ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA ELECTRÓNICA Y AUTOMATIZACIÓN

"IMPLEMENTACIÓN DE UN SISTEMA FIJO MEDIANTE EL USO DE LA VISIÓN ARTIFICIAL PARA LA DETECCIÓN DE LUGARES DISPONIBLES EN EL ESTACIONAMIENTO VEHICULAR DEL PARQUE CENTRAL DE LA CIUDAD DE BAÑOS"

Trabajo de titulación

Tipo: Proyecto Técnico

Presentado para optar al grado académico de:

INGENIERO EN ELECTRÓNICA CONTROL Y REDES INDUSTRIALES

AUTOR: JHONATAN DANIEL MUÑOZ ROMERO

DIRECTOR: ING. PABLO EDUARDO LOZADA YÁNEZ

Riobamba – Ecuador

2021

Yo, Jhonatan Daniel Muñoz Romero, declaro que el presente trabajo de titulación es de mi autoría y los resultados del mismo son auténticos. Los textos en el documento que provienen de otras fuentes están debidamente citados y referenciados.

Como autor asumo la responsabilidad legal y académica de los contenidos de este trabajo de titulación; El patrimonio intelectual pertenece a la Escuela Superior Politécnica de Chimborazo.

Riobamba, 18 de noviembre de 2021

Jhonatan Daniel Muñoz Romero

185017618-9

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA ELECTRÓNICA Y AUTOMATIZACIÓN

El Tribunal del trabajo de titulación certifica que: El trabajo de investigación: tipo proyecto técnico, "IMPLEMENTACIÓN DE UN SISTEMA FIJO MEDIANTE EL USO DE LA VISIÓN ARTIFICIAL PARA LA DETECCIÓN DE LUGARES DISPONIBLES EN EL ESTACIONAMIENTO VEHICULAR DEL PARQUE CENTRAL DE LA CIUDAD DE BAÑOS", realizado por el señor: Jhonatan Daniel Muñoz Romero, ha sido minuciosamente revisado por los Miembros del Tribunal del trabajo de titulación, el mismo que cumple con los requisitos científicos, técnicos, legales, en tal virtud el Tribunal Autoriza su presentación.

	FIRMA	FECHA
Ing. Andrés Fernando Morocho Caiza **PRESIDENTE DEL TRIBUNAL**	ANDRES FERNANDO MOROCHO CAIZA Firmado digitalmente por ANDRES FERNANDO MOROCHO CAIZA DN: cn=ANDRES FERNANDO MOROCHO CAIZA, c=EC, o=SECURITY DATA S.A. 1, ou=ENTIDAD DE CERTIFICACION DE INFORMACION Motivo:Apruebo este documento Ubicación: n Fecha:2021-11-23 12:27:05:00	2021/11/18
Ing. Pablo Eduardo Lozada Yánez **DIRECTOR DEL TRABAJO DE TITULACIÓN**	Firmado digitalmente por PABLO EDUARDO LOZADA YANEZ	2021/11/18
Ing. Fausto Ramiro Cabrera Aguayo **MIEMBRO DEL TRIBUNAL**	FAUSTO RAMIRO CABRERA AGUAYO Digitally signed by FAUSTO RAMIRO CABRERA AGUAYO Date: 2021.11.23 21:05:39 -05'00'	2021/11/18

DEDICATORIA

Dedicado a toda mi familia y seres queridos especialmente a mi mamá Judy y papá Lucho quienes siempre me ha amado, apoyado y confiado ciegamente en mí, este logro, este título, es fruto de su fe; a mi pequeño hermano Johann a quien idolatro con todo el corazón; y a mi novia Soffy mi amor, mi compañera de vida S&D; gracias por brindarme su amor y apoyo incondicional día a día, los amo.

Danny

AGRADECIMIENTO

En agradecimiento a Dios, a mi mamá Judith Romero, a mi papá Luis Muñoz, a mi hermano, Johann Muñoz, a mi novia Sofía León, a toda mi familia, a mis amigos y demás seres queridos, quienes me han brindado en todo momento su amor y apoyo incondicional.

Danny

TABLA DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE GRÁFICOS

ÍNDICE DE ANEXOS

RESUMEN

El presente trabajo de titulación presentó como objetivo principal la implementación de un sistema fijo mediante el uso de la visión artificial para la detección de lugares disponibles en el estacionamiento vehicular del parque central de la ciudad de Baños. Se examinaron los diferentes sistemas de parqueaderos inteligentes para estudiar su funcionamiento. En base al análisis del hardware y software orientado a la interpretación y el procesamiento de imágenes de video, se utilizó una Cámara IP wifi 1080P cuya función es recolectar toda la información visual del mundo exterior y un computador el cual se encargó de realizar todo el procesamiento de la información mediante del lenguaje de programación Python. La interfaz gráfica resultante del programa de detección se mostró a manera de transmisión de video en la página web "*webstarts.com*" con dominio gratuito, concluyendo que el sistema de detección tiene la capacidad para trabajar de manera correcta sin verse afectado por las condiciones climáticas en el sector en específico demostrando una efectividad del 96%, cumpliendo con todos los requerimientos y objetivos planteados. Se recomienda extender el periodo de pruebas y recopilación de datos en ambientes con un mayor número de estacionamientos y afluencia vehicular para evaluar el rendimiento del sistema en pro de encontrar posibles mejoras al sistema.

Palabras clave: <VISIÓN ARTIFICIAL>, <ESTACIONAMIENTO VEHICULAR>, <ALGORITMO DE DETECCIÓN DE OBJETOS>, <LENGUAJE DE PROGRAMACIÓN PYTHON>, <INTERNET DE LA COSAS (IOT)>

ABSTRACT

The main aim of this research was the creation of a stationary system using artificial vision for the detection of available parking spaces for Vehicles in the parking lot located at the central park in Baños city. Different intelligent parking systems were considered in order to study their operation. Based on the analysis of hardware and software focused on the interpretation and processing of video images, we used a 1080P wifi IP camera whose function is to collect all the visual information from the outside world and a computer which was responsible for performing all the information processing using the Python programming language. The graphical interface resulting from the detection program was shown as a video transmission on the web page "webstarts.com" with free domain, concluding that the detection system has the capacity to work correctly without being affected by the weather conditions within the specific area, demonstrating an effectiveness of 96%, fulfilling all the requirements and objectives proposed. It is recommended to extend the period of testing and data collection in environments with a greater number of parking lots and vehicular affluence to evaluate the system performance in order to find possible improvements to the system.

Keywords: ARTIFICIAL VISION / VEHICLE PARKING / OBJECT DETECTION ALGORITHM / PYTHON PROGRAMMING LANGUAGE / INTERNET OF THINGS (IOT) .

Firmado electrónicamente por:
ENRIQUE JESUS
GUAMBO YEROVI

INTRODUCCIÓN

Debido al crecimiento demográfico el número de conductores y vehículos ha aumentado en los últimos años, por ello encontrar un lugar libre para estacionar el vehículo en un área congestionada conlleva un desperdicio de tiempo, resultando muy frustrante y estresante para los conductores, especialmente en horas pico. El problema que esto genera es que el conductor, en su búsqueda de un estacionamiento disponible, descuida el tránsito vehicular generando una situación de peligro, poniendo en riesgo a los demás conductores e inclusive a los transeúntes lo que conlleva a que este problema afecte a población en general y por ende al gobierno. En las ciudades del mundo moderno se han encontrado alternativas a favor de solucionar este grave inconveniente, las cuales se basan en la construcción de nuevas zonas de estacionamiento, implementación de cámaras de video, incorporación de sensores y cualquier otro dispositivo que guíe u oriente a los conductores. (Shoeibi y Shoeibi 2020, pág 177).

En el Ecuador el avance tecnológico no ha prestado demasiada importancia en solucionar los inconvenientes que los conductores experimentan al buscar un lugar libre para el estacionamiento vehicular, este problema es ignorado debido a la falta de implementación de tecnologías que permitan mejorar la eficacia de los. (Xu, Chen y Xie 2000, pág 9613; Erazo y Narváez 2019 , pág 1).

Al revolucionarse la electrónica también lo hicieron las ciencias de la computación haciendo factible la inteligencia artificial. Dado que el cerebro es capaz de realizar muchas funciones de forma inconsciente de manera perceptiva, actualmente es imposible implementarlo artificialmente en su totalidad, no obstante, la visión artificial reconstruye nuevos y sofisticados algoritmos que son capaces de obtener información de un bajo nivel visual realizando tareas repetitivas y alienantes para el ser humano. La visión artificial o visión por computadora emula la visión del ser humano recompilando la información visual del mundo exterior para su posterior procesamiento, tratamiento e interpretación (Platero 2014, págs 11-13; Generalitat de Catalunya 2012, pág 4). En base a lo mencionado anteriormente, el objetivo de esta investigación es hacer uso de la visión artificial para el reconocimiento de disponibilidad en espacios de estacionamiento vehicular.

A continuación, se hace una breve descripción de los capítulos que conforman el presente Trabajo de Titulación.

En el Capítulo I se realiza el diagnóstico del problema conformado por: antecedentes, delimitación, formulación del problema, sistematización del problema, justificación y objetivos, a favor de sustentar el objeto de estudio.

En el Capítulo II se detallan los fundamentos teóricos: parqueaderos inteligentes, internet de las cosas, visión artificial, software de visión artificial y hardware de visión artificial, donde se detallan los conceptos principales de la investigación para fomentar la comprensión de la misma.

El Capítulo III describe el marco metodológico: requerimientos del sistema, concepción general, diseño de la arquitectura, hardware, software, desarrollo y representación del sistema.

El Capítulo IV detalla los resultados obtenidos de las pruebas de: desarrollo del software, latencia de transmisión de video, tiempo de procesamiento, consumo de recursos y la validación del sistema de detección.

Para finalizar, el documento presenta las referencias bibliográficas y anexos que forman parte del Trabajo de Titulación.

CAPÍTULO I

1. DIAGNÓSTICO DEL PROBLEMA

En este capítulo se detalla la información preliminar del objeto de estudio, el lugar donde se llevará a cabo la implementación del sistema de detección, además se presenta la formulación del problema, se describe la respectiva justificación y objetivos del trabajo.

1.1. Antecedentes

La ausencia de un buen sistema de identificación de lugares disponibles para el estacionamiento vehicular incide en el estado de ánimo de los conductores, esto repercute en una experiencia negativa hacia los mismos que mal gastan su tiempo en busca de un lugar de estacionamiento libre; en muchas de las ciudades del mundo moderno esto se ha resuelto mediante construcción de nuevas zonas de estacionamiento, implementación de cámaras de video, incorporación de sensores y demás dispositivos que guíen u orienten a los conductores en su búsqueda, esto ha teniendo también un gran aporte por parte de la tecnología IoT en su integración con estos sistemas inteligente de parqueo (Shoeibi y Shoeibi 2020, pág 177).

En 2012 Tschentscher, M. y Neuhasen, M. en su estudio denominado *"Detección de plazas de aparcamiento por video"*, propusieron un sistema basado en video para detectar espacios de estacionamiento libres a bajo costo, utilizaron una cámara con lente gran angular en combinación con una computadora de escritorio; así evaluaron diferentes extractores de características y algoritmos de aprendizaje automático para recuperar información de estado precisa para cada uno de los estacionamientos observados. Al final encontraron un sistema que incorpora integración temporal con una precisión del 99,8% (Tschentscher y Neuhausen 2012, págs 159-160).

En 2018 Acharya, D., Yan, W. y Khoshelham, K. estudiaron la "*Detección de ocupación de estacionamiento basada en imágenes en tiempo real utilizando aprendizaje profundo*", su investigación se centró en determinar la ocupación de plazas de aparcamiento a partir de imágenes obtenidas por vigilancia de cámaras considerando las características rentables de los sistemas basados en cámaras, adoptando un modelo ImageNet-VGG-f que es una red neuronal convolucional (CNN) profunda previamente entrenada en el conjunto de datos ImageNet, con las características de la CNN se utiliza un clasificador de máquina de vectores de soporte (SVM)

binario. Las caracterizas extraídas de las imágenes de los conjuntos de datos PKLot se utilizaron para entrenar y probar 4 clasificadores SVM binarios utilizando etiquetas de verdad del terreno (tiempo nublado, lluvioso, soleado y todos los tiempos juntos), después evaluaron la precisión de los clasificadores entrenados mediante una validación cruzada de 5 veces para eliminar cualquier sesgo de los conjuntos de datos; para evaluar el rendimiento del aprendizaje por transferencia del método, el clasificador se entrenó con el conjunto de datos PKLot completo y probaron imágenes segmentadas del conjunto de datos de Barry Street. Se logró una alta precisión del 99,7% en el conjunto de datos de entrenamiento y una precisión de aprendizaje profundo de transferencia de 96,6% en un conjunto de datos de prueba independiente, indicando su idoneidad para aplicaciones masivas en todas las condiciones climáticas, además este marco investigativo proporciona una solución barata y confiables para los sistemas de información y orientación sobre estacionamientos (PGI) en ambientes al aire libre (Acharya, Yan y Khoshelham 2018, pág 36).

En 2011, la Escuela Politécnica Nacional desarrolló un "*Prototipo de un sistema de adquisición de imágenes de vehículos, detección y reconocimiento automático de los caracteres de la placa en tiempo real por medio de visión artificial, aplicado al control vehicular*", esta investigación se la desarrolló con la ayuda del software Matlab y el uso de la visión artificial para el procesamiento de las imágenes adquiridas de los vehículos usando la cámara web Genius eye 312 con lo cual se obtuvo un 50 % de efectividad en el reconocimiento de placas vehiculares (Cañadas y Haro 2011, pág xxvii).

Marcelo Márquez en 2015 realizó un proyecto denominado *"Desarrollo de un prototipo de parqueadero inteligentes empleando redes de sensores inalámbricos",* con el fin de reducir el tiempo de búsqueda y el consumo de combustible. El trabajo fue realizado en los parqueaderos de la Universidad de las Fuerzas Armadas – ESPE, colocando tres sensores en estas plazas de parqueo, así toda la información era enviada por medio de una red wifi a una aplicación en los dispositivos móviles de los usuarios, de esta forma se puede observar en tiempo real a través de un mapa la ubicación de los lugares de parqueadero libres con un retardo de aproximadamente de 3 segundos. La información que recibían los usuarios en tiempo real pudo reducir significativamente el tiempo que utilizaban en buscar y estacionar sus vehículos (Márquez Rosero 2015, pág xvi).

En 2016 Leandro Rosales desarrolló un estudio sobre el *"Diseño e implementación de un parqueo inteligente utilizando Arduino Yun basado en internet de las cosas"*, teniendo como objetivo optimizar los procesos de buscar un lugar libre para parquear en la Universidad Politécnica Salesiana sede Guayaquil utilizando sensores ultrasónicos para detectar los vehículos medianos, enviando una señal a la tarjeta receptora de datos y transmitiendo los datos de forma digital al

Arduino Yun; la plataforma Temboo administra y envía la información a través de internet, visualizando así en la cuenta de Twitter los lugares libres de parqueo (Rosales 2016, pág vi).

En 2018 el grupo de investigación en informática de la universidad tecnológica equinoccial (UTE), estudiaron una *"Propuesta arquitectural para un sistema de detección automática de parqueaderos libres"*, el sistema de detección fue en tiempo real usando la visión artificial y el aprendizaje automático; además aprovecharon la existencia de cámaras de vigilancia en la infraestructura para obtener las imágenes y minimizar costos de implantación. Los datos recopilados fueron distribuidos a varios equipos interconectados para procesarlos y proporcionar a los usuarios en sus teléfonos móviles en tiempo real la mejor alternativa de parqueo optimizando el tiempo de acceso a la institución (Ordoñez-Camacho, Gómez y Ávalos 2018, págs 351-352).

A través de un estudio realizado en la Universidad Técnica del Norte se realizó un "*prototipo de detección de aparcamientos libres mediante visión artificial en un parqueadero de la Universidad Técnica del Norte*", analizaron las ventajas y desventajas del uso y distribución de plazas de estacionamiento dentro de dicha universidad, para la implementación del sistema se hizo uso de una cámara ubicada de forma estratégica la cual enviaría imágenes para su procesamiento a través de visión artificial hacia una computadora de placa simple, de esta manera la información sería retrasmitida hacia los usuarios mediante una página web y en dispositivos móviles con sistema operativos Android para su fácil interpretación. Las pruebas se realizaron en el periodo diurno y los lugares de estacionamiento estuvieron al alcance de la visión de la cámara; además se realizó un análisis de la relación costo/beneficio (Erazo y Narváez 2019, pág 1).

1.2. Delimitación

1.2.1. Espacial

Este trabajo se centra en el parqueadero del parque central Palomino Flores de la ciudad de Baños de Agua Santa. En la figura 1-1 se representa el parque central que está ubicado en la calle Vicente Rocafuerte entre Thomas Halflants y Pedro Vicente Maldonado.

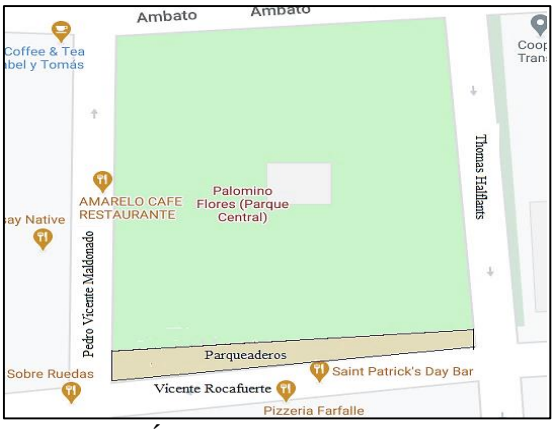


Figura 1-1. Área de parqueaderos
Fuente: (Google Maps, 2020)

1.2.2. Temporal

El presente trabajo de titulación se lo efectuara en el periodo académico comprendido entre los años 2020-2021.

1.2.3. Institucionalmente

Para desarrollar este trabajo se pondrá en práctica todos los conocimientos adquiridos durante el periodo de formación en la facultad de Informática y Electrónica para obtener el título de Ingeniero en Electrónica Control y Redes Industriales de la Escuela Superior Politécnica de Chimborazo, enfatizando en temas relacionados con la programación y visión artificial.

1.3. Formulación del problema

¿Se puede implementar un sistema fijo mediante el uso de la visión artificial para la detección de lugares disponibles en el estacionamiento vehicular del parque central de la ciudad de Baños?

1.4. Sistematización del problema

- ¿Cuál es el funcionamiento de los diferentes sistemas de parqueaderos inteligentes?
- ¿Qué hardware y software son necesarios para desarrollar el algoritmo que permita la interpretación y procesamiento de imágenes?
- ¿El algoritmo es capaz de reconocer lugares libres para el estacionamiento vehicular en la calle Vicente Rocafuerte del Parque Central Palomino Flores, Baños de Agua Santa?
- ¿El sistema desarrollado cumple con los requerimientos?

1.5. Justificación

1.5.1. Justificación Teórica

La implementación de nuevas tecnologías logra automatizar cada vez más las actividades que las personas desarrollan a diario, ya sea por innovación o creación de nuevos procesos automatizados a nuestro alrededor, esto abre un sinfín de posibilidades al mejoramiento de vida de las personas siempre en busca de optimizar tiempo y energía. Debido al crecimiento demográfico el número de conductores y vehículos ha aumentado en los últimos años; el transporte y plazas disponibles para el aparcamiento son factores que no solo afectan a la población sino también al gobierno.

En el Ecuador de acuerdo con el Instituto Nacional de Estadísticas y Censos (INEC), los vehículos matriculados en el periodo del 2008 son de 1.484.743 vehículos y en el 2018 se ha generado un aumento de 2.403.651 obteniendo una tasa de crecimiento anual del 10,7% (Tenesaca y Hidalgo 2019, pág 6). La provincia de Tungurahua en el año 2018 registro 107.224 vehículos matriculados representando un 4,5 % del total, además esta provincia tiene la mayor tasa de matriculación vehicular, es decir que se tiene 186 vehículos por cada mil habitantes (INEC 2019, págs 6-7).

A través del sentido de la vista, el ser humano tiene la capacidad de recolectar la información del mundo exterior e interpretar su entorno, el ojo es el órgano que se encarga de detectar la luz y transformarla en un impulso neuronal, este a su vez pasa hacia el cerebro para ser procesado y generar las sensaciones visuales (Álvarez Durán 2014, pág 5). Desde que surgieron las cámaras de video CCD y los microprocesadores, las ciencias de la computación ha evolucionado día a día haciendo posible la denomina, visión artificial, esta, por medio de sensores (cámaras) capta la información visual del entorno físico para extraer las características más relevantes, es decir, trata de emular la visión del ser humano a través de un proceso de captación de imágenes para su posterior tratamiento a través de técnicas de procesamiento avanzado (Platero 2014, págs 11-13; Generalitat de Catalunya 2012, pág 4).

El avance tecnológico no ha prestado demasiada importancia en solucionar los inconvenientes que los conductores experimentan al buscar un lugar libre para el estacionamiento lo cual origina situaciones adversas en las que el conductor prestar mayor atención a buscar un espacio libre para el estacionarse dejando de lado su total atención al tránsito vehicular de su alrededor.

La presente investigación tiene como finalidad encontrar una solución factible ante el problema que los conductores experimentan día a día buscando un espacio libre disponible para el parqueo, con lo cual se hará uso de la visión artificial para el reconocimiento de disponibilidad en espacios estacionamiento vehicular.

1.5.2. Justificación Aplicativa

Es un sistema inmóvil que mediante el uso de la visión artificial realizará la detección de lugares disponibles en el estacionamiento vehicular del parque central de la ciudad de Baños. En la figura 2-1 se puede visualizar la ubicación tentativa para la colocación del dispositivo de captura de imágenes.

Figura 2-1. Lugar tentativo para la colocación del dispositivo de captura de imágenes.
Fuente: (Google Maps, 2020)

Para recolectar la información del entorno el sistema llevará a cabo un monitoreo en tiempo real de la escena requerida con la ayuda de un dispositivo de captura de imágenes, a través de la ejecución de un algoritmo se hará un reconocimiento de los vehículos presentes para evaluar si están dentro o fuera de la región de interés, esta región estará limitada por el espacio de estacionamiento que podría estar siendo utilizada por determinado vehículo, posterior a esto con la información obtenida se podrá determinar si existe o no un espacio de estacionamiento disponible.

El sistema permanecerá fijo y funcionará cuando todos los dispositivos se encuentren dentro de la misma red, toda la información recogida mediante el monitoreo será analizada y procesada mediante un algoritmo para finalmente representar la información pertinente en una página web para que la misma pueda ser interpretada y usada a criterio de cada persona.

1.6. Objetivos

1.6.1. *Objetivo general*

Implementar un sistema fijo mediante el uso de la visión artificial para la detección de lugares disponibles en el estacionamiento vehicular del parque central de la ciudad de Baños.

1.6.2. Objetivos específicos

- Analizar el funcionamiento de los diferentes sistemas de parqueaderos inteligentes mediante revisión bibliográfica.
- Seleccionar el hardware y software necesarios para desarrollar el algoritmo que permita la interpretación y procesamiento de imágenes.
- Implementar un algoritmo capaz de reconocer lugares libres para el estacionamiento vehicular ubicado en la calle Vicente Rocafuerte del parque central Palomino Flores de Baños de Agua Santa.
- Evaluar si el sistema desarrollado cumple con los requerimientos.

CAPÍTULO II

2. FUNDAMENTOS TEÓRICOS

En este capítulo se describen los conceptos de los temas principales que conforman la presente investigación con el objetivo de fomentar comprensión de la misma para lo cual se hace énfasis en los parqueaderos inteligentes, el internet de las cosas y el hardware y software que conforman los sistemas de visión artificial.

2.1. Parqueaderos inteligentes

Un sistema de parqueo inteligente se define por la capacidad de proporcionar al conductor la información de las plazas disponibles de parqueo ahorrando tiempo para el conductor y gastando menos combustible, a la vez son eficientes para reducir las emisiones producidas por los automóviles dentro de la zona urbana, esto viene como resultado de que los conductores ya no deberán circular innecesariamente en busca de estacionamiento además, aporta a que las ciudades estén en la capacidad de administrar de mejor manera su suministro de estacionamiento llevando a cabo un proceso para la ubicación estratégica de los mismos (Basu 2014, pág 3; Márquez, Lara y Gordillo 2014, pág 176).

Los parqueaderos inteligentes son capaces de realizar tareas de forma automática, mediante el uso de sensores en donde se recepta la señal de las variables a medir y realizan las funciones que contralan al sistema, así cada uno presenta condiciones diferentes ya que dependen mayormente de las características físicas del lugar (Erazo 2019, pág 6).

2.1.1. Ventajas de parqueaderos inteligentes

Los sistemas de parqueo inteligente permitirán lo siguiente:

- Ayudar a que el tráfico de la ciudad fluya con mayor libertad aprovechando la tecnología de IoT.
- Predecir y detectar con precisión los espacios disponibles en tiempo real.
- Guiar a los residentes y visitantes al estacionamiento disponible.
- Optimizar el uso del espacio de estacionamiento.

- Crea un mejor entorno urbano al reducir la emisión de CO2 y otros contaminantes (Basu [sin fecha], pág 3).

2.1.2. Tipos de parqueaderos

Existe una gran variedad de parqueaderos los cuales son implementados dependiendo de acuerdo a su funcionalidad y a las necesidades de las personas.

2.1.2.1. Multiplaza

Los parqueaderos multinivel o también conocidos como robotizados o multiplaza son un ejemplo avanzado de parqueaderos inteligentes, en este sistema los vehículos son parqueados de manera automática por medio de movimientos de elevación de ascensores y son transportados de manera horizontal a través de plataformas hacia los diferentes sitios disponibles en los niveles del parqueadero. Este parqueadero permite guardar de 3 a 5 vehículos y en otros casos guardan de 10 hasta 90 vehículos. Todo esto es de acuerdo a lugar donde está implementado ya que, las ciudades más grandes son las más transitadas y por ende existe una mayor demanda de toda clase como: parqueaderos, hospitales, edificios, instituciones públicas, etc. (Erazo 2019; Parking System 2019).

En la figura 1-2, se presenta el funcionamiento de los parqueaderos multinivel que tienen una plataforma móvil que se eleva verticalmente hasta ubicar al vehículo de forma horizontal en el espacio que esté disponible. Todas las operaciones del sistema están automatizadas y computarizadas, se utiliza una tarjeta de proximidad que acciona el mecanismo para ingresar y retirar el vehículo de forma segura (Parking System 2019).

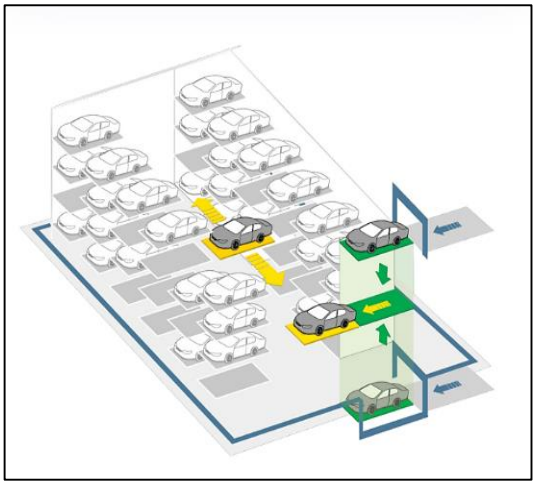


Figura 1-2. Parqueadero multinivel
Fuente: (Ecoparking 2016)

2.1.2.2. Rotatorio

Está diseñado para que el vehículo se ubique en una celda y esta gire hacia arriba hacia abajo y se mueva de derecha a izquierda para ubicar el vehículo. Abarca de 5 a 20 vehículos en un área donde inicialmente solo cabrían dos automóviles como se ve en la figura 2-2 (Ecoparking 2016).

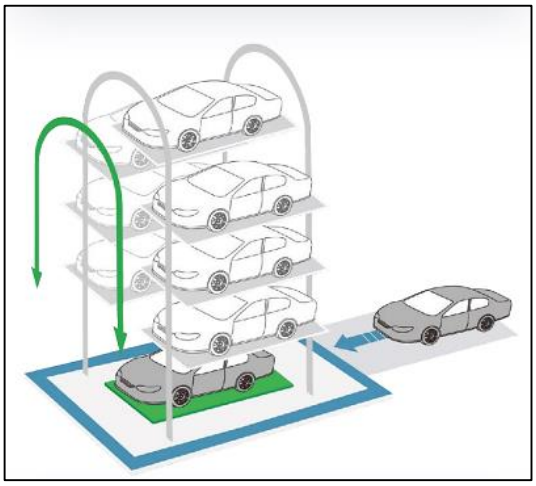


Figura 2-2. Parqueadero rotatorio
Fuente: (Ecoparking 2016)

2.1.2.3. Tipo torre

Está diseñado para que realice movimientos verticales tipo ascensor y colocar el vehículo de manera frontal o como haya sido asignado por el software; cada piso de la torre tiene una altura de 1.61 metros. En la figura 3-2, se presenta este tipo de parqueadero automático con una fácil operatividad (Ecoparking 2016; Parking System 2019).

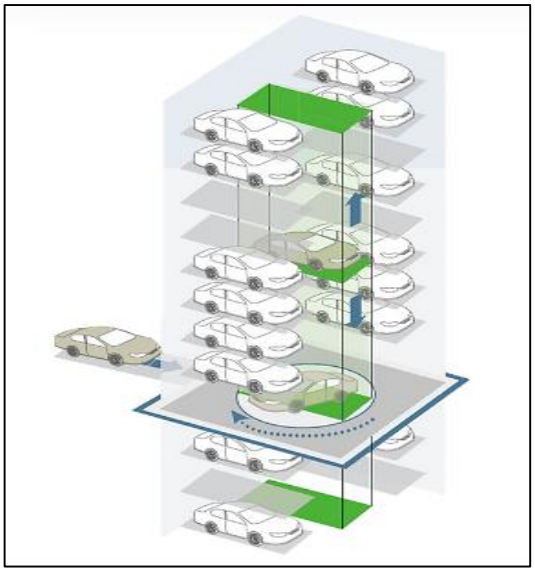


Figura 3-2. Parqueadero tipo Torre
Fuente: (Ecoparking 2016)

2.1.2.4. Basado en sensores

Los parqueaderos inteligentes basado en sensores son los que mayor popularidad han ganado en este terreno, generando un mayor interés por parte de los investigadores a partir de 2005. Para este tipo de parqueaderos se utilizan sensores inalámbricos cuyas redes de comunicación están conformadas por los mismos y tiene varias ventajas ante los otros tipos de sistemas: mayor flexibilidad al momento de ser implementados en casi cualquier lugar, tienen una rápida respuesta ante la detención y su aplicación e implementación tienen un costo razonable (Erazo 2019), este último factor es crucial para las empresas al momento de decidir un sistema a implementar en un parqueadero inteligente.

2.1.2.5. Basado en sistema de posicionamiento global (GPS)

Un Sistema de Posicionamiento Global (GPS) se lo utiliza para determinar la posición de un objeto en cualquier parte del planeta, para este caso el objeto a localizar será el vehículo. Este sistema orientado a parqueaderos inteligentes entrega la información necesaria de posición y disponibilidad de puestos libres en el parqueadero definido determinando la presencia o ausencia de un vehículo (Erazo 2019).

2.1.2.6. Basado en visión artificial

Este sistema proporciona una percepción de información mediante imágenes y la inmunidad a las interferencias y al ruido, mediante el procesamiento de imágenes y la visión artificial se puede detectar un espacio vacío en un lugar de estacionamiento, así mismo también es posible la detección de obstáculos en el estacionamiento. Una vez detectado la plaza del parqueadero libre se debe planificar una ruta factible adecuada para movilizar el vehículo. En la figura 4-2 se visualiza el parqueadero inteligente basado en un sistema de visión artificial en el que determina si hay un lugar de estacionamiento disponible (Xu, Chen y Xie 2000, pág 726).

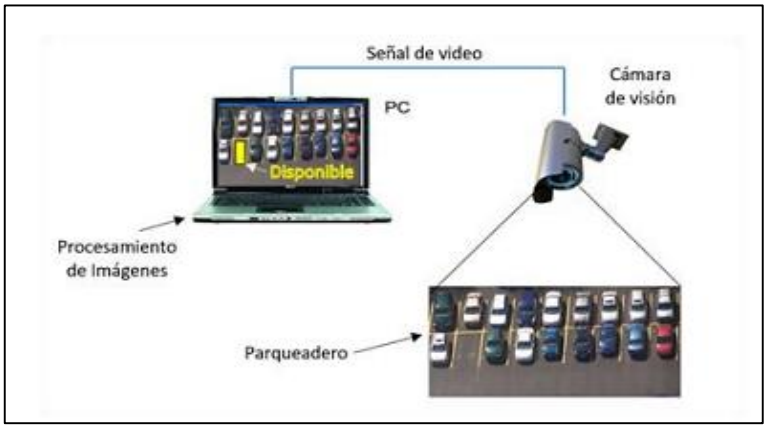


Figura 4-2. Parqueadero inteligente mediante visión artificial
Fuente: (Erazo 2019, pág 8)

2.2. Internet de las cosas (IoT)

El internet es una de las creaciones más importantes y poderosas de toda la historia de la humanidad siendo ahora el IoT la evolución del Internet. IoT se describe como una interconexión digital de las cosas con la red de internet, así mismo se encuentra conformado por muchas tecnologías, por ejemplo, los sensores que admiten conectar el mundo físico con el mundo digital mediante el procesamiento de información por computadores, además existen plataformas web en donde se pueden procesar y almacenar datos; IoT es uno de las avances tecnológicos que pretende hacer más cómoda la vida de todos los seres humanos (IAC 2018).

IoT aporta a tener una vida inteligente, los objetos adoptan un comportamiento inteligente ya que son capaces de identificarse y transmitir información sobre sí mismos habilitando la toma de decisiones. Al estar todo un sistema comunicado entre sí, todos los objetos tendrán acceso a la información proporcionada por los demás objetos. Este concepto de interconexión digital se encuentra en todas las partes del mundo, no solo en sistemas destinados a la domótica en casas inteligentes sino también en servicios de notificaciones, seguridad, comunicación, entretenimiento, computación, ahorro de energía, vehículos inteligentes y demás aplicaciones. De cierta forma se podría ver a una ciudad inteligente como una ciudad del futuro por medio del uso de IoT ya que, con la implementación de estos sistemas se puede mejorar progresivamente el desarrollo de una ciudad, no obstante, esto requiere de una planificación de lo más cuidadosa posible. Adicionalmente las ciudades pueden mejorarse en muchos aspectos como son: infraestructura, transporte público, disminuyendo la congestión en el tránsito vehicular, brindando mayor seguridad a los ciudadanos, mejorando los sistemas de salud, monitorizando el clima y demás aplicaciones (Erazo 2019, págs 16-17).

2.2.1. Aplicaciones del IoT

IoT es la realidad que se está viviendo hoy en día, ya que existe mucha demanda de equipos interconectados entre sí, a continuación, se describen algunas aplicaciones.

2.2.1.1. Monitoreo ambiental

Se hace uso de sensores que ayudan en la protección ambiental monitoreando las situaciones atmosféricas, los movimientos de la vida silvestre y su hábitat. Los sensores se encargan de detectar y medir cualquier tipo de cambio ambiental. En la figura 5-2 se presenta esta aplicación del IoT (Tirupathi y Krishna 2018, pág 86).

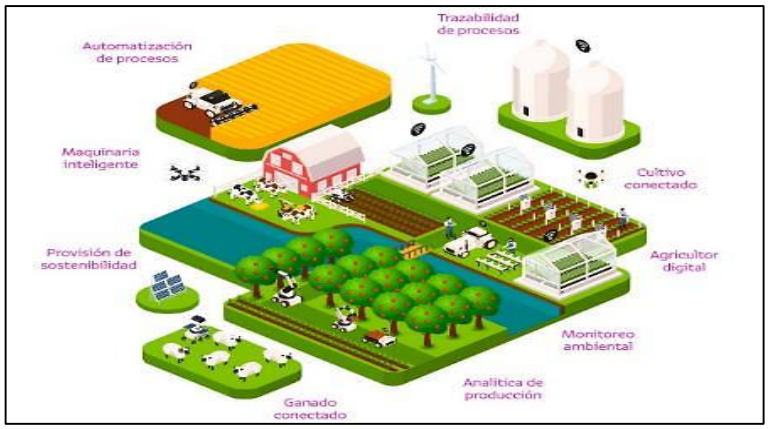


Figura 5-2. Monitoreo ambiental mediante instalación de sensores y equipos IoT
Fuente: (Axity 2015)

2.2.1.2. Transporte

El IoT es difundido a todo el sistema de trasporte, es decir, al conductor o usuario y al automóvil, esta interacción dinámica permite: la comunicación intra e inter vehicular, el control inteligente del tráfico, el estacionamiento inteligente, la gestión electrónica, el control de carros y la seguridad y asistencia en la vía. Los carros modernos se encuentran equipados con sensores que se conectan a internet a través de sistemas de control. En la figura 6-2 (a) se presentan algunos sensores que se utilizan en los autos, además la figura 6-2 (b) representa la detección de colisiones, advertencias de cambio de carril, control de señales de tránsito y programación inteligente del tráfico (Tirupathi y Krishna 2018, pág 86).

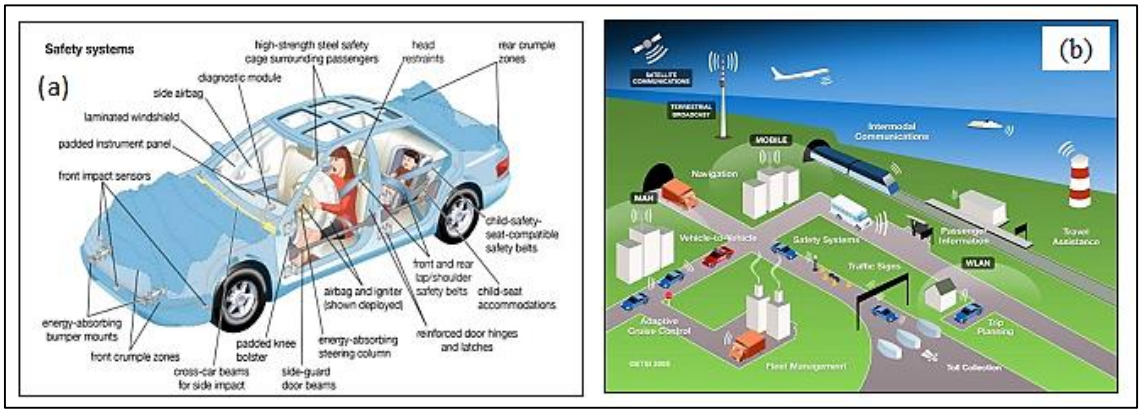


Figura 6-2. Sistema de seguridad: (a) Diseño del automóvil moderno con sensores; (b) Aplicaciones de IoT en el trasporte
Fuente: (Tirupathi y Krishna 2018, pág 86)

2.2.1.3. Atención médica y sanitaria

Los sistemas IoT se pueden utilizar para habilitar sistemas de notificaciones en caso de una emergencia o realizar un control de salud remoto (Tirupathi y Krishna 2018, pág 86).

2.2.1.4. Gestión de la infraestructura

Los dispositivos IoT se utilizan para monitorear cualquier evento o cambio que se requiera en las condiciones estructurales que podrían comprometer la seguridad y aumentar el riesgo de trabajo. Por ejemplo, se puede controlar una infraestructura crítica como puentes para facilitar el ingreso a los barcos (Tirupathi y Krishna 2018, pág 86).

2.2.1.5. Manufactura o fabricación

IoT permite una rápida fabricación de nuevos productos, optimizando en tiempo real la producción y el suministro de fabricación a través del uso de un conjunto de maquinarias, sensores y sistemas de control en red. Los sistemas de control necesitan la ayuda de IoT para automatizar los procesos, la seguridad de la planta y la interconexión de los mismos. En la figura 7-2 se presentan un sistema físico cibernético, que consiste en una estación de procesos productivos entre robots, máquinas e instrumentos; al estar habilitada con competencia de procesamiento de cómputo se convierte una maquina inteligente, además tiene la capacidad para comunicarse con otros sistemas por medio de una red (Tirupathi y Krishna 2017, pág 86; Sineti 2020).

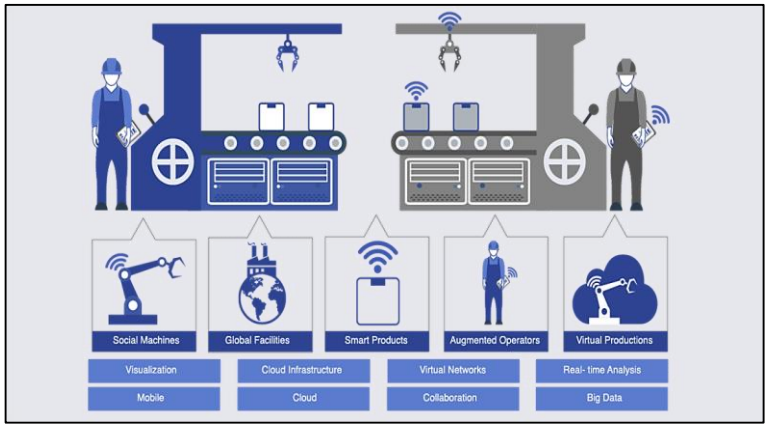


Figura 7-2. Sistema físico cibernético
Fuente: (Sineti 2020)

2.2.1.6. Domótica

Es una extensión residencial de la automatización de edificios que involucra el control y automatización de luminarias, ventilación, calefacción, aire acondicionado, seguridad, y electrodomésticos; todos estos dispositivos utilizan una red wifi para realizar el monitoreo remoto (Tirupathi y Krishna 2018, pág 86).

2.2.2. Ventajas del IoT

La función principal de IoT es hacer mucho más fácil la vida de todos. Utilizando esta tecnología y un equipo móvil se puede interactuar con otros equipos y realizar un seguimiento de las

actividades que se hace en la vida cotidiana, como por ejemplo, verificar la seguridad del hogar, bloquear un automóvil, prender o apagar las luces de la casa, etc., (Franco 2014, pág 17).

2.2.3. Desventajas del IoT

El inconveniente con la IoT es la vulnerabilidad de la privacidad del usuario (Franco 2014, pág 17).

2.2.4. Beneficios del IoT

- Fabricar nuevos productos en la planta de fabricación de forma rápida y con la debida precisión.
- Usar para monitorear a los pacientes en los hospitales.
- Utilizar como dispositivo de seguridad en los hogares.
- Ayuda en los envíos realizando un seguimiento individual.
- Todos los sistemas de IoT utilizan una mínima cantidad de energía realizando así una mejor calidad de vida de forma rápida y precisa.
- Usando IoT en los medios de transporte se minimizan las congestiones y colisiones.
- Transferencia de los datos entre las personas (Tirupathi y Krishna 2018, pág 86).

2.3. Visión artificial

La visión por computador o más comúnmente conocida como visión artificial se define como un campo de la "Inteligencia artificial" que interpreta la realidad del mundo físico a través de la captación de imágenes y el reconocimiento de patrones por medio de sensores (cámaras) con el objetivo de procesar y analizar la información recopilada en un computador, cuya finalidad es reproducir el sentido de la vista (CONTAVAL 2016). En la figura 8-2 se presenta una comparación del sistema de visión artificial y la vista del ser humano.

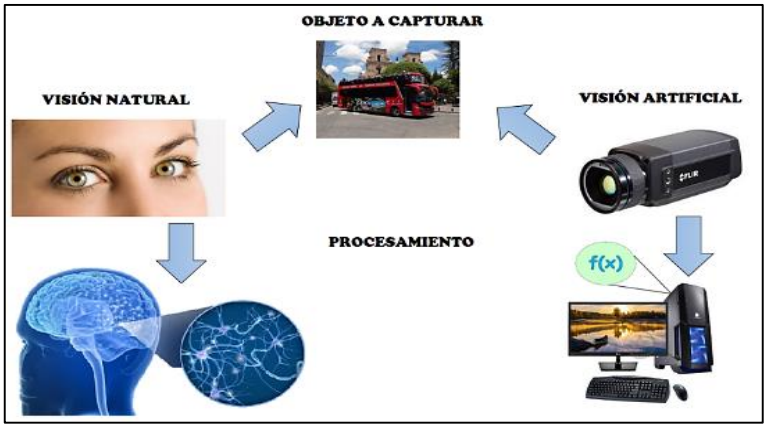


Figura 8-2. Visión artificial vs visión humana
Realizado por: Muñoz Jhonatan, 2021

El proceso de adquisición de imágenes cambia según sea la aplicación y el tipo de sistemas donde se requiere adquirir imágenes de objetos en movimiento, siendo en ocasiones necesaria la utilización de sensores especiales luz estroboscópica para sincronizar el momento en que se dispara la cámara. Para extraer la información de un objeto en una o varias imágenes se debe realizar de forma automática y no se debe establecer un contacto físico con el objeto. En entornos donde es necesario que en la aplicación realice reconocimiento de objetos estáticos, solo es requerida una buena iluminación que permita obtener buenas imágenes (ISRA VISION 2009, pág 11).

El estudio de las imágenes que se obtiene ayuda a detectar las características físicas de un objeto que son invisibles o imperceptibles al ojo humano, o también para comprobar propiedades físicas o químicas. La inteligencia artificial en el campo de la industria aumenta la calidad de los procesos de fábrica y la seguridad de los mismos como: control de calidad, pruebas y calibración de equipos, manipulación de materiales, entre otros; en el campo de la medicina se utiliza para el recuento y búsqueda de células y demás aplicaciones. En la actualidad las técnicas de visión artificial son utilizadas en el campo de diseño participativo por medio de la interacción con objetos, superficies multitáctiles o multitouch y para reconocimiento de señas corporales (ISRA VISION 2009, págs 11-12; García 2002, pág 5).

2.3.1. Configuración básica de un sistema de visión artificial

Los sistemas de visión artificial constan de dos pilares básicos:

- **Adquisición de imágenes:** mediante operaciones transforman la información luminosa de una imagen en señal digital almacenándola en forma digital. Además, este sistema está constituido por los subsistemas de iluminación, de captación de imágenes y el de adquisición de la señal en el ordenador.

- **Análisis de imágenes:** después de recibir la señal se procesa con algoritmos para transformar esa señal en información de alto nivel que puede ser utilizadas en una representación visual. Además, existen múltiples periféricos que son receptores de la información proporcionada y se pueden vincular a este sistema de procesamiento de imágenes (ISRA VISION 2009, págs 14-15; Platero 2014, pág 25).

En la figura 9-2 se presenta la configuración de cómo funcionan los subsistemas físicos de un equipo de visión artificial.

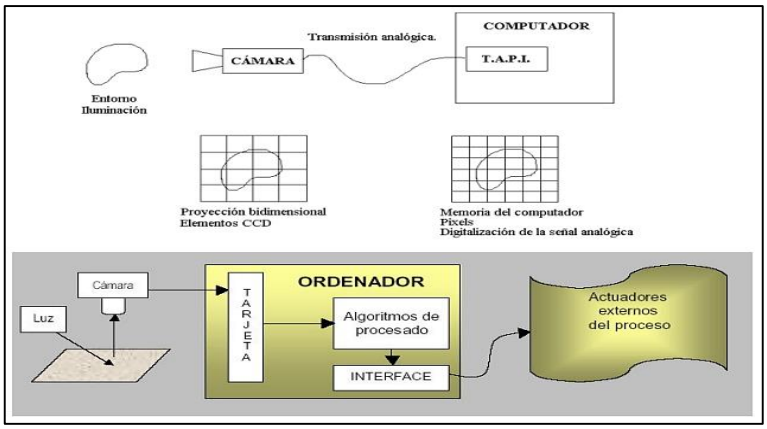


Figura 9-2. Configuración de los subsistemas de un equipo de visión artificial
Fuente: (Platero 2014, pág 26)

2.3.2. Etapas básicas en visión artificial

Los sistemas de visión artificial se asocian con conceptos de software y hardware en conjunción con los procesos teóricos por lo que en la figura 10-2 se presenta un diagrama de bloques de cómo debe ser un proceso de visión artificial.

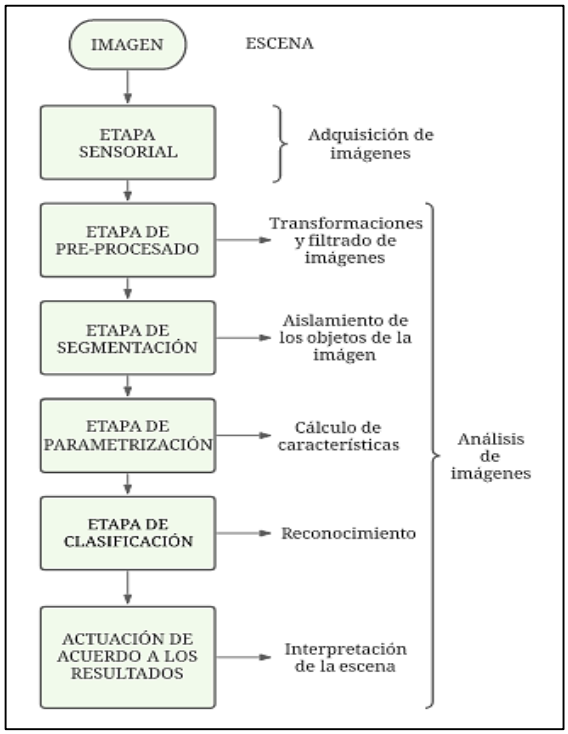


Figura 10-2. Etapas de un sistema de visión artificial
Realizado por: Muñoz, Jhonatan, 2021

2.3.2.1. Etapa sensorial

Esta etapa también es conocida como el sistema de adquisición de imágenes, hace que la imagen sea de lo más digna posible para que se pueda continuar con las otras etapas de la mejor manera.

Adquirir una imagen correcta es un paso muy importante para realizar con éxito el proceso de reconocimiento. En esta etapa existe muchos factores que pertenecen al proceso de captura de imagen: el sistema hardware (cámara, óptica, tarjeta de adquisición de imagen, ordenador y software) y el entorno y posicionamiento de elementos (iluminación, fondo, posición correcta de la cámara, ruido eléctrico-óptico externo, etc.). Por otro lado, en necesario obtener un buen calibrado del posicionamiento de la cámara obteniendo una buena imagen sin deformaciones (ISRA VISION 2009, pág 18).

2.3.2.2. Etapa de pre-procesado

Son un conjunto de técnicas que realizan una mejora de la imagen digital. Cada imagen que se ha obtenido ha pasado por diferentes medios ópticos y electrónicos o electro-ópticos que sufren los efectos de degradación manifestándose en forma de ruido, perdida de definición, perdida de fidelidad en la imagen, etc, para contrarrestar estos efectos se cuenta con operaciones de restauración que están incluidas en esta etapa. En forma general, se repara los desperfectos producidos o no eliminados por el hardware en la imagen. Cuando se realiza un proceso de visión artificial los algoritmos se deben usar en menor proporción, ya que al usarse excesivamente repercutirá en la etapa de adquisición (ISRA VISION 2009, pág 19; Ordieres et al. 2006, pág 15).

Realizar el filtrado en esta etapa es importante para eliminar el ruido existente en la imagen, para esto se hace uso de dos dominios: espacial y frecuencial. El dominio espacial se encarga de manipular directamente la luz de los pixeles; en cambio, el dominio frecuencial modifica indirectamente la luz de cada pixel que utiliza valores de otros pixeles como factores de ponderación o también puede utilizar valores del entorno del punto y las relaciones numéricas. Los filtros están basados en las Transformadas Discretas de Fourier (DFT) y las Transformadas Discretas de Fourier Inversa (IDFT) que determinan las frecuencias en las que se localiza el ruido y utilizan filtros tipo respuesta finita (FIR) e infinita (IIR) (ISRA VISION 2009, pág 19).

2.3.2.3. Etapa de segmentación

En esta etapa se fracciona la imagen en áreas con significado, es decir, se divide la imagen digital en las partes que la constituyen para diferenciar los objetos que componen la señal y determinar la posición con respecto al fondo de esta (Platero 2014, pág 28; Ordieres et al. 2006, pág 15).

2.3.2.4. Etapa de parametrización

También conocida como etapa de selección de rasgos, se dedica a extraer una serie de características que producen la información cuantitativa de interés o rasgos básicos para diferenciar una clase de otra, estas características son en lo posible independientes de su

localización, orientación y cambios de escala de los objetos, abarcando suficiente información para distinguir un objeto de otro (ISRA VISION 2009, pág 15,19; Ordieres et al. 2006, pág 15).

2.3.2.5. Etapa de clasificación

Cada característica analizada en la etapa anterior se debe clasificar e interpretar, diseñando clasificadores que le den a cada área dividida una identificación o etiqueta de alto nivel (Platero 2014, pág 28)

2.3.2.6. Actuación de acuerdo a los resultados

En esta etapa se interpreta la escena, es decir se le asigna un valor al conjunto de objetos que han sido reconocidos (Ordieres et al. 2006, pág 15).

2.3.3. Componentes de la visión artificial

Los componentes básicos de hardware para un sistema de visión artificial (SVA) se detallan a continuación:

- **Sensor óptico:** Son cámaras monocromáticas o a color que hacen una imagen completa del problema cada 1/30 segundos ($0,0\overline{3}$ s). También se puede considerar una cámara escáner produciendo una línea en cada momento; la imagen bidimensional se produce por el movimiento del objeto por la línea del escáner.

- **Tarjeta de adquisición de imagen:** Digitaliza la señal de video entregada por el sensor óptico.

- **Computador:** Después de digitalizar la imagen, se debe almacenar en la memoria del ordenador para luego ser procesada y así manipular el programa.

- **Monitor de video:** Visualiza las escenas captadas o imágenes que han resultado del procesamiento de estas imágenes (Ordieres et al. 2006, pág 16).

La figura 11-2 representan estos componentes del SVA en la que se visualiza el objeto observado, la iluminación, la cámara y el procesamiento que permite visualizar la imagen digital, el proceso para extraer los parámetros y posteriormente analizar y controlar el cómo actuar sobre el entorno del objeto ya sea para: clasificar, desestimar, trabajar sobre él, etc. (Ordieres et al. 2006, pág 16).

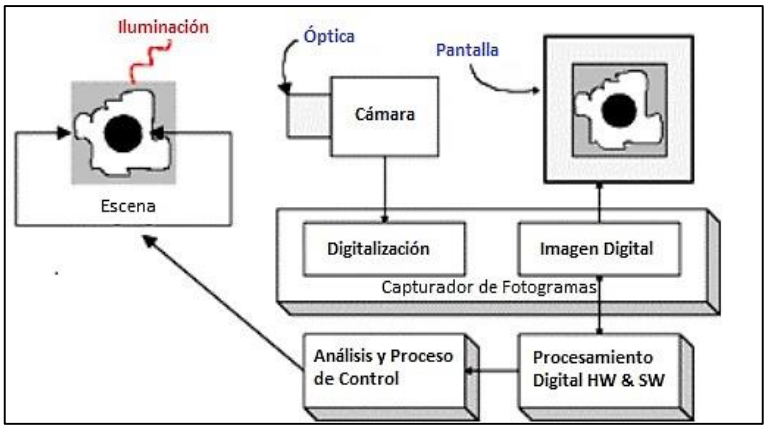


Figura 11-2. Sistema de Visión Artificial
Fuente: (Ordieres et al. 2006, pág 16)

2.3.4. Plataformas de visión artificial

Existen diversas plataformas físicas que dependen de los requisitos de la aplicación, el entorno de desarrollo, la capacidad, la arquitectura y el precio (COGNEX 2014, pág 21). A continuación, se describen las plataformas que existen:

2.3.4.1. Basada en PC

Este sistema se conecta fácilmente con las cámaras de conexión directas y las placas de adquisición de imágenes que son compatibles con el software de aplicaciones de los SVA. Por otro lado, los equipos PC tienen una gran afluencia para las opciones de desarrollo del código, respaldado en los programas de Visual C/C++, Visual Basic, Java y los entornos de programación gráficos pero, su desarrollo es extenso y complejo limitando su uso para grandes instalaciones (Platero 2014, pág 21).

2.3.4.2. Controladores de visión

Esta plataforma ofrece toda la potencia y la flexibilidad del sistema anterior, ya que están mejor preparados para los entornos hostiles. Así mismo mejoran la configuración de aplicaciones de multicámara 2D y 3D siendo esta una alternativa más robusta y con mejores beneficiosa (Platero 2014, pág 21).

2.3.4.3. Sistemas de visión independientes

Este sistema es muy rentable por su configuración rápida y fácil, se componen de un sensor de cámara, un procesador y comunicaciones (ethernet); en algunos sistemas integran la iluminación y óptica de enfoque automático. Como se conecta por ethernet, permite a los usuarios distribuir la visión por todo el proceso y enlazar dos o más sistemas en una red de área de visión, intercambiado datos que serán gestionados por un ordenador anfitrión, además se puede enlazar

con las redes de fábrica y la empresa permitiendo que cualquier estación de trabajo manifieste a distancia: resultados, imágenes, datos estadísticos y demás información. Sus entornos son sencillos y de fácil configuración (Platero 2014, pág 22).

2.3.4.4. Sensores de visión y lectores de códigos de barras basados en imágenes

Por lo general no requieren de programación, su interfaz es accesible y manejable integrándose con cualquier máquina, además cuentan también con comunicación Ethernet teniendo la posibilidad de enlazarse con toda la fábrica (Platero 2014, pág 22).

2.4. Software en la visión artificial

Realizar la ejecución de un algoritmo mediante la visión artificial (VA), en ocasiones resulta de un elevado costo tanto en la licencia del software como en la adquisición de cámaras, además se debe considerar el tiempo de desarrollo e implementación del algoritmo y funciones que lo conforman, para esto se lo debe someter a varias pruebas de funcionamiento en diferentes condiciones para realizar la evaluación del mismo. En el mercado existen diferentes lenguajes, paquetes y librerías de funciones que permiten trabajar con imágenes y sus diferentes formatos de las cuales solo se describirán algunos (ISRA VISION 2009, pág 27).

2.4.1. Matlab

"MATrix LaBoratory" de forma abreviada Matlab es un programa que nos posibilita hacer varios cálculos numéricos con matrices y vectores, en ocasiones se puede utilizar escalares y es excelente para representar gráficos en 2D y 3D. Matlab al permitirnos laborar con matrices es apto para el procedimiento de imágenes, convirtiendo la imagen en una matriz de pixeles que tiene un valor numérico de acuerdo al color y al tono de cada bit, sin embargo se debe añadir paquetes adicionales para hacer uso de esto. (ISRA VISION 2009, pág 27-28).

2.4.2. LabView

LabView permite laborar con imágenes en tiempo real y con mejores equipos para tener un mayor nivel en VA, el programa utiliza el módulo "LabView Ni Vision Development" que permite desarrollar aplicaciones de VA y de imágenes científicas. Incluye una biblioteca "IMAQ Vision" que tiene eficaces funciones para realizar un procesamiento de visión y un entorno muy interactivo llamado "Vision Assistant". Las dos principales funciones de este módulo son el procesamiento de imágenes (escala de grises, a color y binarias) y un alto nivel de VA con funciones y herramientas para procesar las imágenes y visualizarlas (ISRA VISION 2009, pág 28).

2.4.3. Python

Es un lenguaje de programación que se encuentra en diferentes aplicaciones y sistemas operativos siendo multiplataforma y multiparadigma destacado siempre por tener un código legible. Python es idóneo para trabajar con un cuantioso número de datos (Robledano 2019).

2.4.4. OpenCV

OpenCv es una librería que fue desarrollada por Intel, por sus siglas *"Open Source Computer Vision Library"* se basa en el procesamiento de imágenes en tiempo real, está escrito con el lenguaje de programación C/C++ haciéndolo un sistema multiplataforma para poder ejecutarse en diferentes sistemas operativos como Windows, Linux, Mac Os X, etc. Además, se puede utilizar con diferentes lenguajes de programación (ISRA VISION 2009, pág 29).

2.5. Hardware en la visión artificial

En el mercado existen diferentes opciones para seleccionar el hardware adecuado para este trabajo, por lo que se analizarán algunas opciones.

2.5.1. Cámaras fotográficas

Las cámaras son indispensables para los SVA ya que permiten capturar imágenes del objeto, esto se realiza capturando la luz reflejada sobre la superficie de un chip de silicio fotosensible. Las cámaras son los ojos del sistema que se van a encargar de captar o recibir la información del mundo exterior concretamente de la escena u objeto que va a ser el objetivo del estudio. La selección de una cámara depende de los requerimientos del sistema que se va a implementar. La cámara cumple la función de captar la imagen recibida de manera óptica a través del sensor que esta incorpora para poder ser procesada y transferida a un sistema electrónico para su posterior procesamiento (Catalunya 2012, pág 10; Oñate 2020, pág 7).

2.5.2. Componentes de una cámara fotográfica

Los componentes básicos de una cámara se presentan en la figura 12-2.

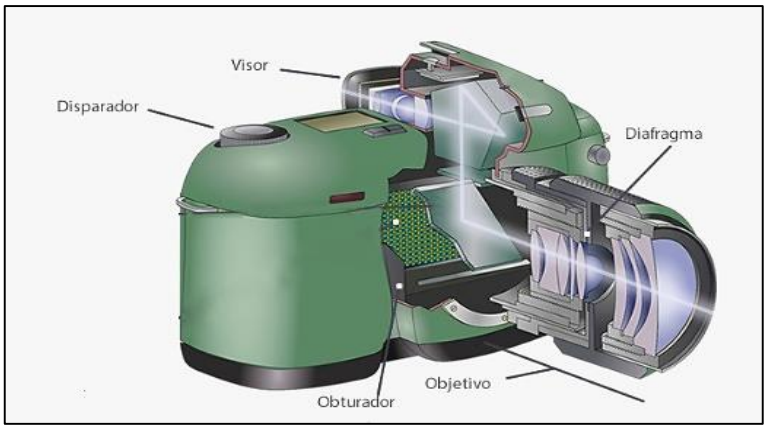


Figura 12-2. Componentes de una cámara fotográfica
Fuente: (Fernández 2013)

- **Objetivo:** Conjunto de lentes convergentes que forman la imagen real e invertida sobre el fondo de la cámara.

- **Diafragma:** Dispositivo regulable que permite controlar la cantidad de luz que accede en el interior de la cámara.

- **Obturador:** Controla el tiempo que la película es expuesta a la luz.

- **Visor:** Visualiza al sujeto a ser fotografiado componiendo y enfocando la escena (EDUCARM 2018).

2.5.3. Clasificación de las cámaras fotográficas

Las cámaras se clasifican en base la tecnología del sensor y su disposición física.

2.5.3.1. Tecnología del sensor

Esto es parecido a las películas fotográficas de las cámaras digitales. Dispuesto por una malla con millones de fototransistores cada uno de estos se denomina pixel como se muestra en la figura 13-2, así la imagen toma forma en base a la cantidad de luz que recibe cada malla (Cañadas y Haro 2011, págs 22-23).

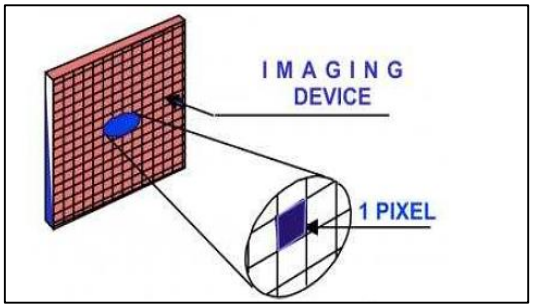


Figura 13-2. Celdas fotosensibles
Fuente: (Cañadas y Haro 2011, pág 23)

2.5.3.1.1. CMOS (Complementary Metal Oxide Semiconductor)

Los CMOS o semiconductores de óxido de metal, efectúan la conversión analógico-digital en el fotodiodo; al tener los sensores pequeños no es necesario utilizar elementos electrónicos para deslizar la carga. Se accede de forma individual a cada pixel ya que no requiere leer toda la imagen. La figura 14-2 representa la estructura del sensor CMOS en la que cada malla o celda es independiente y los pixeles se digitalizan internamente con unos transistores que lleva cada malla (Generalitat de Catalunya 2012, pág 11; Rosales 2017, pág 17; Cañadas y Haro 2011, pág 25).

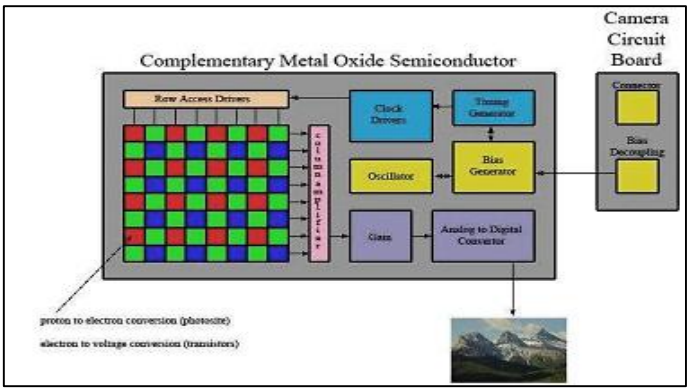


Figura 14-2. Estructura del sensor de imagen CMOS
Fuente: (Cañadas y Haro 2011, pág 24)

Como en cada malla se incorpora un amplificador para la señal, estos se vuelven más sensibles a la luz, logrando así videos con un mayor número de fotogramas (imágenes) por segundo "fps" (Cañadas y Haro 2011, pág 25).

2.5.3.1.2. Cámaras de estado sólido CCD

La tecnología CCD o dispositivo de carga acoplada, transforma la carga de cada malla en voltaje, asignando una señal analógica la cual se digitaliza por medio de un convertidor analógico digital, adicionalmente necesita de un chip para realizar este procedimiento. La figura 15-2 muestra la estructura de este tipo de sensores (Cañadas y Haro 2011, pág 23).

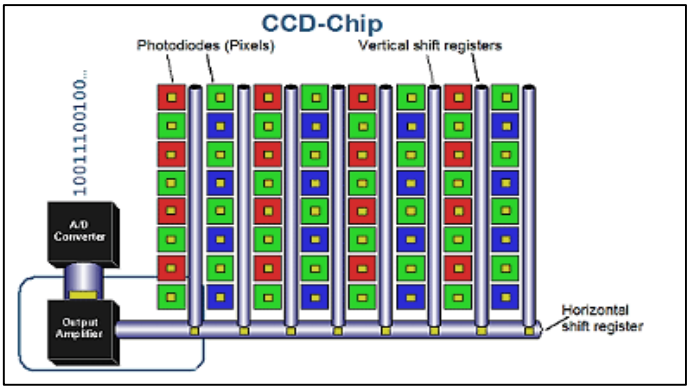


Figura 15-2. Estructura del sensor de imagen tipo CCD
Fuente: (Cañadas y Haro 2011, pág 24)

Los CCD están basados en materiales que son semiconductores fotosensibles los cuales no necesitan un barrido electrónico para la lectura. Al ser sensibles controlan de mejor manera el ruido obteniendo una imagen con una alta calidad, además presentan menos pixeles defectuosos con una ganancia uniforme en cada uno (Generalitat de Catalunya 2012, pág 11; Rosales 2017, pág 17).

2.5.3.2. Disposición física

Estas se clasifican en cámaras lineales las cuales van construyendo la imagen pixel a pixel, y de tipo matricial, que como su nombre lo indica, construye la imagen a través de una matriz de pixeles.

2.5.3.2.1. Cámaras lineales

Se lleva a cabo mediante un desplazamiento longitudinal para lo cual realiza un barrido del objeto para construir una imagen línea a línea como lo muestra la figura 16-2. Este tipo de cámaras utilizan sensores de gran calidad de imagen de entre 512 y 8192 píxeles, y al ser requerida una gran precisión para la obtención de una imagen los más fidedigna para su posterior procesamiento la alineación y sincronismo del sistema son de importancia crítica para el sistema (Generalitat de Catalunya 2012, pág 11).

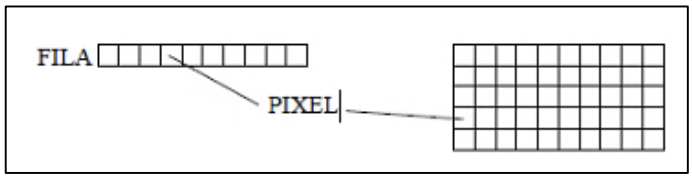


Figura 16-2. Desplazamiento longitudinal de una cámara lineal
Fuente: (Catalunya 2012,pág 11)

A continuación, se detallan varias de las características técnicas a tener en cuenta:

- **Número de elementos del sensor:** cuanto mayor son los números de pixeles, mayor será el tamaño de la óptica.

- **Velocidad:** son los números de pixeles que serán leídos por unidad de tiempo; teniendo un valor mayor que las matriciales.

- **Cámaras lineales a color:** presentan 3 sensores lineales en base a su color rojo, verde y azul, siendo de dos tipos: trisensor (la posición de los sensores CCD juntos a otros se separarán por un pequeño espacio; solo se utiliza en superficies planas) y prisma (se posicionan en las 3 caras del prisma, se usan para cualquier aplicación pero necesita mucha luz) (Generalitat de Catalunya 2012, pág 11).

2.5.3.2.2. Cámaras matriciales

Las cámaras matriciales utilizan la tecnología CCD (Dispositivo de carga acoplada) cuyos sensores están en la mayoría de las cámaras modernas; su estructura está compuesta por miles de diodos fotosensibles los cuales se encuentran posicionadas de manera precisa en toda la extensión del sensor formando una matriz de píxeles (Catalunya 2012, pág 12). Se debe tener en cuenta las siguientes características técnicas:

- **Factor de relleno:** es el porcentaje del área que es sensible a la luz en el pixel.

- **Tipo de transferencia:** se basa en la información obtenida como: transferencia inter-línea (ITL), transfiere los datos de la imagen que han sido registrados en los desplazamientos entre las líneas del pixel, transferencia de cuadro, tiene un área que se dedica a almacenar la luz y está separada del área activa permitiendo mayor relleno, transferencia de cuadro entero, utiliza un registro paralelo para la exposición de los cuantos de luz (Catalunya 2012, pág 12).

2.5.4. Dispositivos de captura de imágenes

Es el dispositivo encargado de capturar las imágenes del mundo exterior creando un conjunto de las mismas para obtener una grabación de video.

2.5.4.1. Cámaras web

Es un tipo de cámara digital que funciona al ser conectada a un computador, por lo general son usadas para video llamadas o grabaciones de videos. Su objetivo es captar las imágenes y transmitirlas a través de internet (López Rodríguez 2007, pág 17).

2.5.4.2. Cámaras IP o cámaras de red

Son cámaras digitales que están combinadas a equipos de unidades inteligentes capturando y transmitiendo imágenes en directo por medio de cualquier red IP, por ejemplo, LAN/Internet. Se debe emplear los protocolos IP para comunicarse con otro servidor de video con acceso a internet. Al comparar esta cámara con las cámaras web, estas no necesita estar conectar al computador a través del puerto USB para ser ejecutado (Oñate 2020, págs 10-11; López Rodríguez 2007, pág 20).

Existen diferentes modelos de cámaras IP que cumplen con los requerimientos que necesita el usuario, en la tabla 1-2 se presentan los modelos de cámaras IP con su representación gráfica y su respectiva descripción.

Tabla 1-2. Modelos de cámaras IP

Modelo	Representación gráfica	Descripción
Fijas		Es el tipo de cámara tradicional que es visible ante los usuarios y también se visualiza la dirección a la que apunta. Están diseñadas con carcasas de protección para los interiores y exteriores.
Domo fijas		Conocidas como mini domo, están constituidos por una cámara fija preinstalada en una carcasa de domo pequeña. Por su diseño son discretas y el usuario no visualiza en qué dirección apunta.
PTZ		Las cámaras PTZ, vertical/horizontal/zoom (Pan/Tilt/ Zoom) tienen una vista panorámica, inclinada o cercana/alejada de una imagen manual o automática. Son utilizadas en los interiores y exteriores, y el zoom, dependiendo del modelo oscila entre los 18x y 26x.
Domo		Permiten una rotación de 360° y tiene una resistencia mecánica para el funcionamiento continuo de los recorridos que realiza la cámara día a día. Este tipo de cámara por lo general supervisan una zona de las 10 posiciones de predefinidas. Tienen un zoom óptico que oscila entre los 18x y 30x, pero para zonas exteriores el zoom mayor es de 20x para no verse afectados por las condiciones externas (viento, vibraciones).
PTZ no mecánicas		Cuenta con un sensor de megapíxeles para una buena calidad de imagen, con una vista de la cámara panorámica, inclinada o cerca/alejada que abarca entre 140° y 360°, apunta en cualquier dirección sin realizar algún movimiento mecánico.

Fuente: (López Rodríguez 2007, págs 20-23)
Realizado por: Muñoz, Jhonatan, 2021

2.5.4.3. Cámaras de sistemas embebidos

Estas cámaras son de un tamaño reducido y tienen una alta calidad de imagen a pesar de su bajo costo aconteciendo su tendencia para los sistemas informáticos (Oñate 2020, pág 12).

2.5.4.4. Teléfonos inteligentes

En la actualidad son los sistemas informáticos más empleados por todos los usuarios por su capacidad de adquirir y almacenar datos a través de la red, existe un sin número de dispositivos que incorporan cámaras con múltiples funcionalidades que permiten capturar las imágenes (Oñate 2020, pág 13).

CAPÍTULO III

3. MARCO METODOLÓGICO

En este capítulo se especifican cada una de las etapas de desarrollo e implementación del sistema de detección detallando los requerimientos del mismo, hardware y software, exponiendo el proceso que se lleva a cabo desde el momento que se captura las imágenes de video hasta su representación en una página web.

3.1. Requerimientos del sistema

Por medio de los estudios realizados en los capítulos anteriores, se puntualizan los requerimientos para el sistema fijo de detección de lugares libres para el estacionamiento vehicular en el parque central de la ciudad de Baños:

- Procesar de las imágenes en tiempo real.
- Identificar disponibilidad en los espacios de parqueo.
- Trabajar de manera correcta sin verse afectado por las condiciones climáticas.
- Capaz de funcionar una vez que se inicie el programa en el computador y el dispositivo de grabación se encuentre conectado a la misma red.
- Tener una interfaz gráfica intuitiva.
- Ser capaz de transmitir la información de disponibilidad de parqueadero mediante una interfaz en una página web.

3.2. Concepción general

El diseño general de sistema está comprendido por 3 bloques principales y una etapa adicional como se puede apreciar en la figura 1-3. El bloque de adquisición de imágenes se encarga de recolectar y transmitir toda la información visual de la escena, el bloque de algoritmo computacional es el corazón del sistema el mismo que tiene la función de receptar, procesar y enviar al servidor web toda la información proporcionada por el bloque anterior, el bloque de difusión de resultados es el encargado de representar de manera gráfica los lugares disponibles para el estacionamiento vehicular capturando la interfaz creada por el programa de computadora

y enviándola a manera de trasmisión de video a una página web; y finalmente la etapa de acceso en la cual los usuarios acceden a la dirección web donde podrán interpretar y usar la información brindada por el sistema.

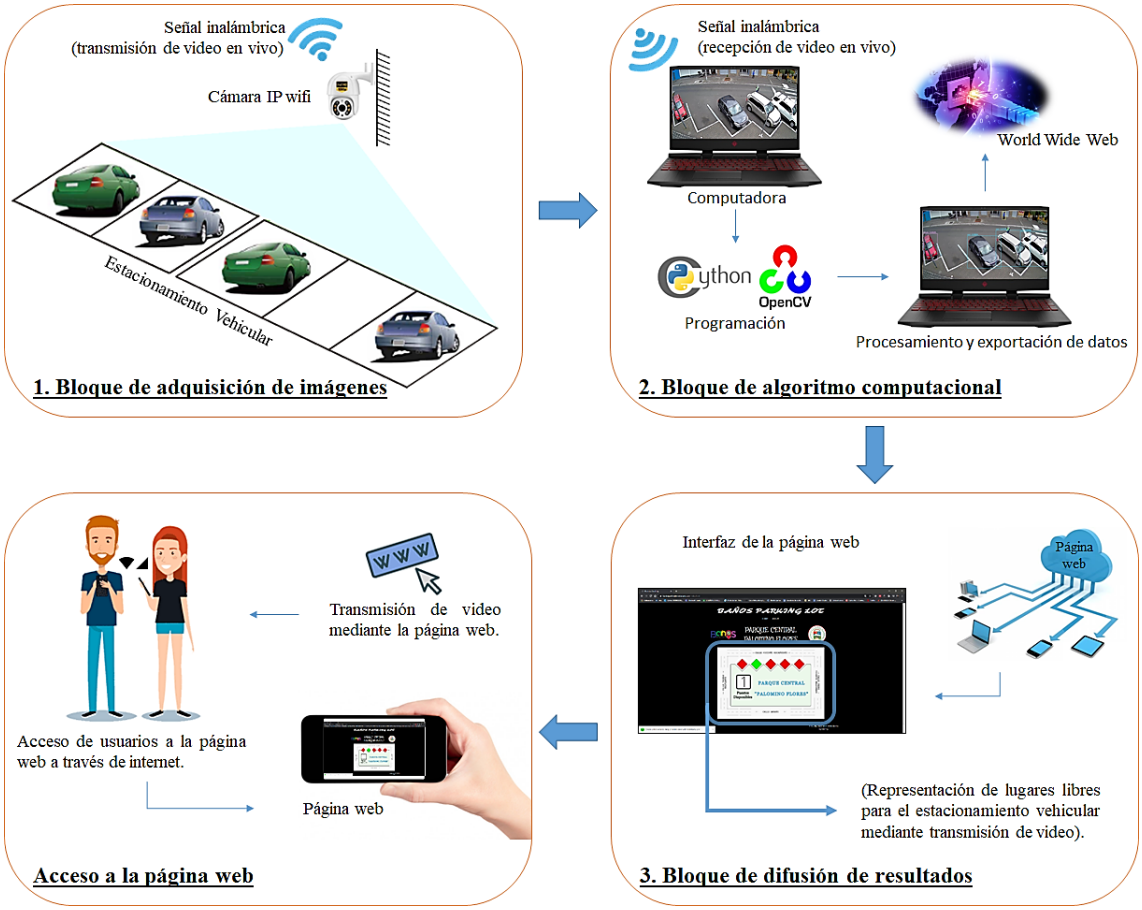


Figura 1-3. Concepción general de diseño del sistema fijo
Realizado por: Muñoz, Jhonatan, 2021

3.3. Diseño de la arquitectura

El proceso general a seguir para el desarrollo del sistema se manifiesta en la figura 2-3, la cámara capta las imágenes de video del estacionamiento vehicular, estas son importadas por el programa mediante líneas de código, sobre el video se establecen las regiones de cada espacio de estacionamiento, se procede con la detección de objetos dentro y fuera de las regiones de interés, la información es procesada y se procede con la extracción de las características pertinentes para la clasificación binaria (libre/ocupado) y su posterior visualización.

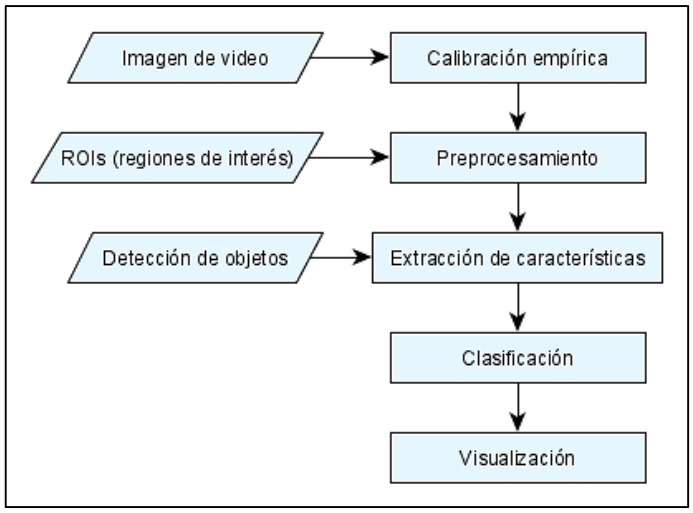


Figura 2-3. Diseño de la arquitectura del sistema fijo
Realizado por: Muñoz, Jhonatan, 2021

3.4. Hardware

A continuación, se describen los equipos que permiten el desarrollar del algoritmo para detectar lugares libres de estacionamiento. En el gráfico 1-3 muestra los elementos hardware que conforman el sistema.

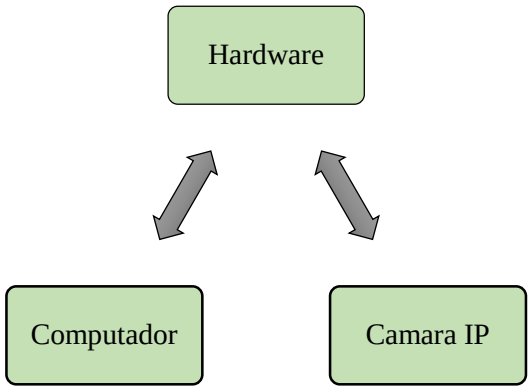


Gráfico 1-3. Elementos involucrados para desarrollar el algoritmo
Realizado por: Muñoz, Jhonatan, 2021

3.4.1. Equipos

La computadora y la cámara IP son las dos herramientas esenciales y necesarias, con estos dos equipos se puede realizar la identificación de disponibilidad de espacios en el parqueadero. Estos equipos son relativamente accesibles para la mayoría de personas, considerando importante mencionar las características principales de los mismos. Cabe recalcar que todo el procesamiento se lo llevo a cabo en una computadora personal (Hp Omen 15 modelo 2018).

3.4.1.1. Computador

Es la herramienta principal para realizar todo el procesamiento, en la tabla 1-3 se detallan las especificaciones del procesador que lleva el computador, siendo este el elemento que realiza la mayor parte del trabajo computacional

Tabla 1-3. Especificaciones técnicas del computador

Características	Descripción
Procesador	Intel Core i7 8750H
Tecnología	14 nn
Cantidad de núcleos	6
Cantidad de subprocesos	12
Frecuencia básica del procesador	2,20 GHz
Frecuencia turbo máxima	4,10 GHz
Caché	9 MB Intel® Smart Cache
Velocidad del bus	8 GT/s
Potencia de diseño térmico (TDP)	45 W
Descenso de TDP configurable	35 W

Fuente: HP Omen 15 (2018)
Realizado por: Muñoz, Jhonatan, 2021

Como datos adicionales a considerar, se puede destacar que la computadora cuenta con 16 GB (gigabytes) de memoria RAM DDR4 y una tarjeta gráfica GTX 1060 de 6GB GDDR5. Los detalles de la CPU del ordenador a utilizar se los obtuvo mediante el programa CPU-Z versión 1.94.0 de 64 bits como se puede observar en la figura 3-3.

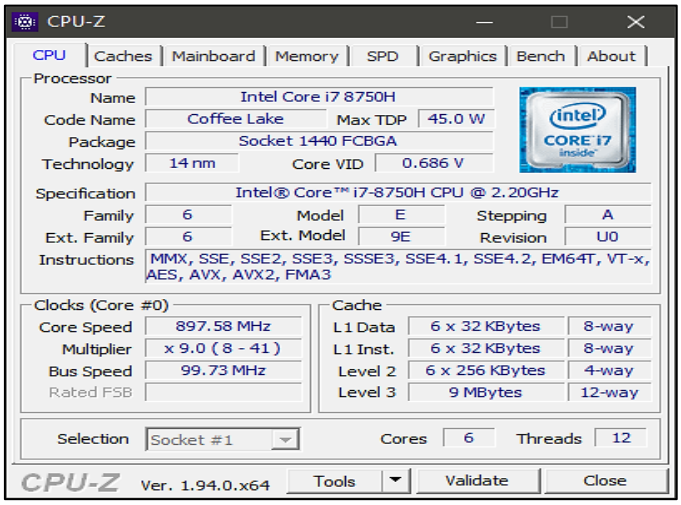


Figura 3-3. Características CPU del ordenador
Realizado por: Muñoz, Jhonatan, 2021

3.4.1.2. Cámara IP wifi 1080P

Las cámaras IP robótica tipo son una combinación entre una cámara y una computadora formando una única unidad, la cual capta y transmite las imágenes en vivo a través de una red, el modelo de la cámara es de tipo PTZ como se puede observar en la figura 4-3 (rnds 2016, pág 140).

Figura 4-3. Wifi Smart Camera
Realizado por: Muñoz, Jhonatan, 2021

3.4.1.3. Características de cámara IP wifi 1080P

La cámara IP wifi tienen diversas características entre las cuales:

- Permite la conexión inalámbrica mediante wifi 2.4GHz.
- La calidad de las grabaciones es en 1080P Full HD.
- El ángulo de grabación es de 100 grados.
- Al ser una cámara robótica el ángulo de grabación se modifica de forma manual o remota.
- La cámara es impermeable y cumple con el estándar IP66, lo que significa que se puede usar en el exterior, es resistente cierta medida al agua y al polvo.
- La temperatura de resistencia para un mal clima es de -10°C hasta 55°C.
- Tiene doble antena permitiendo tener una mejor recepción de red wifi.
- Presenta el modo de visión nocturna, tiene 4 luces led de alta potencia y 4 luces led blancas con un alcance de 10 metros (Montes 2006).

En la tabla 2-3 se presentan varias de las especificaciones técnicas de las cámaras IP wifi.

Tabla 2-3. Características técnicas de la cámara IP wifi 1080 PTZ

Características	Descripción
Modelo	BES-A8B
Lente	3.6 mm focal 2.0
Fuente de alimentación	DC12V/2A

Sistema	PAL
Conectividad	Wireless wifi
Ángulo de rotación	355°
Peso	1.5 lb
Dimensiones	(14 x 9 x 8) cm
Comprensión de video	H.265x
Pixeles efectivos	2 MP
Tamaño máximo de imagen	1920 x 1080

Fuente: Wifi Smart Camera
Realizado por: Muñoz, Jhonatan, 2021

3.5. Software

Para desarrollar el algoritmo que permitirá procesar las imágenes de video e identificar lugares libres para el estacionamiento se utiliza: Python en su versión 3.9.0, OpenCV en su versión 4.4.0, y el algoritmo de detección de objetos YOLO en su versión 3. Cabe recalcar que todo el software será ejecutado sobre el sistema operativos Windows 10 de 64 btis.

3.5.1. Python

Es un lenguaje de programación que está orientado a objetos que contiene una gran cantidad de librerías para desarrollar diversas aplicaciones. En la figura 5-3 se puede apreciar el editor de texto utilizado el cual es “Sublime Text” en su versión 3, también funciona como editor de código fuente, está escrito en escrito en lenguaje C++ y Python.

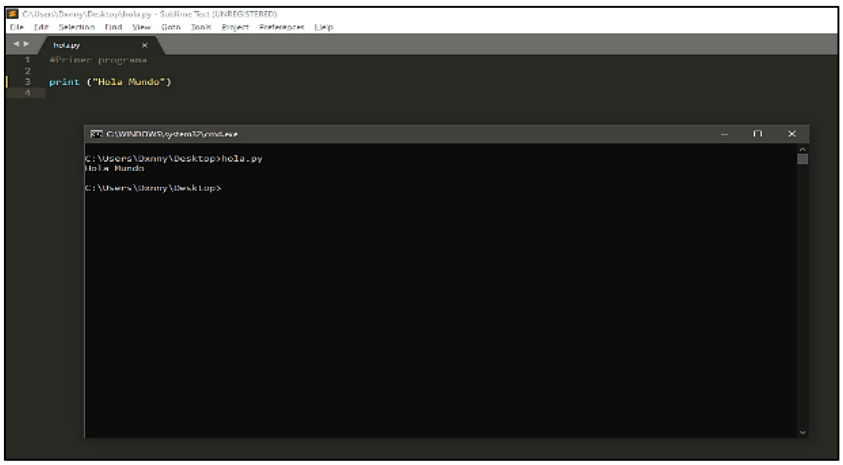


Figura 5-3. Primer programa en lenguaje Python, Sublime Text 3
Realizado por: Muñoz, Jhonatan, 2021

3.5.2. OpenCv

Es una librería de código abierto que fue desarrollada por Intel destinada a la visión por computadora además se la puede usar con diferentes lenguajes de programación; al trabajar en conjunción con Python se obtiene un sistema de visión artificial el mismo que es indispensable para el desarrollo del sistema. Para una fácil instalación de OpenCV se lo realiza mediante una line de código ("pip install opencv-contrib-python") desde el sistema DOS tal y como se puede apreciar en la figura 6-3. Cabe recalcar que este método de instalación no es oficial, es decir que no ha sido realizado por OpenCV.org no obstante es completamente funcional.

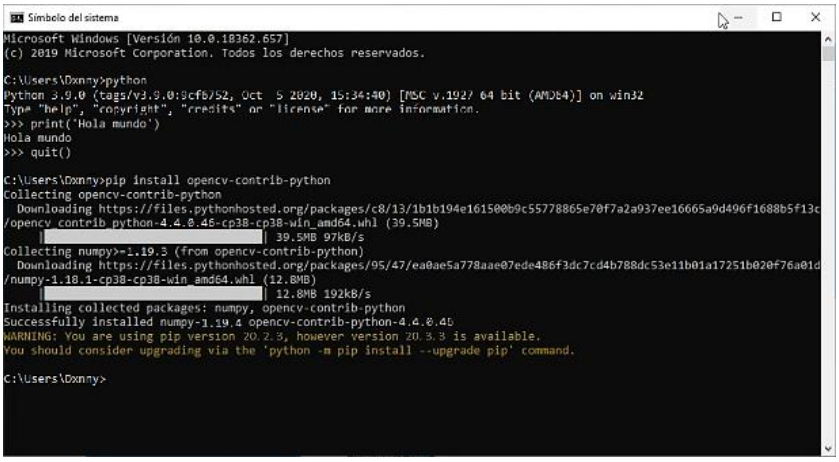


Figura 6-3. Instalación de OpenCV mediante el sistema DOS
Realizado por: Muñoz, Jhonatan, 2021

3.5.3. Yolo

"Solo miras una vez" (you only look once) es un algoritmo de código abierto utilizado para la detección de objetos en tiempo real haciendo uso de una sola red neuronal convolucional con la finalidad de detectar objetos en imágenes. La ventaja de usar este sistema es que se puede intercambiar entre precisión y velocidad sin requerir de reentrenamiento ya que su característica principal es hacer predicciones basándose en una única evaluación de red y por esta razón es más rápido que otros sistemas, como R-CNN y Fast R-CNN.

3.6. Desarrollo y representación del sistema de detección

En el gráfico 2-3 se presenta el proceso general que se seguirá para desarrollar el algoritmo de detección, desde la adquisición de imágenes hasta difusión de resultados.

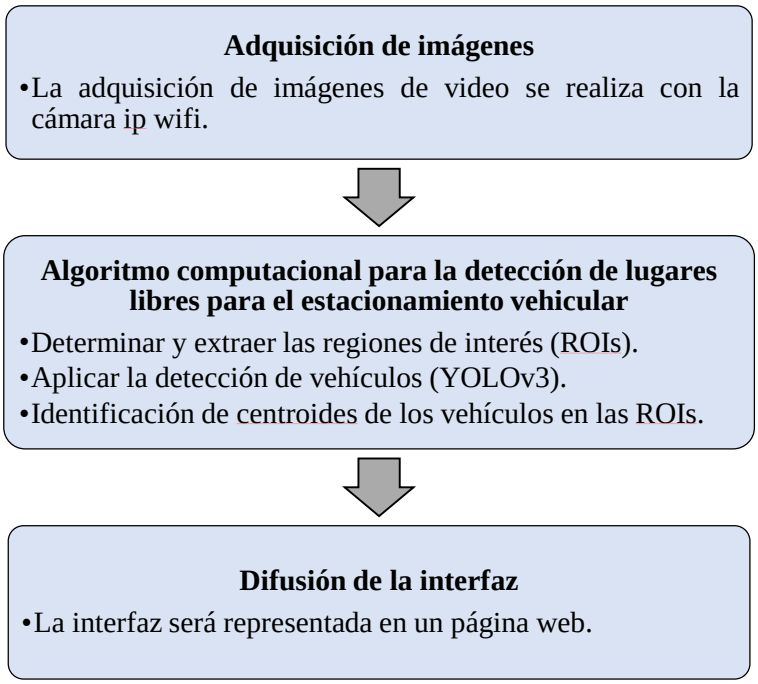


Gráfico 2-3. Diseño general del sistema de detección.
Realizado por: Muñoz, Jhonatan, 2021

3.6.1. Adquisición de imágenes

En esta etapa la herramienta principal es la cámara IP wifi; ya que es la encargada de adquirir las imágenes de video para la detección de vehículos. La conexión entre la cámara y el ordenador se lo realiza de manera inalámbrica cuya red tiene un alcance máximo que oscila entre los 30 metros a 50 metros. La resolución máxima utilizada para la grabación de video es de 1920 pixeles de ancho x 1080 de alto (Full HD). Es importante recalcar que el sensor de la cámara es de 2 mega pixeles consiguiendo una calidad de video relativamente buena.

Para la calibración de la perspectiva de la cámara se hizo uso de la aplicación ICSee, recomendada por el fabricante, esta permite rotar la cámara sobre su propio eje de manera horizontal y vertical facilitando así el encontrar la mejor vista para la extracción de video. Véase el Anexo A, instalación de la cámara

3.6.2. Algoritmo computacional

Para el desarrollo del algoritmo computacional, el encargado del tratar las imágenes de video proporcionadas por la cámara, se hará uso del ordenador cuyo recurso de procesamiento es la CPU, en la figura 7-3 se detalla el diagrama de flujo que seguirá el sistema desde la lectura de video hasta su transmisión en la página web.

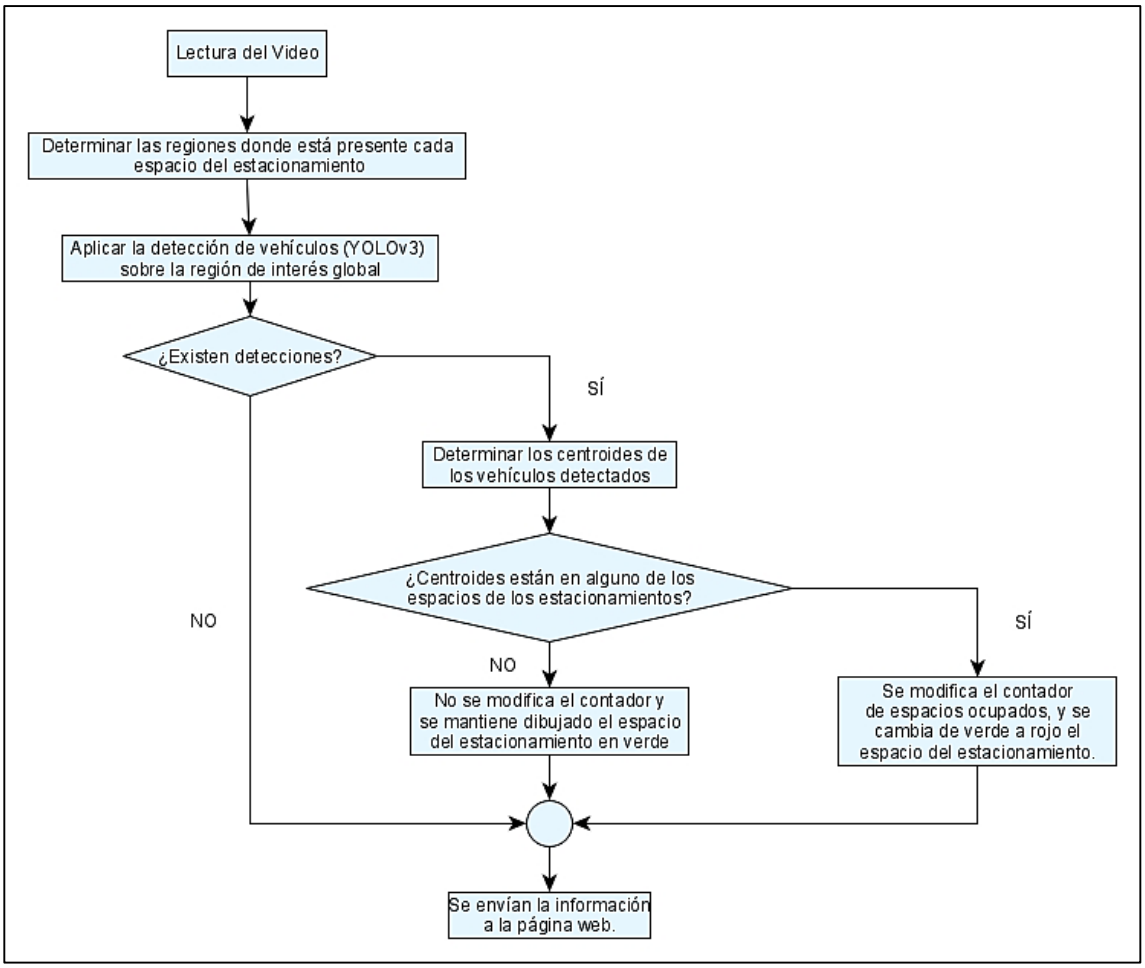


Figura 7-3. Diagrama de bloques de las etapas del sistema
Realizado por: Muñoz, Jhonatan, 2021

3.6.2.1. Lectura de video

La lectura de video de la cámara se la realiza a través de algoritmo que se desarrolla en lenguaje de programación Python. En la figura 8-3 se puede observar la perspectiva de grabación importada de la cámara, la misma que será utiliza para el procesamiento y desarrollo del algoritmo de detección de lugares libres para el estacionamiento.

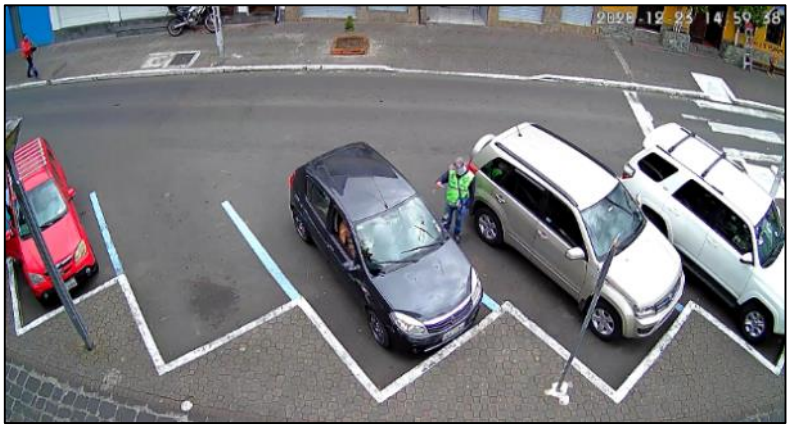


Figura 8-3. Perspectiva de grabación cámara IP wifi
Realizado por: Muñoz, Jhonatan, 2021

3.6.2.2. Detección

La capacidad de detección de objetos varía dependiendo de la calidad de imagen de video sobre la que se está trabajando y la distancia a la que se encuentre el objeto de la cámara, mientras mayor sea la distancia, menos será la capacidad de detección del sistema. La cámara de video tiene un ángulo de visión de 100 grados se encuentra ubicada en uno de los costados frente al estacionamiento, la distancia entre la cámara y los puestos del estacionamiento va desde los 10 metros hasta los 15 metros y la distancia a la calle por la que circulan los mismo es menor a 20 metros.

3.6.2.3. Regiones de Interés

Para determinar las regiones donde está presente cada espacio de estacionamiento se lo realiza de manera empírica analizando el video proporcionado por la cámara. Cabe recalcar que estas ROIs no serán representadas de manera visual superponiéndose al video, pero servirán para la interpretación y desarrollo del algoritmo como se muestra en la figura 9-3.

Figura 9-3. Seccionamiento de la imagen de video
Realizado por: Muñoz, Jhonatan, 2021

3.6.2.4. Algoritmo YOLO

Actualmente son 80 las etiquetas con las que puede trabajar la versión 3 de YOLO en la que se incluye la detección de personas, variedad de animales y demás cosas, algunas de estas etiquetas se pueden evidenciar en la figura 10-3. Para el caso práctico en cuestión se hará uso de 4 tipos de etiquetas diferentes las cuales son: auto, bus, camión y moto

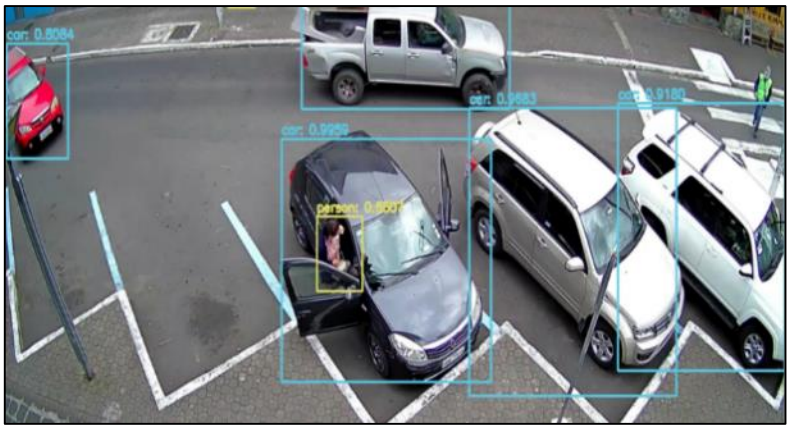


Figura 10-3. Detección de objetos mediante algoritmo YOLO versión 3.
Realizado por: Muñoz, Jhonatan, 2021

3.6.2.5. Centroide

Para determinar si un vehículo evidentemente se encuentra dentro de la región de un espacio de estacionamiento se determina el centroide del objeto, siendo este el punto de intersección con todos los hiperplanos del volumen un cuerpo. Una vez comprobado si el centroide del objeto en cuestión, en este caso el vehículo automotor, está dentro o fuera en la región de interés perteneciente un lugar de estacionamiento, se podrá determinar el estado del lugar de estacionamiento como se muestra en la figura 11-3.

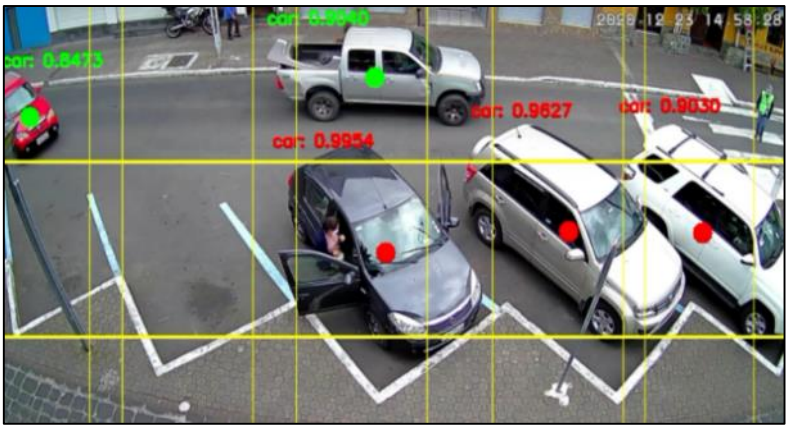


Figura 11-3. Detección de vehículos y sus centroide con regiones en todo el plano
Realizado por: Muñoz, Jhonatan, 2021

3.6.3. Difusión de la interfaz

Una vez se haya realizado todo el procesamiento de imágenes y determinando la existencia de un lugar disponible para el estacionamiento, los resultados serán representados mediante una interfaz gráfica relativamente fácil de interpretar, esta será mostrada a manera de transmisión de video en la página web "webstarts.com" usando un dominio gratuito.

En la figura 12-3 se puede observar el frame de transmisión de video el cual proyecta lo grabado por la cámara IP, sobre este están dibujadas las ROIs en los espacios en los cuales se estaría el vehículo estacionado. Esta ventana es utilizada para analizar de manera visual el correcto funcionamiento del sistema y no será exportada a la web.

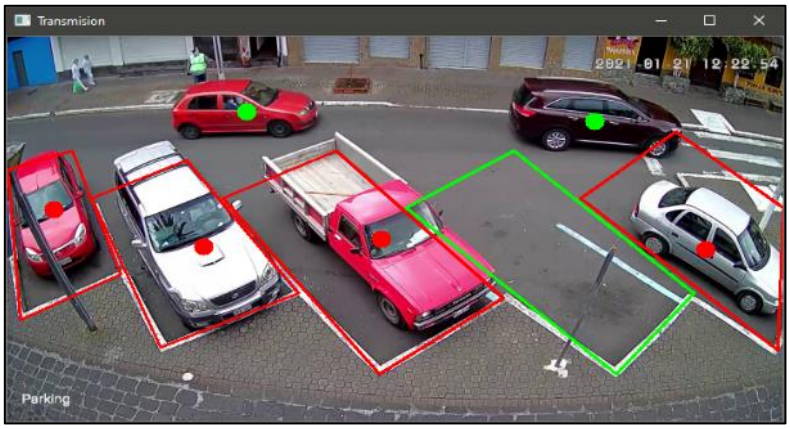


Figura 12-3. Frame de la interfaz de transmisión de video
Realizado por: Muñoz, Jhonatan, 2021

A manera de representar la información procesada por el algoritmo, se implementa una interfaz capaz de brindar solo la información relevante de la disponibilidad de estacionamiento, por esto en una imagen a manera de croquis, se especifica el lugar donde está implementado el sistema, las calles aledañas al sector, el número de puestos disponibles y los indicadores a manera de rombo como se muestra en la figura 13-3, estos se tornarán de color verde (libre) o rojo (ocupado) dependiendo si existe o no una vacante para estacionarse.

Figura 13-3. Frame de la interfaz de Mapa
Realizado por: Muñoz, Jhonatan, 2021

CAPÍTULO IV

4. RESULTADOS

En este capítulo se detallan los resultados en base a las pruebas en el desarrollo e implementación del sistema de detección partiendo de la descripción de las funciones que forman parte del algoritmo en cuestión. Las pruebas y resultados están conformadas principalmente por: tiempo de procesamiento, latencia de transmisión de video, consumo de recursos y la validación del sistema mediante la prueba de efectividad.

4.1. Desarrollo del software

El programa desarrollado se inicia mediante un archivo ejecutable ".py", este está contenido dentro de una carpeta la cual posee todos los archivos necesarios para su correcta ejecución; a continuación, se detallan las funciones que conforman el sistema de detección explicando las tareas que realizan cada una de ellas. Véase el Anexo B, código fuente.

4.1.1. Funciones del algoritmo

En el computador se realiza todo el procesamiento de imágenes de video brindadas por la cámara IP, las mismas a las cuales se puede acceder por medio de líneas de código en el lenguaje de programación Python conociendo a la dirección IP de la cámara dentro de la red. El algoritmo comienza importando las respetivas librerías y funciones destacando entre todas cv2 (openCV) y el algoritmo de detección de objetos yolov3.

4.1.1.1. Verificar desaparecidos

En esta función se verifica si el auto está desaparecido (no es detectado por un tiempo ínfimo) o ya abandonó el espacio de estacionamiento. Debe haber un número máximo de fotogramas en los que el auto no debe ser detectado para que el indicador visual cambie de estado ocupado a disponible.

4.1.1.2. Encontrar centroides

Aquí, con la ayuda de yolov3 se realiza la detección de autos, buses, camiones y motos, vehículos los cuales comúnmente hacen uso del parqueadero de donde se extrae sus respectivos centroides

y se los representa sobre el objeto detectado, representando el mismo con una circunferencia de color verde si está circulando por la calle o roja si se encuentra en un lugar de estacionamiento.

4.1.1.3. Determinar espacios ocupados

Determina si cada uno de los 5 espacios están ocupados asignándole un valor de 1 de ser el caso, o 0 si están disponibles, posterior a esto se devuelve una matriz de 5 elementos que pueden tener 0 o 1. Además esta función delimita cada región de interés perteneciente a cada espacio de estacionamiento.

4.1.1.4. Dibujar espacios

Funciona como indicador visual para el denominado frame (cuadro o ventana) de transmisión de video y de tipo mapa, estos indicadores serán regiones de los frames que se tornarán de color verde de ser el caso que haya un puesto de estacionamiento disponible o rojo en el caso contrario.

4.1.1.5. Video en vivo

Esta es la función general del sistema de detección la cual por medio del uso de las demás funciones anteriormente mencionadas y en base a una serie de condiciones, determina la disponibilidad de los espacios de estacionamiento generando 2 frames para la posterior captura de uno de ellos y su transmisión a modo de video en la página web.

4.1.2. Página web

Con la finalidad de que la información pertinente a la disponibilidad de estacionamiento y el lugar donde está implementado el sistema sea accesible para los usuarios, se opta por montar la información relevante al lugar estipulado en una página web.

Existen dos ventanas o subpáginas, la primera es la principal de inicio “Home” muestra en su encabezado el nombre del sistema además de pocos detalles informativos a manera de centrar la atención en el video montado en la página, video el cual muestra la interfaz tipo mapa o croquis del lugar en específico. Para esto se hace uso de ciertas herramientas: el programa “OBS Studio” se encarga de capturar el frame (ventana, cuadro) generado por el algoritmo en Python y hace las veces de intermediario con la plataforma “Twitch” la cual permite efectuar transmisiones de video en vivo. El video embebido es montado mediante el lenguaje “HTML” importándolo desde la plataforma anteriormente mencionada hacia la página web. La presentación inicial del sitio web puede ser apreciada en la figura 1-4.

Figura 1-4. Página web "ventana Home"
Realizado por: Muñoz, Jhonatan, 2021

La subpágina denominada "About" puede ser visualizada en la figura 2-4, esta muestra la información pertinente a lugar donde el sistema se encuentra implementado, brindando datos de: ubicación, lugar, dirección y una referencia además de incluir una extensión de "Google Maps".

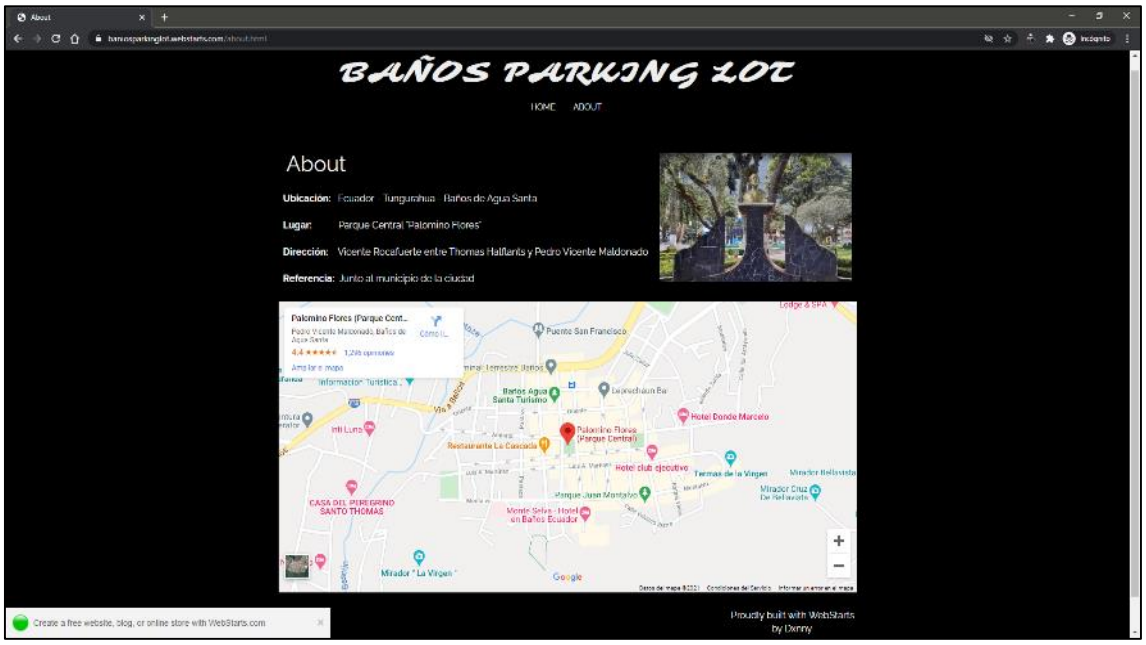


Figura 2-4. Página web "ventana About"
Realizado por: Muñoz, Jhonatan, 2021

4.2. Latencia de transmisión de video

Para determinar la diferencia de tiempo que existe entre los datos que están siendo procesados en el computador y lo que se ve el sitio web, se implementó un reloj en la transmisión para de esta manera encontrar la diferencia de tiempo entre la interfaz del computador y el video de la página web, obteniéndose un tiempo menor a 10 segundos en todo momento tal y como se puede observar en la figura 3-4.

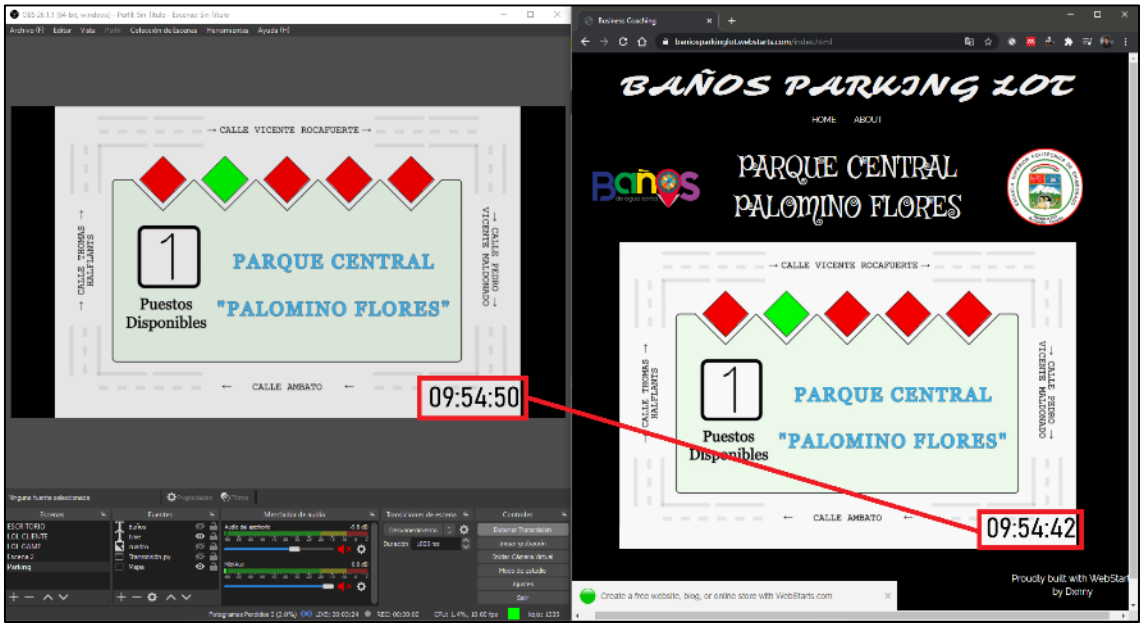


Figura 3-4. Latencia de transmisión de video
Realizado por: Muñoz, Jhonatan, 2021

Cabe recalcar que en todas las transmisiones de video a través de la web existe latencia, este retardo está sujeto a la calidad de conexión de la red (potencia) y a la velocidad de conexión a internet, para este caso se hace uso de una red con velocidad de 20 Mbps (Megabits por segundo) simétrica obteniéndose una latencia máxima aproximada de 8 segundos en todo momento sin ninguna pérdida de fotogramas.

4.3. Tiempo de procesamiento

Esta prueba tiene la finalidad de evaluar el tiempo de procesamiento del sistema de detección, la velocidad de procesamiento va a depender en su mayor parte de la velocidad de procesamiento del hardware y las propiedades del video que se está procesando. La tabla 1-4 muestra los valores de tiempo en ms (milisegundos) que se obtuvieron en la prueba durante la ejecución del sistema. Cabe recalcar que el video que se está procesando es a color, con una resolución escalada de 720p a 15 fps y una velocidad de datos de 1000 kbps (Kilobits por segundo)

Tabla 1-4. Tiempo de procesamiento de imágenes

Muestra	Estado	Tiempo (ms)
1	Detectando	85.77
2	Detectando	85.25
3	Detectando	87.53
4	Detectando	86.22
5	Detectando	86.51
6	Detectando	87.09
7	Detectando	83.79
8	Detectando	90.36
9	Detectando	86.78
10	Detectando	87.01
Media		**86.63**
Desviación estándar		**1.70**
Resultado		**86.63 ± 1.70**

Realizado por: Muñoz, Jhonatan, 2021

En comparación con los datos de la página del sistema de detección yolov3, el sistema de detección de objetos tiene un tiempo de respuesta de 51 ms versus los 86.63 ± 1.70 ms que se obtuvo en la aplicación de este sistema de detección, esta diferencia de aproximadamente 35 ms se lo puede atribuir a las diferencias de hardware que se utilizan en la implementación del sistema. A pesar de esto el tiempo de procesamiento de imágenes sigue siendo relativamente corto sin afectar al desempeño del sistema. Véase el Anexo C, tiempos de procesamiento.

4.4. Consumo de recursos

El objetivo de esta prueba es únicamente tener una noción de los recursos consumidos del computador para la ejecución del sistema, los datos fueron recopilados a través de la herramienta "administrador de tareas" que se incluye por defecto en los computadores con sistema operativo Windows como se muestra en la figura 4-4. Se tomaron como referencia los valores máximos alcanzados durante un periodo de observación de 15 minutos durante la ejecución del sistema. Las mediciones fueran hechas del consumo de CPU, memoria RAM, Red y GPU para el sistema de detección y el programa de transmisión.

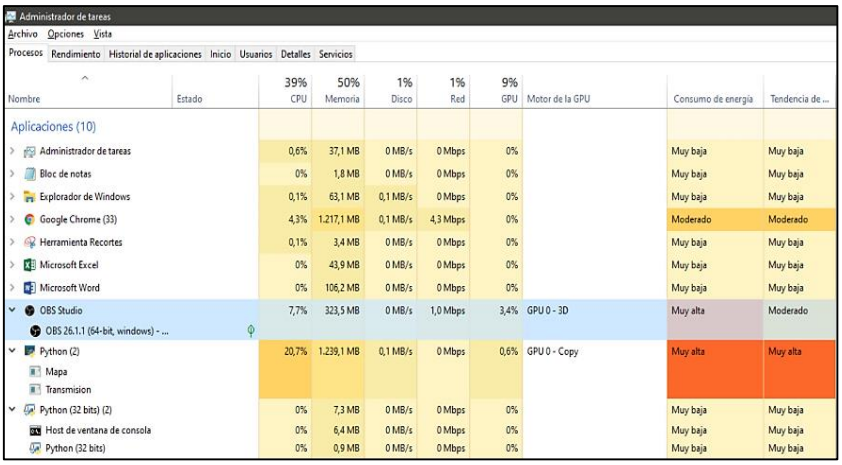


Figura 4-4. Consumo de recursos, administrador de tareas de Windows.
Realizado por: Muñoz, Jhonatan, 2021

4.4.1. Consumo de CPU

La "Unidad central de procesamiento" más conocida como CPU es la encargada de interpretar las instrucciones del computador siendo este el elemento hardware que se encarga casi en su totalidad del procesado del sistema. En el capítulo anterior se detallaron las especificaciones de la CPU que tiene la computadora en cuestión, por lo que ahora únicamente se representa en el gráfico 1-4 el consumo de los recursos en términos de porcentaje; en la ejecución del programa Python se alcanzó un valor máximo del 22%, y, un máximo del 10% para el programa de transmisión (OBS).

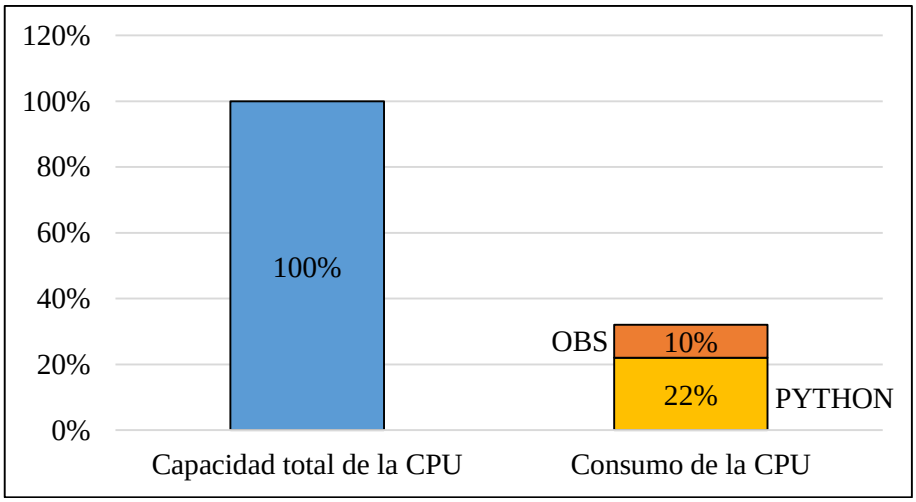


Gráfico 1-4. Recursos de la CPU
Realizado por: Muñoz, Jhonatan, 2021

4.4.2. Consumo de Memoria RAM

Esta memoria es más comúnmente conocida como memoria de almacenamiento temporal, en la ejecución del sistema presenta un consumo máximo de 1.3 GB para el programa Python y 326 MB para el programa OBS en la ejecución del programa, este valor de uso de memoria RAM no representa gran inconveniente ya que en la actualidad la mayor parte de computadores comerciales ya se de escritorio o portátiles tienen al menos 4GB de memoria RAM.

4.4.3. Consumo de la Red

Para la transmisión de video de la interfaz del programa hacia la página web, se hace uso del programa OBS, este por medio de la red, envía la información a través de internet teniendo un consumo máximo de 1.2 Mbps.

4.4.4. Consumo de GPU

En esta aplicación la tarjeta de memoria gráfica (GPU) trabaja en conjunción con la CPU en el proceso de detección de objetos y al mismo participa en la codificación del video a ser transmitido, OBS hace un uso del 2.3% de la GPU mientras que el solo el 0.8% es requerido por el programa de detección Python como se puede observar en el gráfico 2-4.

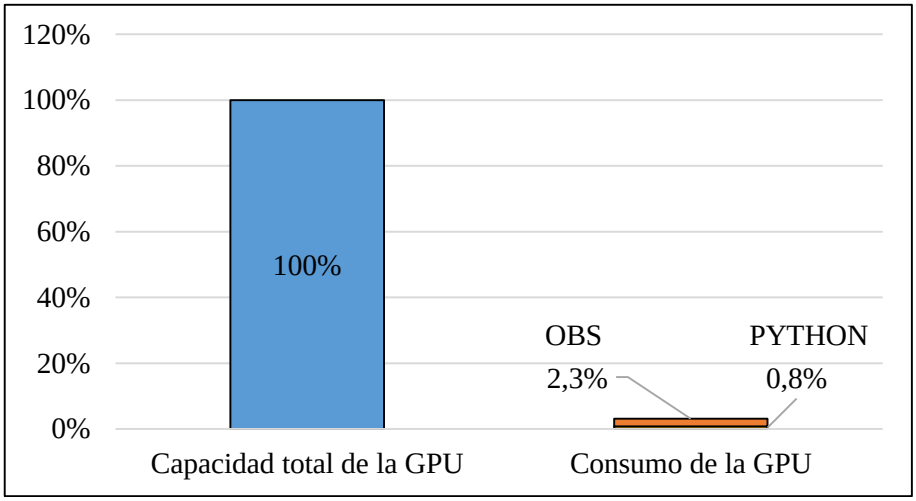


Gráfico 2-4. Recursos de la GPU
Realizado por: Muñoz, Jhonatan, 2021

4.5. Validación del sistema

Las pruebas que se realizan para esta fase se basan en la efectividad que tiene el sistema en detectar los objetos en los lugares de estacionamiento; estas pruebas fueron realizadas durante el periodo de 1 hora al día durante 10 días. El objetivo de estas pruebas es determinar la efectividad del sistema de detección de lugares disponibles en el estacionamiento vehicular del parque central de la ciudad de Baños.

4.5.1. Recopilación de datos

En la tabla 2-4 se muestra la información pertinente a la obtención de datos para la posterior prueba de efectividad. Esta recolección se realizó en un rango de horas entre las 11h00 a 13h30, considerando que en este periodo de tiempo existe mayor afluencia de vehículos. Como resultados

se obtuvo un total de 2563 vehículos en la escena del cual se extrajo una media de 256.3 ± 28.54 vehículos entre los que se incluye autos, camiones, buses y motos.

Tabla 2-4. Prueba de precisión del sistema de detección

Días	Autos	Buses	Camiones	Motos	Total
Día 1	206	4	15	15	240
Día 2	270	4	8	13	295
Día 3	186	4	5	21	216
Día 4	253	4	8	8	273
Día 5	262	4	7	16	289
Día 6	203	4	18	12	237
Día 7	191	4	7	20	222
Día 8	234	4	10	17	265
Día 9	198	4	16	24	242
Día 10	258	7	3	16	284
Media	**226.1**	**4.3**	**9.7**	**16.2**	**256.3**
Desviación estándar	**32.64**	**0.95**	**4.99**	**4.66**	**28.54**
Resultado	**226.1 ± 32.64**	**4.3 ± 0.95**	**9.7 ± 4.99**	**16.2 ± 4.66**	**256.3 ± 28.54**

Realizado por: Muñoz, Jhonatan, 2021

Del total de vehículos que se fueron contabilizados sólo el 4% tuvo la posibilidad de usar el estacionamiento en el periodo de 1 hora, en el gráfico 3-4 se puede observar que este porcentaje representa un total de 101 vehículos.

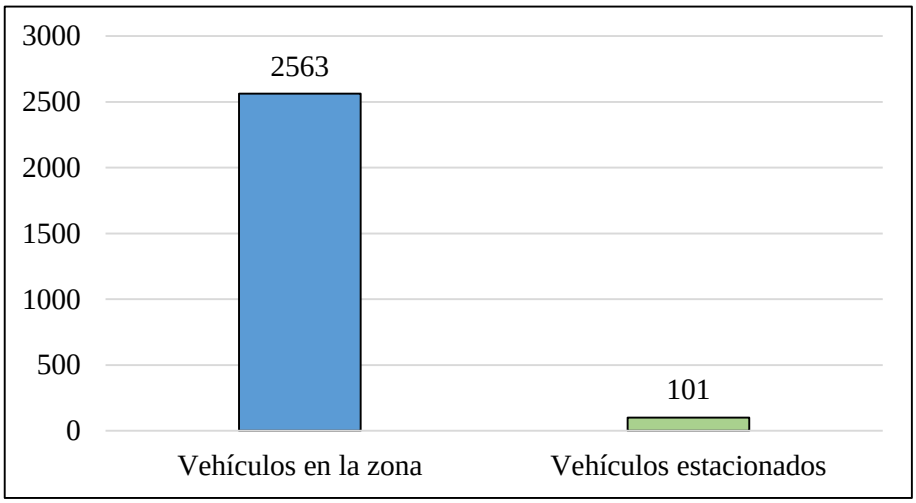


Gráfico 3-4. Flujo del estacionamiento
Realizado por: Muñoz, Jhonatan, 2021

En la tabla 3-4 se representan los datos obtenidos del flujo de vehículos que ocuparon los lugares de estacionamientos, dando como resultado 10.1 ± 1.52 vehículos que hacen uso del parqueadero.

Tabla 3-4. Flujo de vehículos estacionados

Días	Hora	Condición climática	Flujo estacionamiento
Día 1	12h20 – 13h20	Nublado	13
Día 2	11h15 – 12h15	Nublado	10
Día 3	11h25 – 12h25	Nublado	8
Día 4	11h25 – 12h25	Soleado	11
Día 5	11h35 – 12h35	Lluvioso	9
Día 6	11h30 – 12h30	Lluvioso	10
Día 7	11h25 – 12h25	Nublado	10
Día 8	11h55 – 12h55	Soleado	12
Día 9	11h30 – 12h30	Soleado	9
Día 10	11h45 – 12h45	Soleado	9
Media			**10.1**
Desviación estándar			**1.52**
Resultado			**10.1 ± 1.52**

Realizado por: Muñoz, Jhonatan, 2021

Debido a que en la noche no existe demasiada afluencia de vehículos la recolección de datos para las pruebas fue realizada en horas de la mañana, a pesar de esto el sistema no se ve afectado por las condiciones climáticas ya que, aunque el sistema se encuentre trabajando en un ambiente soleado, nublado, lluvioso o inclusive en horas de la noche, es capaz de detectar los objetos sin ningún tipo de problema, esto se debe a que en la zona existe buena iluminaria, específicamente postes de luz led color blanco. Véase el Anexo D, detección en distintas condiciones climáticas.

4.5.1.1. Falsos positivos

Estos datos hacen referencia a que en un determinado caso el sistema haga una falsa detección de un vehículo inexistente. El sistema en cuestión no presentó ningún falso positivo por lo que la precisión del mismo no se vio afectada por ningún dato de este tipo.

4.5.1.2. Falsos negativos

Son los vehículos que el sistema no los reconoce como tal a pesar de que realmente estén presentes. A lo largo de las pruebas realizadas durante los 10 días se obtuvieron 4 falsos negativos. Véase el Anexo E, falsos negativos.

4.5.2. Prueba de efectividad

Esta prueba se basa en el flujo de vehículos que ocuparon un lugar de estacionamiento; existen momentos puntuales que, durante la ejecución del sistema de detección, ciertos vehículos dejan de ser detectados por el sistema, estos datos no fueron considerados debido a que esto ocurre por

un tiempo ínfimo de segundos, para solucionar este inconveniente el sistema de detección cuenta con la función "Verificar desaparecidos" mencionada anteriormente, la misma que incorpora un tiempo de espera de 4 segundos para volver a detectar el vehículo perdido, evitando que el indicador visual libre (verde) y ocupado (rojo) cambie de estado innecesariamente.

En la tabla 4-4, se detallan los datos de la detección de vehículos en la zona de estacionamiento, obteniendo un resultado de 10.1 ± 1.56 para el flujo del estacionamiento, 0 falsos positivos y 0.4 ± 0.52

Tabla 4-4. Datos del flujo del estacionamiento, falsos positivos y falsos negativos

Días	Flujo estacionamiento	Falsos +	Falsos -
Día 1	13	0	0
Día 2	10	0	1
Día 3	8	0	0
Día 4	11	0	0
Día 5	9	0	1
Día 6	10	0	0
Día 7	10	0	0
Día 8	12	0	1
Día 9	9	0	1
Día 10	9	0	0
Media	**10.1**	**0**	**0.4**
Desviación estándar	**1.56**	**0**	**0.52**
Resultado	**10.1 ± 1.56**	**0**	**0.4 ± 0.52**

Realizado por: Muñoz, Jhonatan, 2021

Para realizar esta prueba se tomó en cuenta el flujo vehicular en el estacionamiento y el número de falsos negativos registrados descartando los falsos positivos al no existir alguno. La efectividad del sistema se la realiza comparando el total de vehículos detectados y el total de falsos negativos que, en otras palabras, son los considerados "vehículos no detectados".

En la tabla 5-4 se evidencian los resultados pertenecientes a los 10 días de prueba, con un total de 97 vehículos detectados y 4 vehículos no detectados, que sumados dan un total de 101 vehículos que hicieron uso del parqueadero; en la columna final, se muestra la efectividad de cada día de prueba obteniendo una media de las misma del 96%

Tabla 5-4. Resultados de la prueba de efectividad del sistema de detección

Días	Detectados	No Detectados	Total Flujo Estacionamiento	Efectividad
Día 1	13	0	13	100%
Día 2	9	1	10	90%

Día 3	8	0	8	100%
Día 4	11	0	11	100%
Día 5	8	1	9	89%
Día 6	10	0	10	100%
Día 7	10	0	10	100%
Día 8	11	1	12	92%
Día 9	8	1	9	89%
Día 10	9	0	9	100%
Total	**97**	**4**	**101**	**96%**

Realizado por: Muñoz, Jhonatan, 2021

El gráfico 4-4 muestra la validación del sistema de detección en términos de porcentaje teniendo un 96% de efectividad como ya se había mencionado en el resultado de media de efectividad en los 10 días de prueba.

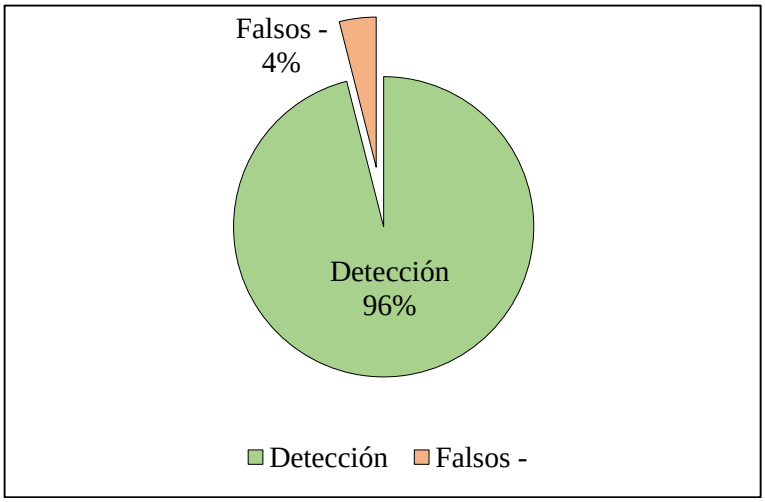


Gráfico 4-4. Efectividad del sistema de detección
Realizado por: Muñoz, Jhonatan, 2021

CONCLUSIONES

- Al realizar la revisión bibliográfica sobre los diferentes sistemas de parqueaderos inteligentes en el Ecuador realizados con diferente metodología como son: basados en sensores inalámbricos, en una tarjeta Raspberry Pi y en un Arduino Yun los mismo que presentaban óptimos resultados, se logró adquirir los conocimientos básicos necesarios para el correcto desarrollo del proyecto.

- En la selección del hardware adecuado para el correcto desarrollo del algoritmo se tomaron en cuenta los siguientes parámetros como son: buena calidad de grabación de video, ángulo de grabación, conexión inalámbrica y capacidad para trabajar en exteriores. La selección del software necesario consideró que: sea multiplataforma, tenga la posibilidad de importar videos fuente y trabajar con un cuantioso número de datos y funciones, y tenga la capacidad de trabajar con bibliotecas y sistemas orientados a la visión artificial.

- Se ha logrado desarrollar e implementar un algoritmo que, a través de la visión artificial orientada a la detección de vehículos, es capaz de reconocer lugares libres para el estacionamiento vehicular y a la vez brindar la información necesaria de la disponibilidad de los mismos en tiempo real mediante la transmisión de video de la interfaz creada en una página web.

- En base a las pruebas realizadas para la validación del sistema de detección, se pudo determinar que este presenta una efectividad del 96%, además tiene la capacidad para trabajar adecuadamente y de forma constante sin verse afectado por las condiciones climáticas en el sector en específico, cumpliendo así, con todos los requerimientos del sistema y todos los objetivos que se plantearon en el presente Trabajo de Titulación.

RECOMENDACIONES

- Extender el periodo de pruebas y recopilación de datos en pro de encontrar posibles mejoras para el sistema de detección.

- Realizar pruebas en ambientes con un mayor número de estacionamientos para evaluar el rendimiento del sistema.

- Al usar comunicación inalámbrica se recomienda el uso de una red con acceso a internet con un mínimo de 5 Mbps para tener un buen desempeño.

- En el sector el servicio eléctrico es constante, no obstante, se puede adicionar un sistema de alimentación ininterrumpida para que, en el caso de existir fallas eléctricas, el sistema pueda seguir operando.

- Realizar pruebas en ambientes con menor cantidad de luz para determinar el mínimo de luminosidad que el sistema necesita para funcionar adecuadamente.

BIBLIOGRAFÍA

ACHARYA, D., YAN, W. y KHOSHELHAM, K., 2018. Real-time image-based parking occupancy detection using deep learning. *CEUR Workshop Proceedings*, vol. 2087, no. September 2020, pp. 33-40. ISSN 16130073.

ISRA VISION, 2009. Introducción al diseño de Micro Robots móviles.

ÁLVAREZ DURÁN, M.A., 2014. *Análisis, diseño e implementación de un sistema de control de ingreso de vehículos basado en visión artificial y reconocimiento de placas en el parqueadero de la Universidad Politécnica Salesiana - Sede Cuenca*. S.l.: Universidad Politécnica Salesiana.

AXITY, 2015. Smart Agro. [en línea]. [Consulta: 15 diciembre 2020]. Disponible en: https://www.axity.com/es/tecnologias/smart-agro/.

BASU, A., 2014. *Smart Parking*. 2014. S.l.: s.n.

CAÑADAS, P. y HARO, G., 2011. *Prototipo de un sistema de adquisición de imágenes de vehículos, detección y reconocimiento automático de los caracteres de la placa en tiempo real por medio de visión artificial, aplicado al control vehicular*. S.l.: Escuela Politécnica Nacional.

COGNEX, 2014. Introducción a la Visión Artificial., no. noviembre, pp. 2-4. DOI 10.13140/2.1.1072.6722.

CONTAVAL, 2016. ¿Qué es la visión artificial y para qué sirve? [en línea]. Disponible en: https://www.contaval.es/que-es-la-vision-artificial-y-para-que-sirve/.

ECOPARKING, 2016. Parqueadero Multinivel. *ecoparking.co* [en línea]. Disponible en: https://ecoparking.co/.

EDUCARM, 2018. La cámara. *Fotografía analógica y digital*.

ERAZO, C., 2019. *Prototipo de detección de aparcamientos libres mediante visión artificial en un parqueadero de la Universidad Técnica del Norte*. S.l.: Universidad Técnica del Norte.

ERAZO, C. y NARVÁEZ, S., 2019. Prototipo de detección de aparcamientos libres mediante visión artificial en un parqueadero de la Universidad Técnica del Norte. ,

FERNÁNDEZ, J., 2013. La Cámara Fotográfica. *FISICALAB* [en línea]. Disponible en: https://www.fisicalab.com/apartado/camara-fotos.

FRANCO, R., 2014. Internet de las cosas. *PMQuality*, pp. 1-19.

GARCÍA, E.M. i, 2002. Visión Artificial. *Inteligencia Artificial*, pp. 115.

GENERALITAT DE CATALUNYA, 2012. Aplicación práctica de la visión artificial en el control de procesos industriales.

IAC, 2018. ¿Qué es IoT? [en línea]. Disponible en: https://www.iac.com.co/que-es-iot/.

INEC, 2019. Anuario de Estadísticas de Transporte 2018. *Informes Anuales*, pp. 46.

LÓPEZ RODRÍGUEZ, J.C., 2007. *Estructura, Funcionamiento y aplicaciones de las cámaras IP*. S.l.: Universidad Autónoma del Estado de Hidalgo.

MÁRQUEZ, M., LARA, R. y GORDILLO, R., 2014. Prototipo de Parqueadero Inteligente mediante Red de Sensores Inalámbricos. *Congreso de Ciencia y Tecnología ESPE*, vol. 9, no. 1, pp. 174-181. ISSN 1390-4663.

MÁRQUEZ ROSERO, M.D., 2015. *Desarrollo de un prototipo de parqueadero inteligente empleando redes de sensores inalámbricos*. S.l.: Universidad de las Fuerzas Armadas-ESPE.

MONTES, W., 2006. Cámara IP Robótica Exterior Seguridad 1080 PTZ Wifi 8 Leds. *Miros Player* [en línea]. Disponible en: https://mirosplayer.com/producto/camara-ip-robotica-exterior-seguridad-1080-ptz-wifi-8-leds/.

OÑATE, F.P., 2020. *Diseño y construcción de nodos inteligentes para detección de armas dentro de una red de video- vigilancia utilizando visión artificial*. S.l.: ESPOCH.

ORDIERES, J., et.al., 2006. *Técnicas y algoritmos básicos de visión artificial*. S.l.: Universidad de La Rioja. Servicio de Publicaciones. ISBN 8495301741.

ORDOÑEZ-CAMACHO, D., GÓMEZ, E. y ÁVALOS, H., 2018. An architectural proposal for an automatic vacant parking detection system. *Proceedings - 3rd International Conference on Information Systems and Computer Science, INCISCOS 2018*, vol. 2018-Decem, pp. 351-355. DOI 10.1109/INCISCOS.2018.00057.

PARKING SYSTEM, 2019. Conoce 4 impresionantes modelos de estacionamiento inteligente. *Parking System.com* [en línea]. Disponible en: https://www.parkingsystems.com.co/conoce-4-impresionantes-modelos-de-estacionamiento-inteligente/.

PLATERO, C., 2014. Tema 1.- Introducción a la Visión Artificial. *Universidad de Córdoba: Escuela Politécnica Superior*, pp. 37.

RNDS, 2016. Cámaras de red / Cámaras IP. *www.rnds.com.ar*,

ROBLEDANO, Á., 2019. Qué es Python: Características, evolución y futuro. [en línea]. [Consulta: 28 diciembre 2020]. Disponible en: https://openwebinars.net/blog/que-es-python/.

ROSALES, C., 2017. *Prototipo de detección de expresiones corporales mediante visión artificial para mejorar la comunicación con niños que tienen parálisis cerebral infantil*. S.l.: Universidad Nacional de Loja.

ROSALES, L., 2016. *Diseño e implementación de un parqueo inteligente utilizando Arduino Yun basado en internet de las cosas (loT)*. S.l.: Universidad Politécnica Salesiana sede Guayaquil.

SHOEIBI, Nastaran y SHOEIBI, Niloufar, 2020. Future of smart parking: Automated valet parking using deep Q-learning. *Advances in Intelligent Systems and Computing*, vol. 1004, pp. 177-182. ISSN 21945365. DOI 10.1007/978-3-030-23946-6_20.

SINETI, 2020. IoT en la manufactura. [en línea]. [Consulta: 15 diciembre 2020]. Disponible en: https://sineti.com/2020/01/15/iot-en-la-manufactura-sap-connected-assets/.

TENESACA, F. y HIDALGO, G., 2019. Boletín técnico N°-01-2019-Transporte. Quito.

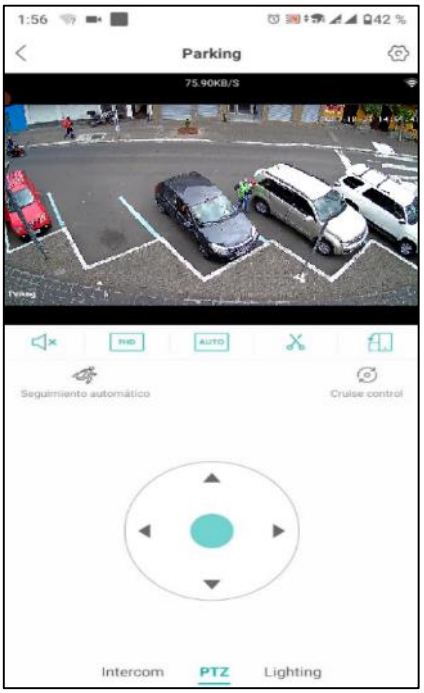
TIRUPATHI, M. y KRISHNA, R., 2017. Applications of IOT: A study. *Indian Journal of Public Health Research and Development*, vol. 9, no. 11, pp. 86-87. ISSN 09760245. DOI 10.5958/0976-5506.2018.01616.9.

TSCHENTSCHER, M. y NEUHAUSEN, M., 2012. Video-based parking space detection. *Proceedings of the Forum Bauinformatik*, no. 2007, pp. 159-166.

XU, J., CHEN, G. y XIE, M., 2000. Vision-guided automatic parking for smart car. *IEEE Intelligent Vehicles Symposium, Proceedings*, pp. 725-730. DOI 10.1109/ivs.2000.898435.

ANEXOS

ANEXO A: Instalación de la cámara IP wifi

Estructura de acoplamiento	**Fijación a nueve metros de altura**
	
Conexión a la red eléctrica	**Aplicación iCSee**
	

ANEXO B: Código Fuente

Programa principal

```
from flask import Response
from flask import Flask
from flask import render_template
import threading
import imutils
import cv2
from scripts import utils

outputFrame = None
outputImage = None
lock = threading.Lock()

# Inicialización a flask object.
app = Flask(__name__)

# Video.
cap=cv2.VideoCapture('rtps://admin:password@ip:554')

# Especificar la imagen de entrada (mapa).
orig = cv2.imread('static/img/interfaz.png')
orig = imutils.resize(orig, width=1080)

# Lectura Labels.
labelsPath = "yolo-coco/coco.names"
LABELS = open(labelsPath).read().strip().split("\n")

# Leer los pesos y estructura de yolov3.
weightsPath = "yolo-coco/yolov3.weights"
configPath = "yolo-coco/yolov3.cfg"
net = cv2.dnn.readNetFromDarknet(configPath, weightsPath)
net.setPreferableBackend(cv2.dnn.DNN_BACKEND_CUDA)
net.setPreferableTarget(cv2.dnn.DNN_TARGET_CUDA)
ln = net.getLayerNames()
ln = [ln[i[0] - 1] for i in net.getUnconnectedOutLayers()]
```

```
# Fin Video.

def video_vivo(frameCount):
	global cap, outputFrame, lock, orig, outputImage

	# Número mínimo de autos detectados.
	min_detectados = 1

	# Contedo de autos detectados.
	count_detectados1 = 0
	count_detectados2 = 0
	count_detectados3 = 0
	count_detectados4 = 0
	count_detectados5 = 0

	# Contedo de autos perdidos (no existe auto en el espacio).
	count_perdidos1 = 0
	count_perdidos2 = 0
	count_perdidos3 = 0
	count_perdidos4 = 0
	count_perdidos5 = 0

	# Auxiliares
	aux1 = 0
	aux2 = 0
	aux3 = 0
	aux4 = 0
	aux5 = 0

	while True:

		ret, frame = cap.read()
		if ret == False: break
		frame = imutils.resize(frame, width=720)
		image = orig.copy()

		# Determinamos los centroides de los vehículos detectados.
```

```
centroides = utils.encontrar_centroides(frame, net, ln,)

# Total de espacios disponibles
disponibles = 0

# Si existen centroides:
if centroides != []:

    espacios = utils.determinar_espacios_ocupados(centroides)

    ### PRIMER ESPACIO
    # Si se detectó más de una vez un auto/bus/camión/moto
    #el espacio está en 1 sino está en 0.

    if espacios[0] == 1:
        count_detectados1 = count_detectados1 + 1
        if count_detectados1 > min_detectados:
            espacios[0] = 1
            aux1 = 1
        else:
            espacios[0] = 0

    if espacios[0] == 0 and aux1 == 1:
        count_perdidos1 = count_perdidos1 + 1
        espacios[0], count_perdidos1, aux1 =
utils.verificar_desaparecidos(espacios[0], aux1, count_perdidos1)

    ### SEGUNDO ESPACIO
    if espacios[1] == 1:
        count_detectados2 = count_detectados2 + 1
        if count_detectados2 > min_detectados:
            espacios[1] = 1
            aux2 = 1
        else:
            espacios[1] = 0

    if espacios[1] == 0 and aux2 == 1:
```

```
                    count_perdidos2 = count_perdidos2 + 1
                    espacios[1], count_perdidos2, aux2 =
utils.verificar_desaparecidos(espacios[1], aux2, count_perdidos2)

                ### TERCER ESPACIO
                if espacios[2] == 1:
                    count_detectados3 = count_detectados3 + 1
                    if count_detectados3 > min_detectados:
                        espacios[2] = 1
                        aux3 = 1
                    else:
                        espacios[2] = 0

                if espacios[2] == 0 and aux3 == 1:
                    count_perdidos3 = count_perdidos3 + 1
                    espacios[2], count_perdidos3, aux3 =
utils.verificar_desaparecidos(espacios[2], aux3, count_perdidos3)

                ### CUARTO ESPACIO
                if espacios[3] == 1:
                    count_detectados4 = count_detectados4 + 1
                    if count_detectados4 > min_detectados:
                        espacios[3] = 1
                        aux4 = 1
                    else:
                        espacios[3] = 0

                if espacios[3] == 0 and aux4 == 1:
                    count_perdidos4 = count_perdidos4 + 1
                    espacios[3], count_perdidos4, aux4 =
utils.verificar_desaparecidos(espacios[3], aux4, count_perdidos4)

                ### QUINTO ESPACIO
                if espacios[4] == 1:
                    count_detectados5 = count_detectados5 + 1
                    if count_detectados5 > min_detectados:
                        espacios[4] = 1
```

```
                                        aux5 = 1
                                else:
                                        espacios[4] = 0

                        if espacios[4] == 0 and aux5 == 1:
                                count_perdidos5 = count_perdidos5 + 1
                                espacios[4], count_perdidos5, aux5 =
utils.verificar_desaparecidos(espacios[4], aux5, count_perdidos5)

                                # Se dibujan los espacios ocupados (rojo), desocupados(verde),
                                # tanto en el video como en la imagen (mapa)
                                utils.dibujar_espacios(frame, espacios)
                                utils.dibujar_espacios_imagen(image, espacios)

                                # Se cuenta cuantos están en 0, y por ende están disponibles
                                disponibles = espacios.count(0)

                        else:
                                # Si no se dejecta ningún centroide entonces todos espacios aparecen
                                # como desocupados
                                utils.dibujar_espacios_imagen(image, [0, 0, 0, 0, 0])
                                utils.dibujar_espacios(frame, [0, 0, 0, 0, 0])
                                disponibles=5

                        # Se muestra la información de disponibles en la imagen(mapa)
                        cv2.putText(image, "{}".format(disponibles), (217, 400),
cv2.FONT_HERSHEY_SIMPLEX, 5, (0, 0, 0), 3)

                        cv2.imshow("Transmision", frame)
                        cv2.imshow("Mapa", image)
                        k = cv2.waitKey(1) & 0xFF
                        if k == 27:
                                break

                        with lock:
                                outputFrame = frame.copy()
                                outputImage = image.copy()
```

```
def generate():
	global outputFrame, lock
	while True:
		with lock:
			if outputFrame is None:
				continue
			(flag, encodedImage) = cv2.imencode(".jpg", outputFrame)
			if not flag:
				continue
		yield(b'--frame\r\n' b'Content-Type: image/jpeg\r\n\r\n' +
			bytearray(encodedImage) + b'\r\n')

def generateInterface():
	global outputImage, lock
	while True:
		with lock:
			if outputImage is None:
				continue
			(flag, encodedImage) = cv2.imencode(".jpg", outputImage)
			if not flag:
				continue
		yield(b'--frame\r\n' b'Content-Type: image/jpeg\r\n\r\n' +
			bytearray(encodedImage) + b'\r\n')

@app.route("/")
def index():
	# return the rendered template
	return render_template("index.html")

@app.route("/video")
def video():
	# return the rendered template
	return render_template("video.html")

@app.route("/video_feed")
def video_feed():
```

```
        return Response(generate(),
                mimetype = "multipart/x-mixed-replace; boundary=frame")

@app.route("/mapa")
def mapa():
        # return the rendered template
        return render_template("mapa.html")

@app.route("/video_interface")
def video_interface():
        return Response(generateInterface(),
                mimetype = "multipart/x-mixed-replace; boundary=frame")

if __name__ == '__main__':
        ip = "0.0.0.0"
        port = "80"

        t = threading.Thread(target=video_vivo, args=(
                32,))
        t.deamon = True
        t.start()

        app.run(host=ip, port=port, debug=True,
                threaded=True, use_reloader=False)

cap.release()
cv2.destroyAllWindows()



'''
Føx
'''
```

Funciones

```
import cv2
import numpy as np

def verificar_desaparecidos(espacio, aux, count_perdidos, max_desaparecidos=60):
        # Verifica si el auto está realmente desaparecido

        if count_perdidos > max_desaparecidos:
                espacio = 0
                count_perdidos = 0
                aux = 0
        else:
                espacio = 1

        return espacio, count_perdidos, aux

def encontrar_centroides(frame, net, ln):
        # Con ayuda de YOLOv3 se detectan los vehículos y se extraemos sus respectivos
        #centroides.

        centroids_output = []

        (H, W) = frame.shape[:2]
        blob = cv2.dnn.blobFromImage(frame, 1/ 255.0, (416, 416),
                swapRB=True, crop=False)
        net.setInput(blob)

        layerOutputs = net.forward(ln)

        boxes = []
        confidences = []
        centroids = []
        classIDs = []

        for output in layerOutputs:
```

```
                for detection in output:

                        scores = detection[5:]
                        classID = np.argmax(scores)
                        confidence = scores[classID]

                        if confidence > 0.5 and (classID == 2 or classID == 3 or classID == 5
or classID == 7):
                                box = detection[0:4] * np.array([W, H, W, H])
                                (centerX, centerY, width, height) = box.astype("int")

                                x = int(centerX - (width / 2))
                                y = int(centerY - (height / 2))

                                boxes.append([x, y, int(width), int(height)])
                                confidences.append(float(confidence))
                                centroids.append((centerX, centerY))
                                classIDs.append(classID)
        idxs = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.3)

        if len (idxs) > 0:
                for i in idxs.flatten():

                        color = (0, 255, 0)
                        # Se retornarán los centroidese que estén entre 170 y 300 en el eje 'y'
                        if  170 <= centroids[i][1] <= 300:
                                color = (0, 0, 255)
                                centroids_output.append(centroids[i])
                        cv2.circle(frame, (centroids[i]), 10, color, -1)
        return centroids_output

def determinar_espacios_ocupados(centroides):
        # Determina si cada uno de los 5 espacios están ocupados (1)
        # o desocupados, y se devuelve un array de 5 elementos
        # que pueden tener 0 o 1.
        espacios = [0, 0, 0, 0, 0]
        for centroide in centroides:
```

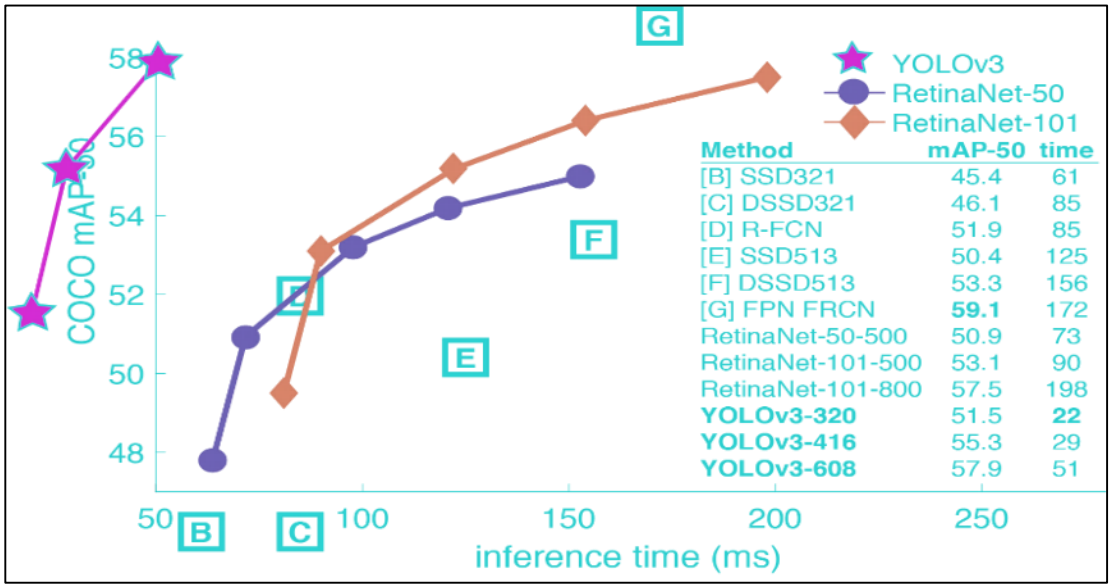
```
                if 0 < centroide[0] < 80:
                        espacios[0] = 1
                if 110 < centroide[0] < 230:
                        espacios[1] = 1
                if 270 < centroide[0] < 390:
                        espacios[2] = 1
                if 450 < centroide[0] < 560:
                        espacios[3] = 1
                if 590 < centroide[0] < 690:
                        espacios[4] = 1

        return espacios

def dibujar_espacios(frame, espacios):

        # Puntos donde se encuentran los 5 espacios del estacionamiento
        area_pts1 = np.array([[0, 140],  [60, 120], [105, 250],[12, 298]])
        area_pts2 = np.array([[78, 170],  [166, 130], [270, 270],[142, 327]])
        area_pts3 = np.array([[200, 170],  [305, 120], [460, 275],[345, 355]])
        area_pts4 = np.array([[370, 180],  [470, 120], [637, 270],[565, 355]])
        area_pts5 = np.array([[532, 170],  [620, 100], [720, 180],[715, 330]])

        # Color (verde o rojo) que toma cada espacio si este está desocupado
        # y ocupado (0, 1)
        color_area_pts1 = (0, 255, 0) if (espacios[0] == 0) else (0, 0, 255)
        color_area_pts2 = (0, 255, 0) if (espacios[1] == 0) else (0, 0, 255)
        color_area_pts3 = (0, 255, 0) if (espacios[2] == 0) else (0, 0, 255)
        color_area_pts4 = (0, 255, 0) if (espacios[3] == 0) else (0, 0, 255)
        color_area_pts5 = (0, 255, 0) if (espacios[4] == 0) else (0, 0, 255)

        # Se dibuja cada espacio dependiendo de si está ocupado o desocupado
        cv2.drawContours(frame, [area_pts1], -1, color_area_pts1, 2)
        cv2.drawContours(frame, [area_pts2], -1, color_area_pts2, 2)
        cv2.drawContours(frame, [area_pts3], -1, color_area_pts3, 2)
        cv2.drawContours(frame, [area_pts4], -1, color_area_pts4, 2)
        cv2.drawContours(frame, [area_pts5], -1, color_area_pts5, 2)
```

```
def dibujar_espacios_imagen(image, espacios):

    # Puntos donde se encuentran los 5 espacios del estacionamiento
    area_pts1 = np.array([[250, 116],  [302, 168], [250, 221],[198, 168]])
    area_pts2 = np.array([[394, 116],  [446, 168], [394, 221],[342, 168]])
    area_pts3 = np.array([[538, 116],  [590, 168], [538, 221],[486, 168]])
    area_pts4 = np.array([[682, 116],  [734, 168], [682, 221],[630, 168]])
    area_pts5 = np.array([[826, 116],  [878, 168], [826, 221],[774, 168]])

    # Color (verde o rojo) que toma cada espacio si este está desocupado y ocupado (0, 1)
    color_area_pts1 = (0, 255, 0) if (espacios[0] == 0) else (0, 0, 255)
    color_area_pts2 = (0, 255, 0) if (espacios[1] == 0) else (0, 0, 255)
    color_area_pts3 = (0, 255, 0) if (espacios[2] == 0) else (0, 0, 255)
    color_area_pts4 = (0, 255, 0) if (espacios[3] == 0) else (0, 0, 255)
    color_area_pts5 = (0, 255, 0) if (espacios[4] == 0) else (0, 0, 255)

    # Se dibuja cada espacio dependiendo de si está ocupado o desocupado
    cv2.drawContours(image, [area_pts1], -1, color_area_pts1, -1)
    cv2.drawContours(image, [area_pts2], -1, color_area_pts2, -1)
    cv2.drawContours(image, [area_pts3], -1, color_area_pts3, -1)
    cv2.drawContours(image, [area_pts4], -1, color_area_pts4, -1)
    cv2.drawContours(image, [area_pts5], -1, color_area_pts5, -1)


'''
Føx
'''
```

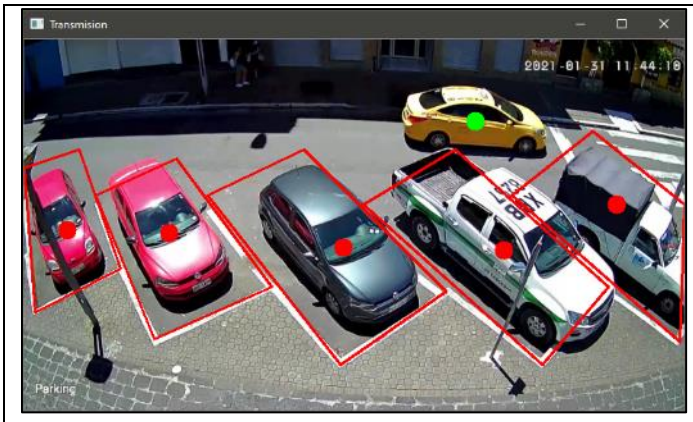
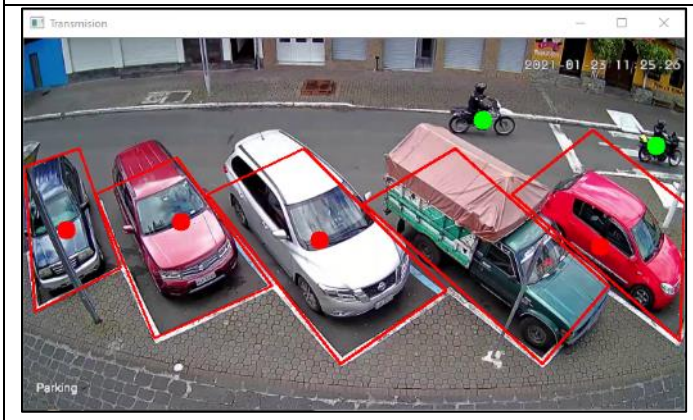
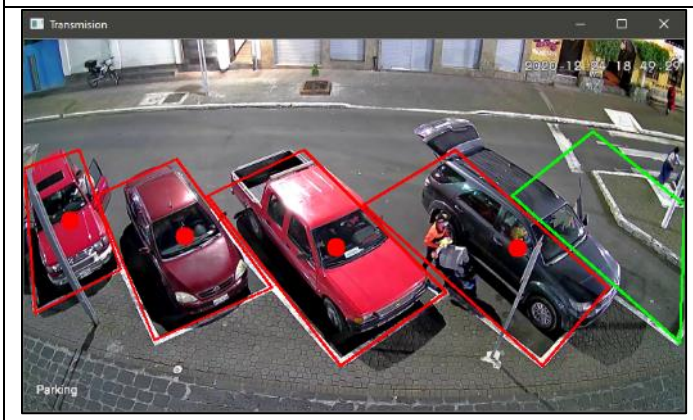
ANEXO C: Tiempos de procesamiento

Adquisición de tiempos de procesamiento

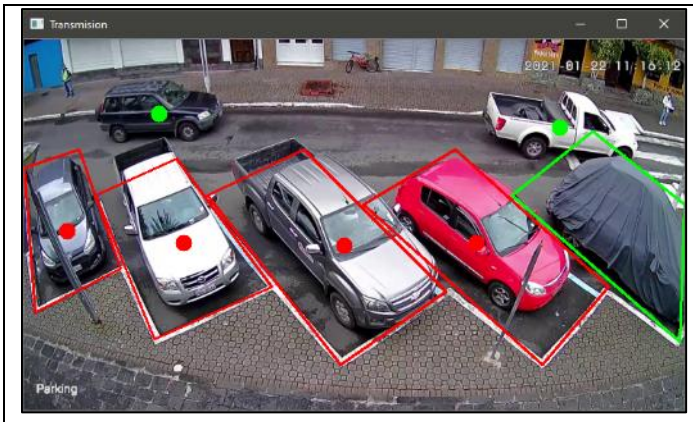
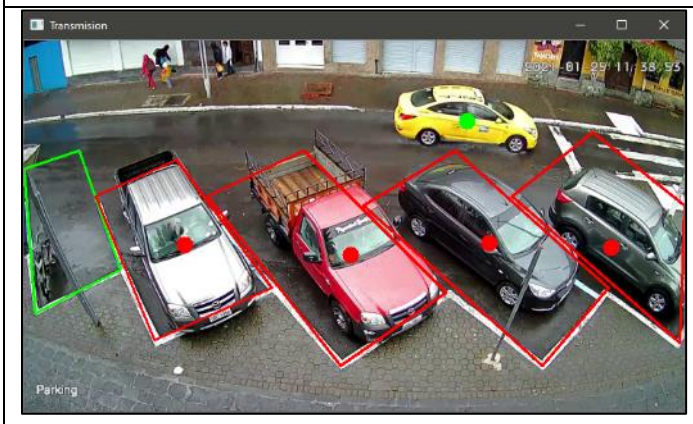
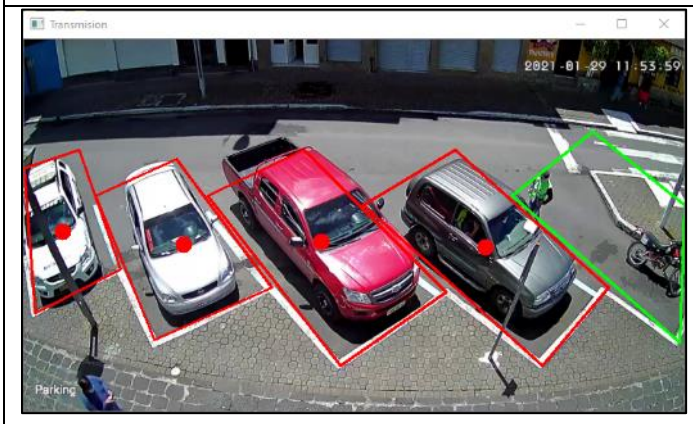
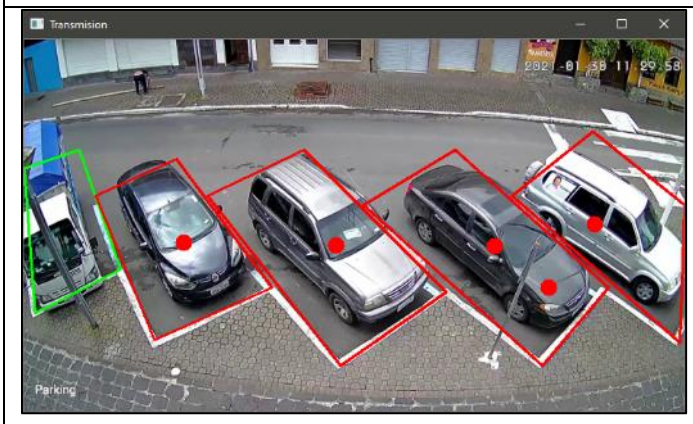
Rendimiento YOLOv3 en base las pruebas de los desarrolladores



ANEXO D: Detección en distintas condiciones climáticas

	**Detección:** Día soleado
	**Detección:** Día nublado
	**Detección:** Día lluvioso
	**Detección:** Noche

ANEXO E: Falsos Negativos

	**Día 2:** Vehículo no detectado
	**Día 5:** Vehículo no detectado
	**Día 8:** Vehículo no detectado
	**Día 9:** Vehículo no detectado

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

DIRECCIÓN DE BIBLIOTECAS Y RECURSOS DEL APRENDIZAJE

UNIDAD DE PROCESOS TÉCNICOS Y ANÁLISIS BIBLIOGRÁFICO Y DOCUMENTAL

REVISIÓN DE NORMAS TÉCNICAS, RESUMEN Y BIBLIOGRAFÍA

Fecha de entrega: *24 / 11 / 2021*

INFORMACIÓN DEL AUTOR/A (S)
Nombres – Apellidos: *Jhonatan Daniel Muñoz Romero*
INFORMACIÓN INSTITUCIONAL
Facultad: *Informática y Electrónica*
Carrera: *Electrónica y Automatización*
Título a optar: *Ingeniero en Electrónica Control y Redes Industriales*
f. Analista de Biblioteca responsable: *Lic. Luis Caminos Vargas Mgs.*

LUIS ALBERTO CAMINOS VARGAS

Firmado digitalmente por LUIS ALBERTO CAMINOS VARGAS
Nombre de reconocimiento (DN): c=EC, l=RIOBAMBA, serialNumber=0602766974, cn=LUIS ALBERTO CAMINOS VARGAS
Fecha: 2021.11.24 08:42:00 -05'00'

0907-DBRAI-UTP-2021