su posterior procesamiento para convertirlas en escala de grises. Este procesamiento es crucial para la clasificación de las imágenes cuando los sujetos llevan casco.

Figuras 22 Captura de imágenes de motociclistas con casco en escala de grises

De igual manera, en la Figura 23 se muestran las imágenes capturadas en color y su procesamiento para convertirlas en escala de grises, pero en este caso para la clasificación de las imágenes cuando los sujetos no llevan casco. Este proceso de conversión es importante, asegura que todas las imágenes utilizadas para entrenar al modelo sean uniformes, mejorando así su capacidad para aprender y distinguir entre las dos clases: con casco y sin casco.

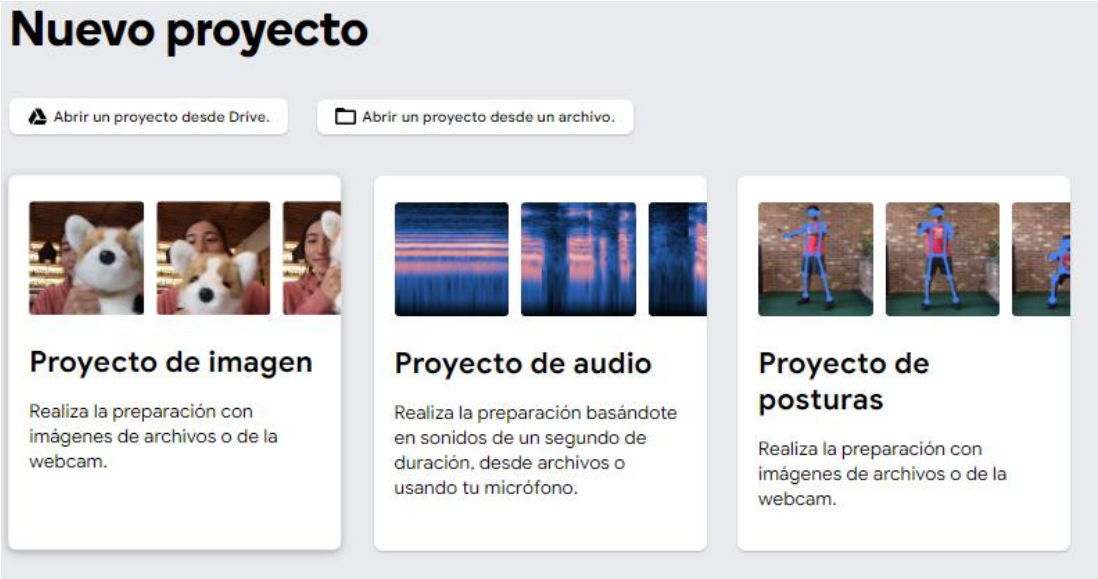
Figuras 23 Captura de imágenes de motociclistas sin casco en escala de grises

Estas conversiones y capturas de imágenes cuando uno de los dos ocupantes de una motocicleta no lleva casco como se muestra en la figura 24 son pasos esenciales en la preparación del conjunto de datos para el modelo de clasificación. Al asegurar que todas las imágenes se procesen de la misma manera, se mejora la calidad del entrenamiento del modelo, lo que resulta en una clasificación más precisa y robusta. El uso de imágenes en escala de grises también puede reducir el tiempo de procesamiento y los recursos computacionales necesarios, haciendo el proceso de entrenamiento más eficiente.

Figuras 24 Captura de imágenes en escala de grises cuando uno de los dos ocupantes de una motocicleta no tiene casco

4.11. MODELO DE ENTRENAMIENTO TEACHABLE MACHINE

Para empezar un entrenamiento se abre Teachable Machine en un navegador web, se selecciona el tipo de proyecto que se desea crear, los cuales pueden ser: clasificación de imágenes, sonidos o poses tal como se muestra en la figura 25. Para este proyecto se usó el modelo de clasificación de imágenes.



Figuras 25 Tipos de proyectos en Teachable Machine

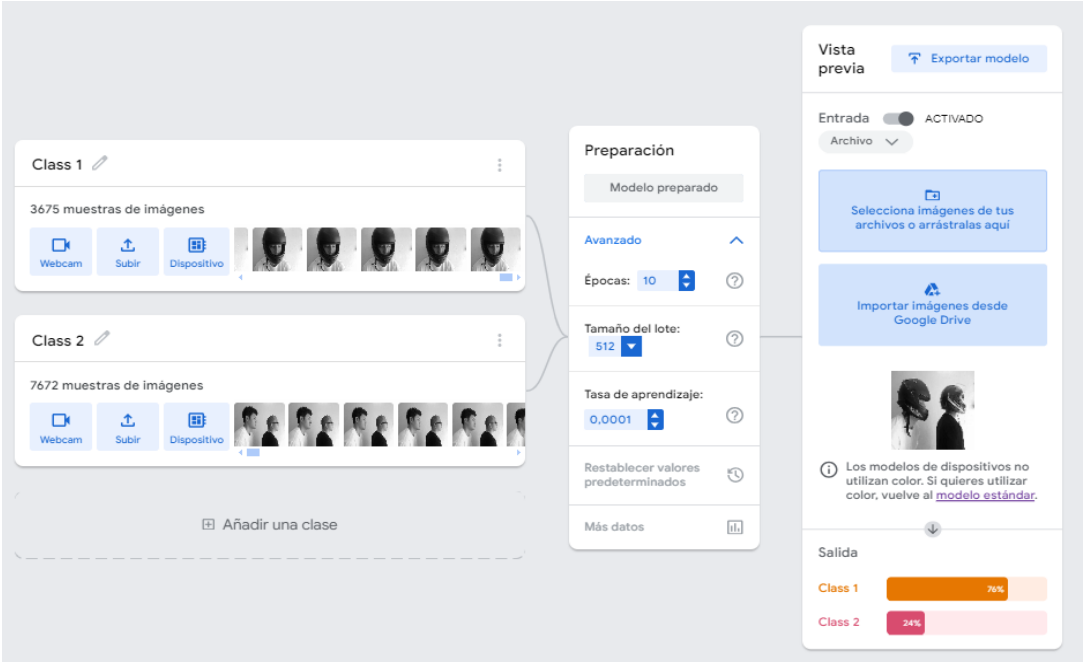
Una vez seleccionado el proyecto se cargan los datos. Se permite subir imágenes desde la computadora o utilizar la cámara web para capturar imágenes directamente en la plataforma, es importante tener varias imágenes por clase para asegurar un entrenamiento efectivo, organizar las imágenes en diferentes clases o categorías, por ejemplo, si se está clasificando una persona que tiene un casco o no.

Con los datos cargados y etiquetados, se procede a entrenar el modelo, se ajusta las configuraciones de entrenamiento si es necesario, generalmente las configuraciones predeterminadas son suficientes para la mayoría de los proyectos básicos. Luego, con un clic en el botón "Entrenar Modelo Insertado". Tal como se muestra en la figura 26. La plataforma procesa las imágenes para entrenar un modelo de aprendizaje automático, este proceso que puede tardar desde unos minutos hasta varias horas, dependiendo del tamaño del dataset.

Figuras 26 Toma de imágenes Teachable Machine

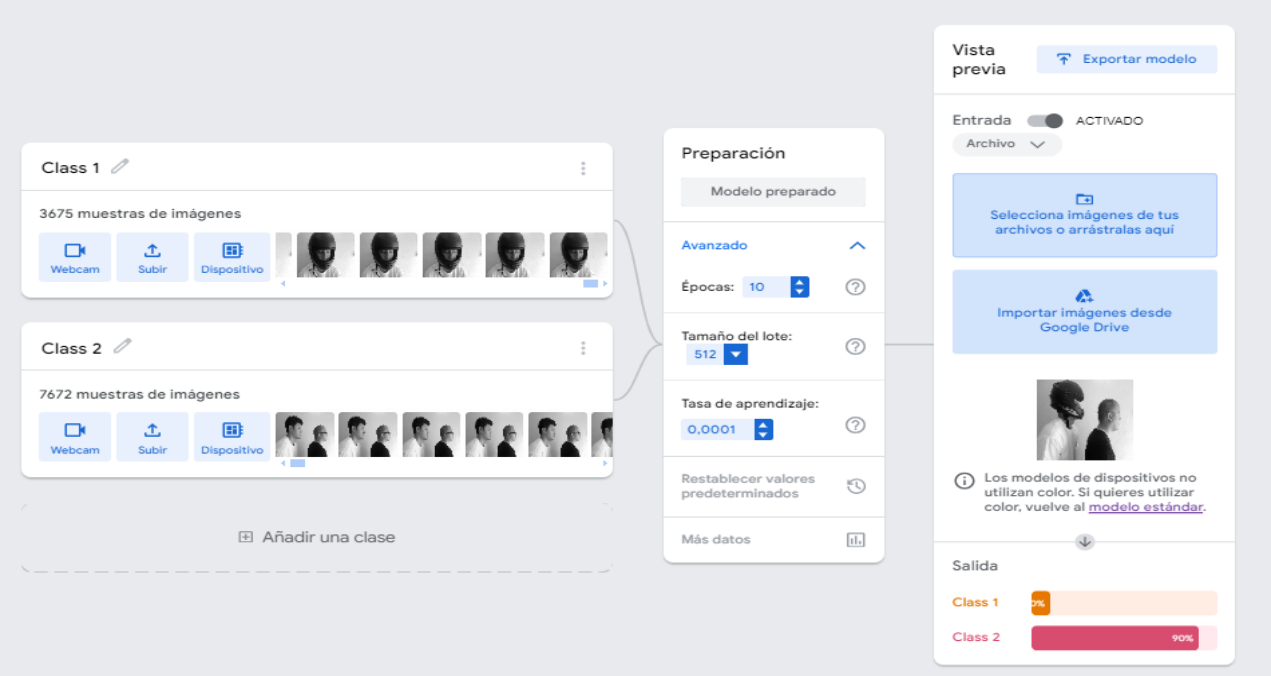
En la figura 127 muestra una plataforma de entrenamiento de un modelo de clasificación de imágenes, con dos clases: "Class 1" con 3675 muestras y "Class 2" con 7672 muestras. Se pueden agregar imágenes desde una webcam, archivos locales o un dispositivo. En la sección de preparación del modelo, se configuran 10 épocas, un tamaño de lote de 512 y una tasa de aprendizaje de 0.0001. La vista previa permite cargar imágenes desde archivos locales o Google Drive, y el modelo, activado para entrada de archivo, muestra las predicciones con 76% para "Class 1" y 24% para "Class 2".

Figuras 27 Entrenamiento del modelo en Teachable Machine



La interfaz de la figura 28 muestra una plataforma para entrenar un modelo de clasificación de imágenes con dos clases: "Class 1" con 3675 muestras y "Class 2" con 7672 muestras. Se pueden agregar imágenes desde una webcam, archivos locales o un dispositivo. La preparación del modelo está configurada con 10 épocas, un tamaño de lote de 512 y una tasa de aprendizaje de 0.0001. La vista previa permite cargar imágenes desde archivos locales o Google Drive, y el modelo, activado para entrada de archivo, muestra las predicciones con 1% para "Class 1" y 90% para "Class 2"

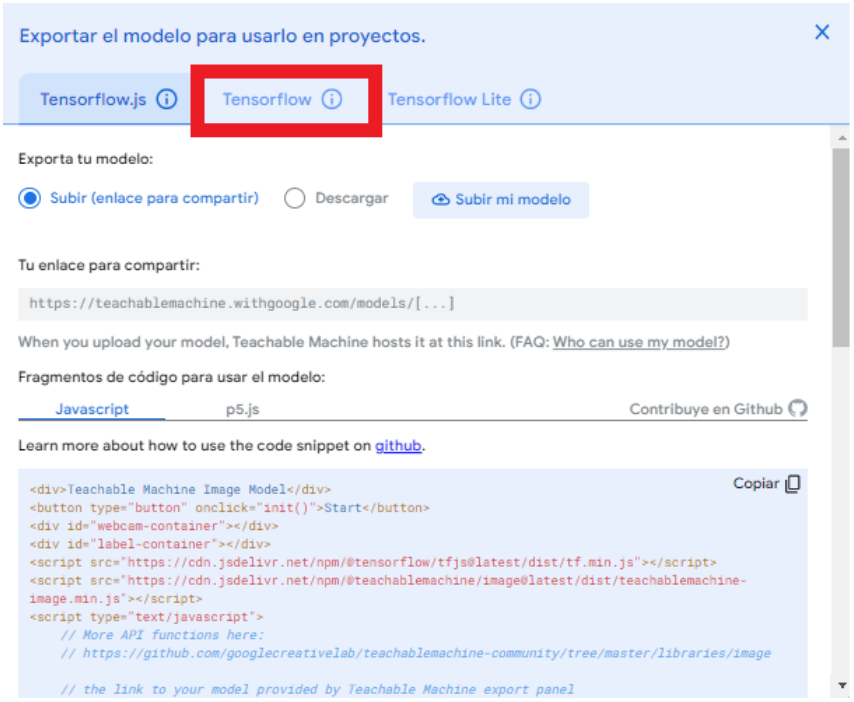
Figuras 28 Comprobación de clases mediante Teachable Machine



Una vez completado el entrenamiento, Teachable Machine muestra una vista previa del modelo en acción. Se puede probar el modelo cargando nuevas imágenes o utilizando la cámara web para ver cómo clasifica en tiempo real. Esta etapa es crucial para evaluar el rendimiento del modelo y hacer ajustes si es necesario. tal como se muestra en la figura 29.

Figuras 29 Vista previa modelo de acción Teachable Machine

Finalmente, Teachable Machine permite exportar el modelo entrenado de varias formas, incluyendo TensorFlow.js para integraciones web y TensorFlow Lite para aplicaciones móviles. Para este proyecto se descarga el modelo TensorFlow. Tal como se muestra en la figura 30, para poder integrarlo al dispositivo móvil raspberry pi 5 y poder clasificar imágenes móviles



Figuras 30 Exportación del modelo en Teachable Machine

4.12. PROGRAMACIÓN PARA CLASIFICACIÓN Y DETECCIÓN

El siguiente script en Python que se muestra en la figura 31 fue diseñado para capturar y procesar imágenes en tiempo real utilizando la biblioteca OpenCV. Además, permite guardar las imágenes capturadas tanto en formato original redimensionado como en escala de grises en una unidad USB específica. A continuación, se detalla cada sección del script.

Primero, se importan las bibliotecas necesarias: cv2 para el manejo de imágenes y video, time para gestionar marcas temporales y para interactuar con el sistema operativo y manejar archivos. Luego, se inicializa la cámara USB mediante el comando cv2.VideoCapture(0), y se verifica si la cámara está accesible. Si no se puede acceder a la cámara, se imprime un mensaje de error y el programa termina.

Se crea una ventana llamada "Frame" para visualizar las imágenes capturadas en tiempo real, redimensionando la ventana a un tamaño de 320x240 píxeles para un manejo más cómodo. Se define una variable saving para controlar el estado de guardado de imágenes y se especifica la ruta del directorio USB donde se guardan las imágenes, asegurándose de que dicho directorio existe. Si no se encuentra la unidad USB, el programa imprime un mensaje de error y finaliza.

El flujo principal del script se ejecuta dentro de un bucle while que captura imágenes continuamente y las muestra en la ventana creada. El tamaño de las imágenes mostradas se ajusta a 320x240 píxeles. El programa permite la interacción a través del teclado: presionando 'q' para salir del bucle y terminar la ejecución, o 's' para iniciar el proceso de guardado de imágenes.

Cuando el modo de guardado de imágenes está activado (presionando 's'), el script redimensiona las imágenes capturadas a 96x96 píxeles y las convierte a escala de grises. Ambas versiones de la imagen se guardan en la unidad USB con nombres únicos generados a partir de la marca de tiempo actual, evitando así la sobreescritura de archivos anteriores. Además, se asegura la sincronización de los datos para que se escriban correctamente en la unidad USB mediante el comando os.sync().

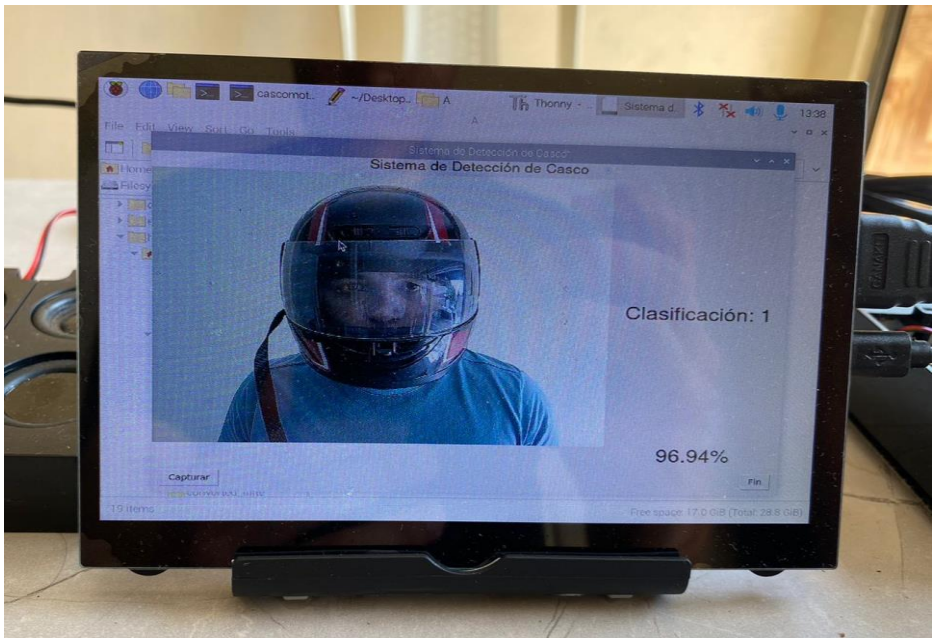
Finalmente, se liberan los recursos de la cámara y se cierran todas las ventanas de OpenCV utilizando cap.release() y cv2.destroyAllWindows(), y se realiza una sincronización final de los datos en la unidad USB para asegurar que todas las imágenes se hayan guardado correctamente. Este script proporciona una solución completa y segura para la captura y almacenamiento de imágenes en tiempo real en un dispositivo de almacenamiento externo.

```python
import cv2
import numpy as np
import tensorflow as tf
import time
cap = cv2.VideoCapture(0)
if not cap.isOpened():
    print("Error: No se puede acceder a la cámara.")
    exit()
cv2.namedWindow("Frame", cv2.WINDOW_NORMAL)
cv2.resizeWindow("Frame", 320, 240)
model_path = '/home/cascomoto/Desktop/converted_tflite/vww_96_grayscale_quantized.tflite'
interpreter = tf.lite.Interpreter(model_path=model_path)
interpreter.allocate_tensors()
input_details = interpreter.get_input_details()
output_details = interpreter.get_output_details()
def classify_image(image):
    image = cv2.resize(image, (96, 96))
    image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    image = np.expand_dims(image, axis=-1)
    image = np.expand_dims(image, axis=0)
    image = image.astype(np.float32) / 255.0
    interpreter.set_tensor(input_details[0]['index'], image)
    interpreter.invoke()
    output_data = interpreter.get_tensor(output_details[0]['index'])
    return np.argmax(output_data)
classifying = False
while True:
    ret, frame = cap.read()
    if not ret:
        print("Error: No se puede recibir la imagen (stream end?). Exiting ...")
        break
    display_frame = cv2.resize(frame, (320, 240))
    cv2.imshow("Frame", display_frame)
    key = cv2.waitKey(1) & 0xFF
    if key == ord('q'):
        break
    if key == ord('c'):
        classifying = True
        print("Comenzando clasificación continua...")
    if classifying:
        classification = classify_image(frame)
        print(f"Clasificación: {classification}")
cap.release()
cv2.destroyAllWindows()
```

Figuras 31 Programación para clasificación y detección

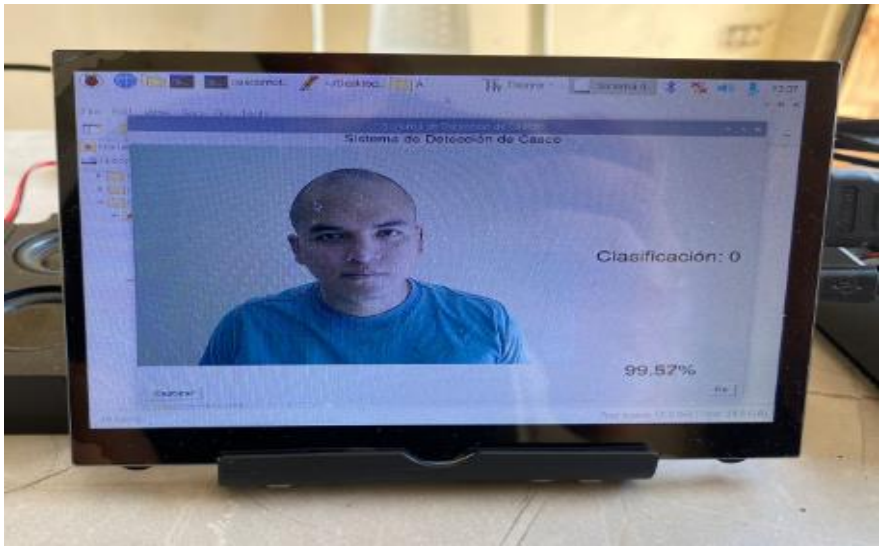
5. RESULTADOS

5.1.RESULTADOS DEL PROTOTIPO EN DETECCIONES DESDE UN ANGULO FRONTAL



Figuras 32 Detección frontal con casco del prototipo en el interior de un inmueble

Al momento de realizar la captura de imagen en el prototipo, se presionó el botón de capturar ubicado en la parte inferior izquierda, en la ejecución se observó que la imagen capturada por la cámara detectó como clasificación 1 que significa ¨motociclista con casco¨, arrojando 96.94% de exactitud, ya que el individuo se encuentra correctamente centrado en el enfoque de la cámara y con buena iluminación, así como se muestra en la figura 32.



Figuras 33 Detección frontal sin casco del prototipo en el interior de un inmueble

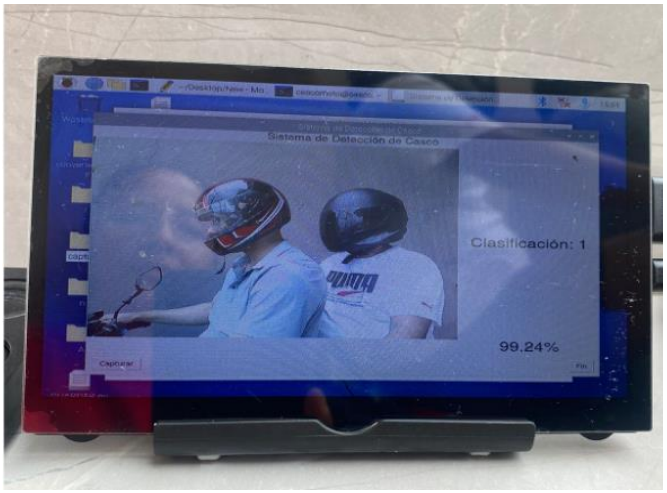
En la imagen 33 se muestra que la imagen capturada por la cámara detectó como clasificación 0 la cual representa a ¨motociclista sin casco¨, dando como resultado un 99.52% de exactitud, debido a que el sujeto está ubicado y centrado correctamente en el enfoque de la cámara y el ambiente también se encuentra con buena iluminación.

Figuras 34 Detección frontal en vía pública 2

Figuras 35 Detección frontal en vía pública 1

En la imagen 34 y 35 se muestra los resultados de la detección del prototipo con una motocicleta en la vía pública y se mostró que la imagen capturada por la cámara detecto como clasificación 1 que significa ¨motociclista con casco¨, arrojando 95.96% de exactitud, ya que esta prueba se realizó en el exterior a la luz del día.

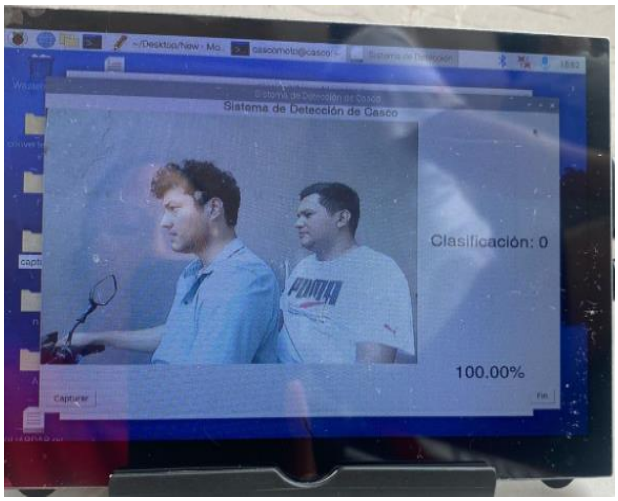
5.2.RESULTADOS DEL PROTOTIPO EN DETECCIONES DESDE UN ANGULO LATERAL



Figuras 36 Detección lateral casco en vida pública dos ocupantes 1

Figuras 37 Detección lateral con casco en vida pública dos ocupantes 2

En la imagen 36 y 37 se muestra que la imagen capturada por la cámara de dos motociclistas, piloto y copiloto, detecto como clasificación 1 la cual representa a “motociclistas con casco”, dando como resultado un 99.24% de exactitud, debido a que los sujetos están ubicados y centrados correctamente en el enfoque de la cámara y el ambiente también se encuentra con buena iluminación.



Figuras 38 Detección lateral sin casco en vía pública dos ocupantes 1

Figuras 39 Detección lateral sin casco en vía pública dos ocupantes 2

En las imágenes 38 y 39 se muestra los resultados de la detección del prototipo con una motocicleta en la vía pública, se mostró que la imagen capturada por la cámara detecto como clasificación 0 que significa ¨motociclistas sin casco¨, arrojando 100% de exactitud, ya que los ocupantes de la motocicleta se encuentran correctamente centrados en el enfoque de la cámara y a la luz del día.

6. CRONOGRAMA

A continuación, se muestra en la tabla 1 que detalla el cronograma de actividades a desarrollar en el diseño e implementación del prototipo del proyecto, así como el tiempo que toma cada una de estas actividades para su presentación.

Tabla 1 Cronograma

N.	TAREAS POR REALIZAR	ABRIL			MAYO				JUNIO				JULIO			
		S 2	S 3	S 4	S 1	S 2	S 3	S 4	S 1	S 2	S 3	S 4	S 1	S 2	S 3	S 4
1	Recolección de imágenes															
2	Entrenamiento Machine learning y obtención de Modelos															
3	Implementación dentro de computadora raspberry															
4	Pruebas del correcto funcionamiento de detecciones de uso de cascos en motociclistas															

7. PRESUPUESTO

En el prototipo se implementará una lista de algunos componentes necesarios para el correcto funcionamiento en la detección de cascos en motociclistas. A continuación, se presenta la tabla 2 con el presupuesto utilizado.

Tabla 2 Presupuesto

DESCRIPCIÓN	PRECIO UNITARIO	CANTIDAD	TOTAL
Kit Raspberry pi 5	$190	1	$190
Cámara	$60	1	$60
Horas ingeniería	$2.60	40	$104
Pantalla HDMI	$60	1	$60
Modulo parlante	$30	1	$30
Total			**$444**

8. CONCLUSIONES

- La combinación de Python, OpenCV y TensorFlow ha demostrado ser una estrategia eficaz para el desarrollo del sistema de detección de casco. Python, con su amplia variedad de bibliotecas y su simplicidad de uso, ha facilitado la integración de OpenCV para la captura y preprocesamiento de imágenes, así como de TensorFlow para la construcción y entrenamiento del modelo de clasificación. Esta sinergia ha permitido crear un sistema robusto y eficiente.

- El uso de OpenCV ha sido crucial para el preprocesamiento de imágenes, incluyendo la conversión de imágenes a escala de grises y el ajuste del tamaño de estas. Estas operaciones han optimizado el rendimiento del modelo de TensorFlow, asegurando que las imágenes de entrada sean uniformes y adecuadas para el entrenamiento y la inferencia, lo que ha resultado en una mayor precisión y velocidad en la clasificación.

- La implementación de una interfaz de usuario en Python ha permitido una interacción sencilla y eficaz con el sistema de detección de casco. La interfaz facilita la captura de imágenes en tiempo real y la visualización de resultados de clasificación, haciendo que el sistema sea accesible incluso para usuarios con conocimientos técnicos limitados. La simplicidad y funcionalidad de la interfaz han sido claves para la usabilidad del sistema.

- Gracias a TensorFlow, se ha logrado entrenar un modelo de clasificación confiable y preciso. La capacidad de TensorFlow para manejar grandes conjuntos de datos y su flexibilidad en la construcción de modelos han permitido desarrollar un clasificador capaz de distinguir eficazmente entre imágenes de personas con y sin casco. La implementación de técnicas de ajuste de hiperparámetros y validación cruzada ha mejorado aún más la efectividad del modelo.

- La estructura modular del proyecto, utilizando Python, OpenCV y TensorFlow, facilita el mantenimiento y la actualización del sistema. La capacidad de agregar nuevos datos de entrenamiento, ajustar parámetros del modelo y actualizar componentes de la interfaz asegura que el sistema pueda evolucionar y adaptarse a nuevas necesidades y condiciones

operativas. Esta flexibilidad y sostenibilidad son fundamentales para el éxito a largo plazo del sistema de detección de casco.

9. RECOMENDACIONES

- Se recomienda realizar una optimización continua del modelo de clasificación mediante el ajuste de hiper parámetros y la incorporación de nuevas técnicas de aprendizaje automático. La implementación de métodos como la validación cruzada y la búsqueda de hiper parámetros puede mejorar la precisión del modelo, reduciendo el riesgo de sobreajuste y asegurando un rendimiento consistente en diferentes conjuntos de datos.

- Es crucial ampliar el conjunto de datos utilizado para entrenar el modelo, incorporando imágenes adicionales de diversas condiciones de iluminación, ángulos y entornos. Esto aumentará la robustez del modelo y su capacidad para generalizar mejor en situaciones del mundo real, garantizando una mayor fiabilidad en la detección de cascos en diferentes contextos.

- Se sugiere realizar pruebas con usuarios finales para identificar áreas de mejora y hacer ajustes necesarios, como la simplificación de la navegación, la clarificación de las instrucciones y la mejora de la visualización de resultados.

- Una vez implementado, es fundamental establecer un plan de monitoreo y mantenimiento regular del sistema. Esto incluye la actualización del modelo con nuevos datos, la revisión de logs para detectar posibles errores, y la realización de auditorías periódicas para asegurar que el sistema sigue funcionando de manera eficiente y precisa, adaptándose a posibles cambios en el entorno operativo.

10. REFERENCIAS BIBLIOGRAFÍAS

[1] OPS/OMS, Organización Panamericana de la Salud. Seguridad vial [Internet]. 2024 [citado 23 de julio de 2024]. Disponible en: https://www.paho.org/es/temas/seguridad-vial

[2] Servicio Integrado de Seguridad ECU 911. En 2023, al 9-1-1 se han reportado 32.687 accidentes de tránsito con motos [Internet]. [citado 23 de julio de 2024]. Disponible en: https://www.ecu911.gob.ec/en-2023-al-9-1-1-se-han-reportado-32-687-accidentes-de-transito-con-motos/

[3] OMS. Global status report on road safety 2023 [Internet]. [citado 29 de julio de 2024]. Disponible en: https://www.who.int/publications/i/item/9789240086517

[4] Organización Mundial de la Salud. A pesar de los notorios progresos, la seguridad vial sigue siendo un problema apremiante para el mundo [Internet]. [citado 29 de julio de 2024]. Disponible en: https://www.who.int/es/news/item/13-12-2023-despite-notable-progress-road-safety-remains-urgent-global-issue

[5] Berrones-Sanz LD. Análisis de los accidentes y las lesiones de los motociclistas en México. GMM [Internet]. 5 de diciembre de 2017 [citado 25 de julio de 2024];153(6):87. Disponible en: http://gacetamedicademexico.com/frame_esp.php?id=61

[6] Montoya Sanabria SM, Rodríguez Hernández JM, Albavera Hernández C, Valero Alvarado OM. Evidencias para la prevención y control de lesiones en motociclistas. Revista Cubana de Salud Pública [Internet]. diciembre de 2016 [citado 25 de julio de 2024];42(4):0-0. Disponible en: http://scielo.sld.cu/scielo.php?script=sci_abstract&pid=S0864-34662016000400011&lng=es&nrm=iso&tlng=es

[7] Chen Cheng C, Chung E, Correa N. inteligencia Artificial y su Impacto en la Industria de la Ingeniería. REICT [Internet]. 4 de julio de 2023 [citado 21 de julio de 2024];3(1):26-40. Disponible en: https://revistas.up.ac.pa/index.php/REICIT/article/view/3948

[8] Corvalán JG. Inteligencia artificial: retos, desafíos y oportunidades - Prometea: la primera inteligencia artificial de Latinoamérica al servicio de la Justicia. Rev Investig Const [Internet]. abril de 2018 [citado 21 de julio de 2024];5:295-316. Disponible en: https://www.scielo.br/j/rinc/a/gCXJghPTyFXt9rfxH6Pw99C/

[9] Caparroso IO, Avilés O, Bello JH. Una introducción a la robótica industrial. Ciencia e Ingeniería Neogranadina [Internet]. 1 de junio de 1999 [citado 27 de julio de 2024];8:53-67. Disponible en: https://revistas.unimilitar.edu.co/index.php/rcin/article/view/1410

[10] Ruiz CA, Basualdo MS, Matich DJ. Redes Neuronales: Conceptos Básicos y Aplicaciones.

[11] Camargo B. C, Cortés R. J, Jiménez T. A. Implementación de sistemas digitales complejos utilizando sistemas embebidos. Ingenium [Internet]. 2012 [citado 21 de julio de 2024];13(25):5-15. Disponible en: https://dialnet.unirioja.es/servlet/articulo?codigo=5038478

[12] Armijos VA. SISTEMAS EMBEBIDOS. 2019 [citado 29 de julio de 2024]; Disponible en: https://www.academia.edu/43320402/_SISTEMAS_EMBEBIDOS_CAP%C3%8DTULO_1_Introducci%C3%B3n_

[13] Raul GD. PYTHON PARA TODOS [Internet]. ESPAÑA; 2011. 160 p. Disponible en: https://mundogeek.net/tutorial-python/

[14] Carney M, Webster B, Alvarado I, Phillips K, Howell N, Griffith J, et al. Teachable Machine: Approachable Web-Based Tool for Exploring Machine Learning Classification. En:

Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems [Internet]. New York, NY, USA: Association for Computing Machinery; 2020 [citado 29 de julio de 2024]. p. 1-8. (CHI EA '20). Disponible en: https://doi.org/10.1145/3334480.3382839

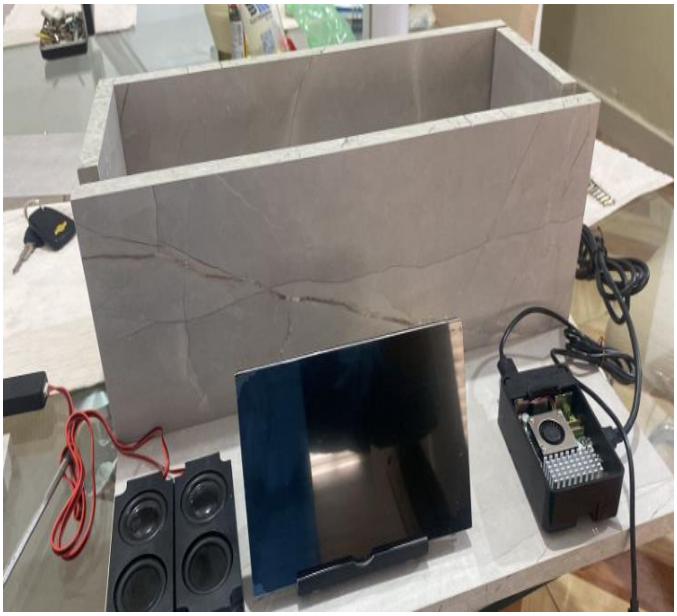
[15] Araujo Mena EM. Implementación de un sistema de video vigilancia para los exteriores de la UPS, mediante mini computadores y cámaras Raspberry PI. [Internet] [bachelorThesis]. 2015 [citado 29 de julio de 2024]. Disponible en: http://dspace.ups.edu.ec/handle/123456789/10379

[16] Vázquez-Bautista O. Raspberry Pi. Con-Ciencia Boletín Científico de la Escuela Preparatoria No 3 [Internet]. 5 de julio de 2022 [citado 27 de julio de 2024];9(18):36-40. Disponible en: https://repository.uaeh.edu.mx/revistas/index.php/prepa3/article/view/9464

[17] Vazquez LAA. Raspberry, un ordenador en una solo placa. RICT Revista de Investigación Científica, Tecnológica e Innovación [Internet]. 7 de marzo de 2023 [citado 27 de julio de 2024];1(1):11-6. Disponible en: https://revista.ccaitese.com/index.php/ridt/article/view/3

[18] Mohapmed Fezari, Al-Dahoud A. Raspberry Pi 5 : The new Raspberry Pi family with more computation power and AI integration. 2023 [citado 29 de julio de 2024]; Disponible en: https://rgdoi.net/10.13140/RG.2.2.13547.52009

11. ANEXOS



UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR

SALESIANOS DON BOSCO

Guayaquil, 05 de agosto del 2024

Ing. Orlando Barcia, Msc.

Director de Carrera de Electrónica y Automatización.

De mis consideraciones:

Yo, Vicente Peñaranda, portador de la cédula de ciudadanía No. 0916113426 tutor de trabajo de titulación **"DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO DE IDENTIFICACIÓN DE USO DE CASCO EN MOTOCICLISTAS."**, informo la calificación al Trabajo de Tituación de los estudiantes de la Malla Rediseño: DARWIN ENRIQUE GARCIA MUÑOZ.

Criterio	Descripción del criterio	Puntaje	Observaciones
Planteamiento e identificación del problema	Se muestra la importancia del problema y la contribución que se quiere alcanzar con el Proyecto técnico.	15	15
Revisión del marco teórico y fuentes de información	Este criterio estable la relación entre la revisión literaria y el problema a abordar en el Proyecto técnico, así como el adecuado nivel de exhaustividad en la revisión de las fuentes de información.	15	15
Contenido Metodológico	Se establecen con claridad y de manera estructurada las distintas fases, uso de métodos, herramientas, diseños, recursos, materiales, etc, para el desarrollo del Proyecto técnico y la propuesta de solución.	20	20
Funcionalidad	Permite evaluar el nivel de funcionalidad del trabajo desarrollado, tomando en cuenta los objetivos del mismo.	30	30
Presentación de Resultados	Se expresan o presentan los resultados alcanzados en el desarrollo del proyecto técnico y cómo se relacionan con el cumplimiento de los objetivos, el impacto y la innovación.	15	15
Conclusiones Recomendaciones	Este criterio establece la claridad con que el autor expone su posición y sus ideas respecto a las conclusiones y recomendaciones expresadas.	5	5
	PUNTAJE FINAL:	**100**	100

Por la atención que se sirva dar a la presente, quedo de usted muy agradecido.

Atentamente,

Ing. Vicente Peñaranda.

Docente Tutor.

SEDE GUAYAQUIL
Chambers # 227 y 5 de Junio • Campus Centenario • Casilla 09-01-47-52 • Teléfono: (593) 4 2590-630 Ext.: 4401