MARE INGENII. INGENIERÍAS

Sistema de seguridad con reconocimiento facial en módulo ESP32

Facial Recognition Security System in ESP32 Module

https://doi.org/10.52948/mare.v4i1.684

RAÚL ALVAREZ SALVADOR
https://orcid.org/0000-0003-4189-9940
Empresa de Telecomunicaciones de Cuba (ETECSA), Cuba
raul.alvarezsalvador@etecsa.cu

CARLOS ALBERTO BAZÁN PRIETO
https://orcid.org/0000-0002-2810-9599
Universidad Central "Marta Abreu" de Las Villas (UCLV), Cuba
cabazan@uclv.edu.cu

Artículo de investigación
Recepción: 18 de enero de 2022
Aceptación: 13 de septiembre de 2022

Cómo citar este artículo
R. Álvarez y C. Bazán, "Sistema de seguridad con reconocimiento facial en módulo ESP32," Mare Ingenii, vol. 4, n.° 2, oct. 2022.

Resumen:
Un sistema de seguridad es un conjunto de instalaciones necesarias para brindar seguridad a las personas y bienes que se encuentran en un lugar específico. Cuando la seguridad es suficiente a través de un sistema de cámaras, se conoce como videovigilancia. En algunos casos, se vincula al reconocimiento facial, para identificar o verificar personas. Este trabajo propuso un sistema de hardware y software de código abierto para crear un sistema de videovigilancia de bajo costo. Utiliza el módulo ESP32-CAM, que dispone, entre otros recursos, de una cámara de dos Mp, tarjeta microSD para almacenar datos y comunicación wifi. Analiza las herramientas de desarrollo disponibles en el entorno Arduino, así como las librerías disponibles para el ESP32 y para el reconocimiento facial. Finalmente, propone una configuración mínima para solucionar el control de centros aislados en la Empresa de Telecomunicaciones de Cuba S.A. (ETECSA).

Palabras clave: videovigilancia; módulo ESP 32; reconocimiento facial.

Abstract:
A security system is a set of necessary facilities that works to provide security to people and property that are in a specific place. When security is enough through a camera system, it is known as video surveillance. In some cases, it linked to facial recognition, to identify or verify people. This work proposed an open-source hardware and software system to create a low-cost video surveillance system. It uses the ESP32-CAM module, which has, among other resources, a two Mp camera, microSD card to store data and Wi-Fi communication. It analyzes the development tools available in the Arduino environment, as well as the libraries available for the ESP32 and for facial recognition. Finally, it proposes a minimum configuration to solve the control of isolated centers in ETECSA.

Keywords: video surveillance; ESP32 module; facial recognition.

Introducción

La videovigilancia tuvo una evolución acelerada en la última década, impulsada por los avances tecnológicos y una preocupación creciente por la seguridad pública y privada. Es un sistema que permite captar imágenes fijas o en movimiento con mayor alcance, visión, resolución; posibilita su almacenamiento, consulta y tratamiento. Sus comienzos se remontan al surgimiento de una nueva forma de vigilar, pasar de aquella de tipo sensorial a la del auxilio de las herramientas y dispositivos electrónicos desde el video analógico. Ahora los avances han permitido la combinación entre aquellos y los digitales hasta llegar a un entorno completamente digital. Este último sigue desarrollándose en el camino del software y del hardware, buscando tener un sistema de vigilancia cada vez más inteligente, lo que conlleva a poseer una amplia cantidad de funciones que son posibles gracias a la tecnología avanzada [1-8].

En contraste con los dispositivos de videovigilancia comerciales, en [7] se analizan los sistemas de videovigilancia que utilizan hardware y software libres. Aquí se concluye que, a diferencia de los sistemas comerciales costosos, estos son escalables y flexibles para aumentar el número de las cámaras, agregar módulos con distintos objetivos como por ejemplo, reconocimiento facial como uno de los principales objetivos.

Uno de los factores prioritarios para la identificación de los rostros es la calidad de la imagen, lo que permitirá que el algoritmo funcione de mejor manera. El algo-

ritmo tiene que tener en cuenta otros factores con los cuales se dificulta el proceso de detección como el estado anímico de la persona, ubicación del rostro, accesorios (lentes, barba, gorros, mascarillas, etc.), condiciones lumínicas y cantidad desconocida de caras en la imagen [9].

Las técnicas para el reconocimiento facial más utilizadas en este tipo de aplicación se clasifican en [4] como: basadas en rasgos faciales (holísticas), basadas en la imagen, basadas en videos, geométricas (reconocimiento facial 2D y 3D) y análisis de la textura de la piel. En [10] se desarrolla un sistema de control de seguridad biométrico de reconocimiento facial para el ingreso y egreso. Se utilizaron dos herramientas y una biblioteca de tipo *Open Computer Vision* (OpenCV) para la detección de rostros con la técnica Haar y el uso del API Kairos. Por otra parte, en [11] se desarrolla un sistema de reconocimiento facial con la metodología de reconocimiento facial Eigen-faces junto al análisis de componentes principales (PCA). Comprueban la baja complejidad computacional y poca utilización de recursos de la imagen, para un reconocimiento del 97% de precisión.

En este trabajo se presenta un análisis de las herramientas de hardware y software libre, y de bajo costo, para la implementación de un sistema de videovigilancia con reconocimiento facial.

Herramientas de software y hardware libres para videovigilancia

El software libre es aquel que respeta la libertad de los usuarios sobre su producto adquirido y, por tanto, debe venir acompañado del código fuente para hacer efectivas las libertades. De forma general, su principal promotor, Richard Stallman, lo define como la libertad para ejecutar, copiar, distribuir, estudiar, modificar y mejorar el software. Al igual que en el software código fuente abierto, la denominación de hardware de arquitectura abierta se refiere a la libertad de poder utilizar el dispositivo y su documentación, no a que sea necesariamente gratuito. Aunque comparta filosofía con el software código fuente abierto, debido a la propia naturaleza de estos componentes físicos, muchos de sus preceptos son limitados [12], [13].

En los últimos años el abaratamiento de los sistemas de computación y comunicación ha sido espectacular. Computadores embebidos en una placa o computador de placa reducida SBC (del inglés, *Single Board Computer*) como el Raspberry Pi y plataformas de hardware abierto para diseño de redes de sensores, como Arduino que permite un diseño muy barato de un sistema de videovigilancia, de forma rápida y con un manejo muy sencillo por parte del usuario final [14].

Un miniordenador o computador de placa reducida SBC es una placa electrónica que tiene todas las características de una computadora funcional, pero a bajo costo. Entre las principales ventajas están el espacio que ocupan y el bajo consumo de energía. Como desventajas, la dificultad de realizar una ampliación en el hardware ya que se trata de una placa base que contiene componentes integrados, así como la difícil reparación [6].

Existen una serie de computadores de placa reducida como: Odroid, Jaguardboard, Orange Pi, Hummingboard, Raspberry Pi, que se usan fundamentalmente como computadoras de propósito general, para servidores de juegos, captura y transmisión de audio y video [10]. Tienen la posibilidad de instalar sistemas operativos como Windows y Ubuntu. Además, se pueden utilizar como esta-

ción de trabajo para el desarrollo de software. Cuentan con un extenso número de proyectos disponibles en internet, así como una amplia bibliografía, además permiten implementar la arquitectura de software empleando frameworks de código abierto [3]. Raspberry Pi, se destaca como uno de los más usados.

Existen varias plataformas de hardware abierto, pero es indudable la popularidad de la plataforma Arduino y de su comunidad de desarrolladores. Su IDE open-source, basado en lenguaje Processing/Wiring, puede ser descargado gratuitamente (actualmente para Mac OS X, Windows y Linux) [15], [2]. Entre las placas más comercializadas de Arduino, según [16] se encuentran: Arduino UNO (basada en el microcontrolador ATmega328), Arduino Mega (basada en Atmega2560, con mayores prestaciones pero compatible con Arduino UNO), Arduino Nano (posee las mismas características que el Arduino UNO, pero diseñado para proyectos con espacio limitado), Arduino Mini (similar a la anterior pero pensado para proyectos que van a quedar en funcionamiento permanente, no posee puerto USB y para programar la placa se debe adaptar un USB-Serie), entre otros.

Otra de las plataformas de hardware abierto, en este caso basadas en un sistema en chip (SoC, del inglés *System on a Chip*) con hardware específico para comunicaciones inalámbricas, entre otros recursos, es el ESP32. Este permite el desarrollo de aplicaciones en diferentes lenguajes de programación, *frameworks*, bibliotecas y recursos diversos. Los más comunes a elegir son: Arduino (en lenguaje C/C++), Esp-idf (Espressif IoT Development Framework) desarrollado por el fabricante del chip; Simba Embedded Programming Platform (en lenguaje Python); RTOS's (como Zephyr Project, Mongoose OS, NuttX RTOS); MicroPython, LUA, Javascript (Espruino, Duktape, Mongoose JS), Basic, entre otros [17].

La posibilidad de trabajar dentro del entorno Arduino permite utilizar un lenguaje de programación conocido y hacer uso de un IDE sencillo de usar, además de hacer uso de toda la información sobre proyectos y bibliotecas disponibles en internet. La comunidad de usuarios de Arduino es muy activa y da soporte a plataformas como el ESP32 y ESP8266. Dentro de las principales placas de desarrollo, o módulos basados en el ESP32, están: ESP32-WROOM-32, NodeMCU-32 ESP32 y ESP32-CAM y de la familia ESP8266 tenemos: ESP-01, ESP-12E, Wemos D1 mini y NodeMCU v2.

El SoC ESP32 de Espressif Systems es la evolución del ESP8266, diseñado para superar a su antecesor en capacidad de procesamiento y conectividad; integra un potente microcontrolador con arquitectura de 32 bits, conectividad wifi y Bluetooth. El sistema en el módulo (SoM, del inglés *System on Module*) ESP-32S fabricado por Ai-Thinker integra en un módulo el SoC ESP32, memoria FLASH, cristal oscilador y antena wifi en PCB [17].

Ahora bien, se destaca el módulo ESP32-CAM (Fig. 2), que incluye una cámara OV2640 y varios GPIO para conectar periféricos usando un ESP32. El módulo también cuenta con una ranura para tarjeta microSD, que permite almacenar imágenes tomadas de la cámara o almacenar archivos y viene con el módulo de cámara de 2MP.

Fig. 1. ESP32-CAM.

Las funcionalidades de este módulo y su capacidad de realizar diversas tareas, lo ha convertido en un componente eficaz para la seguridad. Este módulo se puede utilizar en videovigilancia para transmitir imágenes o funcionar como sensores en un sistema de visión. Aún con toda la carga de procesamiento computacional, este módulo tiene potencia para hacer otras tareas como reconocimiento facial y puede ser implementado en sistemas como cámaras de seguridad CCTV, visión artificial embebida, visión remota para robots móviles, entre otras funciones [5]. Por estas grandes ventajas y su bajo costo, en este trabajo se selecciona al ESP32 CAM como la herramienta de hardware libre a implementar en el sistema de videovigilancia con reconocimiento facial.

Existen varios recursos de software libre para videovigilancia y reconocimiento facial, como la Biblioteca OpenCV (*Open Source Computer Vision Library*). Esta es una biblioteca de código abierto que brinda funciones y algoritmos para la utilización de visión artificial, procesamiento de imágenes y algoritmos numéricos, proporciona herramientas para el procesamiento de imágenes, incluyendo el reconociendo de objetos en fotografías y videos (caras, figuras de personas, textos, etc.) [5], [16]. Lanzada formalmente en enero de 1999, se considera la biblioteca de software libre más relevante cuando de visión artificial se trata [18]. Para su construcción se empleó el lenguaje de programación C++, aunque tiene bibliotecas para realizar proyectos sobre Java, C, C++, Python. Además, se emplea para realizar proyectos sobre diferentes sistemas operativos como: Android, Linux, MAC y Windows. La biblioteca es usada frecuente en proyectos que requieran funciones como reconocimiento y detección facial, seguimiento, identificación de objetos, sistemas de seguridad, etc.

Para el caso de la plataforma Arduino, se dispone de un software de gran utilidad con un entorno de desarrollo integrado IDE (del inglés, *Integrated Development Environment*) que implementa el lenguaje de programación Arduino y el bootloader ejecutado en la placa [2]. El software se llama Wiring, basado en la plataforma processing y utiliza el lenguaje de programación C/C++. El software de Arduino (Arduino IDE) tiene una interface gráfica muy amistosa, dispone de variedad de bibliotecas que permiten el control de los periféricos que se conecten a la placa y, sobre todo, de una gran comunidad de desarrolladores.

En particular, existe la biblioteca ESP-DL disponible en [19], para la configuración del ESP 32 CAM. Incluye recursos de aprendizaje profundo de alto rendimiento dedicada a ESP32, ESP32-S2, ESP32-S3 y ESP32-C3. También proporciona una interfaz de programación de aplicaciones (API, del inglés *Application Programming Interface*) para inferencia de redes neuronales (NN, del inglés Neural Network), procesamiento de imágenes, operaciones matemáticas y algunos modelos de aprendizaje profundo. Con ESP-DL, se pueden utilizar los SoC de Espressif para aplicaciones de inteligencia artificial (IA) de forma fácil y rápida.

Dicha biblioteca se puede utilizar en aplicaciones procesamiento de imágenes, detección y reconocimiento facial, etc.

Para el caso del módulo ESP32 CAM, existe la Biblioteca esp32cam. Proporciona una API orientada a objetos para usar la cámara OV2640 en el ESP32 CAM. Esta biblioteca ha sido probada con la placa AI Thinker ESP32-CAM y la cámara OV2640.

El repositorio del controlador de ESP32 CAM aloja el controlador compatible el SoC de la serie ESP32 para sensores de imagen. También proporciona algunas herramientas que permiten convertir los datos del cuadro capturado a los formatos BMP y JPEG más comunes.

El reconocimiento facial, generalmente, implica tres procesos clave en su tratamiento: detección de rostros, extracción de características y clasificación. Desde inicios del siglo XXI, se desarrollaron soluciones completas, que hoy se conocen como Software Development Kit (SDK) y bibliotecas, por ejemplo, OpenCV, FaceTracker. De igual manera, con el inicio de la tecnología en la nube aparecen las herramientas soportadas por API, como lo es Microsoft Azure [5].

Entre las herramientas de reconocimiento facial recopiladas en [5] se encuentran:

A. Kairos API

Es una aplicación de interfaz programable dedicada al reconocimiento facial cuyo sistema renderiza avatares 3D a partir de imágenes o videos capturados en 2D, donde puede detectar características o estados de ánimo de un individuo por la expresión que realice el sujeto cuando es capturada la imagen. Este sistema no solo identifica una imagen, también permite analizar las emociones de un individuo a través de su expresión.

B. OpenFace

Permite realizar el reconocimiento facial a través de redes neuronales profundas, dicho algoritmo permite identificar los rostros en tiempo real.

C. Open-Source Biometrics (Open BR)

Es un framework para la investigar nuevas modalidades, mejorar los algoritmos existentes, medir el rendimiento del reconocimiento y desarrollo de sistemas biométricos automatizados. Los algoritmos estándares también están disponibles para modalidades específicas que incluyen reconocimiento facial, estimación de edad y estimación de género.

D. API de Microsoft Face

Especializado en IA y en dominio de visión artificial, dispone de aplicaciones centradas en la nube que permiten mejorar los sistemas de reconocimiento. En el caso específico de reconocimiento facial con el módulo ESP32 CAM, se dispone de la biblioteca face_recognition. Esta es una API de reconocimiento facial que permite distinguir el rostro de una persona al comparar con una base de datos preestablecida; se basa en una nueva arquitectura móvil llamada MobileNetV2 y el algoritmo ArcFace (modelo de aprendizaje automático que toma dos imágenes faciales como entrada y las compara usando la distancia del coseno); un método utilizado por los motores de búsqueda y puede calcularse mediante el producto interno de dos vectores normalizados. Se utiliza el algoritmo ArcFace, en lugar de la función tradicional Softmax, y la función de pérdida de entropía cruzada (Cross-Entropy Loss). Para reducir la complejidad del cálculo, se utilizan imágenes de menor tamaño (56x56) en el entrenamiento.

Resultados y discución

Los recursos necesarios para programar y gestionar el módulo ESP32 con el IDE de Arduino en Windows, se describen en los proyectos que se encuentran disponibles en Espressif [20]. Por otra parte, para utilizar el módulo ESP32-CAM se deben incluir una serie de bibliotecas y herramientas junto con el Arduino IDE. Para ello se debe agregar el enlace https://dl.espressif.com/dl/package_esp32_index.json dentro del Archivos de Preferencias del IDE Arduino (Fig. 2). Luego en el menú "Herramientas – Gestor de tarjetas" se busca ESP32 y se instala.

Para el control de la cámara del módulo se necesita la biblioteca "esp32cam" que se encuentra disponible en [21].

Esta biblioteca se puede añadir mediante: Programas – Incluir bibliotecas – Añadir biblioteca ZIP.

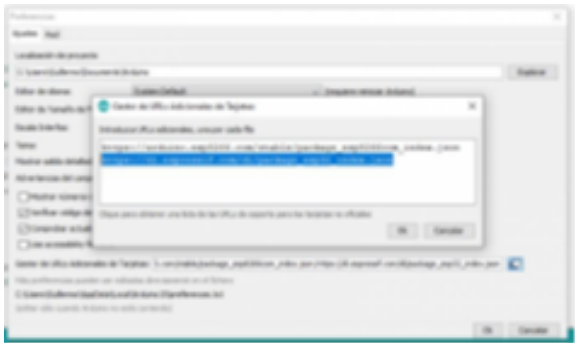

Fig. 2. Biblioteca ESP32.

Para comprobar el funcionamiento, se toma un ejemplo de la biblioteca, que convierte el ESP32-CAM en una cámara IP. Puede encontrarlo en Archivos – Ejemplos – esp32cam – WifiCam.

En el ejemplo solo se deben editar las líneas indicando el nombre de la red inalámbrica y su contraseña.

- const char* WIFI_SSID = "my-ssid"
- const char* WIFI_PASS = "my-pass"

Seleccionar el modelo de la Cámara:

- #define

CAMERA_MODEL_AI_THINKER

Para la detección y reconocimiento de rostros es necesario incluir las bibliotecas, fr_flash.h, fr_forward.h y fd_forward.h disponibles en [22].

```
#include <fd_forward.h> // detection de rostros
#include <fr_forward.h> // reconocimiento de rostros
#include <fr_flash.h> // manejo del almacenamiento en la flash
```

Ahora bien, es necesario definir algunas variables y objetos globales:

```
// Face detection/recognition variables
camera_fb_t *fb; // Frame buffer pointer for picture from camera
box_array_t *detected_face; // Information for a detected face
dl_matrix3du_t *image_matrix; // Image matrix pointer
dl_matrix3du_t *aligned_face; // Aligned face pointer
dl_matrix3d_t *face_id; // Face ID pointer
face_id_node *face_recognized; // Recognized face pointer
mtmn_config_t mtmn_config; // MTMN detection settings
```

Para cargar el programa, el terminal GPIO 0 debe estar conectado a GND en el módulo ESP32. Para cargar el programa en el ESP32, debido a que no cuenta con un puerto USB, será necesario utilizar un módulo USB, se utiliza el siguiente esquema (Fig. 3):

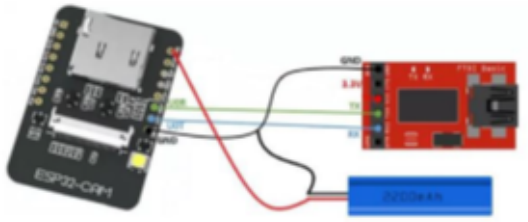

Fig. 3. Cableado ESP32-CAM con Modulo USB.

Presionar el botón de reinicio, después de lo cual el ESP32 entrará en modo flash, lo que le permitirá cargar el programa. Después de cargar el programa se desconecta el terminal GPIO 0 de GND y presionar nuevamente el botón de reinicio, para correr el programa cargado. Al abrir el monitor serial debe mostrar la dirección IP de su ESP32-CAM como se muestra en la Fig. 4.

Fig. 4. Vista dirección IP de la Cámara.

El servicio web que brinda esta aplicación permite configurar la resolución de la cámara (Fig. 5) y activar el módulo de reconocimiento facial y detección de movimiento (Fig. 6).

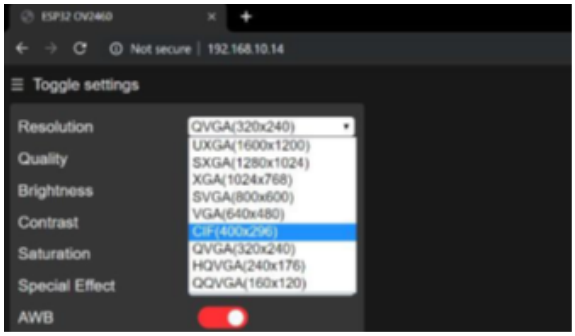

Fig. 5. Resolución de la cámara.

Se recomienda usar la resolución Common Intermediate Format (CIF) (400 x296). Para activar el módulo de reconocimiento facial y detección de movimiento es necesario activar ambos módulos como muestra la Fig. 6.

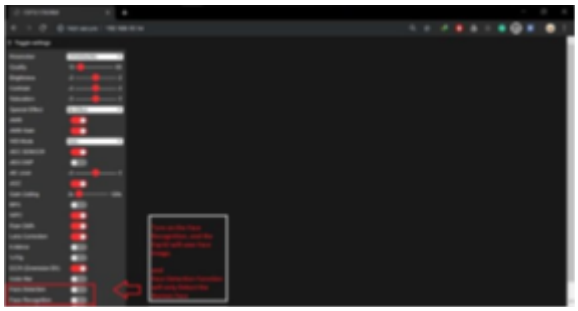

Fig. 6. Activación módulos detección de movimiento y detección de rostro.

Con estas bibliotecas también es sencillo integrar estas aplicaciones con Python. Se pueden tomar imágenes desde Python con el programa disponible en [23]. Solo se debe tener en cuenta reemplazar la IP de la URL por la correcta en el programa, luego ejecutarlo y cada vez que se ejecute toma una foto.

La Empresa de Telecomunicaciones de Cuba (ETECSA) tiene algunos centros aislados, que no disponen de vigilancia. Con el módulo ESP32 CAM y los recursos de software disponibles, es posible instalar un sistema de seguridad con reconocimiento facial y controlarlos a distancia. Estos centros disponen de red y alimentación ininterrumpida, por lo que es posible su funcionamiento seguro.

Conclusiones

Los recursos de hardware y software de código abierto, módulo ESP32-CAM y las herramientas de programación de libre acceso disponibles, permiten crear un sistema de videovigilancia de bajo costo. Entre otros recursos puede disponer de una cámara de 2 Mp, tarjeta microSD para almacenar datos y comunicación wifi. Las herramientas de desarrollo disponibles en el entorno Arduino, así como de las bibliotecas disponibles para el ESP32 y para el reconocimiento facial permiten crear aplicaciones de utilidad para videovigilancia. Esta configuración mínima puede solucionar el control a centros aislados en ETECSA.

Referencias

[1] A. C. Caputo, *Digital Video Surveillance and Security,* 2ª ed., Amsterdam: Butterworth-Heinemann, 2014.

[2] A. Caicedo, *Arduino para Principiantes*, 2ª ed., Carolina del sur, EE. UU.: CreateSpace Independent Publishing Platform, 2017.

[3] A. Herbawi, M. Teeti y S. Hmeed, *Raspberry Pi and Old Personal Computer's Based Face Detection and Recognition System*, trabajo de grado, Facultad de Ingeniería, Universidad Politécnica de Palestina, Hebrón, Palestina, 2017. [Internet]. Disponible en *https://scholar.ppu.edu/handle/123456789/6277*

[4] A. Marina. (2020, nov. 6). "Reconocimiento facial: la tecnología para identificación de personas". [Internet]. Disponible en https://protecciondatos-lopd.com/empresas/reconocimiento-facial/

[5] A. Vera Jiménez y E. F. Ortega España, *Diseño e implementación de un sistema de seguridad basado en reconocimiento facial, utilizando herramientas open source y monitoreo de imágenes por medio de una aplicación móvil con almacenamiento de la información dentro de una base de datos*, trabajo de grado, Facultad de Ciencias Matemáticas y Físicas, Universidad de Guayaquil, Ecuador, 2020. [Internet]. Disponible en http://repositorio.ug.edu.ec/handle/redug/48779

[6] Barra Espaciadora. (2013, dic. 31). "Miniordenadores ¿Qué son y para qué sirven?" [Internet]. Disponible en https://bcninfoservices.wordpress.com/2013/04/25/miniordenadores-que-son-y-para-que-sirven/

[7] B. G. Abril Sarmiento y P. X. Cuzco Arévalo, *Implementación de un sistema de video vigilancia remoto para hogares, utilizando herramientas de software libre*, Universidad Politécnica Salesiana, Cuenca, Ecuador, 2019. [Internet]. Disponible en https://dspace.ups.edu.ec/handle/123456789/17311

[8] F. J. García Mata, *Videovigilancia: CCTV usando vídeos IP*, Buenos Aires: Editorial Vértice, 2010.

[9] G. F. Suárez-Pesántez y J. K. Vizñay-Durán, "Desarrollo de un sistema de alarma domiciliaria con reconocimiento facial y alerta temprana. Caso de estudio: vivienda del Barrio Corazón de María, Cantón Cuenca, Provincia del Azuay", *Polo del Conocimiento*, vol. 6, n.° 7, pp. 935-956, 2021. [Internet]. Disponible en https://polodelconocimiento.com/ojs/index.php/es/article/view/2900

[10] L. N. Solis Calvopiña y L. Puga Torres, *Control de seguridad biométrico de reconocimiento facial como caso de estudio implementación en el área administrativa de la facultad de ingeniería de la Universidad Católica de Santiago de Guayaquil*, trabajo de grado, Facultad de Ingeniería, Universidad Católica de Santiago de Guayaquil, Ecuador, 2016. [Internet]. Disponible en http://repositorio.ucsg.edu.ec/handle/3317/6737

[11] D. Castaño Saavedra y J. D. Alonso Sierra, *Sistema de reconocimiento facial para control de acceso a viviendas*, trabajo de grado, Facultad de Ingenierías, Universidad Católica de Colombia, Bogotá, 2019. [Internet]. Disponible en https://repository.ucatolica.edu.co/handle/10983/24032

[12] I. González, J. González y F. Gómez-Arribas, "Hardware libre: clasificación y desarrollo de hardware reconfigurable en entornos GNU/Linux", en *VI Congreso de Hispalinux*, Madrid, sep. 2003. [Internet]. Disponible en http://www.iearobotics.com/personal/juan/publicaciones/art4/pres-hardware-libre.pdf

[13] J. R. V. Alfonso, "Aplicación del software libre y hardware de bajo costo en la videovigilancia", *Revista Telemática*, vol. 19, n.° 1, pp. 73-85, 2020. [Internet]. Disponible en https://redib.org/Record/oai_articulo2656702-aplicaci%C3%B3n-del-software-libre-y-hardware-de-bajo-costo-en-la-videovigilancia

[14] T. Gualotuña, E. Macias, A. Suarez, R. Fonseca, y A. Rivadeneira, "Mitigando efectos adversos de interrupciones del servicio de video-vigilancia del hogar en clientes WiFi inalámbricos", en *JITEL 2017 XIII Jornadas de Ingeniería telemática* (JITEL 2017), Valencia, España, sep. 2017. 10.4995/JITEL2017.2017.6591

[15] M. A. Gutiérrez Moreira, *Comunicación digital en la vigilancia de tiempo real basado en software libre para la escuela de educación básica del recinto Pueblo Nuevo del Cantón Isidro Ayora de la provincia del Guayas*, trabajo de grado, Facultad de Ciencias Matemáticas y Físicas, Universidad de Guayaquil, Ecuador, 2018. [Internet]. Disponible en http://repositorio.ug.edu.ec/handle/redug/27896

[16] N. Goilav y L. Geoffrey, *Arduino Aprender a desarrollar para crear objetos inteligentes*, Ediciones ENI, 2016.

[17] *ESP32 Series Datasheet*, Espressif Systems, Shanghai, China, pp. 20-50, 2019. [Internet]. Disponible en https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf

[18] R. Marín. (2020, dic. 2). ¿Qué es OpenCV? Instalación en Python y ejemplos básicos, *Revista Digital INESEM*. https://www.inesem.es/revistadigital/informatica-y-tics/opencv/

[19] Espressif Systems. (2022, en. 24). "Esp-dl". [Internet]. Disponible en https://github.com/espressif/esp-dl

[20] Arduino-ESP32 Espressif. "Installing". [Internet]. Disponible en https://docs.espressif.com/projects/arduino-esp32/en/latest/installing.html

[21] M. Sparks. (2018, mar. 18). "Arduino-ds1302". [Internet]. Disponible en https://github.com/msparks/arduino-ds1302

[22] A. Koritnik. (2021, oct. 19). "Alexa-Face-Recognition-with-ESP32CAM". [Internet]. Disponible en https://github.com/AK-Homberger/Alexa-Face-Recognition-with-ESP32CAM

[23] G. Sampallo. (2020, jun. 25). "esp32cam-python". [Internet]. Disponible en https://github.com/gsampallo/esp32cam-python