FACULTAD DE INGENIERÍAS Y CIENCIAS APLICADAS

Trabajo de fin de Carrera titulado:

Diseño de un prototipo de semáforo inteligente mediante sensórica y uso del internet de las cosas (IOT).

Realizado por:

Rodrigo Sebastián Muñoz Proaño

Director del proyecto:

M.Sc Gabriela Mancheno. Ing.

Trabajo de titulación previo a la obtención del título Pregrado en Ingeniería Mecatrónica

QUITO, Febrero del 2024

Declaración Juramentada

Yo, Rodrigo Sebastián Muñoz Proaño, ecuatoriano con Cédula de identidad 172060859-3, declaro bajo juramento la veracidad del trabajo aquí desarrollado es de mi autoría, que no ha sido previamente presentado para ningún grado o calificación profesional, y se basa en las referencias bibliográficas descritas en este documento.

A través de esta declaración, cedo los derechos de propiedad intelectual a la Universidad Internacional SEK, según lo establecido en la Ley de propiedad intelectual, reglamento y normativa institucional vigente.

Rodrigo Sebastián Muñoz Proaño

C.I: 1720608593

Declaración del director de tesis

Declaro haber dirigido este trabajo a través de reuniones periódicas con el estudiante, orientando sus conocimientos y competencias para un eficiente desarrollo del tema escogido y dando el cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación.

M.Sc María Gabriela Mancheno Falconi. Ing.

Los profesores informantes:

Ing. Diana Peralta

Ing. Jaime Molina

Después de revisar el trabajo presentado lo han calificado apto para su defensa oral ante un tribunal examinador

Ing. Diana Peralta

Ing. Jaime Molina

"Cuando escuché al docto astrónomo,
cuando me presentaron en columnas
las pruebas y guarismos,
cuando me mostraron las tablas y
diagramas para medir, sumar y dividir,
cuando escuché al astrónomo discurrir
con gran aplauso de la sala,
qué pronto me sentí inexplicablemente
hastiado, hasta que me escabullí de mi asiento y
me fui a caminar en el húmedo y místico aire nocturno,
mirando de rato en rato, en silencio perfecto a las estrellas."

-Walter Whitman Jr.

Dedicatoria

A mi madre Verónica, por ser mi inspiración y fortaleza. Tus enseñanzas me han llenado de preciosos valores que ahora me permiten cumplir un objetivo más.

A mi padre Rodrigo, por enseñarme el valor del trabajo, estudio y constancia, principios fundamentales del éxito. Tus sacrificios y esfuerzos son la base en la que podido construir todos mis logros. Este proyecto es un tributo a tu dedicación y una muestra de mí más profunda gratitud. Anhelo perpetuar nuestro nombre en la historia.

A mi hermano Mateo, por ser mi compañía en todo momento de mi vida, eres el motor que me impulsa a ser mejor persona siempre. Quiero que este símbolo de dedicación sea tu ejemplo.

A mi Abuela Betty, por ser mi apoyo en cada momento de mi vida.

A mi novia Mare, por su inagotable amor, por no soltar mi mano y siempre creer en mí. Motívame siempre, demostrémosle a todo el mundo de lo que somos capaces.

A mi bisabuela María Elena.

A mis primos, Vanessa, Esteban y a mi tía Silvana.

A cuatro personas que hoy no pueden acompañarme, pero que sus memorias están conmigo en todo momento. Mis tíos, Roberto, Diego, Marcelo y mi abuelo Luis Rodrigo.

Agradecimientos

A Dios, por haberme dado la salud y los medios para poder finalizar mis estudios profesionales.

A mi madre, por su comprensión, compañía y su infaltable e incondicional amor.

A mi padre, por darme las herramientas, valores y llenarme de inspiración para cumplir con este cometido.

A mi hermano, por ser mi motor, fortaleza y ganas de superarme. Sé que puedes ser mucho mejor que yo.

A mi abuela, por acompañarme en cada paso de este proceso. No hubiera sido posible sin tu ayuda.

A mi amigo Andrés Jara, quién estuvo presente en cada etapa de este largo camino.

A todos mis compañeros y amigos que fueron parte de este proceso. Joshua, Daniel, Henry, Mathias, Juan Diego, Fernando, Francisco.

A mi profesor Diego Bustamante por sus consejos en toda mi etapa estudiantil.

A mi tutora de tesis Gabriela por su ayuda y disposición para culminar este proyecto.

A todo el cuerpo docente de la Facultad de Ingeniería y ciencias aplicadas, por sus enseñanzas y profesionalismo a lo largo de mi estancia dentro de las aulas de la Universidad.

A don Edu, por todo el cariño que ha demostrado a los estudiantes en el día a día.

Vemos coronado el esfuerzo de un trayecto arduo, que hoy celebramos con victoria.

Resumen

El presente proyecto se enfocó en abordar la problemática de la congestión peatonal en el campus Miguel de Cervantes de la Universidad Internacional SEK, mediante la implementación de un prototipo de semáforo inteligente utilizando sistemas embebidos. El objetivo de la investigación fue diseñar un prototipo de semaforización inteligente, que funcione mediante visión artificial en un sistema embebido para mejorar el flujo de personas en el cruce peatonal más transitado dentro del campus. Además, el sistema estaba conectado al internet, lo que permitía el monitoreo en tiempo real del lugar en el que se implementó.

Los resultados obtenidos mostraron ventajas significativas en cuanto a la eficiencia del flujo de personas en un cruce peatonal determinado, caracterizada por el movimiento continuo de vehículos y peatones. Se observó una reducción en los tiempos de espera, contribuyendo a una circulación eficaz del movimiento peatonal. Estos resultados indican que la aplicación de la semaforización inteligente que utiliza visión por computador y conexión a internet en conjunto con sistemas embebidos ofrecen beneficios en cuanto a congestión peatonal en lugares urbanos.

Este trabajo es un aporte para la ingeniería del tráfico, proporcionando una solución práctica para mejorar la eficiencia en intersecciones urbanas. Siendo una base para investigaciones futuras que traten con el tráfico vehicular y peatonal.

Palabras claves: **Visión artificial, Machine learning, Sistemas embebidos, Semaforización inteligente, IoT.**

Abstract

This project focused on transmitting the difficulties of pedestrian congestion on the Miguel de Cervantes campus of the Universidad Internacional SEK, through the implementation of a smart traffic light prototype using embedded systems. The research aimed to design a prototype of an intelligent traffic light, operating through computer vision on an embedded system to enhance the flow of people at the busiest pedestrian crossing within the campus. Additionally, the system was connected to the internet, allowing real-time monitoring of the implemented location.

The obtained results demonstrated significant advantages in terms of the efficiency of pedestrian flow at a specific crossing, characterized by continuous movement of vehicles and pedestrians. A reduction in waiting times was observed, contributing to an effective circulation of pedestrian movement. These findings indicate that the application of intelligent traffic lights using computer vision and internet connectivity, along with embedded systems, provides benefits in addressing pedestrian congestion in urban areas.

This work contributes to traffic engineering by offering a practical solution to improve efficiency at urban intersections. This project is the basis for future research addressing both vehicular and pedestrian traffic.

Keywords: **Artificial vision, Machine Learning, Embeeded systems, Smart traffic light systems, IoT.**

Tabla de contenidos

Lista de Figuras

Lista de Tablas

Introducción

Antecedentes

El aumento significativo de la cantidad de vehículos, junto con la creciente necesidad de movilidad de las personas, pone en desafío a los sistemas de transporte en áreas urbanas. Este problema se ha intensificado con el tiempo, afectando tanto a pequeñas como a grandes ciudades. Numerosos estudios han abordado esta problemática, generando modelos matemáticos que analizan el comportamiento del tráfico en las calles. De acuerdo con un artículo reciente de (Aleksey Kolodochkin et al., 2023) se destaca que la implementación de semaforización inteligente no solo mejora la capacidad de flujo, sino que también contribuye a mejorar la seguridad del tráfico de manera generalizada.

La preocupación de la población en cuanto al flujo del tráfico en las calles, junto con las implicaciones para peatones y vehículos, han sido objeto de investigación, mediante la búsqueda de soluciones a los desafíos asociados, promoviendo el desarrollo sostenible como respuesta a la problemática.

Los modelos matemáticos que se pueden desarrollar para describir todos estos procesos están fácilmente adaptados para herramientas de alta capacidad de cálculo, que pueden crear grandes modelos en escala real. Esto es relevante para mejorar la eficiencia del tráfico en intersecciones complicadas, debido a la gran cantidad de personas y vehículos en un espacio urbano reducido en muchos lugares del mundo.

Además, investigadores han propuesto un gran número de variaciones para el control de semaforización inteligente para mejorar el flujo del tráfico en intersecciones. Recientemente se ha desarrollado un proyecto para la revista Sensors que tiene por nombre "*Cyber-Physical System for Smart Traffic Light Control"* en el cual (Deshpande & Hsieh, 2023) afirman que el término de semaforización inteligente se define como emplear sensores y algoritmos para el control y optimización del tráfico vehicular.

El uso y transmisión de datos recopilados en tiempo real permite al sistema modificar el funcionamiento de un semáforo y mejorar el flujo de tráfico vehicular, minimizar la congestión y maximizar la seguridad de conductores y pasajeros.

También se ha presentado un trabajo publicado por (Pratama et al., 2018) en el cual se propone un sistema de semaforización el cual cambia el tiempo de cambio del color verde basado en la densidad de vehículos en los carriles de la avenida. Este método puede ser efectivo cuando hay muchas intersecciones cercanas, afectando la densidad del tráfico en la ciudad.

Existen muchos métodos mediante los que se ha buscado una mejora de los tiempos de espera en los semáforos, muchos de estos proyectos se han realizado con simulaciones y fabricación de prototipos de bajo presupuesto, lo que ha concluido que se puede realizar un proyecto con el mismo fin, sin la necesidad de utilizar muchos recursos, con un presupuesto bajo y que sea la base para futuros proyectos de investigación.

Planteamiento del problema

El tráfico de la ciudad de Quito ha ido empeorando con el paso del tiempo, a pesar de que el municipio ha dado soluciones como el trolebús, unidades de taxis metropolitanos, y el recientemente inaugurado Metro de Quito, la congestión vehicular sigue siendo un problema (Jonathan Machado, 2023).

Además, (Patricia Armijo, 2023) afirma en un artículo publicado por la revista El Comercio que Viviana Morocho coordinadora de la agencia metropolitana de tránsito, corrobora que cuando se regresó a la presencialidad post pandemia del COVID 19, volvieron 6500 unidades de transporte institucional y escolar que se sumaron al tráfico vehicular de la ciudad.

Por lo tanto, es necesario dar una solución a la problemática de alto flujo de tráfico vehicular y peatonal. La solución que plantea el presente trabajo es proponer un sistema que optimice las filas de espera que se generan en las intersecciones semaforizadas, modificando los tiempos de espera según el flujo de personas mediante la construcción de un prototipo de semáforo inteligente conectado al internet de las cosas.

Existe una variable muy importante que debemos tomar en cuenta cuando nos referimos a cruces semaforizados y es el número de peatones que se encuentran esperando a cruzar por una calle o avenida. Mucha gente puede pensar que el flujo peatonal no es importante, al contrario, puede ser igual o más importante que el flujo vehicular en algunos casos. (Raúl Izquierdo & David Nelson, 2023) mencionan en un artículo publicado por el Diario AS que en Tokio podemos encontrar un símbolo internacional del caos organizado en el cruce de Shibuya. En este lugar convergen grandes cantidades de personas, con un total diario de dos millones de personas moviéndose diariamente, ya que llegan a circular hasta 2500 personas cada dos minutos. Por lo que es considerada la intersección más transitada del mundo.

Con el crecimiento constante de la población cada año, el aumento del tráfico peatonal se convierte en un aspecto que no podemos pasar por alto en el futuro. Con el tiempo, fenómenos similares se manifestarán en otros países, siendo los más pequeños los más afectados

por el elevado flujo de tránsito vehicular y peatonal. La disponibilidad de espacio tanto para vehículos como para peatones en las intersecciones semaforizadas tendrá un impacto directo en la movilidad y la seguridad.

Justificación

"Conductores que pitan sin cesar y largas filas de autos son ya parte del paisaje de Quito. El tráfico se ha convertido en una rutina para quienes se movilizan de un lugar a otro de la ciudad". (Patricia Armijo, 2023)

Lamentablemente, la problemática de la congestión vehicular no es exclusiva de Quito; la preocupación se ha extendido a las alcaldías y municipalidades de otras ciudades. La urgencia de abordar este desafío requiere la implementación de medidas correctivas efectivas para mejorar el tráfico en las calles y avenidas urbanas.

Según (Orlando Silva, 2022) la ineficiencia del transporte público ha sido una causa importante para que la congestión vehicular en las zonas urbanas de diferentes ciudades sea un tema de preocupación para los mandatarios respectivos. Se debe abordar de manera integral la problemática del tráfico en entornos urbanos. Enfocándose no solo en la expansión de las infraestructuras viales, sino también en la optimización y fortalecimiento del transporte público.

En cuanto al transporte público se han tomado medidas al respecto, realizando planes de mejora. (Jonathan Machado, 2023) afirma en el artículo publicado por la revista Primicias que el actual alcalde de Quito Pabel Muñoz, presentó el *Plan movilidad de Quito 2024*, que se presentó al *Concejo Metropolitano de Quito*. El plan se llevará a cabo por un plazo de 19 años para reestructurar la movilidad en las calles de la ciudad. Lo más importante mencionado en este plan es mejorar el transporte público y promover su uso para que el porcentaje de personas que lo utilizan incremente del 51.4% al 62.1%. Después de esto se intenta reducir el porcentaje de personas que utilizan sus vehículos que es del 19% para bajarlo al 13%.

Sin embargo, esto no es una solución al problema, estos cambios no garantizan una mejora del tráfico, debe buscar una forma eficiente, que se pueda implementar a un sistema funcional como es el sistema de semaforización actual, solo que para hacerlo mucho mejor.

La gestión del tráfico en la ciudad es un problema que afecta a la seguridad vial, eficiencia del transporte y la calidad de vida de los habitantes. La implementación de sistemas de semáforos inteligentes con tecnología IoT puede mejorar significativamente la gestión del tráfico y reducir los problemas asociados con el mismo.

Actualmente existen muchas actividades que se desarrollan a nivel tecnológico, ejemplos claros de esto son los sistemas de parqueaderos, casas e industrias inteligentes, que tienen como base fundamental los datos que producen los dispositivos que funcionan con el internet de las cosas (*Internet of Things, Iot*). Lo que trae como consecuencia un crecimiento de esta tecnología. Estos equipos en su mayoría son utilizados para monitorizar, recibir, enviar y generar grandes cantidades de datos (Gradinescu et al., 2007).

La gestión de semáforos inteligentes se basa en la integración de dispositivos en la tecnología del internet de las cosas incorporados en los semáforos, lo que permite la recopilación de datos en tiempo real sobre el tráfico y su uso para mejorar la gestión del tráfico. Esta tecnología también podría utilizarse para la comunicación en tiempo real entre los semáforos, lo que permite el desarrollo futuro de un sistema que tenga como objetivo último la reducción de la congestión automovilística y flujo eficiente de peatones en secciones semaforizadas.

Hipótesis

La implementación de semaforización inteligente mediante el uso de IoT y sensórica puede ser una solución para mejorar en 15% la eficiencia en tiempo del flujo peatonal en el Campus Miguel de Cervantes de la Universidad Internacional SEK, al permitir una gestión automatizada del flujo peatonal en tiempo real, lo que reduce el tiempo de espera en los semáforos.

Objetivo General

Diseñar un prototipo de semáforo inteligente mediante visión artificial y aprovechamiento del Internet de las Cosas (IoT), para mejorar la gestión del tráfico peatonal en intersecciones.

Objetivos específicos

- Diseñar un prototipo de semáforo inteligente, para reducir el tiempo de espera en el paso peatonal más transitado del campus Miguel de Cervantes de la Universidad Internacional SEK mediante el uso de visión artificial y tecnología IoT.
- Dimensionar el embebido de la aplicación para su correcto funcionamiento mediante el respectivo proceso ingenieril.

- Implementar un protocolo de comunicación entre el dispositivo y el IoT, para su monitoreo en tiempo real mediante las herramientas de comunicación necesarias.
- Evaluar funcionalidad del prototipo mediante su implementación en el campus Miguel de Cervantes de la Universidad Internacional SEK.

Estado del arte

Internet of things (IOT)

Durante los últimos diez años el internet de las cosas (IoT, Internet of things) ha sido objeto de importantes indagaciones científicas. Según (Ray, 2018) se define como una conexión entre el mundo digital y el físico. Su crecimiento exponencial de cosas que encontramos conectadas a internet nos permitió que en nuestro entorno se convierta en inteligente.

Tal es el caso de hogares, medios de transporte, aeropuertos, gobiernos, educación, logística e industria.

Se busca utilizar esta tecnología que nos brinda muchas facilidades a nivel mundial, las ciudades dentro de un futuro cercano harán uso de la Iot con múltiples propósitos que facilitarán la vida cotidiana en muchos aspectos, mediante la comunicación de dispositivos y el flujo de datos en tiempo real.

IIoT

El internet industrial de las cosas o más conocido por sus siglas en inglés IIoT surge en conjunto con la definición de la Industria 4.0. El concepto surge cuando contamos con actuadores y máquinas en conjunto con sensórica que se encuentran conectados al internet (Abuhasel & Khan, 2020). Es muy importante esta conectividad, ya que gracias a ella podemos monitorear, analizar y actuar a partir de una situación. Cambiando el comportamiento de dispositivos y máquinas, con muy poca participación del ser humano, facilitando tareas que pueden ser complicadas de realizar.

Elementos técnicos del IoT

En su trabajo de titulación (Escobar Duque & Jonathan Steve, 2020)explican que el internet de las cosas está conformado por los elementos descritos a continuación:

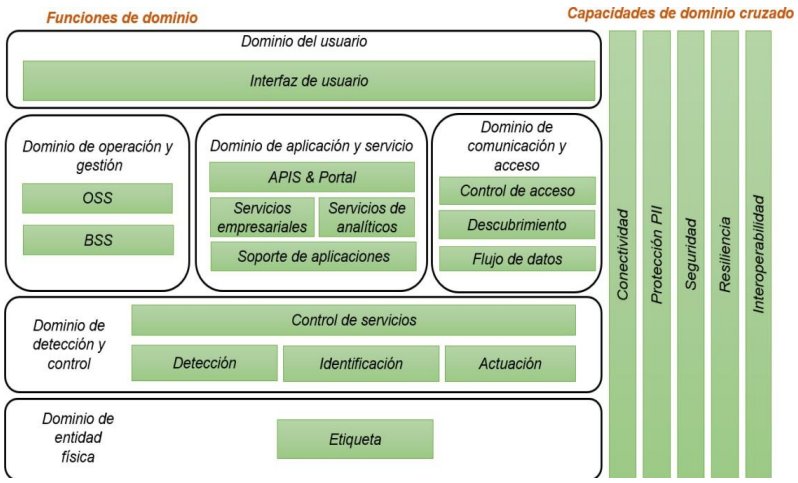
- Hardware: Es todo lo que conforma la unión de actuadores, sensores y elementos necesarios para realizar la comunicación entre dos o más dispositivos.
- Middleware: Esta conformado por herramientas de desarrollo y microcontroladores que trabajando en conjunto pueden circular información y realizar un análisis.
- Dashboard: Plataformas que permiten mostrar información al usuario.

Estándar del Internet de las cosas

Se ha ido desarrollando durante los últimos años el término de IoT. La Organización Internacional de Normalización (ISO) en conjunto con la Comisión Electrotécnica Internacional (IEC) hacen público el primer estándar que define la arquitectura del IoT nombrada ISO/IEC 30141, para cumplir con las características que utilizan estos sistemas. Misma que se puede apreciar en la Figura 1.

Figura 1.

Arquitectura del IoT



Nota: En este gráfico se muestra la Arquitectura del Internet de las cosas

(Organización Internacional de Normalización, 2018). ISO/IEC 30141:2018.

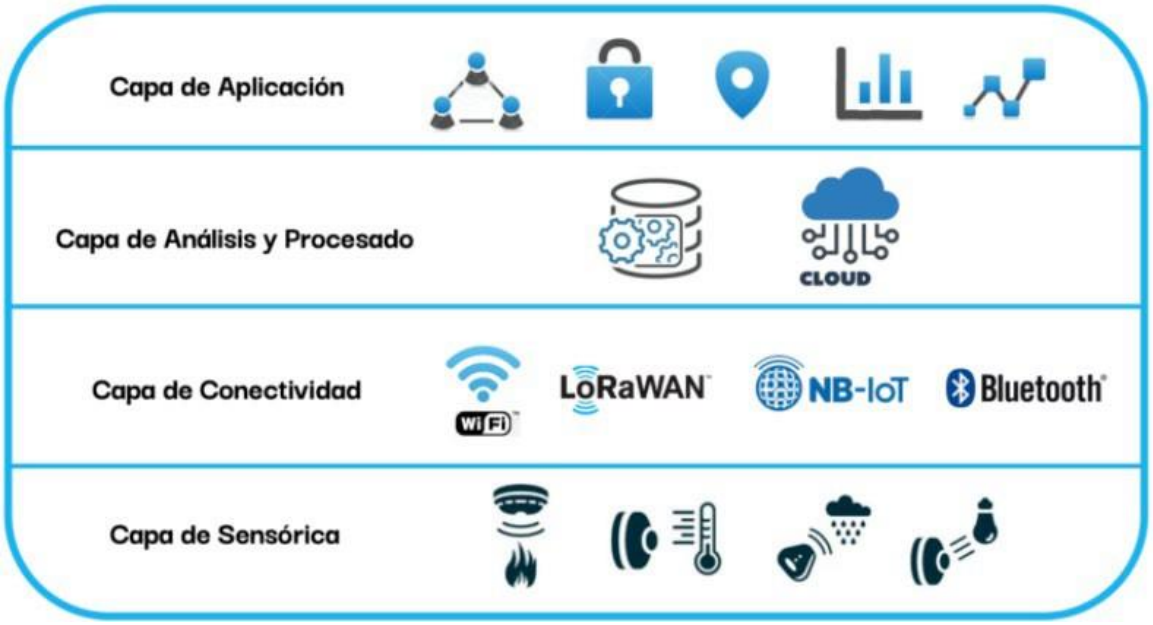
Además, de tener la capacidad de contar con un control sin la necesidad de tener una acción humana. Se han ido añadiendo muchas más tecnologías como Inteligencia Artificial y Machine Learning. Como menciona (Istec, 2022). El IoT se define a grandes rasgos por las siguientes etapas:

- **Sensórica:** Brinda a dispositivos la capacidad de realizar medidas o control de medidas o parámetros determinados. Debe tener un consumo energético muy reducido. Alguno de estos puede presentar la característica de poder realizar acciones específicas.
- **Conectividad:** Establece comunicación entre los sensores y una puerta de acceso, la cual puede estar conectada mediante una máquina concentrada o con una arquitectura en la nube, mediante las amplias opciones de conexiones inalámbricas que tenemos (Bluetooth, WI-FI, Zigbee, etc.).
- **Análisis:** Permite la recopilación entre datos provenientes de los sensores para que sean analizados en tiempo real.
- **Interfaz de usuario:** Permite administrar los dispositivos, ayudando al usuario integrar, monitorizar, y gestionar a distancia los dispositivos, además de mantener la conectividad y seguridad del sistema.

En la Figura 2 se puede apreciar un gráfico de representación de las capas del IoT.

Figura 2.

Capas de la arquitectura del IoT.



Nota: En este gráfico se representa gráficamente las capas del IoT (Istec, 2022). Internet of things, IoT.

Inteligencia Artificial

La *IA* conocida por sus siglas (Artificial Intelligence). Es la manera en la que una máquina tiene comportamientos que requieren cierta inteligencia. Es decir que puede cumplir con un objetivo ya sea por aprendizaje o por pre-programación (Chauca, 2020).

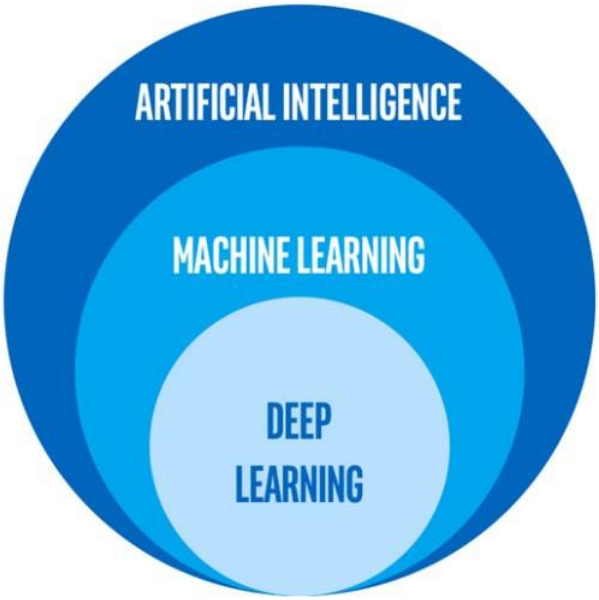
Los avances importantes que han surgido en los últimos años son diversos, están enfocadas a diferentes áreas que se pueden considerar como inteligencia, como pueden ser:

- Visión por computador: El objetivo a cumplir es entender el mundo real a través de una imagen o un grupo de imágenes mediante su debido procesamiento. Tiene la característica de ver.
- Reconocimiento de imágenes: Se debe reconocer una imagen mediante un sistema pre-entrenado mediante diferentes métodos de programación.
- Clasificación de videos: Se detectan cambios determinados en un entorno de video. Así es como un sistema de seguridad puede detectar un intruso en un área determinada.
- Procesamiento de lenguaje: La tarea es entender y procesar el lenguaje a través de un procesador.

Un dispositivo que cuenta con una característica de inteligencia y que lo define como tal debido a su capacidad de aprender y mejorar esta propiedad con un sistema de aprendizaje continuo, comúnmente conocido como *Machine Learning.* Es la forma de generar inteligencia en un dispositivo en la actualidad (Intel, 2020).La Figura 3 nos permite entender la relación entre la *Inteligencia Artificial*, el aprendizaje de máquinas comúnmente conocido como *Machine Learning* y el *Deep Learning* conocido por ser un tipo de aprendizaje automatizado mediante diferentes métodos.

Figura 3.

Relación existente entre deep learning, machine learning e inteligencia artificial



Nota. En este gráfico se representa los diferentes tipos de aprendizaje que contiene la IA (Intel, 2020). *The Difference Between Artificial Intelligence, Machine Learning and Deep Learning.*

Machine Learning

El término machine learning tiene como significado aprendizaje de máquinas. Así como los humanos aprendemos desde las experiencias, las máquinas pueden hacer lo mismo a través de un grupo de algoritmos. En su libro *Machine Learning* (Zhi-Hua Zhou, 2020) nos dice que, esta tecnología mejora el rendimiento del sistema mediante métodos computacionales.

Es muy importante mencionar este tipo de manejo de los datos, ya que se pueden optimizar procesos mediante el manejo de información para obtener una mejora en el sistema al que se le puede aplicar un método de aprendizaje profundo.

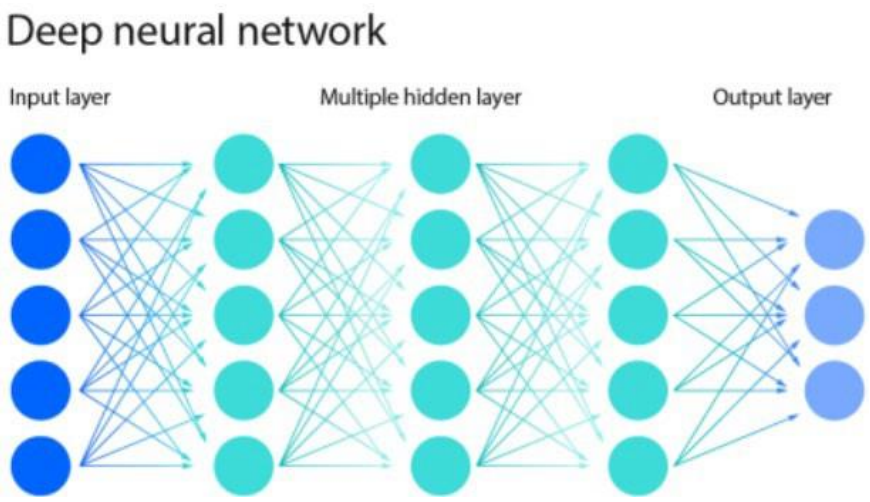
Redes Neuronales

Las redes neuronales se componen a través de subconjuntos del aprendizaje de máquinas y se constituyen en una serie de algoritmos del aprendizaje profundo. Adquieren su nombre y estructura inspirados en el cerebro humano, y tienen un comportamiento similar al de las neuronas. Están conformadas por capas de nodos, clasificándose en capas de entrada y de salida. Cada nodo que es como se denomina a la neurona artificial se conectan entre si cada uno con un peso y umbral diferente. Están basadas en el entrenamiento de datos para que puedan mejorar su precisión mediante aprendizaje, y pueden convertirse en potentes herramientas

informáticas (IBM, 2021). En la Figura 4 se muestra una representación de una red neuronal profunda.

Figura 4.

Red neuronal profunda



Nota. En este gráfico se representa una red neuronal profunda (IBM, 2021). *¿Qué son las redes neuronales?*

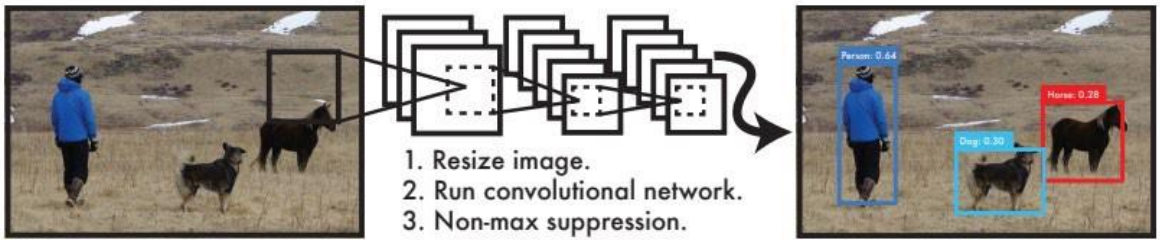
YOLO

YOLO se define por sus siglas inglés (You Only Look Once). Es nuevo enfoque de la detección de objetos. Se hace la detección de objetos mediante una regresión en lugar de hacer una tarea de clasificación, separando cuadros delimitadores mientras se asocian probabilidades a cada una las imágenes utilizando una sola red neuronal convolucional.

Esta arquitectura es extremadamente rápida. Procesa imágenes en tiempo real a 45 cuadros por segundo (Redmon et al., 2015). En la siguiente Figura 5 se puede observar cómo funciona YOLO.

Figura 5.

Funcionamiento de la arquitectura You Only Look Once.



Nota: *En este gráfico se representa el funcionamiento de la arquitectura de YOLO* (Redmon et al., 2015). *You Only Look Once: Unified, Real-Time Object Detection.*

El procesamiento de imágenes mediante la utilización de esta arquitectura es simple ya que consta de tres sencillos pasos:

- El sistema redimensiona la imagen de entrada.
- Inicia el procesamiento de una simple red neuronal en la imagen.
- Se hace un umbral y se muestra las detecciones por la confidencia del modelo.

Este método tiene muchos más beneficios que los métodos de detección tradicionales. Ya que YOLO es extremadamente rápido. Dado que la detección por cuadros es solamente una regresión. Se ejecuta en una red neuronal una serie de imágenes en el momento predecir las detecciones en un cuadro delimitador que diferencia el objeto reconocido del resto de la imagen.

El sistema divide la imagen de entrada en cuadrículas $S \times S$. Si es que el centro de un objeto es detectado en una cuadrícula específica se responsabiliza de identificar ese objeto (Redmon et al., 2015).

Cada cuadrícula determina si tiene una predicción de los cuadros delimitadores del objeto BB y asigna valores para estas. Estos valores reflejan que tan fiable es el modelo para que el cuadro delimitador contenga un objeto. Se define la fiabilidad como:

$$\Pr(oooooooooooo) \times IIIIII(ffffffooffffffffffff).$$

Si no existe ningún objeto dentro de la cuadrícula, la confiabilidad no debería tener ningún valor. También necesitamos que $IIIIII$ (intersection over union) se pueda predecir de tal manera que coincida con la posición real del objeto (Redmon et al., 2015).

Cada cuadro delimitador cuenta con 5 predicciones que son: $x, y, w, h,$ y fiabilidad. Las coordenadas (x, y) representan el centro del cuadro delimitador relacionados con los vínculos de cada cuadrícula de la imagen. Finalmente, la fiabilidad se representa por $IIIIII$ que se encuentra entre la predicción del cuadro delimitador y la posición original del objeto dentro de la imagen (Redmon et al., 2015).

Cada cuadrícula predice el condicional CC definido como condición de clase, $\Pr(CCffffCCoo_{ii}|\text{Objeto})$. Estas probabilidades están condicionadas para la cuadrícula que contiene un objeto. Solo se predice una clase de probabilidades para cada cuadrícula, independientemente de los cuadros delimitadores BB. Por lo que se multiplica la probabilidad de clase condicional y las predicciones de cada cuadro delimitador al momento de la prueba y se obtiene:

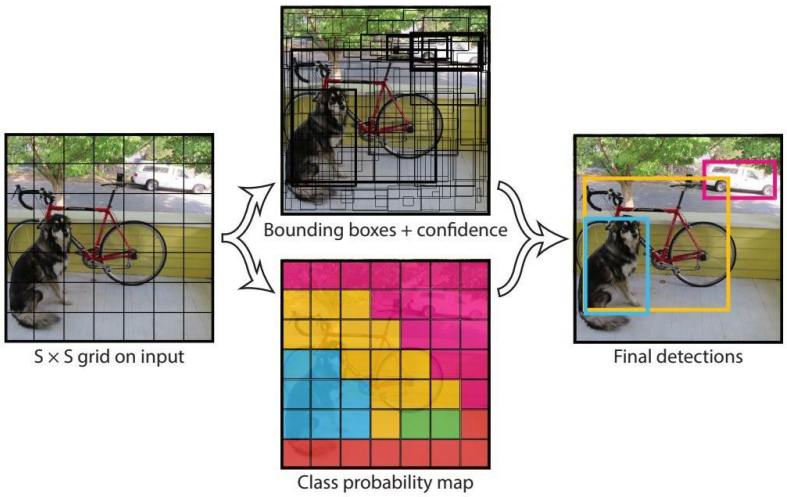
$$\Pr(CCffffCCoo_{ii}|IIoooooooooo) \times \Pr(oooooooooooo) \times IIIIII(ffffffooffffffffffff) = \Pr(CCffffCCoo_{ii}) \times IIIIII(ffffffooffffffffffff)$$

Lo que nos permite obtener valores específicos de confidencia para cada clase y codifican la probabilidad de que una clase aparezca en un cuadro delimitador y que tan preciso es en relación con el objeto identificado (Redmon et al., 2015).

Se puede observar en la Figura 6 como se realiza esta identificación de objetos:

Figura 6.

Resultados de detección de objetos de YOLO



Nota: *En este gráfico se representa el modelo de detección de objetos de YOLO* (Redmon et al., 2015). *You Only Look Once: Unified, Real-Time Object Detection.*

Según (Redmon et al., 2015)el tensor trabaja de la siguiente forma:

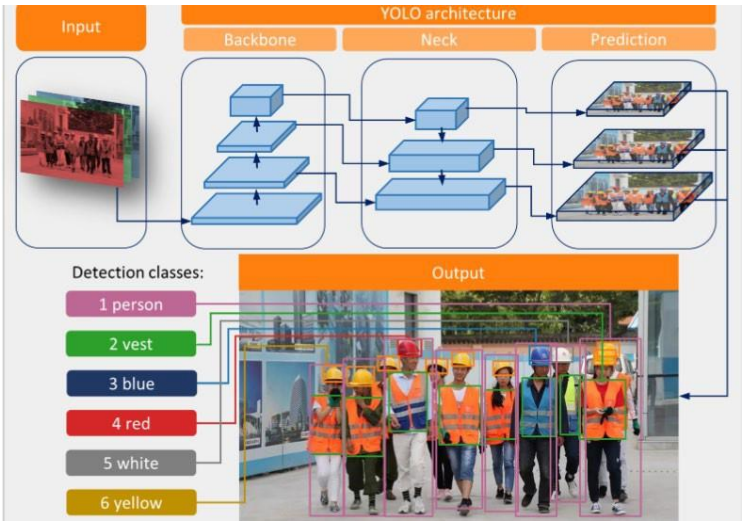
$$S \times S \times (B \times 5 + C)$$

El valor *BB* se obtiene a través del dataset con el que se hizo el entrenamiento de este modelo. Mientras que el valor de C es el número de clases identificadas por cuadro delimitador.

Se puede observar en la Figura 7 como está conformada la red YOLO. Esta red contiene tres partes que se denominan columna, cuello y cabeza. La columna es la que contiene la red neuronal convolucional, responsable de las siluetas características de cada imagen de entrada. El cuello se constituye de capas encargadas de combinar características de cada imagen para enviarlas a la cabeza, con el trabajo de realizar la predicción (Wang et al., 2021).

Figura 7.

Arquitectura de YOLO



Nota: S*e muestra en la figura las diferentes partes de la red de YOLO.*

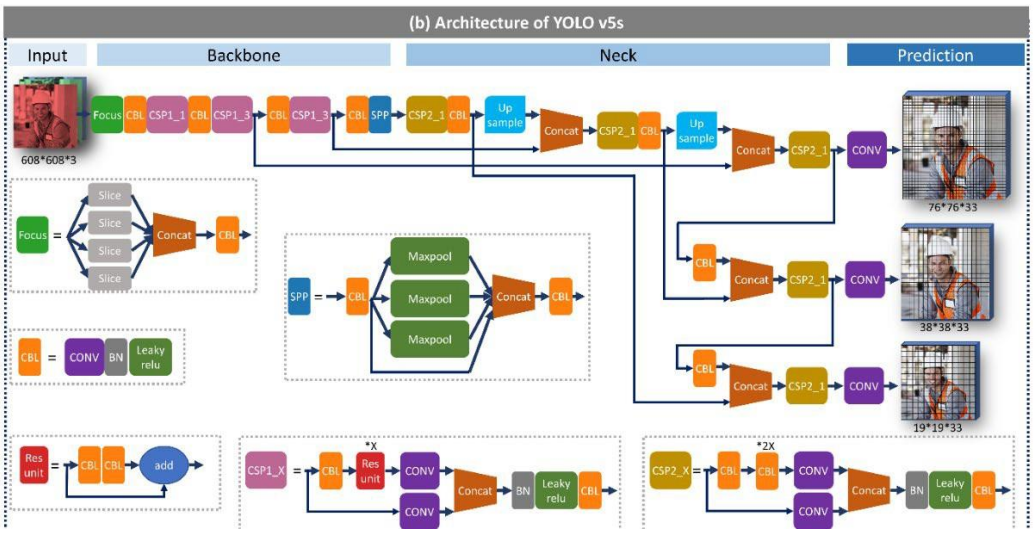
Fast Personal Protective Equipment Detection for Real Construction Sites Using Deep Learning Approaches (Wang et al., 2021).

YOLOv5

Esta versión de YOLO funciona de la misma forma que la primera, solo que esta se enfoca para su uso Python, mejorando su eficiencia y rendimiento utilizando el esquema de trabajo de Pytorch, el cual nos permite ajustar cambios en el programa mediante un software de desarrollo en este lenguaje de programación. Tiene como objetivo principal realizar trabajos de visión artificial con gran precisión y velocidad (Ultralytics, 2024). En la Figura 8 se puede observar su arquitectura.

Figura 8.

Arquitectura de YOLOv5s



Nota: Se muestra en la figura la arquitectura de la red de YOLOv5.
Fast Personal Protective Equipment Detection for Real Construction Sites Using Deep Learning Approaches (Wang et al., 2021).

Semáforo inteligente

Se presenta un control de parámetros en los semáforos que se basa en un estudio estadístico diseñado por Webster durante los años sesenta. Se menciona en el artículo de(Gradinescu et al., 2007). La propagación de las ondas sonoras, y en el caso de los semáforos inteligentes se utilizó para explicar costos y tiempos. En este artículo se presentó una solución en una intersección de la ciudad de Bucarest en la que se utilizó transferencia de datos por medio de redes inalámbricas, las que necesitan adaptar a los vehículos de un dispositivo que emita una señal cuando pase por un semáforo, con el fin de saber el tiempo de ciclo óptimo para los tiempos de espera en los semáforos. Este trabajo de igual manera busca dar una solución a la congestión vehicular.

Se ha presentado un artículo en la página de IOPscience en el cual (Mahesh Kumar & Kumaraswamy, 2020). Exponen que mientras siga existiendo un aumento de vehículos en las calles, el número de accidentes pueden aumentar debido a la mala señalización en conjunto con el flujo normal de personas y vehículos, resaltando así que se deben realizar investigaciones y trabajos futuros en cuanto sistemas de control del tráfico con funciones inteligentes, ya que el número de usuarios de las vías se aumenta y los recursos y sistemas implementados en la actualidad son limitados. Presentando así un sistema de cruce inteligente que utiliza un sistema de procesamiento de datos en tiempo real obtenido de sensores y basado en Raspberry Pi, para que los vehículos que llegan cuando el semáforo está en rojo se eleva la barrera, y cuando este en verde la barrera se retrae. Asegurando que con este sistema existirán menos atascos en la vía.

Teoría de colas

La teoría de colas es una técnica que se basa en la investigación de diferentes operaciones que solucionan problemas en situaciones en las que se forman turnos de espera o también llamados colas para la prestación de un servicio o la resolución de una acción en específico (González & Sepulveda, 2010).

La teoría de colas es un estudio matemático de los comportamientos de las filas de espera. Estas se presentan cuando clientes llegan a un lugar en específico buscando un servicio.

Los modelos justamente sirven para encontrar una relación adecuada entre los costes del sistema y tiempos promedio de una cola de espera para dicho sistema (González & Sepulveda, 2010).

Es importante mencionar la teoría de colas, ya que en cualquier lugar que forme una cola de espera, ya sea para obtener un servicio o beneficio tiene la explicación de su comportamiento según variables determinadas.

Visión artificial

La visión artificial o visión por computador es la tecnología que permite a las máquinas y sistemas "ver", extraer información de las imágenes, resolver algún problema o cumplir alguna tarea mediante el procesamiento de las imágenes por un computador que le indica al sistema cómo comportarse mediante código de programación (González & Sepulveda, 2010).

La visión artificial es lo que hace posible que las cámaras puedan identificar, personas, animales y cosas. Se ha usado en diferentes aplicaciones con el paso de los años, ya que se capacidad de dar información relevante sobre las imágenes al usuario es infinita.

Procesamiento de imágenes

Según el artículo de (Lakshmi & Gayathri, 2017). El procesamiento de imágenes son imágenes procesadas a través de métodos matemáticos, usando cualquier forma de procesamiento de señales dentro de la cual se utiliza una entrada que puede ser una imagen, grupo de imágenes o un video. La salida de este sistema puede ser una imagen o una serie de características de parámetros obtenidos a partir de la imagen.

Lo que quiere decir esto es que podemos recolectar información a través de imágenes con un computador, y lo que se puede hacer con la información obtenida a partir de estas imágenes tiene millones de aplicaciones y dependen totalmente de la habilidad de desarrollo de software del usuario.

OpenCV

Esta librería está orientada casi en su totalidad a la visión por computador en tiempo real. De sus aplicaciones destacan: segmentación y reconocimiento de objetos, seguimiento de objetos, identificadores de gestos (Arévalo Vicente et al., 2004). Mientras utilicemos esta librería contamos con una gran cantidad de herramientas de programación para realizar el correspondiente algoritmo de visión artificial.

Microprocesador

Un microprocesador es un componente LSI que realiza un gran número de operaciones en una sola unidad de circuito integrado (David Perez, 2009). El término LSI, se refiere a la tecnología que permite incorporar miles de transistores en un mismo sistema integrado. Las características de diseño de un microprocesador pueden variar de acuerdo con su funcionalidad.

Raspberry Pi

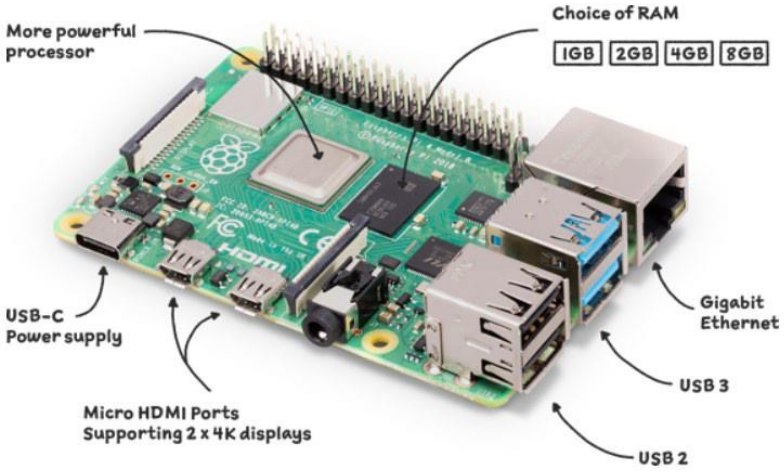
La tarjeta de desarrollo Raspberry Pi puede realizar tareas similares a una computadora. Fue creada en el Reino Unido con la ventaja de su pequeño tamaño y se puede encontrar en el mercado con el concepto de ser una tarjeta de desarrollo de bajo costo que puede ser usada como computadora. Cuenta con memoria RAM, procesador y tarjeta gráfica, puerto ethernet, conexión al Wi-Fi, Pines GPIO, fuente de alimentación y una gran cantidad de interfaces de desarrollo (Lakshmi & Gayathri, 2017).

La importancia de la tarjeta de desarrollo Raspberry Pi es fundamental en el desarrollo del prototipo que se plantea en este trabajo, ya que a través de esta se va a hacer toda la programación referente al procesamiento de imágenes y la necesaria conexión a internet del prototipo. Es decir que todo el embebido del prototipo está basado en este microprocesador.

Para llevar a cabo este cometido, se propone utilizar el microprocesador Raspberry Pi 4 el ambiente de trabajo de micro-Python y su debido desarrollo de software para monitoreo del tráfico peatonal en tiempo real. Se muestra en la Figura 9 y sus características en la Tabla 1.

Figura 9.

Raspberry Pi 4 modelo B



Nota: Se observa la placa Raspberry Pi 4 modelo B y sus partes más importantes (Raspberry Pi, 2024a). Extraído de: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/

Tabla 1.

Especificaciones técnicas de Raspberry Pi 4 modelo B

Característica	**Descripción**
CPU	Quad Core 64-bit ARM a 1.5 GHz
RAM	4 GB
WI-FI	802.11/b/g/n/ac Wireless LAN
Bluetooth	Bluetooth 5.0 BLE
Ethernet	1 puerto Gigabit Ethernet
GPIO	Cabecera de 40 pines
HDMI	dos puertos
USB	dos puertos 2.0 y dos puertos 3.0
Puerto CSI	Permite conexión de una cámara
Puerto DSI	Permite conexión de una pantalla táctil
Puerto Micro SD	Para sistema operativo y almacenamiento
Puerto Micro USB	Alimentar placa 5v

Nota: Resumen de las especificaciones técnicas del Datasheet de la Raspberry PI 4 modelo B (Boris Cutos & Miguel Recalde, 2022).

Lo que se hace tras adquirir este computador es contar con los componentes necesarios para que funcione con su correspondiente sistema operativo. Los componentes necesarios para su funcionamiento se pueden apreciar en la Tabla 2.

Tabla 2.

Componentes necesarios para iniciar el sistema operativo de Raspberry Pi

Componente	**Unidad**
Fuente de poder	1
Disco duro (microSD 32Gb)	1
Monitor con entrada HDMI	1
Teclado	1
Mouse	1
Cable HDMI	1

Nota: Se muestra la cantidad de componentes necesarios para poder iniciar el sistema operativo de Raspberry Pi.

Microcontrolador

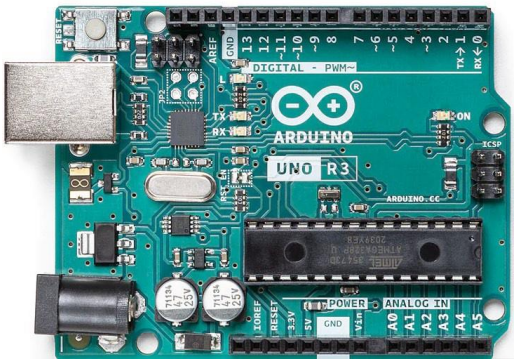
Según (Julio Schwendener, 2023) se puede definir a un microcontrolador como un procesador reducido que contiene temporizadores, memoria, pines de entrada y salida. Todo esto en un espacio pequeño, minimizando los costos empleados en manufactura y tiempo de trabajo. Además, se pueden utilizar en una gran cantidad de aplicaciones de monitoreo, control y procesamiento en sistemas.

Arduino Uno

Es una plataforma de desarrollo de código abierto, la cual tiene un hardware y software muy fácil de usar para cualquier usuario. Las placas de Arduino son capaces de leer señales de entrada, encender sensores y prender leds. A través de los años esta plataforma de desarrollo ha sido la base para una infinidad de proyectos, desde los más comunes hasta complejos instrumentos científicos (Arduino, 2018). Se puede apreciar en la Figura 10 y sus características en la Tabla *3*.

Figura 10.

Arduino Uno R3



Nota: La Figura muestra la última versión de la tarjeta de desarrollo Arduino Uno R3 (Arduino Store, 2021). *Extraído de: https://store.arduino.cc/products/arduino-uno-rev3*

Tabla 3.

Características básicas de Arduino Uno R3

Voltaje de funcionamiento	5v
Pines digitales I/O	14 (6 cuentan con salida PWM)

Pines analógicos	6
Corriente	40 mA
Memoria flash	32 KB (ATmega328)
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)

Nota: Se muestran las características básicas de la tarjeta de desarrollo Arduino Uno R3 (Arduino.cc, 2024). Extraído de: https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

Metodología

Este trabajo se sustenta en una metodología de investigación experimental, la cual se fundamenta en la revisión bibliográfica presentada en este documento y en una fase inicial de investigación. Se modificaron las condiciones preestablecidas en una o más variables con el propósito de evaluar las repercusiones en el sistema, generando interacciones en dichas variables (Ramos, 2021).

La metodología que se va a utilizar en este proyecto se rige por los siguientes pasos.

- Pertinencia: La problemática planteada en un inicio se presenta para la mejora de eficiencia en el flujo del tráfico de personas en el campus Miguel de Cervantes de la Universidad Internacional SEK e implementar tecnología IoT para el monitoreo del flujo en tiempo real.
- Preámbulos: Aquí se establecen las condiciones bajo las cuales se debe someter el desarrollo del programa. Deben ser las adecuadas para alcanzar el objetivo del proyecto.
- Diseño e implementación: Esta etapa de diseño construye un algoritmo que incluyen su respectivo diagrama de flujo que sigan las especificaciones del paso anterior.
- Verificación: En esta etapa se realizarán las pruebas necesarias, mediante las cuales se evalúa la viabilidad y funcionalidad del programa. Se toman en cuenta las condiciones
- Objetivos específicos: para en los resultados verificar si el sistema satisface estos objetivos planteados.

En la Figura 9 se muestra el diagrama de los procesos necesarios para llevar este proyecto a su conclusión.

Figura 11.

Diagrama de procesos del desarrollo del proyecto

Pertinencia

El 16 de noviembre del 2023, se realizó un análisis de datos mediante observación en el campus Miguel de Cervantes de la Universidad Internacional SEK ubicada al norte de la ciudad de Quito, a través de la cual se llegó a la conclusión que se debe tener un control y monitoreo en tiempo real de las variables dependientes del flujo del tráfico dentro del campus. Ya que, se puede comprobar que existe un gran flujo de peatones y vehículos en el cruce peatonal principal en hora pico.

Los correspondientes datos del flujo vehicular tomados en la fecha mencionada se dividen en tres variables importantes; vehículos que circulen en sentido norte a sur, vehículos

circulando en sentido sur a norte y cruce de individuos por el paso peatonal. Se pueden expresar tabulados en la Tabla 1 y se los representa gráficamente en la Figura 10.

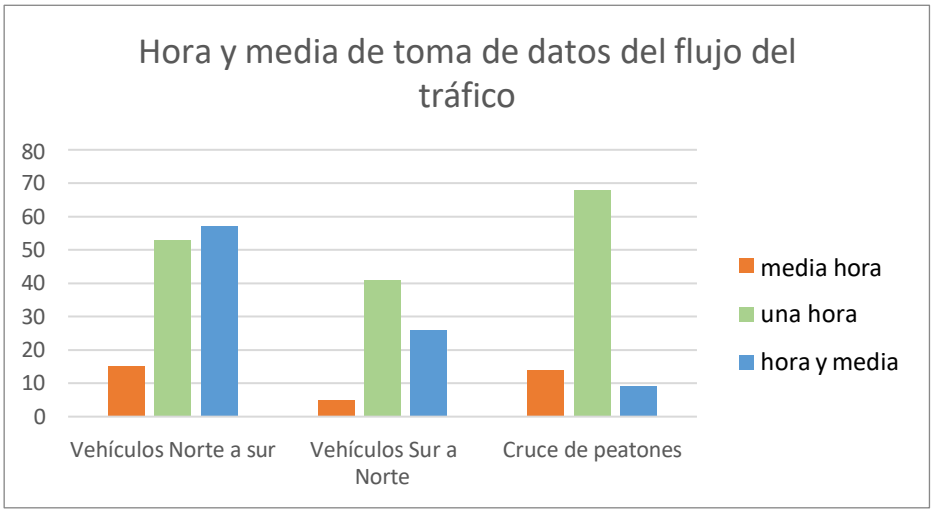
Tabla 4.

Flujo del tráfico vehicular de 6:18 am- 7:48 am en paso peatonal del campus Miguel de Cervantes de la Universidad Internacional SEK

Unidades de vehículos circulando de norte a sur	Unidades de vehículos circulando de sur a norte	Peatones cruzando la calle
125 unidades	72 unidades	91 individuos

Figura 12.

Representación de los datos obtenidos mediante observación del flujo del tráfico en hora de alto flujo vehicular



Nota: En el Anexo A se muestra un cuadro del video utilizado para recolectar estos datos mediante observación en las fechas mencionadas anteriormente.

A partir de este gráfico se puede determinar que hay una gran cantidad de flujo vehicular a partir de las 6:48 hasta las 7:18 de la mañana. Por lo que se debe tener en cuenta estas horas en una comparación futura, cuando el prototipo ya se encuentre en operación para saber el porcentaje de mejora en el flujo peatonal teniendo en cuenta el tiempo de espera promedio para el cruce. Lo que permitirá tener una mejora en la eficiencia en los tiempos de espera para los peatones. Mientras que en el caso del flujo vehicular se asegura tener un monitoreo constante del mismo. En la Tabla 5 se puede apreciar el tiempo de espera por persona para cruzar por el paso peatonal sin la implementación del prototipo.

Tabla 5.

Tiempos de espera por persona para cruce por paso peatonal principal del campus Miguel de Cervantes

Número de peatones en el grupo esperando a cruzar	Tiempo de espera	Momento de tiempo en el video
3	2,53 segundos	29:55
2	4,53 segundos	37:57
3	3,32 segundos	38:50
1	3 segundos	41:15
3	4,48 segundos	42:12
5	5,42 segundos	43:17
3	6,35 segundos	52:10
1	1,80 segundos	1:03:40
1	8,24 segundos	1:29:40
Total: 22 personas	Tiempo promedio de espera por persona: 4.51 segundos	

Nota: Estos datos fueron extraídos mediante observación del video mostrado en el Anexo A, muestra el tiempo de espera por persona para cruzar por el paso peatonal sin la implementación del prototipo de semaforización inteligente.

Preámbulos

Para completar el propósito del proyecto, se deben determinar las condiciones que deben cumplir dando un resultado satisfactorio. Se propone utilizar la arquitectura de YOLO funcionando en un ambiente de micro-python para el microprocesador Raspberry Pi, en conjunto con un algoritmo de semaforización corriendo en el microcontrolador Arduino Uno.

Una vez determinado el microprocesador para desarrollar el algoritmo de procesamiento de imágenes, destacamos que cuenta con pines de entrada y salida de propósito general, pero una de las ventajas más grandes de su uso es la conectividad a Internet, lo que permite que el prototipo que estamos construyendo cuente con la característica de ser un dispositivo del Internet de las cosas.

Cuando contemos con los componentes necesarios para su correcto funcionamiento, se debe descargar el sistema operativo de Raspberry en un computador mediante el Raspberry Pi

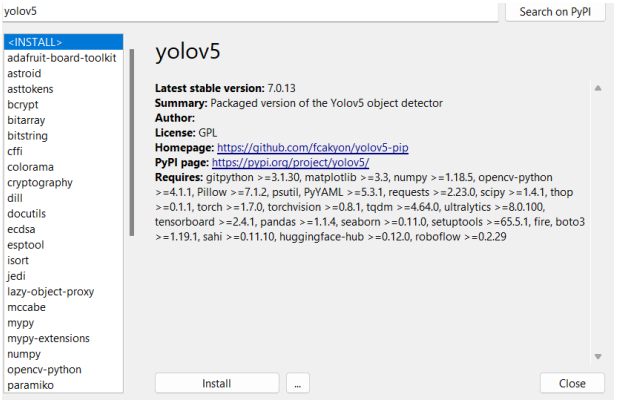
Imager, el cual es proporcionado por la página oficial del desarrollador y se muestra en el Anexo B.

Tras haber seleccionado el modelo, sistema operativo deseado y el destino de almacenamiento del sistema operativo, se descarga en la memoria externa microSD. Una vez descargado el software en la memoria externa, la colocamos en la placa de la Raspberry Pi para su correcto funcionamiento. Posteriormente, se proceden a hacer las conexiones respectivas para iniciar el microprocesador.

Para Realizar visión artificial a través de una cámara conectada a la Raspberry Pi, se utiliza el modelo pre entrenado YOLOv5s para realizar identificación de objetos en tiempo real. Tiene como objetivo principal realizar trabajos de detección y clasificación de objetos con gran precisión y velocidad (Ultralytics, 2024). Se puede observar en la Figura 13 como se puede descargar a través de PyPI.

Figura 13.

Instalación de YOLOv5 en el IDE Thonny

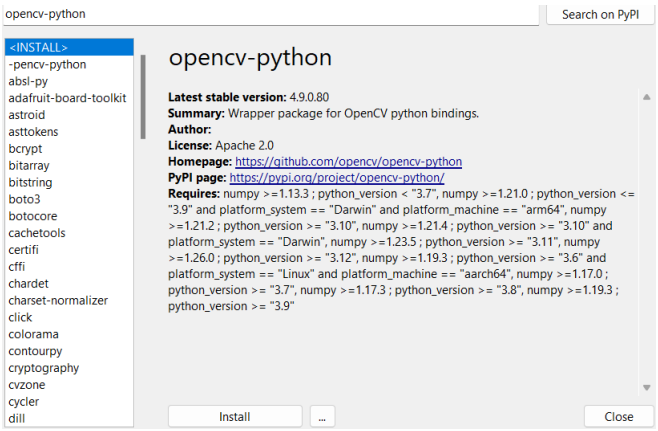


Se procede a realizar la instalación de todos los paquetes necesarios para que pueda funcionar de manera correcta el modelo YOLO para el algoritmo de detección de objetos mediante la inicialización de una sola red neuronal convolucional previamente entrenada por Ultralytics, cuenta con una lista de objetos que pueden ser detectados por esta arquitectura, entre los cuales se pueden encontrar personas. Además, utilizar los pines de entrada y salida de propósito general con las que cuenta la Raspberry Pi para la correspondiente interrupción en el flujo normal del tiempo de espera del semáforo.

La librería OpenCV facilita una serie de herramientas que permiten realizar conteo de personas, mediante la detección de objetos con el modelo pre-entrenado seleccionado anteriormente. Se puede instalar a través del gestor de librerías como se observa en la Figura 14.

Figura 14.

Instalación de OpenCV en el IDE Thonny



También, se hará uso de la interfaz de desarrollo de Arduino con dos propósitos importantes mencionados a continuación:

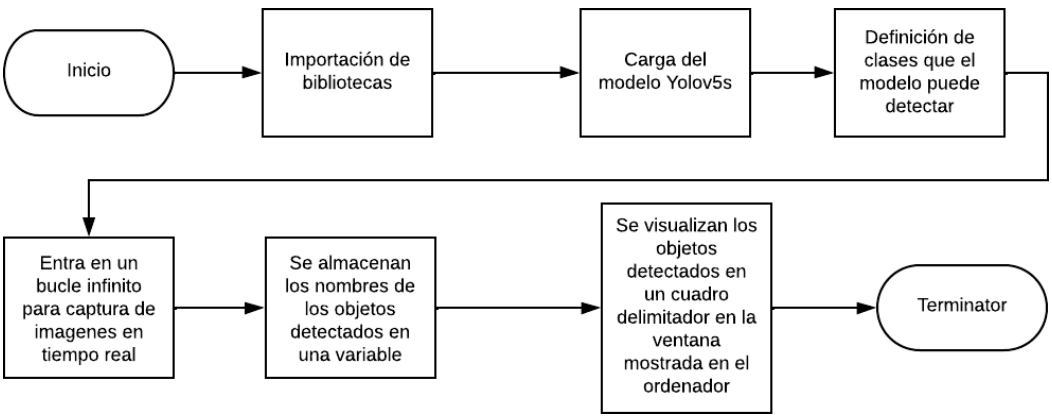
- Crear algoritmo de funcionamiento ordinario de un semáforo.
- Crear una interrupción en el flujo de programación del semáforo mediante la lectura de la señal lógica enviada desde la Raspberry.

Diseño e implementación

En la Figura 15 se representa el diagrama de funcionamiento del programa básico para hacer funcionar al modelo YOLOv5s.

Figura 15.

Diagrama de flujo del algoritmo del modelo pre entrenado YOLOv5 para reconocimiento de objetos



La programación completa definida por el diagrama se escribió en Python y se puede encontrar en el Anexo C.

Para el reconocimiento de objetos en tiempo real necesitamos una cámara que sea compatible con la Raspberry, por lo que se utiliza una que se conecta al puerto CSI de la placa. La cámara que se usa con este programa se puede ver en la Figura 16.

Figura 16.

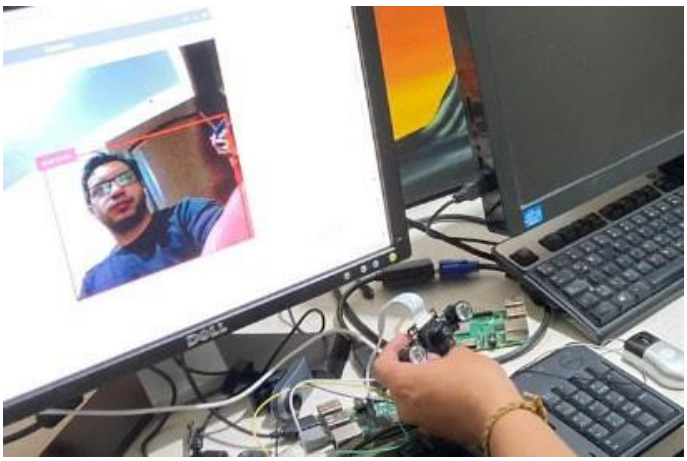
Arducam 5MP para puerto CSI de Raspberry Pi 4 modelo B

Nota: Se muestra en la figura la cámara que se usa para reconocimiento de objetos de YOLOv5s (Pricepulse, 2024). Extraído de: http://tinyurl.com/5n8tw2yp

Se muestran los resultados obtenidos a través de este modelo pre entrenado de reconocimiento de objetos en tiempo real. Determinando que el uso de un modelo pre-entrenado como lo es YOLO nos ayuda a desarrollar el objetivo que tiene el embebido. El uso de los modelos pre- entrenados por Ultralytics.

Figura 17.

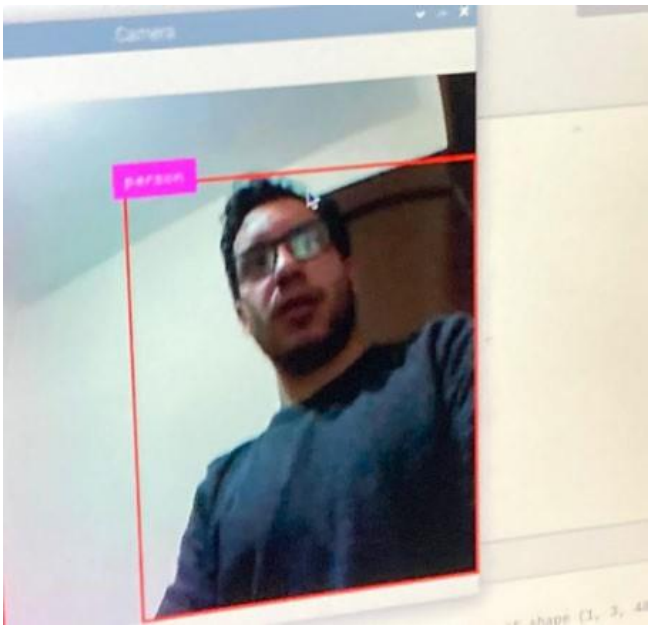
Funcionamiento de la detección de objetos del modelo YOLOv5s



Nota: Esta figura muestral reconocimiento de objetos con YOLOv5 con Arducam.

Figura 18.

Detección de personas con YOLOv5



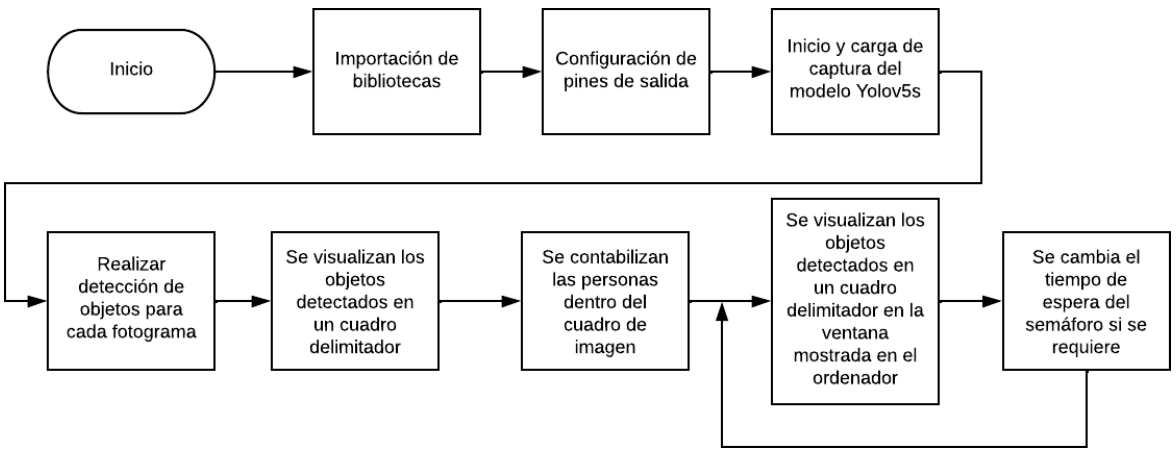
Nota: Esta figura muestra el funcionamiento de YOLOv5 para reconocimiento de personas.

Tras realizar un correcto reconocimiento de objetos queda poner un condicional para que se acople al funcionamiento de un semáforo inteligente, por lo que se coloca un contador de personas que cumplirá con un condicional, que, mientras sea mayor a 0 el sistema de preferencia a los peatones, por lo que interrumpirá de inmediato en el sistema de semaforización y pondrá en luz roja el semáforo para que se pueda dar acceso al cruce de peatones.

El diagrama de flujo para el programa con el modelo YOLOv5s para dar prioridad a los peatones esperando en el cruce semaforizado se observa en la Figura 19 y su correspondiente programación en el:

Figura 19.

Funcionamiento del modelo YOLOv5s para conteo de personas y cambio de tiempos de espera en semáforos



Este flujo describe la operación general del programa, desde la captura de video y detección de objetos hasta el envío de un pulso digital cuando se detecta un cierto número de personas para poder cambiar el tiempo de espera en el semáforo.

Caemos en cuenta de algunos problemas que surgen al utilizar el modelo YOLOv5 en nuestro Raspberry Pi 4. En primer lugar, que el uso de este modelo produjo un calentamiento excesivo del procesador de la Raspberry, por lo que tendríamos que realizar algunas modificaciones en cuanto al uso de esta. Luego se pudo observar que el uso de la Arducam nos causa problemas en cuanto a su uso. Ya que no podríamos mantenerla firme sin hacer uso de las manos. Además, el conector de la cámara era muy corto como para moverlo con facilidad. Se puede observar la Raspberry Pi 4 con la cámara ArduCam en la Figura 20.

Figura 20.

Raspberry Pi 4 conectada a ArduCam

Por lo que se implementa una cámara que se conecte mediante USB que facilite su colocación en el lugar necesario, con mejores características que la cámara mencionada anteriormente. La cámara seleccionada se muestra en la Figura 21 y sus características en la Tabla 4.

Figura 21.

Cámara usb nuroum v11

Nota: Se muestra en la figura la cámara web utilizada (nuroum, 2024). Extraído de https://nuroum.com/product/conference-camera-V11

Tabla 6.

Características cámara nuroum V11

Característica	Especificación
Apertura	F 2.2
Campo de visión	90°
Resolución de pantalla	QHD 1440p
Sistemas operativos compatibles	MAC/ Windows/ Linux/ Chrome OS
Fuente de alimentación	5V/ 0.5A

Nota: Se muestran las características importantes de la cámara nuroum V11 (nuroum, 2024). Extraído de: https://nuroum.com/product/conference-camera-V11

También se realiza el hallazgo de que el uso de una Raspberry Pi 4 con 4gb de memoria RAM tiene un pequeño retraso en cuanto al procesamiento de imágenes al correr el modelo de YOLOv5s, debido a las limitaciones del hardware.

"Para un rendimiento óptimo y una inferencia más rápida, se recomienda usar un GPU con mínimo 8GB de memoria. Los GPUs de NVIDIA con soporte de cuda son ideales para este propósito" (Ultralytics, 2024).

Para solucionar el problema del calentamiento se implementa un sistema de ventilación para la Raspberry mediante una carcasa protectora que viene con un ventilador incorporado, el cual va conectado a los pines GPIO para suministrarse de corriente y poder funcionar. En la Figura 22 se muestra el protector instalado.

Figura 22.

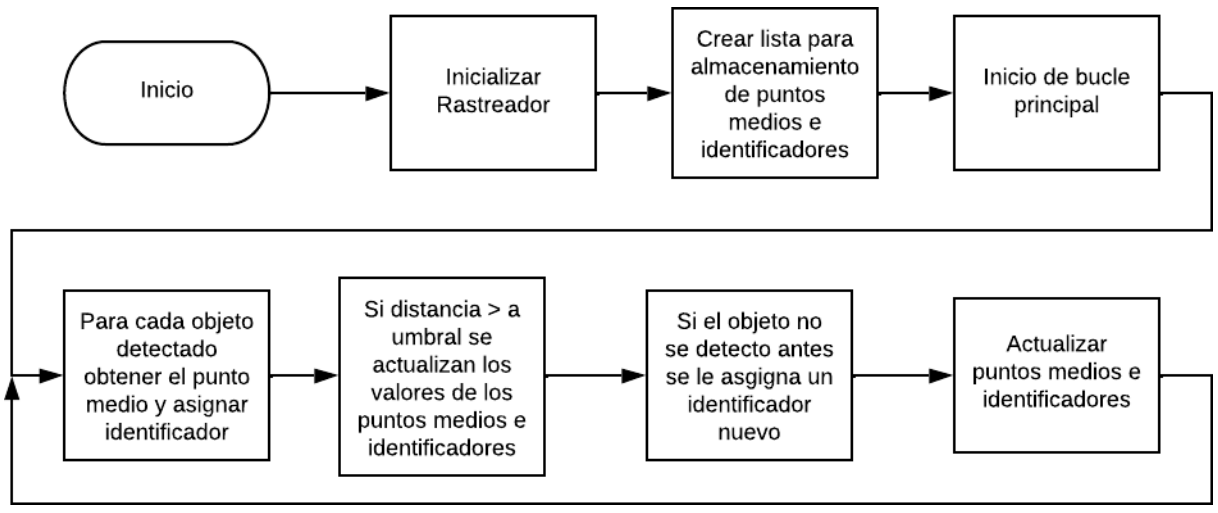
Raspberry con instalación de protector Vilros con ventilador incorporado.

Una vez que se soluciona el problema del calentamiento se evitan posibles daños en el hardware del microprocesador, además tenemos una mejora en el procesamiento de datos del dispositivo, ya que contamos con un sistema de ventilación adecuado para el procesador de la tarjeta.

Posterior a esto se debe desarrollar con un programa que permita realizar un seguimiento de los objetos detectados y que tengan un identificador diferente según las coordenadas que indican la posición del objeto detectado, en la ventana que se crea para mostrar la imagen del video. El diagrama de flujo de este programa se puede visualizar en la Figura 23 y sus resultados en la Figura 24.

Figura 23.

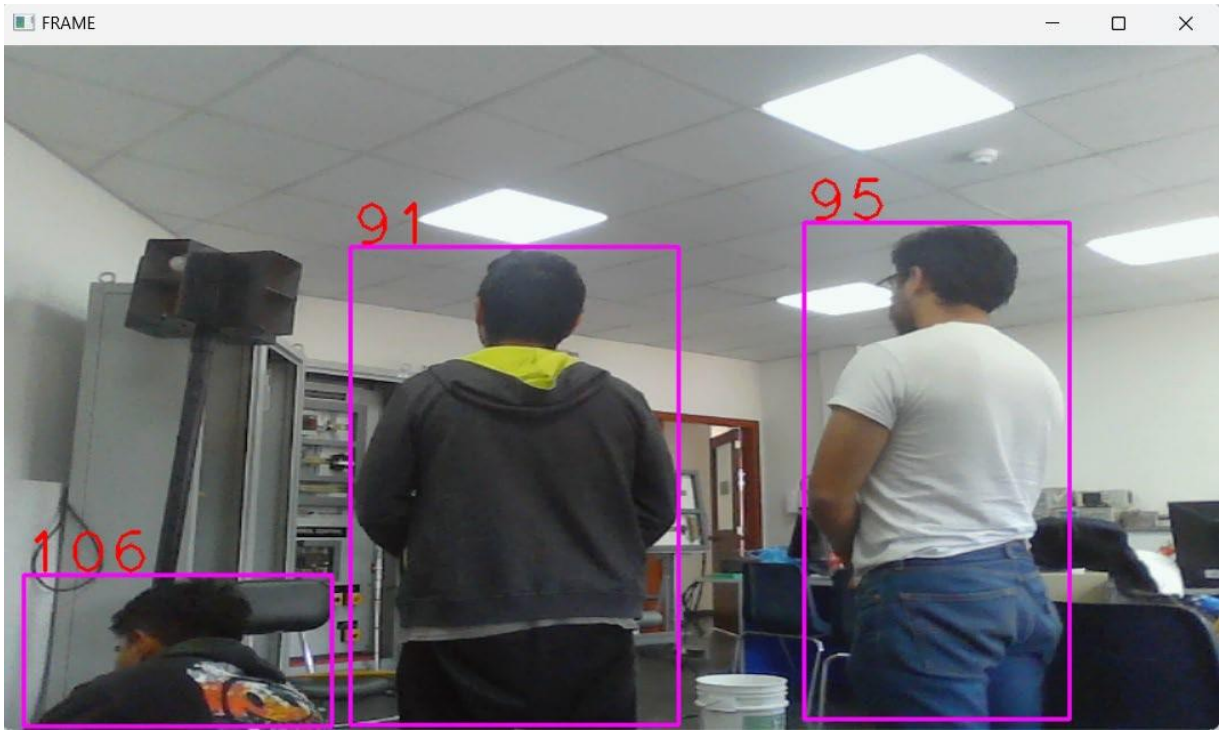
Diagrama de funcionamiento del programa rastreador de objetos



Se muestra el algoritmo completo del programa identificador de personas en el Anexo D.

Figura 24.

Detección de personas con YOLOv5

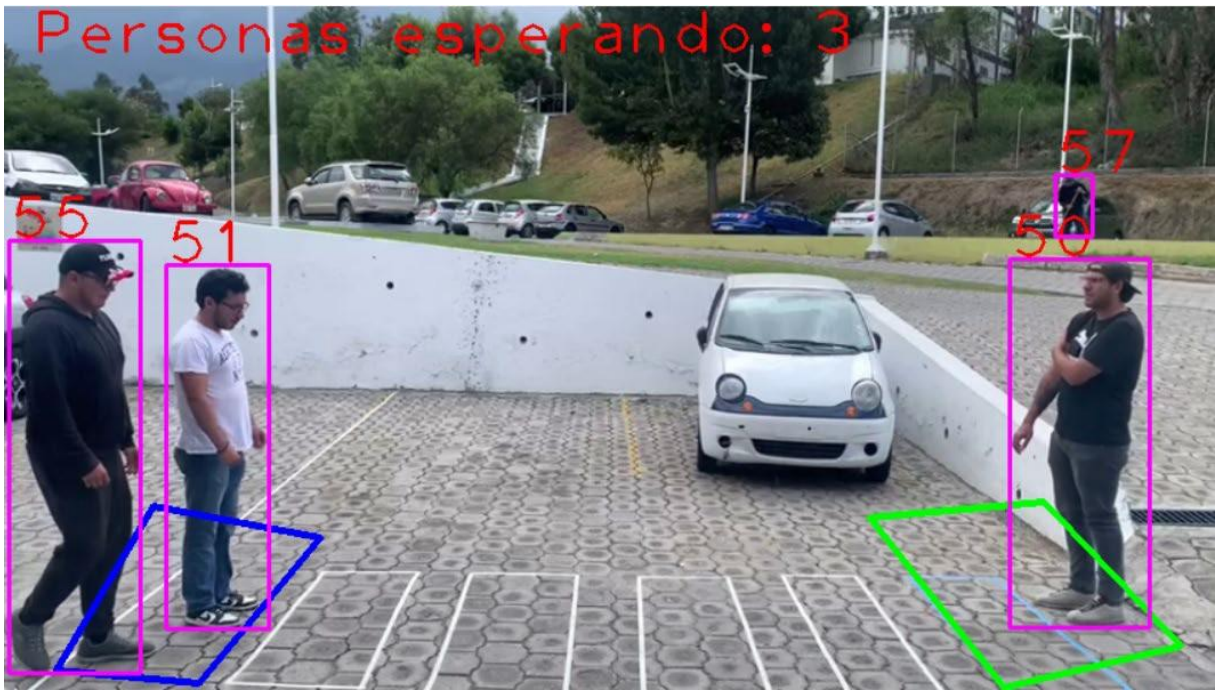


En la Figura anterior se puede observar que cada persona cuenta con un cuadro delimitador y un identificador único para cada persona que depende de las coordenadas en las cuales fue detectada. Además de contar con una función para el seguimiento de la persona para que no se asigne un identificador nuevo a una persona que ya fue identificada en la ventana donde se muestra la imagen. Con esto se descarta que el contador confunda a objetos comunes o animales con personas.

Se diseña un sistema de conteo que funcione de manera precisa. Gracias a la biblioteca de OpenCV seleccionamos el método adecuado, que es dibujar una sección poligonal de tal manera que forme un área donde la gente espera a cruzar la calle por el paso peatonal. Si la gente se encuentra sobre esta sección el contador aumentará, si no están dentro las eliminará. Se muestra en la Figura 25 y el algoritmo completo en el Anexo E.

Figura 25.

Conteo de personas mediante OpenCV y YOLOv5s



Nota: La imagen muestra un cuadro de video simulando el flujo en un paso peatonal, se encuentran tres personas sobre las áreas dibujadas con la librería OpenCV.

Ahora que el contador de personas es el adecuado para el proyecto se debe poner los condicionales para cambiar los tiempos de espera en el semáforo, lo cual se hará utilizando los pines de entrada y salida de propósito general con los que cuenta la Raspberry pi 4.

Lo primero que se hace es importar la biblioteca necesaria para poder utilizar los pines GPIO en conjunto con el programa de detección y conteo de personas. Se muestra la importación para este propósito en la Figura 26.

Figura 26.

Importación de biblioteca necesaria para el uso de los pines GPIO

```
import RPi.GPIO as GPIO
```

Luego se declara un pin de salida e indicar que se usará el pin 17 de la Raspberry, como se observa en la Figura 27.

Figura 27.

Declaración de pin de salida

```
pin_gpio = 17
GPIO.setmode(GPIO.BCM)
GPIO.setup(pin_gpio, GPIO.OUT)
```

Declaración de pin de salida

Una vez declarado el pin de salida escribiremos el condicional para que solamente se cuenten a las personas que se encuentran dentro del área determinada, para esto se hace la programación que se muestra en la **Figura 28**.

Figura 28.

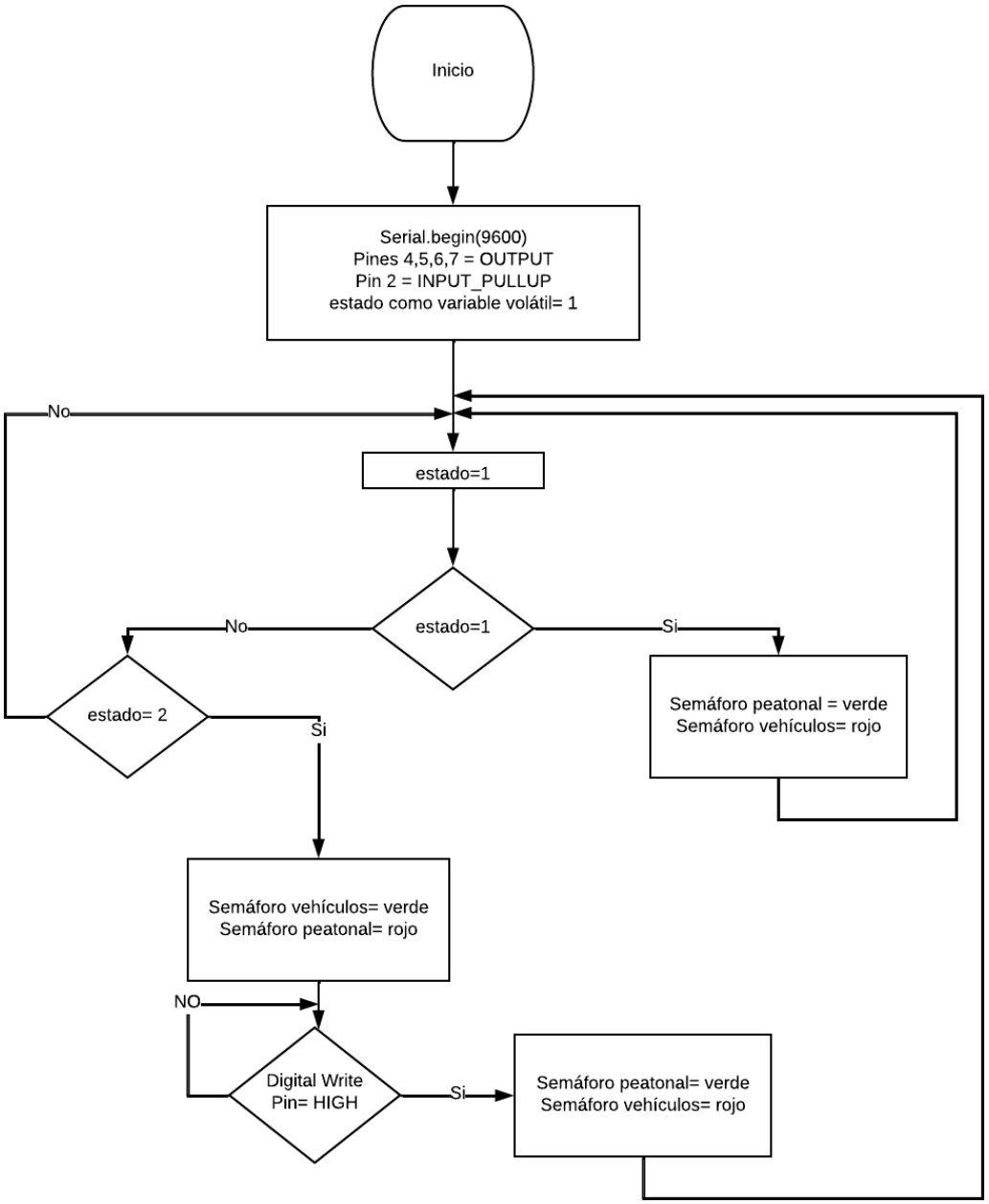
Declaración de salida digital positiva para interrupción en el semáforo

```
if personas_esperando>=3:
    GPIO.output(pin_gpio, GPIO.HIGH)
```

Una vez que contemos esta salida digital se puede realizar la acción de cambiar los tiempos de espera dándole preferencia al peatón. Se debe utilizar un sistema que haga el trabajo normal de un semáforo, pero que pueda leer esta salida digital de la Raspberry e interrumpir el flujo normal de funcionamiento de este. Para hacerlo funcionar usaremos el microcontrolador Arduino Uno R3. Se muestra en la Figura 29 el diagrama de flujo del sistema de semaforización que funciona con interrupciones. El programa completo se muestra en el.

Figura 29.

Diagrama de funcionamiento de algoritmo de semaforización en Arduino

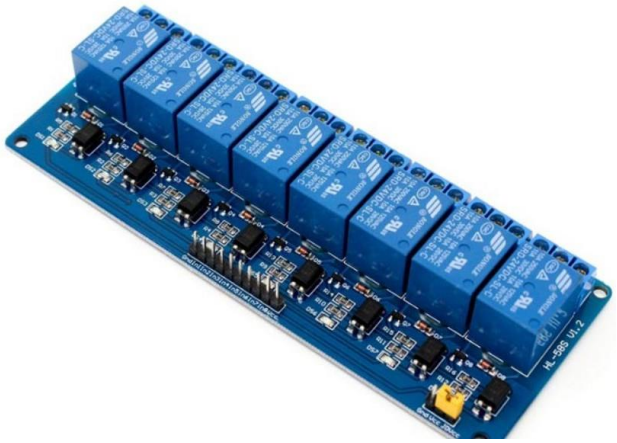


Se debe mencionar que el semáforo en verde para los vehículos siempre estará en verde y solamente se pondrá en rojo para dar paso a individuos en el caso de que el contador indique tres o más que están esperando a cruzar luego volverá a dar paso a los vehículos comenzando de nuevo el bucle.

Como se utilizarán las salidas digitales del Arduino para controlar bombillas de semáforo, que en este caso serán bombillas de luz comunes de corriente alterna. Se debe implementar un circuito de control que dará las instrucciones a un circuito de potencia. Para este cometido utilizaremos un circuito con relés que se activará con las señales digitales enviadas por el microcontrolador para suministrar de energía eléctrica a las bombillas del semáforo. El módulo cuenta con 8 relés se presenta en la Figura 30 y sus características se muestran en la Tabla 6.

Figura 30.

Módulo de 8 relés



Nota: La figura muestra un módulo de 8 relés con optoacopladores (Components 101, 2021).

Una vez seleccionado se procede al diseño del circuito de control de semáforo controlado por salidas digitales desde Arduino, que interrumpen su funcionamiento normal según el conteo de personas.

Tabla 7.

Características del módulo de relé de 8 Canales

Característica	Valor
Voltaje de funcionamiento	5v DC
Señal de control	3.3 o 5v DC
Canales	8
Relé	SRD-24VDC-SL-C
Capacidad máxima de potencia	250 VAC/10 A, 30VDC/10A
Tiempo de reacción	5-10 milisegundos
Leds indicadores	8

Nota: Se muestran las características adaptadas del manual de usuario de 8 Channel 5V Optical Isolated Relay Module (Components 101, 2021).

Circuito de Control del sistema

Para generar señales digitales con niveles de voltaje adecuados para la escritura por el microcontrolador, se desarrolla un circuito de control que responde a:

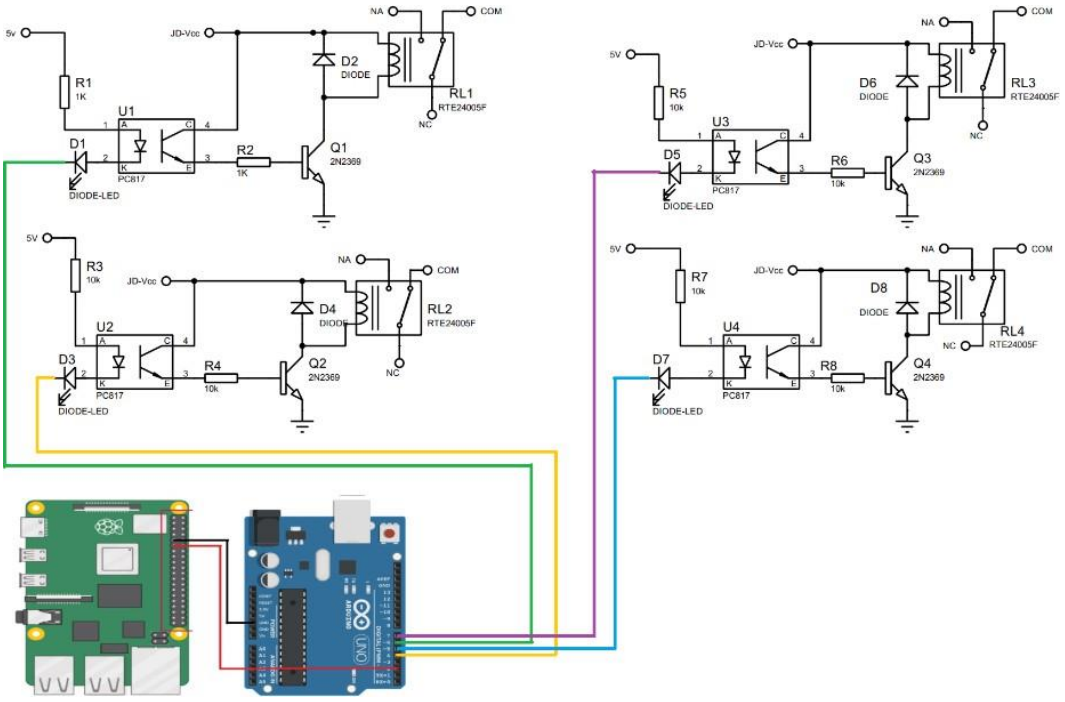
$$SSooñffff\ ffffddffoofffff = \begin{cases} 5vv\ DDCC, & rrooffoo\ ccoocc\ ooccoorrffffff = ffccooffvvffffoo \\ 0vv\ DDCC, & rrooffoo\ ccoocc\ ooccoorrffffff = ffooCCffccooffvvffffoo \end{cases}$$

El circuito de relés funciona en base a estas señales enviadas por el microcontrolador, además las señales digitales enviadas desde la Raspberry funcionan de la misma manera, solo que en ese caso la interpretación de esa señal la hace el microcontrolador.

Para interrumpir el flujo normal de programación del semáforo se hacen las conexiones mostradas en la Figura 31 para comunicar el microprocesador con la placa Arduino Uno R3 en el caso que un grupo de personas requiera de prioridad en el paso peatonal.

Figura 31.

Circuito de control de semáforo



Se puede apreciar en la Figura 32 el diseño final del circuito de control.

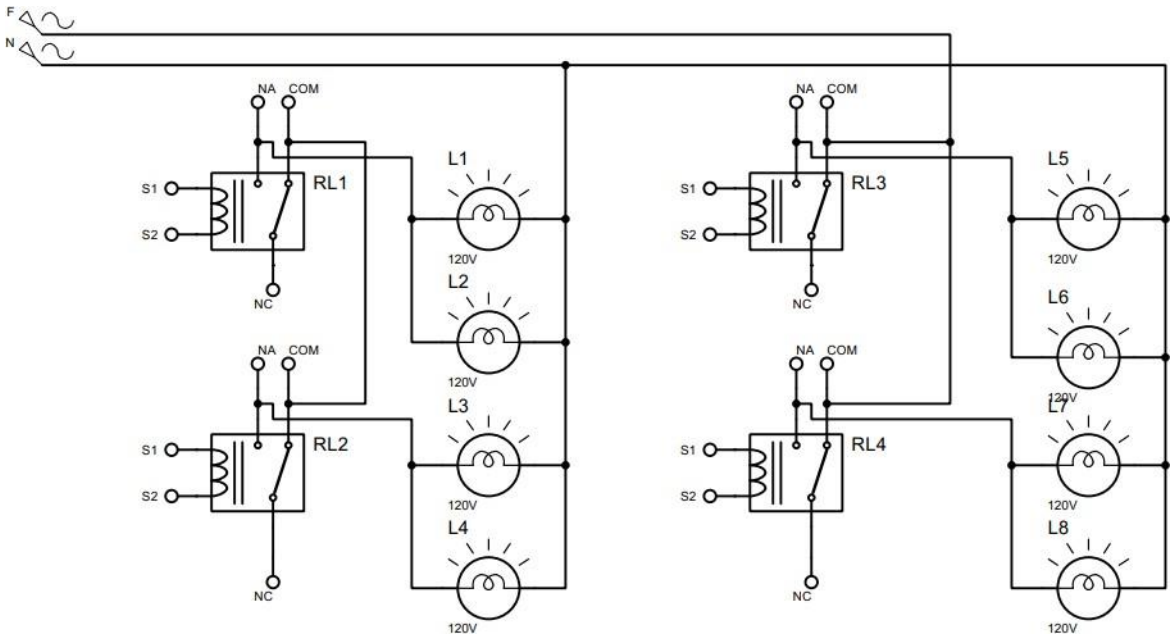
Figura 32.

Diseño final de circuito de control

El circuito de control se encarga de controlar el flujo normal de programación del semáforo, además de actuar con las interrupciones detectadas con el microcontrolador. Se observa el circuito de potencia en la Figura 33.

Figura 33.

Circuito de potencia utilizado para el funcionamiento del semáforo.



Como se puede apreciar en la Figura anterior el semáforo debe funcionar de tal forma que cuando la luz verde o roja para los vehículos se encienda, deberá hacerlo para el cruce en ambos sentidos del flujo vehicular. Funciona de la misma forma para el flujo de individuos, por lo que estaremos controlando 4 sentidos diferentes de movimiento de personas y automóviles.

De esta forma se definen todos los componentes electrónicos que se necesitan para el correcto funcionamiento de este prototipo. Para la estructura del semáforo se utiliza acero de acero galvanizado y perfiles de aluminio para las cajas que recubren los focos, en la parte delantera de estas se colocó acrílico.

Con esta información se desarrolla la Tabla 7 en la que especificamos los diferentes equipos, instrumentos utilizados para el desarrollo del prototipo de semaforización.

Tabla 8.

Lista de materiales para desarrollo de prototipo

Ítem	Cantidad	Costo c/u ($)	Total
Raspberry Pi 4 modelo B	1	$120	$120
Cámara nuroum V11	1	$32	$32
Protector Vilros para Raspberry	1	$15	$15
Cargador de pared 5v 1ª	2	$5	$10
Conectores jumper macho hembra 25 unidades	1	$5	$5
Arduino Uno R3	1	$12	$12
Módulo de 8 relés	1	$12	$12
Breaker 14Amp 2P	1	$18	$18
Bombillas LED A60 9W	8	$5.52	$5.52
Plancha de acrílico	1	$18	$18
Riel Din	1	$3.10	$3.10
Plancha de tool 2mm	1	$26	$26
Conductor de cobre AWG 12 (1 rollo)	1	$62	$62
Conductor de cobre AWG 18 (1 rollo)	1	$17.50	$17.50
Tubo de acero galvanizado 3 x ¼ pulgadas	1	$10	$10
Caja de paso 10 x 10cm	1	$1	$1
Caja de paso 20 x 25 cm	1	$6	$6
Total			$373.12

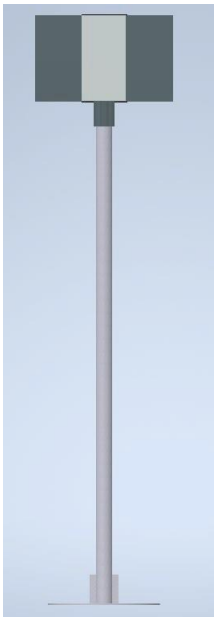
Diseño del prototipo de semáforo inteligente en software CAD Inventor Professional.

Para el diseño del prototipo se utilizaron de referencia los materiales con sus respectivas medidas para obtener un plano de cómo será la construcción del proyecto en físico, para este

cometido utilizamos el software de diseño y simulación CAD Inventor Professional de Autodesk. Se realiza un bosquejo de cada parte en 3D con sus medidas exactas para formar la estructura del semáforo, esta se puede apreciar en la Figura 34, posteriormente se realizan los planos de construcción del semáforo dentro del mismo software los cuales se mestra en la Figura 35.

Figura 34.

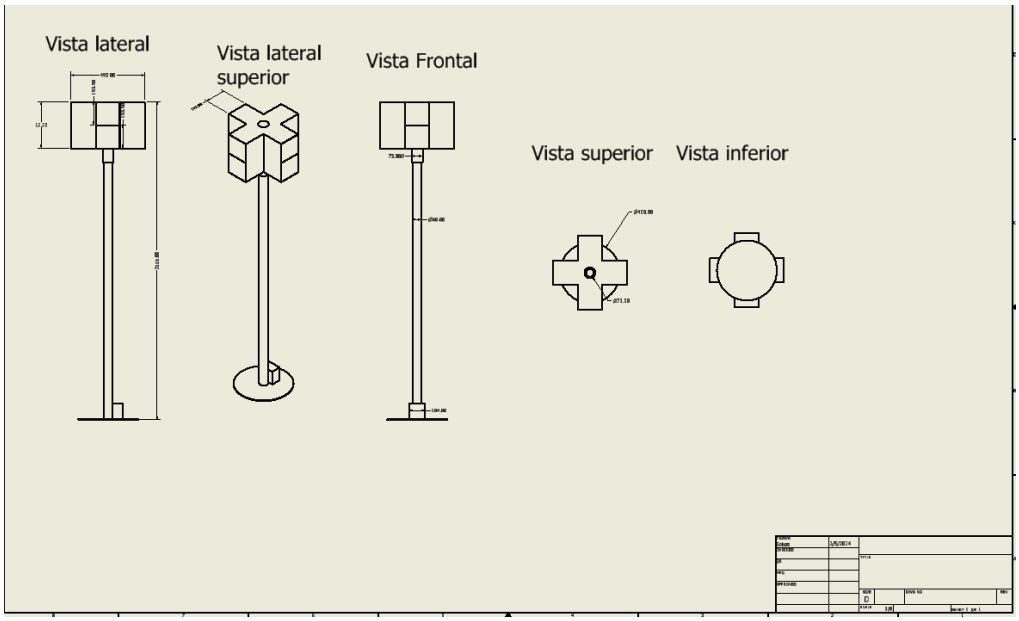
Vista frontal de estructura de semáforo en Inventor



En la siguiente Figura se puede observar un plano con las medidas reales del prototipo en milímetros.

Figura 35.

Plano para construcción de estructura de semáforo diseñado en Inventor



A partir de este plano se procede a realizar la construcción del prototipo en físico tomando en cuenta que la cámara con la que se realiza el proceso de reconocimiento de personas se va a colocar en un lugar distante a la estructura del semáforo. Se puede apreciar la estructura final del prototipo en la Figura 36.

Figura 36.

Prototipo de semáforo final

Control y monitoreo de Raspberry desde dispositivo externo mediante IoT

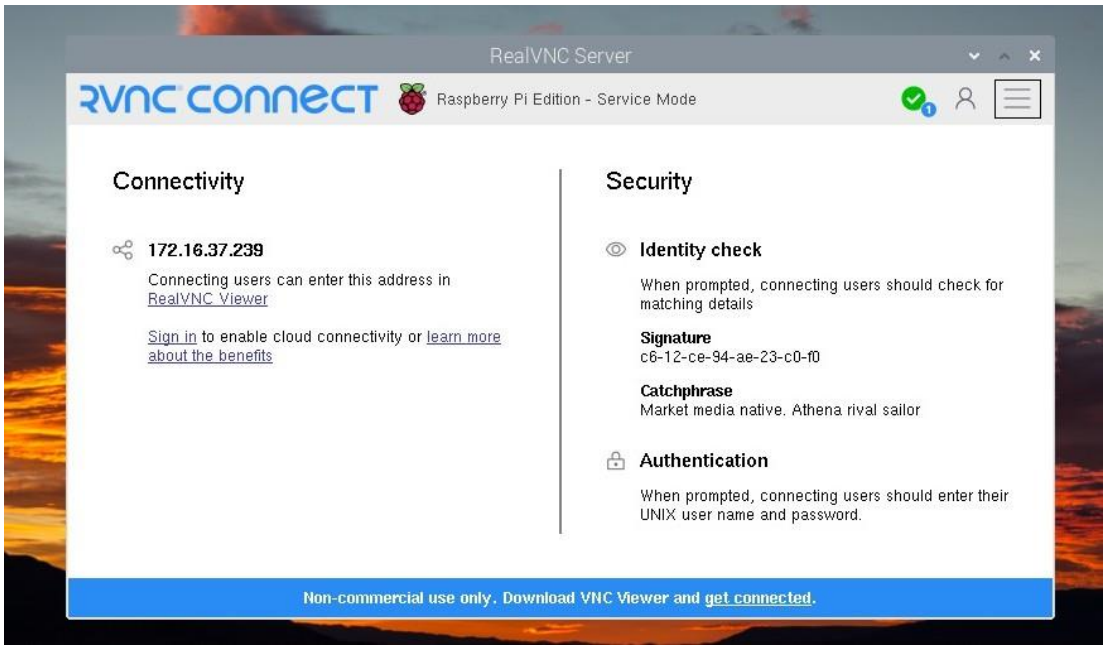
El procesamiento de imágenes que tiene el embebido del proyecto puede ser monitoreado en tiempo real mediante conexión inalámbrica a través de WI-FI, facilitando la

visualización de la de visión artificial a través del celular o un computador. Esto significa que el prototipo está conectado al internet.

Se puede conectar al microprocesador Raspberry Pi desde otro dispositivo, lo único que se necesita es conocer la IP asignada al momento de conectarse a Internet (Raspberry Pi, 2024). Es muy fácil saber la IP que se asigna cuando se conecta al dispositivo en una red de área local, ya que vamos a tener preinstalada la aplicación VNC en nuestra Raspberry, si se apertura aparece en imagen la dirección IP que se asigna al dispositivo al estar conectada a la red tal y como se muestra en la Figura 37.

Figura 37.

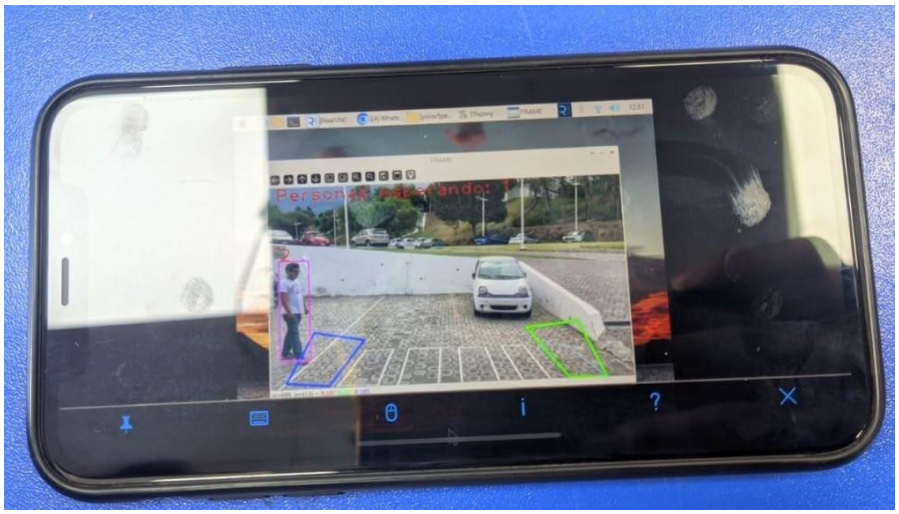
VNC en Raspberry Pi



Se debe instalar VNC viewer ya sea por el App Store o Play Store para poder conectarse a un dispositivo móvil, o ingresar a la página oficial de descarga de VNC para poder utilizarlo en un ordenador. En nuestro caso estaremos utilizando la aplicación desde IOS. Se observa en la Figura 38 la conexión con la Raspberry.

Figura 38.

Conexión a Raspberry Pi desde dispositivo móvil



Este software además de permitir el monitoreo en tiempo real desde el dispositivo móvil, permite controlarlo. Por lo que en el caso de tener que modificar el código de programación o de que se requiera visualizar la imagen de la cámara se realizará fácilmente.

Al haber definido el software y hardware implementado en el desarrollo de este proyecto, realizamos una prueba de laboratorio la cual se puede visualizar en el Anexo F, determinando que el prototipo es apto para realizar pruebas de campo, ya que el embebido indica que es apto para funcionar en una situación real. Se implementa en el cruce peatonal principal del campus, el cual se muestra en la Figura 39.

Figura 39.

Prototipo de semáforo inteligente implementado en campus Miguel de Cervantes

Verificación

El prototipo fue colocado el día 21 de febrero del 2024, de tal forma que las personas que vayan a cruzar puedan observar la luz indicadora desde cualquiera de los dos sentidos del

paso peatonal, los vehículos podrán verla desde ambos sentidos de circulación de la calle, por lo que se cumple con dar una instrucción a los individuos formados en una cola de espera para el cruce.

Se posiciona la cámara de tal forma que pueda reconocer y contar personas según la configuración de la programación, cuando una persona se encuentre esperando a cruzar la calle por el paso peatonal enviando una señal al actuador para darle preferencia y cambiar la luz indicadora para que pueda cruzar. En la Figura 40 se muestra el lugar donde se colocamos la cámara.

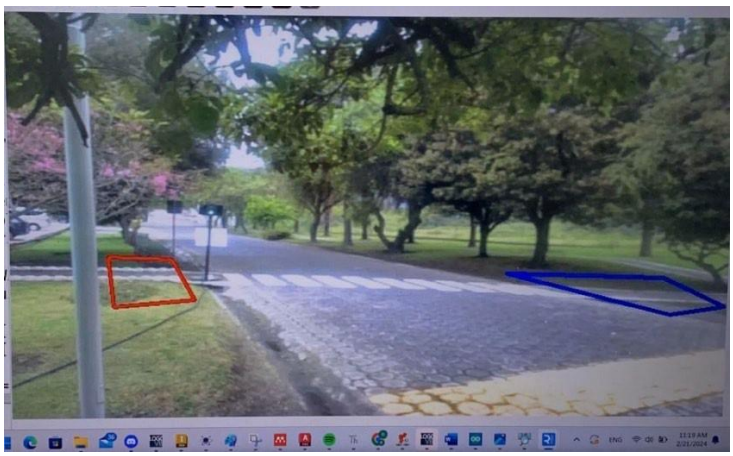
Figura 40.

Posicionamiento de la cámara para identificación y conteo de personas

Mientras el microprocesador se encuentre conectado al internet se puede visualizar en tiempo real la imagen que muestra el comportamiento de los peatones al momento de interactuar con el sistema de control de semaforización. Se puede visualizar en la Figura 41 la imagen que muestra la cámara cuando se inicia el programa de conteo de personas.

Figura 41.

Monitoreo del algoritmo mediante VNC en dispositivo externo



Resultados

Con el precedente del alto flujo de tráfico vehicular y peatonal en el campus Miguel de Cervantes de la Universidad Internacional SEK; se cumple con la propuesta de un sistema que optimice la circulación dentro del mismo, modificando los tiempos de espera según el flujo de personas, mediante la implementación del prototipo de semáforo inteligente desarrollado en el presente trabajo.

Se obtienen los resultados a través de dos pruebas de campo detalladas a continuación.

Prueba de campo 1: Funcionamiento de prototipo en ambiente simulado

En esta prueba de campo, se utiliza el prototipo para verificar su funcionalidad en un ambiente simulado. A su vez se subdivide en dos simulacros diferentes.

En el primero, los peatones hacían uso del sistema colocándose al comienzo de cada lado del cruce peatonal sin la presencia de vehículos. Se observa la funcionalidad del sistema del primer simulacro en las siguientes dos Figuras.

Figura 42.

Proceso previo al cambio de tiempo de espera en el semáforo

Nota: Se muestra en la Figura una persona esperando a cruzar y dos llegando a la sección de espera sin la presencia de vehículos.

Figura 43.

Cambio de tiempo de espera de semáforo

Nota: Se muestra como el semáforo cambia la luz indicadora cuando tres personas esperan a cruzar la calle sin la presencia de vehículos.

El segundo simulacro involucra la presencia de vehículos y peatones esperando a cruzar, los mismos tenían la indicación de colocarse en un punto especifico del cruce de manera intencional, donde el algoritmo detecte mediante visión artificial con una cámara a tres o más personas esperando a cruzar por el paso peatonal para cambiar la luz indicadora y darle preferencia al peatón en todo momento. Se observa lo mencionado en las siguientes Figuras.

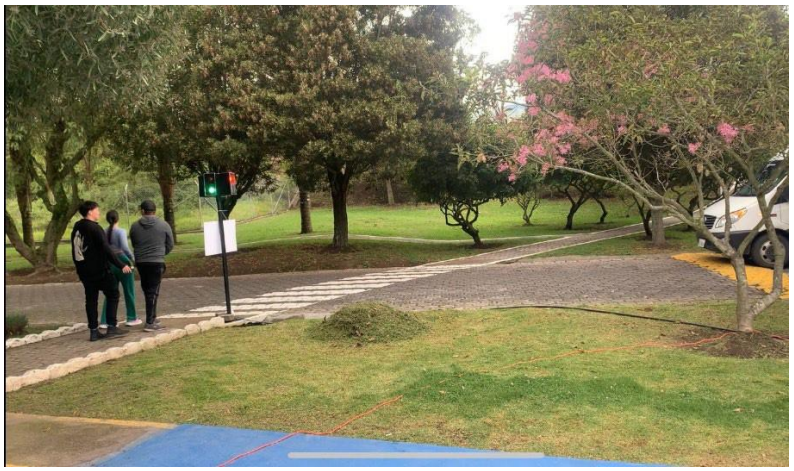
Figura 44.

Proceso previo al cambio de tiempo de espera en el semáforo con la presencia de vehículos.

Nota: Se muestra el proceso previo al cambio de la luz indicadora del semáforo cuando tres personas esperan a cruzar la calle con flujo vehicular.

Figura 45.

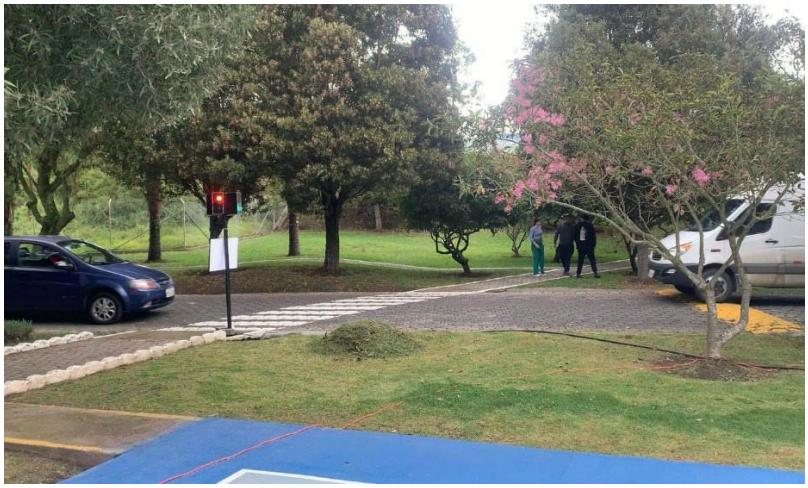
Cambio de tiempo de espera de semáforo con la presencia de vehículos



Nota: Se muestra como el semáforo cambia la luz indicadora cuando tres personas esperan a cruzar la calle con flujo vehicular.

Figura 46.

Cambio de luz indicadora para paso de vehículos

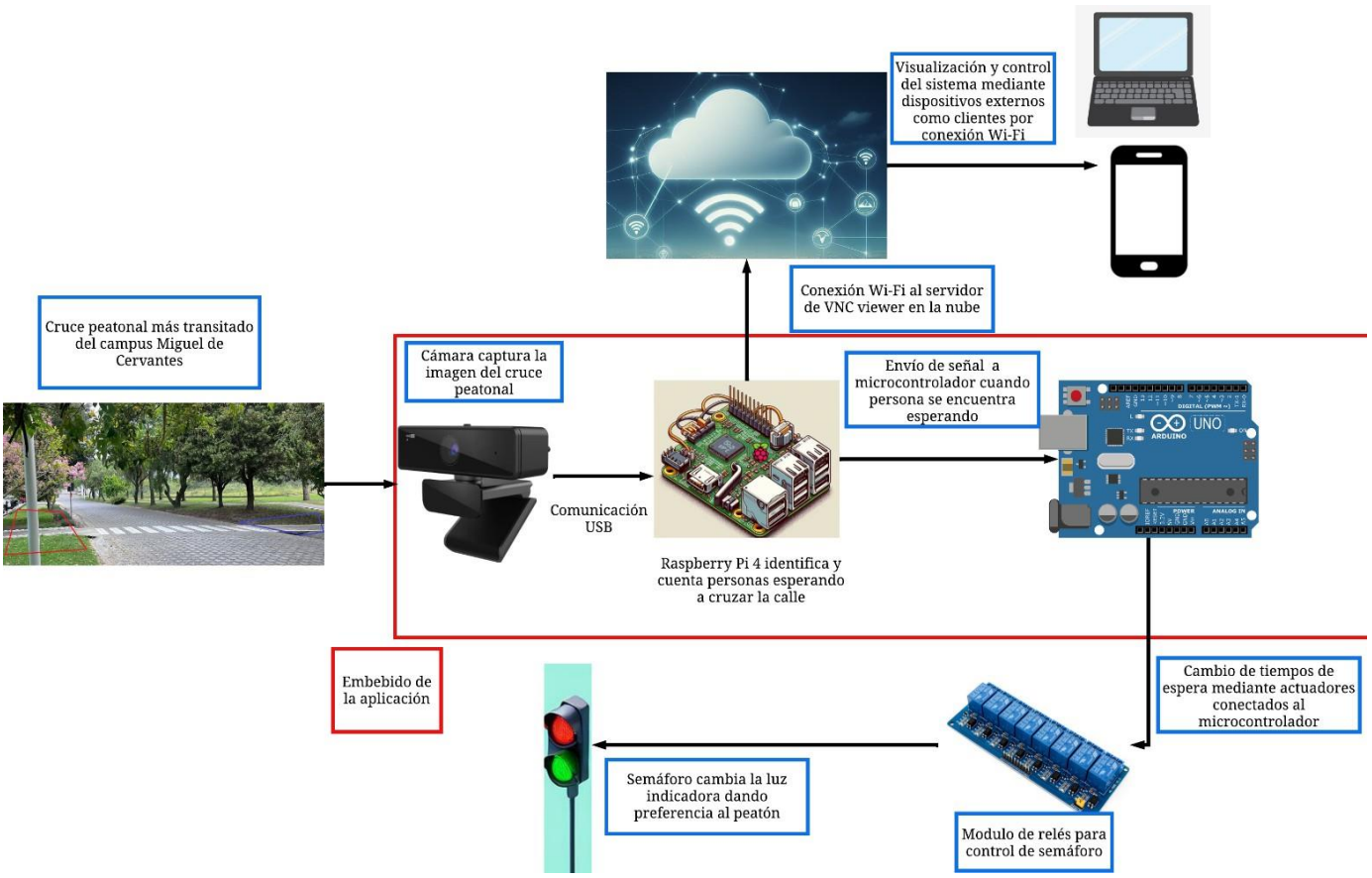


Nota: Se muestra como el semáforo cambia la luz indicadora, dando paso a los vehículos cuando terminan de cruzar los peatones.

Tras verificar la funcionalidad del prototipo en un entorno simulado con la presencia de peatones y vehículos, se determina que el sistema es apto para su uso en una situación real de flujo de tráfico en hora pico. Se presenta la arquitectura final del prototipo en la Figura 47. Bajo la cual se rige el funcionamiento de la segunda prueba de campo.

Figura 47.

Arquitectura final del prototipo de semaforización inteligente



Nota: En la Figura se muestra la arquitectura que rige el funcionamiento del prototipo de semáforo inteligente en etapa de implementación en situación real.

Prueba de campo 2: Funcionamiento de prototipo en situación real.

El funcionamiento del sistema en una situación de tráfico real se realiza el día 22 de febrero del 2024, en la que nuevamente se modifica el algoritmo, para que cambie el tiempo de espera dando preferencia al peatón, aunque solamente haya una persona esperando a cruzar. Se puede observar en la Figura 47 el prototipo en la prueba de campo mencionada.

Figura 48.

Caso real de funcionamiento

Nota: Se muestra un fotograma del video mencionado en el Anexo H.

La segunda prueba de campo indica que las variables dependientes en el sistema para un correcto funcionamiento son las siguientes:

- Vehículos circulando sobre el paso peatonal.
- Personas cruzando por el paso peatonal.
- El semáforo.

Se hace la toma de un video que muestra el comportamiento del sistema mientras todas las variables interactúan en una situación de tráfico real. Se realiza un análisis mediante observación del comportamiento de todas las variables dependientes del sistema. Se puede apreciar el video mencionado en el Anexo H.

Se realiza un análisis de los resultados de la segunda prueba de campo, mediante la observación del video mencionado, que captura el comportamiento de las tres variables cuando el sistema se encuentra en funcionamiento y se determina que existen tres escenarios explicados a continuación.

Escenario 1: Interacción de personas con el prototipo.

Este escenario explica la interacción del prototipo con las personas que cruzaron por el paso peatonal sin la interacción de vehículos. Se determina que cruzaron 33 personas mientras el sistema de semaforización inteligente estaba en funcionamiento, de las cuales 27 se beneficiaron por la preferencia que obtuvieron del mismo, debido a que la luz indicadora les dio el paso, siendo el 81.8% del total.

El prototipo de semaforización le brinda al peatón la seguridad de cruzar, ya que la luz indicadora muestra a la persona que puede hacer uso del cruce peatonal con la confianza de que los vehículos tienen la indicación de parar. El propósito de este sistema es dar preferencia al peatón en todo momento, independientemente si se encuentran vehículos circulando por la calle.

El sistema tuvo dos errores cuantificables en este escenario y se puede dividir en; número veces que no cambió la luz indicadora al momento de haber una persona esperando, y número de veces que hubo demoras en el cambio de la luz indicadora. Siendo 3 y 3 respectivamente. Obteniendo así un 18.18% de error del prototipo en este escenario.

Escenario 2: Interacción de automóviles con el prototipo.

Este escenario explica la interacción que tienen los vehículos que circulan sobre el paso peatonal cuando el sistema no detecta personas esperando a cruzar, ya que los vehículos también se benefician de que el sistema de semaforización de la indicación de que deben parar únicamente cuando haya una persona en el área de espera del paso peatonal, por lo tanto, cuando el algoritmo de visión artificial no detecte individuos esperando a cruzar por el paso peatonal, el vehículo no tendrá que parar en ningún momento.

El número de vehículos que se beneficiaron por el prototipo de semaforización en este escenario son 145. Algo que se puede destacar es que no existieron errores en estas condiciones

Escenario 3: Interacción de automóviles en conjunto con peatones.

Este escenario nos indica cuantas personas fueron beneficiadas por el sistema al momento de darles preferencia parando un vehículo, permitiéndoles el cruce por el paso peatonal. Se determina mediante observación del video mencionado en el Anexo H que 19 personas se beneficiaron por la implementación del prototipo. En la Tabla 9 se muestra el tiempo de espera por persona para cruzar por el paso peatonal habiendo carros en circulación.

Tabla 9.

Tiempos de espera por persona para cruce por paso peatonal principal con implementación de prototipo de semaforización inteligente en el campus Miguel de Cervantes

Número de peatones en el grupo esperando a cruzar	Tiempo de espera	Momento de tiempo en el video

1	1,50 segundos	3:20
1	2,47 segundos	10:05
2	1,34 segundos	13:57
2	2,06 segundos	14:07
2	1,15 segundos	48:50
1	4,02 segundos	55:40
1	2,10 segundos	55:55
1	3,32 segundos	1:00:24
2	2,79 segundos	1:00:50
1	3,11 segundos	1:02:50
2	1,72 segundos	1:05:53
2	2,77 segundos	1:06:10
1	2,42 segundos	1:11:02
Total: 19 personas	Tiempo promedio de espera por persona: 2.24 segundos	

Nota: Estos datos fueron extraídos mediante observación del video mostrado en el Anexo H, muestra el tiempo de espera por persona para cruzar por el paso peatonal con la implementación del prototipo de semaforización inteligente dentro del campus.

Cálculo de porcentaje de mejora con la implementación del prototipo dentro del campus.

Con el tiempo promedio de cruce por persona en el paso peatonal principal del campus Miguel de Cervantes de la Universidad Internacional SEK, y el promedio de tiempo de espera del cruce sin la implementación del sistema desarrollado en este proyecto. Se calcula el porcentaje de la eficiencia del tiempo de espera con la siguiente formula:

$$EEfffccffooccccffff\ ffoo\ mmooooorrff = \left(1 - \frac{TT_{ffüffffff}}{TT_{üffüüüüffff}}\right) \times 100\%$$

Donde:

$TT_{üffüüüüffff}$ = Tiempo de espera promedio por persona sin la implementación del prototipo.

$TT_{ffüffffff}$ = Tiempo de espera promedio por persona con la implementación del prototipo.

Desarrollo:

$TT_{üffüüüüffff}$ = 4.51 tomado de la Tabla 5.

$TT_{üffüüüüffff}$ = 2.24 tomado de la Tabla 9.

Tenemos que:

$$EEfffccffooccccffff\ ffoo\ mmooooooorrff = \left(1 - \frac{2.24}{4.51}\right) \times 100\%$$

$$EEfffccffooccccffff\ \ ffoo\ \ mmooooooorrff\ \ = \ 50.4\%$$

Una vez obtenido el porcentaje de eficiencia de los tiempos de espera del peatón para cruzar por el paso peatonal deteniendo un vehículo, satisfacemos la hipótesis planteada al inicio de este proyecto que fue obtener un porcentaje de mejora del 15% en los tiempos de espera en el cruce peatonal principal dentro del campus.

En este escenario también podemos determinar que el 36.54% de personas que hicieron uso del sistema, se beneficiaron teniendo preferencia sobre un automóvil, por lo que se espera que en días diferentes se tenga un porcentaje similar de personas beneficiadas de esta forma.

También se determina que un 10 % del total de automóviles que pasaron por encima del cruce peatonal pararon dando preferencia a las personas esperando. Siendo un porcentaje bajo teniendo en cuenta que pasaron 160 vehículos. Esto nos indica que, aunque el sistema siempre considere al peatón con preferencia, el flujo de vehículos sigue siendo muy alto.

Algo muy importante a mencionar es que el prototipo no tuvo ningún fallo al momento de interactuar con vehículos y peatones. Por lo que siempre se tuvo en cuenta la seguridad y preferencia del peatón.

Discusión

Las pruebas de campo arrojaron resultados favorecedores al momento de implementar el prototipo de semaforización inteligente dentro del campus Miguel de Cervantes. En primer lugar, la primera prueba indicó la funcionalidad del prototipo en situaciones simuladas, que involucraban a personas y vehículos. Lo que nos ayuda a confirmar que el dimensionamiento del sistema es el adecuado para su implementación y funcionamiento en situaciones reales. Determinando la arquitectura final del sistema con su correspondiente embebido.

Posteriormente, la segunda prueba de campo nos permite identificar que existen tres escenarios posibles cuando el prototipo se encuentra en funcionamiento en una situación real.

Cuando solamente interactúan vehículos, con la interacción de peatones y por último ambos interactuando con el sistema al mismo tiempo.

En el primer escenario se determina que 33 personas interactuaron con el sistema esperando y cruzando por el paso peatonal. Se obtiene que 27 personas fueron beneficiadas por el sistema en este primer escenario, un 81.8% del total. Es muy importante mencionar que este beneficio se caracteriza por la seguridad que brinda el sistema al peatón, ya que la luz indicadora da la señal para que pueda pasar, dándole confianza al cruzar ya que la luz indicadora para los automóviles estará en rojo por lo que no podrán pasar.

El porcentaje de error se define por las demoras y las veces que el semáforo no cambio para darle la preferencia al peatón dando un 18.2% de error. Siendo un error relativamente bajo. Sin embargo, se reconoce la oportunidad de mejorar el prototipo tanto en hardware como software para reducir estos errores.

El segundo escenario solo involucra vehículos con el sistema, lo importante de esto es que también se benefician por el funcionamiento de este. Ya que, comparado con el sistema tradicional de semaforización implementado en las calles de todo el mundo, en el que los vehículos deben parar de manera obligatoriamente independientemente de la haber peatones esperando a cruzar la calle, el nuestro es mucho mejor. Reduce los tiempos de espera innecesarios a cero, ya que un vehículo solamente se detendrá si un peatón se encuentra esperando a cruzar por la calle, es decir cuando es estrictamente necesario.

Para el tercer escenario, arrojó resultados críticos que involucran el comportamiento de peatones y vehículos a través del prototipo de semáforo inteligente. Por lo tanto, el tiempo promedio de espera por persona utilizado fue primordial en la comparación del desempeño general. A través de un análisis se identificó una mejora significativa de la eficiencia del peatón cuando el prototipo estaba operativo y monitoreaba el flujo vehicular. La mejora respecto al tiempo descrito se compara cuando los vehículos circulan libremente sin ninguna restricción o prototipo operativo. En este caso, la eficiencia en los tiempos de espera para el peatón en presencia de vehículos fue del 50.4%, mejorando el 15% estimado en la hipótesis del proyecto.

También, determinamos que el prototipo no tuvo ningún error mientras funcionaba con la interacción de vehículos y peatones al mismo tiempo, garantizando la seguridad de las personas dentro del campus al máximo.

Es muy importante mencionar que el prototipo de semaforización inteligente fue instalado y puesto en funcionamiento en condiciones climáticas adversas, soportando fuertes

lluvias. Esto no desmejoró el sistema, al contrario, se obtuvieron resultados favorables determinando que su diseño y construcción le permiten funcionar en situaciones climáticas complicadas.

En comparación a sistemas de semaforización desarrollados en trabajos previos como el sistema de (Mahesh Kumar & Kumaraswamy, 2020) que utiliza sensores y actuadores con un microprocesador Raspberry Pi, el prototipo presentado en este proyecto destaca por su precisión. Al utilizar un modelo preentrenado basado en una red neuronal para capturar imágenes en tiempo real, evitamos posibles errores asociados con sensores convencionales, que pueden confundir la detección de objetos fácilmente.

Así mismo se presenta una tesis de pregrado en la Universidad Peruana de Ciencias aplicadas elaborada por (Christian Jauregui & María Torres, 2021), en la que se propone el diseño de un ciclo semafórico que es capaz de captar la densidad de peatones, y darles preferencia dependiendo de la situación del entorno. Los resultados obtenidos en este trabajo muestran que se obtuvo un 6.38% de mejora en el tiempo de espera del cruce peatonal en la carretera utilizada para la implementación del sistema.

Comparado con el sistema propuesto en este proyecto se determina que el prototipo implementado en el campus Miguel de Cervantes presenta una mejoría del 50.4% en la eficiencia de los tiempos de espera en la intersección semaforizada. Si bien es cierto que el flujo del trafico dentro del campus es mucho menor que una carretera con mucho flujo peatonal. Se estima que, en el caso de realizar la implementación del sistema en una calle o avenida con un flujo mucho mayor de peatones y vehículos, la mejora de la eficiencia de los tiempos de espera del cruce peatonal no disminuya.

Conclusiones y Recomendaciones

Conclusiones

La primera prueba de campo muestra que el dimensionamiento del embebido de la aplicación es adecuado, mediante la correcta selección de hardware y software, determinando que el embebido del sistema fue crucial para garantizar un rendimiento ideal de un prototipo de semaforización inteligente. La implementación exitosa del sistema y la integración con el IoT dieron paso al monitoreo en tiempo real del tráfico vehicular y peatonal en el campus

universitario. La estabilidad y la fiabilidad en la transmisión de los datos en tiempo real permitieron a quienes monitorean el tráfico visualizar información del cruce semaforizado.

Además, se destaca que el uso de la red neuronal pre-entrenada de YOLO, junto con la programación en micro-Python correspondiente para Raspberry Pi y la integración con la plataforma de desarrollo Arduino, combinaron eficazmente para el desarrollo de un prototipo de semáforo inteligente que mejoró significativamente la eficiencia en los tiempos de espera del flujo peatonal. La selección del microprocesador Raspberry Pi permitió la conectividad a internet, lo que posibilitó que el prototipo ser un dispositivo del IoT, lo que facilitará el monitoreo en tiempo real de todas las variables del sistema mecatrónico. Estos hallazgos confirman la funcionalidad del prototipo, ya que optimiza los tiempos de espera vehicular y peatonal en el semáforo, priorizando la seguridad peatonal en todo momento.

Finalmente se concluye, tras haber realizado la segunda prueba de campo en el escenario que involucra personas y vehículos en conjunto, que el prototipo de semaforización inteligente mejoró la eficiencia del flujo peatonal en un 50.4% y que se optimizan la formación de colas de espera de vehículos, ya que el sistema de semaforización evita la indicación de detenerse a los vehículos sin la existencia de personas esperando a cruzar por el cruce peatonal principal del campus Miguel de Cervantes de la Universidad Internacional SEK.

Recomendaciones

Se recomienda utilizar una cámara de visión artificial, para identificar de manera más precisa a las personas que vayan a ser identificadas por un algoritmo, ya que, si se utiliza una cámara de baja resolución, el modelo pre entrenado no hará el trabajo de identificar personas correctamente debido a que no procesara las imágenes de manera adecuada.

También se propone mejorar el procesador y verificar que cuente con las especificaciones necesarias para el procesamiento de imagen, ya que este proceso utiliza gran parte de procesamiento del CPU y GPU del ordenador, siendo un trabajo demandante para un microprocesador de bajo costo.

La red neuronal utilizada en el desarrollo de este proyecto es muy buena, sin embargo, no es la más precisa, se recomienda hacer uso de la Red neuronal YoloV8, de tal manera que mejore la robustez del software del sistema.

Finalmente, se recomienda hacer uso de conexión directa de ethernet al procesador, para tener una conexión estable al internet y no tener problemas con el Wi-Fi, ya que necesitaremos de esta para contar con un monitoreo del sistema en tiempo real adecuado.

Referencias bibliográficas

Abuhasel, K. A., & Khan, M. A. (2020). A Secure Industrial Internet of Things (IIoT) Framework for Resource Management in Smart Manufacturing. *IEEE Access*, *8*, 117354–117364. https://doi.org/10.1109/ACCESS.2020.3004711

Aleksey Kolodochkin, Irina Kulibaba, & Aleksandr Ogorodnikov. (2023, July 25). Models of traffic and pedestrian flows for organization of smart traffic light traffic. *E3S Web of Conferences*, 1–7.

Arduino. (2018, February 5). *What is Arduino?*

Arduino Store. (2021). *Arduino Uno Rev3*.

Arduino.cc. (2024). *Arduino Uno R3 Product reference manual* (pp. 1–14). https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf

Arevalo Vicente, Gonzalez J, & Gregorio Ambrosio. (2004). *LA LIBRERÍA DE VISIÓN ARTIFICIAL OPENCV APLICACIÓN A LA DOCENCIA E INVESTIGACIÓN*.

Boris Cutos, & Miguel Recalde. (2022). *MONITORIZACIÓN Y CONTROL DE SEGURIDAD EN HOGARES VULNERABLES EN EL SECTOR DE CHILLOGALLO DESDE DISPOSITIVOS MÓVILES MULTIPLATAFORMA, HACIENDO USO DE NODE-RED Y RASPBERRY PI4*. UNIVERSIDAD POLITÉCNICA SALESIANA SEDE QUITO.

Chauca, B. (2020). *Seguimiento y búsqueda de objetivos en entornos complejos usando micro vehículos aéreos con cámaras monoculares para aplicaciones militares*. Universidad de las fuerzas armadas.

Christian Jauregui, & María Torres. (2021). *Propuesta de semaforización actuada con detección de presencia peatonal, en la intersección de la Av. Huandoy con la carretera Panamericana Norte, para reducir el tiempo de cruce peatonal y la longitud de cola vehicular*. Universidad Peruana de Ciencias aplicadas.

Components 101. (2021, January 18). *5V 8-Channel Relay Module* .

David Perez. (2009). *Sistemas Embebidos y Sistemas Operativos Embebidos* .

Deshpande, S., & Hsieh, S.-J. (2023). Cyber-Physical System for Smart Traffic Light Control. *Sensors*, *23*(11), 5028. https://doi.org/10.3390/s23115028

Escobar Duque, & Jonathan Steve. (2020). *Sistema de monitoreo IoT de calidad de aire usando una red de sensores fijos y lorawan en el campus Sangolquí de la Universidad de las Fuerzas Armadas – ESPE*. Universidad de las Fuerzas Armadas - ESPE.

González, M., & Sepulveda, E. (2010). *APLICACIÓN DE TEORIA DE COLAS EN LOS SEMAFOROS PARA MEJORAR LA MOVILIDAD EN LA CARRERA 7 ENTRE CALLES 15 Y 20 DE LA CIUDAD DE PEREIRA*. Universidad tecnológica de Pereira.

Gradinescu, V., Gorgorin, C., Diaconescu, R., Cristea, V., & Iftode, L. (2007). Adaptive Traffic Lights Using Car-to-Car Communication. *2007 IEEE 65th Vehicular Technology Conference - VTC2007-Spring*, 21–25. https://doi.org/10.1109/VETECS.2007.17

IBM. (2021). *¿Qué son las redes neuronales convolucionales?*

Intel. (2020). The Difference Between Artificial Intelligence, Machine Learning and Deep Learning. *Artificial Intelligence (AI), 1*, 1–1.

Istec. (2022, January 5). *Internet of things, IoT*.

Jonathan Machado. (2023a, February 16). *Metro de Quito operará a su máxima capacidad en junio de 2023 Para hacer uso de este contenido cite la fuente y haga un enlace a la nota original en Primicias.ec: https://www.primicias.ec/noticias/sociedad/metro-quito-operacion-junio-viajes/*. 1–1. https://www.primicias.ec/noticias/sociedad/metro-quito-operacion-junio-viajes/

Jonathan Machado. (2023b, December 28). *Plan de Movilidad de Quito sugiere quitarle espacio a los carros privados Para hacer uso de este contenido cite la fuente y haga un enlace a la nota original en Primicias.ec: https://www.primicias.ec/noticias/quito/plan-movilidad-transporte-metro-carros/*.

Julio Schwendener. (2023). *Estudio comparativo entre los microcontroladores PIC16F887 y ATmega328P - pila, entradas y salidas, temporizador, interrupciones y memoria EEPROM*. Universidad del Valle de Guatemala.

Lakshmi, K., & Gayathri, S. (2017). Implementation of IoT with Image processing in plant growth monitoring system. *Journal of Scientific and Innovative Research*, *6*(2), 80–83. https://doi.org/10.31254/jsir.2017.6208

Mahesh Kumar, G., & Kumaraswamy, E. (2020). Smart Traffic Junction Using Raspberry Pi. *IOP Conference Series: Materials Science and Engineering*, *981*(3), 032048. https://doi.org/10.1088/1757-899X/981/3/032048

nuroum. (2024). *V11 2K AI-Powered Business Webcam*.

Organización Internacional de Normalización. (2018). *ISO/IEC 30141:2018 Internet of Things (IoT) Reference Architecture* (Vol. 1). ISO.

Orlando Silva. (2022, March 30). *Unos 30 minutos toma salir de sitios congestionados en Quito*. https://www.elcomercio.com/actualidad/quito/minutos-demora-congestion-avenidas-quito-transito.html

Patricia Armijo. (2023, February 23). El tráfico en Quito se intensifica en una docena de puntos. *El Comercio*, 1–1. https://www.elcomercio.com/actualidad/quito/trafico-quito-intensifica-doce-puntos.html

Pratama, B., Christanto, J., Hadyantama, M. T., & Muis, A. (2018). Adaptive Traffic Lights through Traffic Density Calculation on Road Pattern. *2018 International Conference on Applied Science and Technology (ICAST)*, 82–86. https://doi.org/10.1109/iCAST1.2018.8751540

Pricepulse. (2024). *Arducam Day-Night Vision for Raspberry Pi Camera, Wide Angle Automatic IR-Cut Switching All-Day Image, IR LED for Low Light and Night Vision*. https://docs.ultralytics.com/help/FAQ/

Ramos, C. (2021). Editorial: Diseños de investigación experimental. *CienciAmérica*, *10*(1), 1–7. https://doi.org/10.33210/ca.v10i1.356

Raspberry Pi. (2024a). *Raspberry Pi 4*.

Raspberry Pi. (2024b). *Remote access*. https://www.raspberrypi.com/documentation/computers/remote-access.html

Raul Izquierdo, & David Nelson. (2023, November 3). *This is the Shibuya crossing in Japan: more than 2 million people pass through every day*.

Ray, P. P. (2018). A survey on Internet of Things architectures. *Journal of King Saud University - Computer and Information Sciences*, *30*(3), 291–319. https://doi.org/10.1016/j.jksuci.2016.10.003

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2015). *You Only Look Once: Unified, Real-Time Object Detection*.

Ultralytics. (2024a). *Comprehensive Guide to Ultralytics YOLOv5*. https://docs.ultralytics.com/yolov5/

Ultralytics. (2024b). *Ultralytics YOLO Frequently Asked Questions*.

Wang, Z., Wu, Y., Yang, L., Thirunavukarasu, A., Evison, C., & Zhao, Y. (2021). Fast Personal Protective Equipment Detection for Real Construction Sites Using Deep Learning Approaches. *Sensors*, *21*(10), 3478. https://doi.org/10.3390/s21103478

Zhi-Hua Zhou. (2020). *Machine Learning* (Shawou Liu, Ed.; 1st ed., Vol. 1). Springer. https://doi.org/https://doi.org/10.1007/978-981-15-1967-3

Anexos

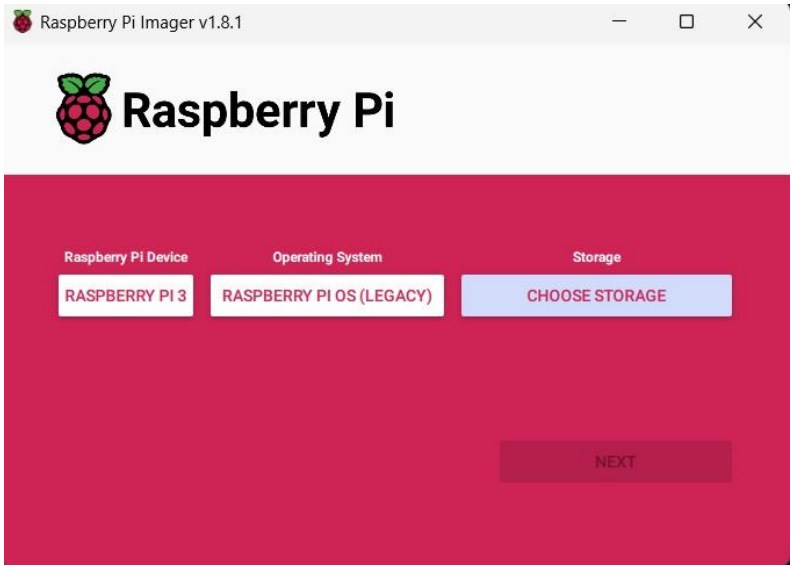
Anexo A.

Cuadro de video utilizado para recolectar información del tráfico vehicular y peatonal.

Extraído de: Video

Anexo B.

Raspberry Pi Imager para descargar el sistema operativo de Raspberry en la microSD



Anexo C.

Programa básico de funcionamiento del modelo YOLOv5s con ArduCam.

```
import torch
from tracker import Tracker
import numpy as np
import time
import picamera
import picamera.array
import cv2

model = torch.hub.load('ultralytics/yolov5', 'yolov5s', pretrained=True)
tracker = Tracker()

start_time = time.time()
reset_interval = 80  # segundos

with picamera.PiCamera() as camera:
    # Configurar la resolución según sea necesario
    camera.resolution = (640, 480)

    with picamera.array.PiRGBArray(camera) as stream:
        time.sleep(2)  # permitir que la cámara se estabilice
        for frame in camera.capture_continuous(stream, format='bgr', use_video_port=True):
            img = frame.array

            results = model(img)
            box_list = []
            for index, row in results.pandas().xyxy[0].iterrows():
                x1 = int(row['xmin'])
                y1 = int(row['ymin'])
                x2 = int(row['xmax'])
```

Anexo D.

Programa para rastreo de objetos

```python
import math
class Tracker:
    def __init__(self):
        self.center_points = {}
        self.id_count = 0
    def update(self, objects_rect)
        objects_bbs_ids = []
        for rect in objects_rect:
            x, y, w, h = rect
            cx = (x + x + w) // 2
            cy = (y + y + h) // 2
            same_object_detected = False
            for id, pt in self.center_points.items():
                dist = math.hypot(cx - pt[0], cy - pt[1])
                if dist < 35:
                    self.center_points[id] = (cx, cy)
                    objects_bbs_ids.append([x, y, w, h, id])
                    same_object_detected = True
                    break
            if same_object_detected is False:
                self.center_points[self.id_count] = (cx, cy)
                objects_bbs_ids.append([x, y, w, h, self.id_count])
                self.id_count += 1
        new_center_points = {}
        for obj_bb_id in objects_bbs_ids:
            _, _, _, _, object_id = obj_bb_id
            center = self.center_points[object_id]
            new_center_points[object_id] = center
        self.center_points = new_center_points.copy()
        return objects_bbs_ids
```

```python
                x2 = int(row['xmax'])
                y2 = int(row['ymax'])
                label = str(row['name'])
                if 'person' in label:
                    box_list.append([x1, y1, x2, y2])

            boxes_ids = tracker.update(box_list)
            for box_id in boxes_ids:
                x, y, w, h, obj_id = box_id
                cv2.rectangle(img, (x, y), (w, h), (0, 255, 0), 2)
                cv2.putText(img, str(obj_id), (x, y), cv2.FONT_HERSHEY_PLAIN, 3, (0, 0, 255), 2)

            cv2.imshow('FRAME', img)
            stream.truncate(0)  # limpiar el búfer de la cámara

            if time.time() - start_time > reset_interval:
                tracker.clear()
                start_time = time.time()

            if cv2.waitKey(1) & 0xFF == 27:
                break

cv2.destroyAllWindows()
```

Anexo E.

Algoritmo de identificación y conteo de personas con YOLO y OpenCV

```python
import cv2
import torch
from tracker import *
import numpy as np
import time
import RPi.GPIO as GPIO
model = torch.hub.load('ultralytics/yolov5', 'yolov5s', pretrained=True)
cap = cv2.VideoCapture('cctv.mp4')
def POINTS(event, x, y, flags, param):
    if event == cv2.EVENT_MOUSEMOVE:
        colorsBGR = [x, y]
        print(colorsBGR)
cv2.namedWindow('FRAME')
cv2.setMouseCallback('FRAME', POINTS)
tracker = Tracker()
area_1 = [(320, 315), (380, 373), (535, 339), (500, 296)]
area_2 = [(203, 230), (404, 186), (361, 153), (181, 167)]
area1 = set()
area2 = set()
pin_gpio = 17
GPIO.setmode(GPIO.BCM)
GPIO.setup(pin_gpio, GPIO.OUT)
start_time = time.time()
reset_interval = 8  # segundos
while True:
    ret, frame = cap.read()
    frame = cv2.resize(frame, (854, 480))
    cv2.polylines(frame, [np.array(area_1, np.int32)], True, (0, 255, 0), 3)
    cv2.polylines(frame, [np.array(area_2, np.int32)], True, (255, 0, 0), 3)
    results = model(frame)
    list = []
    for index, row in results.pandas().xyxy[0].iterrows():
        x1 = int(row['xmin'])
        y1 = int(row['ymin'])
        x2 = int(row['xmax'])
        y2 = int(row['ymax'])
        b = str(row['name'])
        if 'person' in b:
            list.append([x1, y1, x2, y2])
    boxes_ids = tracker.update(list)
    for box_id in boxes_ids:
        x, y, w, h, id = box_id
        cv2.rectangle(frame, (x, y), (w, h), (255, 0, 255), 2)
        cv2.putText(frame, str(id), (x, y), cv2.FONT_HERSHEY_PLAIN, 3, (0, 0, 255), 2)

        result_area1 = cv2.pointPolygonTest(np.array(area_1, np.int32), (int(w), int(h)), False)
        result_area2 = cv2.pointPolygonTest(np.array(area_2, np.int32), (int(w), int(h)), False)

        if result_area1 > 0:
            area1.add(id)
        elif id in area1:
            area1.remove(id)

        if result_area2 > 0:
            area2.add(id)
        elif id in area2:
            area2.remove(id)

    p_area1 = len(area1)
    p_area2 = len(area2)
    personas_esperando = p_area1 + p_area2
    if personas_esperando>=3:
        GPIO.output(pin_gpio, GPIO.HIGH)
    cv2.putText(frame, f"Personas esperando: {personas_esperando}", (20, 34), cv2.FONT_HERSHEY_PLAIN, 3, (0, 0, 255), 2)
    cv2.imshow('FRAME', frame)
    if time.time() - start_time > reset_interval:
```

```
        area1.clear()
        area2.clear()
        start_time = time.time()
    if cv2.waitKey(1) & 0xFF == 27:
        break
cap.release()
cv2.destroyAllWindows()
```

Anexo F.

Algoritmo de semaforización que da preferencia al peatón mediante interrupción con lectura de entrada digital.

semaforo §

```
const int verdepeatonal = 4;  // Primer LED
const int rojovehicular = 5;  // Segundo LED
const int rojopeatonal = 6;  // Tercer LED
const int verdevehicular = 7;  // Cuarto LED
const int pin = 2;  // Pin de interrupción

volatile int estado = 1;  // Estado inicial
volatile unsigned long tiempoInicioEstado = 0;  // Tiempo de inicio del estado actual
const unsigned long duracionEstadoUno = 10000;  // Duración en milisegundos
const unsigned long duracionEstadoDos = 6000000; // Duración en milisegundos

void iniciarSemaforo() {
  // Obtener el tiempo actual
  unsigned long tiempoActual = millis();

  // Estado uno: Encender los primeros dos LEDs durante 30 segundos
  if (estado == 1) {
    digitalWrite(verdepeatonal, HIGH);
    digitalWrite(rojovehicular, HIGH);
    Serial.println("¡estado uno!");

    // Verificar si ha pasado suficiente tiempo en el estado uno
    if (tiempoActual - tiempoInicioEstado > duracionEstadoUno) {
      // Cambiar al estado dos
      estado = 2;
      tiempoInicioEstado = tiempoActual;  // Reiniciar el temporizador
      digitalWrite(verdepeatonal, LOW);  // Apagar LEDs al cambiar de estado
```

```
      digitalWrite(rojovehicular, LOW);
    }
  }

  // Estado dos: Encender los siguientes dos LEDs durante 10 minutos
  else if (estado == 2) {
    digitalWrite(verdepeatonal, LOW);
    digitalWrite(rojovehicular, LOW);
    digitalWrite(verdevehicular, HIGH);
    digitalWrite(rojopeatonal, HIGH);
    Serial.println("¡estado dos!");

    // Verificar si ha pasado suficiente tiempo en el estado dos
    if (tiempoActual - tiempoInicioEstado > duracionEstadoDos) {
      // Cambiar al estado uno
      estado = 1;
      tiempoInicioEstado = tiempoActual;  // Reiniciar el temporizador
      digitalWrite(verdevehicular, LOW);  // Apagar LEDs al cambiar de estado
      digitalWrite(rojopeatonal, LOW);
    }
  } else {
    // En caso de un estado no esperado, reiniciar al estado uno
    estado = 1;
    tiempoInicioEstado = tiempoActual;
  }
}

void setup() {
  Serial.begin(9600);
  pinMode(verdepeatonal, OUTPUT);
  pinMode(rojovehicular, OUTPUT);
  pinMode(rojopeatonal, OUTPUT);
```

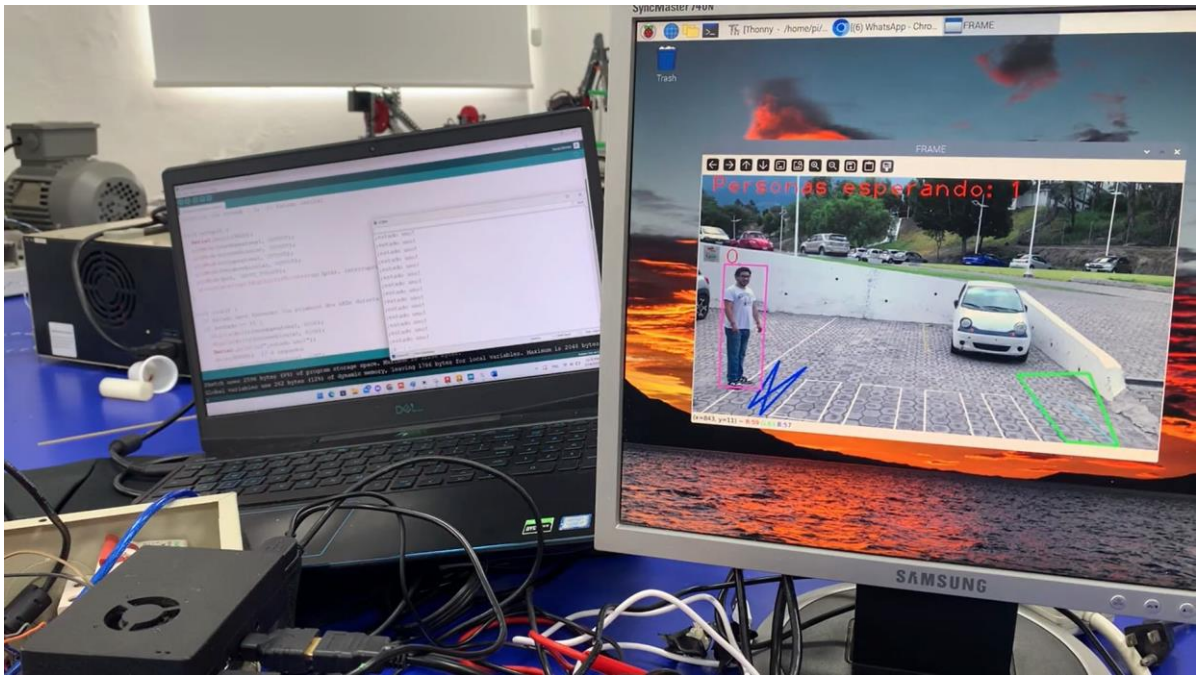
```
    digitalWrite(rojovehicular, LOW);
    digitalWrite(verdevehicular, HIGH);
    digitalWrite(rojopeatonal, HIGH);
    Serial.println("¡estado dos!");

    // Verificar si ha pasado suficiente tiempo en el estado dos
    if (tiempoActual - tiempoInicioEstado > duracionEstadoDos) {
      // Cambiar al estado uno
      estado = 1;
      tiempoInicioEstado = tiempoActual;  // Reiniciar el temporizador
      digitalWrite(verdevehicular, LOW);  // Apagar LEDs al cambiar de estado
      digitalWrite(rojopeatonal, LOW);
    }
  } else {
    // En caso de un estado no esperado, reiniciar al estado uno
    estado = 1;
    tiempoInicioEstado = tiempoActual;
  }
}
void setup() {
  Serial.begin(9600);
  pinMode(verdepeatonal, OUTPUT);
  pinMode(rojovehicular, OUTPUT);
  pinMode(rojopeatonal, OUTPUT);
  pinMode(verdevehicular, OUTPUT);
  pinMode(pin, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(pin), interrupcionPin, CHANGE);

}
void loop() {
  iniciarSemaforo();
}
void interrupcionPin() {
  // Verificar si estamos en el estado dos antes de hacer algo en la interrupción
  if (digitalRead(pin) == LOW && estado == 2) {
    Serial.println("¡Interrupción detectada en estado dos!");
    estado = 1; // Vuelve al estado uno
    tiempoInicioEstado = millis();  // Reiniciar el temporizador
    Serial.println("Volviendo al estado uno.");
  }
}
```

Anexo G.

Prueba de laboratorio para verificación del funcionamiento de actuadores mediante el conteo de personas con visión artificial.



Nota: La imagen es una captura de un video de la prueba de laboratorio que muestra el funcionamiento del embebido del sistema. Extraído de: Video

Anexo H.

Fotograma de video tomado para análisis de resultados en implementación de prototipo en entorno real

Nota: La imagen es una captura de un video de la segunda prueba de campo en la cual el prototipo funciona en una situación real. Extraído de: Prueba de campo 2