Universidad Autónoma de Querétaro
Facultad de Ingeniería

Ingeniería en Automatización

Sistema de seguridad basado en Raspberry Pi con cámara para día y noche con enlace IoT y procesamiento de imágenes

Tesis

Que como parte de los requisitos para obtener el grado de ingeniero en automatización

Presenta:
Aldo Francisco Muñoz Vargas

Dirigida por:
Dr. Juan Manuel Ramos Arreguín

C.U. Santiago de Querétaro, Qro. Octubre de 2021

Universidad Autónoma de Querétaro
Facultad de Ingeniería

Ingeniería en Automatización

Sistema de seguridad basado en Raspberry Pi con cámara para día y noche con enlace IoT y procesamiento de imágenes

Tesis

Que como parte de los requisitos para obtener el grado de ingeniero en automatización

Presenta:

Aldo Francisco Muñoz Vargas

Firma

Sinodales:

Dr. Juan Manuel Ramos Arreguín
Presidente

Firma

Dr. Jesús Carlos Pedraza Ortega
Secretario

Firma

Dr. Edgar Alejandro Rivas Araiza
Vocal

Firma

Dr. Saul Tovar Arriaga
Suplente

Firma

C.U. Santiago de Querétaro, Qro. Octubre de 2021

Dedicatoria

Dedico esta tesis a mi madre.

Agradecimientos

Quisiera agradecer a las siguientes personas que me han ayudado a llevar a cabo este desarrollo:

Mis supervisores Dr. Juan Manuel Ramos Arreguín y Dr. Jesús Carlos Pedraza Ortega por su entusiasmo por el proyecto, su apoyo y paciencia. Gracias por las oportunidades de trabajar en proyectos como este.

A la Universidad Autónoma de Querétaro por proveer de este programa de estudios.

Al coordinador de la carrera de Ingeniería en Automatización M.C. José Luis Avendaño Juárez por su constante cuidado por la carrera y su habilidad para enseñar a lo largo de varias materias.

A mis sinodales por tomarse el tiempo de leer mi tesis dando una retroalimentación enriquecedora para mejorar el proyecto.

A mi madre quien siempre se preocupó porque tuviera lo necesario para tener éxito en una carrera tan laboriosa, impulsándome y ayudándome siempre que pudo.

A todo aquel que se involucró y me acompañó en este viaje. Quisiera expresar mi más sincera gratitud.

Abstract

This development proposes a security system based on an embedded system capable of detecting movement through the use of an infrared sensor, a camera and image processing. Moreover, this system can operate with visible light and in low-light conditions, in this last one by using an infrared lamp and a camera able to see this type of light. The surveillance system divides into an Android application, storage in the Mega cloud, and a Raspberry Pi 3 B+. The application allows the user to interact and go over the data generated by the system, as long as an internet connection is established. On the other hand, the embedded system integrates the surveillance hardware, regarded to the infrared camera and the movement sensor. Besides that, an image-processing algorithm capable of detecting movement was programmed, by using Python 3.5.3 with the OpenCV 3.4 library. This algorithm for ensuring that all the information captioned by the camera guarantees that what was captured corresponded to a change in the range of the system. Finally, a case is proposed where the overall system is mounted. This case is a self-design and was printed in 3D to be installed indoors, making this project a potential option for surveillance appliances.

Keywords: Infrared Camera, Raspberry Pi, Android, Infrared Sensor, Internet of Things (IoT), OpenCV, Image Processing, Movement Detection.

Resumen

Este desarrollo propone un sistema de seguridad basado en un sistema embebido, capaz de detectar movimiento mediante el uso de un sensor infrarrojo, una cámara y procesamiento de imágenes. Además, el sistema es capaz de operar en condiciones de luz visible y obscuridad, en esta última mediante el uso de una lámpara y cámara infrarroja. El sistema de vigilancia se divide en una aplicación móvil para Android, almacenamiento en la nube Mega y el uso de una Raspberry Pi 3 B+. Con la aplicación móvil, el usuario puede configurar y revisar los datos generados por el sistema en cualquier momento, mientras tenga una conexión a internet. Por otra parte, el sistema embebido integra el hardware de vigilancia siendo la cámara infrarroja y un sensor de movimiento. Además, se programó un algoritmo de procesamiento de imágenes para detección de movimiento, usando Python 3.5.3 y se usa la librería OpenCV 3.4 para el procesamiento de las imágenes. Este último, para asegurar la información capturada por la cámara corresponde con algún cambio en el rango del sistema. Finalmente, se propone una carcasa donde es montado el sistema completo. La carcasa es un diseño propio y fue impresa en 3D, para interiores, haciendo de este proyecto una opción potencial para aplicaciones de vigilancia.

Palabras clave: Cámara infrarroja, Raspberry Pi, Android, sensor infrarrojo, Internet de las cosas (IoT), OpenCV, procesamiento de imágenes, detección de movimiento.

Índice

Índice de figuras

Índice de tablas

1. Introducción

Desafortunadamente, México es muy conocido por tener un problema con todo tipo de inseguridad. Desde 2010, la tasa de robos de viviendas por cada 100,000 habitantes ha aumentado de 2,114 casos a 2,598 casos en 2018 [1]. Específicamente, el robo es un tipo de delito que puede evitarse cuando se observa al ladrón tentativo, esto de acuerdo con la teoría de actividades rutinarias [2]. Sin embargo, si el robo ocurre de todos modos, tener un dispositivo que indique el momento en que ocurre puede hacer la diferencia en el proceso de justicia.

Por otro lado, los crecientes dispositivos para el IoT han ido creciendo mucho y están presentes en este proyecto, tomando ese concepto de comunicación que ha evolucionado de humano-humano, humano-cosa, y finalmente entre cosas con cosas [3].

Un sistema Raspberry Pi está pensado para que los estudiantes programen en cursos, aunque se ha convertido en mucho más que eso. Sobre todo, dejando margen al daño del sistema por descuidos por el bajo costo que tiene. Este es un punto realmente importante para un sistema de seguridad que es destinado a funcionar durante largas y varias horas haciendo más probable su avería. Además, la cantidad de IO (Entradas y Salidas) es bastante importante y hace que esta placa sea perfecta para este tipo de proyectos [4].

Algunas aplicaciones relacionadas con este desarrollo han utilizado sistemas integrados como Raspberry Pi, debido al bajo costo y a las excelentes opciones versátiles, como la compatibilidad de cámaras, entradas y salidas digitales, y sus capacidades informáticas como un sistema Linux [5].

1.1. Justificación

El problema de inseguridad actual conlleva al desarrollo de sistemas de vigilancia versátiles, capaces de observar, registrar y dar a conocer al usuario lo que es observado, a través de una aplicación móvil. Una aplicación móvil es ideal al ofrecer a la mayoría de personas la capacidad de estar al tanto de alguna situación, sin necesidad de utilizar sistemas fijos, a los que solo se puede acceder desde un único sitio, tales como los que se pueden observar en algunas escuelas, centros comerciales o industrias. Estos sistemas suelen grabar video en todo momento, lo cual significa que los costos tanto de procesamiento, energía, almacenamiento y dinero se incrementan significativamente en comparación con lo que se plantea en este desarrollo.

Al ser la aplicación enfocada a hogares o sitios específicos donde se desea que el sistema de seguridad solo opere en momentos que el usuario lo desee, no hace falta que se capture video si se detecta movimiento, solo se toman imágenes. Esta decisión,

con base en que el costo de almacenamiento puede ser muy alto al usar una memoria SD espacio en la nube, se plantea sólo la captura de imágenes, para utilizar la menor cantidad de recursos. Además, en caso de que el sistema de seguridad estuviera en funcionamiento y se detectase movimiento, la captura de imágenes debería ser suficiente para que el usuario pudiera decidir si la situación implica tomar alguna medida.

Debido a las condiciones bajo las que se opera un sistema como este, es conveniente que sea capaz de observar y detectar movimiento, bajo circunstancias que van desde poca o nula iluminación, hasta la luz del día. Por lo anterior, una cámara sin filtro infrarrojo es utilizada en este proyecto. Este tipo de cámaras pueden captar la luz infrarroja, la cual no es visible al ojo humano, en condiciones de poca iluminación. En condiciones contrarias, el único problema es que los colores que ofrecen las imágenes capturadas no son muy fieles a la realidad. Idealmente, se requeriría de una lámpara que emitiera luz de este tipo si la iluminación fuera nula. Sin embargo, aún sin esta iluminación mientras haya algo de luz ambiental, las imágenes que esta cámara puede tomar son mucho mejores que las que se podrían capturar con una cámara común.

Además, el uso de visión por computadora hace que un prototipo como este sea mucho más confiable, ya que mediante un algoritmo de detección de movimiento solo se tomará información relevante. Lo anterior debido a que el sensor infrarrojo sería solo la primera fuente de alerta del sistema, activando un algoritmo que desencadena el análisis correspondiente para determinar si la imagen debe ser almacenada y enviada a la nube, asegurando que la menor cantidad de recursos sean utilizados.

Por otro lado, algunos sistemas embebidos usan sistemas operativos como Linux, de tal manera que permiten la programación de aplicaciones de manera directa. En este proyecto se aprovecha tal característica. Esto va a permitir futuras mejoras que el usuario desee implementar, las cuales pueden ir desde el uso de otro tipo de cámaras, sensores, hasta su posible transformación de un centro directo de comunicación con otro tipo de dispositivos inteligentes dentro de un lugar.

1.2. Importancia del tema

Un sistema de seguridad como este contribuye al área de seguridad, que cada vez es más importante. Además, este proyecto puede sembrar las bases de un producto comercializable que podría generar ingresos.

Por otro lado, este desarrollo, al estar en un sistema que puede ser mejorado continuamente, debido al tipo de hardware y software utilizado, podrían aplicarse diferentes algoritmos para detección de movimiento, aplicando visión por computadora.

Desde el punto de vista de la carrera de Ingeniería en Automatización, este tipo de integración implica la aplicación de conocimientos adquiridos relacionados a sistemas embebidos, electrónica, instrumentación, programación y tecnología de

materiales. Así mismo, el desarrollo de nuevas habilidades de investigación respecto al uso de las tecnologías descritas, hasta la documentación correcta y concisa del desarrollo tecnológico. Todo esto, con el objetivo de obtener el grado de ingeniería y abrir la posibilidad de seguir mejorando el proyecto.

1.3. Descripción del problema

Desde 2010 la tasa de robo a casa habitación en México, por cada 100 mil habitantes ha ido incrementando, desde 2114 casos, hasta 2598 casos en 2018 [1]. Esto ha desarrollado una constante amenaza a ser robado en cualquier momento, mientras vivimos en comunidad, aplicándose a hogares, lugares de trabajo, la calle, entre otros. Debido a lo anterior, es necesario desarrollar sistemas que sirvan para vigilar determinadas zonas, ya sea en casa o en la Universidad Autónoma de Querétaro, UAQ.

Especialmente en la UAQ, se ha tenido el problema de robos de equipos y materiales. Debido a lo anterior, con este proyecto se pretende desarrollar un sistema de vigilancia basado en un sistema embebido Raspberry Pi; usando un sensor de presencia PIR, una cámara con capacidad para visión nocturna a través de luz infrarroja; y visión por computadora mediante un algoritmo de detección de movimiento. El propósito es vigilar una determinada zona y detectar movimiento en horarios no hábiles, dando la posibilidad de consultar imágenes de lo acontecido usando tecnología de Internet de las Cosas (IoT) desde una aplicación desarrollada en Android.

1.4. Hipótesis

El uso de tecnologías de sistemas embebidos, combinados con las tecnologías de Internet de las Cosas, pueden permitir el desarrollo de sistemas de vigilancia con capacidad de enviar información a la nube cuando se detecte actividad en un lugar específico, y que pueda ser consultada desde un dispositivo móvil en cualquier lugar.

1.5. Objetivo general

Desarrollar un sistema de vigilancia para detección de personas no autorizadas en áreas específicas y de esta manera la incidencia de robos [2] [6], al usar tecnología de sistemas embebidos, de detección de presencia, IoT y visión por computadora.

1.6. Objetivos específicos

a) Establecer los requerimientos funcionales del sistema de acuerdo al contexto de aplicación.

b) Realizar un estudio comparativo de diferentes sistemas embebidos para justificar el uso de una Raspberry Pi 3 B+ analizando otras posibles opciones que se podrían tomar para un desarrollo como este, comparando características de acuerdo a la necesidad de vigilancia tales como, capacidad

para montar un sistema operativo, capacidad de procesamiento, entradas y salidas, costo, consumo energético e información disponible sobre cómo utilizar el dispositivo.

c) Implementar un algoritmo capaz de tomar imágenes de manera automática cuando ocurra un evento con el detector de movimiento en un horario deseado, para almacenar imágenes de lo acontecido y enviarlas a la nube utilizando comunicación inalámbrica.

d) Desarrollar una interfaz de usuario utilizando *Android Studio*, para recuperar la información generada por el sistema de seguridad, así como controlar su funcionamiento.

e) Implementar un algoritmo que incluya visión por computadora mediante el uso de OpenCV en Python para hacer que la confiabilidad del sistema incremente con respecto al mismo sistema sin un algoritmo como este.

f) Realizar pruebas de funcionamiento del prototipo para evaluar el desempeño del sistema.

g) Diseñar y elaborar una carcasa que contenga el hardware necesario para el prototipo, a excepción de la fuente de alimentación, utilizando tecnología de impresión 3D.

1.7. Alcances y limitaciones

Este prototipo se propone desarrollarlo mediante el uso de una *Raspberry Pi 3 B+*, una cámara *Pi Noir V2* para captura de imágenes en condiciones de buena y poca iluminación, un sensor de infrarrojo (*PIR*) con un alcance máximo de 15m y una carcasa impresa en 3D de diseño propio con la opción de montarse en una pared, un techo o simplemente un escritorio, considerando estar alejado del agua y tener una línea eléctrica de corriente alterna cerca para hacer que el sistema funcione. Además, se requiere una conexión inalámbrica a Internet, para la comunicación con el usuario.

El usuario requiere un teléfono con al menos Android 6.0 y la aplicación que le permita controlar y verificar el sistema en cualquier momento mientras tenga una conexión a Internet local. Cada imagen capturada deberá ser almacenada en el servicio de una nube de terceros (*MEGA*) con la fecha y hora exacta de captura, de modo que, si algo destruye el sistema de seguridad, la información permanece.

1.8. Consideraciones éticas y buenas prácticas de trabajo

Es importante señalar la capacidad que un proyecto como este podría tener de usarse con fines no deseados, como violar la privacidad de otros, por ello, de acuerdo con la Asociación para Maquinaria de Computación (ACM) a continuación se mencionan algunos puntos que se tendrán como importantes bastiones para un profesional [7]:

- El bien de los demás siempre será la primera prioridad en un proyecto como este.
- Se debe contribuir a la sociedad y bienestar humano, reconociendo que todos son importantes para un proyecto de esta naturaleza.
- Se evita el daño a los demás a toda costa.
- Se debe respetar la privacidad de todos los individuos que hayan tenido o no contacto con el prototipo.
- Respetar la confidencialidad de cualquier información capturada es primordial en este desarrollo.
- Se accederá a recursos tales como espacios de trabajo o redes de comunicación solo cuando esté autorizado o sea por un bien público.

Se debe mencionar que el alcance del proyecto no es la implementación y puesta en marcha del sistema. Solamente es el diseño de un prototipo y la demostración de que este funciona correctamente. Quien lo desee implementar deberá atender la parte de resguardo de la información y privacidad de las personas que en su momento aparezcan en las fotos.

Estos fueron algunos importantes puntos a considerar para respetar el código de conducta y ética del ACM que podrían tener una enorme relación con el desarrollo e implementación de un prototipo como este.

2. Revisión de literatura

2.1. Antecedentes

A continuación, se mencionan varios ejemplos que podrían ser importantes de mencionar sobre cómo han evolucionado este tipo de trabajos.

Un antecedente específico sobre el uso de sensores piroeléctricos *PIR* ocurrió en 2004, cuando se analizó su funcionamiento para el monitoreo de ocupación de sitios específicos, mediante un algoritmo capaz de manejar la información proveída por los sensores, y determinar la presencia humana en un determinado espacio [8].

En 2014, Sanjana [9] desarrolló un sistema de Monitoreo de Vigilancia Inteligente aplicando estas tecnologías combinadas con sensores PIR mediante la captura de video cuando se detectaba algún movimiento, enviándolo por una red 3G a un teléfono inteligente mediante una aplicación web. Otro buen ejemplo es un sistema de seguridad basado en *Raspberry Pi* con capacidad de visión nocturna que utiliza *Open Source Computer Vision* (*OpenCV*) con detección de humanos y humo, lo que lo hace adecuado para instalaciones de almacén en cuanto a posibles delitos o incendios [10].

En 2015, Cocorullo desarrolla un sistema de vigilancia basado en la diferencia de fondo, y es implementado en una tarjeta Raspberry Pi [11]. El sistema es robusto a pesar de las condiciones ambientales externas. Por otro lado, Vaydia presenta un sistema de vigilancia embebido con captura automática de imagen y envío por correo electrónico, ahorrando costos de almacenamiento y de energía [12]. Así mismo, Chandana presenta un sistema de vigilancia inteligente basado en Raspberry Pi, donde al detectarse un movimiento, automáticamente la cámara toma una foto y es enviada por correo electrónico; se usa como base la plataforma Thing Speak para el uso de características IoT [13].

En 2016 Hossen desarrollan un sistema de vigilancia basado en detección y estimación de movimiento usando flujo óptico [14], basado en el algoritmo Horn-Schunck, resultando en un sistema computacionalmente rápido sin requerir de hardware especial para el procesamiento de las imágenes. También se presenta un sistema de vigilancia por parte de Jayakumar y Muthulakshmi [15], un sistema embebido implementado en una tarjeta Raspberry Pi y usando IoT para almacenamiento de fotografías y envío de alertas; se usa detección de rostros para saber si la persona está autorizada o no. Yang presentan un sistema de vigilancia para interiores, con la capacidad de detección múltiple de personas, seguimiento y análisis de comportamiento [16].

Asimismo, en 2017 utilizando los mismos elementos (*Raspberry Pi*, sensores infrarrojos y una cámara) se desarrolló un prototipo de sistema de vigilancia para detectar movimiento en la puerta de una tienda cada 10 segundos, enviando un

correo electrónico en caso de movimiento con la imagen adjunta, haciendo saber al propietario de la situación [17].

En 2018 se desarrolló un sistema de seguridad basado en un sensor de contacto, un alambrado eléctrico, una alarma sonora, una cámara y una Raspberry Pi. Este consiste en el control de los elementos anteriores desde una aplicación en Android y en la cual es posible la visualización de forma remota del inmueble que la cámara vigila, dando así al usuario la oportunidad de tomar acciones y activar los elementos previamente mencionados si así lo considera, evitando o al menos retrasando la entrada de un posible delincuente [18].

En 2018, se continúa desarrollando sistemas de vigilancia basados en sistemas embebidos, como el trabajo de Jayakumar [15], donde se propone el desarrollo de un sistema de vigilancia usando una tarjeta Raspberry Pi y una cámara, que se basa en detección de movimiento de personas y detección de rostros, agregando seguimiento de la persona.

También en 2018 Pawlenka desarrolla un sistema de seguridad considerando el incremento de la automatización del día a día, al estar este basado en un microcontrolador de un solo chip y sensores de para detectar el acceso no autorizado a un área basados en contacto magnético, medición de monóxido de carbono, detección de movimiento y medición de temperatura y humedad; todo este conjunto de sensores justificados debido a que las casas inteligentes requieren de una manera de obtener información de su entorno. En este desarrollo un sensor PIR (detector de infrarrojos) fue utilizado como detector de movimiento, entre otros para el resto de los parámetros. El procesamiento de las señales fue realizado mediante una red de microcontroladores y comunicación con protocolos como RS232 y RS485 [19].

En 2019 se presentó un sistema de seguridad físico que utiliza internet de las cosas mediante una plataforma llamada *Zolertia Remote*, tratando de dar una alternativa al de vigilancia. Consta de tres secciones, una red de sensores inalámbrica, un servidor privado *MQTT* (*Message Queue Telemetry Transport*) y una aplicación *Android* [20].

En 2020 se llevó a cabo una aplicación similar a las anteriores en la que mediante una *Raspberry Pi* como servidor se controló un conjunto de cámaras a través de una aplicación en *Android*. Este sistema fue desarrollado de modo que, en caso de detección de cambio del entorno mediante el software *Motion*, se tomarían imágenes o videos guardándose en la memoria de almacenamiento interno micro SD de la *Raspberry* y enviando un correo de alerta o notificaciones en la aplicación móvil, siendo estos datos accesibles a través de la aplicación que consultaría los datos del servidor [21].

De igual forma, en 2020 como trabajo de Zong Chen, un sistema de seguridad basado en sensores de movimiento e identificación de rostros fue desarrollado para detectar actividad sospechosa y reportar al posible infractor, este desarrollo utiliza Python, una Raspberry Pi, una base de datos de rostros y un sensor piroeléctrico, siendo

probado en diferentes situaciones grabando video de las situaciones de prueba a las que se sometió el dispositivo. El procesamiento de imágenes se llevó a cabo utilizando Java Script [22].

Por otro lado, existen sistemas comerciales como es el caso de los dispositivos de la marca SEEDDARY (entre otras), los cuales suelen constar de una cámara con capacidad para visión diurna y nocturna, detección de movimiento, ajuste de ángulo de visión de la cámara dependiendo del modelo, grabación de video y vigilancia en tiempo real, así como opciones para consultar tales como son aplicaciones móviles y en el navegador [23]. Algo importante a destacar es que, aunque este tipo de sistemas es completamente funcional, estos son cerrados y no permiten la integración o experimentación con capacidades nuevas tales como podría hacerse mediante el uso de una librería como OpenCV o algún otro software, lo cual se busca en este desarrollo. Además, un sistema abierto como es en el que se desarrolló este prototipo, puede ser útil como sistema central, de monitoreo y posible control de factores ambientales dentro de un área específica.

3. Marco teórico

3.1. Sistemas embebidos

Este término es la intersección entre los componentes físicos y computacionales, donde su comportamiento está definido tanto por las partes cibernéticas como físicas, usualmente se tienen ciclos de retroalimentación donde los procesos físicos afectan los cálculos y viceversa. Esto se puede nombrar como sistema ciber físico (*CPS*) [24].

Hay tres partes principales que componen un CPS, la planta física que es la parte física del sistema, algo que no depende directamente de las computadoras o redes digitales, luego están las plataformas computacionales donde se computan las señales, y finalmente la red a través del cual el sistema se comunica. Dentro de este tipo de sistemas se pueden clasificar tres granes grupos, aquellos que montan un sistema operativo tales como Raspberry Pi, Beagle Board o Onion Omega; aquellos que se operan únicamente mediante un programa cargado al dispositivo como lo hacen microcontroladores como PIC, ESP o Arduino y aquellos basados plenamente en lógica booleana como las tarjetas reconfigurables FPGA [24].

3.2. El espectro electromagnético y la luz infrarroja

La luz visible, radiación ultravioleta y radiación infrarroja son algunos tipos de ondas electromagnéticas del espectro la cuales son perturbaciones periódicas que mantienen su forma mientras avanzan en el espacio. Los tipos de ondas electromagnéticas se diferencian entre sí como cualquier onda, debido a su frecuencia o su recíproco que es el periodo y su amplitud, además estas ondas poseen un campo eléctrico "E", magnético "H" y velocidad de la misma "c" [25].

La figura 1 muestra el espectro electromagnético, donde se puede apreciar el intervalo correspondiente a la luz visible, así como el infrarrojo.

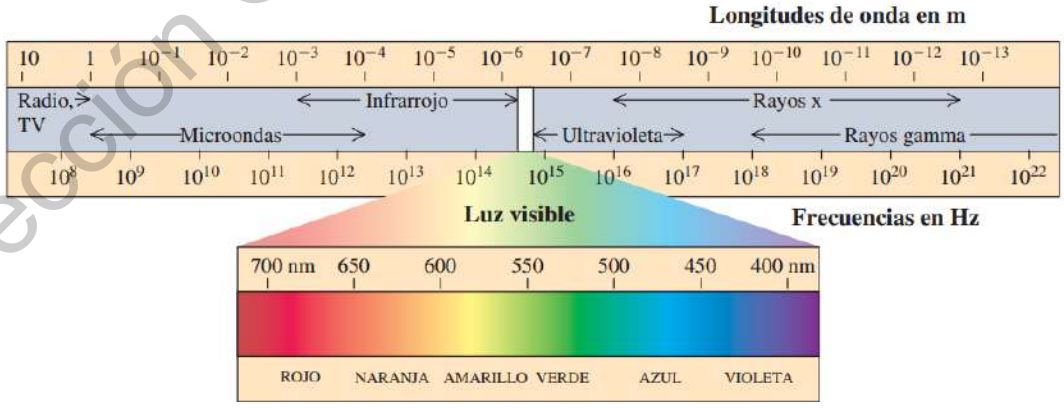


Figura 1. Espectro electromagnético [26].

El infrarrojo es radiación electromagnética con una longitud mayor que la luz visible, por tanto, con una frecuencia menor, se encuentra como lo indica su nombre, después del rojo y además antes de las microondas. Este espacio infrarrojo se extiende dentro del espectro desde los 700nm hasta los 1000µm. Aunque esta zona no es visible al ojo humano es posible detectarla mediante instrumentos. Debe señalarse que este tipo de radiación no es capaz de atravesar muchos materiales tales que la luz visible si puede atravesar tales como el vidrio, el agua o el plástico. Un cuerpo caliente también emite radiación, en su mayoría infrarroja, la emisión de radiación aumenta conforme aumenta la temperatura hasta que en cierto punto se alcanza la frecuencia de onda requerida para que un objeto empiece a emitir luz visible [27].

3.3. Detectores de radiación infrarroja

Existen diversos tipos de detectores de esta radiación, uno de ellos son los detectores térmicos, estos absorben la radiación incidente, cambiando la temperatura del detector y permaneciendo en un estado de que tiende al equilibrio en función de los cambios de la radiación, tres de los más utilizados son: los bolómetros termopilas y detectores piroeléctricos. Para este caso se explorarán los detectores piroeléctricos. La figura 2 muestra una configuración de un detector piroeléctrico.

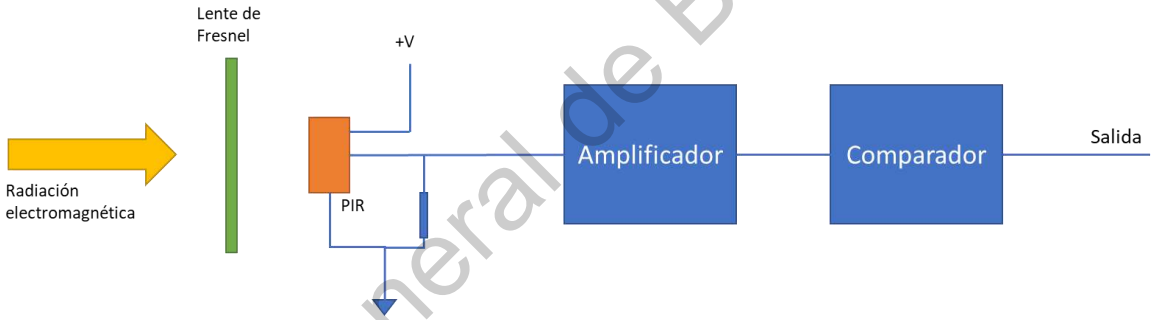


Figura 2. Configuración típica de un detector piroeléctrico [27].

3.4. Detectores piroeléctricos y fotodiodos

Están hechos de un material que genera una carga eléctrica cuando se expone a la radiación infrarroja, cuando la cantidad de radiación infrarroja cambia, también lo hace la carga, esto genera un voltaje que al ser amplificado puede tener diversas aplicaciones, en la figura 2 se observa un diagrama del funcionamiento. Suele utilizarse una lente de Fresnel la cual focaliza la radiación sobre el elemento primario [27]. A continuación, en la figura 3 se observa un sensor conocido como Sensor Piroeléctrico Infrarrojo *PIR* el cual consta de lo descrito anteriormente.

Figura 3. Sensor Piroeléctrico Infrarrojo PIR [28] [8].

Los fotodiodos por otro lado son semiconductores que generan un cambio de voltaje al existir un cambio en la radiación infrarroja recibida [27].

3.5. Cámaras y formación de imágenes

Una imagen se forma cuando una escena luminosa en 3D es proyectada sobre un plano 2D, las cámaras realizan este proceso; en una cámara cada punto de la escena se debe proyectar a un punto de la imagen dando así lugar a una imagen enfocada, lo anterior se muestra en la figura 4 [29].

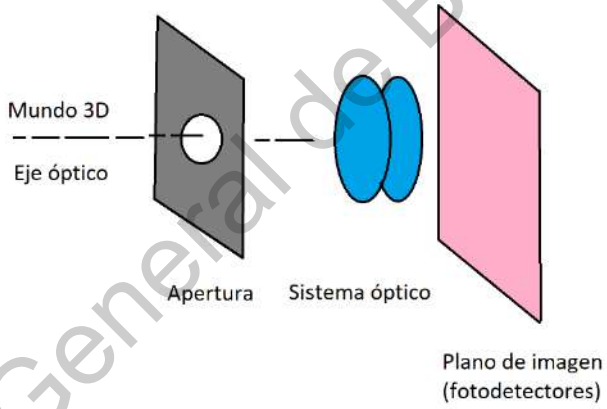


Figura 4. Modelo de cámara simplificado [29].

Algunas características sobre las imágenes se presentan a continuación [29]:

- Una imagen digital es una matriz de números en la que cada elemento de la matriz es un pixel.
- Una imagen digital es el muestreo discreto de una señal continua.
- Cada pixel representa el valor de una magnitud física como la cantidad de luz, cantidad de radiación en las longitudes de onda rojo, verde y azul, nivel de radiación fuera de espectro visible, profundidad entre otros.

En el proceso de toma de imágenes se hace uso de un proceso compuesto por captura y digitalización, donde una señal analógica es discretizada y enviada a como señal digital a un ordenador. Este proceso se muestra en la figura 5 [29].

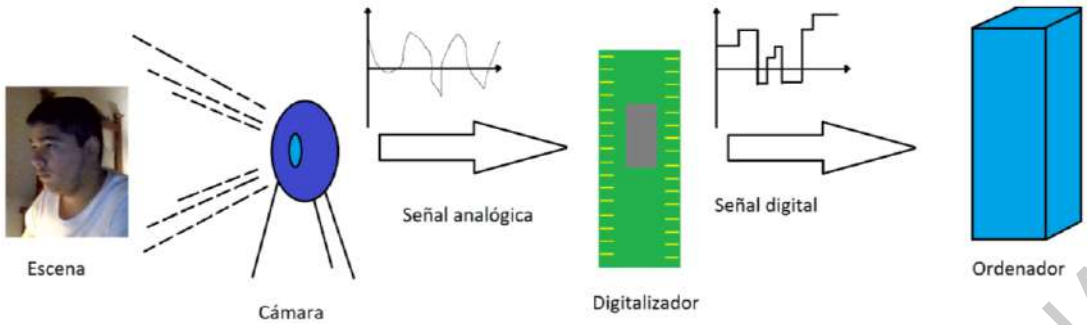


Figura 5. Proceso de captura y digitalización de imágenes [29].

Un dispositivo de captura determina los factores que tendrá la imagen final, tales como el tamaño de imagen, tamaño de pixel, el tipo de radiación que la cámara es capaz de captar entre otras [29].

Dentro de un dispositivo de captura en el filtro de color existen una mayor cantidad de pixeles verdes debido a que el ojo humano es más sensible a esta región del espectro [30] donde el arreglo de pixeles se observa en la figura 6.

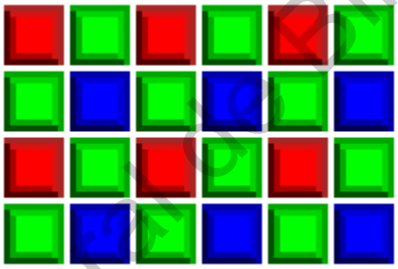


Figura 6. Distribución típica de un filtro de color [31].

La cámara NoIR utilizada en este prototipo, no posee un filtro infrarrojo haciendo posible que se detecte este tipo de luz [32]. En la figura 7 se observa una imagen tomada por la noche, usando solamente una lámpara infarroja, y en la figura 8 se muestra la imagen tomada con iluminación artificial.

Figura 7. Imagen capturada por la cámara NoIR con iluminación infrarroja.

Figura 8. Imagen capturada por la cámara NoIR con luz visible.

3.6. Internet de las cosas (IoT)

De acuerdo con la compañía Deloitte y Oracle, el internet de las cosas (Internet of Things) podría definirse como la agrupación e interconexión de dispositivos y objetos a través de una red de comunicación, donde todos estos podrían ser visibles e interaccionar entre sí. Los objetos o dispositivos podrían ser cualquiera, desde sensores hasta sistemas complejos como refrigeradores, procesos de producción, entre muchos otros. El objetivo es una conexión de máquina a máquina sin la intervención humana, conocida como M2M (Machine to Machine) [33] [34].

Combinar los sistemas embebidos con el Internet de las cosas (*IoT*) es algo muy útil en este proyecto, debido a que este tipo de objetos interconectados, equipados con algún tipo de capacidad computacional que requiere un vínculo con el mundo físico. A esto se les conoce también como un CPS [35].

Con base en lo anterior, se puede argumentar que un sistema como el propuesto, cumple con las características para ser considerado como un desarrollo dentro del internet de las cosas. Al tratarse de principalmente tres sistemas que intercambian información a través de una red, en este caso el teléfono celular, el sistema en la Raspberry Pi y la nube *Mega*, se puede tener un sistema M2M. La intervención del humano solo ocurre cuando se desea inicializar el sistema o configurarlo. Una vez activado, el sistema funciona de una manera autónoma.

3.7. Seguridad y conexiones con la nube

Además de las enormes oportunidades que brinda *IoT*, existen varios problemas y preocupaciones relacionados con los sistemas integrados en términos de sus vulnerabilidades de red, estos problemas se pueden prevenir mediante técnicas de cifrado en las que un mensaje puede leerse con una firma especial, estas técnicas son muy computacionalmente intensivas [36], de manera que, para evitar estos problemas de seguridad, se propone utilizar una nube de terceros para asignar todas esas cargas computacionales a otros, manteniendo al mismo tiempo almacenada y accesible la información recopilada por el sistema en cualquier momento y lugar.

Mega es un servicio en la nube que se propone para esta aplicación debido a la compatibilidad con Linux, en este servicio, los archivos se cifran antes de cargarlos y se descifran después de descargarlos utilizando una clave creada por el cliente, que se deriva de la contraseña. Utiliza un estándar de cifrado avanzado (*AES*), que es un algoritmo común para el cifrado de datos. Esto ofrece un buen nivel de seguridad con respecto a la información capturada por el sistema en caso de que alguien quiera robar o borrar el contenido. Es importante notar que existen varias vulnerabilidades sobre los datos que deja *Mega* mientras se usa en un navegador, afortunadamente, la aplicación para *Linux* no tiene ese problema [37] [38].

3.8. Visión por computadora

La manera en que los humanos perciben su entorno a través de la vista, es algo que se intenta replicar mediante técnicas matemáticas con el objetivo de recuperar una forma tridimensional en una imagen, se han tenido numerosos avances, sin embargo, aún se está lejos de ser comparable con lo que los seres humanos son capaces de percibir.

Una de las razones por las que la visión por computadora es tan compleja es debido a que se intenta obtener información a partir de un problema a la inversa, donde se trata de obtener una solución con datos insuficientes. Por ello, se recurre a modelos probabilísticos para distinguir entre posibles soluciones. Estos modelos suelen basarse en física tal como la óptica, radiometría y diseño de sensores; así como en gráficos de computadora.

Hoy en día, la visión por computadora es utilizada en diversos campos como la detección de caracteres, inspección de máquinas, aplicaciones médicas, vigilancia, entre muchos otros [39].

3.9. Detección de movimiento

Existen infinidad de aplicaciones en las cuales es necesario detectar movimiento, por ejemplo, aquellas en las cuales se requiere contar a las personas que pasan por un determinado lugar, carros, o algún objeto, entre otros. Para cualquier caso, se debe extraer al objetivo de la escena. Algunos métodos comúnmente utilizados están basados en extracción o substracción de fondo siendo lo que precisamente se utilizó en este prototipo [40].

3.10. Métodos de substracción de fondo

3.10.1. Substracción con imagen de referencia

Si se tiene una imagen con el fondo estático o solo, por ejemplo, una imagen del salón sin sujetos presentes en él, o una imagen de la avenida sin autos, es sencillo llegar a una solución, consistiendo únicamente en extraer o

substraer la imagen nueva (con sujetos u objetos) de la imagen sin alteraciones como se muestra en la figura 9 [41].

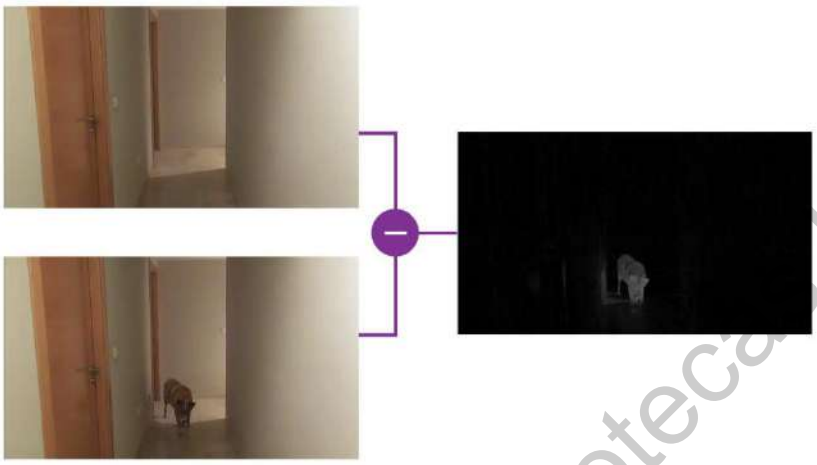


Figura 9. Ejemplo de substracción de fondo mediante una imagen de referencia [40].

Este método es muy sensible a los cambios de iluminación como se puede observar en la imagen a la derecha dentro de la figura 9 en la zona a la izquierda del canino. Otra desventaja es si el objeto o sujeto tiene un color parecido al fondo.

Comúnmente se utilizan los siguientes pasos para la detección de movimiento mediante este tipo de metodología [40]:

- Conversión a escala de grises y eliminación de ruido.
- Substracción entre ambas imágenes.
- Umbralización (los pixeles a partir de cierto valor se cambian por blancos mientras que el resto se cambian a negro).
- Detección de formas o contornos, mediante la contabilización de los pixeles como resultado de la umbralización, o en su defecto con la detección de bordes y operaciones morfológicas.

Desafortunadamente en muchas situaciones no es posible obtener una imagen como del fondo tendiéndose además situaciones con sombras las cuales son detectadas como objetos. Es por esto que algunos métodos capaces de trabajar con estas condiciones han sido creados, a continuación, se presentarán algunos [42].

3.10.2. Método “BackgroundSubtractorMOG”

Es un método gaussiano que mezcla segmentación del fondo y el primer plano. Ideado por KadewTraKuPong y R. Bowden en 2001. Cada pixel del

fondo es modelado por una mezcla de K distribuciones gaussianas (K=3 a 5). El peso de la mezcla representa las proporciones de tiempo que esos colores permanecen en la escena. Los colores probables del fondo son aquellos que permanecen más tiempo y estáticos en la imagen [40] [42].

3.10.3. Método "BackgroundSubtractorMOG2"

Similar al anterior, basado en trabajo de Z.Zivkovic; una característica de este algoritmo es que selecciona el número adecuado de distribución gaussiana para cada pixel, proveyendo así, mejor adaptabilidad a cambios de iluminación [40] [42].

3.10.4. Método "BackgroundSubtractorGMG"

Este método combina la estimación estadística de la imagen de fondo y la segmentación bayesiana por pixel. Introducido por Andrew B. Godbehere, Akihiro Matsukawa, Ken Goldberg en 2012. Utiliza por default 120 cuadros para modelar el fondo. Emplea un algoritmo probabilístico de segmentación del primer plano de la imagen que identifica posibles objetos usando inferencia bayesiana. La estimación se adapta, nuevas capturas tienen una mayor importancia que anteriores, con el objetivo de que variaciones en la iluminación afecten poco. Operaciones de apertura y cierre son aplicadas varias veces para remover ruido [42].

3.10.5. Substracción de fondo "KNN"

Se basa en el algoritmo de los k-vecinos más cercanos (k-NN). La entrada consta de los k ejemplos de entrenamiento más cercanos en el espacio de características. El resultado depende de si se utiliza k-NN para clasificación o regresión [43]:

- En la clasificación k-NN, la salida es una pertenencia a una clase. Un objeto se clasifica mediante un voto de pluralidad de sus vecinos, y el objeto se asigna a la clase más común entre sus k vecinos más cercanos (k es un número entero positivo, típicamente pequeño). Si k = 1, entonces el objeto simplemente se asigna a la clase de ese único vecino más cercano.

- En la regresión k-NN, la salida es el valor de propiedad del objeto. Este valor es el promedio de los valores de k vecinos más cercanos.

3.11. Teoría de las actividades rutinarias

Esta teoría forma parte de la criminología ambiental y fue desarrollada por los criminólogos Lawrence Cohen y Marcus Felson. Dicha teoría establece que un delito

puede ocurrir por tres factores elementales: un objetivo conveniente, ausencia de seguridad que pueda prevenir que el delito suceda y la probabilidad de que exista un delincuente [6].

En cuanto al objetivo conveniente este puede ser una persona, un lugar o un objeto; además, al tenerse la ausencia de un guardián las probabilidades sufrir un delito crecen, entre algunos ejemplos de estos guardianes se encuentran patrullas policiacas, guardias de seguridad privada, amigos, vecinos o un sistema de seguridad con cámaras [6].

4. Metodología

4.1. Introducción

El proceso para llevar a cabo este desarrollo cumpliendo con los objetivos planteados parte de un proceso de ocho etapas, cada una cumpliendo con una parte del prototipo.

Este capítulo describe explícita y detalladamente el proceso llevado a cabo en cada etapa, mostrando información de las características requeridas, materiales y software, así como algunas comparaciones con otras opciones viables para un desarrollo como este.

4.2. Esquema general del proyecto

En la figura 10 se observa un diagrama que muestra de manera general la metodología detrabajo.

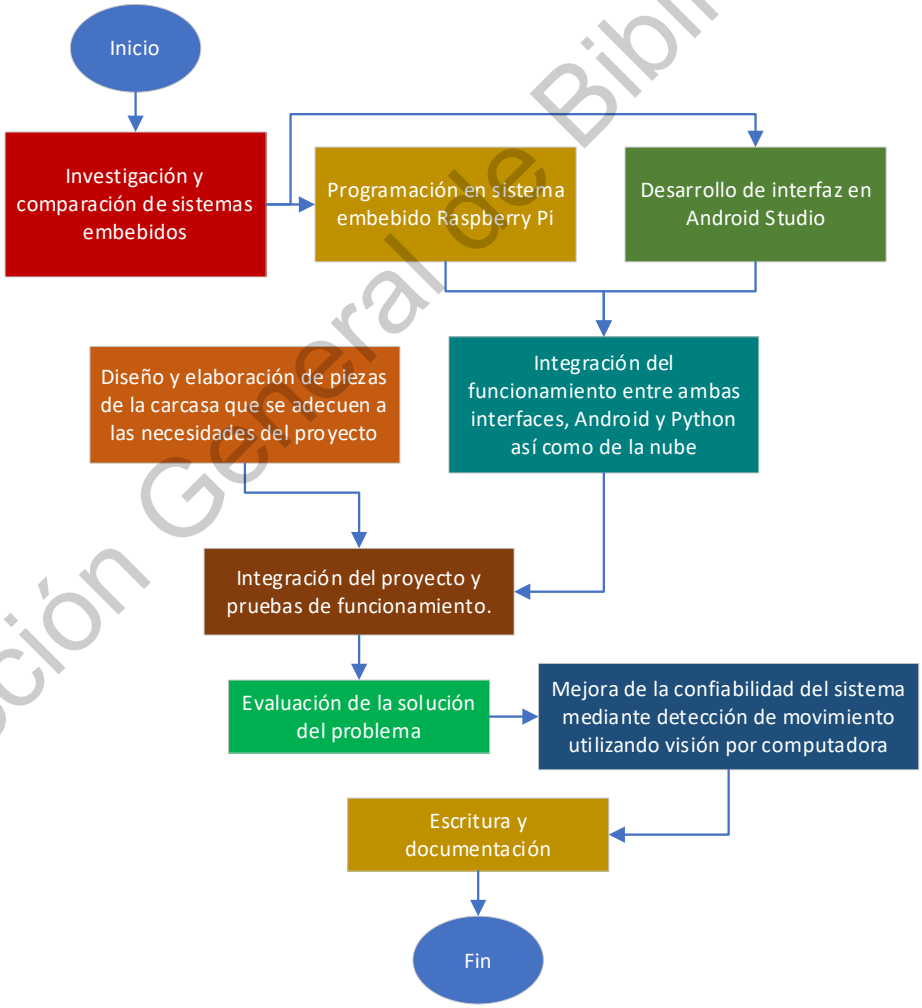


Figura 10. Diagrama general del proyecto.

4.3. Investigación y comparación sobre sistemas embebidos.

En esta etapa se comparan diferentes sistemas embebidos para justificar el uso de una Raspberry Pi 3 B+ analizando otras posibles opciones que se podrían tomar para un desarrollo como este, comparando características de acuerdo a la necesidad de vigilancia tales como, capacidad para montar un sistema operativo, capacidad de procesamiento, entradas y salidas, costo, consumo energético e información disponible sobre cómo utilizar el dispositivo.

Tabla 1. Comparativa entre opciones de sistemas embebidos.

Producto	**Sistema Operativo**	**Especificaciones de hardware**	**Entradas y salidas**	**Consumo energético**	**Costo**
Raspberry Pi 3 B+	Raspberry Pi OS	Broadcom BCM2837B0 64-bit SoC a 1.4GHz 1GB de RAM DDR2 4 USB 2.0 Puerto de cámara	40 pines 28 entradas y salidas digitales	5V hasta 2.5A	900 mxn
Raspberry Pi 4	Raspberry Pi OS	Broadcom BCM2711 64-bit SoC a 1.5GHz Hasta 8GB de RAM 2 USB 2.0 2 USB 3.0	40 pines 28 entradas y salidas digitales	5V hasta 3A	1500mxn a 2000mxn
Onion Omega 2	OpenWRT Linux	MT7688 SoC (580 MHz CPU) 128MB de RAM DDR2	32 pines 18 entradas y salidas digitales	3.3V Sin carga 200+-40mA Hasta 800mA	800 mxn
ODROID-XU4	Ubuntu 16.04 Android 4.4, 5.0 y 7.1	Samsung Exynos5422 Cortex™-A15 2Ghz 2Gbyte LPDDR3 RAM	42 pines 26 entradas y salidas digitales	5V hasta 4A	2800mxn a 3000mxn

Una gran ventaja de las tarjetas capaces de correr alguna distribución de Linux es la versatilidad que este sistema ofrece considerado aspectos como la programación en múltiples lenguajes tales como Python, C, C++, entre otros así como una posible

interacción directa a través de Bash (intérprete de órdenes predeterminado para versiones de Linux, sus siglas significan Bourne-again shell) [44].

Uno de los principales puntos por los cuales se decidió utilizar la Raspberry Pi 3 B+ es como se puede notar en la tabla 1 es su precio, lo que la hace atractiva, además es fácil de conseguir en México y no requiere de soldársele otros componentes para su funcionamiento. Por otro lado, existe una gran documentación al respecto a tratarse de una de las tarjetas más comunes de este tipo en el mercado.

4.4. Programación en sistema embebido Raspberry Pi.

Se requiere primero de establecer una conexión entre el sistema embebido utilizando Python con otros dispositivos a través de internet mediante conexiones locales, por otro lado, se requiere implementar un algoritmo en una interfaz en Linux capaz de capturar imágenes de manera automática y desde el cual se puedan realizar pruebas del hardware como el sensor PIR, con la cámara.

Para la primera parte, se realiza una investigación en relación a conexiones locales a través de sockets, estos son utilizados para enviar mensajes a través de una red, dando lugar a una forma de comunicación llamada "ICP" (inter-process communication). Esta red puede ser local conectada a la computadora o una conectada físicamente a una red externa conectada a su vez a otras redes [45].

Las aplicaciones más comunes son aquellas en las que existe un cliente y un servidor, donde un servidor espera a que uno o diferentes clientes se conecten a él solicitando o recibiendo información [46].

Una conexión a través del protocolo TCP (Transmission Control Protocol) es la que es más común y que es empleada en el desarrollo de este prototipo al ser confiable y recibir los datos en el orden en el orden en que fueron enviados [45].

Python provee de varias funciones a través de un módulo para sockets, siendo algunas de ellas mencionadas en la tabla 2.

Estas funciones son utilizadas para el funcionamiento del programa maestro o servidor que funcionaria en el sistema embebido, siendo la interfaz presente en Android un cliente que se conecta cada vez que es necesario.

Por otro lado, para realizar pruebas del hardware, siendo la cámara y el sensor, y siendo el funcionamiento deseado un algoritmo capaz de capturar imágenes cada vez que se detectase movimiento a través del sensor se utiliza Python 3.5.3 y las siguientes librerías:

- Picamera (para el uso de la cámara).

- Tkinter 8.6 (para realizar una interfaz gráfica local en Linux desde la cual interactuar el hardware).
- Datetime de Python 3.5.3 (para leer la hora del sistema).
- Gpiozero 1.6.2 (para controlar las entradas y salidas digitales y con ello el sensor PIR).
- OpenCV 3.4.0 (utilizado para guardar imágenes en esta etapa).

Tabla 2. Instrucciones comunes utilizadas en Python para la comunicación a través de sockets [47].

Instrucción	Descripción
socket()	Retorna un objeto tipo socket principal del cual se parte para la comunicación.
bind()	Enlaza el objeto tipo socket con una dirección, también se especifica el puerto a través del cual se hará la conexión.
accept()	Acepta una conexión. El objeto tipo socket debe estar previamente conectado mediante la instrucción bind().
listen()	Habilita al servidor para aceptar conexiones.
connect()	Conexión a un socket en una dirección, por ejemplo, de un cliente al servidor.
recv()	Recibe datos en forma de byte del objeto tipo socket, especificándose la longitud.
send()	Envía datos a un objeto tipo socket, debe haberse establecido previamente comunicación.
close()	Termina la comunicación, toda operación o llamada al socket fallará después de haberlo terminado.

En los resultados se muestran capturas de dicha interfaz, así como algunos fragmentos de códigos de las funciones creadas en Python.

4.5. Desarrollo de interfaz en Android Studio.

Para el desarrollo de una interfaz en Android que pueda comunicarse con la Raspberry Pi se hizo también uso de sockets, esto debido a que sin importar que dispositivo se utilice, un cliente y un servidor pueden comunicarse mediante el protocolo TCP. Un socket es un punto final de comunicación entre dos máquinas [48].

Se utiliza la clase Socket integrada dentro de Android. Las funciones utilizadas son muy parecidas a las de Python. En este caso únicamente se buscaba transmitir y recibir cadenas mediante las cuáles fuera posible intercambiar información entre sistemas.

Los principales métodos utilizados en Android de la clase Socket fueron:

- OutputStream (mediante el cual se envían bytes).
- InputStream (mediante al cual se reciben bytes).

Debe señalarse que se decidió fijar la dirección IP de la Raspberry Pi para poder acceder a ella siempre que fuera necesario y sin el riesgo de que esta cambiase.

Por otro lado, hilos de ejecución son implementados dentro de esta sección, debido a que estos permiten operar de una manera más eficiente al tener múltiples tareas que ejecutar al mismo tiempo [49]. Esto se justifica debido a que se considera recibir datos localmente cada vez que se captura una imagen. Dicho evento ocurriría de manera aleatoria y se requiere que, además, el resto de funciones de la interfaz en Android sean operativas, por lo que los hilos permiten que múltiples tareas se ejecuten al mismo tiempo, como podría ser enviar una cadena desde la aplicación móvil mientras en el mismo instante se reciben datos por parte de la aplicación en Python.

En la figura 11 se observa un diagrama general del algoritmo implementado en Android Studio.

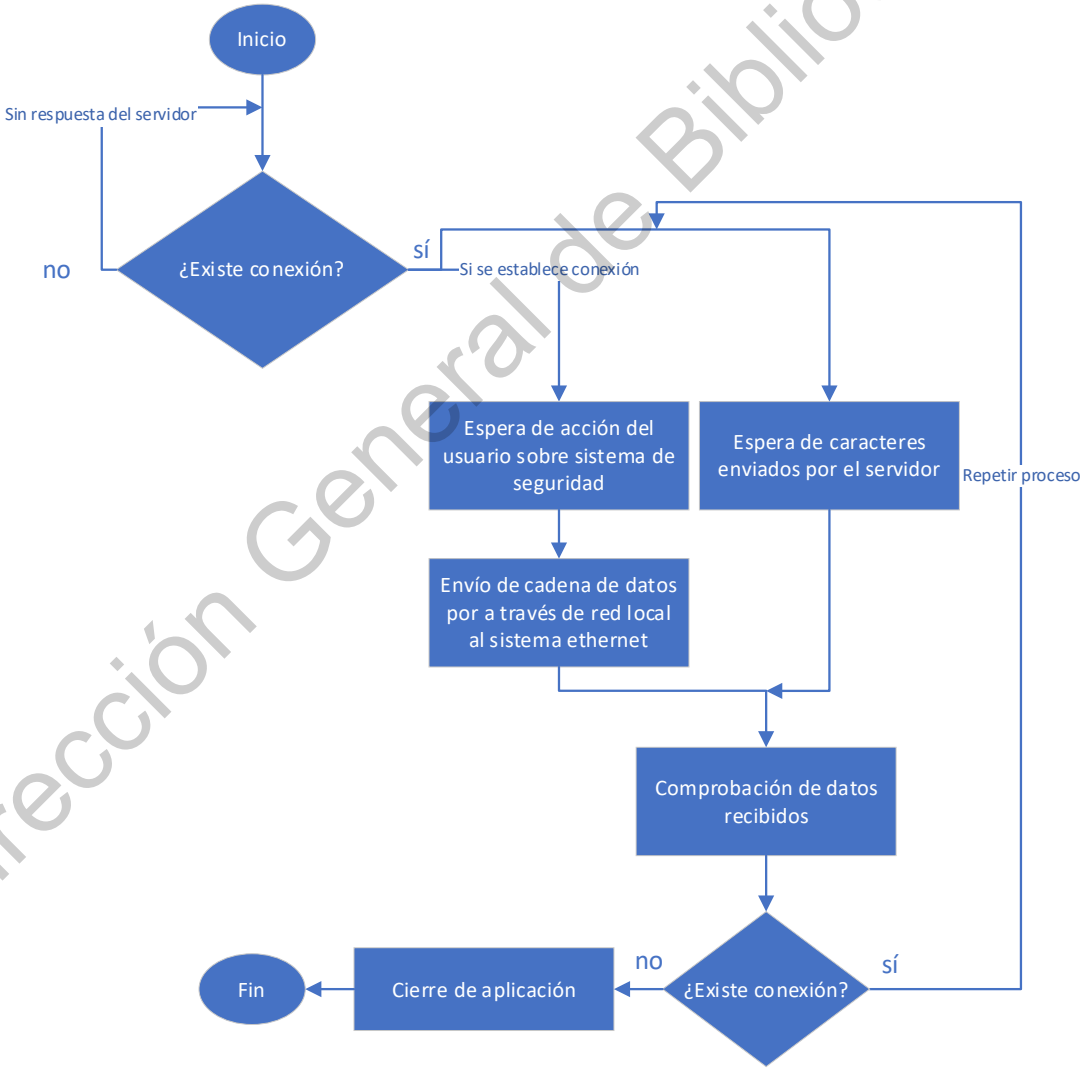


Figura 11. Diagrama del funcionamiento de la aplicación Android.

4.6. Integración del funcionamiento entre ambas interfaces.

Completadas las dos etapas anteriores, es posible realizar la interconexión entre todos los elementos de hardware (Raspberry Pi, cámara, PIR y teléfono celular), a través de una interfaz en la que se pueda controlar el algoritmo desarrollado en el sistema embebido, mediante una aplicación en Android. Por otro lado, se considera la integración de la nube.

Se opta por hacer que el sistema embebido con todos sus componentes sea el servidor, mientras el cliente una aplicación en Android Studio.

El alcance de este sistema solo permite que el usuario modifique parámetros del sistema de seguridad al estar en la misma red, por lo que para monitorear de manera remota se utiliza una carpeta en la nube, en la que todos los archivos recolectados (imágenes), al detectarse movimiento, se almacenen con su respectiva fecha y hora de captura, siendo así accesibles desde cualquier lugar con acceso a internet.

En la figura 12 se observa la interacción entre el dispositivo móvil, el servidor y el almacenamiento en la nube.

Figura 12. Interacción entre el dispositivo móvil, el servidor y el almacenamiento en la nube.

Algunas ventajas del uso de almacenamiento en la nube de un tercero son por ejemplo, que no es necesario el uso de una aplicación para consultar las imágenes del sistema de seguridad, se podría acceder desde cualquier navegador en cualquier dispositivo, además estos proveedores suelen tener una buena seguridad al

almacenar información, haciendo difícil que cualquier persona tuviera acceso a las imágenes capturadas, otra importante ventaja es que en caso de pérdida total del sistema todas las imágenes capturadas previo al incidente se encontrarían disponibles en la nube.

En este caso, una descripción tanto para el servidor como para el cliente se presenta a continuación:

- Servidor

  Este tiene como función principal siempre estar "escuchando", ya que en todo momento es capaz de recibir un comando con una instrucción desde el cliente, que para este caso existen cuatro posibles situaciones, captura de foto, activación del sistema de seguridad, desactivación del sistema de seguridad y desconexión del cliente.

  El servidor está desarrollado en Python debido a la compatibilidad que existe con el sistema embebido y Linux, así como la facilidad para el control del hardware empleado; la respuesta del algoritmo es mediante el uso de hilos, el cliente puede activar el sistema y salir de su aplicación móvil, esto dejará el sistema en operación por lo que cada vez que se detecte movimiento se hará una captura y se almacenará en la nube. En paralelo, si el cliente desea conectarse una vez más para realizar una captura o desactivar el sistema sería "escuchado".

  Por otro lado, para el control del sensor PIR y la cámara se hizo uso de librerías ya existentes descritas anteriormente, las cuales permiten un uso eficaz y simple de estos dispositivos.

- Cliente

  La tarea principal de este software consta de enviar mensajes al servidor con la indicación de qué es lo que se desea que haga, que como ya se mencionó se consta de cuatro mensajes, tres de ellos controlables por el cliente (captura de foto, activación del sistema de seguridad, desactivación del sistema de seguridad) y uno ejecutado en automático por la aplicación al salir de ella debido a su funcionalidad (desconexión del cliente).

  Mientras el servidor y el cliente se encuentren en la misma red, el servidor al capturar una imagen es capaz de enviar la información sobre la fecha y hora directamente al cliente. De cualquier modo, la imagen se almacena en la nube con estos datos y puede ser consultada respectivamente.

La visualización de las imágenes es posible realizarse desde la aplicación mediante un visualizador web, pero también desde la aplicación propia de la compañía de la nube en cualquier dispositivo.

4.7. Diseño y elaboración de piezas de la carcasa.

En la figura 13 se presenta una vista del diseño de la carcasa usando software de CAD. Las dimensiones del prototipo en largo, ancho y alto son aproximadamente de 9.5 cm, 6.5cm y 9.5 cm correspondientemente. Se diseñó de manera que la zona superior pudiera ajustarse, ya que sobre ella se montarían tanto la cámara como el sensor PIR mediante tornillería. El resto de las piezas que conforman este modelo fueron creadas pensando en que serían ensambladas entre sí mediante sus propias geometrías.

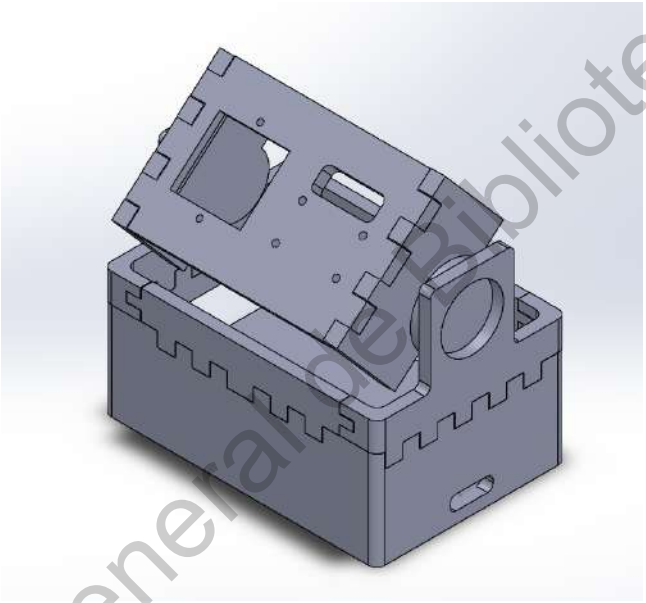


Figura 13. Modelo 3D de la carcasa

5. Resultados

5.1. Integración de los elementos y pruebas de funcionamiento.

En la figura 14 se observa el prototipo con su respectiva carcasa impresa en 3D, mediante manufactura aditiva. Además, es posible observar la cámara montada, así como el sensor PIR, y la tarjeta Raspberry Pi se encuentra dentro de la carcasa. La fuente de alimentación se encuentra fuera del montaje siendo independiente del diseño.

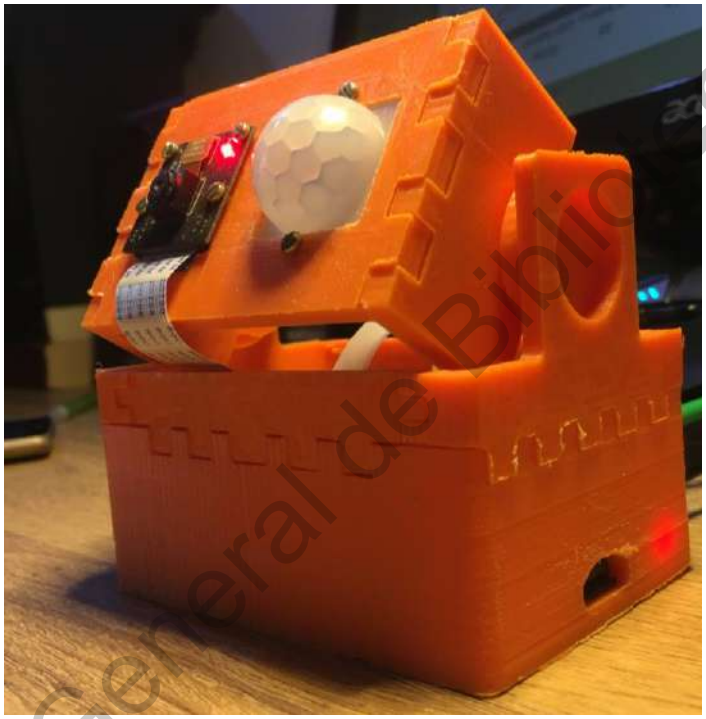


Figura 14. Carcasa impresa con los elementos de hardware.

Debe señalarse, además, que el prototipo fue diseñado de modo que pudiera ser montado en distintas posiciones para no limitar el ángulo de visión. En la figura 14 se tiene sobre una mesa o escritorio, donde se espera que los objetivos a vigilar vengan de la zona superior al dispositivo. Por otro lado, en la figura 15 se observa el sistema en una posición rotada 90° respecto a la figura 14. Esta última dando la posibilidad de vigilar desde una zona superior al área de interés y ofreciendo la posibilidad de montar el sistema en una pared o superficie vertical al poseer perforaciones donde introducir tornillería en la zona debajo de la Raspberry Pi.

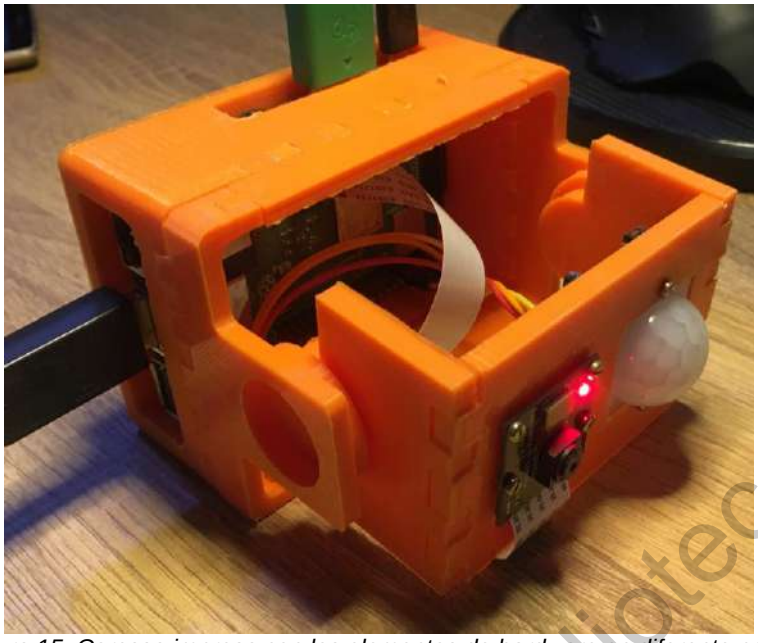


Figura 15. Carcasa impresa con los elementos de hardware en diferente pose.

Para alcanzar la integración total de los elementos del sistema, se desarrolló una aplicación en Python, versión 3.0 en el sistema embebido. Esto se hizo para probar el hardware en conjunto sin la aplicación móvil.

En la figura 16 se muestra la interfaz desarrollada a través de la cual se probaron los elementos de hardware, así como el funcionamiento deseado. Como se mencionó en la sección 4.4 referente a la programación del sistema embebido, se hizo uso de la librería "Tkinter" para el desarrollo del entorno gráfico.

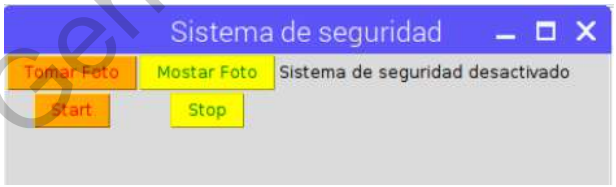


Figura 16. Interfaz gráfica para prueba de hardware.

La funcionalidad de los botones se describe en la tabla 3.

Debe señalarse que la estructura del código funciona mediante funciones, cada función asociada a un botón siendo esta llamada cada vez que es necesario llevar a cabo la acción correspondiente.

Como se describió en la sección 3.7 referente a seguridad y conexiones con la nube, Mega se eligió como la compañía para almacenamiento en la nube debido a dos principales características, su fácil integración con Linux y el almacenamiento que pueden proveer al brindar de forma gratuita 50 GB.

Tabla 3. Funcionalidad de botones de la interfaz en Raspberry Pi.

Botón	**Funcionalidad**
Tomar foto	Captura una imagen y esta es guardada en el almacenamiento.
Mostrar foto	Abre una ventana donde la foto es exhibida, la ventana se cierra al tocar una tecla.
Start	Habilita la captura de imágenes de manera autónoma al detectarse movimiento a través del sensor PIR, el cual envía una señal a través de una entrada digital en la tarjeta, esta señal es leída mediante la librería "GPIO Zero" mencionada en la sección 4.4.
Stop	Detiene la captura autónoma de imágenes en caso de detección de movimiento.

La integración con la nube consiste en crear una carpeta compartida dentro de Linux, la cual funcionaría tanto como almacenamiento interno dentro de la Raspberry Pi, así como almacenamiento en la nube, cada imagen guardada en dicho directorio es puesta en línea tan pronto como los servicios de Mega y la capacidad de la red lo permiten.

El desarrollo de la interfaz en Android consistió en el envío y recepción de caracteres tal y como se explicó en la sección 4.5. La primera versión se presenta en la figura 17 y 18, donde en la primera se muestra al cliente sin ningún tipo de conexión, y en la segunda se observa la conexión realizada con el sistema embebido.

La versión mínima de Android es Android 6 o Marshmallow, siendo esta la versión en la que se desarrolló la aplicación, por otro lado, las características del dispositivo en cuanto a rendimiento para que esta aplicación corra son mínimas, siempre y cuando se encuentre en la misma red que el sistema de seguridad.

Figura 17. Interfaz en Android sin conexión.

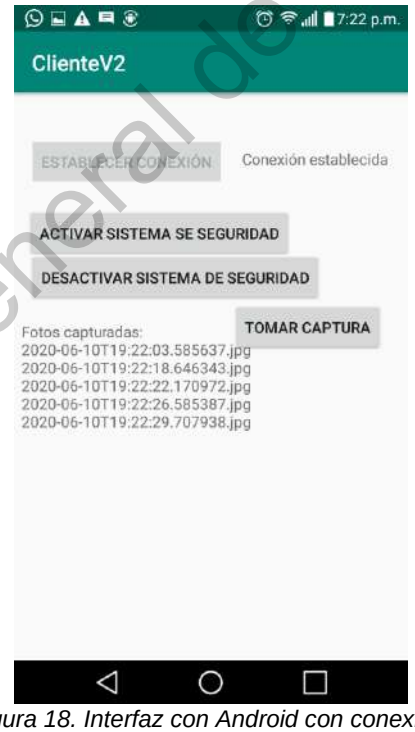


Figura 18. Interfaz con Android con conexión.

Nótese que en la figura 18, debajo de la etiqueta fotos capturadas se observan algunos números, cada renglón representa una imagen capturada, siendo el nombre de cada imagen dichos números, cada renglón contiene la fecha y hora en la que fue tomada cada imagen. Estos datos son recibidos de manera local a través de la red en la que se

encuentra el dispositivo con Android y la Raspberry Pi. También se cuenta con una etiqueta que indica el estado de la conexión en caso de que esta haya sido exitosa.

El funcionamiento de los botones de la interfaz se describe en la tabla 4.

Tabla 4. Funcionalidad de botones de la interfaz en Android.

Botón	**Funcionalidad**
Establecer conexión	Crea la conexión local mediante sockets con la Raspberry Pi a través de la dirección IP de esta.
Activar sistema de seguridad	Envía caracteres al sistema de seguridad como se describió en la sección 4.5 habilitando la captura de imágenes de manera autónoma al detectarse movimiento a través del sensor PIR.
Desactivar sistema de seguridad	Envía caracteres al sistema de seguridad como se describió en la sección 4.5 deshabilitando la captura de imágenes de manera autónoma al detectarse movimiento a través del sensor PIR.
Tomar captura	Envía caracteres los cuales indican la captura de una imagen.

Debe señalarse que en esta sección todos los eventos ocurridos desencadenan una acción contenida dentro de hilos de ejecución, asegurando así que, en caso de múltiples eventos al mismo tiempo, el sistema pueda responder adecuadamente como se describió en la sección 4.5.

Posteriormente la interfaz desarrollada se mejoró mediante tres elementos.

- Se reordenaron los elementos de una manera más amigable con el usuario.
- Se agregó la posibilidad de acceder directamente a la nube de Mega mediante un visualizador web dentro del área de la aplicación.
- Se eliminó la información de cada imagen capturada que era enviada de manera local debido a que no era de utilidad para controlar el sistema por parte del usuario al poder este consultar las imágenes directamente.

En la figura 19 se observa la aplicación móvil con dichas mejoras. De la mitad hacia arriba de la aplicación se encuentra el visualizador web donde se observan imágenes capturadas por el sistema con sus respectivas fechas.

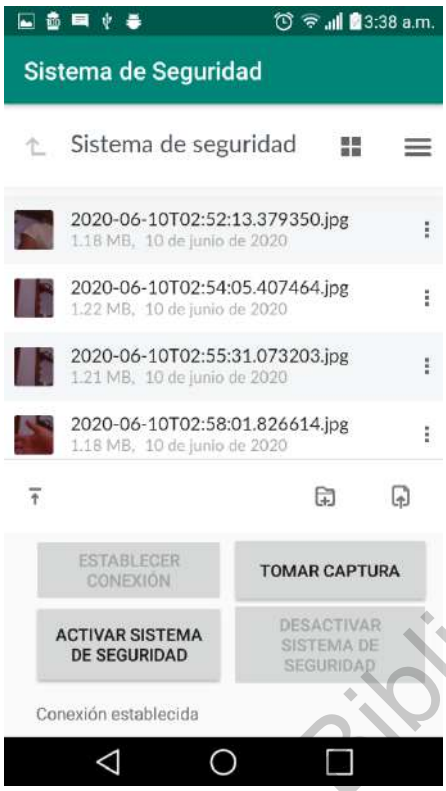


Figura 19. Aplicación móvil en funcionamiento.

En la figura 20 se observa la aplicación mostrando una imagen desde la nube en el visualizador web. El funcionamiento de la aplicación permaneció sin modificaciones a excepción de la eliminación del nombre de las fotos transmitidas de manera local como se mencionó anteriormente.

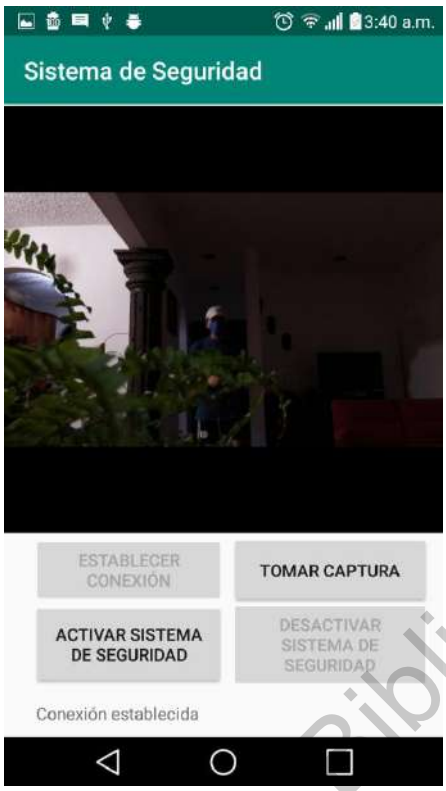


Figura 20. Aplicación móvil en funcionamiento mostrando una imagen desde la nube.

La integración de todos los elementos en un solo prototipo se muestra en la figura 21, debe señalarse que este es el resultado del prototipo sin procesamiento de imágenes, utilizando únicamente el sensor PIR como elemento de detección de movimiento.

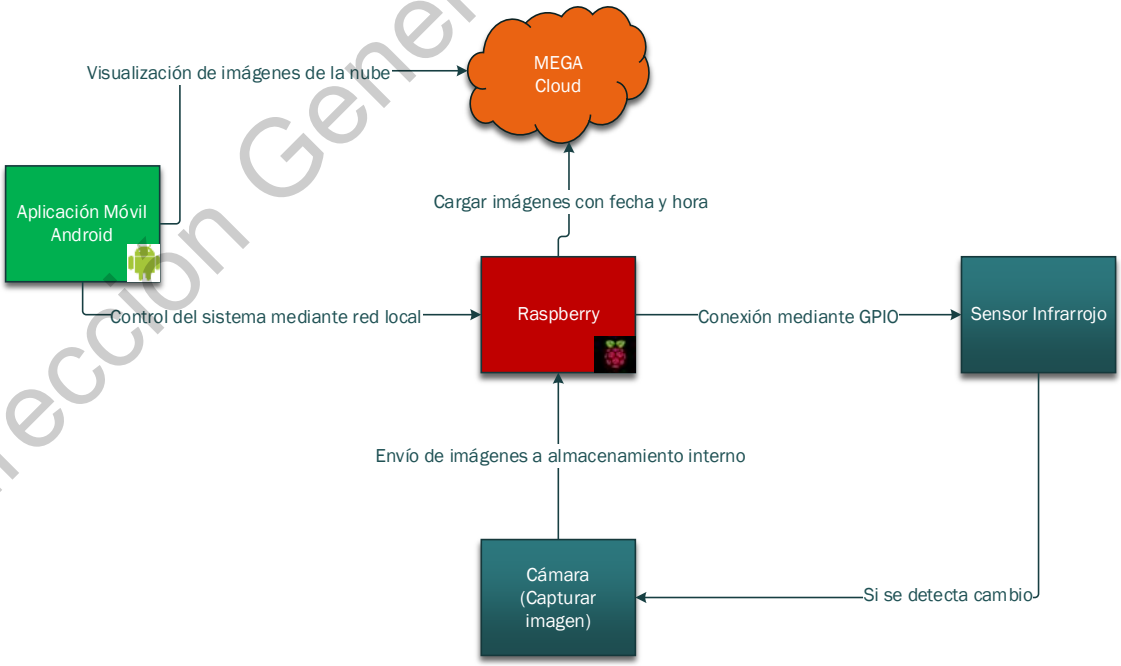


Figura 21. Funcionamiento general del sistema de seguridad (sin procesamiento de imágenes).

La pieza central es la tarjeta Raspberry Pi, que tiene una comunicación continua con el dispositivo móvil. Este dispositivo se usa para configurar la manera de funcionar del sistema de seguridad. La nube MEGA se usa para almacenar las imágenes, y se permite la consulta de las mismas desde cualquier lugar con una conexión a internet. Debe señalarse que por cuestiones de ahorro de almacenamiento tanto local como en la nube se decidió que, habiendo detectado movimiento, solo se capturaran imágenes en un intervalo que ronda los 3 segundos o menos. Como se observa en el diagrama, cada imagen registrada es nombrada con la fecha de captura para facilitar al usuario información sobre la misma.

5.2. Validación de la solución del problema.

En la figura 22, se muestra una imagen donde el sistema captura a una persona pasando dentro del alcance del sistema de vigilancia. En este caso, la imagen se toma con luz visible. La figura 23 muestra el mismo caso, pero en esta ocasión, el lugar está en obscuridad, por lo que se enciende la lámpara de infrarrojos.

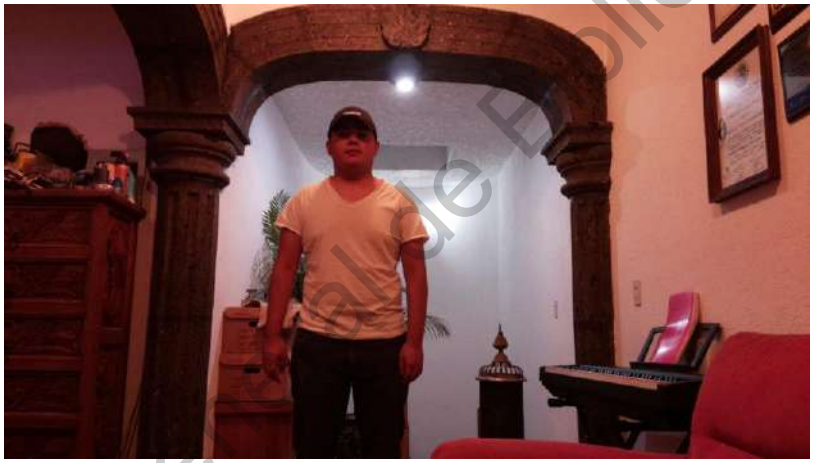


Figura 22. Imagen en condiciones de luz visible.

Figura 23. Imagen en condiciones de luz infrarroja.

Después de pruebas de funcionamiento del sistema se determinó que puede usarse como una medida de prevención al dejarse funcionando al salir del sitio donde este se encontrase, mandando este imágenes en caso de que alguien entrase en la zona donde este opera, sin embargo, también se encontraron algunos problemas relacionados principalmente con imágenes capturadas por el sistema cuando nada sucedía en la imagen, la mayoría de ellas cercanas en tiempo al desencadenamiento de la señal de salida del sensor PIR debido al movimiento de alguien, pero también algunas otras totalmente aleatorias.

Por otro lado, la frecuencia con la que se capturan imágenes una vez que detectó movimiento podría mejorarse.

5.3. Problemática del sistema de seguridad.

El sensor infrarrojo PIR, realiza su funcionamiento de acuerdo a lo esperado, enviando una señal cuando un objeto provoca un cambio en su elemento primario, sin embargo, por cuestiones que no se tienen bien delimitadas, este también enviaba la señal de detección de movimiento cuando no existía algún tipo de perturbación en el rango de operación, especialmente en momentos cercanos a una detección correcta, pero también aleatoriamente, resultando en toma de imágenes sin información útil para el usuario, utilizando espacio de memoria tanto interna de la Raspberry como de la nube y por lo tanto, desperdiciando recursos.

En la figura 24, se observa una captura de la muestra de imágenes tomadas con información que no correspondía con algún evento de detección de movimiento y algunas imágenes correctas, es posible notar algunas personas pasar, pero también imágenes sin ningún tipo de sujeto; esta captura es tomada del sistema de archivos generado por el sistema.

Figura 24. Muestra de imágenes del sistema de archivos del sistema de seguridad.

El sistema operó de una mejor forma en otras pruebas, teniendo una correspondencia clara entre las imágenes capturadas con algún objeto o sujeto en la imagen y el total de las imágenes capturadas, pero el caso mostrado en la figura 24 muestra uno de los peores escenarios que se presentaron, independientemente de ello, se puede observar y a través de más pruebas que se realizaron que era totalmente seguro que si algún objeto activaba el sensor infrarrojo, se tomaría una imagen. Esta es la base para el desarrollo utilizando detección de movimiento mejorar la confiabilidad del sistema.

5.4. Mejora del prototipo mediante algoritmo de detección de movimiento.

De acuerdo al funcionamiento explicado en la sección anterior, se partió del hecho de que era seguro que el sensor detectase si algo se movía en su rango de operación, por lo que, a partir de ello, sería posible desarrollar un algoritmo mediante el cual, se analizaran una serie de imágenes consecutivas y así determinar si efectivamente algo se movía, tomando la decisión de capturar las imágenes.

Algunos métodos de detección de movimiento que fueron considerados en la implementación de esta solución se presentan en la sección 3.10.

5.4.1. Metodología del desarrollo de la mejora del sistema empleando visión artificial.

Investigación de algoritmos de detección de movimiento: En esta etapa se investigaron varias formas de detectar movimiento en una serie de imágenes consecutivas, cuyo resultado se puede encontrar en el marco teórico.

Captura de imágenes constante: Para llevar a cabo este desarrollo, fue necesario capturar imágenes a una tasa de fotogramas constante que permitiera el análisis por visión artificial utilizando Python y OpenCV, por lo que se investigó y experimentó una forma de lograr este objetivo a través de funciones disponibles para las herramientas mencionadas.

Prueba de los algoritmos investigados: Con el fin de averiguar qué algoritmo es el más útil para esta aplicación, se probaron algunos de los métodos de substracción de fondo presentados anteriormente, posteriormente se mostrará con mayor detalle, pero la mejor opción es substracción de fondo considerando los requisitos del prototipo y su bajo costo computacional.

Implementación del algoritmo de detección de movimiento en la Raspberry Pi: Una vez conocido el funcionamiento y comportamiento del algoritmo de detección de movimiento, se modificó para que operara como una función de entrada y salida, donde se recibían dos parámetros en la entrada, el primero la imagen de fondo y el segundo la imagen contra la

cual comparar, y la salida, que devolvería una respuesta booleana en caso de detectar o no movimiento, por lo que dadas las características operativas del código (secuencial), se decidió que el imagen de fondo o sin cambios debería ser tomada en el instante en que el sensor de movimiento detectase un cambio, esto debido a que el rango de visión del sensor es mayor que el de la cámara, por lo que el encuadre obtenido como base quedaría sin ningún objeto o sujeto en el marco; cada cuadro siguiente sería analizado por el algoritmo de detección de movimiento comparándolo con la imagen de fondo.

Ajustes de la integración del algoritmo de detección de movimiento: Una de las grandes limitaciones del algoritmo utilizado es su susceptibilidad a fallas en caso de cambios de iluminación, como primera consideración para abordar este problema, se implementa lo descrito anteriormente relacionado con la imagen sin cambios tomada en el instante en que el sensor detecta movimiento, sin embargo, debido a cómo se integra el algoritmo en Python, si ocurre un cambio drástico en la iluminación, este se toma como un movimiento, capturando imágenes consecutivas sin detenerse hasta saturar la memoria, por lo que se establece un límite por un contador haciendo que este proceso se detenga luego de cierta cantidad de imágenes, siendo esto conveniente si lo que dispara el sistema, se mantiene mucho tiempo frente a él (mayor que el número de cuadros permitidos), tan pronto como este objeto o sujeto se mueve nuevamente, las imágenes son capturadas nuevamente, esto ocurre consecutivamente hasta que el fondo se mantenga constante.

5.5. Resultado de la mejora del prototipo mediante algoritmo de detección de movimiento

Tres métodos fueron probados, aunque otros dos se mencionaron anteriormente, esto debido a errores con OpenCV en la Raspberry Pi.

Todos los resultados presentados a continuación se probaron con condiciones de luz visible e infrarroja.

Método de sustracción de fondo: Para este método los pasos seguidos fueron los mencionados en la sección 3.10:

La ecuación general que representa la substracción de fondo se presenta a continuación:

$$O_{img} = A_{img} - S_{img} \quad (1)$$

Donde O_{img} representa la imagen resultada debido a la substracción, A_{img} es la imagen con alteraciones respecto a S_{img} y S_{img} es la imagen del fondo estático.

Cabe mencionar que este es el método que se decidió utilizar para la operación final, esto es debido a que la primera imagen tomada o fondo funciona excelente para la aplicación, cada vez que el sensor detectaba algún movimiento el algoritmo comenzaba a funcionar, finalmente, la decisión de si se debe tomar una imagen o no depende de la imagen umbral, además de eso, los otros métodos están destinados a casos en los que el fondo no es constante o no es posible obtener una imagen del fondo.

La serie de pasos donde esta imagen y otras del proceso se pueden ver en las figuras 26 y 27.

Una imagen captada por el prototipo en una zona sin ningún tipo de cambio es la que se observa en la figura 25, siendo esta una puerta y una pared.

Figura 25. Imagen de fondo.

Una vez que se activa el sistema de seguridad y se ha detectado un cambio, se realiza el procesamiento para determinar si hay algo en la imagen, como se verá a continuación. La figura 26 muestra una mano, que desencadena el proceso de análisis, como imagen de referencia se utiliza la imagen de la figura 25.

a b c d e

Figura 26. Proceso de detección de movimiento utilizando substracción de fondo en condiciones de luz visible.

En la figura 26 se observa el proceso que ocurre en el algoritmo, en la primera imagen de izquierda a derecha se observa la imagen RGB (a), luego se observa la imagen en escala de grises (b), el siguiente paso es aplicar un filtro gaussiano para reducir detalles innecesarios (c), la cuarta fase es la resta entre la imagen de fondo original

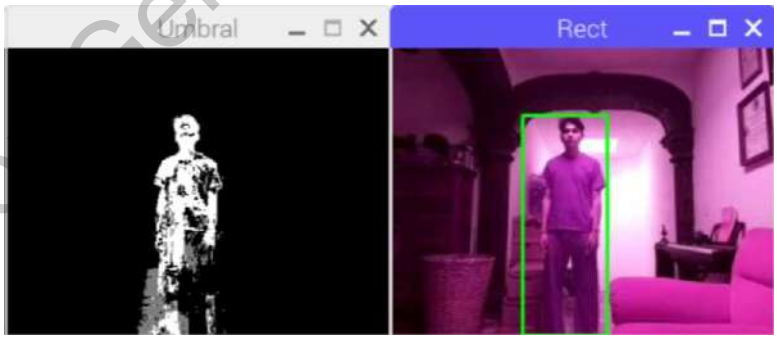
(que también pasó por los pasos antes mencionados) y la imagen con el cambio resultante en la cuarta imagen (d), finalmente, después de una operación de umbralización, una imagen con solo dos tonos, blanco o negro es obtenida (e), así al encontrar un píxel en blanco es suficiente para inferir detección de movimiento y por lo tanto guardar la imagen RGB original en la memoria. Para eso, un bucle que analiza varios píxeles en toda la imagen decide con una condición; se podría hacer analizando cada píxel, pero para hacer el proceso menos intensivo computacionalmente, se analizaron solo partes de la matriz de píxeles. El mismo proceso se observa en la figura 27, pero con luz infrarroja.

a b c d e

Figura 27. Proceso de detección de movimiento utilizando substracción de fondo en condiciones de luz infrarroja.

Método de substracción de fondo MOG2: Para este método, se utilizó la función que estaba incluida en OpenCV. La integración con el sistema de seguridad fue sencilla, aunque no hubo control de los parámetros como en el método de sustracción de fondo. También se encontraron los contornos del objeto para que se pudiera dibujar un rectángulo alrededor de la imagen para indicar dónde estaba la diferencia, este proceso también se implementó para los resultados finales.

La imagen (a) para las figuras 28 a 31 llamada "Umbral" es el resultado del método, mientras la imagen (b) con título "Rect" es la imagen capturada y almacenada.



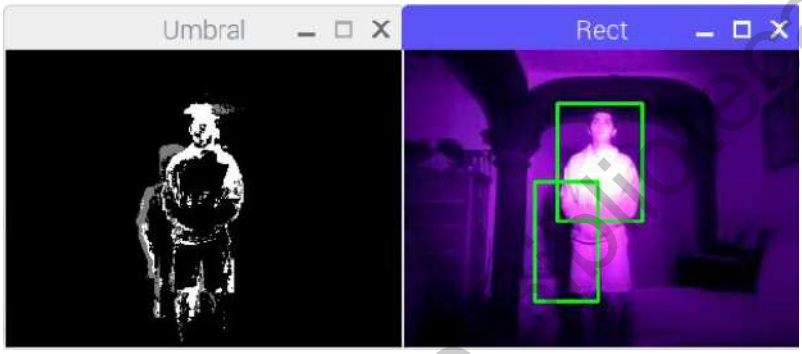
a b

Figura 28. Detección de movimiento utilizando substracción de fondo MOG2 en condiciones de luz visible.

En la figura 29, se presenta el mismo resultado con luz infrarroja, nótese que este método también es susceptible a la detección de sombras como objetos adicionales.

Método de substracción de fondo KNN: La forma en que se implementó es la misma que la anterior. En las figuras 30 y 31 se pueden ver los resultados en condiciones de luz visible e infrarroja.

Se puede notar que los dos últimos métodos son bastante similares en resultados, aunque no se puede mostrar en este texto, cuando el sujeto en las imágenes se mantuvo quieto, ambos algoritmos comenzaron a hacer como si este desapareciera, mientras que el método de substracción de fondo puro no, esa es otra razón de que sea el método a implementar en el prototipo final, sin embargo, cualquier método podría ser útil para esta aplicación ya que el seguimiento no es relevante para este proyecto.

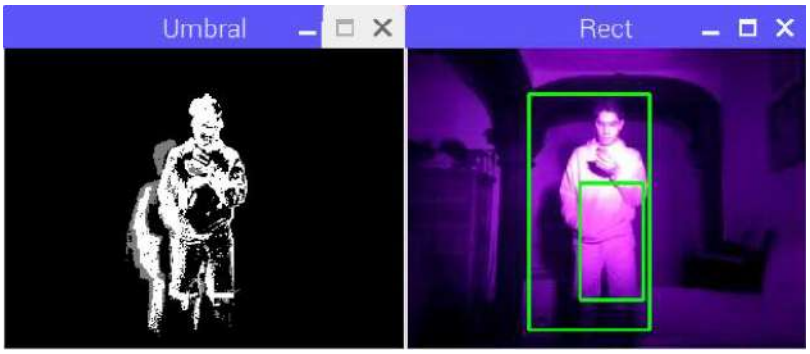


a b

Figura 29. Detección de movimiento utilizando substracción de fondo MOG2 en condiciones de luz infrarroja.

a b

Figura 30. Detección de movimiento utilizando substracción de fondo KNN en condiciones de luz visible.



a b

Figura 31. Detección de movimiento utilizando substracción de fondo KNN en condiciones de luz infrarroja.

5.6. Resultados finales

En las siguientes figuras, el algoritmo de substracción de fondo se presenta nuevamente, en la ubicación donde se probaron los otros dos métodos MOG2 y KNN, también, se agregó el rectángulo de seguimiento.

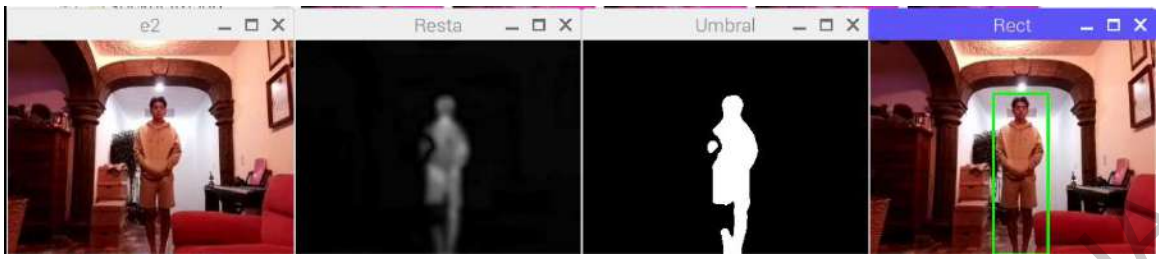
En la figura 32 se muestra la implementación mediante Python de tres barras utilizadas para determinar los parámetros que funcionan bien para este proceso. La primera cambia el primer parámetro del filtro gaussiano, el parámetro "ksise", la segunda el parámetro “sigma” que es parte del filtro de desenfoque gaussiano y la tercera barra “Thresh” cambia el umbral de lo que se considera parte del objeto en movimiento y lo que no.

Figura 32. Barras para configurar parámetros del algoritmo de detección de movimiento.

La conclusión de la calibración de estos parámetros fue que "ksize" debe ser alto como se presenta en la última imagen, en realidad tiene un valor de 21 mientras sigma es cero, esto hace que la imagen con filtrado gaussiano sea extremadamente borrosa, que es lo mejor para este método, de modo que cuando alguien entra en el rango de visión, puede ser tratado como un solo objeto.

Para el valor de umbral, se determinó que alrededor de 30 de 255 era lo suficientemente bajo para esta aplicación.

Los resultados con condiciones de luz visible se presentan en la figura 33.



a b c d

Figura 33. Resultados en condiciones de luz visible.

Por otro lado, con condiciones de luz infrarroja, el resultado es mostrado en la figura 34.

a b c d

Figura 34. Resultados en condiciones de luz infrarroja.

El proceso llevado a cabo en las figuras 33 y 34 es el mismo de izquierda a derecha y corresponde con lo descrito en la figura 26, en las imágenes (a) en este caso se muestra la imagen a analizar, en las imágenes (b) la resta descrita en la sección de sustracción de fondo.

Cada vez que se detecta un píxel blanco en la imagen (c) (figura 33 y 34), en el código se almacena el último cuadro correspondiente a movimiento, también debe tenerse en cuenta que ninguna de las imágenes anteriores está planeada para ser observada por el usuario, estas se muestran solo con fines ilustrativos, el usuario puede ver la imagen contenida en la ventana “rect” únicamente al ser cargada en la nube o en el almacenamiento local.

Cada imagen se almacena en una carpeta con la fecha, y cada imagen también tiene la fecha en que se tomó como nombre, como se observa en la figura 24.

5.7. Análisis de resultados

Comparación con el prototipo sin detección de movimiento: En la figura 34, se tiene una gráfica del funcionamiento del prototipo sin detección de movimiento por visión por computadora, durante 24 horas.

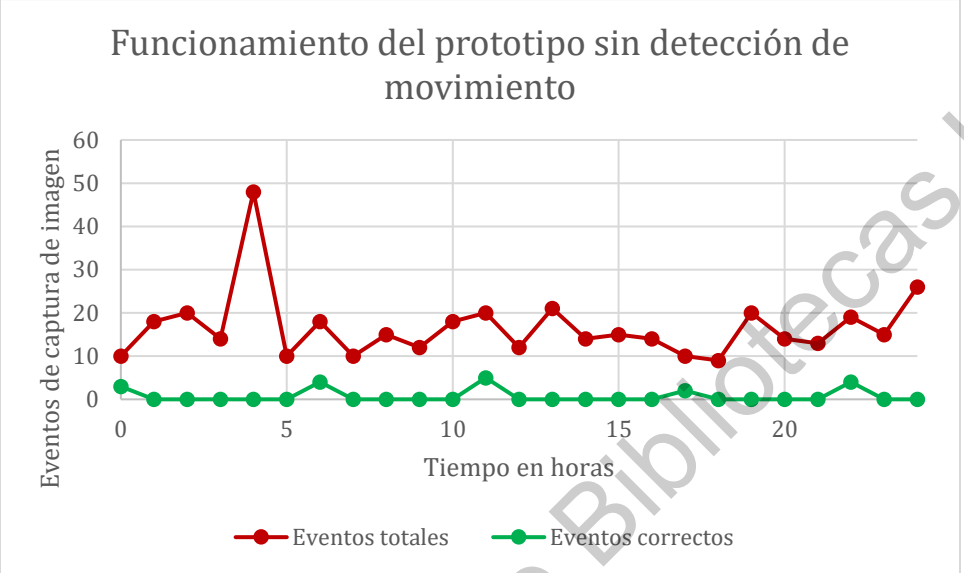


Figura 35. Funcionamiento del prototipo.

En la figura 35, en el eje horizontal se tiene el tiempo de funcionamiento, mientras en el vertical, eventos de captura de imágenes, cada punto verde representa un dato donde la imagen muestra información que corresponde con detección de movimiento por parte del sensor, por otro lado, cada punto rojo corresponde con imágenes capturadas en total; en general cada punto representa el número de imágenes capturadas en un intervalo de una hora, por lo tanto, el número de imágenes erróneas es la diferencia entre el número total de imágenes capturadas menos el número de imágenes correctas.

Por lo tanto, se puede observar en la figura 35, que el sistema, aunque cumple con su objetivo de registrar imágenes de algún objeto o sujeto que cruzó en su campo de operación, también captura imágenes en momentos donde no existió ningún tipo de evento. Debe señalarse que esta gráfica corresponde con el funcionamiento en una de las peores condiciones observadas. Se sospecha que este hecho podría estar relacionado con algún tipo de interferencia con señales electromagnéticas debido al tipo de cableado utilizado en el prototipo al no haber utilizado circuitos impresos como medio de conexión entre el sensor y la tarjeta embebida.

En contraste, en la figura 36 se observan los resultados de una prueba de funcionamiento de alrededor de 24 horas que fue realizada empleando la substracción de fondo para la detección de movimiento, esta vez con resultados satisfactorios al capturar las imágenes debidas a movimiento en el rango de

operación, donde se aprecia que los eventos totales y correctos son casi iguales. En el evento cercano a 40 capturas de imágenes se puede observar que existen algunas imágenes tomadas más con respecto al número de eventos correctos, esto fue debido al cambio de posición de un objeto del fondo, por lo que al ser comparada la imagen con el objeto en una posición diferente contra la imagen con el fondo estático se obtuvo una diferencia.

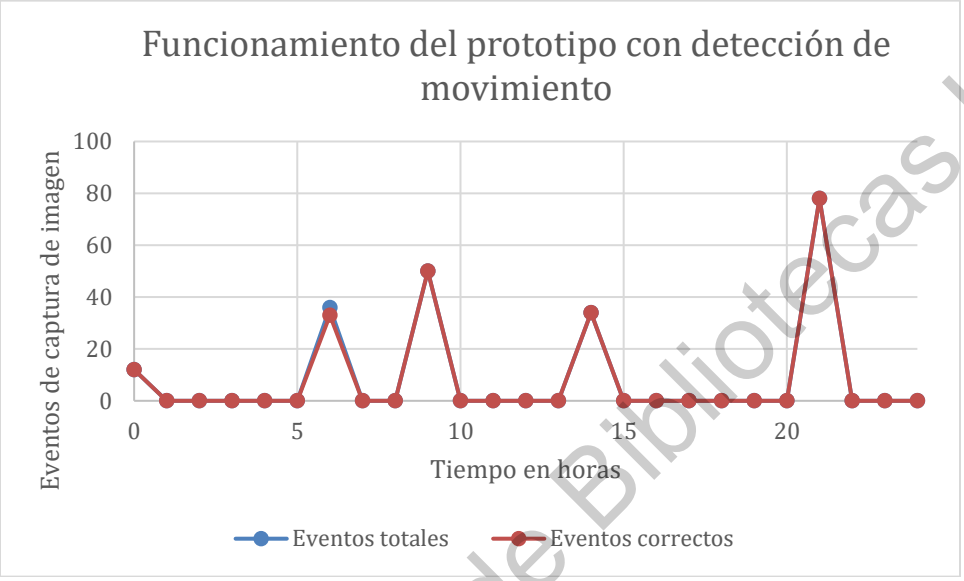


Figura 36. Funcionamiento del prototipo después de la implementación del algoritmo de detección de movimiento.

6. Conclusiones y trabajo futuro

El uso de un sistema embebido Raspberry Pi, combinado con internet de las cosas y Android permitió el desarrollo de un sistema de seguridad capaz de enviar información a la nube, así como la implementación de una interfaz móvil desde la cual controlar y consultar el prototipo. Además, la tecnología integrada y el software desarrollado permiten la operación del sistema en condiciones de baja o nula iluminación, así como condiciones de luz de día.

La sección de detección de movimiento basada en visión por computadora funciona como un método para asegurar que las imágenes capturadas contienen información relevante, elevando la efectividad del sistema respecto al mismo sin procesamiento de imágenes.

Por otro lado, si bien el método elegido de visión por computadora no es perfecto por las condiciones de luz, al tener problemas con las sombras, la forma no es tan importante para el sistema de vigilancia, con solo poder detectar un píxel blanco es suficiente para tomar imágenes, esto hace el prototipo más confiable que la versión anterior. Además, al ser un dispositivo desarrollado en software libre (OpenCV y Python) y una plataforma como es Linux, es totalmente viable, realizar mejoras o emplear el prototipo en alguna otra aplicación donde se requiera de detección de movimiento.

Como aplicación, este dispositivo funcionará de forma activa vigilando un área del edificio de mecatrónica de la facultad de ingeniería, aun se determinará el lugar óptimo para colocarlo.

Disponiendo aun de entradas y salidas digitales, otro tipo de sensores podrían instalarse al dispositivo ofreciendo mayor información acerca de un área. Por otro lado, desde el punto de vista de un producto al público es viable trabajar en volverlo comercializable si se aplican algunas mejoras en cuanto a la integración de los elementos como es el caso de la lámpara y una carcasa más pequeña que sea fabricable mediante un método diferente (más barato) a la impresión 3D. Además, se requerirían modificaciones en el software tanto en Android y Python para hacerlo más fácil para el público en general.

Finalmente, este trabajo puede sembrar el precedente para el desarrollo de sistemas basados en procesamiento de imágenes más complejos, como es el caso de aplicaciones que incluyan inteligencia artificial o redes neuronales sumamente útiles en dispositivos de este tipo.

7. Referencias

[1] I. N. de E. y Geografía (INEGI), "Incidencia delictiva", *Encuestas en establecimientos. Especiales. Encuesta Nacional de Victimización de Empresas. ENVE, Encuestas en hogares. Encuesta Nacional de Victimización y Percepción sobre Seguridad Pública. ENVIPE*, ene. 01, 2010. https://www.inegi.org.mx/temas/incidencia/ (consultado sep. 28, 2020).

[2] R. Macías Acosta, J. C. Macías Ponce, y M. Díaz Flores, "Programa para la Seguridad Nacional en México y su gasto aplicando sistemas de preferencias", *rgv*, 2019, doi: 10.37960/revista.v24i2.31496.

[3] Lu Tan y Neng Wang, "Future internet: The Internet of Things", en *2010 3rd International Conference on Advanced Computer Theory and Engineering(ICACTE)*, ago. 2010, vol. 5, pp. V5-376-V5-380. doi: 10.1109/ICACTE.2010.5579543.

[4] M. Richardson y S. Wallace, *Getting Started with Raspberry Pi*. O'Reilly Media, Inc., 2012.

[5] "Buy a Raspberry Pi 3 Model B – Raspberry Pi". https://www.raspberrypi.org/products/raspberry-pi-3-model-b/ (consultado oct. 01, 2020).

[6] "TEORÍA DE LA ACTIVIDAD RUTINARIA". https://criminalistica.mx/77-uncategorised/1628-teoria-de-la-actividad-rutinaria (consultado mar. 02, 2021).

[7] "The ACM Code of Ethics arose from the experiences, values and aspirations of computing professionals around the world, and captures the conscience of the profession. It affirms an obligation of computing professionals to use their skills for the benefit of society." https://www.acm.org/about-acm/code-of-ethics-in-spanish (consultado nov. 01, 2020).

[8] M. Moghavvemi y Lu Chin Seng, "Pyroelectric infrared sensor for intruder detection", en *2004 IEEE Region 10 Conference TENCON 2004.*, nov. 2004, vol. D, pp. 656-659 Vol. 4. doi: 10.1109/TENCON.2004.1415018.

[9] S. Prasad, P. Mahalakshmi, A. J. C. Sunder, y R. Swathi, "Smart Surveillance Monitoring System Using Raspberry PI and PIR Sensor", vol. 5, p. 4, 2014.

[10] W. F. Abaya, J. Basa, M. Sy, A. C. Abad, y E. P. Dadios, "Low cost smart security camera with night vision capability using Raspberry Pi and OpenCV", en *2014 International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM)*, nov. 2014, pp. 1–6. doi: 10.1109/HNICEM.2014.7016253.

[11] G. Cocorullo, P. Corsonello, F. Frustaci, L. Guachi, y S. Perri, "Embedded surveillance system using background subtraction and Raspberry Pi", en *2015 AEIT International Annual Conference (AEIT)*, oct. 2015, pp. 1–5. doi: 10.1109/AEIT.2015.7415219.

[12] R. R. Vaidya, "EFFICIENT EMBEDDED SURVEILLANCE SYSTEM WITH AUTO IMAGE CAPTURING AND EMAIL SENDING FACILITY", vol. 3, núm. 1, p. 4, 2015.

[13] Dr. S. Hussain, "Smart Surveillance System using Thing Speak and Raspberry Pi", vol. 4, ago. 2015, doi: 10.17148/IJARCCE.2015.4749.

[14] M. K. Hossen y S. H. Tuli, "A surveillance system based on motion detection and motion estimation using optical flow", en *2016 5th International Conference on*

Informatics, Electronics and Vision (ICIEV), may 2016, pp. 646–651. doi: 10.1109/ICIEV.2016.7760081.

[15] A. J. K. Jayakumar y S. Muthulakshmi, "Raspberry Pi-Based Surveillance System with IoT", en *Intelligent Embedded Systems*, Singapore, 2018, pp. 173–185. doi: 10.1007/978-981-10-8575-8_19.

[16] C. Yang, T. Chou, F. Chang, C. Ssu-Yuan, y J. Guo, "A smart surveillance system with multiple people detection, tracking, and behavior analysis", en *2016 International Symposium on VLSI Design, Automation and Test (VLSI-DAT)*, abr. 2016, pp. 1–4. doi: 10.1109/VLSI-DAT.2016.7482569.

[17] C. A. G. Godoy y O. J. S. Parra, "Sistema de seguridad para locales comerciales mediante Raspberry Pi, cámara y sensor PIR", *Revista Virtual Universidad Católica del Norte*, vol. 0, núm. 51, Art. núm. 51, ago. 2017.

[18] J. S. Agressoth Cardona y S. Murillo Román, "Sistema de seguridad electrónico IOT contra intrusión para inmuebles, con estructura de control, mediante aplicación en Android", nov. 2018, Consultado: oct. 14, 2020. [En línea]. Disponible en: http://repository.udistrital.edu.co/handle/11349/22444

[19] T. Pawlenka y J. Škuta, "Security system based on microcontrollers", en *2018 19th International Carpathian Control Conference (ICCC)*, may 2018, pp. 344–347. doi: 10.1109/CarpathianCC.2018.8399653.

[20] A. Cunalata y D. Alberto, "Desarrollo de un prototipo de sistema de seguridad contra intrusos utilizando protocolos de IoT sobre la plataforma Zolertia Remote", jul. 2019, Consultado: oct. 14, 2020. [En línea]. Disponible en: http://bibdigital.epn.edu.ec/handle/15000/20318

[21] J. Marín Rodríguez, "Desarrollo de un sistema de video vigilancia con servidor VPN Raspberry Pi y APP para móvil.", may 2020, Consultado: oct. 14, 2020. [En línea]. Disponible en: https://riunet.upv.es/handle/10251/143050

[22] Dr. J. I. Zong Chen, "Smart Security System for Suspicious Activity Detection in Volatile Areas", *JITDW*, vol. 02, núm. 01, pp. 64–72, mar. 2020, doi: 10.36548/jitdw.2020.1.006.

[23] "Seedary | Tienda Online". https://seedaryoficial.mercadoshops.com.mx/ (consultado abr. 29, 2021).

[24] E. A. Lee y S. A. Seshia, *Introduction to Embedded Systems: A Cyber-Physical Systems Approach*. MIT Press, 2016.

[25] R. Royo Pastor, "TERMOGRAFÍA INFRARROJA. FUNDAMENTOS, INVESTIGACIÓN Y APLICACIONES", presentado en Colección Manual de referencia, jul. 2013. Consultado: oct. 23, 2020. [En línea]. Disponible en: https://riunet.upv.es/handle/10251/66062

[26] Fonrouge Sergio, "Espectro electromagnético - Ondas electromagnéticas". https://sites.google.com/site/ondaselecmag/espectro-electromagnetico (consultado oct. 25, 2020).

[27] B. Fontal, *El espectro electromagnetico y sus aplicaciones*, vol. 1. Escuela Venezolana para la Enseñanza de la Química, 2005.

[28] "Artículo | Omniblug". http://www.omniblug.com/sensor-movimiento-pir-arduino.html (consultado oct. 23, 2020).

[29] García, Ginés, "Procesamiento Audiovisual", *ginés garcía mateos*, 2010. http://dis.um.es/profesores/ginesgm/pav.html (consultado oct. 23, 2020).

[30] "Understanding Digital Camera Sensors". https://www.cambridgeincolour.com/tutorials/camera-sensors.htm (consultado jul. 13, 2021).
[31] Cambridge in Colour, "Compact vs. Digital SLR Cameras", 2020. https://www.cambridgeincolour.com/tutorials/compact-vs-digital-slr-cameras.htm (consultado nov. 01, 2020).
[32] "Camera Module - Raspberry Pi Documentation". https://www.raspberrypi.org/documentation/hardware/camera/ (consultado oct. 23, 2020).
[33] "¿Qué es Internet of Things (IoT)?" https://www.oracle.com/mx/internet-of-things/what-is-iot/ (consultado ago. 15, 2021).
[34] "¿Qué es IoT (Internet Of Things)?", *Deloitte Spain*. https://www2.deloitte.com/es/es/pages/technology/articles/IoT-internet-of-things.html (consultado ago. 15, 2021).
[35] F. Xia, L. T. Yang, L. Wang, y A. Vinel, "Internet of Things", *Int. J. Commun. Syst.*, vol. 25, núm. 9, pp. 1101–1102, sep. 2012, doi: 10.1002/dac.2417.
[36] A. Ukil, J. Sen, y S. Koilakonda, "Embedded security for Internet of Things", en *2011 2nd National Conference on Emerging Trends and Applications in Computer Science*, mar. 2011, pp. 1–6. doi: 10.1109/NCETACS.2011.5751382.
[37] S. Thamburasa, S. Easwaramoorthy, K. Aravind, S. B. Bhushan, y U. Moorthy, "Digital forensic analysis of cloud storage data in IDrive and Mega cloud drive", en *2016 International Conference on Inventive Computation Technologies (ICICT)*, ago. 2016, vol. 3, pp. 1–6. doi: 10.1109/INVENTIVE.2016.7830159.
[38] F. Daryabar, A. Dehghantanha, B. Eterovic-Soric, y K.-K. R. Choo, "Forensic investigation of OneDrive, Box, GoogleDrive and Dropbox applications on Android and iOS devices", *Australian Journal of Forensic Sciences*, vol. 48, núm. 6, pp. 615–642, nov. 2016, doi: 10.1080/00450618.2015.1110620.
[39] R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer Science & Business Media, 2010.
[40] "Detección de movimiento con OpenCV y Python". https://programarfacil.com/blog/vision-artificial/deteccion-de-movimiento-con-opencv-python/ (consultado mar. 21, 2021).
[41] "OpenCV: How to Use Background Subtraction Methods". https://docs.opencv.org/3.4/d1/dc5/tutorial_background_subtraction.html (consultado mar. 22, 2021).
[42] "Background Subtraction — OpenCV 3.0.0-dev documentation". https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_video/py_bg_subtraction/py_bg_subtraction.html (consultado mar. 21, 2021).
[43] "Detección de movimiento con OpenCV y Python", *Programar fácil con Arduino*. https://programarfacil.com/blog/vision-artificial/deteccion-de-movimiento-con-opencv-python/ (consultado dic. 27, 2020).
[44] "Omega2 – Onion". https://onion.io/omega2/ (consultado abr. 12, 2021).
[45] R. Python, "Socket Programming in Python (Guide) – Real Python". https://realpython.com/python-sockets/ (consultado abr. 18, 2021).
[46] "HOW TO - Programación con sockets — documentación de Python - 3.9.4".

https://docs.python.org/es/3/howto/sockets.html (consultado abr. 18, 2021).
[47] "socket — Low-level networking interface — Python 3.9.4 documentation". https://docs.python.org/3/library/socket.html (consultado abr. 18, 2021).
[48] "Socket | Desarrolladores de Android", *Android Developers*. https://developer.android.com/reference/java/net/Socket?hl=es-419 (consultado may 17, 2021).
[49] "Java Threads". https://www.w3schools.com/java/java_threads.asp (consultado may 17, 2021).
[50] A.-F. Muñoz-Vargas, J.-M. Ramos-Arreguín, S.-T. Arriaga, M.-A. Aceves-Fernandez, y J.-C. Pedraza-Ortega, "Sistema de seguridad basado en un sistema embebido con cámara para día y noche con enlace IoT", vol. 10, núm. 1, p. 12, 2021.
[51] "2021-Libro-DisenoColaborativoenMecatronica.pdf". Consultado: oct. 03, 2021. [En línea]. Disponible en: http://www.mecamex.net/Libros/2021-Libro-DisenoColaborativoenMecatronica.pdf

8. Anexos

8.1. Artículo publicado en La Mecatrónica en México [50]

La Mecatrónica en México, Enero 2021, Vol. 10, No. 1, páginas 35 – 46
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

Sistema de seguridad basado en un sistema embebido con cámara para día y noche con enlace IoT

Aldo-Francisco Muñoz-Vargas[1], Juan-Manuel Ramos-Arreguín✉, Saúl-Tovar Arriaga, Marco-Antonio Aceves-Fernandez, Jesus-Carlos Pedraza-Ortega

Universidad Autónoma de Querétaro, Facultad de Ingeniería
✉jsistdig@yahoo.com.mx, [1]aldomuoz98@gmail.com

Resumen

El desarrollo de sistemas de seguridad es muy útil y puede incluir dispositivos como cámaras infrarrojas y sistemas embebidos basados en microprocesadores y microcontroladores. Además, con el uso de sensores infrarrojos, se aumenta el tiempo de vigilancia, ya que estos pueden trabajar aún con la ausencia de luz. Una cámara de infrarrojos permite que la cámara siga captando imágenes incluso en condiciones de oscuridad, usando una lámpara de luz infrarroja. La integración que aquí se propone, lleva todos esos conceptos combinándolos con el internet de las cosas y una aplicación basada en Android, en la que una persona puede interactuar y revisar los datos generados por el sistema en cualquier momento mientras tenga una conexión a internet. El sistema operativo de la tarjeta Raspberry Pi se llama Raspberry Pi OS, la cual es una versión ligera de Linux. Así mismo, se propone una carcasa donde es montado el sistema completo, para contar con un prototipo funcional. La carcasa es un diseño propio y fue impresa en 3D, para ser montado en interiores, haciendo de este proyecto una opción potencial para aplicaciones de vigilancia. Se desarrollaron pruebas en condiciones de ausencia de luz, y los resultados fueron satisfactorios.

Palabras clave: Cámara infrarroja, cámara Pi, Raspberry Pi, Android, sensor de infrarrojos, Internet de las cosas (IoT).

Abstract

The development of security systems is especially useful and can include devices such as infrared cameras and embedded systems based on microprocessors and microcontrollers. In addition, with the use of infrared sensors, the monitoring time is increased since they can work even in the absence of light. An infrared camera allows the camera to continue to capture images even in dark conditions, using an infrared light lamp. The integration proposed here takes all these concepts, combining them with the Internet of Things and an Android-based application, in which a person can interact and review the data generated by the system at any time while having an Internet connection. The operating system of the Raspberry Pi board is called Raspberry Pi OS, which is a lightweight version of Linux. Likewise, a housing is proposed where the complete system is assembled, to have a functional prototype, with an own design and was printed in 3D, to be mounted indoors, making this project a potential option for surveillance applications. Tests were carried out in the absence of light, and the results were satisfactory.

Keywords: Infrared Camera, Pi camera, Raspberry Pi, Android, Infrared Sensor, Internet of Things (IoT).

La Mecatrónica en México, Enero 2021, Vol. 10, No. 1, páginas 35 – 46
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

1. Introducción

Desafortunadamente, México es conocido por tener un problema con todo tipo de inseguridad, hasta 2745 casos de robos a casa habitación por cada 100,000 personas ocurrieron en 2017 [1]. Específicamente, el robo de bienes materiales puede prevenirse, evitarse o frustrarse cuando se observa al presunto ladrón [2]. Sin embargo, si el robo ocurre de todos modos, tener un dispositivo que capturó evidencias del momento puede hacer la diferencia en el proceso de justicia.

Cada vez más aplicaciones han sido desarrolladas utilizando internet de las cosas, por lo que en este proyecto también se hace uso de esta tecnología, tomando ese concepto de comunicación que ha evolucionado de humano-humano, humano-cosa y finalmente entre cosas con cosas [3].

El sistema embebido *Raspberry Pi* fue ideado para cursos de programación de estudiantes, por lo que es económico. Este es un punto realmente importante para un sistema de seguridad que está destinado a funcionar durante largas horas haciendo probable su avería. Además, la cantidad de *IO* (Entradas y Salidas) es importante y hace que esta placa sea perfecta para este tipo de proyectos [4].

Algunas aplicaciones relacionadas con este desarrollo han utilizado sistemas embebidos como *Raspberry Pi* debido al bajo costo y las opciones versátiles como la cámara, las entradas y salidas digitales, y sus capacidades informáticas con un sistema *Linux* [5].

Asimismo, en 2017, utilizando los mismos elementos (*Raspberry Pi*, sensores infrarrojos y una cámara) se desarrolló un prototipo de sistema de vigilancia para detectar movimiento en la puerta de una tienda cada 10 segundos, enviando un correo electrónico en caso de movimiento con la imagen adjunta, haciendo saber al propietario de la situación [6].

En 2018 se desarrolló un sistema de seguridad basado en un sensor de contacto, un alambrado eléctrico, una alarma sonora, una cámara y una Raspberry Pi. Este consiste en el control de los elementos anteriores desde una aplicación en Android y en la cual es posible la visualización de forma remota del inmueble en vigilancia, dando así al usuario la oportunidad de tomar acciones y activar los elementos previamente mencionados si así lo considera, evitando o al menos retrasando la entrada de un posible delincuente [7].

En 2018, se continúa desarrollando sistemas de vigilancia basados en sistemas embebidos, como el trabajo de Jayakumar [8], donde se propone el desarrollo de un sistema de vigilancia usando una tarjeta Raspberry Pi y una cámara, que se basa en detección de movimiento de personas y detección de rostros, agregando seguimiento de la persona.

En 2019 se presentó un sistema de seguridad físico que utiliza internet de las cosas mediante una plataforma llamada *Zolertia Remote*, tratando de dar una alternativa al de vigilancia. Consta de tres secciones, una red de sensores inalámbrica, un servidor privado *MQTT* (*Message Queue Telemetry Transport*) y una aplicación *Android* [9].

En 2020 se llevó a cabo una aplicación similar a las anteriores en la que mediante una *Raspberry Pi* como servidor se controló un conjunto de cámaras a través de una aplicación en *Android*, este sistema fue desarrollado de modo que en caso de detección de cambio del entorno se tomarían imágenes o videos guardándose en la memoria de la *Raspberry* y enviando un correo de alerta o notificaciones en la aplicación móvil, siendo estos datos accesibles a través de la aplicación que consultaría los datos del servidor [10].

En el presente trabajo se propone un sistema de seguridad capaz de operar en condiciones de poca o nula iluminación mediante el uso de una cámara con capacidad para observar luz infrarroja y una lámpara que emita esta luz, este tipo de situación relacionado en variaciones de la iluminación no es muy tomado en cuenta en antecedentes consultados relacionados con sistemas basados en

La Mecatrónica en México, Enero 2021, Vol. 10, No. 1, páginas 35 – 46
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

Raspberry Pi, así como la accesibilidad de los datos desde cualquier punto. Por lo que resulta en un área de oportunidad en el desarrollo de sistemas de seguridad basados en Raspberry Pi y mediante el uso del concepto de IoT.

2. Materiales y metodología

En esta sección se describen algunos de los elementos de hardware y software utilizados para el funcionamiento de este prototipo, así como la metodología utilizada.

2.1 Materiales

Detectores de radiación infrarroja: Existen diversos tipos de detectores de esta radiación, uno de ellos son los detectores térmicos, estos absorben la radiación incidente, cambiando la temperatura del detector y permaneciendo en un estado de que tiende al equilibrio en función de los cambios de la radiación, tres de los más utilizados son: los bolómetros termopilas y detectores piroeléctricos. Para este caso se explorarán los detectores piroeléctricos. La figura 1 muestra una configuración de un detector piroeléctrico.

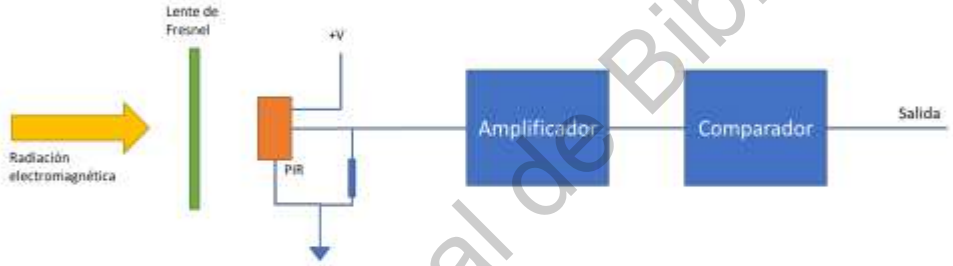


Figura 1. Configuración típica de un detector piroeléctrico [11].

Detectores piroeléctricos: Están hechos de un material que genera una carga eléctrica cuando se expone a la radiación infrarroja. Cuando la cantidad de radiación infrarroja cambia, también lo hace la carga, esto genera un voltaje que al ser amplificado puede tener diversas aplicaciones. En la figura 1 se observa un diagrama del funcionamiento de este detector. Suele utilizarse una lente de Fresnel la cual focaliza la radiación sobre el elemento primario, mientras el amplificador y comparador tratan la señal de modo que esta esta pueda a ser leída por un microcontrolador o sistema embebido [11]. A continuación, en la figura 2 se observa un Sensor Piroeléctrico Infrarrojo (*PIR*) el cual consta de lo descrito anteriormente.

Figura 2. Sensor Piroeléctrico Infrarrojo PIR [12] [13].

La Mecatrónica en México, Enero 2021, Vol. 10, No. 1, páginas 35 – 46
Disponible en linea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

Cámaras y formación de imágenes: Una imagen se forma cuando una escena luminosa en 3D es proyectada sobre un plano 2D, las cámaras realizan este proceso; en una cámara cada punto de la escena se debe proyectar a un punto de la imagen dando así lugar a una imagen enfocada, lo anterior se muestra en la figura 3 [14].

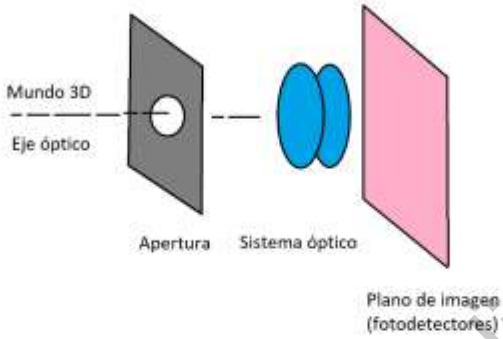


Figura 3. Modelo de cámara simplificado [14].

En el proceso de toma de imágenes, se hace uso de un proceso compuesto por captura y digitalización, donde una señal analógica es discretizada y enviada como señal digital a un ordenador. Este proceso se muestra en la figura 4 [14].

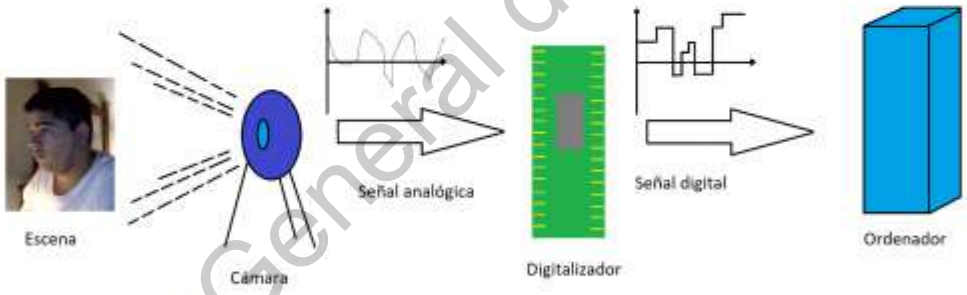


Figura 4. Proceso de captura y digitalización de imágenes [14].

Un dispositivo de captura determina los factores que tendrá la imagen final, tales como el tamaño de imagen, tamaño de pixel, el tipo de radiación que la cámara es capaz de captar, entre otras [14].

Dentro de un dispositivo de captura en el filtro de color existe un arreglo de sensores, el cual se denomina patrón de Bayer, donde el 50% de los sensores son verdes, el 25% son rojos y el otro 25% son azules. Esto se debe a que el ojo humano es más sensible al color verde que a los demás colores primarios. Esto se puede observar en la figura 5.

La Mecatrónica en México, Enero 2021, Vol. 10, No. 1, páginas 35 – 46
Disponible en linea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

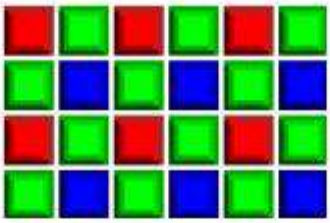


Figura 5. Distribución típica de un filtro de color [15].

La cámara NoIR utilizada en este prototipo, no posee un filtro infrarrojo haciendo posible que se detecte este tipo de luz [16]. En la figura 6 se observa una imagen tomada por la noche, usando solamente una lámpara infrarroja, y en la figura 7 se muestra la imagen tomada con iluminación artificial.

Figura 6. Imagen capturada por la cámara NoIR con iluminación infrarroja.

Figura 7. Imagen capturada por la cámara NoIR con luz visible.

2.2 Seguridad y conexiones con la nube.

Además de las enormes oportunidades que brinda *IoT*, existen varios problemas y preocupaciones relacionados con los sistemas integrados en términos de sus vulnerabilidades de red, estos problemas se pueden prevenir mediante técnicas de cifrado en las que un mensaje puede leerse con una firma especial. Estas técnicas son computacionalmente intensivas [17], de manera que, para evitar estos problemas de seguridad, se propone utilizar una nube de terceros para asignar todas esas cargas computacionales a otros, manteniendo al mismo tiempo almacenada y accesible la información recopilada por el sistema en cualquier momento y lugar.

La Mecatrónica en México, Enero 2021, Vol. 10, No. 1, páginas 35 – 46
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

Mega es un servicio en la nube que se propone para esta aplicación debido a la compatibilidad con Linux, en este servicio, los archivos se cifran antes de cargarlos y se descifran después de descargarlos utilizando una clave creada por el cliente, que se deriva de la contraseña. Utiliza un estándar de cifrado avanzado (*AES*), que es un algoritmo común para el cifrado de datos. Esto ofrece un buen nivel de seguridad con respecto a la información capturada por el sistema en caso de que alguien quiera robar o borrar el contenido. Es importante notar que existen varias vulnerabilidades sobre los datos que deja *Mega* mientras se usa en un navegador, afortunadamente, la aplicación para *Linux* no tiene ese problema [18] [19].

A continuación, en la tabla 1 se muestran los materiales que componen al sistema completo.

Tabla 1. Tabla de materiales.

Descripción	Cantidad
Raspberry Pi 3B+	1
NoIR Camera	1
PI Camera	1
5 V DC Power Supply	1
32 GB SD card	1
PIR sensor	1
Case	1

2.3 Metodología

La metodología seguida en el desarrollo e integración de este prototipo consta de las etapas mostradas en la figura 8 y descritas posteriormente.

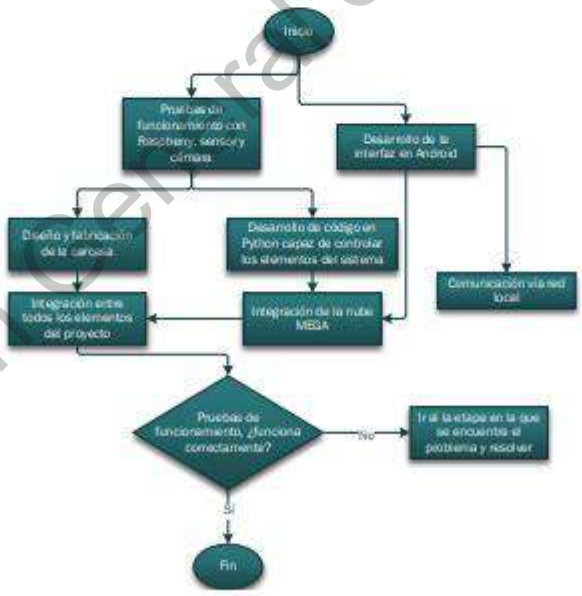


Figura 8. Diagrama de la metodología.

La Mecatrónica en México, Enero 2021, Vol. 10, No. 1, páginas 35 – 46
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

En la sección de pruebas de funcionamiento con Raspberry, sensor y cámara, se requiere implementar un algoritmo en una interfaz en Linux, desde la cual se pueden realizar pruebas del hardware como el sensor PIR, con la cámara.

Del mismo modo se requiere de una investigación sobre cómo conectar el sistema embebido, utilizando Python, con otros dispositivos a través de internet mediante conexiones locales, así como el desarrollo del código en Python que controlara todos los elementos. Por lo que esta tarea se observa en el recuadro de desarrollo de código en Python capaz de controlar los elementos del sistema.

En paralelo, se comenzó con el desarrollo de la interfaz en Android Studio. Esta interfaz móvil debe controlar a la Raspberry Pi, por lo que se requirió primero de una investigación de conexiones de red local en Android entre un teléfono y un sistema embebido como la Raspberry Pi. Una vez logrado esto se plantea el desarrollo de la interfaz en Android capaz de controlar el sistema de seguridad.

De igual forma, en paralelo, se trabaja en el diseño de una carcasa que se adapte a las necesidades del proyecto, para contener a los elementos del sistema y poder ajustar el ángulo de visión de la cámara.

En el trabajo con Internet de las Cosas, se usa la plataforma MEGA, para el almacenamiento de las fotos, debido a sus capacidades de integración con Linux y Android.

Finalmente, se elaboró la interconexión entre todos los elementos de hardware (Raspberry Pi, cámara, PIR y teléfono celular) a través de una interfaz en la que se puede controlar el algoritmo desarrollado en el sistema embebido mediante una aplicación en Android.

3. Resultados

En la figura 9, se puede observar el funcionamiento general del sistema de seguridad. La pieza central es la tarjeta Raspberry Pi, que tiene una comunicación continua con el dispositivo móvil. Este dispositivo se usa para configurar la manera de funcionar del sistema de seguridad. La nube MEGA se usa para almacenar las imágenes, y se permite la consulta de las imágenes capturadas desde cualquier lugar con una conexión a internet. Debe señalarse que por cuestiones de ahorro de almacenamiento tanto local como en la nube se decidió que, habiendo detectado movimiento, solo se capturaran imágenes en un intervalo que ronda los 3 segundos o menos. Como se observa en el diagrama, cada imagen registrada es nombrada con la fecha de captura para facilitar al usuario información sobre la misma.

3.1 Software.

Para este caso, existen varias formas de interactuar con el sistema, una de ellas consiste en el uso de sockets a través de un servidor y un cliente, donde el servidor puede recibir y responder a los mensajes del cliente en todo momento mientras están en la misma red. Se decidió usar el sistema embebido de la Raspberry Pi como servidor, mientras que el cliente, al ser cualquier tipo de dispositivo compatible con el uso de sockets, es la aplicación en Android Studio.

El uso de sockets permite ofrecer compatibilidad entre dispositivos, independientemente de su arquitectura. Por otro lado, el alcance de este software solo permite al usuario modificar parámetros del sistema de seguridad cuando se encuentra en la misma red, por lo que para monitorear remotamente se usa una carpeta en la nube, en la cual se recopilan todos los archivos (imágenes). Cuando se detecte movimiento, se almacenan las imágenes con su respectiva fecha y hora de captura.

La Mecatrónica en México, Enero 2021, Vol. 10, No. 1, páginas 35 – 46
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

En la figura 10 se muestra el menú de las imágenes capturadas. En el título de cada imagen se muestra la fecha y hora. En la figura 11 se muestran dos capturas de la aplicación en Android, donde una persona fue captada caminando frente al sistema mientras estaba funcionando.

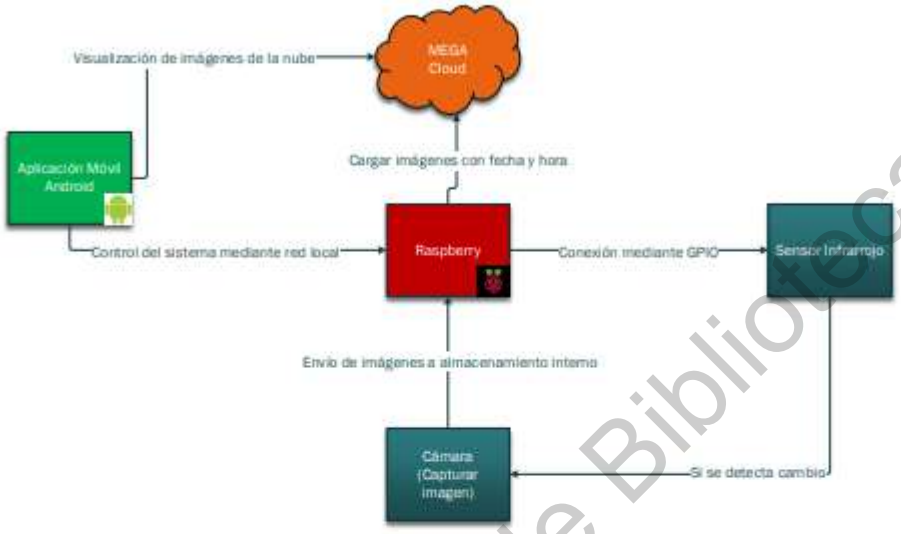


Figura 9. Funcionamiento general del sistema de seguridad.

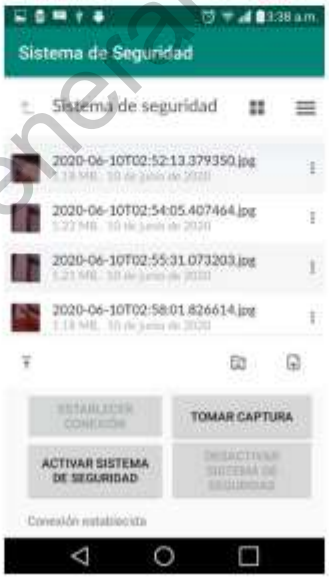
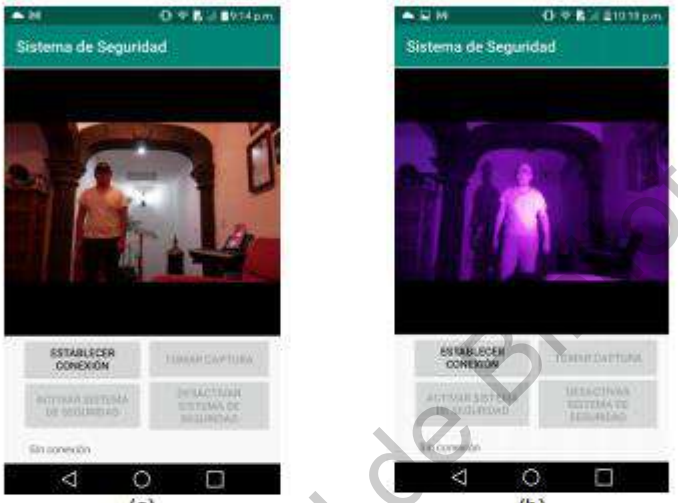


Figura 10. Aplicación móvil en funcionamiento.

La Mecatrónica en México, Enero 2021, Vol. 10, No. 1, páginas 35 – 46
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

1.1 Diseño de la carcasa.

En la figura 12 se presenta una vista del diseño de la carcasa usando software de CAD/CAM. Para el diseño se toma en cuenta los componentes que el sistema va a requerir, como son: el sistema embebido Raspberry Pi, cámara, sensor de presencia PIR, alimentación, principalmente. La figura 13 es el resultado de la impresión en 3D de la carcasa, con todos los componentes electrónicos instalados. Las dimensiones de la carcasa son 9.5 x 6.5 x 9.5 cm.



(a) (b)

Figura 11. Aplicación móvil en funcionamiento mostrando una imagen desde la nube. (a) Con luz artificial. (b) En ausencia de luz.

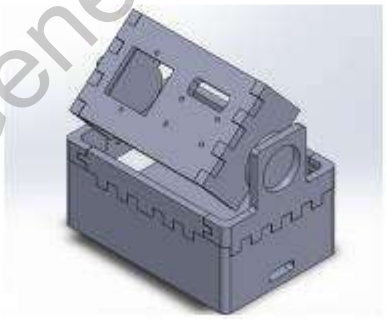


Figura 12. Modelo en 3D de la carcasa.

En la figura 14, se muestra una imagen donde el sistema captura a una persona pasando dentro del alcance del sistema de vigilancia. En este caso, la imagen se toma con luz visible. La figura 15 muestra el mismo caso, pero en esta ocasión, el lugar está a obscuras, por lo que se enciende la lámpara de infrarrojos.

La Mecatrónica en México, Enero 2021, Vol. 10, No. 1, páginas 35 – 46
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

Figura 13. Carcasa impresa con los elementos de hardware.

Figura 14. Imagen en condiciones de luz visible.

Figura 15. Imagen en condiciones de luz infrarroja.

2. Conclusiones

Después de pruebas de funcionamiento del sistema se determinó que puede usarse como una medida de prevención al dejarse funcionando al salir del sitio donde este se encontrase, mandando este imágenes en caso de que alguien entrase en la zona donde este opera, sin embargo, también se

encontraron algunos problemas relacionados principalmente con imágenes capturadas por el sistema cuando nada sucedía en la imagen, la mayoría de ellas cercanas en tiempo al desencadenamiento de la señal de salida del sensor PIR debido al movimiento de alguien, pero también algunas otras totalmente aleatorias, por ello futuro trabajos principalmente desde el punto de vista de procesamiento de imágenes podrían ser sumamente útiles para un problema como este.

Por otro lado, la frecuencia con la que se capturan imágenes una vez que detectó movimiento podría mejorarse, afortunadamente, al estar este prototipo desarrollado en Python y en Linux, es sumamente accesible realizar cambios o implementar nuevas secciones en cuanto a código se refiere.

Referencias

[1] I. N. de E. y Geografía (INEGI), «Incidencia delictiva», *Encuestas en establecimientos. Especiales. Encuesta Nacional de Victimización de Empresas. ENVE, Encuestas en hogares. Encuesta Nacional de Victimización y Percepción sobre Seguridad Pública. ENVIPE*, ene. 01, 2010. https://www.inegi.org.mx/temas/incidencia/ (accedido sep. 28, 2020).

[2] R. Macías Acosta, J. C. Macías Ponce, y M. Díaz Flores, «Programa para la Seguridad Nacional en México y su gasto aplicando sistemas de preferencias», *Rev. Venez. Gerenc.*, 2019, doi: 10.37960/revista.v24i2.31496.

[3] Lu Tan y Neng Wang, «Future internet: The Internet of Things», en *2010 3rd International Conference on Advanced Computer Theory and Engineering(ICACTE)*, ago. 2010, vol. 5, pp. V5-376-V5-380, doi: 10.1109/ICACTE.2010.5579543.

[4] M. Richardson y S. Wallace, *Getting Started with Raspberry Pi*. O'Reilly Media, Inc., 2012.

[5] «Buy a Raspberry Pi 3 Model B – Raspberry Pi». https://www.raspberrypi.org/products/raspberry-pi-3-model-b/ (accedido oct. 01, 2020).

[6] C. A. G. Godoy y O. J. S. Parra, «Sistema de seguridad para locales comerciales mediante Raspberry Pi, cámara y sensor PIR», *Rev. Virtual Univ. Católica Norte*, vol. 0, n.º 51, Art. n.º 51, ago. 2017.

[7] J. S. Agressoth Cardona y S. Murillo Román, «Sistema de seguridad electrónico IOT contra intrusión para inmuebles, con estructura de control, mediante aplicación en Android», nov. 2018, Accedido: oct. 14, 2020. [En línea]. Disponible en: http://repository.udistrital.edu.co/handle/11349/22444.

[8] A. J. K. Jayakumar y S. Muthulakshmi, «Raspberry Pi-Based Surveillance System with IoT», en *Intelligent Embedded Systems*, Singapore, 2018, pp. 173-185, doi: 10.1007/978-981-10-8575-8_19.

[9] A. Cunalata y D. Alberto, «Desarrollo de un prototipo de sistema de seguridad contra intrusos utilizando protocolos de IoT sobre la plataforma Zolertia Remote», jul. 2019, Accedido: oct. 14, 2020. [En línea]. Disponible en: http://bibdigital.epn.edu.ec/handle/15000/20318.

[10] J. Marín Rodríguez, «Desarrollo de un sistema de video vigilancia con servidor VPN Raspberry Pi y APP para móvil.», may 2020, Accedido: oct. 14, 2020. [En línea]. Disponible en: https://riunet.upv.es/handle/10251/143050.

[11] B. Fontal, *El espectro electromagnetico y sus aplicaciones*, vol. 1. Escuela Venezolana para la Enseñanza de la Química, 2005.

[12] «Artículo | Omniblug». http://www.omniblug.com/sensor-movimiento-pir-arduino.html (accedido oct. 23, 2020).

[13] M. Moghavvemi y Lu Chin Seng, «Pyroelectric infrared sensor for intruder detection», en *2004 IEEE Region 10 Conference TENCON 2004.*, nov. 2004, vol. D, pp. 656-659 Vol. 4, doi: 10.1109/TENCON.2004.1415018.

[14] García, Ginés, «Procesamiento Audiovisual», *ginés garcía mateos*, 2010. http://dis.um.es/profesores/ginesgm/pav.html (accedido oct. 23, 2020).

[15] Cambridge in Colour, «Compact vs. Digital SLR Cameras», 2020. https://www.cambridgeincolour.com/tutorials/compact-vs-digital-slr-cameras.htm (accedido nov. 01, 2020).

[16] «Camera Module - Raspberry Pi Documentation». https://www.raspberrypi.org/documentation/hardware/camera/ (accedido oct. 23, 2020).
[17] A. Ukil, J. Sen, y S. Koilakonda, «Embedded security for Internet of Things», en *2011 2nd National Conference on Emerging Trends and Applications in Computer Science*, mar. 2011, pp. 1-6, doi: 10.1109/NCETACS.2011.5751382.
[18] S. Thamburasa, S. Easwaramoorthy, K. Aravind, S. B. Bhushan, y U. Moorthy, «Digital forensic analysis of cloud storage data in IDrive and Mega cloud drive», en *2016 International Conference on Inventive Computation Technologies (ICICT)*, ago. 2016, vol. 3, pp. 1-6, doi: 10.1109/INVENTIVE.2016.7830159.
[19] F. Daryabar, A. Dehghantanha, B. Eterovic-Soric, y K.-K. R. Choo, «Forensic investigation of OneDrive, Box, GoogleDrive and Dropbox applications on Android and iOS devices», *Aust. J. Forensic Sci.*, vol. 48, n.º 6, pp. 615-642, nov. 2016, doi: 10.1080/00450618.2015.1110620.

Autores

Aldo Francisco Muñoz Vargas. Estudiante de la carrera de Ingeniería en Automatización de la Facultad de Ingeniería de la Universidad Autónoma de Querétaro. Miembro del capítulo de honor Mu Psi - Eta Kappa Nu de la IEEE.

Juan Manuel Ramos Arreguín. Tiene Doctorado en Ciencias y Tecnología con Especialidad en Mecatrónica en el Centro de Ingeniería y Desarrollo Industrial. Con Maestría en Ingeniería Eléctrica con opción en Instrumentación y Sistemas Digitales en la Universidad de Guanajuato. La ingeniería en Comunicaciones y Electrónica en la Universidad de Guanajuato. Fue Presidente de la Asociación Mexicana de Mecatrónica en el periodo 2013 a 2016. Pertenece al SNI en nivel I. Cuenta con reconocimiento al perfil PRODEP. A partir de 2017 es Senior Member en el IEEE. Profesor de tiempo completo en la Universidad Autónoma de Querétaro, en la Facultad de Ingeniería.

Saúl Tovar Arriaga. Obtuvo su grado de Licenciatura en Ingenieria en Electrónica en el Instituto Tecnológico de Querétaro, su Maestria en Ciencias en Mecatrónica en la Universidad de Siegen, Alemania, y su Doctorado en Ciencias Biomédicas en la Universidad de Erlangen-Nuremberg, Alemania. Actualmente es profesor de tiempo completo e investigador en la Universidad Autónoma de Querétaro. Sus intereses de investigación incluyen robótica médica, diagnostico automatizado por imagen y visión por computadora.

Marco Antonio Aceves Fernández. El Dr. Marco Antonio Aceves Fernández es Ingeniero en Telemática por la Universidad de Colima en el año 2000, obtuvo su Maestría y su Doctorado en el área de Sistemas Inteligentes en la University of Liverpool, Reino Unido, éste último en el año 2005. Ha sido reconocido como miembro del Sistema Nacional de Investigadores (SNI) por parte del CONACyT de manera ininterrumpida desde el 2009. Es miembro Senior de la IEEE y Presidente honorario de la Asociación Mexicana de Software Embebido. Sus intereses incluyen Sistemas Inteligentes y Embebidos.

Jesús Carlos Pedraza Ortega. Realizó sus estudios de Maestría en la FIMEE, Universidad de Guanajuato. Obtuvo el Doctorado en Ingeniería Mecánica con especialidad en Robótica - Sistemas de Reconstrucción 3D en la University of Tsukuba en Japón, donde trabajó con el desarrollo de un sistema monocular de reconstrucción 3D utilizando. Como docente, ha impartido diferentes cursos en los tres niveles de estudios (Licenciatura, Maestría y Doctorado) desde 1997, actualmente en la Universidad Autónoma de Querétaro. Es Senior Member por la IEEE y es miembro de la Academia Mexicana de Ciencias. Sus líneas de investigación son sistemas de reconstrucción 3D, inteligencia artificial aplicada a sistemas de visión, entre otros.

8.2. Capítulo de libro "Diseño Colaborativo en Mecatrónica" [51]

Diseño Colaborativo en Mecatrónica, Capítulo 11, pp. 141 - 153
ISBN: 978-607-9394-23-3, 2021

Detección de movimiento mediante procesamiento de imágenes aplicada a un sistema de seguridad

Aldo-Francisco Muñoz-Vargas[1], Juan-Manuel Ramos-Arreguín, Saúl-Tovar Arriaga, Marco-Antonio Aceves-Fernández, Jesús-Carlos Pedraza-Ortega, Edgar Alejandro Rivas Araiza

Universidad Autónoma de Querétaro, Facultad de Ingeniería
jsistdig@yahoo.com.mx, [1]aldomuoz98@gmail.com

Resumen

Este desarrollo presenta la solución a un problema de un sistema de seguridad en etapa de prototipo, aplicando visión por computadora con OpenCV. El prototipo es capaz de tomar fotografías cada vez que el sensor infrarrojo detecta el movimiento de alguna persona u objeto en su rango de operación. El sistema de vigilancia está formado por una tarjeta de desarrollo Raspberry Pi 3B+, y un sensor de movimiento PIR. Se toma fotografías siempre que se detecta movimiento, y en su primera versión, se tienen falsos positivos, en especial después de que el sensor envió una señal de cambio, tomando fotografías sin que exista movimiento, por lo que se almacenan fotos sin el requisito adecuado de movimiento. Debido a esto, se mejora el funcionamiento aplicando visión por computadora, aplicando un método de substracción de fondo y algunas adecuaciones mediante programación para obtener una solución funcional reduciendo el número de falsos positivos. Además, se analizan y comparan algunos otros métodos de detección de movimiento para determinar cuál es más adecuado, tales como MOG2 y KNN.

Palabras clave: Sistema de seguridad, visión por computadora, detección de movimiento, substracción de fondo, OpenCV, Raspberry Pi.

1. Introducción

1.1 Antecedentes

Algunos antecedentes de sistemas similares, tanto comerciales como prototipos desarrollados en sistemas embebidos como es el caso del sistema con el que se trabaja en este proyecto son presentados a continuación.

En 2016 Hossen et al. desarrollan un sistema de vigilancia basado en detección y estimación de movimiento usando flujo óptico [1], basado en el algoritmo Horn-Schunck, resultando en un sistema computacionalmente rápido sin requerir de hardware especial para el procesamiento de las imágenes. También se presenta un sistema de vigilancia por parte de Jayakumar y Muthulakshmi [2], un sistema embebido implementado en una tarjeta Raspberry Pi y usando IoT para almacenamiento de fotografías y envío de alertas; se usa detección de rostros para saber si la persona está autorizada o no. Yang et. al. presentan un sistema de vigilancia para interiores, con la capacidad de detección múltiple de personas, seguimiento y análisis de comportamiento [3].

En 2018 Pawlenka et al. desarrollan un sistema de seguridad considerando el incremento de la automatización del día a día, al estar este basado en un microcontrolador de un solo chip y sensores de

para detectar el acceso no autorizado a un área basados en contacto magnético, medición de monóxido de carbono, detección de movimiento y medición de temperatura y humedad; todo este conjunto de sensores justificados debido a que las casas inteligentes requieren de una manera de obtener información de su entorno. En este desarrollo un sensor PIR (detector de infrarrojos) fue utilizado como detector de movimiento. El procesamiento de las señales fue realizado mediante una red de microcontroladores y comunicación con protocolos como RS232 y RS485 [4].

En 2019 se presentó un sistema de seguridad físico que utiliza internet de las cosas mediante una plataforma llamada *Zolertia Remote*, tratando de dar una alternativa al de vigilancia. Consta de tres secciones, una red de sensores inalámbrica, un servidor privado *MQTT* (*Message Queue Telemetry Transport*) y una aplicación *Android* [5].

En 2020 se llevó a cabo una aplicación similar a las anteriores en la que mediante una *Raspberry Pi* como servidor se controló un conjunto de cámaras a través de una aplicación en *Android*, este sistema fue desarrollado de modo que en caso de detección de cambio del entorno se tomarían imágenes o videos guardándose en la memoria de la *Raspberry* y enviando un correo de alerta o notificaciones en la aplicación móvil, siendo estos datos accesibles a través de la aplicación que consultaría los datos del servidor [6].

De igual forma, en 2020 como trabajo de Zong Chen, un sistema de seguridad basado en sensores de movimiento e identificación de rostros fue desarrollado para detectar actividad sospechosa y reportar al posible infractor, este desarrollo utiliza Python, una Raspberry Pi, una base de datos de rostros y un sensor piroeléctrico, siendo probado en diferentes situaciones grabando video de las situaciones de prueba a las que se sometió el dispositivo [7].

Por otro lado, existen sistemas comerciales como es el caso de los dispositivos de la marca SEEDDARY (entre otras), los cuales suelen constar de una cámara con capacidad para visión diurna y nocturna, detección de movimiento, ajuste de ángulo de visión de la cámara dependiendo del modelo, grabación de video y vigilancia en tiempo real, así como opciones para consultar tales como son aplicaciones móviles y en el navegador [8]. Algo importante a destacar es que, aunque este tipo de sistemas es completamente funcional, estos son cerrados y no permiten la integración o experimentación con capacidades nuevas tales como podría hacerse mediante el uso de una herramienta como OpenCV o algún otro software, lo cual se busca en este desarrollo. Además, un sistema abierto como es en el que se desarrolló este prototipo, puede ser útil como sistema central, de monitoreo y posible control de factores ambientales dentro de un área específica.

1.2 Funcionamiento previo del dispositivo

En la figura 1, se puede observar el funcionamiento general del sistema de seguridad. La pieza central es la tarjeta Raspberry Pi 3B+, que tiene una comunicación continua con el dispositivo móvil Android. Este dispositivo se usa para configurar el funcionamiento del sistema de seguridad desde la aplicación móvil, la cual permite activar o desactivar el sistema y tomar capturas a voluntad, siempre y cuando se encuentre en la misma red local. La nube MEGA se usa para almacenar las imágenes capturadas y permite la consulta de las mismas desde cualquier lugar con una conexión a internet. Al detectar el sensor infrarrojo una presencia, se envía una señal para la captura de imágenes. Cuando lo que provocó el cambio desaparece del rango del sensor, se deja de enviar la señal. Como se observa en el diagrama, cada imagen registrada es nombrada con la fecha de captura para facilitar al usuario información sobre la misma.

El prototipo real se muestra en la figura 2, el diseño de la carcasa fue propuesto y diseñado de acuerdo a las necesidades, que en este caso eran poder ajustar el ángulo de visión de la cámara y el sensor, así como un diseño compacto [9]. En la tabla 1 se muestran los materiales que componen al sistema completo.

Diseño Colaborativo en Mecatrónica, Capítulo 11, pp. 141 - 153
ISBN: 978-607-9394-23-3, 2021

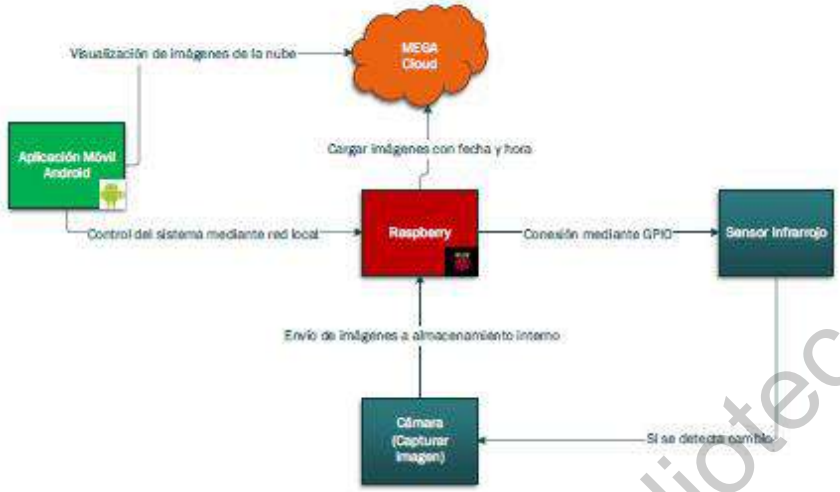


Figura 1. Funcionamiento general del sistema de seguridad.

Figura 2. Carcasa impresa con los elementos de hardware.

Tabla 1. Tabla de materiales.

Descripción	Cantidad
Raspberry Pi 3B+	1
NoIR Camera	1
PI Camera	1
5 V DC Power Supply	1
32 GB SD card	1
PIR sensor	1
Case	1

Es importante mencionar que la cámara que se usa tiene capacidades para visión nocturna. Sin embargo, las imágenes capturadas en condiciones de luz de día no son fieles en cuanto los colores. La cámara utilizada es propia de Raspberry siendo esta el modelo NoIR. Además, una lámpara de luz infrarroja es utilizada de manera independiente, y solo se activa en condiciones de baja iluminación, proveyendo de la radiación infrarroja necesaria para que la cámara pueda capturar información. En la

Diseño Colaborativo en Mecatrónica, Capítulo 11, pp. 141 - 153
ISBN: 978-607-9394-23-3, 2021

figura 3 y 4 se muestran ejemplos de captura del sistema en las dos condiciones anteriormente mencionadas, con luz visible e infrarroja, en esta última sin luz visible o en condiciones de obscuridad.

Figura 3. Imagen en condiciones de luz visible.

Figura 4. Imagen en condiciones de luz infrarroja.

Finalmente, se muestra la interfaz diseñada para la aplicación en Android en la figura 5, presentando imágenes capturadas. En el título de cada imagen se muestra la fecha y hora. En la figura 6 se muestran dos capturas de la aplicación en Android, donde una persona fue captada caminando frente al sistema mientras estaba funcionando.

Figura 5. Aplicación móvil en funcionamiento mostrando imágenes como archivos en la nube.

Diseño Colaborativo en Mecatrónica, Capítulo 11, pp. 141 - 153
ISBN: 978-607-9394-23-3, 2021

Figura 6. Aplicación móvil en funcionamiento mostrando una imagen desde la nube.

1.3 Problemática

El sensor infrarrojo PIR envía una señal cuando un objeto provoca un cambio en su elemento primario. Sin embargo, también enviaba la señal de detección de movimiento cuando no existía algún tipo de perturbación en el rango de operación, especialmente en momentos cercanos a una detección correcta, resultando en toma de imágenes sin información útil para el usuario, utilizando espacio de memoria tanto interna de la Raspberry como de la nube.

En la figura 7, se observa una captura de la muestra de imágenes tomadas con información que no correspondía con algún evento de detección de movimiento y algunas imágenes correctas, es posible notar algunas personas pasar, pero también imágenes sin ningún tipo de sujeto; esta captura es tomada del sistema de archivos generado por el sistema.

Figura 7. Muestra de imágenes del sistema de archivos del sistema de seguridad.

El sistema operó de una mejor forma en otras pruebas, teniendo una correspondencia clara entre las imágenes capturadas con algún objeto o sujeto en la imagen y el total de las imágenes capturadas, pero el caso mostrado en la figura 7 muestra uno de los peores escenarios que se presentaron, independientemente de ello, se puede observar y a través de más pruebas que se realizaron que era

Diseño Colaborativo en Mecatrónica, Capítulo 11, pp. 141 - 153
ISBN: 978-607-9394-23-3, 2021

totalmente seguro que si algún objeto activaba el sensor infrarrojo, se tomaría una imagen. Esta es la base para el desarrollo utilizando detección de movimiento con procesamiento de imágenes para mejorar la confiabilidad del sistema.

2. Metodología

De acuerdo al funcionamiento explicado en la sección anterior, se partió del hecho de que era seguro que el sensor detectase si algo se movía en su rango de operación, por lo que, a partir de ello, sería posible desarrollar un algoritmo mediante el cual, se analizaran una serie de imágenes consecutivas y así determinar si efectivamente algo se movía, tomando la decisión de capturar las imágenes.

2.1 Etapas de desarrollo

Investigación de algoritmos de detección de movimiento. En esta etapa se investigaron varias formas de detectar movimiento en una serie de imágenes consecutivas.

Captura de imágenes. Es necesario capturar imágenes a una tasa de fotogramas constante que permita el análisis utilizando Python y OpenCV, por lo que se investigó y experimentó una forma de lograr este objetivo a través de funciones disponibles para las herramientas mencionadas.

Prueba de los algoritmos investigados. Se probaron algunos de los métodos de substracción de fondo y se concluye que la mejor opción es substracción de fondo, considerando los requisitos del prototipo y su bajo costo computacional.

Implementación del algoritmo de detección de movimiento en la Raspberry Pi. Una vez conocido el funcionamiento y comportamiento del algoritmo de detección de movimiento, se modificó para que operara como una función de entrada y salida. Para esto, se reciben dos parámetros en la entrada, el primero es la imagen de fondo y el segundo es la imagen contra la cual comparar. La salida, que devuelve una respuesta booleana en caso de detectar o no movimiento. Debido a las características operativas del código (secuencial), se decide que la imagen de fondo debe ser tomada en el instante en que el sensor de movimiento detecte un cambio, debido a que el rango de detección del sensor es mayor que el alcance de visión de la cámara. Por lo tanto, la imagen obtenida como base queda sin ningún objeto o sujeto en la imagen. Cada imagen siguiente es analizada por el algoritmo de detección de movimiento comparándola con la imagen de fondo.

Ajustes de la integración del algoritmo de detección de movimiento: Una de las grandes limitaciones del algoritmo utilizado es su susceptibilidad a fallas en caso de cambios de iluminación. Como primera consideración para resolver este problema, se implementa en la Raspberry Pi la detección de movimiento, tomando la imagen sin cambios en el instante en que el sensor detecta movimiento. Sin embargo, debido a cómo se integra el algoritmo en Python, si ocurre un cambio drástico en la iluminación, se interpreta como un movimiento, capturando imágenes consecutivas sin detenerse hasta saturar la memoria. Para evitar esta saturación, se programa un límite haciendo que este proceso se detenga luego de cierta cantidad de imágenes. Esto es conveniente si lo que dispara el sistema, se mantiene mucho tiempo frente a él (mayor que el número de cuadros permitidos). Tan pronto como este objeto o sujeto se mueve, las imágenes son capturadas nuevamente, y así hasta que el fondo se mantenga constante.

A continuación, se presentan algunos métodos de detección de movimiento que fueron considerados en la implementación de esta solución.

2.2 Métodos de detección de movimiento mediante visión por computadora.

Substracción de fondo: La substracción de fondo es un paso de preprocesamiento importante en muchas aplicaciones basadas en la visión. Por ejemplo, un contador de visitantes donde una cámara

estática toma el número de personas que entran o salen de una zona, o una cámara de tráfico que extrae información sobre los vehículos. En todos estos casos, primero debe extraer a la persona o vehículos solos.

Si se tiene una imagen del fondo solo, como una imagen de la zona sin visitantes o una imagen de la carretera sin vehículos simplemente se resta la nueva imagen del fondo obteniendo imágenes solo de los objetos sobre el fondo. Sin embargo, es posible que no se tenga una imagen de este tipo, por lo que se deben utilizar otros métodos. Incluso objetos como sombras hacen más complejo el análisis dado que la sombra también se está moviendo, una simple resta marcará eso también como primer plano [10].

Substracción de fondo MOG y MOG2: MOG es un algoritmo de segmentación de fondo o primer plano basado en una mezcla gaussiana. Utiliza un método para modelar cada píxel de fondo mediante una mezcla de distribuciones K gaussianas (K = 3 a 5). Los pesos de la mezcla representan las proporciones de tiempo que esos colores permanecen en la escena. Los probables colores de fondo son los que permanecen más tiempo y más estáticos [10].

MOG2 también es un algoritmo de segmentación de fondo o primer plano basado en mezclas gaussianas. Una característica importante de este algoritmo es que selecciona el número apropiado de distribución gaussiana para cada píxel [10]. Este último es uno de los algoritmos probados en el sistema como se mostrará, a diferencia de MOG con el que se tuvo dificultades de implementar debido a incompatibilidad con versiones de OpenCV.

Substracción de fondo KNN: Se basa en el algoritmo de los k-vecinos más cercanos (k-NN). La entrada consta de los k ejemplos de entrenamiento más cercanos en el espacio de características. El resultado depende de si se utiliza k-NN para clasificación o regresión [11]:

- En la clasificación k-NN, la salida es una pertenencia a una clase. Un objeto se clasifica mediante un voto de pluralidad de sus vecinos, y el objeto se asigna a la clase más común entre sus k vecinos más cercanos (k es un número entero positivo, típicamente pequeño). Si k = 1, entonces el objeto simplemente se asigna a la clase de ese único vecino más cercano.

- En la regresión k-NN, la salida es el valor de propiedad del objeto. Este valor es el promedio de los valores de k vecinos más cercanos.

3. Resultados

3.1 Comparación entre métodos de detección de movimiento

Tres métodos fueron probados, aunque otros dos se mencionaron anteriormente, esto debido a errores con OpenCV en la Raspberry Pi. Todos los resultados presentados a continuación se probaron con condiciones de luz visible e infrarroja.

Método de sustracción de fondo: Para este método los pasos seguidos fueron los siguientes:

- Conversión de escala de grises y eliminación de ruido.
- Operación de resta entre el fondo y el primer plano.
- Aplicar un umbral a la imagen resultante de la resta.
- Detección de contornos o manchas.

La ecuación general que representa la substracción de fondo se presenta a continuación:

$$O_{img} = A_{img} - S_{img} \quad (1)$$

Diseño Colaborativo en Mecatrónica, Capítulo 11, pp. 141 - 153
ISBN: 978-607-9394-23-3, 2021

Donde O_{img} representa la imagen resultada debido a la substracción, A_{img} es la imagen con alteraciones respecto a S_{img} y S_{img} es la imagen del fondo estático.

Cabe mencionar que este es el método que se decidió utilizar para la operación final, esto es debido a que la primera imagen tomada o fondo funciona excelente para la aplicación, cada vez que el sensor detectaba algún movimiento el algoritmo comenzaba a funcionar, finalmente, la decisión de si se debe tomar una imagen o no depende de la imagen umbral, además de eso, los otros métodos están destinados a casos en los que el fondo no es constante o no es posible obtener una imagen del fondo.

Una imagen captada por el prototipo en una zona sin ningún tipo de cambio es la que se observa en la figura 8, siendo esta una puerta y una pared.

Figura 8. Imagen de fondo.

Una vez que se activa el sistema de seguridad y se ha detectado un cambio, se realiza el procesamiento para determinar si hay algo en la imagen, como se verá a continuación. La figura 9 muestra una mano, que desencadena el proceso de análisis, como imagen de referencia se utiliza la imagen de la figura 8.

Figura 9. Proceso de detección de movimiento utilizando substracción de fondo en condiciones de luz visible.

En la figura 9 se observa el proceso que ocurre en el algoritmo, en la primera imagen de izquierda a derecha se observa la imagen RGB, luego se observa la imagen en escala de grises, el siguiente paso es aplicar un filtro gaussiano para reducir detalles innecesarios , la cuarta fase es la resta entre la imagen de fondo original (que también pasó por los pasos antes mencionados) y la imagen con el cambio resultante en la cuarta imagen, finalmente, después de una operación de umbralización, una imagen con solo dos tonos, blanco o negro es obtenida, así al encontrar un píxel en blanco es suficiente para inferir detección de movimiento y por lo tanto guardar la imagen RGB original en la memoria. Para eso, un bucle que analiza varios píxeles en toda la imagen decide con una condición; se podría hacer analizando cada píxel, pero para hacer el proceso menos intensivo computacionalmente, se analizaron solo partes de la matriz de píxeles. El mismo proceso se observa en la figura 10, pero con luz infrarroja.

Diseño Colaborativo en Mecatrónica, Capítulo 11, pp. 141 - 153
ISBN: 978-607-9394-23-3, 2021

Figura 10. Proceso de detección de movimiento utilizando substracción de fondo en condiciones de luz infrarroja.

Método de substracción de fondo MOG2: Para este método, se utilizó la función que estaba incluida en OpenCV. La integración con el sistema de seguridad fue sencilla, aunque no hubo control de los parámetros como en el método de sustracción de fondo. También se encontraron los contornos del objeto para que se pudiera dibujar un rectángulo alrededor de la imagen para indicar dónde estaba la diferencia, este proceso también se implementó para los resultados finales. La imagen de la izquierda en la figura 11 llamada "Umbral" es el resultado de este método.

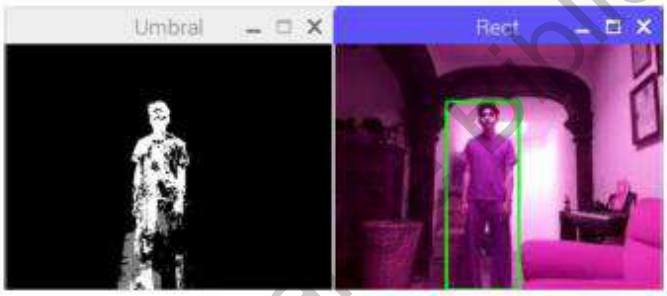


Figura 11. Detección de movimiento utilizando substracción de fondo MOG2 en condiciones de luz visible.

En la figura 12, se presenta el mismo resultado con luz infrarroja, nótese que este método también es susceptible a la detección de sombras como objetos adicionales.

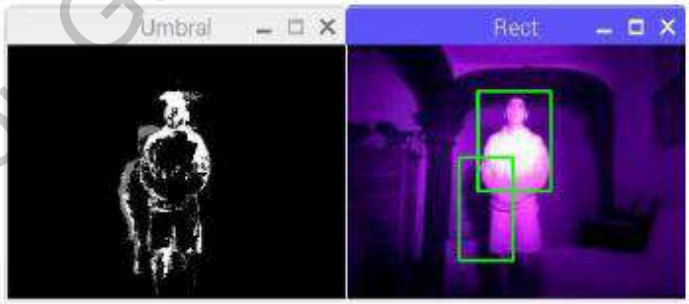


Figura 12. Detección de movimiento utilizando substracción de fondo MOG2 en condiciones de luz infrarroja.

Método de substracción de fondo KNN: La forma en que se implementó es la misma que la anterior. En las figuras 13 y 14 se pueden ver los resultados en condiciones de luz visible e infrarroja.

Diseño Colaborativo en Mecatrónica, Capítulo 11, pp. 141 - 153
ISBN: 978-607-9394-23-3, 2021

Figura 13. Detección de movimiento utilizando substracción de fondo KNN en condiciones de luz visible.

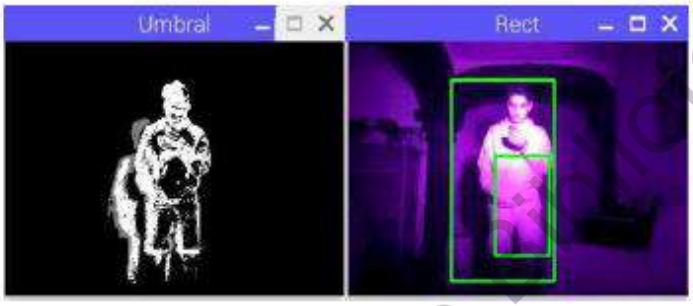


Figura 14. Detección de movimiento utilizando substracción de fondo KNN en condiciones de luz infrarroja.

Se puede notar que los dos últimos métodos son bastante similares en resultados, aunque no se puede mostrar en este texto, cuando el sujeto en las imágenes se mantuvo quieto, ambos algoritmos comenzaron a hacer como si este desapareciera, mientras que el método de substracción de fondo puro no, esa es otra razón de que sea el método a implementar en el prototipo final, sin embargo, cualquier método podría ser útil para esta aplicación ya que el seguimiento no es relevante para este proyecto.

3.2 Resultados finales

En las siguientes figuras, el algoritmo de substracción de fondo se presenta nuevamente, en la ubicación donde se probaron los otros dos métodos MOG2 y KNN, también, se agregó el rectángulo de seguimiento.

En la figura 15 se muestra la implementación mediante Python de tres barras utilizadas para determinar los parámetros que funcionan bien para este proceso. La primera cambia el primer parámetro del filtro gaussiano, el parámetro "ksise", la segunda el parámetro "sigma" que es parte del filtro de desenfoque gaussiano y la tercera barra "Thresh" cambia el umbral de lo que se considera parte del objeto en movimiento y lo que no.

Figura 15. Barras para configurar parámetros del algoritmo de detección de movimiento.

Diseño Colaborativo en Mecatrónica, Capítulo 11, pp. 141 - 153
ISBN: 978-607-9394-23-3, 2021

La conclusión de la calibración de estos parámetros fue que "ksise" debe ser alto como se presenta en la última imagen, en realidad tiene un valor de 21 mientras sigma es cero, esto hace que la imagen con filtrado gaussiano sea extremadamente borrosa, que es lo mejor para este método, de modo que cuando alguien entra en el rango de visión, puede ser tratado como un solo objeto.

Para el valor de umbral, se determinó que alrededor de 30 de 255 era lo suficientemente bajo para esta aplicación.

Los resultados con condiciones de luz visible se presentan en la figura 16.

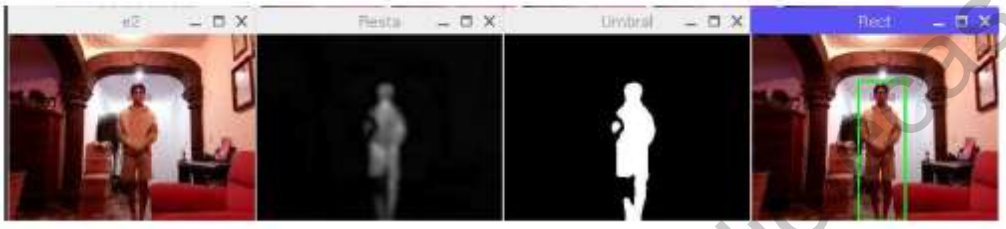


Figura 16. Resultados en condiciones de luz visible.

Por otro lado, con condiciones de luz infrarroja, el resultado es mostrado en la figura 17.

Figura 17. Resultados en condiciones de luz infrarroja.

Cada vez que se detecta un píxel blanco en la imagen "Umbral" (figura 16 y 17), el código almacena el último cuadro correspondiente a movimiento, también debe tenerse en cuenta que ninguna de las imágenes anteriores está planeada para ser observada por el usuario, estas se muestran solo con fines ilustrativos, el usuario puede ver la imagen contenida en la ventana "rect" únicamente.

Cada imagen se almacena en una carpeta con la fecha, y cada imagen también tiene la fecha en que se tomó como nombre, como se observa en la figura 5.

Comparación con el prototipo sin detección de movimiento: En la figura 18, se tiene una gráfica del funcionamiento del prototipo sin detección de movimiento por aproximadamente 24 horas, en el eje horizontal se tiene el tiempo de funcionamiento, mientras en el vertical, eventos de captura de imágenes, cada punto verde representa un dato donde la imagen muestra información que corresponde con detección de movimiento por parte del sensor, por otro lado, cada punto rojo corresponde con imágenes capturadas en total, cada punto representa el número de imágenes capturadas en un intervalo de una hora, por lo tanto el número de imágenes erróneas es la diferencia entre el número total de imágenes capturadas menos el número de imágenes correctas.

Por lo tanto, se puede observar en la figura 18, que el sistema, aunque cumple con su objetivo de registrar imágenes de algún objeto o sujeto que cruzó en su campo de operación, también captura imágenes en momentos donde no existió ningún tipo de evento, debe señalarse que esta gráfica

Diseño Colaborativo en Mecatrónica, Capítulo 11, pp. 141 - 153
ISBN: 978-607-9394-23-3, 2021

corresponde con el funcionamiento en una de las peores condiciones, sin embargo, este solía entregar imágenes que sí correspondían con movimiento. Se sospecha que este hecho podría estar relacionado con algún tipo de interferencia con señales electromagnéticas debido al tipo de cableado utilizado en el prototipo al no haber utilizado circuitos impresos como medio de conexión entre el sensor y la tarjeta embebida.

Figura 18. Funcionamiento del prototipo anterior.

En contraste, una prueba de funcionamiento de alrededor de 24 horas fue realizada empleando la substracción de fondo para la detección de movimiento, esta vez con resultados satisfactorios al capturar imágenes debidas a movimiento en el rango de operación, lo anterior se describe en la figura 19, donde se aprecian eventos de captura de imágenes correctos. En el evento cercano a 40 capturas de imágenes se puede observar que existen algunas imágenes tomadas extra con respecto al número de eventos correctos, esto fue debido al cambio de posición de un objeto del fondo, por lo que al ser comparada la imagen con el objeto en una posición diferente contra la imagen con el fondo estático se obtuvo una diferencia.

Figura 19. Funcionamiento del prototipo después de la implementación del algoritmo de detección de movimiento.

Diseño Colaborativo en Mecatrónica, Capítulo 11, pp. 141 - 153
ISBN: 978-607-9394-23-3, 2021

4. Conclusiones

Si bien el método elegido no es perfecto por las condiciones de luz, al tener problemas con las sombras, la forma no es tan importante para el sistema de vigilancia, con solo poder detectar un píxel blanco como se explicó anteriormente, es suficiente para tomar imágenes, esto hace el prototipo más confiable que la versión anterior. Además, al ser un dispositivo desarrollado en software libre (OpenCV y Python) y una plataforma como es Linux, es totalmente viable, realizar mejoras o emplear el prototipo en alguna otra aplicación donde se requiera de detección de movimiento. Finalmente, disponiendo aun de entradas y salidas digitales, otro tipo de sensores podrían instalarse al dispositivo ofreciendo mayor información acerca de un área.

Referencias

[1] M. K. Hossen y S. H. Tuli, "A surveillance system based on motion detection and motion estimation using optical flow", en *2016 5th International Conference on Informatics, Electronics and Vision (ICIEV)*, may 2016, pp. 646–651. doi: 10.1109/ICIEV.2016.7760081.

[2] A. J. K. Jayakumar y S. Muthulakshmi, "Raspberry Pi-Based Surveillance System with IoT", en *Intelligent Embedded Systems*, Singapore, 2018, pp. 173–185. doi: 10.1007/978-981-10-8575-8_19.

[3] C. Yang, T. Chou, F. Chang, C. Ssu-Yuan, y J. Guo, "A smart surveillance system with multiple people detection, tracking, and behavior analysis", en *2016 International Symposium on VLSI Design, Automation and Test (VLSI-DAT)*, abr. 2016, pp. 1–4. doi: 10.1109/VLSI-DAT.2016.7482569.

[4] T. Pawlenka y J. Škuta, "Security system based on microcontrollers", en *2018 19th International Carpathian Control Conference (ICCC)*, may 2018, pp. 344–347. doi: 10.1109/CarpathianCC.2018.8399653.

[5] A. Cunalata y D. Alberto, "Desarrollo de un prototipo de sistema de seguridad contra intrusos utilizando protocolos de IoT sobre la plataforma Zolertia Remote", jul. 2019, Consultado: oct. 14, 2020. [En línea]. Disponible en: http://bibdigital.epn.edu.ec/handle/15000/20318

[6] J. Marín Rodríguez, "Desarrollo de un sistema de video vigilancia con servidor VPN Raspberry Pi y APP para móvil.", may 2020, Consultado: oct. 14, 2020. [En línea]. Disponible en: https://riunet.upv.es/handle/10251/143050

[7] Dr. J. I. Zong Chen, "Smart Security System for Suspicious Activity Detection in Volatile Areas", *JITDW*, vol. 02, núm. 01, pp. 64–72, mar. 2020, doi: 10.36548/jitdw.2020.1.006.

[8] "Seedary | Tienda Online". https://seedaryoficial.mercadoshops.com.mx/ (consultado abr. 29, 2021).

[9] Muñoz A., Ramos J., et al. "*Sistema de seguridad basado en un sistema embebido con cámara para día y noche con enlace IoT*", La Mecatrónica en México, México, Vol. 10 No. 1 Consultado: abril 25, 2021. [En línea]. Disponible en: http://www.mecamex.net/revistas/LMEM/revistas/LMEM-V10-N01-03.pdf

[10] "Background Subtraction — OpenCV 3.0.0-dev documentation". https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_video/py_bg_subtraction/py_bg_subtraction.html (consultado dic. 27, 2020).

[11] "Detección de movimiento con OpenCV y Python", *Programar fácil con Arduino*. https://programarfacil.com/blog/vision-artificial/deteccion-de-movimiento-con-opencv-python/ (consultado dic. 27, 2020).