UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y REDES DE COMUNICACIÓN

TRABAJO DE GRADO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y REDES DE COMUNICACIÓN

TEMA:

PROTOTIPO DE DETECCION DE APARCAMIENTOS LIBRES MEDIANTE VISION ARTIFICIAL EN UN PARQUEADERO DE LA UNIVERSIDAD TECNICA DEL NORTE

AUTOR: CRISTIAN ANDRES ERAZO ESTRADA

DIRECTOR: Msc SANDRA KARINA NARVAEZ PUPIALES

Ibarra- Ecuador

UNIVERSIDAD TÈCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÈCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

La UNIVERSIDAD TÉCNICA DEL NORTE dentro del proyecto Repositorio Digital Institucional, determinó la necesidad de disponer de textos completos en formato digital con la finalidad de apoyar los procesos de investigación, docencia y extensión de la Universidad. Por medio del presente documento dejo sentada mi voluntad de participar en este proyecto, para lo cual pongo a disposición la siguiente información.

DATOS DEL CONTACTO	
Cédula de identidad	0401541297
Apellidos y Nombres	Cristian Andrés Erazo Estrada
Dirección	Barrio Olivo, calle Aníbal Guzmán Lara y Dr. Cristóbal Tobar Subia
E-mail	caerazoe@utn.edu.ec
Teléfono fijo	0999484259
Teléfono móvil	0999484259
DATOS DE LA OBRA	
Título	PROTOTIPO DE DETECCION DE APARCAMIENTOS LIBRES MEDIANTE VISION ARTIFICIAL EN UN PARQUEADERO DE LA UNIVERSIDAD TECNICA DEL NORTE
Autor	Cristian Andrés Erazo Estrada
Fecha	Marzo 2019
Programa	Pregrado
Título	Ingeniero en Electrónica y Redes de Comunicación
Director	MSc Sandra Karina Narváez

iii

2. AUTORIZACIÓN DE USO A FAVOR DE LA UNIVERSIDAD.

Yo, Erazo Estrada Cristian Andres, con cedula de identidad Nro. 0401541297, en calidad de autor y titular de los derechos patrimoniales de la obra o trabajo de grado descrito anteriormente, hago entrega del ejemplar respectivo en forma digital y autorizo a la Universidad Técnica del Norte, la publicación de la obra en el Repositorio Digital Institucional y uso del archivo digital en la biblioteca de la universidad con fines académicos, para ampliar la disponibilidad de material y como apoyo a la educación, investigación y extensión, en concordancia con la ley de Educación Superior Articulo 144.

3. CONSTANCIAS.

Yo, ERAZO ESTRADA CRISTIAN ANDRES declaro bajo juramento que el trabajo aquí escrito es de mi autoría; y que este no ha sido previamente presentado para ningún grado o calificación profesional y que he consultado las referencias bibliográficas que se presentan en este documento.

Ibarra, marzo del 2019

EL AUTOR

Cristian Andres Erazo Estrada
040154129-7

iv

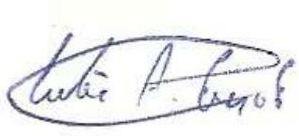
UNIVERSIDAD TÈCNICA DEL NORTE
FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

DECLARACIÓN

Yo, Cristian Andrés Erazo Estrada, declaro bajo juramento que el trabajo aquí escrito es de mi autoría; y que este no ha sido previamente presentado para ningún grado o calificación profesional y que he consultado las referencias bibliográficas que se presentan en este documento. A través de la presente declaración cedo los derechos los derechos de propiedad intelectual correspondiente a este trabajo, a la Universidad Técnica Del Norte, según lo establecido por las leyes de propiedad intelectual, reglamentos y normatividad vigente de la Universidad Técnica Del Norte.

Ibarra, marzo 2019

EL AUTOR



Cristian Andrés Erazo Estrada
040154129-7

v

UNIVERSIDAD TÈCNICA DEL NORTE
FACULTAD DE INGENIERIA EN CIENCIAS APLICADAS

CERTIFICACIÓN DEL DIRECTOR

El Sr. Cristian Andres Erazo Estrada, portador de la cedula de identidad número: 040154129-7, ha trabajado en el desarrollo del proyecto de grado **"PROTOTIPO DE DETECCION DE APARCAMIENTOS LIBRES MEDIANTE VISION ARTIFICIAL EN UN PARQUEADERO DE LA UNIVERSIDAD TECNICA DEL NORTE"**, previo a la obtención del Título en Ingeniería Electrónica y Redes de Comunicación, realizado con interés profesional y responsabilidad, que certifico en honor a la verdad.

Ibarra, 12 de marzo del 2019

.....
MSc. Sandra Narváez
Director de Trabajo de Grado

UNIVERSIDAD TÈCNICA DEL NORTE
FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

AGRADECIMIENTO

Agradezco a Dios por bendecirnos la vida, por guiarme a lo largo de nuestra existencia, ser el apoyo y fortaleza en aquellos momentos de dificultad y de debilidad.

Gracias a mi Madre y Abuelitos: Carmen, Rosa y Alfonso, por ser los principales promotores de mis sueños, por confiar y creer en mis expectativas, por los consejos, valores y principios que me han inculcado.

Agradezco a los docentes de la carrera de Ingeniería en Electrónica y Redes de Comunicación de la Universidad Técnica del Norte, por haber compartido sus conocimientos a lo largo de la preparación de mi profesión, de manera especial, a la Magister Sandra Narváez tutora de mi proyecto de investigación quien me ha guiado con su paciencia, y su rectitud como docente.

Cristian Erazo

UNIVERSIDAD TÈCNICA DEL NORTE
FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

DEDICATORIA

El presente trabajo investigativo lo dedico principalmente a Dios, por ser el inspirador y darme fuerza para continuar en este proceso de obtener uno de los anhelos más deseados.

A mi Madre y Abuelitos, por su amor, trabajo y sacrificio en todos estos años, gracias a ustedes he logrado llegar hasta aquí y convertirme en lo que soy.

A mis amigos y familiares por estar siempre presentes, acompañándome y por el apoyo moral, que me brindaron a lo largo de esta etapa de mi vida.

A todas las personas que me han apoyado y han hecho que el trabajo se realice con éxito en especial a aquellos que me abrieron las puertas y compartieron sus conocimientos.

Cristian Erazo

RESUMEN

El presente Proyecto consiste de un Sistema de Monitoreo de Disponibilidad de plazas de un estacionamiento en la Universidad Técnica del Norte, está basado en el uso de técnicas de visión artificial para lo cual se usarán cámaras para recolectar información sobre plazas libres de estacionamiento.

Para el diseño se realizó un estudio de la situación actual en cuanto al uso y distribución de plazas de estacionar en la Universidad Técnica del Norte, así como también analizar las ventajas y falencias que este presenta.

Para determinar las mejores herramientas tanto de hardware y de software adecuadas para el proceso y envío de información recolectada por el prototipo, se realizó un estudio de los dispositivos electrónicos existentes y se elegirá el que mejor se adapte a las necesidades del proyecto. El sistema cuenta con una cámara ubicada estratégicamente, la cual envía la información hacia una computadora de placa simple que permita el procesamiento de imágenes mediante visión artificial, estos datos serán subidos a la nube para que puedan ser visualizados por los usuarios mediante una página web y también en una pantalla ubicada en una puerta de un acceso vehicular del campus.

Se realizó pruebas de funcionamiento en un parqueadero de la Universidad Técnica del Norte en el cual se determinó la funcionalidad del sistema, la prueba se realizó durante el periodo diurno y el número de estacionamientos estuvo por el alcance de visión de la cámara.

Se realizó un análisis de la relación Beneficio/Costo del prototipo que además contará con el costo final aplicando el prototipo a todos los parqueaderos de la Universidad Técnica Del Norte, con el fin de determinar la viabilidad del proyecto para que a futuro pueda ser implementado en su totalidad.

ASTRACT

The present Project consists of a System of Monitoring Availability of parking spaces in the Technical University of the North, is based on the use of artificial vision techniques for which cameras will be used to collect information about free parking spaces.

For the design, a study was made of the current situation regarding the use and distribution of parking spaces at the Technical University of the North, as well as analyzing the advantages and shortcomings that this presents.

To determine the best hardware and software tools suitable for the process and sending information collected by the prototype, a study of the existing electronic devices was made and the one that best suits the needs of the project will be chosen. The system has a strategically located camera, which sends the information to a single-board computer that allows the processing of images through artificial vision, these data will be uploaded to the cloud so that they can be viewed by users through a web page and also on a screen located in a door of a vehicular access of the campus.

Functional tests were performed in a parking lot of the Technical University of the North in which the functionality of the system was determined, the test was conducted during the daytime period and the number of parking lots was by the scope of vision of the camera.

A cost analysis of the prototype was carried out, which will also have the final cost by applying the prototype to all the parking lots of the Universidad Técnica Del Norte, in order to determine the viability of the project so that in the future it can be implemented in its entirety.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Capítulo I

1. Antecedentes

1.1. Tema

Prototipo de detección de aparcamientos libres mediante visión artificial en un parqueadero de La Universidad Técnica del Norte

1.2. Problema

El constante aumento de vehículos en las ciudades ha hecho que el paisaje urbano se vea invadido de automóviles, ya que estos desempeñan una función imprescindible en la vida moderna, uno de los problemas que genera este incremento es el de la generación de tráfico y esto va de la mano con la gestión en los parqueaderos de las ciudades, en pocas palabras, el estacionamiento normal es ineficiente, frustrante y lleva mucho tiempo. Actualmente la Universidad Técnica del Norte no cuenta con un sistema automatizado de parqueo vehicular, adicional a esto se suma el incremento de vehículos que ingresan a la institución, lo cual ha generado que exista dificultad a la hora de buscar un lugar libre.

La mayor incidencia de vehículos en la institución se da en los horarios de ingreso en la mañana y en la tarde puesto que a estas horas aquellas personas que cuenten con vehículos ingresan a realizar sus respectivas labores, el problema es que los estacionamientos simplemente se han ignorado ya que no se han implementado tecnologías que permitan mejorar la eficiencia de estos, a partir de ello se genera la pérdida de tiempo en los conductores cuando buscan un lugar libre. También se generan situaciones peligrosas ya que mientras el conductor se enfoca en la búsqueda de un lugar libre se distrae de lo que pasa a su alrededor lo que aumenta la probabilidad de que cause un accidente.

En base a lo anteriormente dicho se propone diseñar e implementar un prototipo de detección de aparcamientos que permita determinar y asignar puestos libres mediante la detección de objetos utilizando la visión artificial. El proyecto se lo realizará mediante el uso de software y plataformas libres que permitirá reducir el tiempo y la congestión vehicular a la hora de buscar un espacio libre.

1.3. Objetivos

1.3.1. Objetivo General

Diseñar un prototipo de detección de aparcamientos libres mediante visión artificial en la Universidad Técnica del Norte

1.3.2. Objetivos Específicos

- Realizar una revisión sistemática de literatura para determinar las bases teóricas acerca de sistemas de parqueo y detección de objetos mediante monitoreo visual.
- Realizar un estudio de la situación actual del sistema de parqueo en la Universidad Técnica del Norte, así como también realizar encuestas a los usuarios que ayuden a determinar el diseño óptimo del prototipo.
- Diseñar el prototipo acorde con los requerimientos obtenidos en la parte del estudio.
- Analizar y determinar el hardware y el software idóneo en relación a los requerimientos establecidos para el diseño y desarrollo del prototipo.
- Realizar las pruebas de funcionamiento del prototipo para la depuración de errores.

- Analizar la relación beneficio costo que permita determinar la viabilidad del prototipo.

1.4. Alcance

Con la finalidad de obtener mayor información y comprensión del sistema en cuestión, se realizará un estudio tanto de contexto como de referencias mundiales en cuanto a los sistemas existentes desarrollados y que tengan el mismo fin, para obtener una perspectiva del diseño del sistema.

Se realizará un estudio de la situación actual en cuanto al uso y distribución de plazas de estacionar en la Universidad Técnica del Norte, así como también encuestas dirigidas hacia los usuarios que hagan uso de este, que permitan establecer los requerimientos para el diseño óptimo del prototipo.

Mediante la información recolectada en la parte del estudio, se realizará el diseño del prototipo que cumpla los requerimientos de los beneficiados, que en este caso serán los conductores que hagan uso del parqueadero.

Para determinar las mejores herramientas tanto de hardware y de software adecuadas para el proceso y envío de información recolectada por el prototipo, se realizará un estudio de los dispositivos electrónicos existentes y se elegirá el que mejor se adapte a las necesidades del proyecto. El sistema contará con una cámara ubicada estratégicamente, la cual enviará la información hacia una computadora de placa simple que permita el procesamiento de imágenes mediante visión artificial, mediante esta técnica se alimenta al computador con una gran cantidad de imágenes las cuales a través de varios algoritmos le permiten al computador

buscar patrones y detectar formas, en este caso permitirá determinar cuándo un lugar este libre u ocupado, estos datos serán subidos a la nube para que puedan ser visualizados por los usuarios mediante una página web y también en una pantalla ubicada en una puerta de un acceso vehicular del campus.

Se realizarán las pruebas de funcionamiento en un parqueadero de la Universidad Técnica del Norte en el cual se determinará la funcionalidad del sistema, las pruebas serán realizadas durante el periodo diurno y el número de estacionamientos será limitado por el alcance de visión de la cámara, luego se procederá a la depuración de errores para mejorar la eficiencia del prototipo.

Se realizará un análisis de la relación beneficio costo del prototipo que además contará con el costo final aplicando el prototipo a todos los parqueaderos de la Universidad Técnica del Norte, con el fin de determinar la viabilidad del proyecto para que a futuro pueda ser implementado en su totalidad.

1.5. Justificación

El desarrollo de nuevas tecnologías abre un mundo de posibilidades y mejoras en la calidad de vida de las personas debido a que se puede innovar en los procesos que están pasando a nuestro alrededor, mediante estas tecnologías se puede automatizar las actividades de las personas permitiendo así reducir energía y tiempo.

En el Ecuador los proyectos tecnológicos han logrado un gran respaldo por parte del Gobierno Nacional, ya que con el progreso de éstos se contribuye directamente al cumplimiento del Plan Nacional del Buen Vivir, específicamente se hace un aporte al objetivo 11, en su sección 11.3 cuyo principio es democratizar la prestación de servicios de

telecomunicaciones y el acceso universal a las TICs (Senplades, 2013). El cumplimiento de este objetivo no solo contribuye a una sociedad con mayor conocimiento tecnológico, sino que también asevera el correcto desarrollo de este proyecto al disponer de los recursos tecnológicos en las instituciones o sectores comerciales donde podría ser implementado.

En la actualidad una de las aplicaciones que más se está usando es la de la visión artificial, esta consiste en realizar el procesamiento de imágenes mediante cámaras, ya sea de cualquier situación o circunstancia, este tipo de tecnología ofrece variadas ventajas una de las más remarcadas es la de automatizar sistemas permitiendo así mejorar su rendimiento. Al aplicar este concepto junto con un sistema embebido se obtiene un sistema inteligente capaz de tomar decisiones por su propia cuenta el cual nos permite un monitoreo a tiempo real.

El objetivo del proyecto es reducir el tiempo de búsqueda de un espacio libre en la Universidad Técnica del Norte mediante el uso de plataformas libres, este tipo de plataformas permite que se incentiven y se fortalezcan los proyectos de innovación logrando así dejar la dependencia de las plataformas pagadas. Además, el desarrollo de este proyecto permitirá ser un sistema modelo que se podrá aplicar en parqueaderos de instituciones, centros comerciales y parques con el fin de que se reduzcan los tiempos y el tráfico vehicular en la ciudad.

Este proyecto contribuirá al desarrollo de las Smart Cities ya que es una de las soluciones más importantes, así como una de las más básicas requeridas en la implementación de una ciudad inteligente. El estacionamiento inteligente optimiza el espacio de estacionamiento de manera efectiva, lo que además conduce a la mejora de la eficiencia de las operaciones de estacionamiento.

Capítulo II

2. Marco Teórico

2.1. Parqueaderos Inteligentes

Una de las funciones de las Smart City es el desarrollo de urbanismos con edificios, estacionamientos o parqueaderos, zonas recreativas, entre otros con la categoría "Inteligente". Es decir que de alguna manera son capaces de realizar tareas de forma automática, para lo cual necesita sensores donde se capta la señal de las variables a medir y actuadores que realizan las funciones que se controlan del sistema. Las condiciones de los parqueaderos inteligentes son relativas, es decir, no siempre son las mismas, estas dependen en muchos casos de las características físicas donde se encuentran. Un parqueadero inteligente debe al menos realizar tareas de manera autónoma observando las condiciones del entorno y actuando en función de estas (Basu, 2014).

Un ejemplo de parqueadero inteligente son los robotizados también llamado parqueadero de multinivel, este es un sistema donde los vehículos son parqueados automáticamente a través de los movimientos de elevación del ascensor junto con los movimientos horizontales de la plataforma en los diferentes niveles del parqueadero. En la Figura 1 se muestra el interior de un parqueadero inteligente robotizado que realiza el parqueo de dos automóviles simultáneamente y totalmente automatizado. Este tipo de parqueadero utiliza planchas ubicadas en los puestos disponibles, el ascensor robótico retira la plancha y la lleva hasta la entrada donde se encuentra el automóvil en espera de parqueo, de forma automática coloca el automóvil sobre la plancha y procede a llevarlo hasta el puesto disponible donde lo ubica con la plancha correspondiente (Stadt Wien, 2015).

Figura 1 Vista interior de parqueadero inteligente robotizado.
Fuente: https://www.smartparkingsolution.com/about-us/

2.1.1. Parqueaderos inteligentes basados en visión artificial

Recientemente, como lo dice Fraife y Fernström muchos investigadores se han centrado en el uso de la visión artificial tales como: Takizawa (2004), Xu (2000), Zhu (2007), Funck (2004), Fabian (2008), Jung (2010), Banerjee (2011), entre otros. Este campo de estudio incluye métodos para adquirir, procesar y analizar imágenes. Utiliza computadoras para emular la visión artificial, incluyendo el aprendizaje y la capacidad de hacer inferencias y tomar medidas basadas en imágenes capturadas en la computadora. El objetivo de la computadora con la visión artificial es captar en imágenes la realidad del entorno y procesarlas con el fin de observar y actuar en función de los hechos y de patrones de conducta establecidos.

Por lo general, la técnica implica analizar uno o varios cuadros por segundo y luego enviar los datos a una base de datos central, después de lo cual, el usuario puede recuperar información sobre los cambios en el parqueadero. Takizawa (2004), desarrolló un sistema

que utilizando CCTV determina la presencia de un automóvil o vehículo en un puesto del parqueadero. El estudio de los pixeles que conforman una imagen que se usa para detectar la presencia de un vehículo en cada puesto del parqueo. En la Figura 2 se observa una descripción funcional de un sistema que basado en visión artificial determina si un puesto del parqueadero está disponible.

Figura 2 Detección de puestos libres en parqueadero basado en visión artificial.

Fuente: Autoría propia.

El procesamiento de las imágenes del video utiliza cierto número de píxeles en la escala de grises como el umbral para diferenciar los píxeles del vehículo y del lote desocupado. Las cámaras CCTV se instalan en los lugares estratégicos para detectar automáticamente los puestos desocupados. Sin embargo, estos métodos pueden detectar de manera incorrecta los vehículos estacionados. Funck (2004), propuso un sistema similar con cámaras CCTV instaladas en el parqueadero para detectar automáticamente los puestos vacíos en el parqueadero de automóviles. Dentro de sus resultados establece que estos métodos no siempre son precisos en los casos en que se requieren valores de ocupación.

Bong (2008), propuso un proyecto de investigación que fue desarrollado para adquirir información de la ocupación de los puestos en un parqueadero de automóviles utilizando un enfoque integrado de algoritmos de procesamiento de imágenes, que mediante filtros separan mediante un proceso de segmentación las zonas de interés y mediante un reconocimiento se determina o se extrae la información relevante para determinar si el puesto en revisión está disponible. La motivación para desarrollar este sistema proviene del hecho de que se requiere un costo mínimo porque se utilizan técnicas de procesamiento de imágenes en lugar de técnicas basadas en sensores. Este proyecto se llama Sistema de información de ocupación de estacionamiento (COINS).

2.1.2. Parqueaderos basados en redes de sensores

Estos tipos de sistemas han generado un mayor interés en los investigadores desde 2005. Son la técnica más popular en la última década, ya que las redes de sensores inalámbricos tienen varias ventajas, como flexibilidad, inteligencia, costo razonable, despliegue rápido y detección, por lo general consiste en una red de sensores (WSN), conformada por nodos con comunicación inalámbrica con protocolos especiales de estos sistemas. En la Figura 3 se presenta un diagrama que describe el funcionamiento de un sistema basado en sensores.

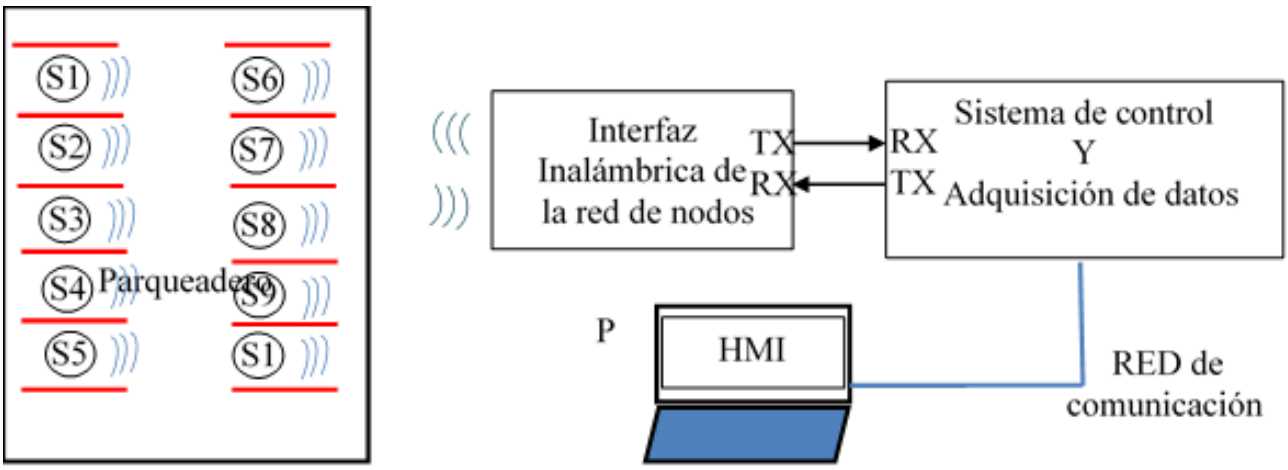


Figura 3 Diagrama funcional de parqueadero basado en IoT.

Fuente: Autoría propia.

Fraife y Fernström (2016) afirman que los siguientes investigadores han propuesto sistemas de parqueadero inteligentes basados en WSN Yan y Olariu (2008), Zheng (2006), Yan (2009), Miura (2006), O'flynn (2005), Kumar (2007), Lee (2008), Park (2008), Tubaishat (2009), Reve (2012), Sharma (2011), entre otros. Este tipo de sistema, generalmente utiliza sensores para monitorear las condiciones ambientales. Para parqueaderos se pueden usar sensores que permiten detectar la presencia de un vehículo en un puesto o en un punto de ingreso al parqueadero, tales como: sensores de ultrasonido, sensores de luz infrarrojo o ultravioleta, sensores magnéticos, sensores de proximidad inductivos o capacitivos, entre otros. Zheng (2006) desarrolló un sistema que utiliza productos de ballesta, que tienen un bajo costo unitario, este sistema permite que un automóvil detecte la entrada al estacionamiento y guía al conductor de manera eficiente hacia un puesto de parqueo vacío a través de letreros que se muestran al conductor.

Por otra parte, Kianpisheh (2012) presentó un nuevo sistema de parqueadero inteligente que utiliza un detector ultrasónico, para cada puesto individual, fijados en el techo. El sensor

transmite un sonido que rebota al chocar con un objeto sólido (automóvil o tierra) y se refleja de vuelta al sensor. Lee propuso el uso de una combinación de sensores magnéticos y ultrasónicos para la detección precisa y confiable de vehículos en un estacionamiento, y describieron una versión modificada del algoritmo para la detección de vehículos que usan magnetómetros.

2.1.3. Parqueaderos inteligentes basados en el modelo de agente

Los sistemas basados en agentes, según Fraife y Fernström han sido propuestos por Mateo, Yang y Longfei (2009); Khoukhi (2010); Li (2004); entre otros. Estos son un tipo de entidad capaz de observar el entorno a través de sensores y actuar sobre este basado en toma de decisiones. Tiene características útiles, como autonomía, reactividad y adaptabilidad. Básicamente, un sistema de agentes múltiples es un método de representar sistemas con entidades, autonomía e interacción con su entorno. Los sistemas de parqueadero inteligente basados en agentes son una forma de tecnología de agente móvil con un sistema de agente múltiple. Los agentes se modelan en capas jerárquicas que realizan negociaciones por las diferentes alternativas de soluciones posibles de un problema, por ejemplo: la búsqueda de puestos libres en un parqueadero. (Ver Figura 4)

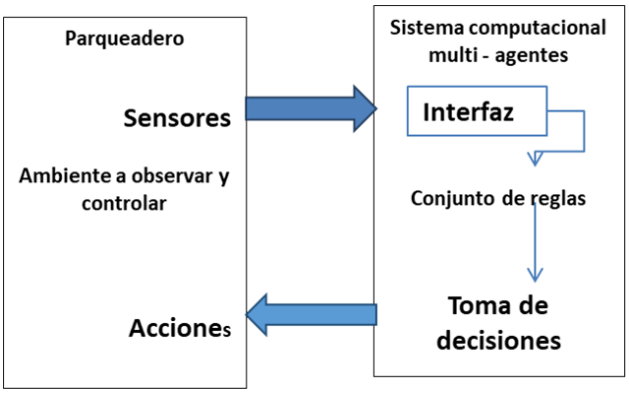


Figura 4 Parqueadero controlado mediante sistema computacional multi-agentes.

Fuente: Autoría.

El enfoque de distribución implica la construcción de un sistema de información de orientación de estacionamiento activo (APGIS). El APGIS está compuesto por automóviles, aparcamientos y un centro de servicio de información de estacionamiento (PISC), que tiene cuatro funciones: búsqueda de espacio de estacionamiento, negociación de precio de estacionamiento, reserva de espacio de estacionamiento y negociación y orientación de ruta de estacionamiento. Algunos investigadores han propuesto sistemas multi-agente con algoritmos de negociación para solucionar problemas propios de parqueaderos inteligentes.

2.1.4. Parqueaderos inteligentes basados en comunicación (V2I)

Como lo afirma Fraife y Fernström en su publicación, Stibor (2007), Holfelder (2004), Yousefi (2006), Bilstrup (2008), Panayappan, Geng (2012),entre otros propusieron utilizar el término (CVT) para referirse a un vehículo con tecnología de la transmisión inalámbrica de datos entre el vehículo y la infraestructura (V2I). Se refiere a los sistemas de comunicación vehicular, en los que los vehículos y las unidades de carretera son los nodos de comunicación, es decir, se comunican e intercambian información entre sí, como advertencias de seguridad o suministro de la información de congestión de tráfico e incluso para encontrar puestos de parqueo vacantes.

Básicamente, se considera que las redes vehiculares contienen dos tipos de nodos: vehículos y estaciones de carretera. Ambos se clasifican bajo el término dispositivos dedicados de comunicaciones de corto alcance (DSRC) que funciona en bandas de 5,9 GHz con un ancho de banda de 75 MHz y un alcance de aproximadamente 1000 m. Este es un método de comunicación bidireccional que incluye la comunicación vehículo a

infraestructura (V2I) e infraestructura a vehículo (I2V). En el sistema de "parqueadero inteligente", por lo general, la comunicación V2I incluye a los conductores que envían sus solicitudes de estacionamiento, proporcionando información del conductor y confirmando esa reserva en el sistema. La comunicación I2V implica que el DRPC envía resultados de asignación, indicaciones de manejo, detalles de pago, entre otra a los vehículos. Vale la pena mencionar que las redes celulares generalmente se aplican en soluciones V2I e I2V.

En la Figura 5 se representa un esquema de como es el funcionamiento este sistema.

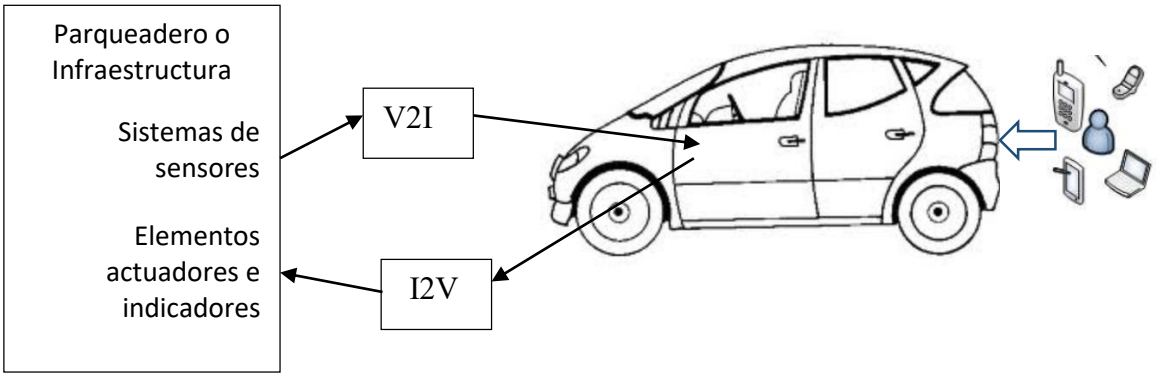


Figura 5 Diagrama funcional del parqueadero basado en V2I e I2V.

Fuente: Autoría propia.

Stibor (2007), propuso un novedoso sistema de estacionamiento llamado SPARK, que consta de cuatro partes: configuración del sistema, navegación de estacionamiento en tiempo real, protección antirrobo inteligente y diseminación de información de estacionamiento amigable. El sistema usa sensores de luz, y en el esquema SPARK propuesto, todo el parqueadero está bajo la vigilancia de los tres RSU de estacionamiento. Este sistema depende de las unidades de carretera desplegadas para retransmitir los mensajes de parqueaderos y un GPS para ubicar las posiciones de puestos libres.

2.1.5. Parqueaderos inteligentes basados en GPS

La tecnología de Sistemas de Posicionamiento Global (GPS) se usa para determinar y rastrear la ubicación precisa de un vehículo. En algunos sistemas se utilizan para entregar información sobre la ubicación y la disponibilidad de puestos libres en el parqueadero destino. Esta técnica fue propuesta en trabajos de parqueaderos inteligentes por Chon (2002), Hanif (2010). Chon presentó un sistema basado en la ubicación llamado NAPA donde el servidor en el sistema asocia edificios en la región con parqueaderos clasificados según la distancia desde el vehículo al edificio. (Ver Figura 6).

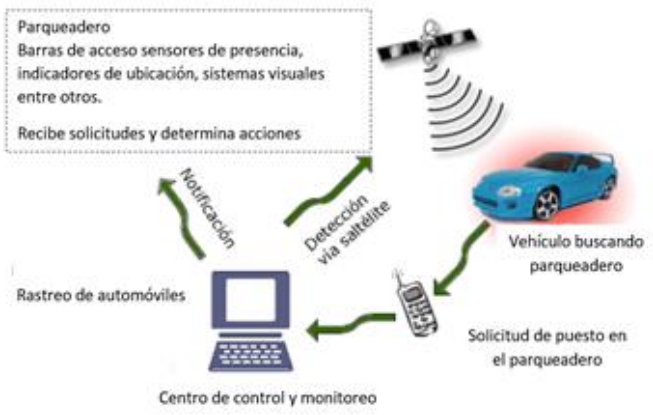


Figura 6 Diagrama funcional del parqueadero basado en GPS.
Fuente: Autoría propia.

Después de localizar el parqueadero disponible más cercano, el usuario envía un mensaje al servidor de NAPA que ha parqueado y el servidor actualiza la información. Cuando el usuario abandona el parqueadero, el servidor de NAPA puede cobrar automáticamente la tarifa de estacionamiento adecuada si es necesario. Hanif (2010) propuso un nuevo sistema de parqueadero inteligente que utiliza servicios de SMS. Este sistema es capaz de encontrar puestos libres en áreas específicas de para parqueo de automóviles. Un sistema de reserva de

parqueo les permite a los usuarios reservar sus lugares de parqueo por mensajes cortos de texto (SMS) usando el GPS. El SMS es procesado por un dispositivo de instrumentación de comunicación inalámbrica llamado unidad terminal remota RTU (por sus siglas en inglés, Remote Terminal Unit).

2.1.6. Parqueaderos inteligentes basados en tecnología RFID

La tecnología (RFID) como lo establece Fraife y Fernström (2016) en su publicación lo proponen en los siguientes investigadores Pala (2007), Gueaieb y Miah (2008), Jian (2008), Cervantes (2007), Hsieh (2008); Liu (2010), entre otros. En muchos trabajos, las soluciones de RFID de parqueaderos inteligentes hacen posible administrar fácilmente el permiso de parqueo, especialmente en las etapas de prototipo. En la Figura 7 está representado un diagrama de bloques de un sistema de control de un parqueadero controlado por RFID, que permite al usuario mediante un sistema de prepago solicitar un puesto disponible y el sistema lo ubica y lo asigna, también permite controlar el acceso.

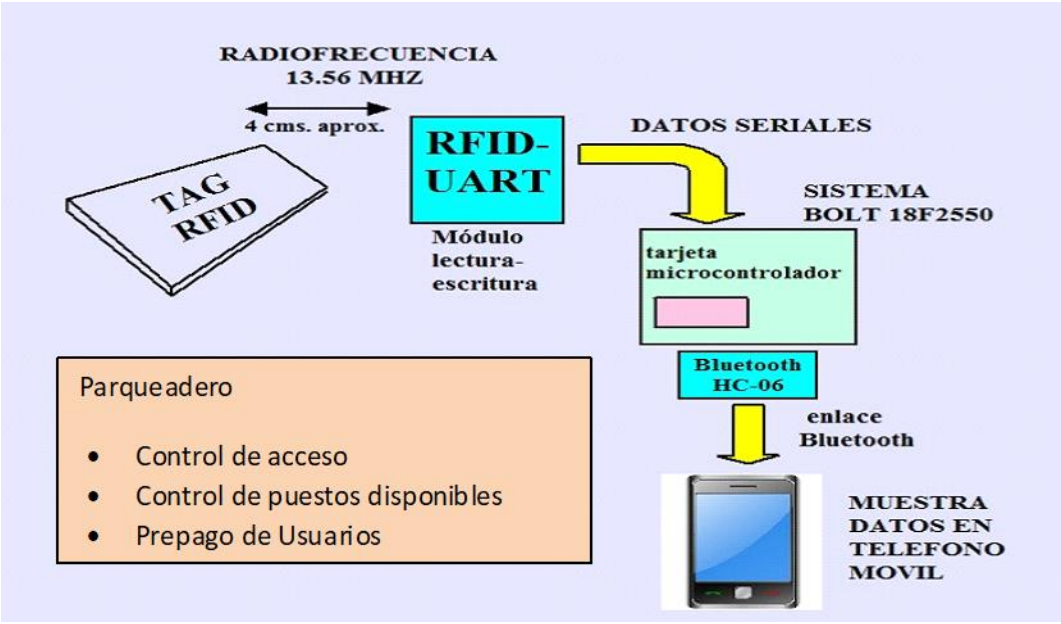


Figura 7 Sistema de control de parqueadero basado en RFID.
Fuente: Autoría.

El mecanismo principal de la tecnología RFID depende de un campo electromagnético para identificar y rastrear automáticamente las etiquetas adheridas a los objetos. Pala y Nihat (2007), utilizaron la tecnología RFID en la automatización de un parqueadero inteligente mediante un programa de software para controlar e informar los cambios en el estado en el puesto de parqueo, y para la operación de tareas tales como elegir el puesto disponible más cercano, y luego envía el informe al conductor. Mientras tanto, Jian (2008), propuso un Sistema RFID y una Plataforma Modular de Gestión de parqueo. La mayoría de los sistemas de administración de parqueaderos con RFID son modulares y pueden ser sustituidos por cualquier otro sistema o hardware similar.

2.2. IoT o Internet de las Cosas

El Internet de las Cosas (IoT), a veces denominado Internet de los objetos, cambiará todo debido a que ha causado un gran impacto en varias áreas como la educación, la comunicación, los negocios, la ciencia, el gobierno y la humanidad. Claramente, Internet es una de las creaciones más importantes y poderosas de toda la historia de la humanidad y ahora con el concepto de Internet de las cosas este se vuelve más favorable para tener una vida inteligente en todos los aspectos (Sayed & Mohamed, 2017).

Internet de las Cosas es una nueva tendencia tecnológica de acceso a internet, mediante esta técnica, los objetos se reconocen y obtienen un comportamiento de inteligencia al tomar o habilitar decisiones relacionadas, ya que pueden comunicar información sobre sí mismos. Estos objetos pueden acceder a la información que ha sido agregada por otras cosas,

o pueden agregarse a otros servicios, este concepto creará un nuevo tipo de aplicaciones que pueden incluir, vehículos inteligentes y casas inteligentes, para proporcionar muchos servicios como notificaciones, seguridad, ahorro de energía, automatización, comunicación, computadoras y entretenimiento. En la Figura 8 se presenta en esquema del parqueadero basado en IoT.

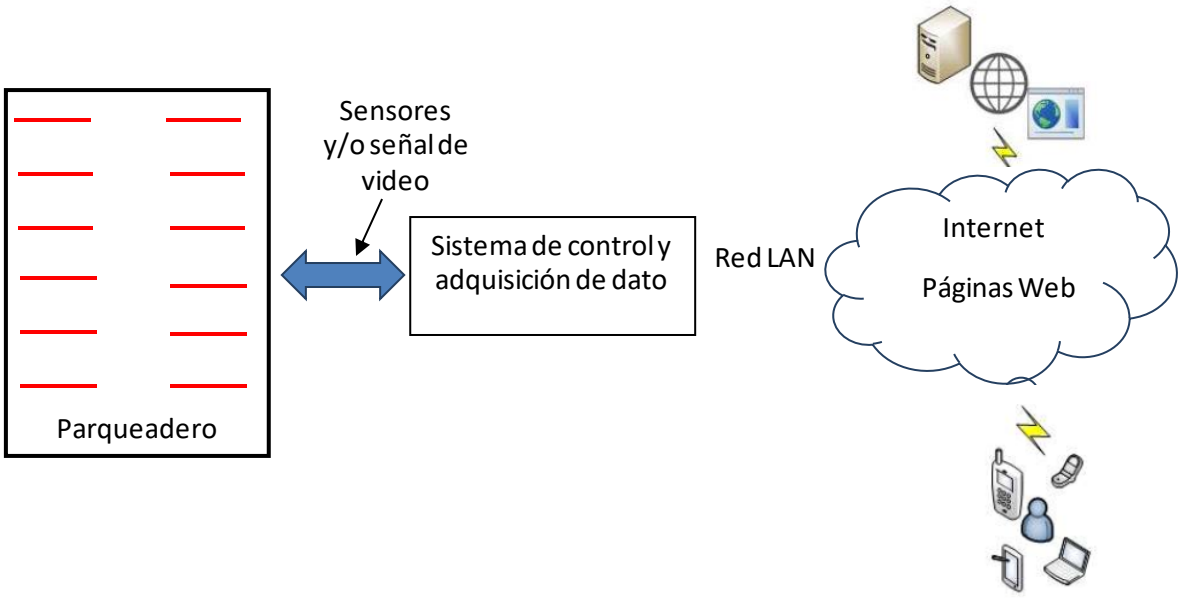


Figura 8. Esquema de parqueadero basado en IoT.
Fuente: Autoría propia.

Al desarrollar el Internet de las Cosas, se estará muy cerca de implementar entornos inteligentes para 2020 y en un futuro cercano, los servicios de almacenamiento y comunicación serán muy generalizados y distribuidos: personas, máquinas, objetos inteligentes, espacio circundante y plataformas conectadas con sensores inalámbricos / cableados, dispositivos M2M, etiquetas RFID crearán recursos altamente descentralizados interconectados por una red dinámica de redes (Sayed & Mohamed, 2017).

2.2.1 Protocolos y Estandarización en IoT

El éxito de IoT depende de la estandarización, que proporciona interoperabilidad, compatibilidad, confiabilidad y operaciones efectivas a escala global, hoy en día, más de 60 empresas de tecnología líder, en comunicaciones y energía, trabajan con estándares, como IETF, IEEE y ITU para especificar nuevas tecnologías basadas en IP para Internet de las Cosas (Sayed & Mohamed, 2017).

El diseño de los estándares de IoT debe considerar el uso eficiente de la energía y la capacidad de la red, así como respetar otras restricciones, como las bandas de frecuencia y los niveles de potencia para las comunicaciones de radiofrecuencia. A medida que IoT evoluciona, puede ser necesario revisar tales restricciones e investigar formas de garantizar la capacidad suficiente para la expansión, por ejemplo, en el caso de una asignación de espectro de radio adicional cuando esté disponible (Sayed & Mohamed, 2017).

IEEE Standards Association (IEEE-SA) desarrolla una serie de estándares relacionados con la necesidad del entorno para un IoT. El enfoque principal de las actividades de estandarización de IEEE está en las capas física y MAC. El IEEE proporciona una base temprana para la IoT con el estándar IEEE802.15.4 para radios de corto alcance de baja potencia, que normalmente funcionan en la banda industrial, científica y médica, además de utilizar la tecnología ZigBee. El IEEE-SA tiene más de 900 estándares activos y más de 500 estándares en desarrollo. En su investigación en IoT, ha identificado más de 140 estándares y proyectos existentes que son relevantes para IoT. El proyecto base relacionado con IoT es IEEE P2413, que actualmente está considerando la arquitectura de IoT (Sayed & Mohamed, 2017).

ETSI produce estándares aplicables a nivel mundial para las tecnologías de la información y las comunicaciones (TIC), incluidas las tecnologías fija, móvil, radio, convergente, de transmisión e Internet, y analiza un concepto similar en la etiqueta de comunicación de máquina a máquina (M2M). Estos estándares se consideran uno de los estándares básicos de IoT, ya que se asocian con la tecnología M2M, que es una de las técnicas básicas relacionadas con IoT. El Grupo de trabajo de ingeniería de Internet (IETF, por sus siglas en inglés) se ocupa de la evolución de la arquitectura de Internet y del buen funcionamiento de Internet, y es conocido como una comunidad grande, abierta a la comunidad internacional de diseñadores, operadores, proveedores e investigadores de redes (Sayed & Mohamed, 2017).

IETF proporciona su propia descripción de IoT, que proporciona una mejora más reconocible para admitir IPv6, con el 6LoWPAN. El Grupo de Trabajo 6TiSCH se está formando en el IETF para abordar el elemento de red de esa norma de unificación. Basado en estándares abiertos, 6TiSCH proporcionará un conjunto completo de protocolos para la operación de enrutamiento distribuido y centralizado a través del IEEE802.15.4e TSCH MAC. El Sector de Normalización de las Telecomunicaciones de la UIT (UIT-T) está considerado como la primera organización del desarrollo de normas y la coordinación de la Internet de las Cosas. Se ajustan a los estándares para beneficiarse de la capacidad integrada de procesamiento de información y de los productos industriales con capacidades inteligentes. Además de realizar desarrollos en identidades electrónicas que se pueden consultar de forma remota, o estar equipados con sensores para detectar cambios físicos a su alrededor (Sayed & Mohamed, 2017).

2.2.2 Aplicaciones de IoT

En la medición de consumo existe un porcentaje creciente de aplicaciones IoT donde se incluyen: automóviles conectados, entrenamiento físico, entre otros. Esos dispositivos IoT son creados para el consumo. Algunos ejemplos de aplicaciones de consumo incluyen automóviles conectados, entretenimiento, automatización del hogar, ropa inteligente, salud conectada y electrodomésticos como lavadoras, secadoras, aspiradoras robóticas, purificadores de aire, hornos, refrigeradores que utilizan tecnologías de comunicación inalámbrica para monitoreo remoto.

2.2.2.1 Smart Cities

Las ciudades inteligentes aún pueden verse como ciudades del futuro y de vida inteligente, y por la tasa de innovación de la creación de ciudades inteligentes en la actualidad, será muy factible ingresar a la tecnología de IoT en el desarrollo de ciudades estas requieren una planificación cuidadosa en cada etapa, con el apoyo del acuerdo de los gobiernos se podrá implementar la tecnología de Internet de las cosas en todos los aspectos.

Según la IoT, las ciudades pueden mejorarse en muchos niveles, mejorando la infraestructura, mejorando el transporte público, reduciendo la congestión del tráfico y manteniendo a los ciudadanos seguros, saludables y más comprometidos con la comunidad, como se muestra en la Figura 9, mediante la conexión de todos los sistemas en las ciudades, como el sistema de transporte, el sistema de atención médica, los sistemas de monitoreo del clima, etc., permiten brindar asistencia a las personas a través de Internet en todos los lugares para acceder a la base de datos de aeropuertos, ferrocarriles, seguimiento de transporte que opera bajo protocolos específicos, las ciudades (Sayed & Mohamed, 2017).

Figura 9. IoT Smart Cities.
Fuente: (Sayed & Mohamed, 2017).

2.2.2.2 Hogares y Edificios Inteligentes

Las tecnologías de Wi-Fi en la automatización del hogar se han utilizado principalmente debido a la naturaleza en red de los dispositivos electrónicos implementados en los que los dispositivos electrónicos como televisores, dispositivos móviles, etc. generalmente son compatibles con Wi-Fi. Wi-Fi ha comenzado a formar parte de la red IP doméstica y debido a la creciente tasa de adopción de dispositivos informáticos móviles como teléfonos inteligentes, tabletas, etc. Por ejemplo, una red para proporcionar servicios de transmisión en línea o red en los hogares puede proporcionar un medio para Control de la funcionalidad del dispositivo a través de la red.

Al mismo tiempo, los dispositivos móviles aseguran que los consumidores tengan acceso a un 'controlador' portátil para la electrónica conectada a la red, ambos tipos de dispositivos pueden utilizarse como puertas de enlace para aplicaciones IoT. Muchas empresas están considerando desarrollar plataformas que integren la automatización del edificio con el entretenimiento, el monitoreo de la atención médica, el monitoreo de la energía y el monitoreo inalámbrico de sensores en los entornos del hogar y del edificio. Por el concepto de Internet de las cosas, los hogares y edificios pueden operar muchos

dispositivos y objetos de manera inteligente, de las aplicaciones más interesantes de IoT en hogares y edificios inteligentes son iluminación inteligente, medio ambiente y medios inteligentes, control de aire y calefacción central, control de energía y seguridad como se muestra en la Figura 10 a continuación.

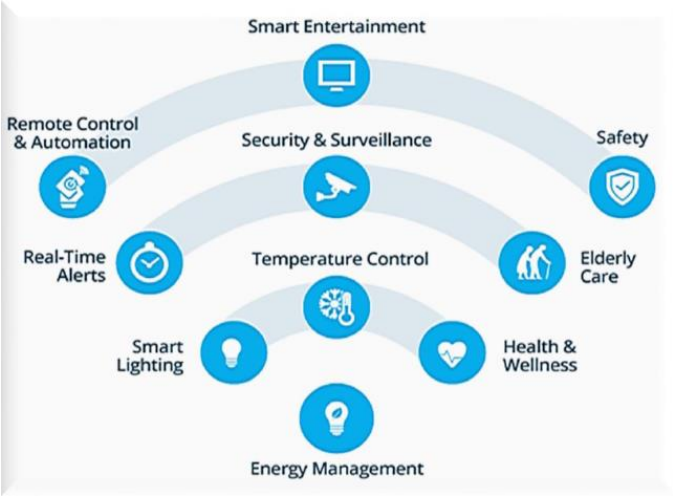


Figura 10. IoT Edificios y casas inteligentes.
Fuente: (Sayed & Mohamed, 2017).

2.2.2.3 Red de Energía Inteligente

Una red inteligente está relacionada con la información y el control y se desarrolló para tener una gestión inteligente de la energía. Una red inteligente que integra las tecnologías de la información y las comunicaciones (TIC) a la red eléctrica permitirá una comunicación bidireccional en tiempo real entre proveedores y consumidores, creando una interacción más dinámica en el flujo de energía, que ayudará a entregar la electricidad de manera más eficiente y sostenible.

Los elementos clave de las tecnologías de la información y las comunicaciones incluirán tecnologías de detección y monitoreo de flujos de energía; infraestructura de comunicaciones digitales para transmitir datos a través de la red; medidores inteligentes con pantalla en el hogar para informar el uso de energía; sistemas de coordinación, control y automatización para agregar y procesar diversos datos, y para crear una electricidad de respuesta altamente interactiva.

Muchas aplicaciones pueden ser manejadas debido a la Internet de las cosas para redes inteligentes, tales como industrial, energía solar, energía nuclear, vehículos, hospitales y control de energía de ciudades. La Figura 11 muestra que la aplicación más importante puede ser habilitada por Internet de las cosas como en el aspecto de la red inteligente.

Figura 11. IoT Redes de energía Inteligente.
Fuente: (Sayed & Mohamed, 2017).

2.2.2.4 Salud y Bienestar

El uso de las tecnologías de monitoreo de IoT requiere una atención especial que se requiere a los pacientes hospitalizados cuyo estado fisiológico debe monitorearse continuamente. Para la salud inteligente, los sensores se utilizan para recopilar información fisiológica completa y utilizan puertas de enlace y la nube para analizar y almacenar la información y luego enviar los datos analizados de forma inalámbrica a los cuidadores para un análisis y revisión adicionales, como se muestra en la Figura 12 a continuación.

Reemplaza el proceso de tener un profesional de la salud a intervalos regulares para verificar los signos vitales del paciente, en lugar de proporcionar un flujo continuo de información automatizado. De esta manera, mejora simultáneamente la calidad de la atención mediante la atención constante y reduce el costo de la atención al reducir el costo de las formas tradicionales de atención, además de la recopilación y el análisis de datos.

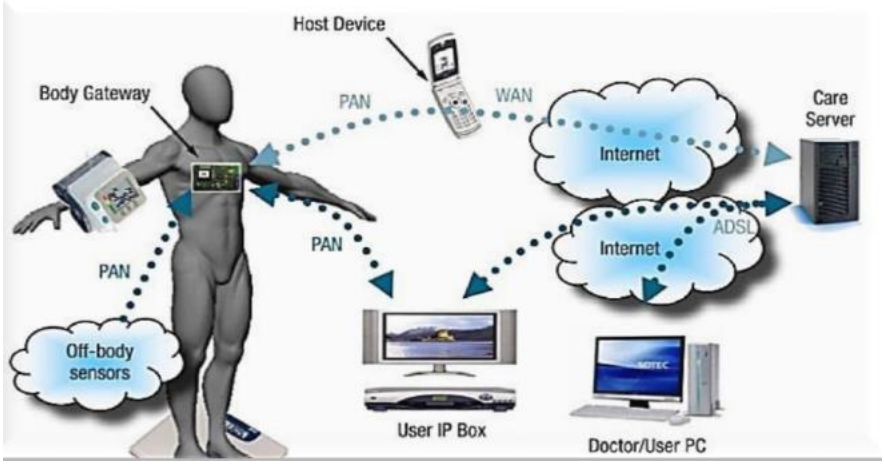


Figura 12. IoT Applicationes de Salud.
Fuente: (Sayed & Mohamed, 2017).

2.2.2.5 Transporte Inteligente

El desarrollo en el transporte es uno de los factores que indican el bienestar del país. Una aplicación de alerta y monitoreo de la condición del camino es una de las más

importantes de la aplicación de transformación de IoT. La idea principal del concepto de transporte inteligente y movilidad es aplicar los principios de la contratación de personas y la detección participativa.

El proceso comenzó cuando el usuario identificó los deseos de la ruta y marcó algunos puntos como baches en la aplicación del teléfono inteligente, el transporte inteligente se ocupa de tres conceptos principales, como se muestra en la Figura 13, que son analítica de transporte, control de transporte y conectividad de vehículos. La analítica del transporte representa el análisis de predicción de demanda y detección de anomalías. La ruta de los vehículos y el control de velocidad, además de la gestión del tráfico, se conocen como control de transporte, que en realidad están estrechamente relacionados con la conectividad de los vehículos (comunicación V2X), y en general están regidos por la difusión de múltiples tecnologías.

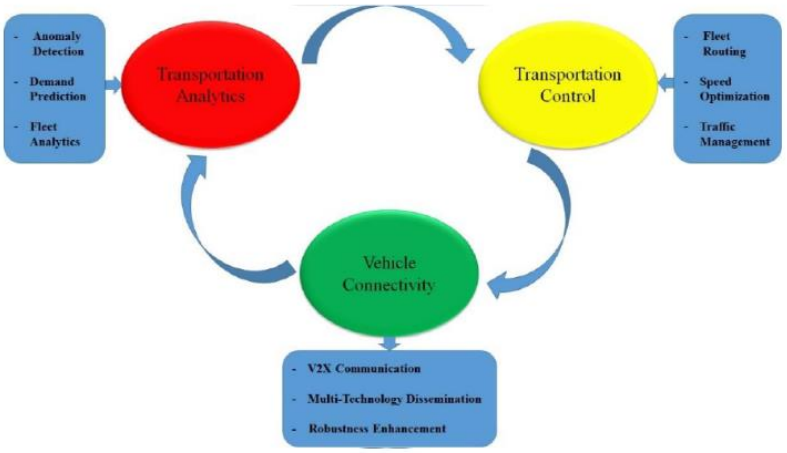


Figura 13. IoT Transporte Inteligente.
Fuente: (Sayed & Mohamed, 2017).

2.2.2.6 Fabrica Inteligente

La Fábrica Inteligente agregó nuevos valores en la revolución de la fabricación al integrar la inteligencia artificial, el aprendizaje automático y la automatización del trabajo de conocimiento y la comunicación M2M con el proceso de fabricación, además cambiará fundamentalmente cómo se inventan, fabrican y envían los productos.

Al mismo tiempo, mejorará la seguridad de los trabajadores y protegerá el medio ambiente al permitir bajas emisiones y una baja fabricación de incidentes, estos avances en la forma en que las máquinas y otros objetos se comunican y la forma resultante en que la toma de decisiones pasa de los humanos a los sistemas técnicos significa que la fabricación se vuelve "más inteligente", nuevas tecnologías tales como; la automatización, la robótica y la movilidad autónoma son todos un medio de fabricación inteligente, pero las comunicaciones M2M habilitadas por la Internet "industrial" de las cosas proporcionan un significado completo de fábrica inteligente y fabricación inteligente por el concepto de Big Data que, en este contexto, se refiere a las posibilidades analíticas ofrecidas por el volumen y la variedad de datos que genera una economía en red para optimizar los procesos industriales para implicar menos tiempo de inactividad de mantenimiento, menos interrupciones y un consumo de energía muy reducido.

La industria inteligente como una cuarta generación conocida como industria 4.0 se basa en sistemas físicos de cifrado que pueden conectarse a Internet, este concepto de con el Internet de las cosas puede lograr grandes expectativas para las industrias debido a que esta resolución abarca muchos aspectos como se muestra en la Figura 14.

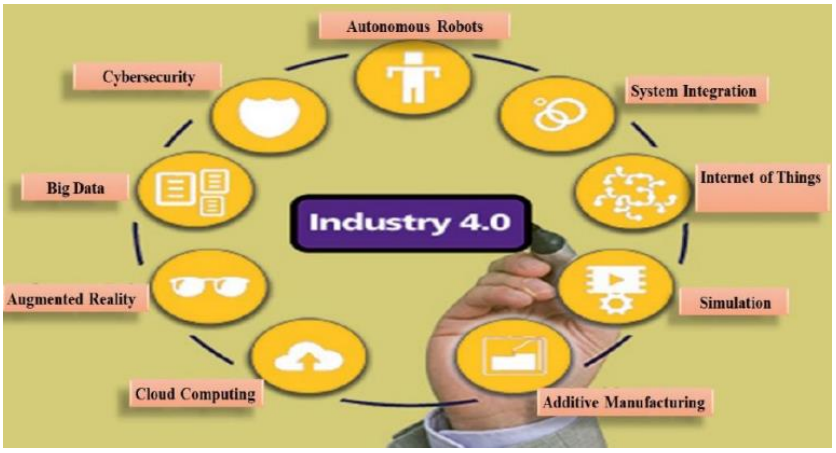


Figura 14. IoT Fabrica Inteligente.
Fuente: (Sayed & Mohamed, 2017).

2.2.3 Desafíos IoT

El hecho de que las aplicaciones y los escenarios de Internet de las cosas explicados anteriormente son muy interesantes y proporciona tecnologías para todo tipo de aplicaciones, pero hay algunos desafíos para la aplicación del concepto de Internet de las cosas, tales como los costos de implementación. A continuación, se detallan los desafíos más importantes presentes para el desarrollo de aplicaciones IoT

2.2.3.1 Escalabilidad

En IoT las cosas se cooperan en un entorno abierto, por lo tanto, la funcionalidad básica, como la comunicación y el descubrimiento del servicio, deben funcionar con la misma eficiencia en entornos tanto a pequeña escala como a gran escala, además IoT requiere nuevas funciones y métodos para obtener una operación eficiente para la escalabilidad.

2.2.3.2 Auto Organización

Las cosas inteligentes no deben administrarse como computadoras que requieren que sus usuarios las configuren y las adapten a situaciones particulares, estas necesitan establecer

conexiones espontáneamente, y pueden organizarse y configurarse para adaptarse a su entorno particular.

2.2.3.3 Volúmenes de Datos

Algunos escenarios de aplicaciones de Internet de las cosas implicarán una comunicación poco frecuente, y la recopilación de información de las redes de sensores, o la logística y las redes a gran escala, recopilará una gran cantidad de datos en los nodos o servidores de la red central. El término que representa este fenómeno es big data, que requiere muchos mecanismos operativos además de nuevas tecnologías para el almacenamiento, procesamiento y gestión.

2.2.3.4 Interpretación de Datos

Para apoyar a los usuarios de cosas inteligentes, es necesario interpretar el contexto local determinado por los sensores con la mayor precisión posible. Para que los proveedores de servicios se beneficien de los datos que se generarán, deben poder extraer algunas conclusiones generalizables a partir de los datos del sensor interpretados.

2.2.3.5 Interoperabilidad

Cada tipo de objetos inteligentes en Internet of Things tiene diferentes capacidades de información, procesamiento y comunicación, estos objetos inteligentes también estarían sujetos a diferentes condiciones, como la disponibilidad de energía y los requisitos de ancho de banda de las comunicaciones, para facilitar la comunicación y la cooperación de estos objetos, se requieren estándares comunes.

2.2.3.6 Complejidad de Software

Se necesitará una infraestructura de software más extensa en la red y en los servidores de fondo para administrar los objetos inteligentes y proporcionar servicios que los respalden. eso debido a que los sistemas de software en objetos inteligentes tendrán que funcionar con recursos mínimos, como en los sistemas integrados convencionales.

2.2.3.7 Seguridad Privacidad

Además de los aspectos de seguridad y protección de Internet, como la confidencialidad de las comunicaciones, la autenticidad y la confiabilidad de los interlocutores de comunicación y la integridad de los mensajes, otros requisitos también serían importantes en una Internet de las cosas, ya que existe la necesidad de acceder a ciertos servicios o evitar la comunicación con otras cosas en IoT y también las transacciones comerciales que involucran objetos inteligentes deberían estar protegidas de las miradas indiscretas de los competidores.

2.2.3.8 Fuente de Energía

Las cosas normalmente se mueven y no están conectadas a una fuente de alimentación, por lo que su inteligencia debe ser alimentada por una fuente de energía autosuficiente. Si bien los transpondedores RFID pasivos no necesitan su propia fuente de energía, su funcionalidad y rango de comunicaciones son muy limitados, las esperanzas están puestas en futuros procesadores de baja potencia y unidades de comunicaciones para sistemas integrados que pueden funcionar con significativamente menos energía. El ahorro de energía es un factor no solo en el hardware y la arquitectura del sistema, sino también en el software, por ejemplo, la implementación de pilas de protocolos, donde cada byte de transmisión tendrá que justificar su existencia.

2.2.3.9 Comunicaciones Inalámbricas

Desde un punto de vista energético, las tecnologías inalámbricas establecidas como GSM, UMTS, Wi-Fi y Bluetooth son mucho menos adecuadas; los estándares WPAN más recientes, como ZigBee y otros aún en desarrollo, pueden tener un ancho de banda más estrecho, pero usan significativamente menos energía.

2.3. Visión Artificial

La visión artificial se puede definir como un campo científico que extrae información de imágenes digitales. El tipo de información obtenida de una imagen puede variar según la identificación, las mediciones de espacio para la navegación o las aplicaciones de realidad aumentada (Krishna, 2017). Otra forma de definir la visión por ordenador es a través de sus aplicaciones. La visión artificial está creando algoritmos que pueden comprender el contenido de las imágenes y utilizarlo para otras aplicaciones. Los orígenes de la visión por computadora se remontan a un proyecto de verano de licenciatura del MIT en 1966. En ese momento se creía que la visión por computadora podía resolverse en un verano, pero ahora tenemos un campo científico de 50 años que aún está lejos de resolverse (Krishna, 2017).

En la Figura 15 se muestra un ejemplo de una aplicación de visión artificial industrial donde se chequea si es correcto el nivel de líquido en las botellas que se desplazan por una banda transportadora, si no es correcto es rechazada mediante actuadores especiales.

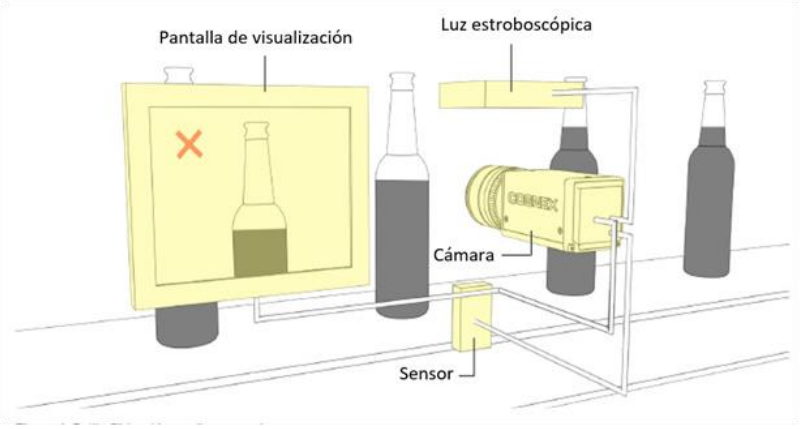


Figura 15 Sistema de inspección de nivel en botellas usando de visión artificial.

Fuente: (Krishna, 2017)

La posición de la botella frente a la cámara se detecta mediante un sensor y se ilumina mediante una luz estroboscópica en el momento de capturar la imagen que será procesada para determinar si el nivel de líquido en la botella está dentro del rango de tolerancia en el proceso de producción.

Existen una gran cantidad de aplicaciones industriales automatizadas usando visión artificial, tales como: inspección para detección de fallas, control de calidad de productos, ubicación de piezas para ensamblaje, soldadura, presencia de vehículos en un parqueadero, clasificación de productos, entre otras. Muchas de estas aplicaciones son parte de sistemas robóticos. En general, la visión artificial tiene un elevado número de aplicaciones en instrumentación y sistemas de control de diferentes áreas del conocimiento.

La función principal de la visión artificial es extraer del mundo físico imágenes digitales en una computadora o procesador con la finalidad de realizar un trabajo relacionado con la información del entorno donde se obtuvo, en la Figura 16 se presenta algunas formas de reconocimiento para diferentes tipos de trabajos.

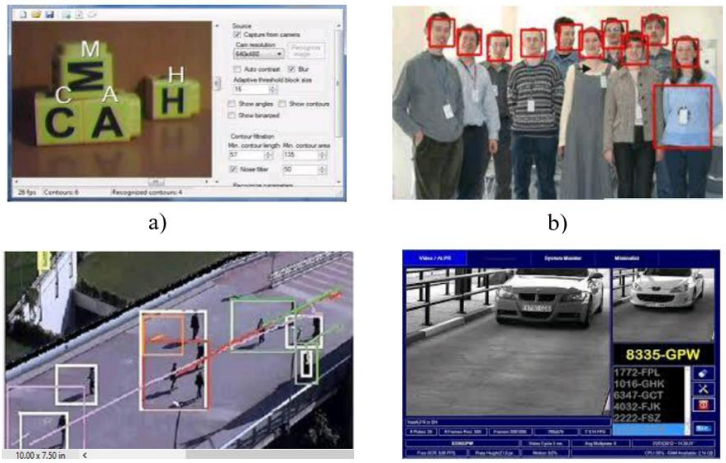


Figura 16 Reconocimiento de diferentes formas. a) Letra en un cubo. b) Rostros en una fotografía. c) Personas en un puente. d) Números en placas de vehículos.

Fuente: Imágenes tomadas de la web.

Una imagen digital se representa mediante una matriz bidimensional que se almacena en la memoria del computador, cada elemento de la matriz se denomina pixel y contienen la información numérica de un byte que caracteriza el nivel de gris cuando las imágenes son en escala de grises o por tres byte que contienen la intensidad de cada color base en (RGB) en imágenes de color.

En la Figura 17 se presenta un diagrama de bloques donde se muestra las etapas de un sistema de visión artificial según Platero (2009). La adquisición de la imagen se realiza con sensores que se encuentran en la cámara digital y es almacenada en la memoria del computador como parte del conocimiento que se utiliza en las siguientes etapas. El procesamiento de la imagen se realiza para preparar la imagen eliminando las partes que no son útiles y sobresaltando las de mayor interés. La segmentación tiene como finalidad aislar los elementos de la imagen que aportan datos de interés en la aplicación. En la siguiente etapa las porciones producto de la segmentación se representa y describen para finalmente realizar el reconocimiento e interpretación de la imagen según sea la aplicación.

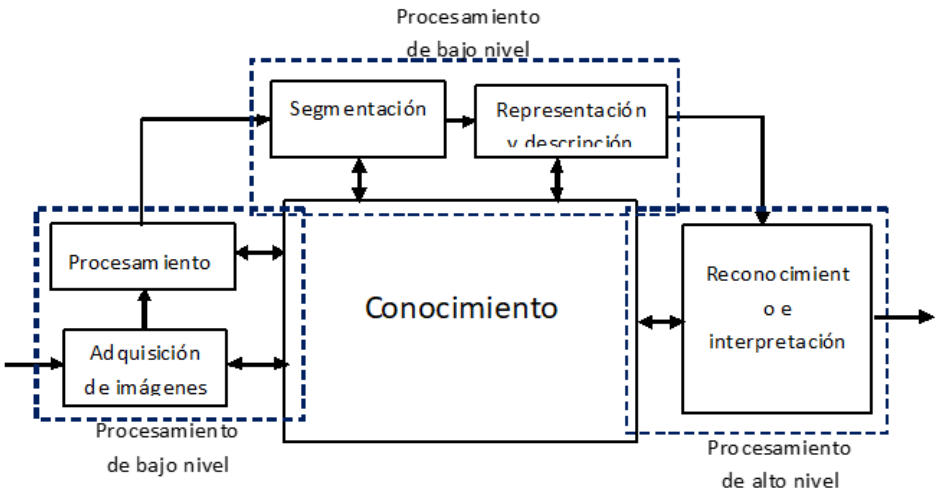


Figura 17 Etapas del proceso de un sistema de visión artificial
Fuente: Platero (2009).

2.3.1. Adquisición de Imágenes

El proceso de adquisición de imágenes cambia según sea la aplicación, en sistemas donde se requiere adquirir imágenes de objetos en movimiento es necesario sensores especiales y luz estroboscópica para sincronizar el momento en que se dispara la cámara. En entornos donde es necesario que en la aplicación se realice reconocimiento de objetos estáticos solo es necesario una buena iluminación que permita obtener buenas imágenes.

2.3.1.1. Cámaras lineales

Las cámaras lineales como su nombre lo indica hacen un barrido lineal para realizar la construcción de una imagen línea a línea, utilizando un sensor lineal que generalmente tienen entre 512 y 12.000 píxeles, lo más cercanos posible y de muy buena calidad con el fin de obtener la mejor sensibilidad y prestaciones (Cognex, 2016). Las cámaras lineales inicialmente se utilizan en aplicaciones de inspección de materiales fabricados de forma continua, como papel, tela, planchas metálicas, entre otros. En la producción de este tipo de

materiales no hay un inicio o un fin definido, por lo que la longitud del material a inspeccionar suele ser indeterminada. Las cámaras lineales por tanto pueden capturar una imagen de una anchura conocida que determina el tamaño del sensor y una longitud ilimitada. (Ver Figura 18).

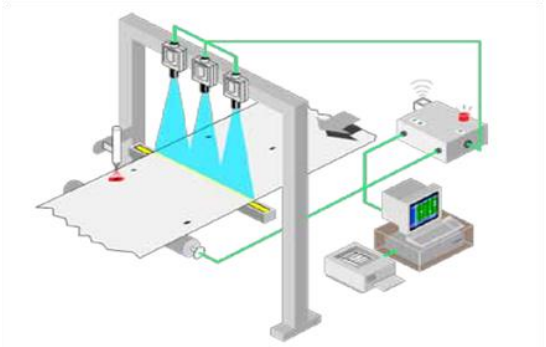


Figura 18 Ejemplo de inspección de productos continuos usando cámaras lineales.
Fuente: Cognex, (2016)

2.3.1.2. Cámaras matriciales

Las cámaras matriciales (ver Figura 19) o cámaras de área son una de las más usadas en visión artificial, tienen un sensor que cubre un área representada por una matriz de píxeles. Esta produce una imagen de un área normalmente con una relación de aspecto de 4 a 3, esta relación viene de las cámaras Vidicón y de los formatos de cine y televisión. Actualmente existen muchas cámaras que se han adaptado a nuevos formatos de alta definición (Cognex, 2016). Las cámaras modernas en su mayoría son dispositivos de acoplamiento de carga CCD (Por sus siglas en inglés, Charge Coupled Devices) que utilizan material sensible a la luz para convertir los fotones en carga eléctrica. Las imágenes son captadas por miles de diodos fotosensibles posicionados de forma muy precisa en una matriz y mediante registros de desplazamiento con las condiciones de los parámetros de los pixeles se forma una señal de video.

Figura 19 Cámaras matriciales 768x494.

Fuente: http://photo.techmaniak.es/23338/camera-matricial

2.4 Procesamiento de imágenes

Las imágenes matriciales también llamadas imágenes de área se representan mediante un matriz donde cada uno de sus elementos tiene la información de intensidad si imagen es en escala de grises y el contenido de colores rojo R, verde, G y azul B si la imagen es a color (RGB). El procesamiento de la imagen consiste aplicar a la matriz una técnica o algoritmo que realice cambios que permita segmentar las partes de la imagen que son relevantes para el reconocimiento de patrones propios de la aplicación. A continuación, se desarrolla como se representan las imágenes.

2.4.1 Representación Digital Simple

La representación más común de una imagen es una recopilación de puntos llamados pixeles en una matriz bidimensional, donde cada elemento de la matriz almacena parámetros propios de la imagen. Casi todos los dispositivos simples permiten adquirir imágenes en escala de grises de 256 valores o escalas, logivcamente se representa con un número tipo

byte, es decir 8 bits (Martin, 2013). En las imágenes de color se representa con tres byte que cada uno contiene la cantidad de color RGB.

$$I = \begin{bmatrix} x_{1,1} & \cdots & x_{1,n} \\ \vdots & \ddots & \vdots \\ x_{m,1} & \cdots & x_{m,n} \end{bmatrix} \quad \text{Ecu. 1}$$

Donde I es la matriz imagen, $x_{i,j}$ es el pixel ubicado en la i-ésima fila y j-ésima columna

El número de bits de un pixel determina establece el número de escales en una imagen, en la Tabla 1 se observa el caso de 1, 4 y 8 bits.

Tabla 1 *Niveles de escala de grises en función del número de bits.*

Número de bits	**Descripción**
1	Imagen Monocromática / Blanco y Negro / Alto contraste
4	Imagen de 16 niveles
8	Imagen de 256 niveles

Fuente: Martín (2013).

La representación de las imágenes a color cada pixel tiene tres componentes de color RGB y el color del pixel se expresa como una combinación lineal de los tres colores básicos con la ecuación Ecu. 2

$$C = rR + gG + bB \quad \text{Ecu. 2}$$

Donde un color C es la combinación lineal del vector unitario cromático (R, G, B) y la proyección en cada eje cromático o coordenadas de color (r, g, b), y C se puede decir que es la mezcla de los colores primarios con una cantidad r de rojo, g de verde y b de azul. Si los

colores tienen cada una escala de 256 valor es decir representados por 8 un byte entonces se dice que la imagen está representada por 24 bits (Martin, 2013).

2.4.2 Imágenes y capacidad de visión

La visión del ser humano está limitada para ver solo señales luminosas entre el rojo y el violeta, por la longitud de onda en que oscilan, en la Figura 20, se muestra la ubicación de los colores visibles en el espectro para longitud de onda en nanómetros (10^{-9} m).

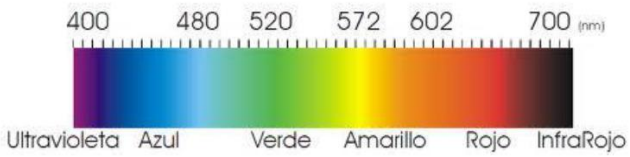


Figura 20 Colores y su posición en el espectro.

Fuente: Martín (2009)

2.4.3 Funciones de transformación de imágenes

Las operaciones que permiten la transformación de la imagen se realizan cambiando un pixel a la vez, independientemente de la condición de los pixeles vecinos. La transformación se puede aplicar a toda la imagen o solo a una parte. Supongamos que x es un pixel de la imagen I, es decir $x \in$ I, y tenemos una función de transformación de la imagen

$$y = f(x) \qquad \text{Ecu. 3}$$

que cambia el pixel *x* por y mediante la función *f* , se puede generar una nueva imagen I' aplicando esto a todos o una parte de los pixeles de I. Se puede expresar como

$$I' = f(I) \qquad \text{Ecu. 4}$$

Donde y $\in$ I' y de diremos que I' es una nueva imagen, producto de aplicar *f* sobre I.

El proceso de transformación en la mayor parte de los casos tiene como condición que

$$sí\ x = I[i, j] \Rightarrow y = I'[i, j], \text{ donde } y = f(x).$$

Para que la transformación *f* no ocasione problemas de representación, si el dominio de *x* está en el intervalo $D = [0, L - 1]$, donde $L = 2p$, donde *p* es la profundidad en bits de la imagen, entonces se debe cumplir que y $\in D'$, donde en general $D' \subseteq D$. Lo que implica que el mecanismo de representación de la imagen sobre elementos de la clase *x*, seguirá siendo válido para la clase a la que pertenece y. Esta condición permite que los métodos desarrollados para la visualización de la imagen I se pueden utilizar para I'.

El algoritmo básico de transformación bajo *f* para una región rectangular de I definida por

$$R = [i_1, \dots, i_2, j_1, \dots, j_2,]$$

Es el siguiente, que se muestra en la Figura 21

```
for i = i1, i2 {
    for j = j1, j2 {
        I'[i, j] = f( I[i, j] )
```

Figura 21 Algoritmo básico de transformación parcial de una imagen I con la función *f*.

Fuente: Martín (2013).

En el caso que: i_1=0, i =M (donde M=Imagen.Ancho-1), j_1=0, N (donde N= Imagen.Alto-1); el proceso modificará a toda la imagen. Al cambiar *f* la transformación será diferente. Si definimos la composición de transformaciones de la manera habitual, tendremos que:

$$f_1 \circ f_2 (I) = f\,1\,(f\,2\,(I)) \qquad \text{Ecu. 5}$$

En general al aplicar dos transformaciones a una imagen en diferente orden, no se debe esperar que la imagen resultante sea la misma, es decir, la composición de transformaciones no es conmutativa, simbólicamente tendremos que:

$$f_1 \circ f_2 (I) \neq f_2 \circ f_1 (I) \qquad \text{Ecu. 6}$$

Se definirá una batería o serie de transformaciones f_k mediante la constitución de ellas. Varias de las operaciones de mejora de la imagen, detección de bordes, etc., se detallan como una batería. El sentido de ésta es equivalente a la composición de las funciones que forman cada transformación. Sean f_1, f_2 ,…, f_n las funciones que definen cada proceso sobre la imagen, entonces la transformación compuesta o batería será:

$$F(I) = f_1 \circ f_2 \circ \ldots \circ f_n (I) = f_1 (f_2 (\ldots f_{n-1} (f_n (I))\ldots)) \qquad \text{Ecu. 7}$$

Gráficamente se puede representar el proceso de transformación múltiple mediante celdas, donde cada celda representa una transformación o filtro, como se ilustra en la Figura 22.

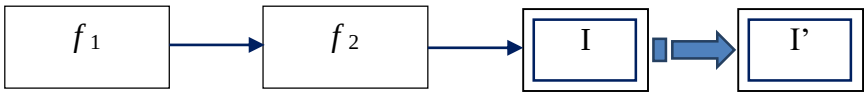


Figura 22 Representación gráfica de la composición de procesos.

Fuente: Martín (2013).

2.4.4 Operaciones básicas con imágenes

Introduciremos ahora algunas operaciones simples sobre la imagen. Sea I una imagen en colores, con un dominio [0, L] para los valores del tono de cada canal para los pixeles, en la representación RGB estándar, de ancho M y alto N. La operación más simple es la Identidad, ésta deja a la imagen igual. Podemos usar ésta para realizar por ejemplo copias de una imagen. La función correspondiente es: y = x, de donde f(x) = x. Si representamos ésta función de manera gráfica visualizaremos una ecuación de mapeo lineal simple Figura 23.

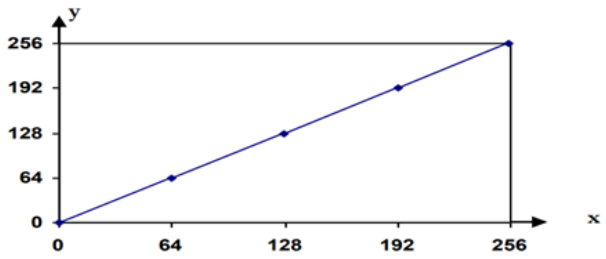


Figura 23 Representación gráfica de la función identidad con $y = f(x) = x$, (p=8, L =255).

Fuente: Martín (2013).

2.4.4.1. Negativo de una imagen

Esta transformación es muy simple y se construye de alguna de las siguientes maneras, sea x=(r, g, b) un pixel de la imagen I, entonces el negativo de x se puede hallar simplemente como se muestra en la ecuación Ecu. 8, que es la siguiente:

$$x' = (\sim r, \sim g, \sim b) = (\Lambda - r, \Lambda - g, \Lambda - b) \qquad \text{Ecu. 8}$$

Donde $\Lambda = L - 1$ y $L = 2^p$. De forma gráfica para cada canal el negativo se puede interpretar como una línea de transformación con pendiente negativa, como se ve en la Figura 24. Este proceso es muy claro de entender para una imagen monocromática, pues en ella p = 1, y entonces L = 2 y en consecuencia $\Lambda = 1$, de donde dado que los valores permitidos para un pixel (en cada plano) serán únicamente $x = \{0, 1\}$, de donde las transiciones serán: $0 \rightarrow 1$ y $1 \rightarrow 0$. De donde un pixel en 0 (negro) se transformará en 1 (blanco) y viceversa.

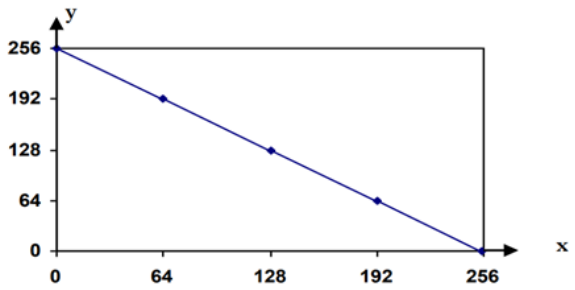


Figura 24 Representación gráfica de función negativa con un canal con *x=255- x*, (p=8).

Fuente: Martín (2013)

En el siguiente ejemplo se aplica la transformación a una imagen en tonos de gris (r=g=b=z), de donde el negativo de un pixel x = (z, z, z) será x' = (~z, ~z, ~z). En la Figura 25 se muestra el proceso sobre dos imágenes de ejemplo una monocroma y otra en tonos de gris. Imagen Original Negativo de la Imagen Monocroma

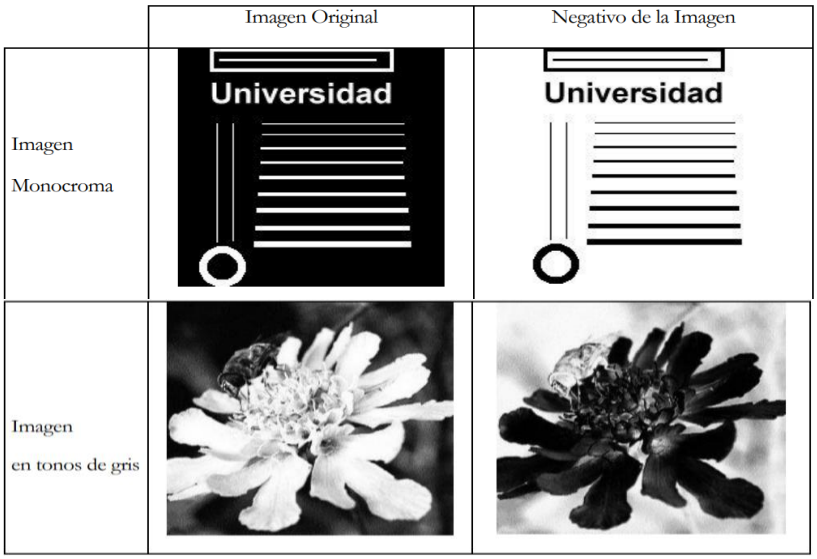


Figura 25 Imágenes y sus negativos.

Fuente: Martín (2013).

2.4.4.2. Transformaciones funcionales

En general, uno puede pensar en una función que transforme cada canal z' = fi (z), donde z puede ser cualquiera de los tres canales que forman a una imagen en color z $\in$ {r, g, b}. La transformación más general tendrá la forma:

$$x' = (r', g', b') = F(x) = (f1(r, g, b), f2(r, g, b), f3(r, g, b)) \quad \text{Ecu. 9}$$

Donde las fi (i=1, 2, 3) son funciones de salida entera cualesquiera. En general la respuesta en un canal puede depender de los valores de entrada en los demás canales considerándolo a él mismo. Los procesos más simples de interpretar son aquellos en los cuales la respuesta para un canal solo depende de la entrada en él, a estos se les llamará directos, es decir

$$x' = (r', g', b') = F(x) = (f1\ (r), f2\ (g), f3\ (b)) \quad \text{Ecu. 10}$$

Si además las formas funcionales de las fi son iguales, se tendrá una transformación directa simétrica, a estos filtros se les llamará directos simétricos, su forma es

$$x' = (r', g', b') = F(x) = (f(r), f(g), f(b)) \qquad \text{Ecu. 11}$$

Para las ecuaciones Ecu. 10 y Ecu 11 se analizan algunas situaciones que se producen cuando las funciones f, fi son monótonas crecientes o decrecientes y además tocan los puntos de referencia (0, 0) y (Λ, Λ). En general las transformaciones funcionales se denominan "Transformaciones u operaciones puntuales", esto se debe a que aplica a un punto de la imagen en la posición (i,j) y generan un nuevo valor que se asigna a la imagen transformada en su coordenada (i,j). A las transformaciones en el lenguaje del procesamiento de imágenes se les denominan filtros por su análogo en la óptica. Una transformación básica en esta familia es el filtro de corrección de luz o corrección gamma (γ), también por su forma se le llama función potencia. Ésta tiene la forma normalizada

$$z' = \Lambda\left(\frac{z}{\Lambda}\right)^{\gamma} \qquad \text{Ecu. 12}$$

Donde γ es un real positivo y $z \in [0, \Lambda]$, por lo cual también $z' \in [0, \Lambda]$. Se pueden identificar dos clases de transformaciones: la primera corresponde al caso cuando $\gamma \in (0, 1]$. Estas transformaciones van a realizar un proceso de "aclarado de la imagen", si observamos las siguientes gráficas de la Figura 26, podemos ver que las funciones son monótonas crecientes y están sobre la línea de la identidad, de donde para todos los valores de z (excepto el blanco y el negro) z' estará arriba de la identidad, es decir el tono será más claro.

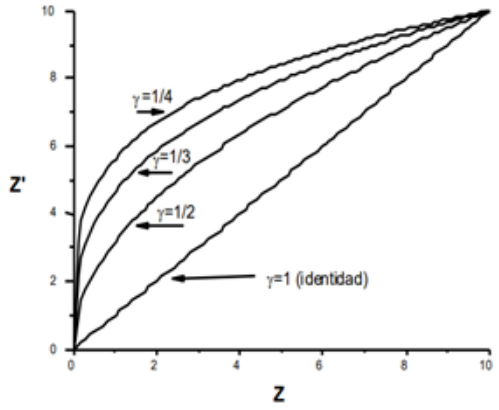


Figura 26 Función potencia $11(z/11)^{\gamma}$

Fuente: Martín (2013).

El segundo caso corresponde a la situación cuando $\gamma > 1$. Lo que sucede ahora es que todos los puntos están por debajo de la identidad, por lo tanto, la transformación "obscurecerá la imagen" (Fig. 27) en consecuencia. A esta transformación (para los valores de $\gamma > 0$) se le llama corrección de gamma, debido a la letra que se utiliza en la función potencia.

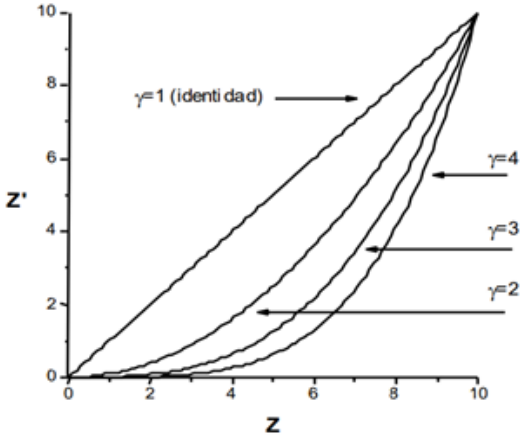


Figura 27 Función potencia $11(z/11)^{\gamma}$

Fuente: Martín (2013).

En la siguiente tabla se muestra el resultado de aplicar ésta transformación para una imagen, usando los valores $\gamma = ½$ y $\gamma = 2$. (Ver Figura 28)

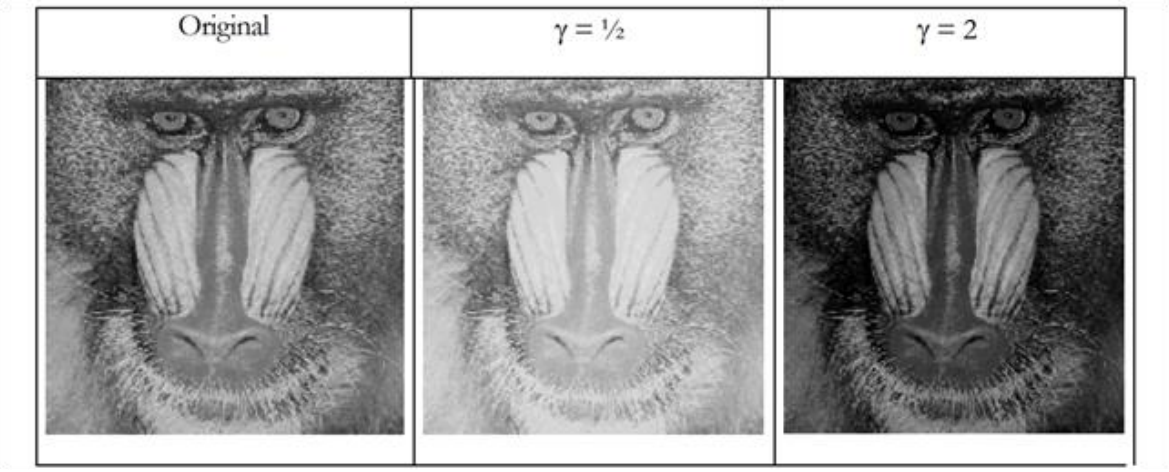


Figura 28 Aplicación de la función de potencia.

Fuente: Martín (2013).

2.5 Computadores de Placa Simple (SBC)

Las computadoras de placa única (SBC, son dispositivos informáticos pequeños que pueden usarse para una variedad de propósitos que incluyen experimentación, aprender a programar, crear un reproductor de medios o unidad NAS, robótica doméstica y automatización del hogar, y realizar tareas informáticas como la navegación web o el procesamiento de textos. Los SBC también se utilizan cada vez más para una amplia gama de aplicaciones industriales en áreas que incluyen robótica e Internet de las cosas (IoT).

2.5.1 Raspberry Pi

Raspberry Pi, es una tarjeta electrónica de computador del tamaño de tarjeta de crédito que se le conecta una pantalla o monitor y un teclado. Soporta diversos componentes propios

del computador personal (PC). Tiene la capacidad de utilizar software que se utilizan en un PC de escritorio o en una laptop, tales como: como hojas de cálculo, procesadores de texto y juegos. También reproduce vídeo de alta definición. Desde el año 2012 hasta la fecha actual han introducido al mercado nuevas versiones cada una con mejoras importantes, sin incrementos de coste significativos (Raspberry, 2018). A continuación, se presenta un resumen de las versiones de Raspberry.

2.5.1.1 Raspberry Pi 1 Modelo B y B+

Se desarrolló en 2012, es similar al modelo A con un conjunto de mejoras, se incluyó el doble de memoria RAM, pasando de 256MB a 512MB. Se le agregó un puerto USB más y un conector Ethernet (RJ-45) Tiene el mismo tamaño y coste que el modelo A. Mantuvo el mismo procesador y la parte gráfica. Poco tiempo después se lanzó el Modelo B+ que incluyó 4 puertos USB y se comenzó a usar una MicroSD en vez de la SD (Raspberry, 2018).

2.5.1.2 Raspberry Pi 1 Modelo B

Poco tiempo después, en el año 2014, aparece el Raspberry Pi 2 B que incluye por primera vez un nuevo procesador modelo BCM2836 con importantes mejoras con respecto a versiones anteriores. Pasa de ser de un núcleo a cuatro, y de 700 MHz a 900 MHz. No obstante, emplea la misma gráfica VideoCore IV. Dobla la cantidad de memoria RAM, pasando de 512MB a 1GB (Algo menos en realidad) esta memoria está compartida con la gráfica. También incluye 40 pines GPIO, y mantiene los cuatro puertos USB. Suprime la conexión RCA (Raspberry, 2018).

2.5.1.1 Raspberry Pi 3 Modelo B

Esta nueva versión mantiene el mismo procesador QUAD Core Broadcom BCM2837 de 64bit ARMv7. A continuación se presenta las mejoras de Pi 2 Modelo B a Pi 3 Modelo B: La velocidad del procesador aumentó de 900 MHz en Pi 2 a 1.25Ghz en el Pi 3 Modelo BBCM43143 Wi-Fi embebido. Bluetooth de bajo consumo (BLE) embebido. Fuente de alimentación conmutada actualizada de hasta 2.5 A, permite alimentar dispositivos aún más potentes a través de puertos (USB) (Raspberry, 2018). Las principales diferencias son la CPU de núcleo cuádruple de 64 bits y Wi-Fi y Bluetooth a en la tarjeta. (Ver Figura 29).

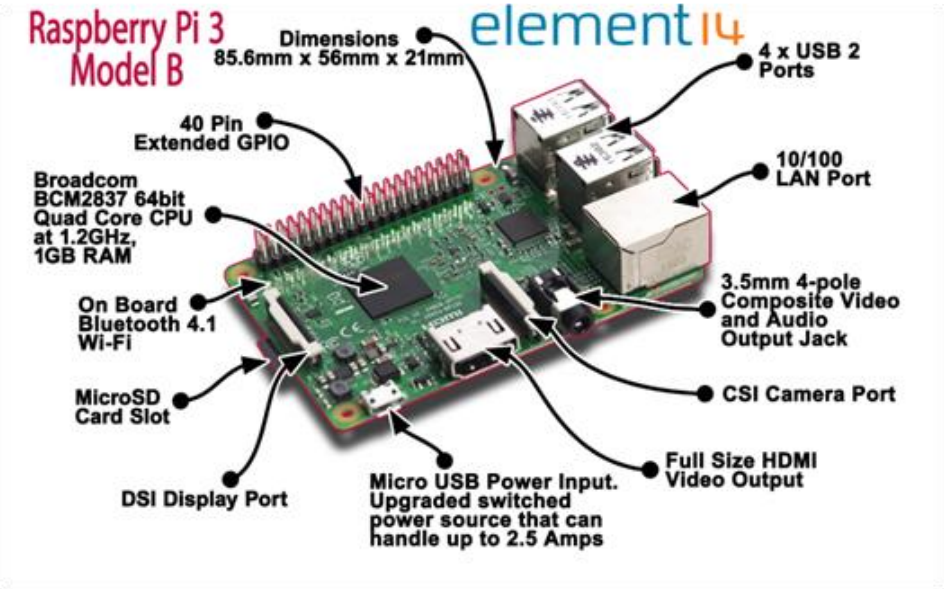


Figura 29 Tarjeta Raspberry Pi 3 modelo B.

Fuente: https://www.raspberrypi.org/

La memoria RAM sigue siendo de 1 GB y no hay cambios en los puertos USB o Ethernet. Sin embargo, la administración de energía mejorada significa que el Pi 3 puede hacer uso de más dispositivos USB que requieran mayor consumo. El procesador del Pi 3 fue remplazado

por el BCM2837. Esta conserva la misma arquitectura básica que sus predecesores BCM2835 y BCM2836, por lo que todos los proyectos y tutoriales que se basan en los detalles del hardware Raspberry Pi continuarán vigentes. El complejo CPU ARM Cortex-A7 de cuatro núcleos y 900 MHz de 32 bits ha sido reemplazado por un procesador ARM Cortex-A53 de cuatro núcleos de 64 GHz a prueba de cambios. En términos de tamaño, es idéntico a B + y Pi 2. Todos los conectores y orificios de montaje están en el mismo lugar, por lo que todos los complementos, sombreros y estuches existentes deberían quedar bien aunque los LED de potencia y actividad se hayan movido para dejar espacio para la antena WiFi. El rendimiento del Pi 3 es aproximadamente 50-60% más rápido que el Pi 2, lo que significa que es diez veces más rápido que el Pi original. Todos los conectores están en el mismo lugar y tienen la misma funcionalidad, y la placa aún se puede ejecutar desde un adaptador de alimentación micro USB de 5V. Se recomienda un adaptador de 2.5 A si desea conectar dispositivos USB con mucho consumo de energía a la Raspberry Pi (Raspberry, 2018).

2.5.2 JaguarBoard

Jaguar Electronic HK Co., Ltd lanzó oficialmente la primera computadora de placa única basada en X86 llamada Jaguarboard. Basado en el procesador Intel Atom de cuatro núcleos, proporciona 1 GB de memoria RAM DDR3L y 16 GB de almacenamiento flash eMMC incorporado. Ofrece a los desarrolladores integrados y expertos en bricolaje experiencia mejorada con el desarrollo de software y simplifica significativamente los kits de herramientas de desarrollo (Jaguar Board, 2018). En la Figura 30 y 31 se detallan las especificación e interfaces de esta tarjeta.

JaguarBoard Specifications		
	Jaguar One	Jaguar One Plus
Dimension	101.9mm x 64.5mm x 1.6mm	
CPU Model	Intel Atom Z3735G	Intel Atom Z3735F
Memory	1GB DDR3L	2GB DDR3L
Storage Capacity	On Board 16GB eMMC	
Operating Systems	Linux/Android/Windows	
Power Supply	5V/2A	
HDMI	1 x HDMI1.4, up to 1080P	
Network Interface	1 x 10/100M	
USB	3 x USB 2.0	
TF Socket	1	
Audio	Stero speaker, Mono mic in, 3.5mm	
COM Ports	2	
I2C	1	
GPIO	4 GPIO Pins	

Figura 30 Especificaciones tarjeta JaguarBoard.

Fuente:
http://www.jaguarboard.org/index.php/com_virtuemart_menu_configuration/products/product_show/jaguarboard-industry-first-x86-based.html

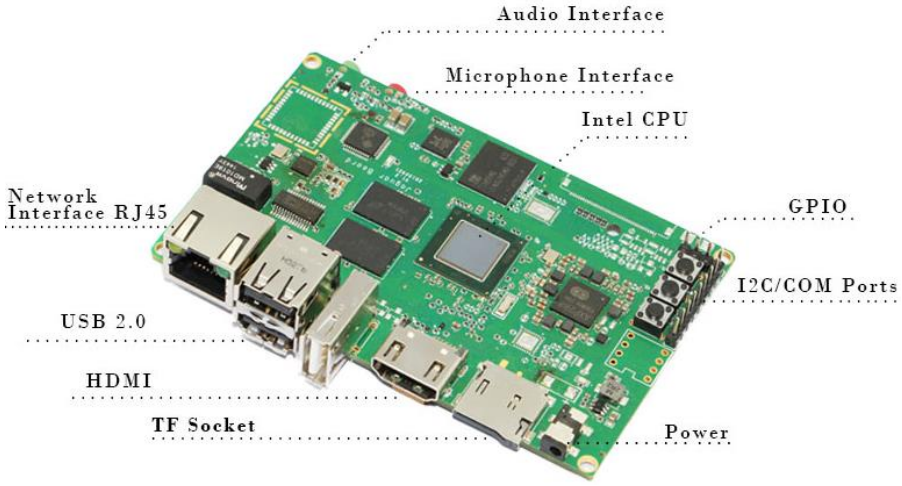


Figura 31 Diagrama de interfaces JaguarBoard

Fuente:
http://www.jaguarboard.org/index.php/com_virtuemart_menu_configuration/products/product_show/jaguarboard-industry-first-x86-based.html

2.5.3 BeagleBone Black

El BeagleBone Black es el miembro más nuevo de la familia BeagleBoard. Es un BeagleBoard enfocado de bajo costo y alta expansión que usa un procesador Sitara XAM3359AZCZ100 Cortex A8 ARM de bajo costo de Texas Instruments. Es similar al Beaglebone, pero con algunas características eliminadas y algunas características agregadas.

Lanzada el 25 de abril a un precio de $45(USD). Entre otras diferencias, incrementa la RAM a 512 MB, el reloj de procesador a 1 GHz, y añade HDMI y 2 GB de memoria flash eMMC. La BeagleBone Black también se entrega con kernel Linux 3.8, actualizado del kernel Linux 3.2 de la BeagleBone original, permitiendo a la BeagleBone Black tener la ventaja del Gestor de Renderizado Directo (DRM).

La revisión C de la BeagleBone Black (lanzada en 2014) incrementa el tamaño de la memoria flash a 4 GB. Esto le permite ser entregada con Debian GNU/Linux instalado (BeagleBoard , 2018). En la Figura 32 y 33 se detallan las especificación e interfaces de esta tarjeta.

	Feature	
Processor	Sitara AM3358BZCZ100 1GHz, 2000 MIPS	
Graphics Engine	SGX530 3D, 20M Polygons/S	
SDRAM Memory	512MB DDR3L 800MHZ	
Onboard Flash	4GB, 8bit Embedded MMC	
PMIC	TPS65217C PMIC regulator and one additional LDO.	
Debug Support	Optional Onboard 20-pin CTI JTAG, Serial Header	
Power Source	miniUSB USB or DC Jack	5VDC External Via Expansion Header
PCB	3.4" x 2.1"	6 layers
Indicators	1-Power, 2-Ethernet, 4-User Controllable LEDs	
HS USB 2.0 Client Port	Access to USB0, Client mode via miniUSB	
HS USB 2.0 Host Port	Access to USB1, Type A Socket, 500mA LS/FS/HS	
Serial Port	UART0 access via 6 pin 3.3V TTL Header. Header is populated	
Ethernet	10/100, RJ45	
SD/MMC Connector	microSD , 3.3V	
User Input	Reset Button Boot Button Power Button	
Video Out	16b HDMI, 1280x1024 (MAX) 1024x768,1280x720,1440x900 ,1920x1080@24Hz w/EDID Support	
Audio	Via HDMI Interface, Stereo	
Expansion Connectors	Power 5V, 3.3V , VDD_ADC(1.8V) 3.3V I/O on all signals McASP0, SPI1, I2C, GPIO(69 max), LCD, GPMC, MMC1, MMC2, 7 AIN(1.8V MAX), 4 Timers, 4 Serial Ports, CAN0, EHRPWM(0,2),XDMA Interrupt, Power button, Expansion Board ID (Up to 4 can be stacked)	
Weight	1.4 oz (39.68 grams)	
Power	Refer to Section 6.1.7	

Figura 32 Especificaciones tarjeta Beaglebone Black.

Fuente: https://elinux.org/Beagleboard:BeagleBoneBlack

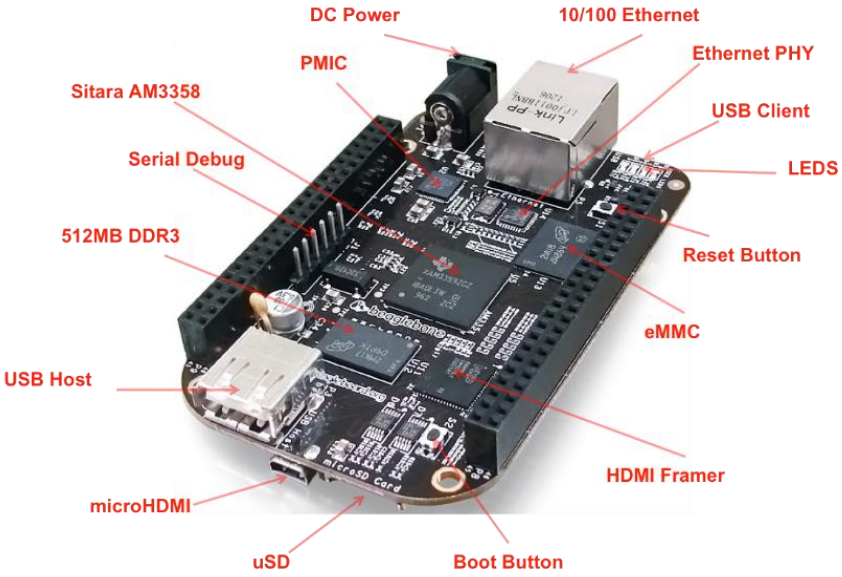


Figura 33 Diagrama de interfaces Beaglebone Black

Fuente: https://elinux.org/Beagleboard:BeagleBoneBlack

2.8. Software para procesamiento de Imágenes

Actualmente se cuenta con lenguajes de programación que dentro de sus herramientas tienen librerías especializadas para el procesamiento de Imágenes, tales como C, C++, Matlab, entre otros. También existen librerías como OpenCV especialmente para aplicaciones de visión artificial. La primera versión fue desarrollada por Intel en 1999 y desde ese momento se ha utilizado en muchas aplicaciones. Actualmente, OpenCV cuenta con más de 500 funciones y una extensa documentación útiles para aplicaciones de visión artificial, tales como: reconocimiento de objetos, detector de desplazamiento, en muchas y diversas aplicaciones de control de proceso y de robótica, entre otras. OpenCV es multiplataforma, y existen versiones para Windows, Linux, Mac OS X, iOS y Android.

OpenCV contiene cuatro módulos, que son: **cv** contiene las funciones principales de la librería, cvaux contiene funciones auxiliares (Experimentales), cxcore contiene estructuras de datos y funciones para álgebra lineal, Highgui contiene funciones de GUI.

Capitulo III

3. Diseño del Sistema

En este capítulo se presenta en primer lugar la situación actual del parqueadero de la UTN, enseguida se hace un análisis de los sistemas embebidos que conforman el prototipo de detección de puestos disponibles en el parqueadero de la UTN. Luego se presenta los requerimientos de los usuarios basado en el estándar ISO/IEC/IEEE29148 con el fin de optimizar los recursos necesarios del sistema. Finalmente se presenta y se realiza un análisis del diseño del prototipo del sistema de detección de puestos disponibles mediante visión artificial y por último se presenta el diseño de la aplicación con IoT.

3.1. Estado actual parqueaderos de la UTN

La Universidad Técnica del Norte es una institución con prestigio de nuestro país, la excelencia académica y su compromiso social permiten el desarrollo científico y tecnológico del país. El campus Universitario del Olivo está ubicado en la Av. 17 de julio 5-21 y General José María Córdova, cuenta con una extensión de 102.460 m2, diez edificios con modernas instalaciones, equipadas con tecnología de vanguardia, cuentan con auditorios, biblioteca, centro de copias e impresión, salas de exposición, salas de cómputo, laboratorios de investigación, talleres de diseño, salas de clase, entre otros servicios, cubiertos de amplias áreas verdes, acoge a más de siete mil personas entre docentes, estudiantes y funcionarios en jornada diurna y nocturna.

Entre uno de los diversos servicios que ofrece la institución a la comunidad universitaria es otorgar sitios de parqueo vehicular, siendo esta nuestra área de interés. Se detallará algunos

aspectos de la situación actual de los parqueaderos del campus tales como: usuarios, demanda, funcionamiento, distribución, seguridad, entre otros.

3.1.1. Parqueaderos de la Universidad Técnica del Norte

La UTN cuenta con siete parqueaderos, los cuales operan en un horario establecido a partir de las 6 am hasta las 10 pm. En la Figura 34, se detalla los ingresos a la institución y la ubicación de los estacionamientos con los que cuenta la Universidad.

Figura 34 Distribución de parqueaderos UTN.

Fuente: UTN.

Las zonas de parqueo de la Figura 34 son las siguientes:

1. Parqueadero Bar Universitario y FACAE.

2. Parqueadero FECYT

3. Parqueadero canchas UTN

4. Parqueadero administración central

5. Parqueadero FICA FICAYA

6. Parqueadero CAI, EDUCACION FISICA

7. Parqueadero piscina UTN

A1. Acceso vehicular 1 por la Avenida 17 de Julio

A2. Acceso vehicular 2 por la calle General José María Córdova

3.1.2 Sistema de parqueo y seguridad

La UTN cuenta con sistema de parqueo y seguridad realizado por guardias y cámaras de vigilancia. A continuación, se detallan sus características y modo de funcionamiento.

3.1.2.1 Procedimiento para el uso de los parqueaderos de la UTN

Una persona que pertenece a la Institución para poder hacer uso de los parqueaderos debe acercarse a la secretaria ubicada en el edificio central y cancelar un costo aproximado de 67 dólares americanos para disponer del servicio por un año. Los requisitos para adquirir un puesto son la matrícula del vehículo y la credencial universitaria.

Luego de realizar los trámites pertinentes se habilita el acceso mediante la credencial Universitaria la cual cuenta con tecnología RFID y funciona como llave de entrada en los accesos vehiculares de la universidad.

3.1.2.1 Control de acceso vehicular

El control de ingreso de los vehículos a los parqueaderos se realiza mediante tecnología RFID, en los puntos de acceso a la Universidad hay colocados sensores magnéticos que habilitan el paso con los carnets Universitarios. Al ingresar un vehículo por cualquiera de las puertas de la Institución, la persona encargada de controlar revisa que el conductor haga uso del carnet magnético para ingresar al campus universitario este proceso puede verse en la Figura 35.

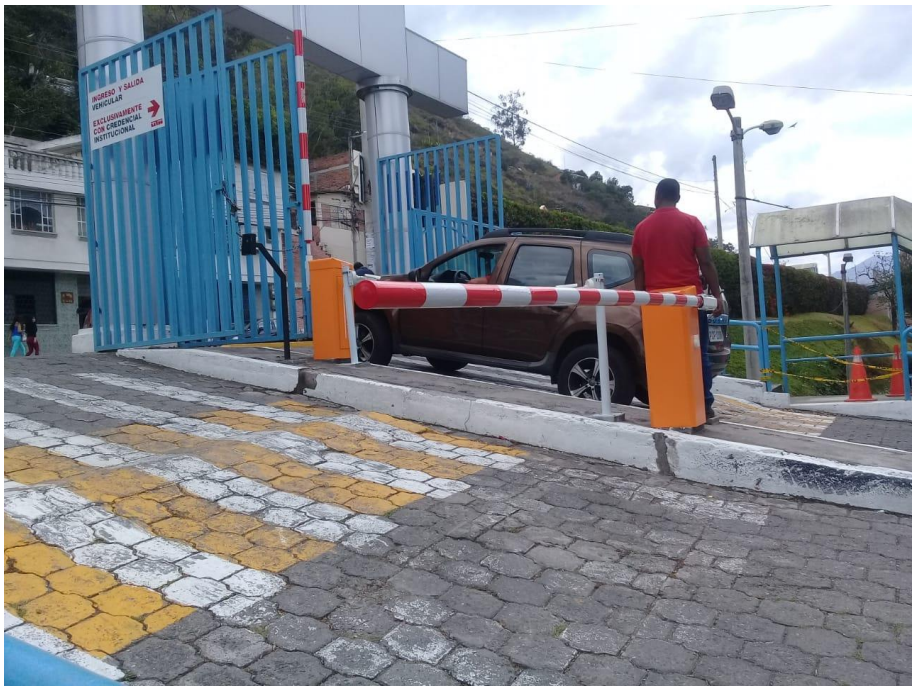


Figura 35 Acceso vehicular de la UTN.

Fuente: UTN.

3.1.2.2 Distribución y capacidad de los parqueaderos de la UTN

En la actualidad la UTN acoge más de nueve mil personas entre docentes, estudiantes y funcionarios tanto en la jornada diurna como nocturna. Algunos parqueaderos son de uso exclusivo para estudiantes y el resto de parqueaderos es para el uso de docentes y trabajadores. En la Tabla 2, se describe las características de los parqueaderos con los que cuenta la Institución, en la cual se presenta la ubicación, el nombre asignado y la capacidad con la que cuenta cada uno de los parqueaderos. Se pueden observar que hay un total de 384 puestos de parqueo. Para el diseño del prototipo se seleccionó el parqueadero ubicado en la Facultad de Educación Ciencia y Tecnología que tiene una capacidad de 66 puestos de parqueo.

Tabla 2 *Descripción de parqueaderos de la UTN*

Parqueadero	**Ubicación**	**Nombre asignado**	**Numero de aparcamientos**
Parqueadero 1	Bar Universitario y FACAE	Parqueadero Bar Universitario y FACAE	61
Parqueadero 2	FECYT	Parqueadero FECYT	66
Parqueadero 3	Canchas UTN	Parqueadero canchas UTN	91
Parqueadero 4	Administración Central	Parqueadero Administración Central	14
Parqueadero 5	FICA –FICAYA	Parqueadero FICA – FICAYA	95
Parqueadero 6	Educación Física	Parqueadero Educación Física y CAI	42

Parqueadero 7	Piscina UTN	Parqueadero Piscina UTN	15
		TOTAL:	384

Fuente: Autoría propia.

3.1.2.3 Ventajas y desventajas del sistema actual de parqueo de la UTN

Luego del estudio del estado actual de los parqueaderos de la UTN, se ha determinado que las ventajas y desventajas del sistema de los parqueaderos de nuestra Universidad son las siguientes:

Ventajas

- El uso de los parqueaderos es exclusivo para personas que pertenecen a la Universidad.
- Los costos del servicio de parqueo son bajos en comparación a parqueaderos de la zona que realizan cobros ya sea por horas o por mensualidades.
- Los parqueaderos cuentan con espacios preferenciales para personas discapacitadas.

Desventajas

- El no contar con un sistema de detección de puestos libres causa saturación y pérdida de tiempo en los usuarios al momento de buscar un sitio de parqueo.
- No cuenta con una base de datos que permita identificar a los usuarios autorizados para ingresar a los parqueaderos o localizarlos en caso de que así amerite.

- Existe un gasto representativo en los salarios del personal de vigilancia, pues el sistema actual requiere dos o más personas por parqueadero, para poder brindar seguridad.

- No existe un sistema de información sobre la disponibilidad de sitios de parqueo, que le permitan al usuario encontrar lugares de una manera eficiente.

- La capacidad de los parqueaderos no es suficiente para abastecer la demanda actual por lo cual vehículos se parquean en sitios no permitidos

3.2. Diagrama de Bloques

El diagrama que se presenta a continuación es el que comprende las fases de diseño del sistema, está formado por 4 bloques los cuales a su vez están contienen varios subprocesos, se han planteado los bloques dependiendo de las funciones que cada uno debe desarrollar. En la Figura 36 se puede observar cada uno de ellos.

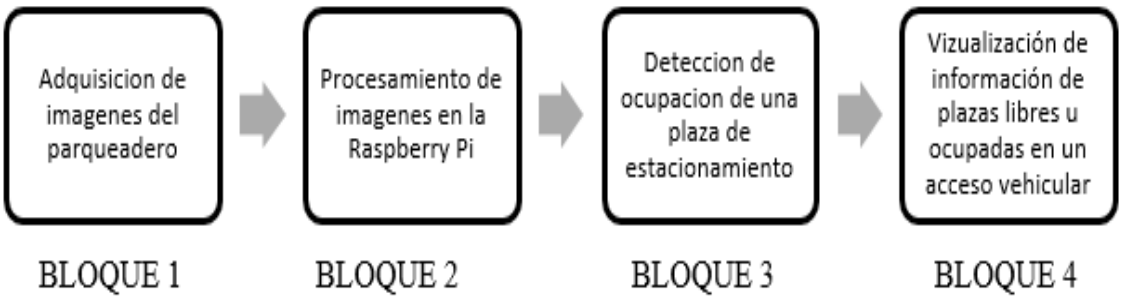


Figura 36 Diagrama de bloque del sistemas

Fuente: Autoría propia.

Para dar inicio con la adquisición de datos se inicia con el bloque 1, este es el encargado de que la cámara se inicie y comience a adquirir imágenes. En el bloque 2 se

inicia el trabajo de OpenCV, aquí se realiza un preprocesamiento a las imágenes adquiridas en el bloque anterior, además en este bloque se añaden los ROI. Las regiones ROI, de sus siglas en inglés “Region of Interest”, son funciones que permiten enfocarnos específicamente en un lugar de la imagen para sacar de éste sus características más importantes. La ROI determinada para nuestra imagen permite enfocar un lugar de estacionamiento y determinar si existe algún objeto ocupándole. Con los datos obtenidos en el bloque 2 y bloque 3 se realiza el proceso de análisis que permite determinar si hay plazas de estacionamiento libres. Finalmente, los datos del bloque 3 deben ser visualizados en un acceso vehicular.

3.3. Análisis de Sistemas Embebidos

Los sistemas embebidos son sistemas de computación que realizan una o varias funciones dedicadas generalmente para cubrir necesidades específicas en aplicaciones de tiempo real. El componente principal de un sistema embebido es el procesador o unidad de procesamiento computacional (CPU) que está basada en dispositivos tales como: un microprocesador, un microcontrolador, un procesador digital de señales (DSP) o un computador dedicado.

La CPU o procesador debe tener la capacidad de cómputo que permita realizar las funciones del sistema, mediante un microcontrolador que debe incluir memoria interna y/o externa, puertos de entrada/salida (I/O), puertos seriales, periféricos para conexión de teclados, pantallas, sensores entre otros. El sistema propuesto en este trabajo requiere puertos que le permitan conectar cámaras de video y conexión a una red Ethernet. El módulo computacional Raspberry PI 3 es una tarjeta electrónica que contiene todos los requerimientos del sistema embebido, antes mencionados, que conformará el prototipo para la detección de puestos libres en un parqueadero, incluyendo la capacidad para el

procesamiento de imágenes. El análisis de los requerimientos del sistema se describe en los siguientes puntos.

3.4 Requerimientos del Sistema

El estándar ISO/IEC/IEEE29148 de 2011 contiene normas para el tratamiento de los requisitos que debe cumplir los elementos que conforman un sistema de ingeniería específicamente en productos de software y servicios a lo largo de su ciclo de vida. El estándar define pautas para la elaboración de un buen requisito que le proporciona los mejores atributos y características al sistema.

El análisis de los requerimientos del sistema se realizó aplicando el estándar antes mencionado con el fin de optimizar la selección de los componentes del hardware y del software del sistema. Para esto se diseñaron tres tablas que contienen: los requerimientos iniciales del sistema, requerimientos de arquitectura y los requerimientos de stakeholders. El diseño propuesto para cada tabla incluye una columna donde se identifica el número de requerimiento, una columna destinada a la descripción detallada del requerimiento, la siguiente columna está destinada a indicar la prioridad del requerimiento.

A continuación, se presenta en la tabla 3 los requerimientos iniciales donde se establecen las condiciones funcionales del sistema, en la primera columna se utiliza la abreviatura RFS (Requisitos Funcionales del Sistema) numerada para facilitar la identificación cada requerimiento, en la segunda columna está la descripción y en la tercera columna se establece la prioridad de cada requerimiento. La prioridad se califica con tres valoraciones que son prioridad alta, prioridad media y prioridad baja, que permitirán evaluar los requisitos para obtener la mejor opción al escoger los elementos del sistema. Como se observa en la Tabla

3, los requisitos de subdividen en requisitos de funciones, requisitos de uso, requisitos de interfaces, requisitos de modo y estado y requisitos físicos. Se ha establecido que todos estos requisitos son de alta prioridad por lo tanto sus cumplimientos son de gran importancia para el funcionamiento del sistema.

Tabla 3 *Requerimientos iniciales del sistema RFS*

No.	Descripción del requerimiento	Prioridad		
		Alta	Media	Baja
	Requerimientos de funciones			
RFS1	El sistema debe estar instalado en una ubicación con condiciones de temperaturas y humedad que cumpla con los rangos de trabajo de sus componentes.	☒	☐	☐
	Requerimientos de uso			
RFS2	El sistema deberá estar conectado a la corriente eléctrica de forma ininterrumpida para que asegure la alimentación continua.	☒	☐	☐
RFS3	Debe tener Internet por vía WiFi y a través de una red Ethernet para la trasmisión de datos entre los usuarios y el sistema.	☒	☐	☐
RFS4	Debe tener una cámara de video conectada de forma permanente para la captura de imágenes cuando se requiera.	☒	☐	☐
	Requerimientos de interfaces			
RFS5	El sistema debe tener interfaz de: USB, Ethernet y una interfaz para cámara de video CS-2.	☒	☐	☐
	Requerimientos de Modos/Estados			
RFS6	El sistema debe permanecer activo durante el horario de servicio del parqueadero.	☒	☐	☐
	Requerimientos físicos			
RFS7	Las cámaras de video deben estar instaladas de forma que permitan la captura de todos los puestos del parqueadero y el control en una sala dedicada a la administración de los puestos.	☒	☐	☐

Fuente: Autoría propia.

El siguiente paso en el análisis se presenta Tabla 4 que contiene los Requerimientos de Arquitectura donde se establecen las necesidades de hardware, software y eléctricas del sistema. En este caso para la numeración de los requisitos se utiliza la abreviatura RAS (Requisitos de Arquitectura del Sistema). Los requisitos que se describen en esta tabla se utilizan para la selección de los componentes de hardware y software del sistema que se presentan más adelante en este trabajo, por eso la particular importancia de estos requisitos en el análisis de la escogencia de los elementos del diseño.

Tabla 4 *Requerimientos de arquitectura del sistema RAS*

No.	Descripción del requerimiento	Prioridad		
		Alta	Media	Baja
	Requerimientos de Software			
RAS1	Se requiere un sistema operativo que sea de distribución libre y compatible con el sistema embebido.	☒	☐	☐
RAS2	Se requiere software para el procesamiento de imágenes en tiempo real con funciones que permitan determinar la presencia de un vehículo en un puesto del parqueadero.	☒	☐	☐
RAS3	Se requiere software de programación que sea compatible con el software de tratamiento de imágenes.	☒	☐	☐
RAS4	Se requiere software que ejecute una HMI desde la consola de administración del sistema de detección de puestos de parqueo libres.	☒	☐	☐
	Requerimientos de Hardware			
RAS5	Es necesario un sistema embebido que tenga una entrada para conectar una cámara de alta resolución.	☒	☐	☐
RAS6	El sistema embebido debe tener la capacidad de procesamiento de imágenes en tiempo real.	☒	☐	☐
RAS7	El sistema embebido debe tener una conexión a una red Ethernet alámbrica e inalámbrica WiFi para la conexión con Internet.	☒	☐	☐
RAS8	El sistema embebido debe tener una conexión para una pantalla de video.			

RAS9	El sistema embebido debe tener conexión al menos dos puertos USB para conexión de teclado y una cámara de video.	☒	☐	☐
	Requerimientos Eléctricos			
RAS10	El sistema debe tener una conexión directa a la red eléctrica con capacidad para el consumo de todo el sistema con alimentación continua.	☒	☐	☐

Fuente: Autoría propia.

Por último, la Tabla 5 se describen los requerimientos de Stakeholders, recuérdese que un Stakeholder que son los grupos o individuos que tiene algún interés directo de los resultados obtenidos del proyecto desarrollado. Se especifican también los requerimientos funcionales y requerimientos de usuarios. La abreviatura numeradas empleada en esta tabla para la es RSS (Requisitos Stakeholder del Sistema) que facilita la identificación de los requisitos.

Tabla 5 *Requerimientos Stakeholders del sistema RSS*

No.	Descripción del requerimiento	Prioridad		
		Alta	Media	Baja
	Requerimientos de Stakeholders			
RSS1	La ubicación de la cámara debe permitir enfocar todos los puestos del parqueadero donde se detectan los puestos disponibles.	☒	☐	☐
RSS2	Los vehículos deben estar bien parqueados, es decir tiene que estar dentro de los límites de señalización donde este parqueado	☒	☐	☐
	Requerimientos Operacionales			
RSS5	Se requiere comunicación entre el sistema de administración del parqueadero con los usuarios a través de una APP Android en el teléfono móvil.	☒	☐	☐
RSS6	Se requiere al menos una cámara por cada parqueadero de la UNT.	☒	☐	☐
RSS7	Se requiere un operador que administre la solicitud de puestos disponibles y detecte a	☒	☐	☐

	través del sistema un puesto disponible y se lo asigne al solicitante.			
	Requerimientos de Usuarios			
RSS8	Los usuarios deben ubicar su vehículo dentro de los límites de puesto de parqueo.	☒	☐	☐
RSS9	Los usuarios deben tener un móvil inteligente donde puedan instalar la aplicación donde realizaran las solicitudes de los puestos.	☒	☐	☐

Fuente: Autoría propia.

3.5. Selección de Hardware y Software del Sistema

La selección de los componentes de hardware y el software se realiza en base a los requerimientos establecidos haciendo una valoración de cada una de las opciones de los componentes del sistema embebido y los periféricos que lo integran y del software se valoran el sistema operativo y lenguaje de programación y de procesamiento de imágenes. Las opciones que obtengan la mayor valoración serán los utilizados para la implementación del sistema.

3.5.1. Selección del Software

Para la selección del software se realizó una valoración de software de programación para el procesamiento de imágenes y el lenguaje de programación donde se realizarán los programas del procesador embebido, mediante tablas comparativas del cumplimiento de los recursos antes establecidos. En la Tabla 6, se muestra la valoración de opciones para la programación del procesamiento de imágenes. En la primera columna de esta tabla se presentan dos opciones que se encuentran disponibles ambas de fuente abierta que son la librería de funciones OpenCV y la SimpleCV, en la segunda columna se tienen los requerimientos con la abreviatura establecida y en la tercera y última columna se presenta la

valoración total de cada opción. La valoración para cada requisito es cero si no se cumple y uno si se cumple. En esta tabla se comparan los requisitos del software que RAS3, RAS4 y RAS5 de la tabla 6 para valorar las librerías de procesamiento de imágenes a usar.

Tabla 6 *Valoración de opciones de software de procesamiento de imágenes*

Software de tratamiento de imagen	**Requerimiento (ver tabla 4)**		**Valoración total**
	RAS1	RAS2	
OpenCV	1	1	2
SimpleCV	1	0	1

Fuente: Autoría propia.

El resultado de la valoración establece que el SimpleCV a pesar de ser software libre y es compatible con el sistema embebido no asegura tener funciones específicas que permitan la detección de vehículos en los puestos del parqueadero, en cambio OpenCV dispone de más de 500 funciones de reconocimiento de objetos y de fácil adaptación. Los clasificadores de tipo Haar de la librería OpenCV se han utilizado en diversos sistemas de detección, reconocimiento y seguimiento objetos, mejorando el tiempo de respuesta de los sistemas y reduciendo el margen de error de los mismos. Por esta razón se seleccionó la librería abierta para procesamiento de imágenes OpenCV para facilitar la programación del sistema de detección de puestos disponibles en el parqueadero.

Para la selección del software del lenguaje de programación se compararon dos de las opciones más comunes utilizadas Python y Java. Se realizó una tabla 7 con las mismas características de la Tabla 7 cambiando las opciones por estos dos lenguajes.

Tabla 7 *Valoración de opciones de software de procesamiento de imágenes*

Software de tratamiento de imagen	**Requerimiento (ver tabla 4)**		**Valoración total**
	RAS3	RAS4	
Python	1	1	2
Java	1	0	1

Fuente: Autoría propia.

Como se observa en la Tabla 7 donde se comparan los requerimientos RAS3 y RAS4 de la tabla 4, específicamente de software de programación, la valoración Python cumple con ambos requisitos, es decir es compatible con las librerías abiertas de procesamiento de imágenes y permite la creación de una HMI en la consola del administrador del sistema. Por esta razón se escogió el lenguaje de programación Python.

3.5.2 Selección de Hardware

El sistema embebido del sistema debe cumplir los requerimientos establecidos en la tabla 4 y además debe estar basado en los resultados obtenidos en el análisis de la selección del software. Para el sistema embebido se seleccionará un procesador y una cámara que sea de fácil conexión a la cámara y con estos dos dispositivos se tomará la imagen del parqueadero y que será procesada para determinar cuáles puestos están disponibles.

Para la selección del sistema embebido se analizó varias opciones disponibles en el mercado, de las cuales se tomaron las tres que se muestra en la Tabla 8 que son las que más se adaptan a las necesidades del sistema. En esta tabla se compararon los módulos integrados Raspberry, el JaguarBoard y Beaglebone Black que son los más resientes y compiten en costo y en prestaciones. De estas tres opciones la que cumple con todos los requisitos de

hardware es la Raspberry Pi 3 Modelo B y además soporta los sistemas operativos de software libre por lo que se puede programar en Python con las librerías OpenCV para realizar el procesamiento de imágenes necesario.

Tabla 8 *Valoración de opciones del sistema embebido*

Sistema embebido	Requerimiento (ver tabla 4)			Valoración total
	RAS5	RAS6	RAS7	
Raspberry Pi 3 B	1	1	1	3
JaguarBoard	1	0	1	2
Beaglebone Black	1	1	0	2

Fuente: Autoría propia.

El formato de la tabla para la selección del sistema embebido es similar al detallado en la selección del software, con la diferencia de que las opciones son módulos de sistemas embebidos y se consideraron tres los requerimientos en la valoración. Por las razones antes presentadas se seleccionó el Raspberry Pi 3 modelo B.

La selección de la cámara está condicionada por el requisito RAS5 y el resultado de la selección del sistema embebido, gracias a sus puertos USB es posible conectar periféricos tales como cámaras web, dicho esto se ha seleccionado la cámara web Logitech C170 que proporciona los requisitos suficientes que demanda el prototipo a continuación se detallan las especificaciones de esta cámara.

- Módulo de visión de 5MP,
- Ángulo de visión de 175 grados,

- Resolución de imagen fija: 2952 x 1944

- Video: Admite 360 p @ 30 fps,

Esta cámara es de conexión directa al módulo Raspberry a través del puerto USB por lo que cumple también con el requisito funcional de interfaces RFS5.

3.6. Hardware del sistema

En la Figura 37, se presenta un diagrama de bloques que describe el sistema detector de puestos libres basado en un computador Raspberry Pi 3 (RP3) que realizará las funciones del sistema. El RP3 mediante el software dedicado captura una imagen del parqueadero donde se visualizan los puestos de parqueo y es procesada para determinar en tiempo real cuáles puestos están desocupados y asignar mediante dispositivos móvil al usuario que lo solicita.

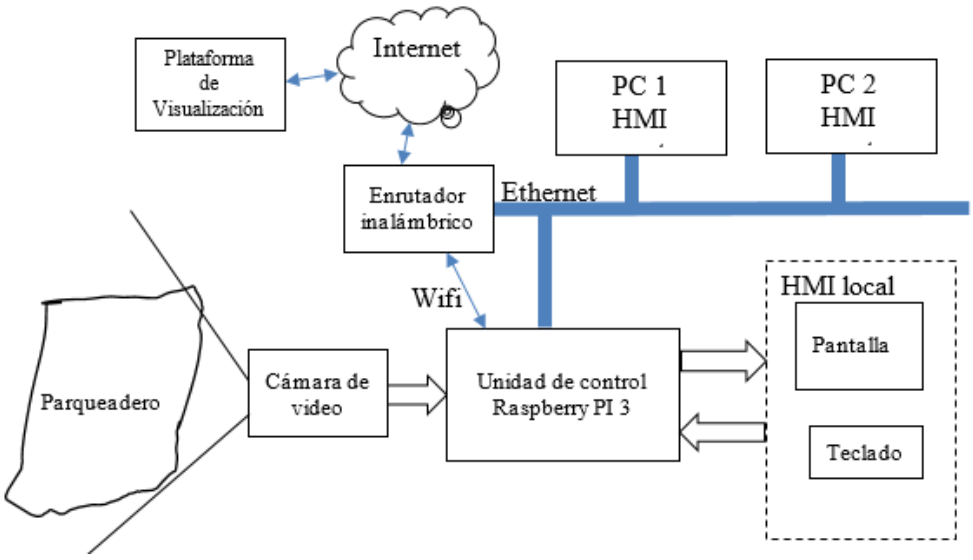


Figura 37 Hardware del detector de puestos disponibles en parqueadero con IoT.

Fuente: Autoría propia.

El Raspberry puede conectarse mediante la red Ethernet y/o una conexión inalámbrica Wifi a uno o varios PC y/o dispositivos móviles con aplicaciones de HMI que le permiten al usuario solicitar la búsqueda de un puesto de parqueo disponible y le se asignado por el sistema. Por otro lado, el RP3 tiene una conexión HDMI donde se puede conectar una pantalla y mediante uno de los puertos USB se puede conectar un teclado, ambos conformarían una HMI local, es decir, estará ubicada en el sistema de control.

3.6.1. Unidad de control

El Raspberry PI 3 es una plataforma ideal para crear un sistema embebido que tiene como funciones básicas la captura y procesamiento de imágenes que exige una velocidad de procesamiento y una capacidad de memoria alta, por otro lado el sistema está basado en cosas de Internet (IoT) por lo que requiere conexión a una red Ethernet y conexión inalámbrica Wifi. El RP3 tiene todos los requerimientos necesarios para conformar unidad de control del sistema de detección de puestos disponibles a un costo relativamente bajo inclusive comparando con versiones anteriores. A continuación, se presentan las especificaciones técnicas del RP3.

3.6.1.1. Procesador

- **Broadcom BCM2387**. De 4 núcleos ARMv8, 2GHz, 64 bits.
- **Wi-Fi** IEEE 802.11 b/g/n. Protocolos WEP, WPA y WPA2
- **Bluetooth 4.1.** con un el rango máximo de 50 metros.

- **GPU** Co-procesador Multimedia Dual Core Video Core IV. Con Open GL ES 2.0, hardware acelerado Open VG y decodificación de alta perfil 1080p30 H.264. Capacidad un Gpixel/s, 1.5Gtexel/s o 24GFLOPs con filtrado de texturas y estructura DMA.

- **Memoria:** un GB LPDDR2.

- **Sistema operativo**: Software de arranque desde la tarjeta Micro SD, pueden ser alguna versión de Linux o Windows 10 IoT.

- **Dimensiones**: 85 x 56 x 17mm.

- **Alimentación:** 5 V, 2.5 A por conector micro USB.

3.5.1.2. Conectores de periféricos

- **Puerto Ethernet** 10/100 Base T Ethernet.

- **Salida de Video** HDMI (1.3 & 1.4) PAL y NTSC

- **Salida de Audio** Audio Output RCA de 3.5mm.

- **USB 2.0:** 2 conectores dobles de USB, es decir 4 puertos x USB 2.0.

- **Expansión de 40 pines.** Tipo cinta 2x20 de 2,54mm. Proporciona 27 pines GPIO, así como líneas de alimentación de +3.3 V, +5 V y GND.

- **Cámara MIPI.** De 15 pines Interfaz Serial (CSI-2).

- **Pantalla Interfaz serial (DSI).** Conector de cable flexible plano de 15 vías con dos líneas de datos y una línea de reloj.

- **Tarjeta de memoria.** Ranura Empujar/tirar Micro SDIO

3.5.1.3. Accesorios mínimos necesarios para el diseño del prototipo con el RP3

- Tarjeta μSD de 8 GB clase 4.

- Fuente con conector micro-USB 5V, 2.5 A.

- Monitor o TV con conexión HDMI

- Cable HDMI

- Teclado y ratón con cable de conexión USB 2.0 (no inalámbricos).

- Cable red Ethernet RJ-45 para acceso a Internet

- Módulo Cámara para Raspberry Pi 3 con cable de 15 Pin.

3.6.2. Conexión de cámaras de video

Actualmente en el mercado existen una gran variedad de cámaras con prestaciones que aumentan día a día por el desarrollo tecnológico y en consecuencia hay una disminución de costos de equipos que tiene muy buenas prestaciones. Para aplicaciones de visión artificial donde se realiza procesamiento de las imágenes capturadas con cámaras, la resolución depende de la aplicación donde se esté usando. Por ejemplo, cuando se quieren extraer características de objetos pequeños con respecto al área de total donde se está realizando la

búsqueda se requieren cámaras con alta resolución con la posibilidad de hacer zoom en el área donde se encuentran los objetos con una resolución que permita la captura de imágenes que puedan ser procesadas, es decir representadas con matrices de pixeles de un byte en escala de grises o de tres bytes en color con formato RGB. Otra opción es tener cámaras memos exigentes en cuanto a sus prestaciones utilizando un sistema de control de posición para hacer un barrido de la zona de búsqueda, en este caso hay que considerar los costos adicionales y el mantenimiento de servo-control.

En la aplicación final de este trabajo, en el caso de parqueaderos que no son techados, la cámara y su instalación deben tener la robustez necesaria para soportar las condiciones ambientales, como es el caso de las cámaras vigilancia que se utilizan en exteriores. En el prototipo propuesto se utilizará una cámara web, de bajo costo y conexión directa a la tarjeta RP3 por la interfaz USB que permitirá realizar las pruebas necesarias para verificar la funcionabilidad del sistema de detección de puestos libres en el parqueadero. En la Figura 38, se muestra una imagen de la cámara web Logitech C170.

Figura 38 Cámara Web Logitech C170.

Fuente: https://support.logitech.com/es_es/product/webcam-c170

Para una implementación total del sistema a todos los parqueaderos de la Universidad se recomienda utilizar una cámara que soporte las condiciones climáticas presentes en el exterior, además de que debe tener una gran apertura focal para cubrir el mayor número de espacios de estacionamiento, también deberá tener visión nocturna para que el sistema pueda trabajar en horarios nocturnos, dicho esto se recomienda la cámara Logitech Circle 2, a continuación en la Figura 39 se muestra la cámara antes mencionada

Figura 39 Cámara Web Logitech Circle 2.

Fuente: https://www.logitech.com/en-us/product/circle-2-home-security-camera

Características

- Full HD 1080p

- Visión Nocturna

- Campo visual 180°

-Resistente al exterior

3.6.3. HMI local

La HMI local está conformada por una pantalla donde el usuario del sistema puede monitorear y verificar los resultados de la búsqueda realizada por el software de procesamiento de imágenes, y un teclado para realizar las operaciones con los usuarios. El Raspberry pi tiene un puerto HDMI para conexión de la pantalla y el ratón se conecta a uno de los puertos USB.

3.6.3.1. Conexión pantalla al puerto HDMI

El puerto HDMI de la Raspberry permite la conexión de una pantalla de televisión o un monitor. Algunos monitores y televisores modernos tienen puertos HDMI con los que se realiza la conexión mediante un cable estándar HMDI, que en ambos extremos tiene un conector que se muestra en la Figura 40 que se presenta a continuación.

Figura 40 Conector estándar de un cable HDMI.

Fuente: Autoría propia

En el caso de monitores que solo tienen puerto DVI se puede usar un cable de adaptador de HDMI a DVI, En la Figura 41, se observan los conectores de este cable adaptador. El estándar de conexión de monitores DVI no admite audio.

Figura 41 Cable adaptador de estándar de televisión HDMI a monitor DVI.

Fuente: Autoría propia.

Otra alternativa son los televisores o monitores VGA, en este caso se puede utilizar un adaptador de HDMI a VGA y se debe considerar que no es compatible con audio. En la Figura 42, se muestra el cable adaptador mencionado. En cada caso de los anteriores se requieren manejadores que están incluidos en el software de configuración de los Raspberry.

Figura 42 Cable adaptador de HDMI a VGA.

Fuente: Autoría propia.

Los Raspberry Pi B + y posterior combinan la salida de audio y el compuesto en la misma toma de 3.5 mm. Esto requiere un tipo particular de cable, con audio en la punta, audio en el anillo 1, tierra en el anillo 2 y video en la manga. Esto es lo mismo que los cables utilizados en Zune y en dispositivos Apple. En la Figura 43, se presenta una imagen de este conector.

Figura 43 Conector de 3.5 mm para de audio y video de la Raspberry.

Fuente: Autoría propia.

3.6.3.2. Conexión del teclado y el ratón por puertos USB

Actualmente existe una gran variedad de opciones de teclados y ratones en el mercado incluyendo teclados con alfombras integradas con conexión por cable USB o inalámbrica con transmisor receptor USB. Todas estas opciones pueden ser usadas en la Raspberry a través de sus puertos USB.

3.6.4 Conexión a Internet

La conexión a Internet se puede realizar de dos formas cableada o inalámbrica. En la Figura 44, se muestran ambas opciones, en la conexión inalámbrica las cámaras de los parqueaderos pueden ser conectadas mediante una conexión IP a través de wifi y a la red Ethernet. La conexión cableada presenta dos alternativas, una conectando las cámaras con cables UTP a la red Ethernet o directamente al procesador Raspberry Pi.

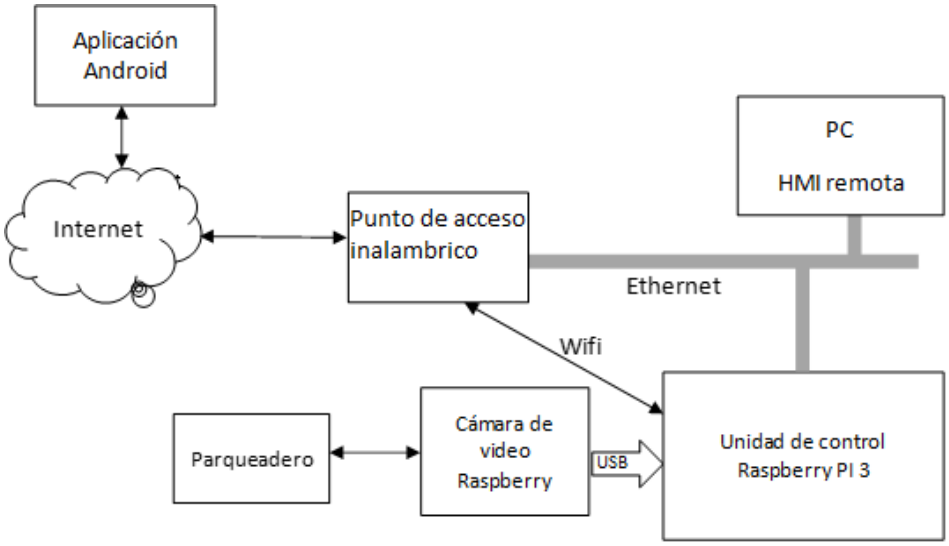


Figura 44 Conexión a Internet inalámbrica, cableada en red Ethernet y conexión USB

Fuente: Autoría propia.

Para el prototipo del sistema de detección de puestos libres se utiliza la cámara Web Logitech C170 conexión directa al puerto USB. En este caso la cámara está conectada al procesador, es decir, el procesador y la cámara están juntas en el área del parqueadero y la conexión por Internet se haría a través de una red Ethernet, que puede ser cableada o inalámbrica. Es de notar que en este caso se puede hacer una expansión del sistema colocando en cada parqueadero un dispositivo como el prototipo diseñado y conectarlos en red para centralizar la información en un PC donde se administre los puestos de todos los parqueaderos recibiendo información de los módulos ubicados en cada uno de los parqueaderos.

3.7. Software del sistema

El software para del prototipo de detección de parqueos disponibles está constituido por entornos diferentes e independientes. Uno es el software del Raspberry y el otro es la aplicación en Android para aplicaciones de telefonía móvil. En la Figura 45, se presenta el

diagrama de flujo del algoritmo del programa para detección de puestos disponible en un parqueadero con visión artificial

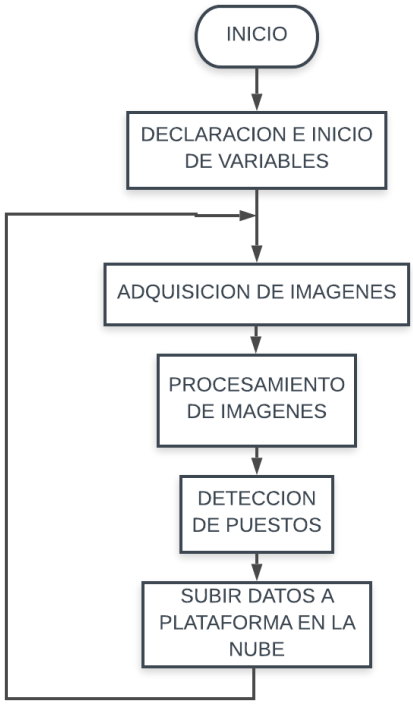


Figura 45 Diagrama de flujo del algoritmo en Raspberry para detección de puestos disponibles en un parqueadero con visión artificial.

Fuente: Autoría propia.

Para el inicio del algoritmo se comienza con la declaración de variables, que en este caso serán los puestos de estacionamiento, seguido a esta se inicia la cámara que realizará la lectura de imágenes del parqueadero, cuando las imágenes sean capturas se realiza un preprocesamiento de estas que permitirán determinar la disponibilidad de un lugar de estacionamiento. Una vez que se detecta el estado de un sitio de parqueo se procede a enviar los datos a una plataforma de visualización para los usuarios.

3.8. Distribución de Zonas y Ubicación de las Cámaras

En este tipo de proyecto sobre visión artificial además de los requerimientos del sistema, es importante considerar el software donde se va a realizar la programación de la aplicación y que los dispositivos que van a interactuar con la aplicación sean compatibles con el software.

La distribución de zonas permite abarcar todo el espacio físico e identificar lugares de interés como: ingresos, puertas, pasillos, etc. De este modo se podrá definir las necesidades de cada sector, a fin de que con el menor número de cámaras se cubra todo el parqueadero.

Dado que antes de realizar una distribución de zonas se eligió previamente un tipo de cámara, se debe establecer un método práctico, que en base a pruebas permita determinar la ubicación de las cámaras en puntos con visibilidad privilegiada, además en base al alcance de las cámaras se pueda realizar la distribución de zonas. Se debe optimizar los recursos tratando de que por cada zona se use una cámara mientras sea posible.

Al ubicar las cámaras se debe considerar: que no sean accesibles a cualquier persona, facilidades de instalación, alimentación de energía. Se debe hacer constar el alcance de las cámaras en un plano y además el conjunto de cámaras deben cubrir el lugar en su totalidad.

El ángulo de visión está relacionado con la zona de cobertura que se observan desde las cámaras, este parámetro varía de acuerdo al modelo, longitud focal y distorsión del lente.

3.8.1 Ubicación de las cámaras

En este apartado se definen los lugares en los que deben ir ubicados las cámaras para lo cual se toma en cuenta los parámetros previamente establecidos en el análisis de requerimientos, y de esta manera lograr que es sistema funcione adecuadamente.

Los lugares donde serán ubicados cada uno de las cámaras son los estacionamientos, los mismos que se encuentran señalizados en todos los parqueaderos del campus universitario, cada cámara ira acompañada de su placa base para adquirir y procesar las imágenes recolectadas que en este caso es la Raspberry Pi.

Los parámetros más importantes a tomar en cuenta para la ubicación de las cámaras, es que estos requieren tener acceso a Internet ya se mediante cable o conexión inalámbrica, adicional a esto deben tener conexión eléctrica y deberán ser ubicadas a una altura especial que permita abarcar la mayor distancia posible de los parqueaderos. En la Figura 46 se puede observar la distribución de cámaras por cada parqueadero.

Figura 46 Ubicación de las cámaras

Fuente: Autoría propia.

Luego de establecer el lugar de ubicación de las cámaras, a continuación, en la Tabla 9 se detallan el número de cámaras que se necesita para cada parqueadero.

Tabla 9 *Numero de cámaras necesarias por cada parqueadero*

Parqueadero	**Ubicación**	**Nombre asignado**	**Numero de cámaras necesarias**
Parqueadero 1	Bar Universitario y FACAE	Parqueadero Bar Universitario y FACAE	1

Parqueadero 2	FECYT	Parqueadero FECYT	1
Parqueadero 3	Canchas UTN	Parqueadero canchas UTN	2
Parqueadero 4	Administración Central	Parqueadero Administración Central	1
Parqueadero 5	FICA –FICAYA	Parqueadero FICA – FICAYA	2
Parqueadero 6	Educación Física	Parqueadero Educación Física y CAI	2
Parqueadero 7	Piscina UTN	Parqueadero Piscina UTN	1
		TOTAL:	10

Fuente: Autoría propia.

Cabe recalcar que en algunos parqueaderos se ha propuesto más de una cámara debido al tamaño que posee y al número de aparcamientos que posee.

En la siguiente Figura 47 se ubican los dispositivos de visualización en donde se podrá observar la información de puesto libres y ocupados obtenida mediante las cámaras.

Figura 47 Ubicación de las pantallas informativas

Fuente: Autoría

Capitulo IV

Una vez concluida la fase de diseño se procede a realizar el desarrollo e implementación del prototipo del sistema, el mismo que permitirá verificar el funcionamiento del diseño propuesto, para lo cual se lo aplicara en un área limitada donde se realizaran las pruebas respectivas.

4.1. Desarrollo Del Software Del Prototipo

A continuación, se describen los procedimientos y los elementos que forman parte del sistema, para llevar a cabo la implementación de este prototipo se han tomado dos aspectos fundamentales de software que son: una aplicación para procesamiento de imágenes y una aplicación para la visualización de los datos obtenidos, para el desarrollo de este apartado se ha utilizado un video de prueba grabado previamente.

4.1.1 Instalación de Raspbian y OpenCV

El prototipo fue realizado en la placa Raspberry Pi puesto que es necesario instalar un Sistema Operativo para que pueda ser utilizada como un mini computador. La guía de instalación se puede observar en el ANEXO 1. Además, se debe instalar el software de procesamiento de imágenes OpenCV el cual se especifica un manual en el ANEXO 2.

En la Figura 48 se puede observar la pantalla de inicio de Raspbian.

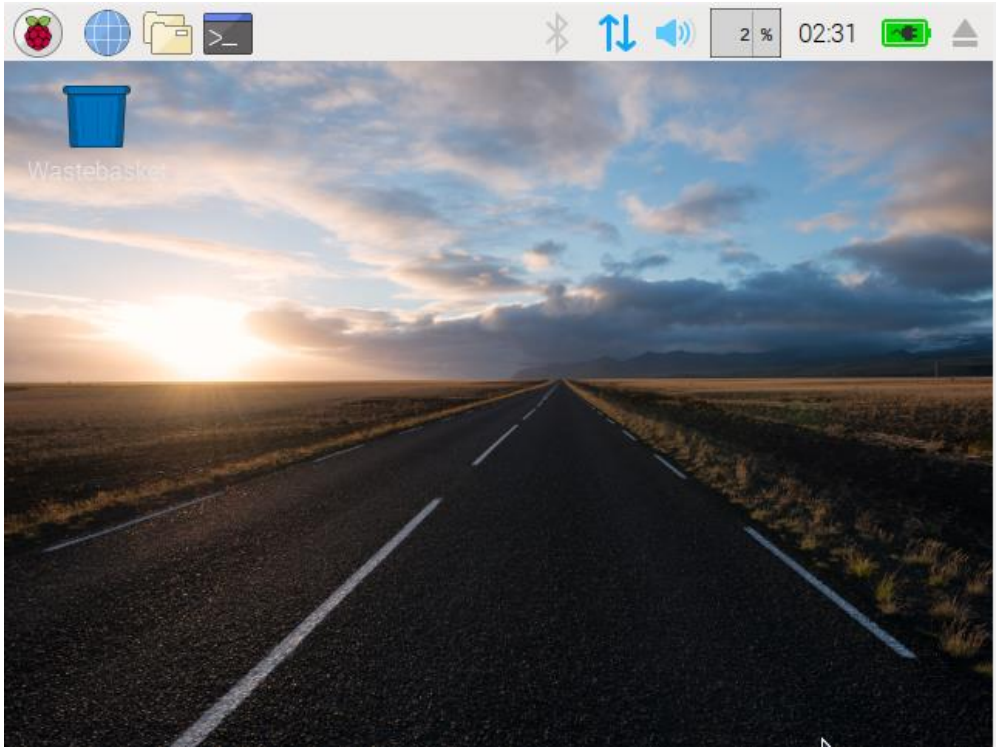


Figura 48 Pantalla de inicio Raspbian

Fuente: Autoría propia.

4.1.2 Adquisición de Imágenes

Como primer paso para el desarrollo del prototipo es la obtención de imágenes del parqueadero en el que va a implementar el sistema, dado que la toma de imágenes será en un lugar externo se tendrá en cuenta las variaciones de luz presentes, para que la detección de plazas sea exitosa las imágenes deben poseer buena iluminación, no deben tener cambios de escala, ni tampoco debe existir oclusión. Las imágenes capturadas serán RGB

Gracias a OpenCV se puede capturar la transmisión de video en vivo mediante el uso de una interfaz muy simple. Para realizar la captura es necesario crear una función denominada VideoCapture, su parámetro será el nombre de un archivo de video o el número

de las cámaras que están conectadas a la placa que generalmente empieza desde 0 (OpenCV , 2018).

A continuación, se muestra la función completa utilizada para la captura de video:

cap = cv2.VideoCapture(*número de la cámara o nombre del archivo de video***)**

Para evitar errores en la captura de video creamos un ciclo while, su parámetro será la función cap.isOpened(), esta devuelve un valor verdadero si el video se está capturando de manera correcta.

Para mostrar las imágenes se usará la función cv2.imshow(), su parámetro será el nombre que llevará la ventana y la variable en la que se está almacenando las capturas de imagen, a continuación, se muestra un ejemplo del uso de esta función:

cv2.imshow(‘nombre de la ventana**’, *variable de video*)**

4.1.3 Regiones de Interés (ROI)

Los denominados ROI son básicamente un subconjunto de una imagen que ha sido identificado para un propósito en particular, por ejemplo, un ROI puede ser encontrar los pixeles que representen un objeto como una persona, un auto, etc (MathWorks, 2018).

Los ROI pueden tener diferentes formas geométricas, pero en este caso tendrá la forma de un cuadrilátero que se asemeja a la forma que posee un lugar de estacionamiento en un parqueadero.

Como se observa en la Figura 49 se extrae solo una parte de la imagen original que es la parte de interés, esta será tratada y estudiada con un fin.

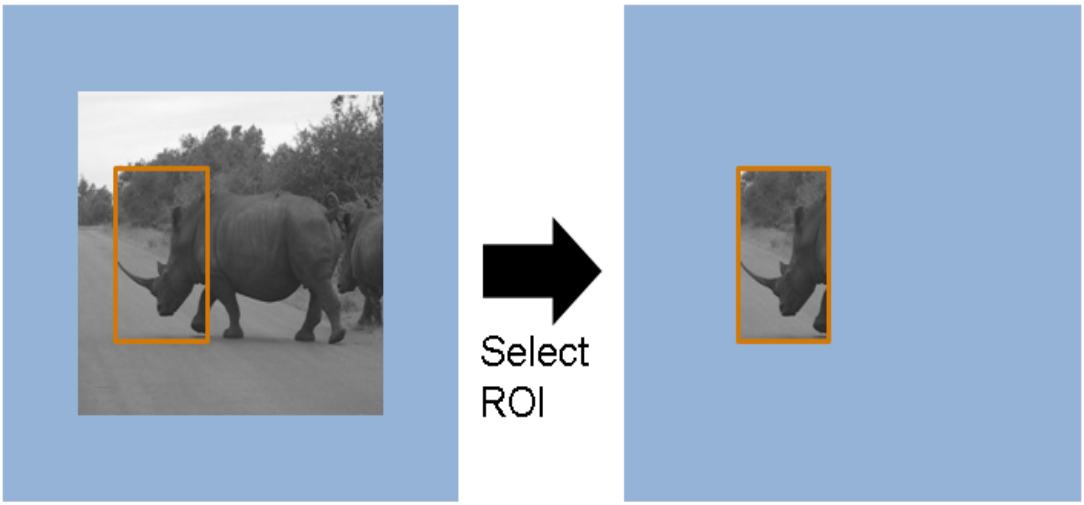


Figura 49 Ejemplo ROI(Region of Interet)

Fuente: https://www.mathworks.com/help/visionhdl/ref/roiselector.html

La importancia del ROI en el desarrollo del prototipo radica en que el objetivo es determinar si una plaza de estacionamiento está libre u ocupada, en efecto un ROI es ideal en este caso, los ROI en el prototipo serán las plazas de estacionamiento que sean visibles en la captura de video.

4.1.3.1 Asignación de Regiones de interés.

Para asignar los ROI a la captura de video se utilizará un formato de serialización de datos conocido como YAML (YAML Ain't Markup Language), este es un lenguaje más comprensible para personas, se usa particularmente para archivos de configuración, traducciones y representación de información. La importancia de este formato reside en que es amigable, fácil y permite el mapeo de estructura de datos (YAML, 2018).

Se usará YAML como archivo de configuración para los cuadriláteros, en el archivo de configuración se incluirán las coordenadas tanto en X como en Y que representarán los vértices del ROI.

Mediante OpenCV se puede obtener las coordenadas de manera muy fácil, basta con ubicar el puntero del mouse en la ventana generada por OpenCV cuando se muestra la captura de video, en la parte inferior se muestra los valores en X y Y en la actual posición del puntero.

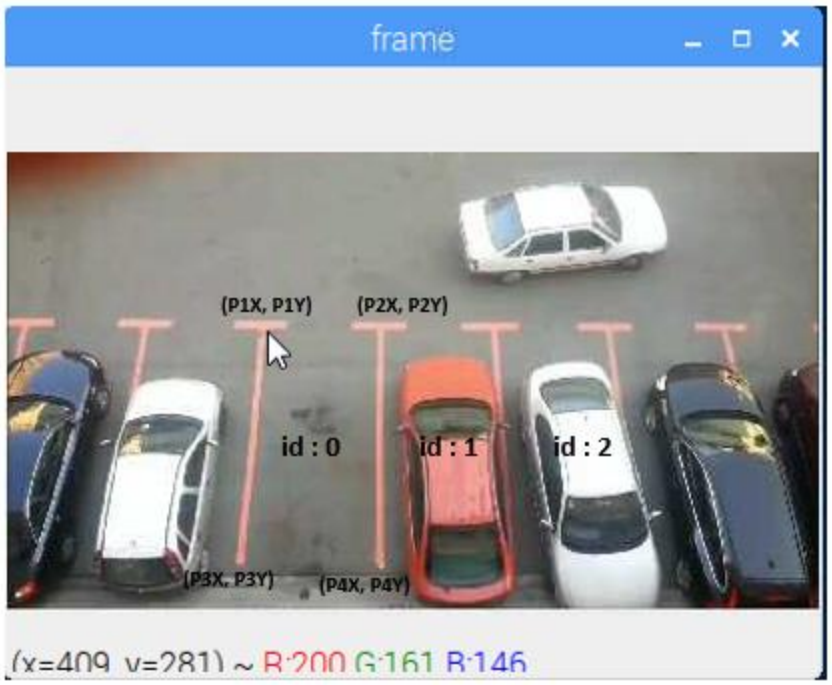
En la Figura 50 se muestra un ejemplo de la adquisición de las coordenadas mediante la ubicación del puntero en los vértices de los ROI.

Figura 50 Toma de coordenadas de vértices del ROI

Fuente: Autoría propia.

Una vez que se tiene las coordenadas de los 4 vértices se procede a estructurar el archivo YAML, en la Figura 51 se muestra un ejemplo en el cual se detalla la estructura.

a)



b)

```
-
    id: 0
    points: [[P1X,P1Y],[P2X,P2Y],[P3X,P3Y],[P4X,P4Y]]
```

Figura 51 a) Vértices ROI. b) Estructura de configuración YAML

Fuente: Autoría propia.

El valor id: 0 corresponde al número del puesto de parqueo, este puede ir desde 0 hasta el número de puestos que sean visibles por el alcance de la cámara, en la variable points se agrega las coordenadas de los vértices previamente adquiridas. Para añadir más ROI se agrega otro id con el numero siguiente y los valores de los vértices.

4.1.4 Sustracción de Fondo

La sustracción de fondo es un método ampliamente utilizado para detectar objetos en movimiento en videos de cámaras estáticas. El fundamento del enfoque es el de detectar los objetos en movimiento a partir de la diferencia entre el marco actual y el marco de referencia, a menudo llamado "imagen de fondo" o "modelo de fondo". La imagen de fondo debe ser una representación de la escena sin objetos en movimiento y debe actualizarse periódicamente para adaptarse a las distintas condiciones de luminancia y los ajustes de geometría (OpenCV , 2018)). En la Figura 52 se muestra un ejemplo de la sustracción de fondo de fondo.

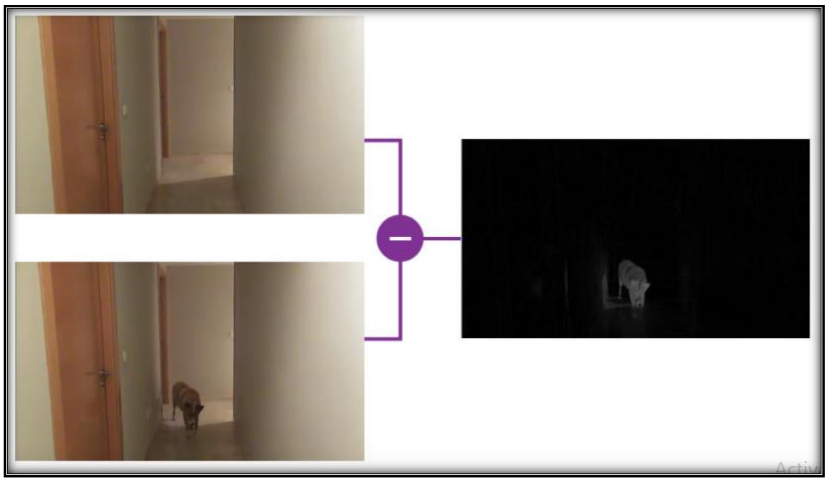


Figura 52 Sustracción de fondo para identificar objetos.

Fuente: https://programarfacil.com/blog/vision-artificial/deteccion-de-movimiento-con-opencv-python/

Como se observa en la Figura 52 el resultado de la sustracción de fondo muestra en este caso al perro que es el objeto que ha sido identificado mediante esta técnica. La aplicación de este método en el prototipo ayudará a identificar objetos que se sobrepongan en las plazas de estacionamiento.

Para obtener la sustracción de fondo previamente se debe realizar una serie de pasos:

- Conversión a escala de grises

-Eliminación del ruido

-Suavizado de la imagen

4.1.4.1 Conversión a escala de grises

Una de las razones para usar imágenes en escala de grises en lugar de imágenes a todo color es simplificar el espacio de color tridimensional (R, G y B) a una representación dimensional única, es decir, monocromática. Este hecho es importante porque reduce el costo computacional y simplifica los algoritmos (Macedo, Melo, & Kelner, 2015).

OpenCV proporciona varias funciones de conversiones de color mediante el método cvtColor, básicamente se basa en la ecuación de luminancia para obtener este efecto, a continuación, se muestra la ecuación antes mencionada:

$$\text{RGB [A] to gray: } Y \leftarrow 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B$$

El método ctvColor permite realizar las siguientes transformaciones de color:

-BGR2GRAY

-RGB2GRAY

-GRAY2BGR

-GRAY2RGB

A continuación, en la Figura 53 se muestra un ejemplo de la conversión a escala de grises mediante el uso de la siguiente función:

```
cv2.ctvColor(frame, cv2.COLOR_BGR2GRAY)
```

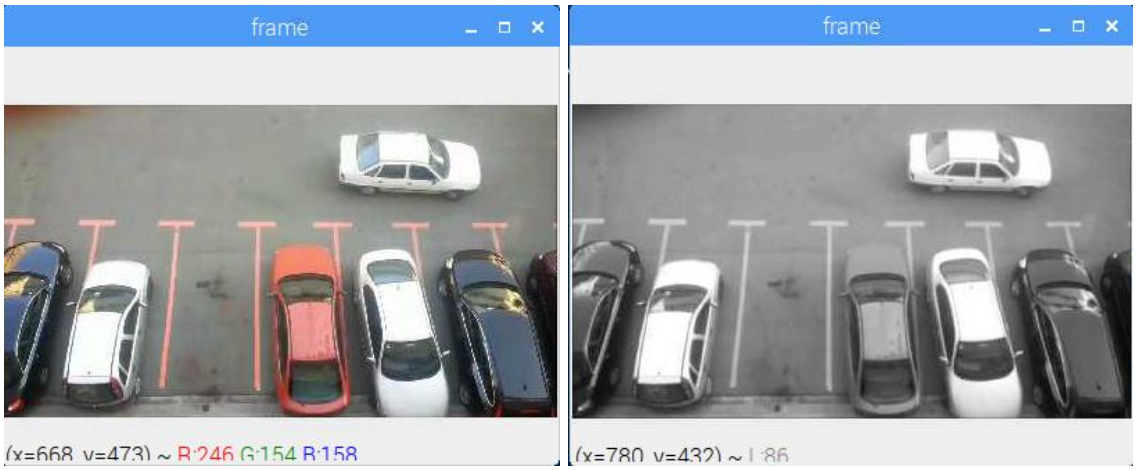


Figura 53 Conversión a escala de grises

Fuente: Autoría propia.

4.1.4.2 Suavizado de Imagen

El objetivo principal del suavizado de imágenes es eliminar el ruido en las imágenes digitales. La calidad de la imagen es un factor importante para el punto de vista de la visión humana (Goyal, Bijalwan, & Chowdhury, 2012). El suavizado de imágenes es un método para mejorar la calidad de las imágenes, la imagen generalmente tiene un ruido que no se elimina fácilmente en el procesamiento de la imagen. La calidad de la imagen se ve afectada por la presencia de ruido.

OpenCV proporciona varias técnicas para obtener el suavizado de imagen que se mencionan a continuación:

-Averaging

- Gaussian Blurring

- Median Blurring

- Bilateral Filtering

Para este caso se utilizará el filtro de suavizado gaussiano o Gaussian Blurring, este es el filtro más ventajoso, pero no el más rápido, además proporciona la función de quitar el ruido de alta frecuencia para una secuencia de imágenes (smoot), esto significa que es muy ideal para casos en los cuales existe cambios de pixeles muy rápido.

En la Figura 54 se muestra un ejemplo del suavizado de imagen utilizando el método Gaussian Blurring mediante la siguiente función:

cv2.GaussianBlur(frame, (5,5), 3)

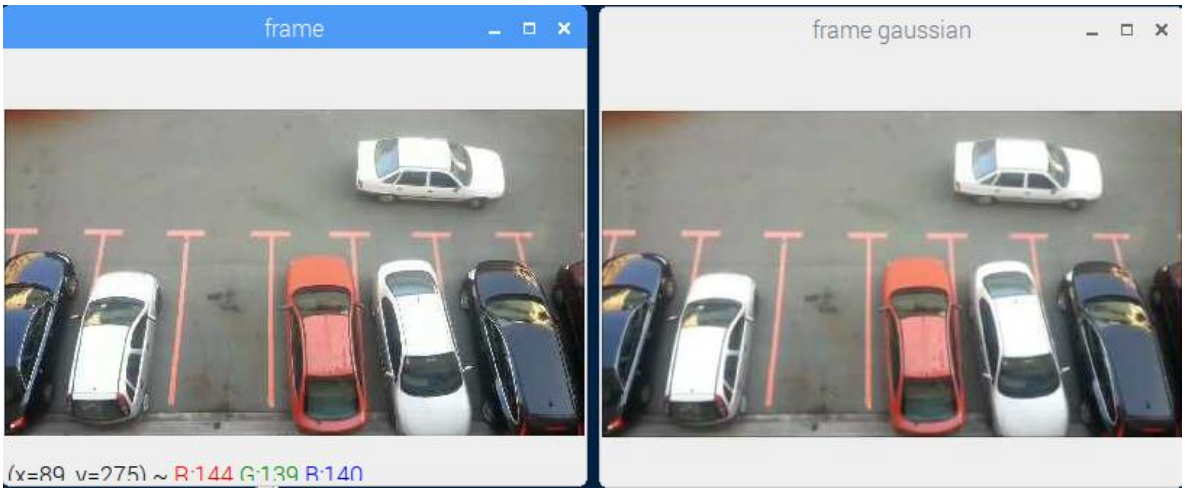


Figura 54 Suavizado de imagen mediante Gaussian Blurring

Fuente: Autoría propia.

4.1.4.3 Resta de fondo

Una vez que se ha eliminado el ruido mediante la conversión a escala de grises y el suavizado de imagen se procede a realizar la resta de fondo, como ya se ha explicado este proceso consiste en realizar la operación de la resta entre dos imágenes, la primera será la

imagen de fondo sin ningún objeto presente y la segunda será la imagen que se esté capturando y que ya ha sido tratada.

OpenCV proporciona la función cv2.absdiff(), sus parámetros son la imagen de fondo y la imagen que se esté capturando para detectar posibles objetos, esta función permite restar imágenes y obtener valores positivos de cada pixel debido a que es posible obtener valores negativos. A continuación, se muestra un ejemplo de la función resta:

resta=cv2.absdiff(fondo, imagen tratada)

Como se muestra en la Figura 55 mediante la resta de fondo se ha identificado el auto en movimiento.

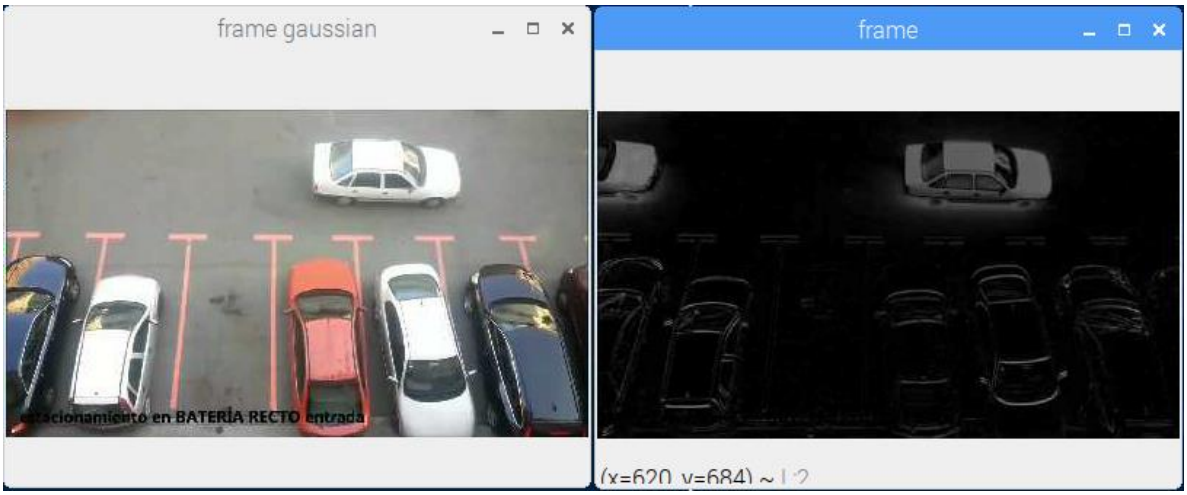


Figura 55 Resta de fondo

Fuente: Autoría propia.

4.1.5 Detección de Plazas de Aparcamiento

Una vez que se ha tratado la imagen se inicia con la etapa de detección que es el principal objetivo del prototipo, para este proceso se realizará una serie de fases que son las siguientes:

-Detección de Bordes

-Algoritmo de detección

4.1.5.1 Detección de Bordes

El proceso de detección de bordes sirve para simplificar el análisis de imágenes al reducir drásticamente la cantidad de datos a procesar y al mismo tiempo preservar información estructural útil sobre los límites de los objetos. La detección de bordes apunta a identificar puntos en una imagen digital en los que el brillo de la imagen cambia bruscamente. Esta técnica trata principalmente de extraer bordes en una imagen identificando píxeles donde la variación de intensidad es muy alta (Savant, 2014). Los bordes se utilizan para medir el tamaño de los objetos en una imagen; para aislar objetos particulares de su fondo; para reconocer o clasificar objetos. Como se observa en el ejemplo de la Figura 56 se puede ver como se detallan los bordes característicos del objeto.

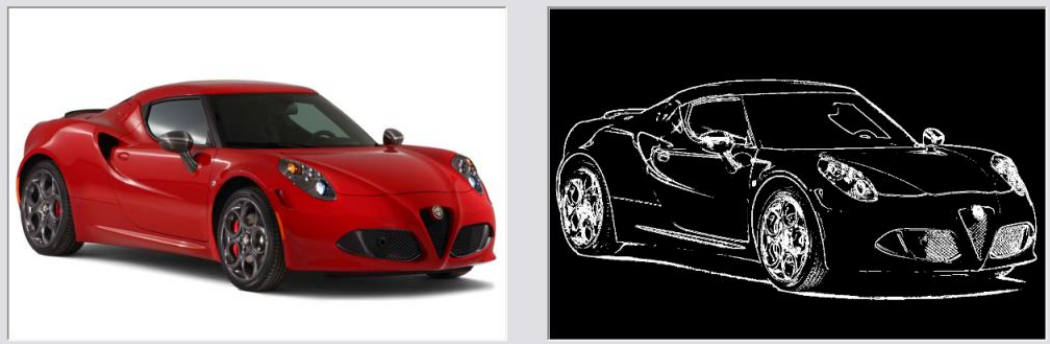


Figura 56 Detección de bordes

Fuente: https://www.codeproject.com/Articles/990093/Image-Processing-for-Dummies-with-C-and-GDI-Part

En general, las técnicas de detección de bordes se agrupan en dos categorías:

-Gradiente (Métodos usando la primera derivada)

-Laplaciano (Métodos usando la primera derivada)

Para este caso utilizáremos el operador Laplaciano que utiliza la segunda derivada para la detección de bordes, puesto que las imágenes que se están tratando son en 2D se debe considerar la derivada en ambas direcciones, esto hace que el operador Laplaciano sea útil en estos casos (OpenCV , 2018).

OpenCV proporciona el operador Laplaciano mediante la función Laplacian(), hay que destacar que este operador utiliza internamente el operador de Sobel para ejecutar este cálculo, este operador pertenece a la categoría de los gradientes.

El operador Laplaciano se define por la siguiente ecuación:

$$Laplace(f) = \frac{\partial^2 F}{\partial X^2} + \frac{\partial^2 F}{\partial y^2}$$

A continuación, en la Figura 57 se muestra un ejemplo de la detección de bordes mediante el uso del operador Laplaciano ejecutando la siguiente función:

laplacian = cv2.Laplacian(imagen tratada, cv2.CV_64F)

Los parámetros de la función Laplacian() son la imagen que se ha estado tratando y el termino cv2.CV_64F hace referencia al tipo de imagen que se está trabajando, ya que es una imagen en escala de grises, solo posee un canal por lo cual se utiliza este término para especificarlo.

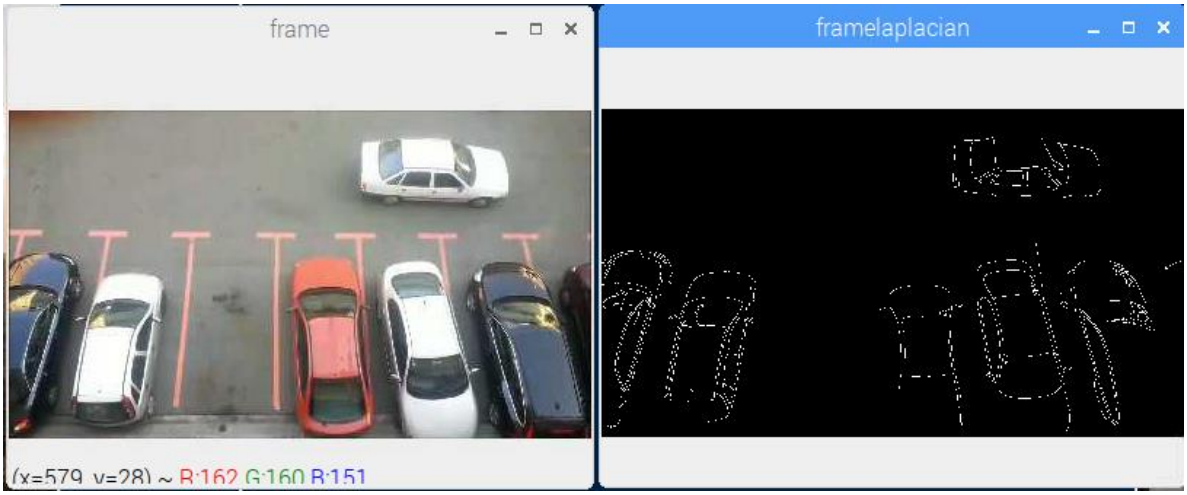


Figura 57 Detección de bordes mediante Operador Laplaciano

Fuente: Autoría propia

4.1.5.2 Algoritmo de detección

El siguiente punto es detectar si una plaza de estacionamiento está libre u ocupada, básicamente el algoritmo se basa en detectar la presencia de bordes, es decir que se obtendrá ciertos valores cuando la plaza de estacionamiento este vacío y se obtendrá otros cuando este ocupada por un auto.

Primeramente, debemos obtener valores numéricos para poder establecer un umbral, hay que tener en cuenta que una imagen es una matriz con varios valores, dicho esto se debe realizar un proceso que permita obtener un único valor numérico.

Mediante la librería matemática Numpy se puede obtener la media aritmética de una matriz utilizando la función mean(). Además se utilizó la función abs() para obtener valores absolutos en el cálculo de la media aritmética. A, continuación se muestra un ejemplo del uso de estas funciones.

```
delta = np.mean(np.abs(laplacian))
```

La media aritmética a obtener será la de la variable en donde se ha almacenado el cálculo del operador Laplaciano que para este caso es la variable laplacian. Esta media aritmética será almacenada en la variable delta, hay que destacar que este cálculo se realiza para cada ROI que se ha creado.

A continuación, en la Figura 58 se muestran los valores de las medias aritméticas que cada ROI arroja.

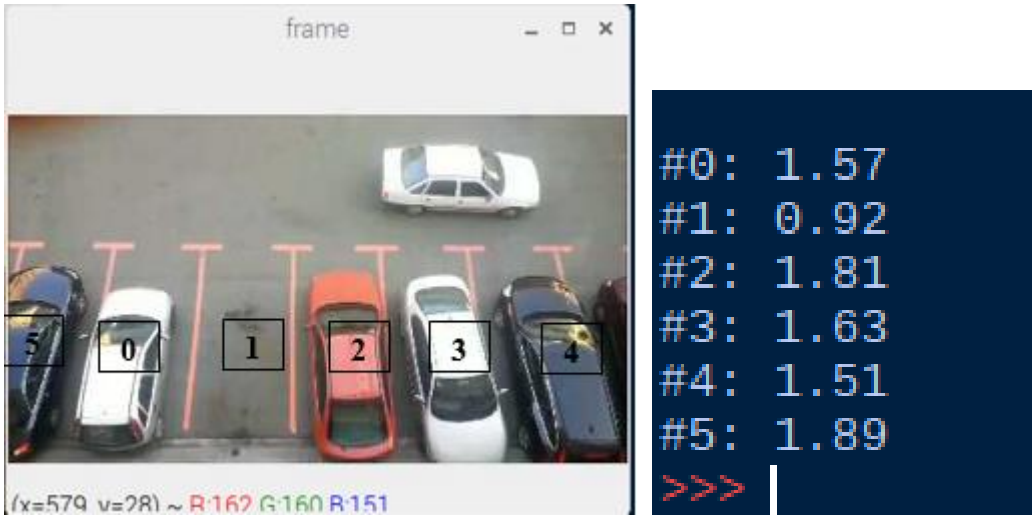


Figura 58 Calculo de la media aritmética del operador Laplaciano de cada ROI

Fuente: Autoría propia.

Como se observa en la figura 58 los valores obtenidos en los ROI con vehículos rondan en valores mayores que 1 a diferencia del valor obtenido en el ROI en el cual no hay vehículo presente se obtiene un valor menor que 1.

Dicho esto, se puede establecer un umbral de valor 1, esto quiere decir que cuando exista valores que sobrepasen el umbral existirá presencia de vehículos y si este valor es menor al umbral indicara que el puesto está vacío.

Para realizar este proceso se utilizará una comparación básica que se muestra en la siguiente línea de código:

status = delta < 1

Se almacena el estado en la variable status, esta comparación arrojara un valor lógico TRUE si delta es menor que 1 y un valor FALSE si delta es mayor que 1, en la Figura 59 se puede observar un ejemplo de este proceso.

Figura 59 Detección de bordes Detección de plazas

Fuente: Autoría propia.

A manera de demostración se puede dibujar los ROI y asignarles color dependiendo de su estado, en la Figura 60 se puede observar los ROI asignados a cada plaza de estacionar, se ha asignado el color verde para las plazas libre y el color rojo para las que están ocupadas.

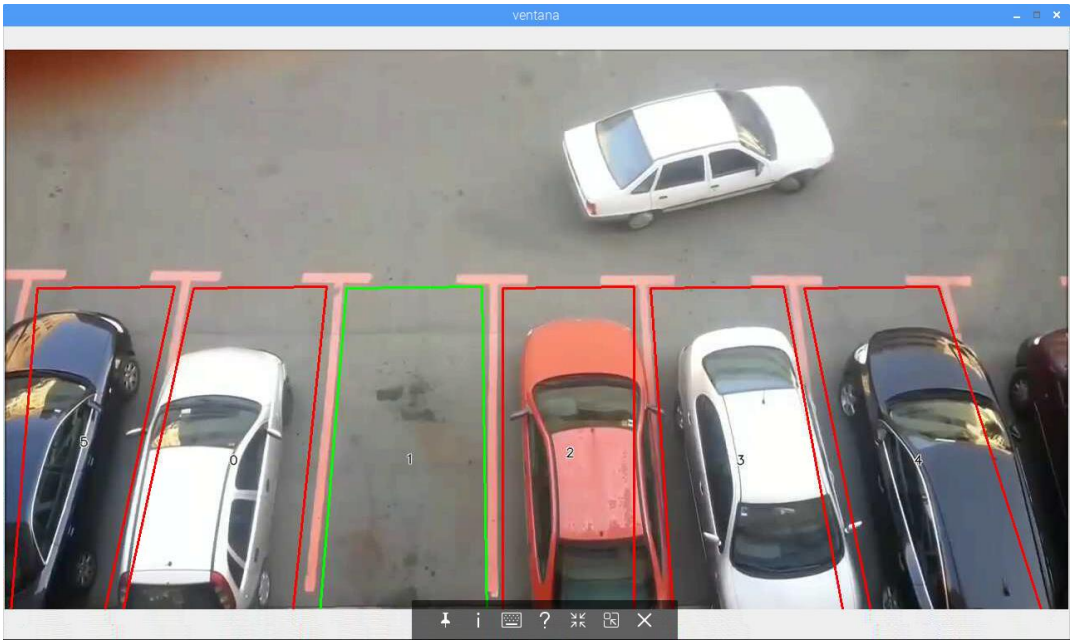


Figura 60 Trazo de ROIs dependiendo de su estado

Fuente: Autoría propia.

Una vez realizado el proceso de detección de plazas es posible determinar el número total de lugares libres existentes en el estacionamiento, dicho esto el siguiente paso es la visualización de los datos en una plataforma amigable al usuario.

4.1.6 Plataforma de Visualización

La siguiente fase de nuestro prototipo es la visualización de los datos obtenidos en el proceso de detección, dicho esto se ha utilizado la plataforma Ubidots que es muy conocida dentro de la industria de hardware, software E ingeniería integrada dentro del ecosistema de IoT.

La ventaja de esta plataforma es su fácil uso y manejo en lo que respecta al almacenamiento y visualización de datos en la nube, además proporciona versiones de prueba

gratuita, por lo que lo hace ideal en la implementación de este prototipo como plataforma de visualización.

4.1.6.1 Subiendo datos a Ubidots

El primer paso es registrarse en su página web oficial https://ubidots.com/ mediante el uso de un correo electrónico que en el cual se enviara un mensaje de confirmación para poder validar el registro. En la Figura 61 se muestra la página inicial de la plataforma Ubidots.

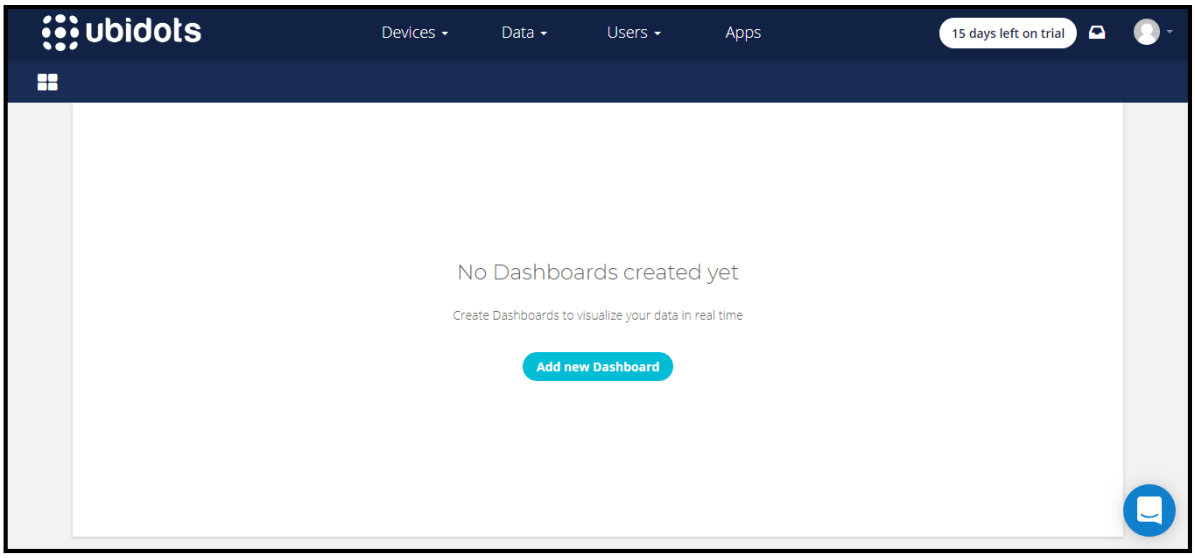


Figura 61 Página web plataforma Ubidots

Fuente: Autoría propia.

Luego se busca el TOKEN de Ubidots que es un ID exclusivo que se le asigna a cada cuenta o usuario para que se realice el proceso de autenticación, en la Figura 62 se puede observar el TOKEN asignado a nuestra cuenta.

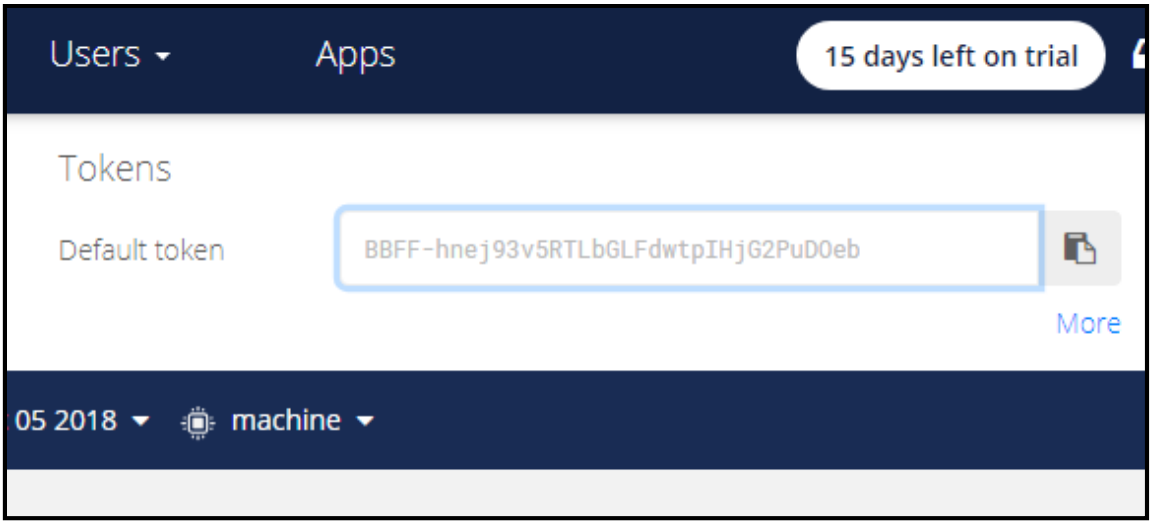


Figura 62 TOKEN asignado por Ubidots

Fuente: Autoría propia.

Ahora se procede a modificar el script que proporciona la página de ayuda de Ubidots https://help.ubidots.com/connect-your-devices/connect-the-raspberry-pi-with-ubidots para conectar la Raspberry Pi con Ubidots, en la Figura 63 se muestra los parámetros más importantes que hay que modificar en el script. En la variable "TOKEN" se asigna el ID obtenido anteriormente, la variable "DEVICE_LABEL" especifica el nombre que va a llevar el dispositivo, finalmente para las variables "VARIABLE_LABEL_X" se asigna las etiquetas que llevaran los datos importantes que se han obtenido en la detección de aparcamientos que son el total de puestos disponibles y el estado de cada puesto.

```
TOKEN = "BBFF-hnej93v5RTLbGLFdwtpIHjG2PuDOeb"
DEVICE_LABEL = "machine"  # Put your device la
VARIABLE_LABEL_1 = "totalpuestos"  # Put your
VARIABLE_LABEL_2 = "puesto_1"  # Put your seco
VARIABLE_LABEL_3 = "puesto_2"  # Put your seco
VARIABLE_LABEL_4 = "puesto_3"  # Put your seco
VARIABLE_LABEL_5 = "puesto_4"  # Put your seco
VARIABLE_LABEL_6 = "puesto_5"  # Put your seco
VARIABLE_LABEL_7 = "puesto_6"  # Put your seco
```

Figura 63 Variables de Ubidots

Fuente: Autoría

Cuando el script se ha modificado se procede a ejecutarlo y automáticamente aparecerá en la página de Ubidots como se muestra en la Figura 64.

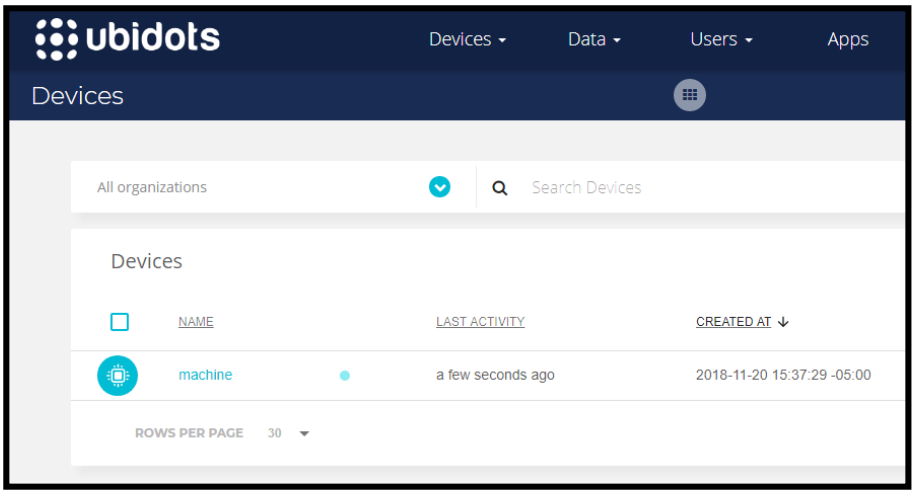


Figura 64 Raspberry conectado a Ubidots.

Fuente: Autoría propia.

A continuación, en la Figura 65 se muestra los datos en las variables asignadas anteriormente.

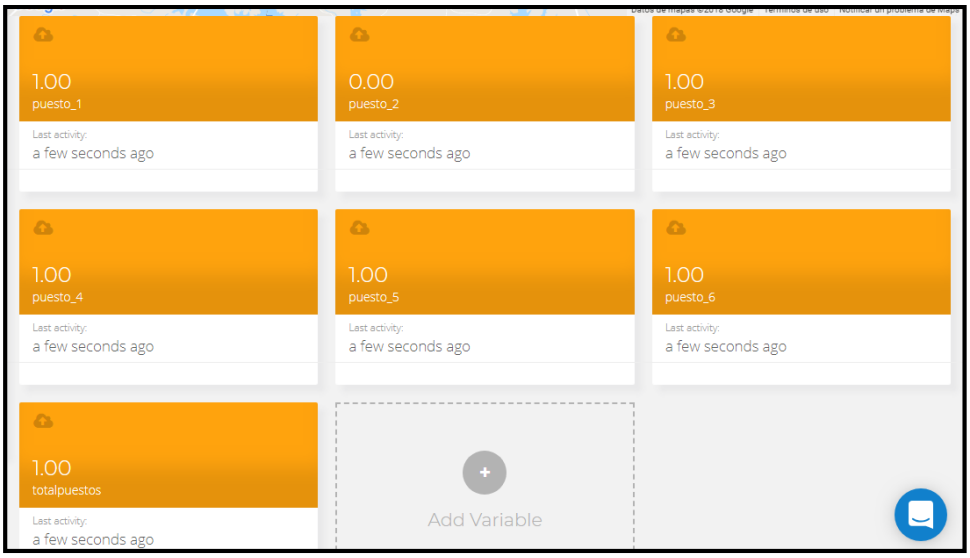


Figura 65 Variables subidas a Ubidots.

Fuente: Autoría

Si bien los datos se están almacenando en Ubidots estos aun no son entendibles, hay que tener en cuenta que para el estado de los puestos se ha asignado valores de 1 y 0, es decir que cuando cualquier variable de “puesto_x” tenga un valor de 1 indicará que el lugar está ocupado y cuando este libre indicara un valor de 0.

Ubidots no solo proporciona el almacenaje de datos, también es posible añadir widgets que permitan mejorar la visualización de los datos almacenados, este proceso se lo realiza en el dashboard de Ubidots, este proporciona widgets de todo tipo como se observa en la Figura 66.

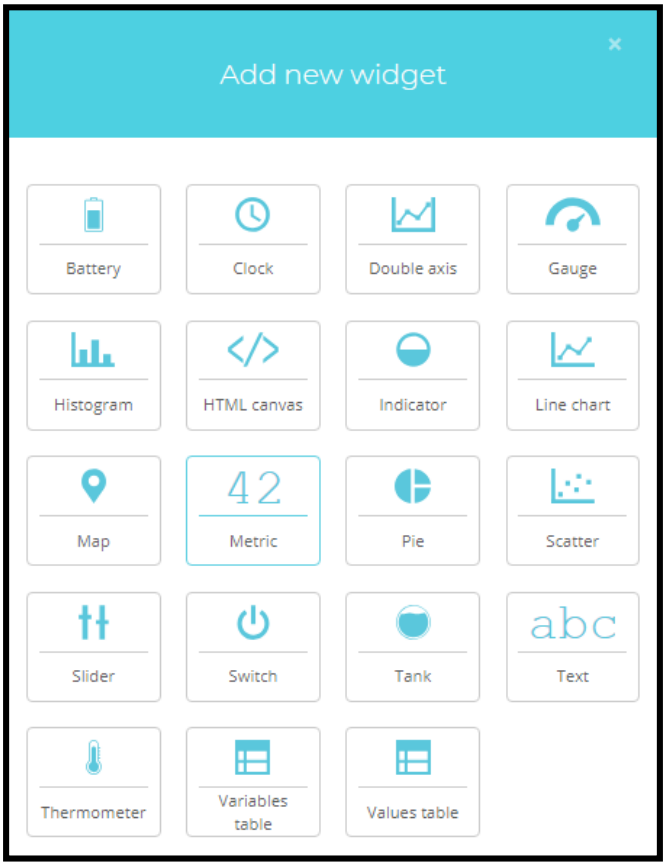


Figura 66 Widgets disponibles en el dashboard de Ubidots.

Fuente: Autoría propia.

Para el prototipo se usará el widget “Metric” para mostrar el número de puesto disponibles y el widget “Indicator” para mostrar el estado de cada puesto.

Para asociar las variables enviadas desde la Raspberry Pi con el widget, se elige el comportamiento del widget en modo estático, como se muestra en la Figura 67.

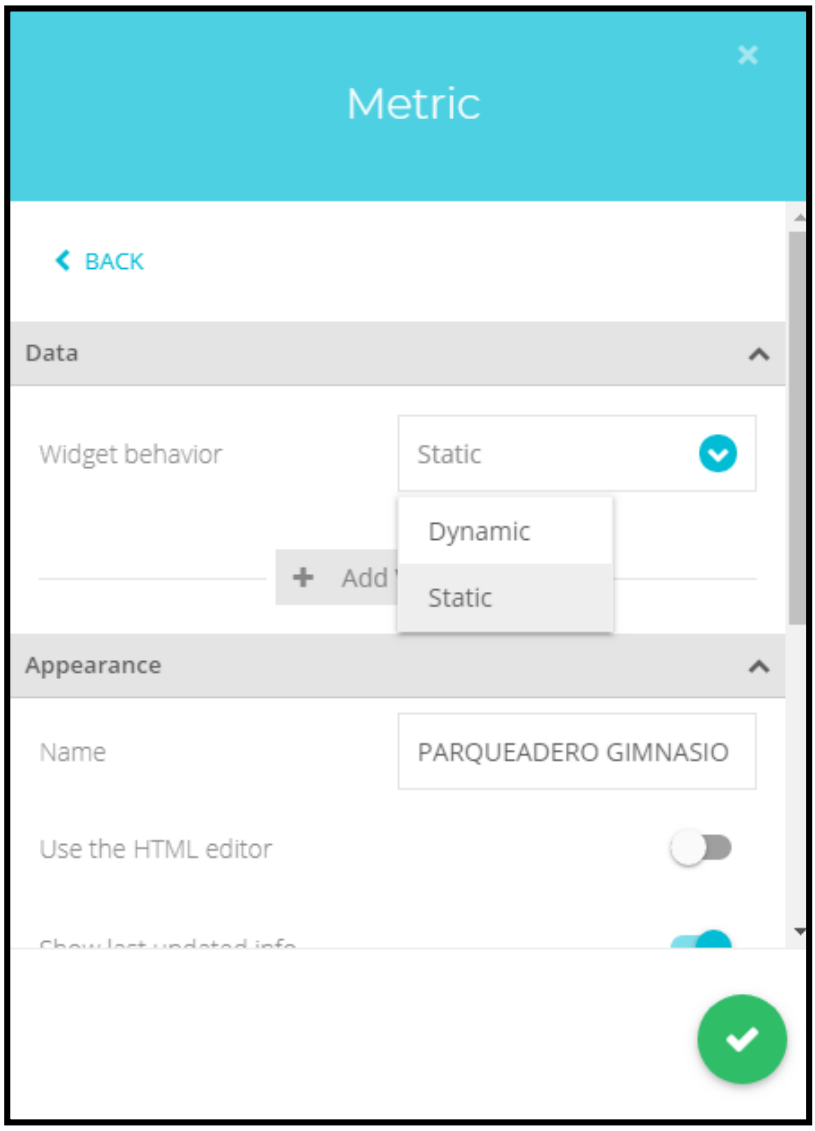


Figura 67 Widget en modo estático.

Fuente: Autoría propia.

Ahora de proceder a enlazar la variable con el widget, en la Figura 68 se observa el dispositivo que se ha creado previamente con todas sus variables, en este caso se escoge la variable “totalpuestos”.

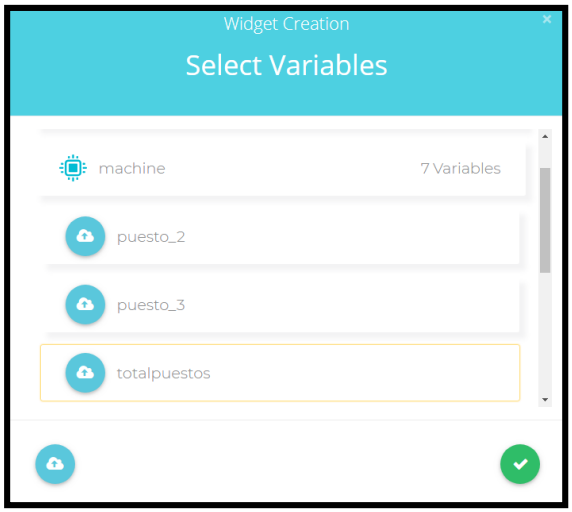


Figura 68 Asociación de variable con el Widget

Fuente: Autoría propia.

En la figura 69 se muestra el widget que se ha creado.

Figura 69 Widget para indicar puestos disponibles.

Fuente: Autoría propia.

Para añadir los estados de cada puesto se usará el widget “Indicator”, el proceso para asociar la variable con el widget es el mismo que anteriormente se ha explicado, en la Figura 70 se puede observar la configuración para asignar colores lógicos al widget, cuando el rango este en valor “0” se mostrará un texto que diga “LIBRE” y el widget será de color verde, cuando el rango este en valor “1” se mostrará un texto que diga “OCUPADO” y el widget será de color rojo.

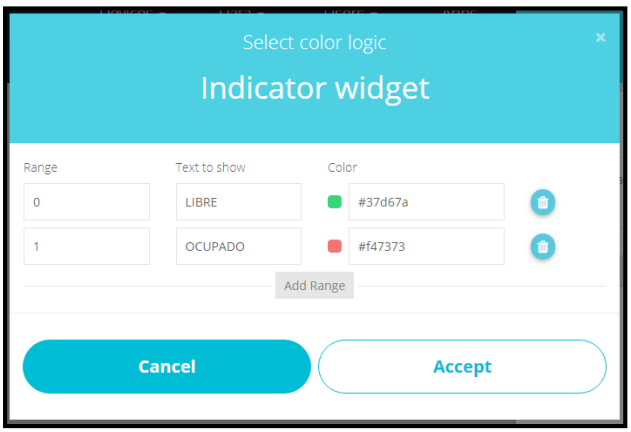
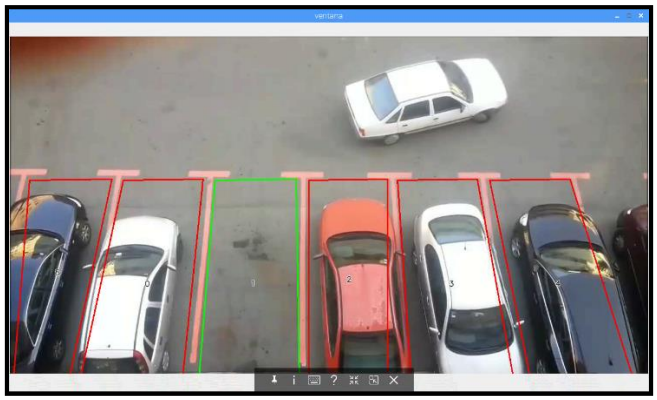


Figura 70 Widget para indicar puestos disponibles.

Fuente: Autoría propia.

En la Figura 71 se puede observar los widgets asignados para cada puesto con sus respectivos estados en la obtención de datos que previamente se ha realizado



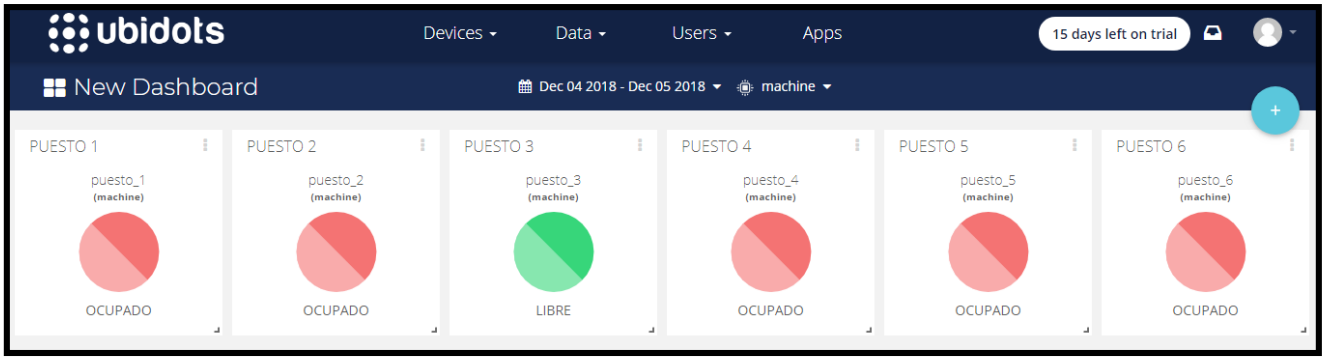


Figura 71 Widget para indicar puestos disponibles.

Fuente: Autoría

4.1.6.2 Página WEB para Visualización de Datos

Se diseñó una página WEB para mejorar la representación de información en la Figura 72 se puede observar la página inicial.

Figura 72 Página web inicial.

Fuente: Autoría propia

Cuando se presiona el botón “Ingresar” se redirige a la página en donde se mostrará los lugares vacantes de cada parqueadero, en la Figura 64 se muestra los widgets antes configurados que mostraran el número de espacios disponibles de cada estacionamiento.

Figura 73 Página informativa de lugares disponibles de cada parqueadero.

Fuente: Autoría propia.

Si el usuario desea informarse más a fondo sobre que puestos son los que están disponibles, se añadió un botón para cada parqueadero que al presionarlo nos redirige a otra página que nos muestra el estado de cada lugar del parqueadero, en la Figura 74 se puede observar el estado de cada lugar usando los widgets que anteriormente se configuró.

Figura 74 Página informativa sobre el estado de cada lugar.

Fuente: Autoría

4.1.6.3 Visualización de Información en Plataforma Móvil

Se elaboró una aplicación para dispositivos con Sistema Operativo Android mediante la herramienta WebView proporcionada por el software Android Studio, esta herramienta permite integrar un navegador en una aplicación, esto quiere decir que se integrara la página web antes creada en una aplicación para un fácil acceso.

En la figura 75 se puede observar algunas capturas de la aplicación móvil.

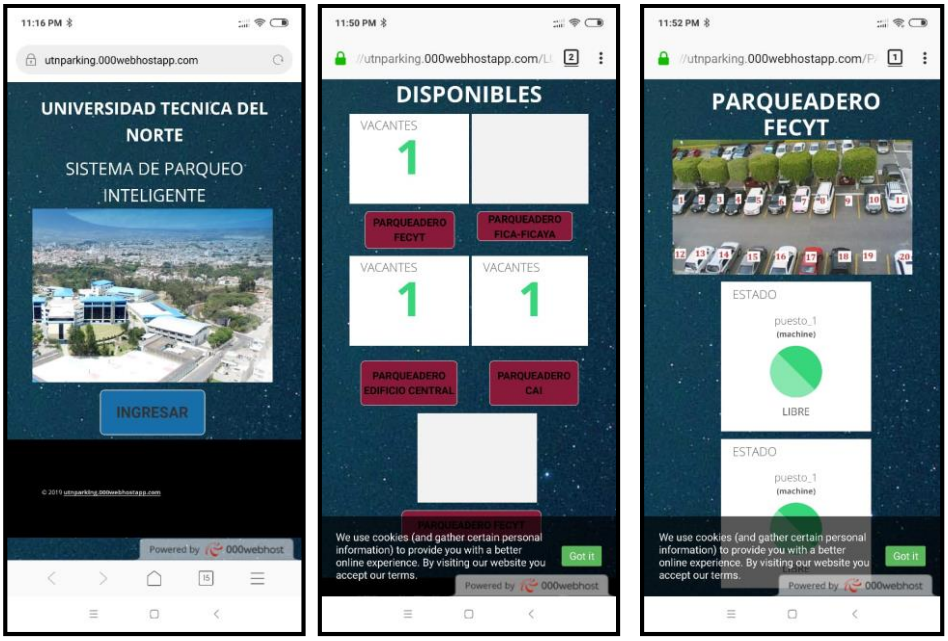


Figura 75 Aplicación informativa para dispositivos Android.

Fuente: Autoría propia

4.1.7 Diagrama de Flujo del Algoritmo de Detección

En la Figura 76 se muestra el proceso de funcionamiento del algoritmo desarrollado en OpenCV.

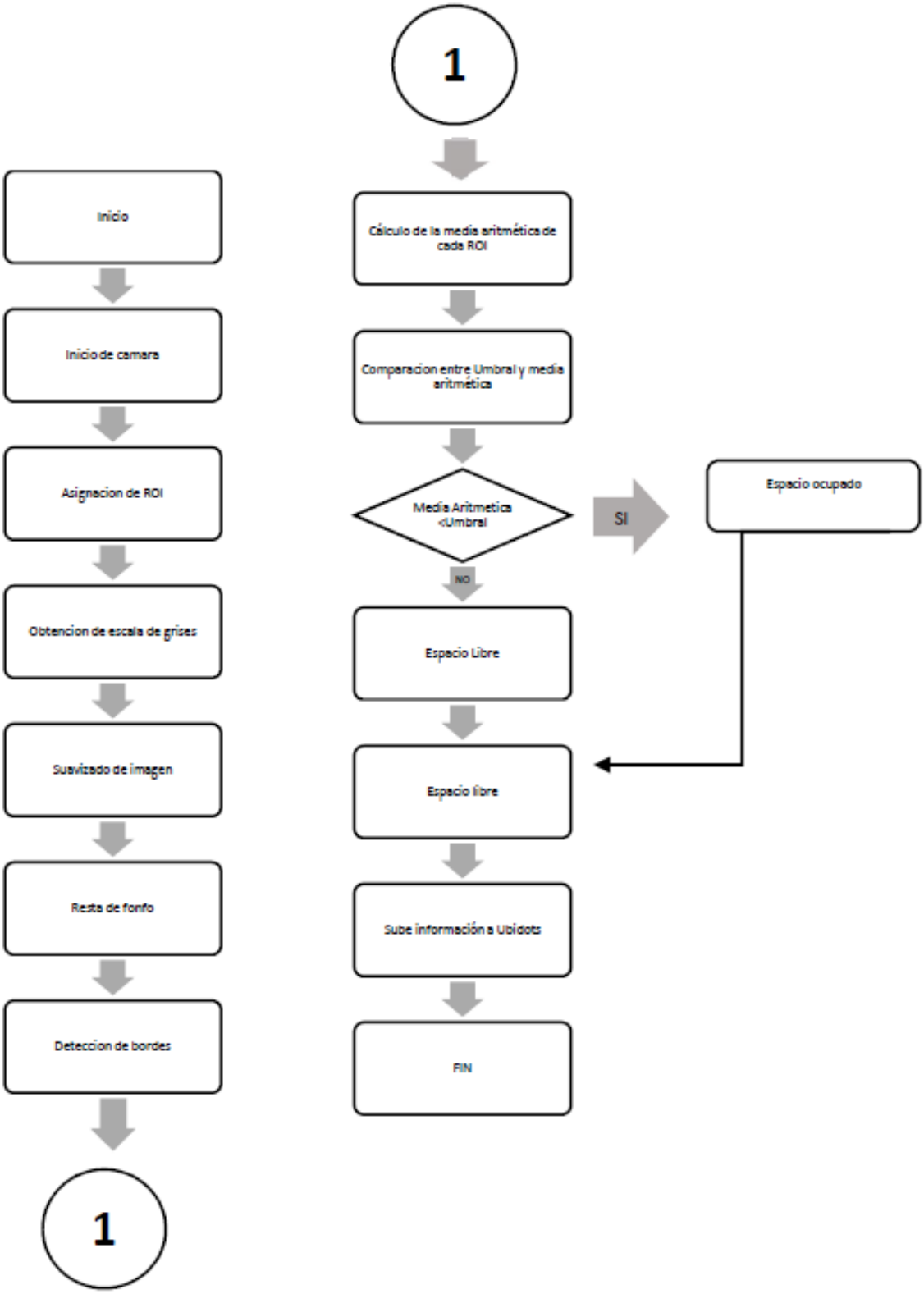


Figura 76 Diagrama de flujo del Prototipo.

Fuente: Autoría propia

4.2. Implementación del Prototipo

En este apartado se realiza la implementación del prototipo, lo cual abarca la instalación y colocación del dispositivo en el parqueadero que se realizaran las pruebas de funcionamiento.

4.2.1 Diagrama Circuital

A continuación, en la Figura 77 se muestra los componentes que forman parte del dispositivo, cabe recalcar que este contara con la placa Raspberry Pi, la cámara y la fuente de alimentación.

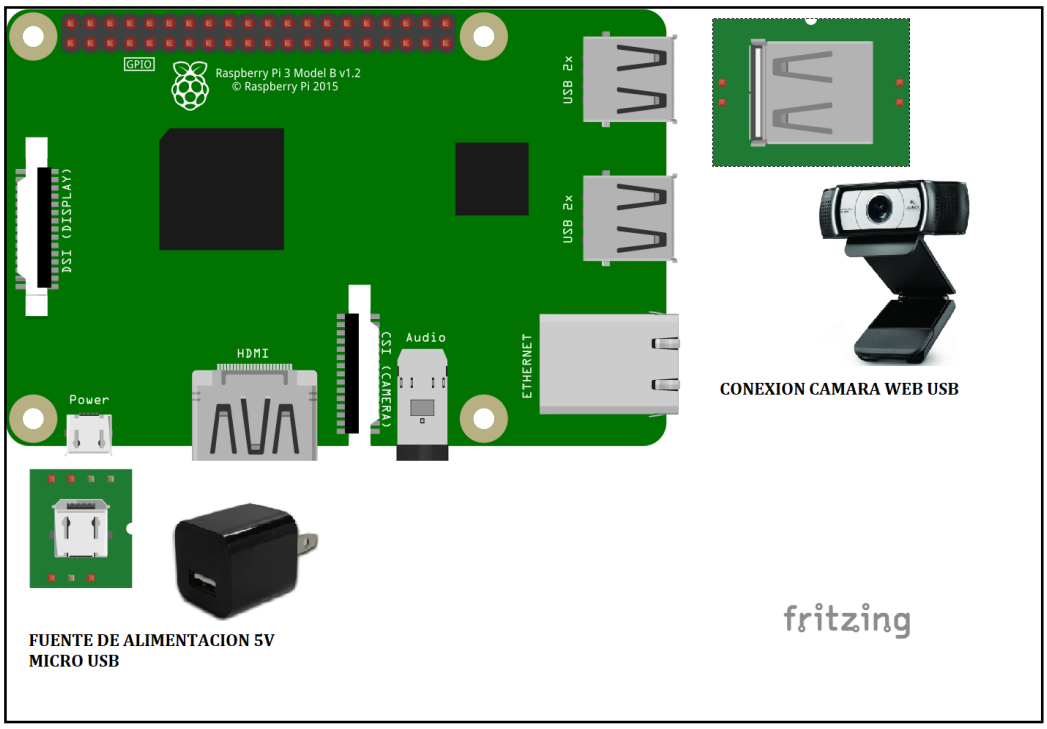


Figura 76 Diagrama circuital del prototipo.

Fuente: Autoría propia.

Básicamente el prototipo cuenta con la placa Raspberry Pi, la cámara web que estará conectada al puerto USB de la placa, la fuente de energía que es un adaptador de 5V 2A que se conecta a la placa a través de una conexión micro USB tipo B y finalmente tendrá conexión WIFI para tener acceso a Internet.

4.2.2 Ubicación del Prototipo

Hay que tener en cuenta que el prototipo estará ubicado en el exterior por lo que surge la necesidad de que este protegido de factores externos como el sol, la lluvia, el polvo, etc. Dicho esto, se utilizó una caja de protección de 20x20x6 centímetros, además cuenta con un soporte que permite la movilidad para calibrar el alcance de visión de la cámara. En la Figura 77 se muestra los materiales utilizados para el montaje de los dispositivos en la caja de protección.

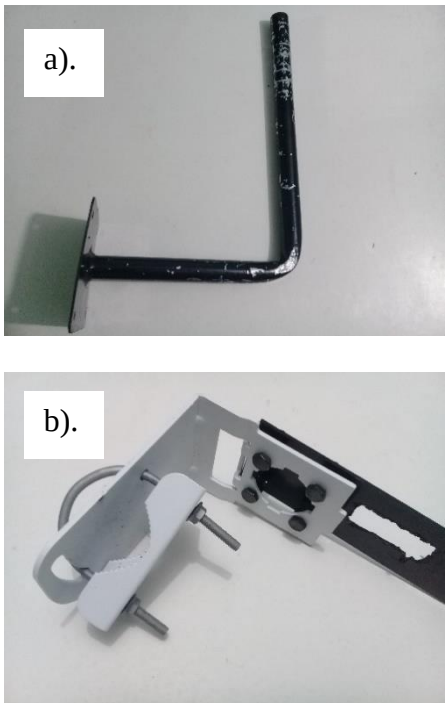


Figura 77 Materiales para el montaje del prototipo. a). Base. b). Soporte y herraje. c). Caja Protectora

Fuente: Autoría propia

Una vez que el dispositivo este montado en su protección se procede a ubicarlo, el prototipo fue ubicado en un lugar alto con el fin de que se pueda abarcar el número máximo de puestos disponibles mediante el alcance de visión de la cámara. En la figura 78 se puede observar la vista interior de los dispositivos en su caja protectora.

Figura 78 Vista interior de la caja.

Fuente: Autoría propia.

A continuación, en la Figura 79 se logra observar el alcance de visión de la cámara hacia el parqueadero.

Figura 79 Alcance de visión en el parqueadero.

Fuente: Autoría propia.

4.2.3 Consumo de Corriente

En la Tabla 10 se detallan los consumos de corriente de los dispositivos presentes en el prototipo.

Tabla 10 *Consumo de corriente.*

Dispositivo	Consumo de corriente
Raspberry Pi 3	1000 mA
Teclado	100 mA

Mouse	100 mA
Conexión HDMI	50 mA
Cámara USB	250 mA
Total	1500 mA

Fuente: Autoría propia.

En el ANEXO 3 se muestra el datasheet de la placa Raspberry Pi en el cual se especifica un voltaje y corriente máximo de entrada de 5v y 2.5 A. Como se observa en la tabla el consumo total es de 1,5 A, dicho esto se utilizará una fuente de carga USB de 2 A que satisface los requerimientos de consumo del prototipo.

4.3. Pruebas de Funcionamiento

En esta fase se realiza el análisis del desempeño y rendimiento del prototipo, las pruebas fueron realizadas en el parqueadero ubicado frente al edificio de la FECYT de la UTN y en varios horarios del periodo diurno. Además de verificar el rendimiento del prototipo mediante este periodo de pruebas se podrá corregir errores que se presenten en esta fase.

4.3.1 Pruebas de Detección

Como primer paso para esta fase de pruebas se procedió a asignar los ROI, en la Figura 80 se puede apreciar la asignación de ROI para cada lugar de estacionamiento.

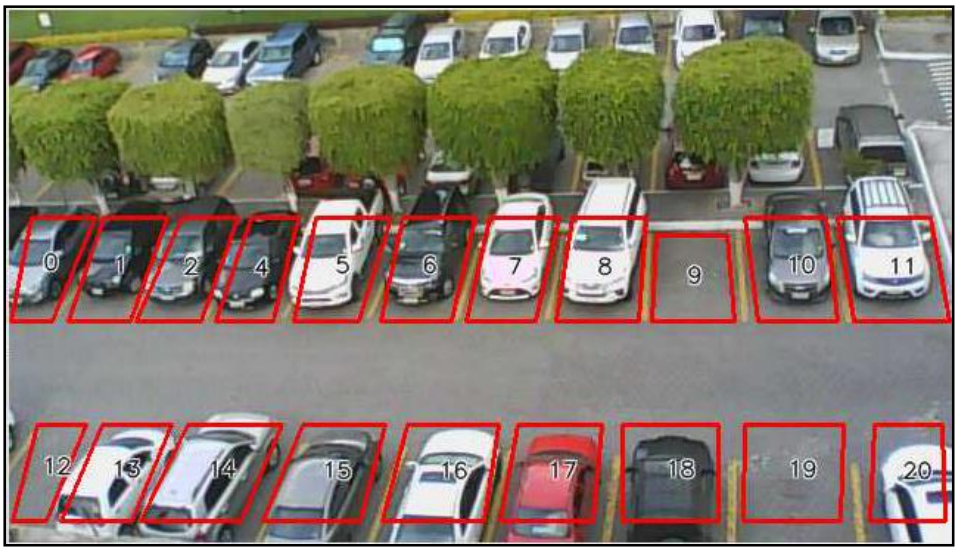


Figura 80 Asignación de ROI.

Fuente: Autoría

Una vez que se han asignado los ROI se procede a calibrar el umbral, para esto se ejecuta el script y se observa los valores arrojados por cada ROI, en la Figura 81 se observa los valores de las media aritméticas de cada ROI.

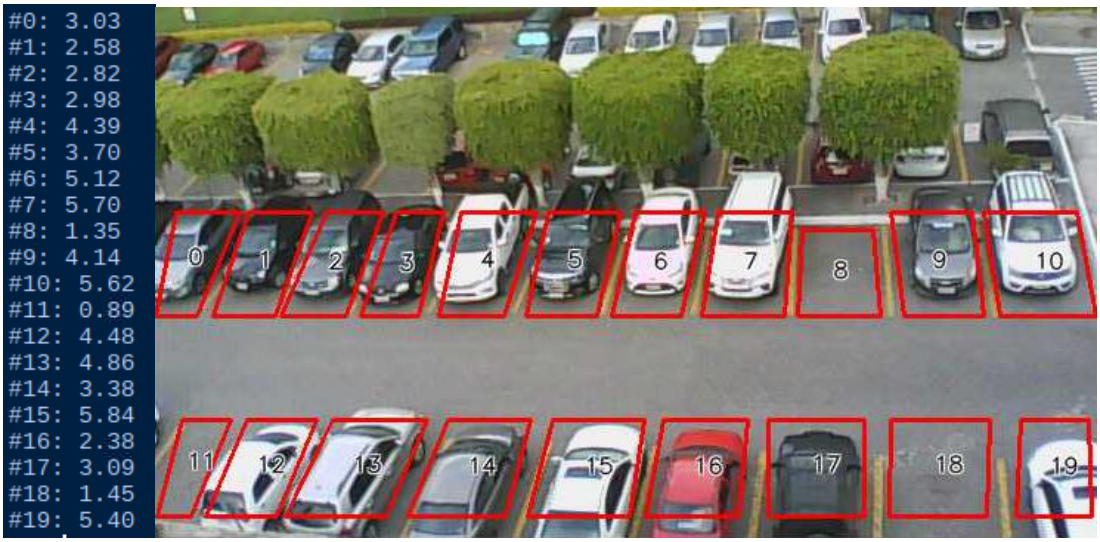


Figura 81 Medias aritméticas de cada ROI.

Fuente: Autoría propia.

Se puede observar que en los ROI 8, 11 y 18 tenemos valores de 1.35, 0.89, 1.45 respectivamente, se usa estos valores como referencia para establecer el umbral, dicho esto se establecerá el umbral en un valor de 1.50.

Se ejecuta de nuevo el script y en la Figura 82 se puede observar el resultado obtenido.

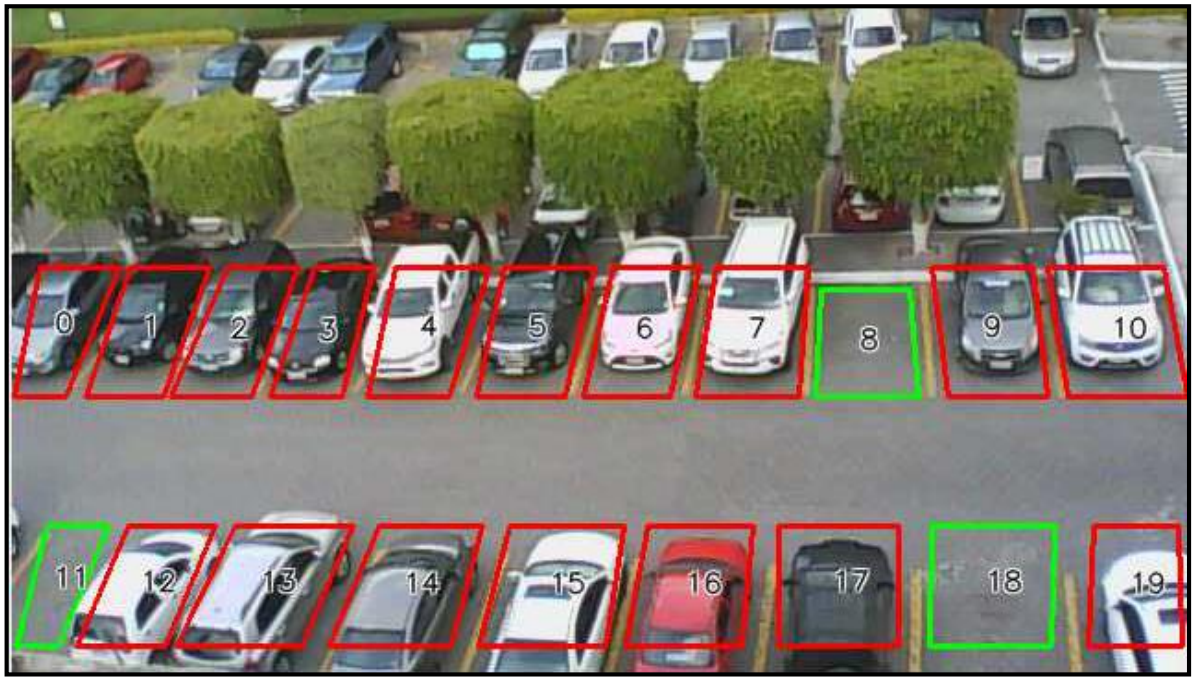


Figura 82 Detección de plazas de estacionamiento.

Fuente: Autoría propia.

Como se observa aquellas plazas que están vacías se colorean de verde indicando que están libres.

Se realizó pruebas en varias horas del día con la finalidad de probar el prototipo con varios cambios de luminosidad, en la Tabla 11 se puede observar el resultado obtenido en varias horas del día.

Tabla 11 *Pruebas de funcionamiento de detección.*

Hora	Condición climática	Plazas Totales	Plazas censadas correctamente
7:30	Cielo despejado	20	20
8:30	Cielo despejado	20	20
9:30	Cielo despejado	20	20
10:30	Cielo despejado	20	20
11:30	Cielo despejado	20	19
12:30	Cielo despejado	20	19
13:30	Cielo despejado	20	20
14:30	Cielo despejado	20	18
15:30	Nublado	20	18
16:30	Nublado	20	19
17:30	Nublado	20	19
18:30	Nublado	20	19

Fuente: Autoría propia.

4.4. Costos

En esta parte se presentan los gastos involucrados tanto en la construcción del prototipo, así como también los gastos totales del sistema implementado en todos los parqueaderos de la Universidad.

Los costos tomados en cuenta son hardware, software e infraestructura.

4.4.1 Costos de Hardware del Prototipo

Los materiales y costos para la elaboración del prototipo se detallan en la Tabla 12.

Tabla 12 *Costos de hardware del prototipo.*

Descripción	Cantidad	Precio Unitario	Precio total
Raspberry Pi 3 Model B	1	$75,00	$75,00
Cámara USB	1	$30,00	$30,00
Micro SD Kingston 16 GB	1	$16,00	$16,00
Fuente de poder 5v 2A	1	$10,00	$10,00
Adaptador HDMI-VGA	1	$20,00	$20,00
Cable VGA 2m	1	$5,00	$5,00
Mouse	1	$10,00	$10,00
Teclado	1	$10,00	$10,00
TOTAL			$176,00

Fuente: Autoría propia.

4.4.2 Costos De Software Del Prototipo

Tabla 13 *Costos de software del prototipo.*

Descripción	Cantidad	Precio Unitario	Precio total
Sistema Operativo Raspbian	1	0	0
OpenCV	1	0	0
Ubidots	1	0	0
Hosting página WEB	1	0	0
TOTAL			$0,0

Fuente: Autoría propia.

4.4.3 Costos De Infraestructura Del Prototipo

Tabla 14 *Costos de infraestructura del prototipo.*

Descripción	Cantidad	Precio Unitario	Precio total
Soporte	1	5	5
Caja protectora	1	5	5
Herrajes	1	5	5
Base de metal	1	5	5
TOTAL			$20,00

Fuente: Autoría propia.

4.4.4 Costos Totales Del Prototipo

Tabla 15 *Costos totales del prototipo*.

Descripción	Precio total
Costos hardware	176,00
Costos software	0,00
Costos infraestructura	20,00
TOTAL	$196,00

Fuente: Autoría

4.4.5 Costos Totales De Implementación Del Sistema

Previamente se eligió un total de 12 cámaras para cubrir todos los parqueaderos de la Universidad, dicho esto se realizará el cálculo para 15 dispositivos detectores de plazas de estacionamiento teniendo en cuenta que ciertos parqueaderos son más grandes que otros y podrían requerir más de un dispositivo.

Para este cálculo solo se toman en cuenta los materiales necesarios que necesita el dispositivo detector, por lo tanto, se descartan materiales que se tomaron en cuenta en el cálculo anterior como el mouse, teclado, cables, etc.

4.4.6 Costos De Hardware Del Sistema

Tabla 16 *Costos de hardware del sistema.*

Descripción	Cantidad	Precio Unitario	Precio total
Raspberry Pi 3 Model B	15	$75,00	$1125,00
Cámara USB	15	$30,00	$450,00
Micro SD Kingston 16 GB	15	$16,00	$240,00
Fuente de poder 5v 2A	15	$15,00	$225,00
TOTAL			$2040,00

Fuente: Autoría

4.4.7 Costos De Software Del Sistema

Tabla 17 *Costos de software del sistema.*

Descripción	Cantidad	Precio Unitario	Precio total
Sistema Operativo Raspbian	1	0	0
OpenCV	15	0	0
Ubidots	15	$20,00 (Suscripción de un año para 20 dispositivos)	$20,00
Hosting página WEB	1	20(Suscripción de un año)	$20,00

TOTAL	$40,00

Fuente: Autoría propia.

4.4.8 Costos De Infraestructura Del Sistema

Tabla 18 *Costos de infraestructura del prototipo.*

Descripción	Cantidad	Precio Unitario	Precio total
Soporte	15	5	75
Caja protectora	15	5	75
Herrajes	15	5	75
Base de metal	15	5	75
TOTAL			$300,00

Fuente: Autoría propia.

4.4.4 Costos Totales Del Sistema

Tabla 19 *Costos totales del sistema.*

Descripción	Precio total
Costos hardware	$2040,00
Costos software	$40,00
Costos infraestructura	$300,00
TOTAL	$2380,00

Fuente: Autoría

4.5. Beneficios

Una vez culminado el periodo de pruebas y el estudio de costos se pueden analizar los beneficios que proporciona este tipo de soluciones.

Gracias a la visión artificial es posible abarcar muchos puestos de estacionar por lo que es mucho más eficiente que otras soluciones como las redes de sensores en las cuales se necesita un nodo sensor por cada lugar.

Como se ha visto anteriormente es una solución mucho menos costosa en vista de que se necesita menos nodos sensores para abarcar la totalidad de un parqueadero, a diferencia de una red de sensores en la cual el costo es muy elevado ya que un nodo sensor solo permite detectar un lugar de estacionamiento, dicho esto el costo de esta solución es directamente proporcional al número de puestos disponibles en un parqueadero por lo cual su costo es mucho mayor a la solución que se presenta en este proyecto.

En lo que se refiere al consumo energético también presenta beneficios ya que como se ha dicho anteriormente un solo dispositivo puede abarcar varios puestos y en efecto el consumo de energía es totalmente reducido a diferencia de otras soluciones que necesitan un dispositivo por cada lugar de estacionamiento lo que significa que va a tener un consumo de energía elevado.

Gracias a las plataformas de visualización en la nube esta solución permite dar al usuario un interfaz amigable y simple para observar la información sobre la disponibilidad de lugares de estacionamiento.

Capítulo V

5.1. Conclusiones

- El prototipo de detección de aparcamientos libre mediante el uso de visión artificial con un aplicativo IoT permitió informar sobre puestos libres en un parqueadero de la UTN que fue desarrollado sobre una placa Raspberry Pi.
- Los conceptos básicos adquiridos fueron de gran ayudad para el desarrollo del prototipo, primordialmente aquellos relacionados con las técnicas de procesamiento de imágenes, visión artificial e IoT, que facilitaron el diseño del prototipo.
- Mediante el análisis de la situación actual de los parqueaderos de la UTN se consiguió encontrar falencias en el sistema de parqueo vehicular, facilitando así establecer requerimientos en el diseño del prototipo para reducir estos fallos.
- Raspberry Pi 2 modelo B, OpenCV y Python fueron las opciones más ideales para el desarrollo del prototipo, todo esto basado en los requerimientos de software y hardware del prototipo y además en un análisis de dispositivos existentes en el mercado.
- El algoritmo de detección de plazas de estacionamiento utilizó herramientas de visión artificial, este se basó en la detección de bordes presentes en el lugar de parqueo mediante librerías de procesamiento de imágenes proporcionadas por OpenCV.
- La página web informativa permitió mostrar los datos obtenidos en el prototipo para que sean de fácil visualización para los usuarios, además gracias al desarrollo de el aplicativo móvil para Android se puede observar la información de la página web en terminales con este Sistema Operativo.

- Las pruebas de funcionamiento en el parqueadero ubicado en la FECYT permitieron determinar el funcionamiento y desempeño del prototipo, para ello se realizaron capturas de video en varias horas del día, permitiendo así testear el prototipo en diferentes escenarios en los cuales existieron variaciones de luz.
- El análisis de costos realizado permitió analizar la viabilidad de este tipo de soluciones fueron necesarios analizar tanto del presupuesto utilizado para el desarrollo del prototipo, así como también los costos necesarios para la implementación del sistema en todos los parqueaderos de la Universidad.

5.2. Recomendaciones

- El prototipo de detección de aparcamientos debe estar ubicado en un lugar de gran altura que permita abarcar la mayor cantidad posible de lugares de estacionamiento,
- El prototipo debe estar ubicado de manera frontal al parqueadero puesto que si se lo ubica de manera lateral no se obtendrá un buen funcionamiento del algoritmo de detección.
- Se recomienda que la cámara tenga un gran alcance de visión angular que permita visualizar la mayor cantidad de puestos de estacionar,
- Es recomendable ubicar el dispositivo en una caja protectora contra agua y polvo que garantice una larga vida a los dispositivos que están en su interior.

REFERENCIAS BIBLIOGRÁFICAS

Takizawa, H., Yamada, K., and Ito, T. Year. *"Vehicles detection using sensor fusion,"* Intelligent Vehicles Symposium, 2004 IEEE, IEEE2004, pp. 238-243.

Xu, J., Chen, G., and Xie, M. Year. *"Vision-guided automatic parking for smart car,"* Proceedings of the IEEE Intelligent Vehicles Symposium2000, pp. 725-730.

Zhu, Z., Zhao, Y., and Lu, H. Year. *"Sequential architecture for efficient car detection,"* 2007 IEEE Conference on Computer Vision and Pattern Recognition, IEEE2007, pp. 1-8.

Funck, S., Mohler, N., and Oertel, W. Year. *"Determining car-park occupancy from single images,"* Intelligent Vehicles Symposium, 2004 IEEE, IEEE2004, pp. 325-328.

Fabian, T. Year. *"An algorithm for parking lot occupation detection,"* Computer Information Systems and Industrial Management Applications, 2008. CISIM'08. 7th, IEEE2008, pp. 165-170

Jung, H. G., Kim, D. S., Yoon, P. J., and Kim, J. Year. *"Light stripe projection based parking space detection for intelligent parking assist system,"* 2007 IEEE Intelligent Vehicles Symposium, IEEE2007, pp. 962-968.

Banerjee, S., Choudekar, P., and Muju, M. Year. *"Real time car parking system using image processing,"* Electronics Computer Technology (ICECT), 2011 3rd International Conference on, IEEE2011, pp. 99-103.

Bong, D., Ting, K., and Lai, K. 2008. *"Integrated approach in the design of car park occupancy information system (COINS),"* IAENG International Journal of Computer Science (35:1), pp 7-14.

Yan, G., Weigle, M. C., and Olariu, S. Year. *"A novel parking service using wireless networks,"* Service Operations, Logistics and Informatics, 2009. SOLI'09. IEEE/INFORMS International Conference on, IEEE2009, pp. 406-411.

Zheng, Y., and Cao, J. Year. *"An intelligent car park management system based on wireless sensor networks,"* 2006 First International Symposium on Pervasive Computing and Applications, IEEE2006, pp. 65-70.

O'Flynn, B., Bellis, S., Delaney, K., Barton, J., O'Mathuna, S. C., Barroso, A. M., Benson, J., Roedig, U., and Sreenan, C. Year. *"The development of a novel minaturized modular platform for wireless sensor networks,"* IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005., IEEE2005, pp. 370-375.

Kumar, R., Chilamkurti, N. K., and Soh, B. Year. *"A comparative study of different sensors for smart car park management,"* Intelligent Pervasive Computing, 2007. IPC. The 2007 International Conference on, IEEE2007, pp. 499-502.

Lee, S., Yoon, D., and Ghosh, A. Year. *"Intelligent parking lot application using wireless sensor networks,"* Collaborative Technologies and Systems, 2008. CTS 2008. International Symposium on, IEEE2008, pp. 48-57.

Park, W.-J., Kim, B.-S., Seo, D.-E., Kim, D.-S., and Lee, K.-H. Year. *"Parking space detection using ultrasonic sensor in parking assistance system,"* Intelligent Vehicles Symposium, 2008 IEEE, IEEE2008, pp. 1039-1044.

Tubaishat, M., Zhuang, P., Qi, Q., and Shang, Y. 2009. *"Wireless sensor networks in intelligent transportation systems,"* Wireless communications and mobile computing (9:3), pp 287-302.

Reve, S. V., and Choudhri, S. 2012. *"Management of car parking system using wireless sensor network,"* Int. J. Emerg. Technol. Adv. Eng (2), pp 262-268.

Sharma, A., Chaki, R., and Bhattacharya, U. Year. *"Applications of wireless sensor network in Intelligent Traffic System: A review,"* Electronics Computer Technology (ICECT), 2011 3rd International Conference on, IEEE2011, pp. 53-57.

Kianpisheh, A., Mustaffa, N., Limtrairut, P., and Keikhosrokiani, P. 2012. *"Smart parking system (SPS) architecture using ultrasonic detector,"* International Journal of Software Engineering and Its Applications (6:3), pp 55-58.

Mateo, R. M. A., Lee, Y.-s., and Lee, J. Year. *"Collision detection for ubiquitous parking management based on multi-agent system,"* KES International Symposium on Agent and Multi-Agent Systems: Technologies and Applications, Springer2009, pp. 570-578.

Khoukhi, A. Year. *"An intelligent multi-agent system for mobile robots navigation and parking,"* Robotic and Sensors Environments (ROSE), 2010 IEEE International Workshop on, IEEE2010, pp. 1-6.

Li, C.-C., Chou, S.-Y., and Lin, S.-W. Year. *"An agent-based platform for drivers and car parks negotiation,"* Networking, Sensing and Control, 2004 IEEE International Conference on, IEEE2004, pp. 1038-1043.

Stibor, L., Zang, Y., and Reumerman, H.-J. Year. *"Evaluation of communication distance of broadcast messages in a vehicular ad-hoc network using IEEE 802.11 p,"* 2007 IEEE Wireless Communications and Networking Conference, IEEE2007, pp. 254-257.

Holfelder, W. Year. *"Vehicle-to-vehicle and vehicle-to-infrastructure communication: recent developments, opportunities and challenges,"* Workshop: Future Generation Software Architectures in the Automotive Domain, La Jolla2004.

Yousefi, S., Mousavi, M. S., and Fathy, M. Year. *"Vehicular ad hoc networks (VANETs): challenges and perspectives,"* 2006 6th International Conference on ITS Telecommunications, IEEE2006, pp. 761-766.

Bilstrup, K., Uhlemann, E., Strom, E. G., and Bilstrup, U. Year. *"Evaluation of the IEEE 802.11 p MAC method for vehicle-to-vehicle communication,"* Vehicular Technology Conference, 2008. VTC 2008-Fall. IEEE 68th, IEEE2008, pp. 1-5.

Panayappan, R., Trivedi, J. M., Studer, A., and Perrig, A. Year. *"VANET-based approach for parking space availability,"* Proceedings of the fourth ACM international workshop on Vehicular ad hoc networks, ACM2007, pp. 75-76.

Chon, H. D., Agrawal, D., and El Abbadi, A. Year. *"NAPA: Nearest available parking lot application,"* Data Engineering, 2002. Proceedings. 18th International Conference on, IEEE2002, pp. 496-497.

Hanif, N. H. H. M., Badiozaman, M. H., and Daud, H. Year. *"Smart parking reservation system using short message services (SMS),"* Intelligent and Advanced Systems (ICIAS), 2010 International Conference on, IEEE2010, pp. 1-5.

Pala, Z., and Inanc, N. Year. *"Smart parking applications using RFID technology,"* RFID Eurasia, 2007 1st Annual, IEEE2007, pp. 1-3.

Gueaieb, W., and Miah, M. S. 2008. *"An intelligent mobile robot navigation technique using RFID technology,"* IEEE Transactions on Instrumentation and Measurement (57:9), pp 1908-1917.

Basu, A. (2014). *SMART PARKING*. Obtenido de Happiest Minds Web Site: https://www.happiestminds.com/whitepapers/smart-parking.pdf

Bawany, N. Z., & Shamsi, J. A. (2015). *Smart City Architecture: Vision and Challenges*. International Journal of Advanced Computer Science and Applications, 246-255.

BeagleBoard . (2018). BeagleBoard . Obtenido de BeagleBoard : https://beagleboard.org/black

Fraifer, M., & Fernström, M. (2016). *Investigation of Smart Parking Systems and their technologies*. Thirty Seventh International Conference on Information Systems.

Geng, Y., & Cassandras, C. (2012). *A new "smart parking" system infrastructure and implementation*. Procedia-Social and Behavioral Sciences, 1278-1287.

Goyal, A., Bijalwan, A., & Chowdhury, K. (2012). *A Comprehensive Review of Image Smoothing Techniques.* International Journal of Advanced Research in Computer Engineering & Technology, 315-320.

Jaguar Board. (2018). Jaguar Board. Obtenido de Jaguar Board: http://www.jaguarboard.org/

Krishna, R. (2017). *COMPUTER VISION FOUNDATIONS AND APPLICATIONS.* Stanford: Stanford University.

Macedo, S., Melo, G., & Kelner, J. (2015). *A comparative study of grayscale conversion techniques applied to SIFT descriptors*. SBC Journal on Interactive Systems, 30-35.

Mahmud, S. (2013). *A survey of intelligent car parking system.* Journal of applied research and technology, 714-726.

Martin, M. (2013). PROCESAMIENTO DIGITAL DE IMAGENES. Mexico: Universidad Autonoma de Puebla.

MathWorks. (2018). MathWorks. Obtenido de MathWorks: https://www.mathworks.com/help/images/roi-based-processing.html

OpenCV . (2018). OpenCV . Obtenido de OpenCV : https://opencv.org/

Raspberry. (2018). Raspberry. Obtenido de Raspberry Web Site: https://www.raspberrypi.org/

Savant, S. (2014). *A Review on Edge Detection Techniques for Image Segmentation*. International Journal of Computer Science and Information Technologies, 5898-5910.

Sayed, A., & Mohamed, A. (2017). *Internet of Things Applications, Challenges and Related Future Technologies.* world scientific news. World Scientific News, 127-135.

Senplades. (2013). *PLAN NACIONAL DEL BUEN VIVIR*. Obtenido de Senplades Web site: http://www.buenvivir.gob.ec/versiones-plan-nacional

Stadt Wien. (2015). *SMART CITY.* Obtenido de Stadt Wien: https://www.wien.gv.at/stadtentwicklung/studien/pdf/b008403j.pdf

YAML. (2018). YAML. Obtenido de YAML: https://yaml.org/

Cervantes, L. F., Lee, Y.-S., Yang, H., and Lee, J. Year. *"A hybrid middleware for RFID-based parking management system using group communication in overlay networks,"* Intelligent Pervasive Computing, 2007. IPC. The 2007 International Conference on, IEEE2007, pp. 521-526.

Castellano R. Sergio S. (2016). *Sistema de asistencia y monitoreo para un parqueadero en entorno abierto usando visión artificial.* Trabajo de grado de ingeniería Electrónica. Fundación Universitaria Los Libertadores. Facultad de Ingeniería, Colombia.

Cognex. (2016) *Introducción a la visión artificial. Una guía para la automatización de procesos y mejoras de calidad.*

Fernström Mikael y Fraifer Muftah Fraifer (2016).Thirty Seventh International Conference on Information Systems, Dublin 2016 1, At Dublin, Ireland, Volume: *"IoT & Smart City Challenges and Applications"* – ISCA 2016. Bajado de Internet de la página siguiente:https://www.researchgate.net/publication/311535831_Investigation_of_Smart_Parking_Systems_and_their_technologies

Furero Alejandro (2010). *Manejo de bibliotecas OpenCV*. Presentación bajada de la Web.

Martín O., Manuel (2013). *Procesamiento digital de imágenes*. Universidad Autónoma de Puebla. Facultad de computación. México.

Platero D., Carlos. (2009) *Introducción a la visión artificial*. Universidad Politécnica de Madrid, Dpto. de Electrónica, Automática e Informática. España.

Sánchez Onofre, Julio (2015). *Gráfico de IBM de la Smart City*. Gráfico realizado con datos tomados de IBM del gráfico de J. Ríos. Madrid España. Tomado de https://ovacen.com/smart-city-ventajas-y-desventajas/#Que_es_el_eMobility

Chamizo, Alberto (2013). *Entorno de programación gráfico OpenCV*. Ingeniería electrónica industrial y automática. Departamento de ingeniería de sistemas y automática Escuela politécnica. Universidad Carlos III de Madrid, España.

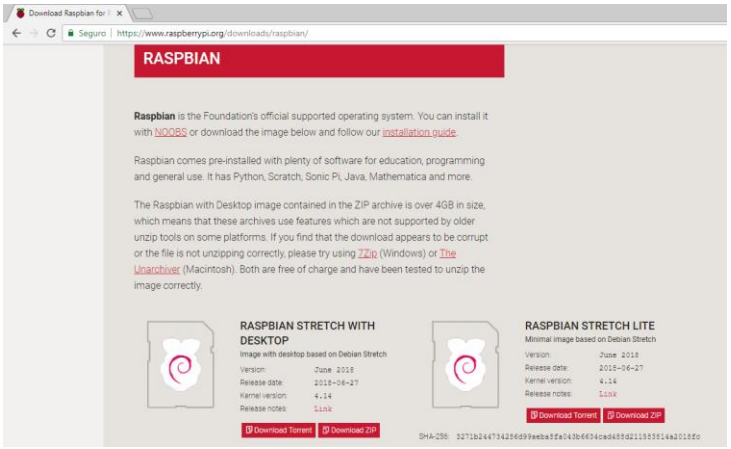
ANEXOS

ANEXO 1

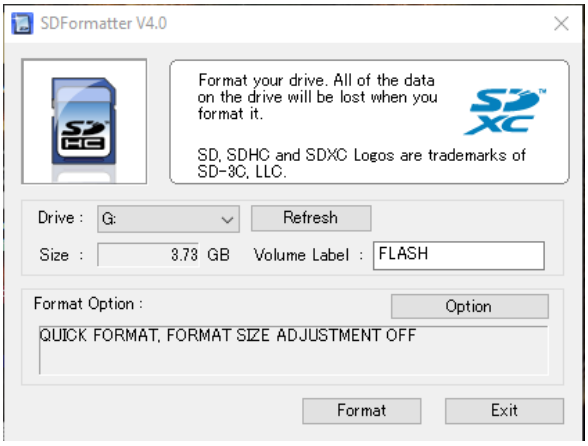
Guia de instalacion de RASPBIAN STRECH en Raspberry Pi 3

La presente guia muetra el proceso para instalar el sistema operativo RASPBIAN STRECH en la placa Raspberry Pi 3.

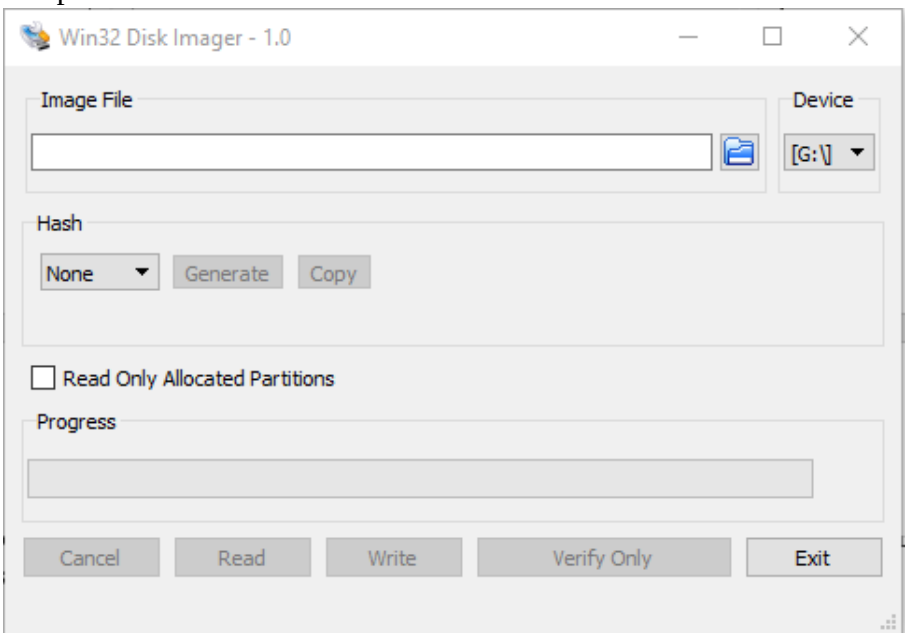
1. Descargamos la imagen iso de la pagina oficial de Raspberry en el siguiente enlace https://www.raspberrypi.org/downloads/raspbian/



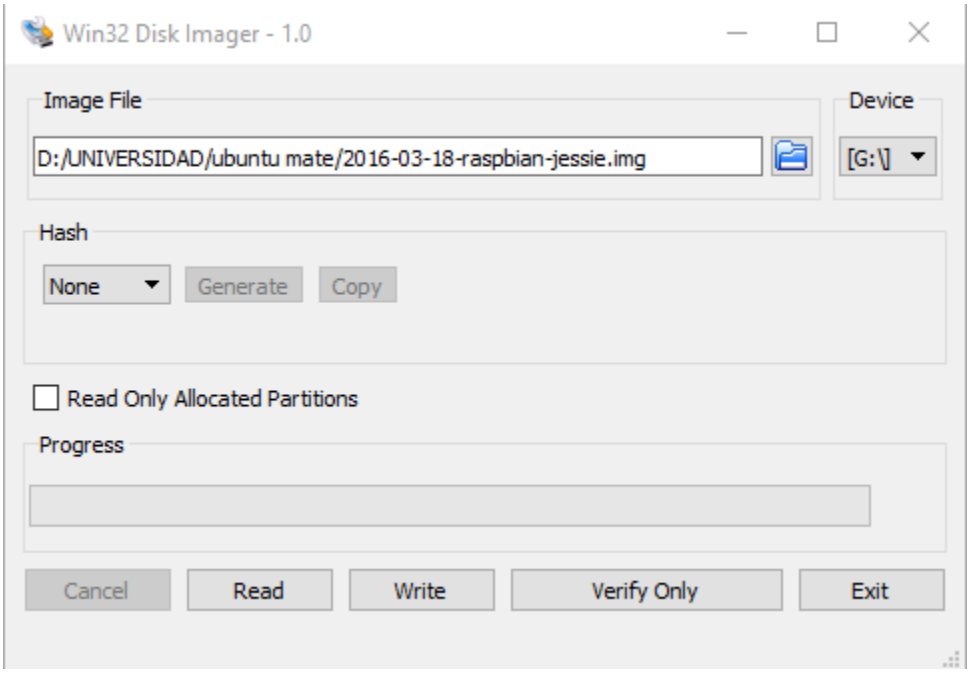
2. Mediante el software SD FORMATER vamos a formatear nuestra memoria SD.



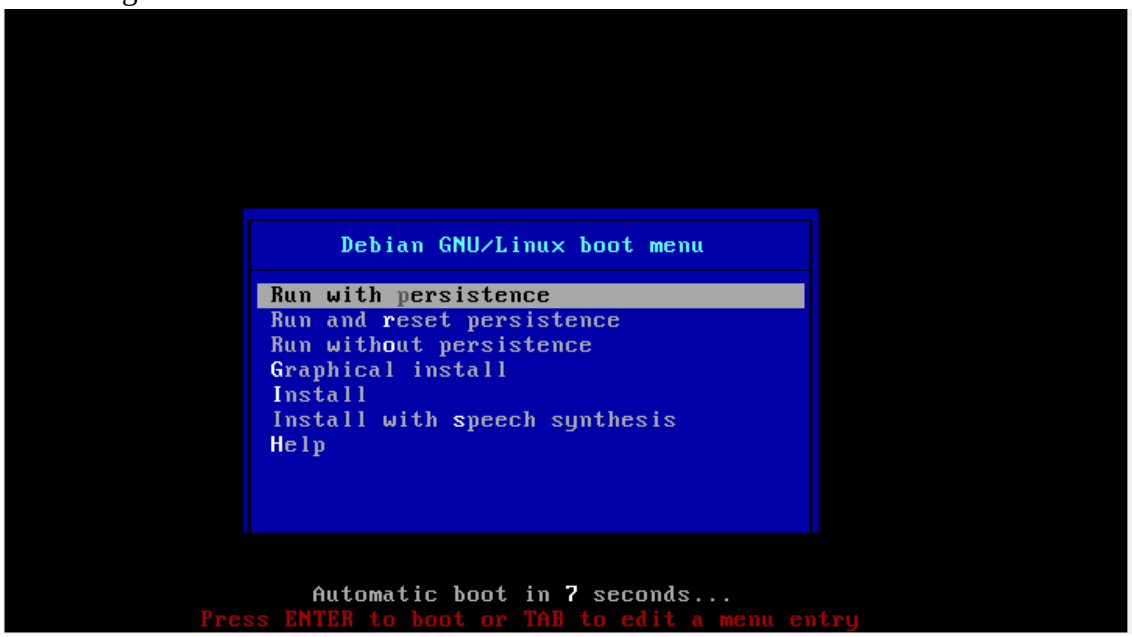
3. Ejecutamos el programa WIN32 DISK IMAGER, el cual nos ayudara a grabar el sistema operativo en la memoria SD.



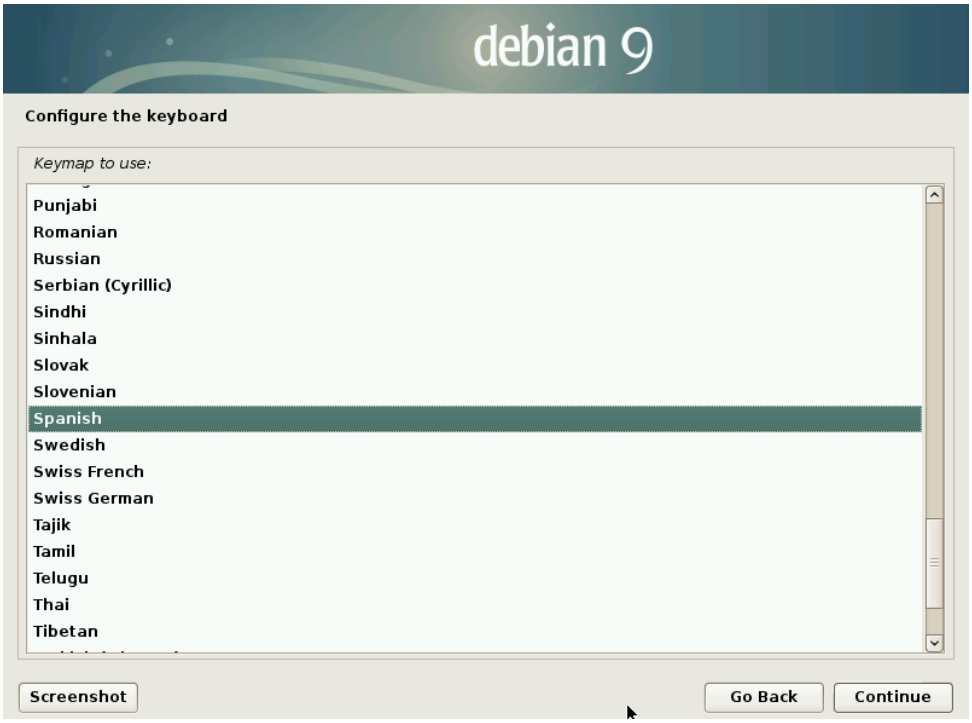
Seleccionamos la imagen .img que descargamos anteriormente y la unidad sd en la cual vamos a quemar el sistema operativo, seguido de esto damos clic en write para proceder a la escritura del sistema operativo en la SD.



4. Una vez terminado el proceso de escritura, introducimos la SD en la placa Raspberry, conectamos la fuente de alimentación y el cable HDMI a la pantalla para la visualización.
5. Nos mostrara una ventana con varias opciones, en la cual seleccionamos Graphical Install. Mediante esta opción realzaremos la instalación con una interfaz gráfica amigable al usuario.

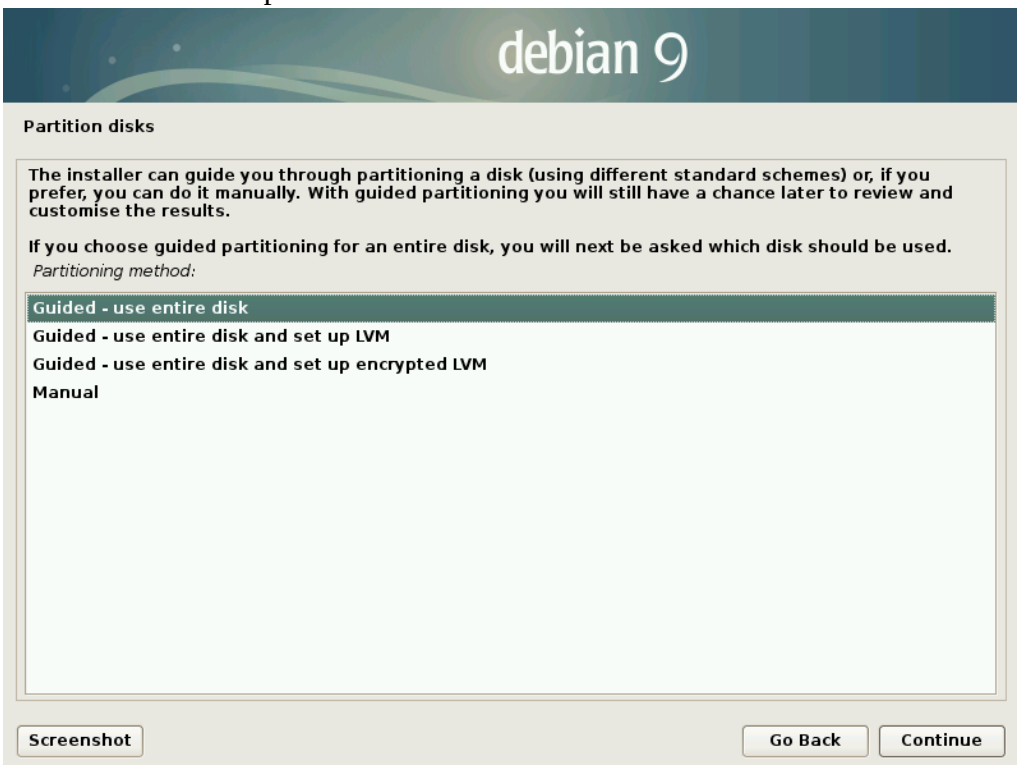


6. Selecionamos el idioma del teclado en este caso vamos a seleccionar español



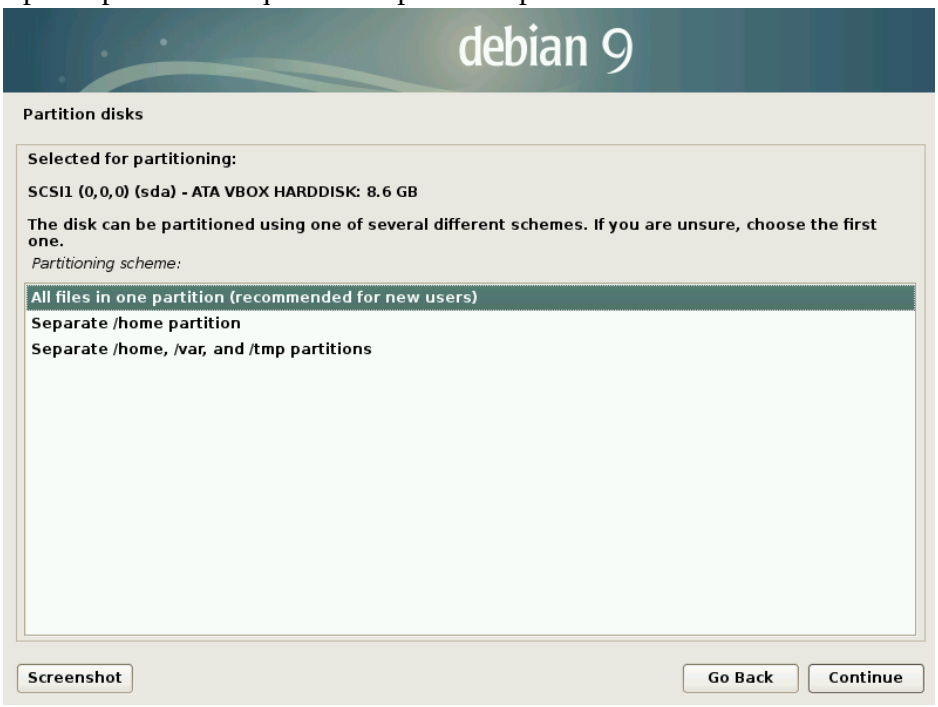
7. Se empiezan a cargar los componentes necesarios para la instalación.

8. Seleccionamos la opción Guided-use entire disk la cual borrara todo el disco duro.

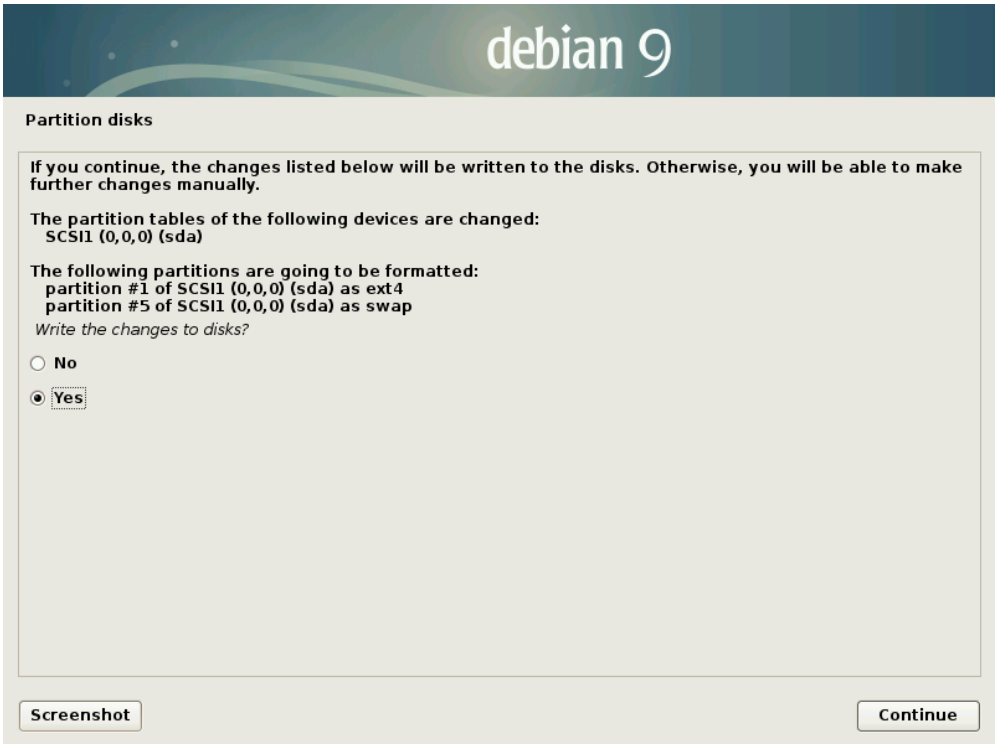


9. Elegimos la unidad en donde queremos que se realice la instalación.

10. Seleccionamos el número de particiones que vamos a realizar, en este caso dejamos la opción por defecto que es una partición para todos los archivos.

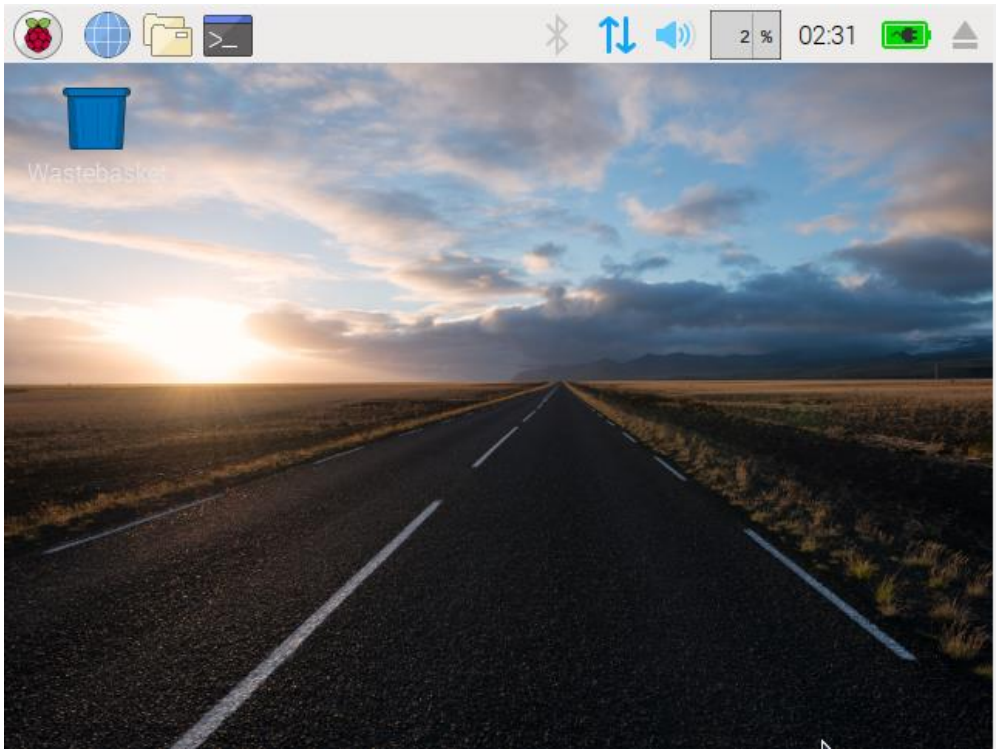


11. Damos clic en Yes si estamos de acuerdo con las particiones que se van a crear.



12. Se empieza el proceso de instalación, esto puede tardar algunos minutos.

13. Una vez finalizado el proceso de instalación nos mostrara la ventana de inicio de Raspbian.


2 %
02:31
Wastebasket

ANEXO2

Guia de instalacion de OpenCV 3 en Raspberry Pi 3

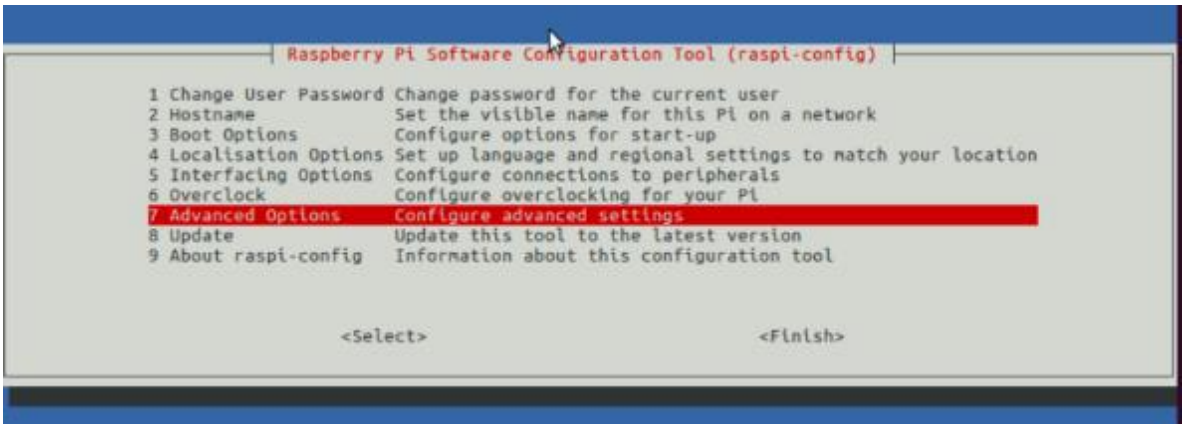
La presente guia muetra el proceso para instalar el software de vision artificial OpenCV 3.

1. Expandir el sistema de archivos

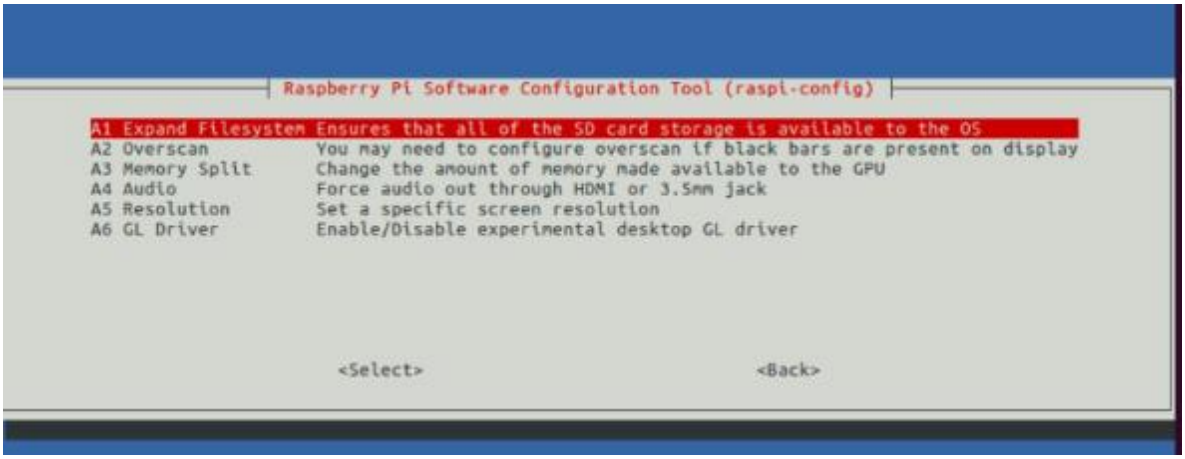
Lo primero que se debe hacer es expandir su sistema de archivos para incluir todo el espacio disponible en la tarjeta micro-SD:

$ sudo raspi-config

Y luego se selecciona la opción de menú "Opciones avanzadas":



A continuación, se selecciona "Expandir sistema de archivos":



Se debe seleccionar la primera opción, "A1. Expandir sistema de archivos “, presione Enter en su teclado, flecha hacia abajo hasta el botón " <Finish> " , y luego reinicie su Pi - es posible que se le solicite que reinicie, pero si no lo está, puede ejecutar:

$ sudo reboot
Después de reiniciar, su sistema de archivos debería haberse expandido para incluir todo el espacio disponible en su tarjeta micro-SD. Puede verificar que el disco se haya expandido ejecutando df - h y examinando la salida:

```
pi@raspberry:~ $ df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            227M     0  227M   0% /dev
tmpfs            50M  1.8M   48M   4% /run
/dev/sda1       7.4G  4.6G  2.4G  66% /
tmpfs           247M     0  247M   0% /dev/shm
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
tmpfs           247M     0  247M   0% /sys/fs/cgroup
tmpfs            50M     0   50M   0% /run/user/1000
pi@raspberry:~ $
```

Como puede ver, mi sistema de archivos Raspbian se ha expandido para incluir los 16 GB de la tarjeta micro-SD.
Sin embargo, incluso con mi sistema de archivos expandido, ya he usado el 15% de mi tarjeta de 16GB.
Si está utilizando una tarjeta de 8 GB, es posible que esté utilizando cerca del 50% del espacio disponible, por lo que una cosa simple es eliminar el motor de LibreOffice y Wolfram para liberar espacio en su Pi:

```
1 $ sudo apt-get purge wolfram-engine
2 $ sudo apt-get purge libreoffice*
3 $ sudo apt-get clean
4 $ sudo apt-get autoremove
```

2. Instalar dependencias

El primer paso es actualizar y actualizar cualquier paquete existente:

```
pi@raspberry:~ $ sudo apt-get update && sudo apt-get upgrade
```

Luego necesitamos instalar algunas herramientas de desarrollador, incluido CMake , que nos ayudan a configurar el proceso de construcción de OpenCV:

```
pi@raspberry:~ $ sudo apt-get install build-essential cmake pkg-config
```

A continuación, debemos instalar algunos paquetes de E / S de imágenes que nos permiten cargar varios formatos de archivos de imágenes desde el disco. Ejemplos de tales formatos de archivo incluyen JPEG, PNG, TIFF, etc.

```
pi@raspberry:~ $ sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev
```

Al igual que necesitamos paquetes de E / S de imágenes, también necesitamos paquetes de E / S de video. Estas bibliotecas nos permiten leer varios formatos de archivo de video desde el disco, así como trabajar directamente con secuencias de video:

```
pi@raspberry:~ $ sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev
```

La biblioteca OpenCV viene con un submódulo llamado highgui que se usa para mostrar imágenes en nuestra pantalla y crear interfaces gráficas de usuario básicas. Para compilar el módulo highgui , necesitamos instalar la biblioteca de desarrollo GTK:

```
pi@raspberry:~ $ sudo apt-get install libgtk2.0-dev libgtk-3-dev
```

Muchas operaciones dentro de OpenCV (es decir, operaciones matriciales) pueden optimizarse aún más instalando algunas dependencias adicionales:

```
pi@raspberry:~ $ sudo apt-get install libatlas-base-dev gfortran
```

Estas bibliotecas de optimización son *especialmente importantes* para dispositivos con recursos limitados, como la Raspberry Pi.
Por último, instalemos los archivos de encabezado de Python 2.7 y Python 3 para poder compilar OpenCV con enlaces de Python:

```
pi@raspberry:~ $ sudo apt-get install python2.7-dev python3-dev
```

Si está trabajando con una nueva instalación del sistema operativo, es posible que estas versiones de Python ya estén en la versión más reciente (verá un mensaje en el terminal que lo indica).

3. Descarga el código fuente de OpenCV

Ahora que tenemos nuestras dependencias instaladas, tomemos el archivo 3.3.0 de OpenCV del repositorio oficial de OpenCV.

```
pi@raspberry:~ $ cd
pi@raspberry:~ $ wget -O opencv.zip https://github.com/Itseez/opencv/archive/3.3.0.zip
```

4. Instalar PIP

Antes de que podamos comenzar a compilar OpenCV en nuestra Raspberry Pi 3, primero necesitamos instalar pip , un administrador de paquetes de Python:

```
pi@raspberry:~ $ wget https://bootstrap.pypa.io/get-pip.py
pi@raspberry:~ $ sudo python get-pip.py
pi@raspberry:~ $ sudo python3 get-pip.py
```

Necesitamos actualizar nuestro archivo de perfil para incluir las siguientes líneas en la *parte inferior* del archivo:

```
pi@raspberry:~ $ export WORKON_HOME=$HOME/.virtualenvs
pi@raspberry:~ $ export VIRTUALENVWRAPPER_PYTHON=/usr/bin/python3
pi@raspberry:~ $ source /usr/local/bin/virtualenvwrapper.sh
```

```
pi@raspberry:~ $
pi@raspberry:~ $ echo -e "\n# virtualenv and virtualenvwrapper" >> ~/.profile
pi@raspberry:~ $ echo "export WORKON_HOME=$HOME/.virtualenvs" >> ~/.profile
pi@raspberry:~ $ echo "export VIRTUALENVWRAPPER_PYTHON=/usr/bin/python3" >> ~/.profile
pi@raspberry:~ $ echo "source /usr/local/bin/virtualenvwrapper.sh" >> ~/.profile
pi@raspberry:~ $
```

Ahora que tenemos nuestro ~ /. perfil actualizado, necesitamos volver a cargarlo para asegurarnos de que los cambios surtan efecto. Puedes forzar una recarga de tu ~ /. archivo de perfil por:

1. Cerrar sesión y luego volver a iniciar sesión.
2. Cerrar una instancia de terminal y abrir una nueva.

Creando tu entorno virtual Python

A continuación, vamos a crear el entorno virtual de Python que usaremos para el desarrollo de la visión artificial:

```
pi@raspberry:~ $ mkvirtualenv cv -p python2
```

Este comando creará un nuevo entorno virtual de Python llamado cv usando ***Python 2.7***.
Si, en cambio, desea usar ***Python 3***, querrá usar este comando en su lugar:

```
pi@raspberry:~ $ mkvirtualenv cv -p python3
```

Cómo comprobar si estás en el entorno virtual "cv"

El comando mkvirtualenv está diseñado para ejecutarse solo una vez: para *crear* realmente el entorno virtual.
Después de eso, puede usar workon y se lo colocará en su entorno virtual:

```
pi@raspberrypi:~ $ workon cv
(cv) pi@raspberrypi:~ $
```

Instalando NumPy en tu Raspberry Pi
Nuestra única dependencia de Python es NumPy , un paquete de Python utilizado para el procesamiento numérico:

```
pi@raspberry:~ $ pip install numpy
```

5. Compilar e instalar OpenCV

Una vez que se haya asegurado de estar en el entorno virtual cv, podemos configurar nuestra compilación utilizando CMake:

```
pi@raspberry:~ $ cd ~/opencv-3.3.0/
pi@raspberry:~ $ mkdir build
pi@raspberry:~ $ cd build
pi@raspberry:~/build $ cmake -D CMAKE_BUILD_TYPE=RELEASE \
>     -D CMAKE_INSTALL_PREFIX=/usr/local \
>     -D INSTALL_PYTHON_EXAMPLES=ON \
>     -D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib-3.3.0/modules \
>     -D BUILD_EXAMPLES=ON ..
```

Configure el tamaño de su espacio de intercambio antes de compilar
Antes de comenzar el proceso de compilación, debe aumentar el tamaño de su espacio de intercambio. Esto permite que OpenCV compile con los cuatro núcleos de la Raspberry PI sin que la compilación se bloquee debido a problemas de memoria.
Abra su / etc / dphys - swapfile y luego edite la variable CONF_SWAPSIZE:

```
pi@raspberry:~/build $ cat /etc/dphys-swapfile
# /etc/dphys-swapfile - user settings for dphys-swapfile package
# author Neil Franklin, last modification 2010.05.05
# copyright ETH Zuerich Physics Departement
#   use under either modified/non-advertising BSD or GPL license

# this file is sourced with . so full normal sh syntax applies

# the default settings are added as commented out CONF_*=* lines


# where we want the swapfile to be, this is the default
#CONF_SWAPFILE=/var/swap

# set size to absolute value, leaving empty (default) then uses computed value
#   you most likely don't want this, unless you have an special disk situation
CONF_SWAPSIZE=100

# set size to computed value, this times RAM size, dynamically adapts,
#   guarantees that there is enough swap without wasting disk space on excess
#CONF_SWAPFACTOR=2

# restrict size (computed and absolute!) to maximally this limit
#   can be set to empty for no limit, but beware of filled partitions!
#   this is/was a (outdated?) 32bit kernel limit (in MBytes), do not overrun it
#   but is also sensible on 64bit to prevent filling /var or even / partition
#CONF_MAXSWAP=2048
```

Observe que he comentado la línea de 100 MB y he agregado una línea de 1024 MB. Este es el secreto para compilar con múltiples núcleos en el Raspbian Stretch.

Para activar el nuevo espacio de intercambio, reinicie el servicio de intercambio:

```
pi@raspberry:~/build $ sudo /etc/init.d/dphys-swapfile stop
[ ok ] Stopping dphys-swapfile (via systemctl): dphys-swapfile.service.
pi@raspberry:~/build $ sudo /etc/init.d/dphys-swapfile start
[ ok ] Starting dphys-swapfile (via systemctl): dphys-swapfile.service.
```

Finalmente, ahora estamos listos para compilar OpenCV:

```
pi@raspberry:~ $ make -j4
```

Una vez que OpenCV 3 haya terminado de compilar, todo lo que debe hacer es instalar OpenCV 3 en su Raspberry Pi 3:

```
pi@raspberry:~ $ sudo make install
```

```
pi@raspberry:~ $ sudo ldconfig
```

6. Probando tu instalación de OpenCV 3

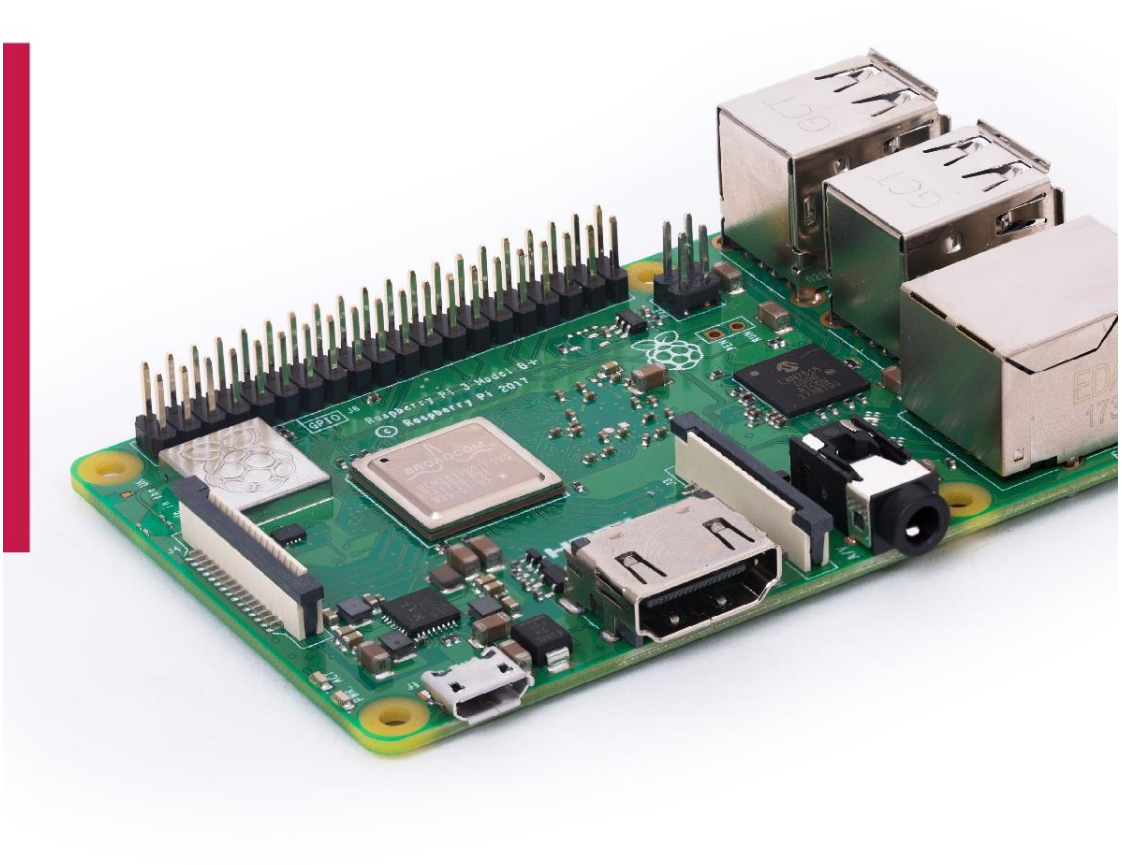
Abra un nuevo terminal, ejecute los comandos source y workon , y finalmente intente importar los enlaces de Python + OpenCV:

```
(cv) pi@raspberrypi:~ $ workon cv
(cv) pi@raspberrypi:~ $ python3
Python 3.5.3 (default, Sep 27 2018, 17:25:39)
[GCC 6.3.0 20170516] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> cv2.__version__
'3.4.3'
>>>
```

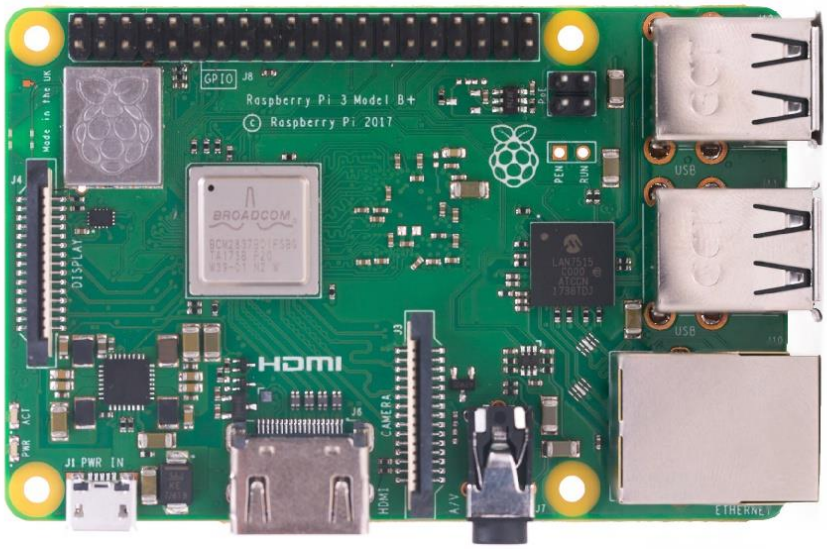
Como puede ver en la captura de pantalla de mi propio terminal, OpenCV 3 se instaló correctamente en mi entorno Raspberry Pi 3 + Python 3.5.

ANEXO 3

Raspberry Pi 3 Model B+



Overview



The Raspberry Pi 3 Model B+ is the latest product in the Raspberry Pi 3 range, boasting a 64-bit quad core processor running at 1.4GHz, dual-band 2.4GHz and 5GHz wireless LAN, Bluetooth 4.2/BLE, faster Ethernet, and PoE capability via a separate PoE HAT

The dual-band wireless LAN comes with modular compliance certification, allowing the board to be designed into end products with significantly reduced wireless LAN compliance testing, improving both cost and time to market.

The Raspberry Pi 3 Model B+ maintains the same mechanical footprint as both the Raspberry Pi 2 Model B and the Raspberry Pi 3 Model B.

Specifications

Processor: Broadcom BCM2837B0, Cortex-A53 64-bit SoC @ 1.4GHz

Memory: 1GB LPDDR2 SDRAM

Connectivity:
- 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE
- Gigabit Ethernet over USB 2.0 (maximum throughput 300Mbps)
- 4 × USB 2.0 ports

Access: Extended 40-pin GPIO header

Video & sound:
- 1 × full size HDMI
- MIPI DSI display port
- MIPI CSI camera port
- 4 pole stereo output and composite video port

Multimedia: H.264, MPEG-4 decode (1080p30); H.264 encode (1080p30); OpenGL ES 1.1, 2.0 graphics

SD card support: Micro SD format for loading operating system and data storage

Input power:
- 5V/2.5A DC via micro USB connector
- 5V DC via GPIO header
- Power over Ethernet (PoE)–enabled (requires separate PoE HAT)

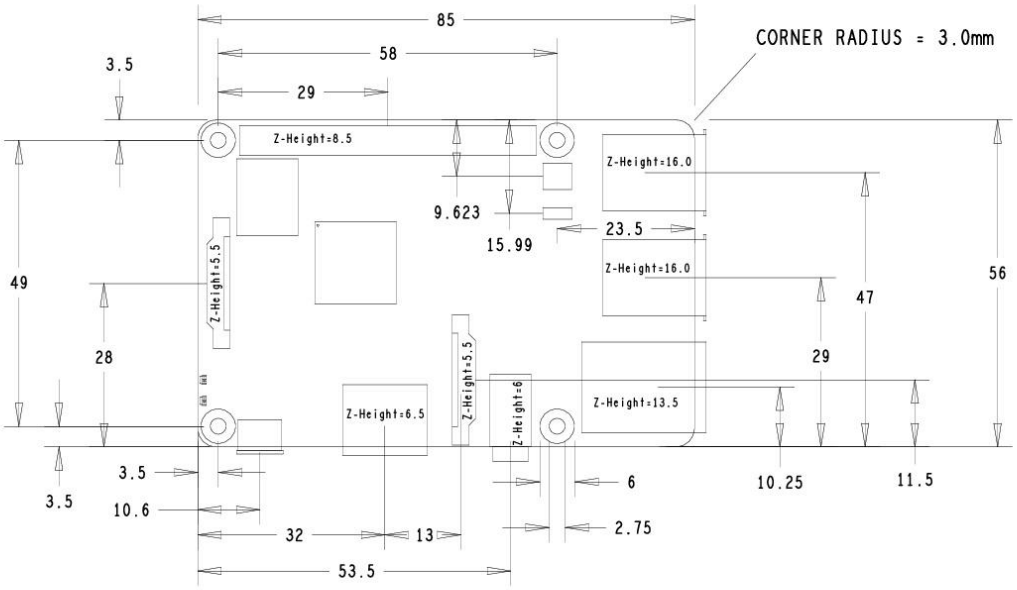
Environment: Operating temperature, 0–50°C

Compliance: For a full list of local and regional product approvals, please visit www.raspberrypi.org/products/raspberry-pi-3-model-b+

Production lifetime: The Raspberry Pi 3 Model B+ will remain in production until at least January 2023.

raspberrypi.org

Physical specifications



Warnings

- This product should only be connected to an external power supply rated at 5V/2.5A DC. Any external power supply used with the Raspberry Pi 3 Model B+ shall comply with relevant regulations and standards applicable in the country of intended use.
- This product should be operated in a well-ventilated environment and, if used inside a case, the case should not be covered.
- Whilst in use, this product should be placed on a stable, flat, non-conductive surface and should not be contacted by conductive items.
- The connection of incompatible devices to the GPIO connection may affect compliance, result in damage to the unit, and invalidate the warranty.
- All peripherals used with this product should comply with relevant standards for the country of use and be marked accordingly to ensure that safety and performance requirements are met. These articles include but are not limited to keyboards, monitors, and mice when used in conjunction with the Raspberry Pi.
- The cables and connectors of all peripherals used with this product must have adequate insulation so that relevant safety requirements are met.

Safety instructions

To avoid malfunction of or damage to this product, please observe the following:

- Do not expose to water or moisture, or place on a conductive surface whilst in operation.
- Do not expose to heat from any source; the Raspberry Pi 3 Model B+ is designed for reliable operation at normal ambient temperatures.
- Take care whilst handling to avoid mechanical or electrical damage to the printed circuit board and connectors.
- Whilst it is powered, avoid handling the printed circuit board, or only handle it by the edges to minimise the risk of electrostatic discharge damage.