UNIVERSIDAD TECNOLÓGICA ISRAEL

ESCUELA DE POSGRADOS "ESPOG"

MAESTRÍA EN
ELECTRÓNICA Y AUTOMATIZACIÓN
Resolución: RPC-SO-09-No.265-2021

PROYECTO DE TITULACIÓN EN OPCIÓN AL GRADO DE MAGÍSTER

Título del proyecto:
SISTEMA DE CONTROL DE ACCESO POR RECONOCIMIENTO FACIAL Y COMANDO DE VOZ EN PYTHON
Línea de Investigación:
Ciencias de la ingeniería aplicadas a la producción, sociedad y desarrollo sustentable
Campo amplio de conocimiento:
Ingeniería, industria y construcción
Autor/a:
Ing. Edisson Roberto Escobar Sailema
Tutor/a:
Mgs. René Ernesto CortiCortijo

Quito – Ecuador

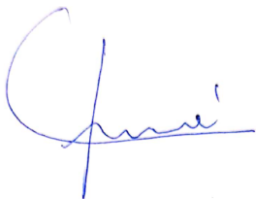
2022

APROBACIÓN DEL TUTOR

Yo, Mg. **René Ernesto Cortijo Leyva** con C.I: **1719010108,** en mi calidad de Tutor del proyecto de investigación titulado: **"Sistema de control de acceso por reconocimiento facial y comando de voz en Python"**.

Elaborado por: **Edisson Roberto Escobar Sailema**, de C.I: **1804262044**, estudiante de la Maestría: **Electrónica y Automatización**, de la **UNIVERSIDAD TECNOLÓGICA ISRAEL (UISRAEL)**, como parte de los requisitos sustanciales con fines de obtener el Título de Magister, me permito declarar que luego de haber orientado, analizado y revisado el trabajo de titulación, lo apruebo en todas sus partes.

Quito 13 de septiembre del 2022



_____

Firma

Tabla de contenidos

Índice de tablas

Índice de figuras

INFORMACIÓN GENERAL

Contextualización del tema

Hoy en día con un avance tecnológico en aumento y la información distribuida en la red, la seguridad es un punto esencial en el ámbito personal e industrial, como menciona Ramesh y Prasad (2019), por lo cual es una necesidad poseer sistemas de acceso que brinden la seguridad, confianza y se adapte a las limitaciones del usuario. Los sistemas de autenticación tradicional como el pin y contraseña pueden ser adquiridos mediante la observación encubierta directa, por tal motivo si un usuario no registrado obtiene estos datos lograría tener acceso a toda la información, por ende, un método que brinda una mayor seguridad es el uso de características biométricos, sean estas enfocadas a señales con rasgos fisiológicos como el rostro, el iris, las huellas dactilares, etc. o conductual como la voz, firma, etc., ambas son intransferibles e incluso únicas (Meng et al., 2015).

Pese al uso de métodos de reconocimiento unimodal, es decir al utilizar un solo patrón biométrico, continúa existiendo un bajo nivel de seguridad. Ahmed et al. (2020) señaló al crear su aplicación en Android, la identificación unimodal es deficiente pues se centra en un solo patrón biométrico, lo cual es solventado con el uso de la autentificación multimodal esto quiere decir que se utiliza más de un patrón. En su investigación pudo llegar a la conclusión que estos datos pueden combinarse para mejorar la forma de reconocimiento normal, es decir la base para un mejor sistema de acceso.

Además, es de gran interés contar con un sistema que posea una base de datos que interactúe de forma fácil, con la familia o grupo de trabajo. Estos deben contar con una interfaz amigable para que el usuario pueda asignar identidades. Debido a que la toma de datos biométricos es de manera confidencial se requiere que ésta sea capaz de interactuar con el usuario de manera que pueda solicitar su autorización antes de proceder con el registro. Igualmente, como menciona Pardo (2020), un proceso de identificación debe ser capaz de distinguir personas no registradas y anunciarlas de manera audible. Obteniendo de esta forma un sistema personalizado con un alto porcentaje de reconocimiento del individuo registrado, intuitivo al usuario y adaptable en cuanto al control de acceso biométrico se refiere (Muñoz, 2021).

Otro punto importante en la generación de sistemas de acceso es cubrir con las necesidades actuales de inclusión, es decir que puedan ser utilizados por un sector como aquellos con capacidades motoras reducidas, por lo cual se necesita una interfaz cuyo uso no solamente se limite al ámbito manual, sino controlado por una característica de uso general como lo es la voz

y de esta manera permite una interacción con el sistema de reconocimiento multimodal, lo cual es un enfoque en el diseño de dispositivos actuales como menciona Santander et al. (2020) y Martínez (2016).

Como punto adicional, se requiere que un dispositivo debe estar en constante evolución acoplándose a las tecnologías en surgimiento para no ser desechado prontamente, esto se logra con un mejoramiento continuo, dicha característica se manifiesta en aquellos elementos programados en software libre, con lenguajes de programación y librerías accesibles, para que de esta manera sea capaz de adicionar funcionalidades extras como lo expone Platero (2015) y Juárez (2018) y ser actualizados fácilmente.

El sistema de acceso enfocado al uso de la autentificación multimodal implementado en una plataforma libre, llegaría a beneficiar al usuario en general, partiendo desde el ciudadano común el cual mejoraría su seguridad personal y familiar sea esto en su hogar limitando el acceso a su entorno familiar y amigos, o en su ámbito laboral pues su versatilidad amplia su uso al resguardo de bienes físicos tanto como digitales. En la industria brindaría protección a los activos de la empresa sean estos equipos o herramientas, restricción de acceso a áreas clasificadas o talleres especializados en los cuales se localiza equipo sofisticado, insumos o materia prima necesaria para el proceso, brindando de esta manera un enfoque directo hacia el fortalecimiento de la seguridad. En el campo de la medicina restringiría el acceso de personal no autorizado a expedientes de pacientes, los cuales siempre se han considerado elementos confidenciales, junto con la protección de equipos y elementos biológicamente peligrosos, estos últimos son una constante preocupación pues su acceso inequívoco podría desencadenar un riesgo a la salud de las personas. En fin, su campo de aplicación es extenso, pero siempre convergiendo hacia una seguridad ampliada y fortalecida.

Problema de investigación

A nivel mundial la problemática en seguridad abarca una gran parte de países, con zonas que resaltan por su incremento delincuencia con índices de criminalidad altos y de seguridad bajos como lo es Latinoamérica y África en las cuales persevera la delincuencia (Adamovik , 2022).

En este contexto, como se menciona en Buvinik et al. (2005), existe un incremento en el número de actos vandálicos en los últimos años, sean estos violentos, no violentos, direccionados al individuo o la propiedad privada, incluso un robo o hurto puede desembocar en una reacción violenta por parte de los atracadores, esto afecta tanto a la propiedad privada como a la integridad física y mental de las personas, como el estudio concluye, los actos

vandálicos forman un barrera para el progreso y bienestar de la población por lo cual la seguridad en la actualidad es un punto esencial no solo a nivel personal sino global.

En 2022, INEC señaló que Ecuador presenta un incremento en la categoría de robo a domicilios, con valores según el número de reportes de 3.198 entre enero y mayo del 2021 a 3.413 entre enero y mayo del 2022, tomando en cuenta los casos registrados hasta finalizar el año 2021 que fueron 8176 casos reportados, de continuar esta tendencia se prevé un incremento de casos conforme avanza el año en curso, en vista de estos antecedentes se evidencia una inseguridad en aumento en varias áreas no solo centrándose en la domiciliaria, por lo cual se ve la necesidad de fortalecer las medidas de seguridad actual.

Por tal razón se ve necesario fortalecer los sistemas de seguridad domiciliarios, específicamente los de control de acceso, ya que estos son una barrera que protege al usuario contra intrusiones no autorizadas. Los sistemas de acceso tradicionales cumplen parte de esta función, pero debido a que la delincuencia va mejorando sus herramientas estos sistemas no pueden quedar rezagados, deben ir a la par con la evolución tecnológica, menguando la inseguridad conforme esta se incremente, insertando en sí las herramientas de análisis moderno y actualizando continuamente sus plataformas.

Los sistemas de control de acceso son parte de un blindaje en la seguridad, complementándose con la tecnología actual como las redes neuronales, machine learning, deep learning entre otros, pueden llegar a mejorar los sistemas tradicionales, cubriendo deficiencias como las analizadas en Oloyede y Hancke (2016), en el mismo se demuestra que una combinación de factores biométricos, es decir sistemas multimodales, conllevan a un fortalecimiento significativo en la seguridad público-privada, resguardando de esta forma el entorno personal y laboral a un incremento delincuencial pronosticado.

Objetivo general

Diseñar un sistema de control de acceso por reconocimiento facial y comando de voz en Python.

Objetivos específicos

- Contextualizar los fundamentos teóricos necesarios para la implementación del reconocimiento facial y vocal.
- Determinar los dispositivos, lenguaje y librerías necesarias para la elaboración del programa en un software libre, tomando en cuenta los requerimientos del sistema y su posible optimización.
- Elaborar un programa integrado que vincule el reconocimiento facial y la

disparidad entre los patrones de voz.

- Realizar una interfaz entre el sistema y el usuario para lograr una interacción mediante comandos de voz.
- Validar la mejora obtenida al complementar el sistema de reconocimiento facial con el de voz, al igual que el uso de su interfaz.

Vinculación con la sociedad y beneficiarios directos:

Este proyecto al ser direccionado a la seguridad y con índices de criminalidad en aumento, brinda un enfoque directo al bienestar individual y colectivo de órganos tanto públicos como privados y como se mencionó anteriormente una mayor seguridad individual conlleva a un progreso y bienestar de la población.

De igual manera representa un aporte social, debido a que vincula a la sociedad con la tecnología actual, brindando seguridad a la población y la industria, actualizándose en el uso de sistemas inteligentes, mediante la creación de un sistema tecnológico capaz de interactuar con el usuario y reconocerlo, mediante un sistema multimodal basado en patrones biométricos como el rostro y la voz, situando al individuo en un ambiente tecnológico moderno.

Al ser diseñado sobre una plataforma libre aumenta el tiempo de vida del sistema ya que puede ser adaptado a nuevas tecnologías en desarrollo, este enfoque va direccionado a la actualización de sistemas convirtiéndolos en productos reutilizables, dado que no se los desechará sino se los utilizaría pues el hardware contará con la capacidad de ser multiusos para ampliar su vida útil y el software contará con la cualidad de ser editable.

El proyecto al estar direccionado al uso de software libre brinda la capacidad de publicar los scripts de manera libre y en diversas plataformas como Github, para que la lógica utilizada pueda servir como base para futuros proyectos. Como menciona Motta y Rogers (2002) en su análisis sobre el uso de software Open Source, estos dan la oportunidad para que ingeniero o estudiante pueda realizar estudios y avances con total acceso a la publicación y lectura de proyectos relacionados, debido a su amplia apertura a las librerías y líneas de código.

Al realizarse en base a un punto de partida moderno, se ve necesario la concatenación con sistemas tecnológicos actuales, como lo es el control e interacción por voz, esto brindaría al sistema la característica de inclusividad, dado que permitiría el registro y reconocimiento de aquellas personas que no cuenten con las capacidades motoras necesarias para interactuar manualmente con sistemas tradicionales de acceso.

Finalmente, los beneficiarios directos serían los ciudadanos brindándoles la capacidad de proteger sus hogares y el empresario contando con un sistema de fácil uso que le permitirá

proteger las diferentes áreas de su empresa y la flexibilidad para adicionar un gran número de usuarios.

CAPÍTULO I: DESCRIPCIÓN DEL PROYECTO

Contextualización general del estado del arte

El proyecto en curso se enfoca en brindar un sistema de acceso multimodal seguro, mediante dos patrones biométricos en este caso el facial y la voz, para ello es necesario explorar los proyectos precedentes relacionados con este tema, analizando sus fortalezas y debilidades, tomando en cuenta aquellos métodos y técnicas que puedan ser enfocados hacia un sistema libre, de esa forma plasmarlo en un ambiente de programación basado en Python. A continuación, se resumen los siguientes proyectos.

En 2019, Ramesh y Prasad fusionaron el rostro y la voz en un sistema biométrico de reconocimiento multimodal para probar la eficacia de esta combinación sobre los sistemas unimodales, para ello utilizaron métodos como el de HOG (Histograma Orientado a Gradientes), usado para el análisis facial y la FFT (Transformada Rápida de Fourier) complementada con el modelo GMM (Modelo de Mezcla Gaussiana) para el análisis de voz.

El experimento tomó 1000 muestras faciales y de voz para dar como resultado una mejora al reemplazar sistemas unimodales por multimodales con los patrones biométricos del rostro y la voz como base. En este experimento resalta una mejora en la TAR (Tasa de Aceptación Verdadera), la cual se enfoca en cuántos rostros reconoce inmediata y correctamente, demostrando un aumento en el índice de reconocimiento sobre cada biometría de forma independiente.

Existe una característica que cabe mencionar que es la fusión, dicho en otras palabras, la forma en que las señales de voz y facial se concatenan, para este caso en particular de sistemas multimodales se utiliza la fusión por nivel de puntuación o niveles de confianza, en el cual se crea un vector con los puntajes mínimos y máximos resultado de las señales para dar como resultado un vector de coincidencia específico. Los resultados de este trabajo demostraron que al combinar rasgos faciales y de voz generan un incremento de la TAR, disminución de la FAR (Tasa de aceptación falsa) y FRR (Tasa de rechazo falso) en comparación al uso de estos de forma independiente, concluyendo que el modelo de reconocimiento multimodal genera una alta seguridad y confidencialidad tomando en cuenta los métodos antes mencionados.

De este proyecto se concluye un buen resultado en la concatenación de la voz y el rostro para el sistema multimodal, además resalta el uso de la FFT con GMM para la voz y HOG para el rostro.

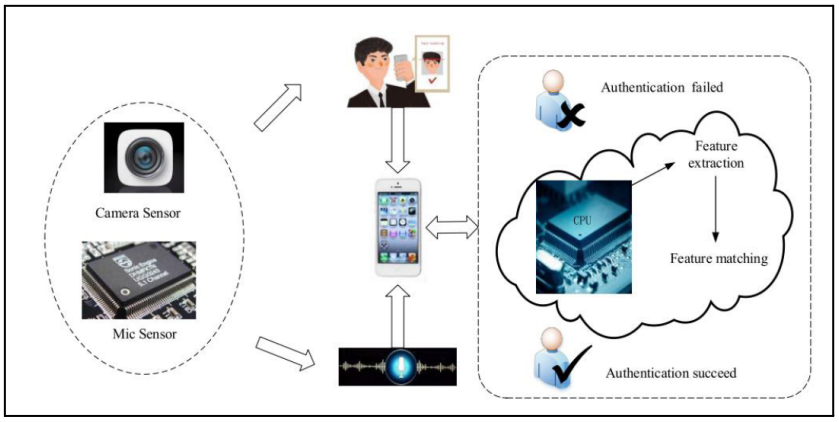
En 2020, Zhang et al. en su proyecto “An Efficient Android-Based Multimodal Biometric Authentication System With Face and Voice”, crean una aplicación basada en Android y programada en Java en la cual combinan el uso de la biometría facial con la voz, comprobando una mejora significativa al integrar dos sistemas de reconocimiento de patrones biométricos, incrementando la precisión y eficacia en cuanto a la autentificación del usuario, dando prioridad a la optimización de la RAM, CPU y GPU.

La ligereza en el programa se dio mediante el uso de análisis y métodos con poca carga al sistema, esto es esencial al momento de plasmarlo en una plataforma libre por su baja capacidad de procesamiento y almacenamiento de datos, por lo cual es necesario el uso de algoritmos de análisis profundo, pero con una baja exigencia de la RAM.

De este trabajo existen puntos importantes a mencionar que son; el modo de análisis, es decir como datos biométricos se ejecutarán puede ser en serie, paralelo o modo jerárquico, al igual que los niveles de fusión de resultados, como la fusión a nivel del sensor, de la función, del puntaje de coincidencia, del rango y de la decisión, en la figura 1 se puede observar el principio de funcionamiento de la aplicación.

Figura 1

Aplicación multimodal en Android



Nota. Tomado de An Efficient Android-Based Multimodal Biometric Authentication System with Face and Voice (p. 2), por X. Zhang, D. Cheng, P. Jia, Y. Dai y X. Xu, 2020, IEEE, 8(1).

El estudio resalta la base de datos XJTU, donde se observa un mejor índice de reconocimiento facial con el uso de Haarcascades y algoritmo de Adaboost que con el de HOG. En cuanto al reconocimiento de voz recomienda el uso del método VAD (Detección de Actividad de Voz) que fue mejorado para una mejor discriminación del ruido y un análisis de datos GMM.

Como conclusión de este análisis se toma en cuenta las recomendaciones como el uso del método LBP para reconocimiento facial, ya que este resalta por su insensibilidad a la luz y

variaciones de grises. En cuanto al análisis de voz, ejecutarlo con principios de GMM para disminuir el índice (FAR) y filtros para el ruido. Finalmente se evidencia una desventaja ya que dependerá de la calidad de la cámara y del micrófono del dispositivo donde se instale la aplicación, pues de ello depende la adquisición de las señales y además el sistema cuenta con un control totalmente manual.

En 2021, Moreno et al. elabora un sistema que combina las biometrías como la EEG(Electroencefalograma), la voz y el rostro, generando una base de datos para la comparación de las mismas, comparando los mejores índices de reconocimiento.

El protocolo de adquisición se basa en capturar señales de video, voz y EEG mientras se pronuncia una secuencia de dígitos al azar emitidos por el CPU, para la adquisición de las señales EEG se utilizó un dispositivo inalámbrico EmotivTM Epoc, para la voz un micrófono dinámico cardioide SennheiserTM MD421-II en conjunto con una mezcladora de audio Yamaha TM MG06X y una cámara web todo colocado en una cabina anecoica, el sistema es controlado mediante un Script de Matlab, analizado mediante modelos DL (Deep Learning) y CNN (Redes Neuronales Convolucionales).

Los resultados del sistema en la fusión cara y voz fueron una precisión media de 99.51 %, una desviación estándar de 0.69 en 100 intentos con una EER de 1.08 % y una desviación estándar de 0.19 en 10 ensayos realizados. En cuanto al sistema de fusión de voz y EEG se obtuvo una mejora en la precisión de 0.931 y 0.942 de forma individual a 0.977 combinados, esto se realizó con el uso de redes neuronales para entrenar el modelo de reconocimiento.

Como punto importante de este estudio resalta la necesidad de un sistema de supresión de ruido para una mejor adquisición de voz, como desventaja el micrófono necesita una mezcladora adicional para el sistema y el uso de CNN con DL mejoran significativamente el modelo de entrenamiento, lo cual da pautas para la selección del hardware.

Se omite el uso del EEG para el proyecto en curso por su dificultad en el posicionamiento de sensores pues esto no fue contemplado para un sistema de reconocimiento rápido.

Proceso investigativo metodológico

La investigación para el proyecto en curso se realiza con un enfoque cuantitativo el cual según Sampieri (2014), está caracterizado por la medición del fenómeno a investigar, que en este caso es la adquisición de señales biométricas y sus respectivos componentes, enfocados en sus magnitudes y características específicas, utiliza estadísticas como los resultados obtenidos en proyectos de comparación multimodal, que contrastan varios sistemas de análisis y su validez en cuestiones de seguridad.

El proyecto es de tipo experimental ya que se manipula la variable independiente que en este caso son la biometría del rostro y la voz, descomponiéndose en sus características específicas, para determinar según su análisis cuál es el mejor carácter que convierte en única a esa variable, el ambiente en el cual la puedo obtener y la mejor forma en la cual estos se concatenar exitosamente.

Se realizará mediante técnicas de observación de datos y resultados ofrecidos por los diferentes métodos de análisis en cada parte del proceso de adquisición y procesamiento de las variables, para determinar mejoras posibles y debilidades del sistema.

Al estar enfocado hacia los ciudadanos y el fortalecimiento de la seguridad en la industria, para la validez del sistema multimodal y el uso de los patrones biométricos se tomará en consideración los estudios antes realizados con sus índices de confianza y seguridad, se considera una muestra de 6 participantes para validar las tasas como (TAR-FAR-FRR).

Se usarán metodologías cuantitativas como diseños experimentales para probar la eficacia de una lógica u otra, análisis comparativo de resultados para elegir el método más confiable y modelaciones matemáticas para el manejo de los datos numéricos generados con el uso de cada biometría.

CAPÍTULO II: PROPUESTA

Fundamentos teóricos aplicados

Este proyecto se enfocará en el diseño de un sistema de acceso multimodal combinando las biometrías del rostro y la voz para ejecutar las labores de reconocimiento, se elaborará en una plataforma libre con las capacidades suficientes para procesar las variables con las que se va a trabajar. La programación se efectuará en Python un lenguaje abierto con las librerías necesarias para el análisis de la información de cada patrón biométrico, como valor agregado el sistema de interacción y control se lo realizará mediante comandos de voz brindando un punto de partida acorde con la tecnología actual.

Este módulo se enfocará en los siguientes segmentos, los principios para adquisición de las biometrías, el procesamiento de datos, el modo de análisis, el método de decisión, adaptación del sistema a comando de voz y acondicionamiento del hardware.

Adquisición de biometría facial

En Python existe una librería con herramientas de análisis diversas para realizar el procesamiento de imágenes y video en tiempo real, esta herramienta se llama OpenCV, en el transcurso de este proyecto se usarán estas librerías como parte de los scripts.

OpenCV. Visión por computadora de código abierto o OpenCV por sus siglas en inglés, es una biblioteca gratuita y de libre acceso enfocada al procesamiento de imágenes y videos. La biblioteca que se usará es la basada en Python con módulos específicos para detección y reconocimiento de rostro y procesamiento de imágenes y video.

Para poder realizar el reconocimiento de un rostro es necesario pasar por las siguientes etapas:

Detección de rostro. En este paso se analiza si en la imagen se encuentra un rostro, para esto existen métodos como el HARR o LBP para detectarlo, cada una con sus respectivas características.

Haar cascade. Es un método de detección de objetos basado en machine learning, se enfoca en la concatenación de clasificadores los cuales analizan la imagen en diferentes fragmentos, como se observa en la Figura 2, la clasificación se realiza entre imágenes positivas (imágenes que queremos que se reconozcan) y negativas (imágenes que no queremos que se reconozcan) (Hasan y Sallow, 2021).

Figura 2

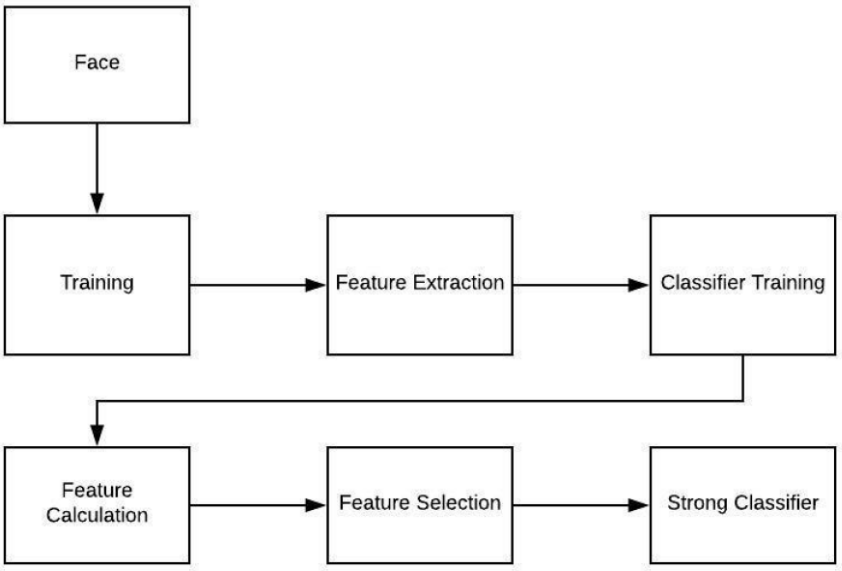
Clasificador de Haar cascade

Nota. Tomado de *Facial recognition using Haar cascade and LBP classifiers* (p.2), por K. Kushsairy, K. Mohd, N. Haidawati, S. Sairul, B. Zulkifli, 2014, *International Conference on Engineering Technology and Technopreneuship (ICE2T), 10*(1).

La secuencia de acciones que realiza este método se puede observar en la Figura 3, cada proceso cuenta con su propio criterio de ejecución.

Figura 3

Diagrama de flujo Haar cascade

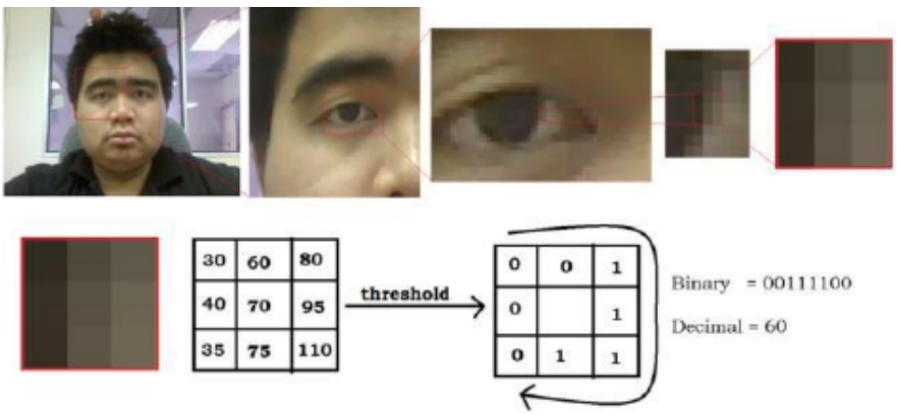


Nota. Tomado de *Face Detection and Recognition Using OpenCV* (p. 4), por R. Hasan y A. Sallow, 2021, Journal of Soft Computing and Data Mining*, 2*(1).

Patrones binarios locales. LBP o Patrones Binarios Locales, es un método de detección de rostros en el cual se realiza una comparación de píxeles cada uno con respecto a sus vecinos como se puede observar en la Figura 4, obtiene más información que las características Haar, es altamente discriminativo óptimo para exigentes tareas de reconocimiento (Merchán et al., 2016).

Figura 4

Cálculo de Patrones Binarios Locales



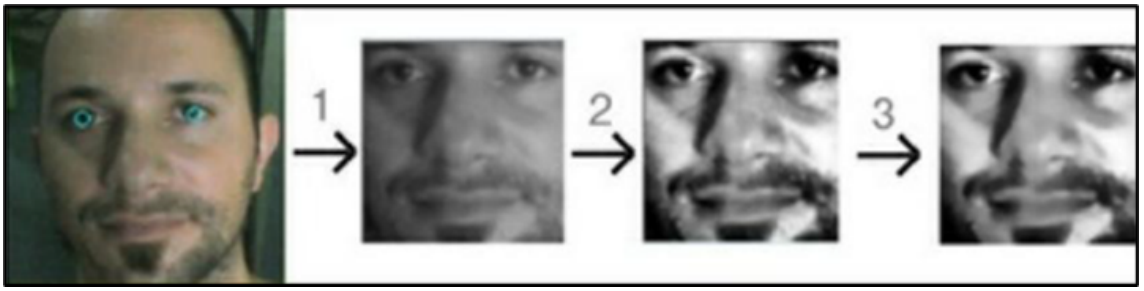
Nota. Tomado de *Facial recognition using Haar cascade and LBP classifiers* (p.3), por K. Kushsairy, K. Mohd, N. Haidawati, S. Sairul, B. Zulkifli, 2014, International Conference on Engineering Technology and Technopreneuship (ICE2T)*, 10*(1).

Procesamiento de la imagen detectada. Para poder contar con una imagen con características fácilmente diferenciables es necesario efectuar una serie de procedimientos como se observa en la Figura 5, caso contrario factores como la luz o la sombra podría afectar al reconocimiento, a continuación, se presentan los más relevantes:

- Transformación geométrica en la cual se reduce el tamaño, no cambia los valores si la captura de la imagen se realiza con una cámara web o con una de alta calidad de 5 Mpx, en este proceso se recortará, rotará y centrará el rostro.
- Ecualización de histogramas, este estandariza el brillo del lado izquierdo y derecho de manera independiente, para que la luz o sombras no lo afecten.
- Suavizado, reduce el ruido de la imagen.

Figura 5

Secuencia en procesamiento de imagen



Nota. Tomado de *Mastering OpenCV with Practical Computer Vision Projects* (p. 242), por D. Baggio, 2012, Packt Publishing Open Source.

Recolección de imágenes y aprendizaje de modelo. La recolección de imágenes se basa en crear una matriz con un número considerable de capturas para poder entrenar al reconocedor, debido a esto es necesario que cada imagen sea diferente a su predecesora, caso contrario no serviría tomar varias fotos si son iguales, para esto se recomienda tomarlas en diferentes ángulos y realizando varios gestos.

Posterior al almacenamiento de imágenes, es necesario entrenar al sistema para que reconozca los rostros con un algoritmo de aprendizaje automático. Entre los algoritmos más usados tenemos ANN (Redes Neuronales Artificiales), Eigenfaces (Análisis de componentes principales), Fisherfaces (Análisis discriminante lineal) o LBPH (Histograma de Patrones Binarios Locales), estos calculan el promedio matemático de las imágenes usadas en el entrenamiento, creando un grupo de vectores y valores propios, relacionados con los pixeles y características promedio cada imágen y así lograr una comparativa más rápida en el momento del reconocimiento facial.

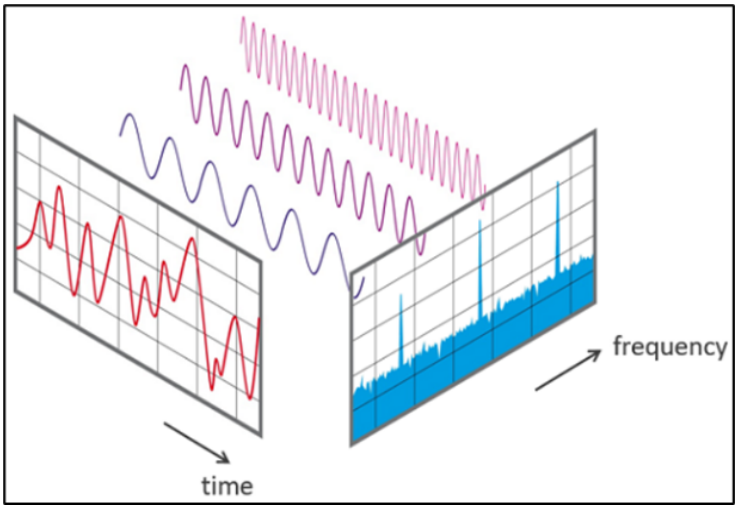
Reconocimiento de rostro. En este punto se realizará la comparación entre el rostro detectado con el del entrenamiento, se asignará un número de etiquetas en el entrenador que determinará a qué grupo de imágenes corresponde la imagen detectada, para esto se utiliza FaceRecognizer de OpenCV, devuelve una métrica de confianza con la función predict, es decir que tan confiable es el parentesco entre ambas imágenes del reconocimiento actual, el reconocimiento es más confiable mientras más bajo sea, un valor promedio oscila entre 70 a 100, a partir de este punto se le asignará el puntaje necesario para dar la seguridad que está reconociendo al usuario correcto (Baggio et al., 2012).

Adquisición de biometría de voz

Transformada rápida de Fourier (FFT). Este método es usado para transformar las señales de audio obtenidas del dominio del tiempo al de la frecuencia como se visualiza posteriormente en la Figura 6, se realiza separando la señal en sus componentes espectrales y así estos sean manejables. Mediante este método se puede analizar características específicas o un patrón en el grupo de datos (Singh y Khan, 2015).

Figura 6

Muestra de voz en el dominio del tiempo y junto a su FFT en la frecuencia



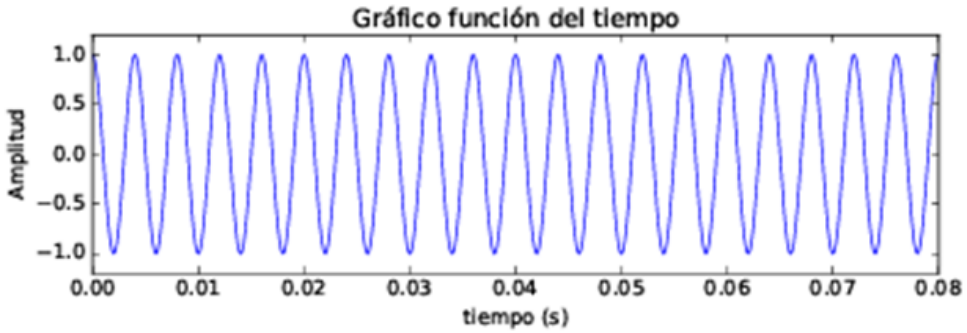
Nota. Tomado de *Transformación rápida de Fourier FFT - Conceptos básicos* [1], por NTi Audio, 2022, Nti-audio (https://www.nti-audio.com/es/servicio/conocimientos/transformacion-rapida-de-fourier-fft#:~:text=La%20%22Transformaci%C3%B3n%20r%C3%A1pida%20de%20Fourier,proporciona%20informaci%C3%B3n%20sobre%20su%20composici%C3%B3n).

Este método usado en Python cuenta con limitaciones, al ser utilizado con un número grande de muestras su velocidad de procesamiento disminuye, por tal motivo es recomendable

su uso con una muestra corta de audio, no se implementa para un análisis en tiempo real pues trabaja con una muestra (Spilbury y Euceda, 2017). En la Figura 7 se puede observar un fragmento de una señal simulada en el dominio del tiempo.

Figura 7

Gráfico en función del tiempo con la función Acos(2πf0t)

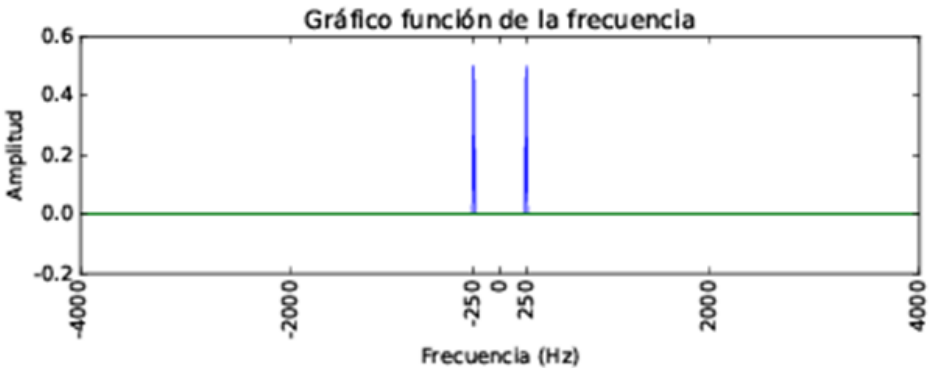


Nota. Tomado de *Transformada Rápida de Fourier utilizando Python* (p. 3), por M. Spilbury y A. Euceda, 2017, Revisita de la Escuela de Física UNAH, 5(1).

Al realizar la FFT se obtiene la señal antes adquirida ahora en el dominio de la frecuencia, como se observa en la Figura 8 genera una señal simétrica en el eje x correspondiente a la frecuencia por lo cual se recomienda el uso de la mitad de la señal generada.

Figura 8

Transformación en función de la frecuencia con A=1 y f0=250Hz



Nota. Adaptado de *Transformada rápida de Fourier utilizando Python* (p. 3), por M. Spilbury y A. Euceda, 2017, Revisita de la Escuela de Física UNAH, 5(1).

Principios sistemas interactivo

La base en un sistema interactivo es la comunicación entre el sistema y el usuario, esto se da al momento de convertir nuestras órdenes o instrucciones al lenguaje de máquinas, en un

principio se obtiene mediante los conversores de texto a voz o viceversa y posteriormente en la lógica con la cual se use estas funciones.

Conversor de texto a voz. Como se mencionó en 2022, Afrin et al. existe un traductor en Google que puede transformar el texto en voz, este sistema cuenta con una versatilidad en idiomas y su programación basada en Python, con una gran gama de voces y acentos, la librería en Python se denomina GTTS es una de las más usadas, se puede visualizar un fragmento de código en la Figura 9 y los lenguajes que soporta en la Figura 10.

Figura 9

Programación de GTTS en Python

Nota. Tomado de *Language Convertidor Using Python* (p. 2), por N. Afrin, G. Aditi, K. Gopi y K. Rathan, 2022, International Research Journal of Modernization in Engineering Technology and Science*, 4*(1).

Figura 10

Lenguajes soportados por GTTS

Nota. Tomado de *Conversión de texto a voz TTS con python y GTTS* [1], por Parzibite, 2019, Parzibite (https://parzibyte.me/blog/2019/07/06/conversion-texto-voz-tts-python-gtts/).

Conversor de voz a texto. En este caso “Speech Recognition”, se basan en sistemas de reconocimiento de voz, estos sistemas tienen como principio el modelo oculto de Markov

(MMM), funciona bajo la premisa de que una señal de voz, cuando se ve en una escala de tiempo corta, puede asumirse como un proceso estacionario, es decir uno donde las propiedades estadísticas no cambian. De esta forma una señal de voz se divide en fragmentos y estos a su vez se los transforma en un conjunto de vectores, los grupos de vectores se emparejan con uno o más fonemas, es por ello que la traducción varía de persona a persona incluso dentro del mismo idioma tal como se explica en Amos (2020).

La instancia cuenta con 7 métodos para reconocer el audio de la fuente, para ellos utiliza las siguientes API como se observa en la Figura 11.

Figura 11

API de Speech Recognition

`recognize_bing()`: Microsoft Bing Speech
`recognize_google()`: Google Web Speech API
`recognize_google_cloud()`: Google Cloud Speech - requires installation of the google-cloud-speech package
`recognize_houndify()`: Houndify by SoundHound
`recognize_ibm()`: IBM Speech to Text
`recognize_sphinx()`: CMU Sphinx - requires installing PocketSphinx
`recognize_wit()`: Wit.ai

Nota. Tomado de The Ultimate Guide To Speech Recognition With Python (p. 1), por D. Amos, 2019.

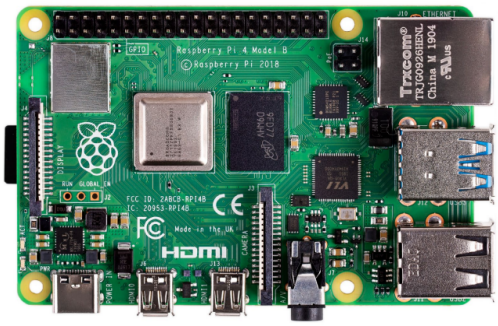
Sistemas de código abierto

Las plataformas open source se basan en su diseño de código abierto cuya ventaja radica en la libre edición de sus componentes tanto su hardware como el software.

Raspberry Pi 4B. De las diferentes plataformas existentes resalta la Raspberry Pi, una mini computadora con posibilidad de acceso remoto o mediante su escritorio, cuenta con un procesador ARM Cortex-172 con cuatro núcleos a 1.5 GHz, GPU (VideoCore VI), memoria ram de 4GB, puertos GPIO de 40 pines, 2 salidas micro HDMI, 2 puertos USB 2.0, 2 puertos USB 3.0, puerto Micro SD, un conector audio jack, alimentación mediante USB-C. Además, puede anexar en sí una pantalla táctil de 3.5 in, con bluetooth 5.0 y Wifi 802.11as, esto implementado en una tarjeta compacta como se puede observar en la Figura 12 (Raspberry PI, 2021).

Figura 12

Placa Raspberry PI 4 B con 4 de RAM



Nota. Tomado de *Raspberry Pi 4 Computer Model 4B* [2], por Raspberry Pi Trading Ltd, 2021, Raspberrypi (https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-product-brief.pdf).

No cuenta con un sistema de protección contra sobre corrientes o pérdida de energía por lo cual el uso de un UPS es necesario.

Permite la instalación de varios sistemas operativos de los cuales el más común es Raspbian basado en Ubuntu, compatible con software libre como Python el cual cuenta con librerías que permiten el acceso a los recursos físicos de la plataforma y preinstalados editores de código para facilitar la creación de scripts.

UPS Plus SKU. Un UPS diseñado para plataformas Raspberry PI, con tecnología de monitoreo de corriente/voltaje en la fuente de alimentación y las baterías, en conjunto con baterías 18650 brinda una autonomía energética del sistema. Su sistema compacto se acopla a la plataforma optimizando el espacio como se observa en la Figura 13, La Figura 14 muestra los leds indicadores del estado de la carga y porcentaje restante de la batería en uso (52PI Wiki, 2022).

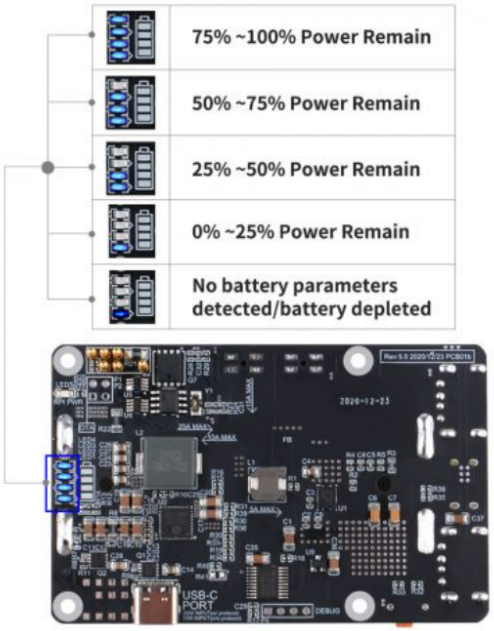
Figura 13

Acoplamiento de Raspberry Pi y UPS

Nota. Tomado de *EP-0136,* por 52Pi Wiki, 2022, Wiki52pi (https://wiki.52pi.com/index.php?title=EP-0136&oldid=12046).

Figura 14

Indicadores de carga



Nota. Tomado de *EP-0136,* por 52Pi Wiki, 2022, Wiki52pi (https://wiki.52pi.com/index.php?title=EP-0136&oldid=12046).

Descripción de la propuesta

Se basará en la elaboración de un sistema de control de acceso por reconocimiento facial y comandos de voz elaborado en Python, dicho sistema va a contar con tres partes para la adquisición de señales biométricas, la fusión de las mismas, el diseño del sistema interactivo y el armado del hardware.

Estructura general

El proyecto contará con un sistema de adquisición de biometrías facial y de voz, cada uno con sus respectivos dispositivos de lectura y un elemento de salida, todo conectado a una plataforma Raspberry PI 4 B, se puede visualizar en la Figura 15. Además, se puede observar el diagrama de flujo en la Figura 16 y 17.

Figura 15

Diagrama general de funcionamiento

Nota: El sistema cuenta con 2 dispositivos para el ingreso de datos una cámara web para la biometría facial, un micrófono cardioide para la voz y un dispositivo de salida en este caso un parlante, todo procesado por una Raspberry PI 4 B.

Figura 16

Diagrama de flujo parte 1

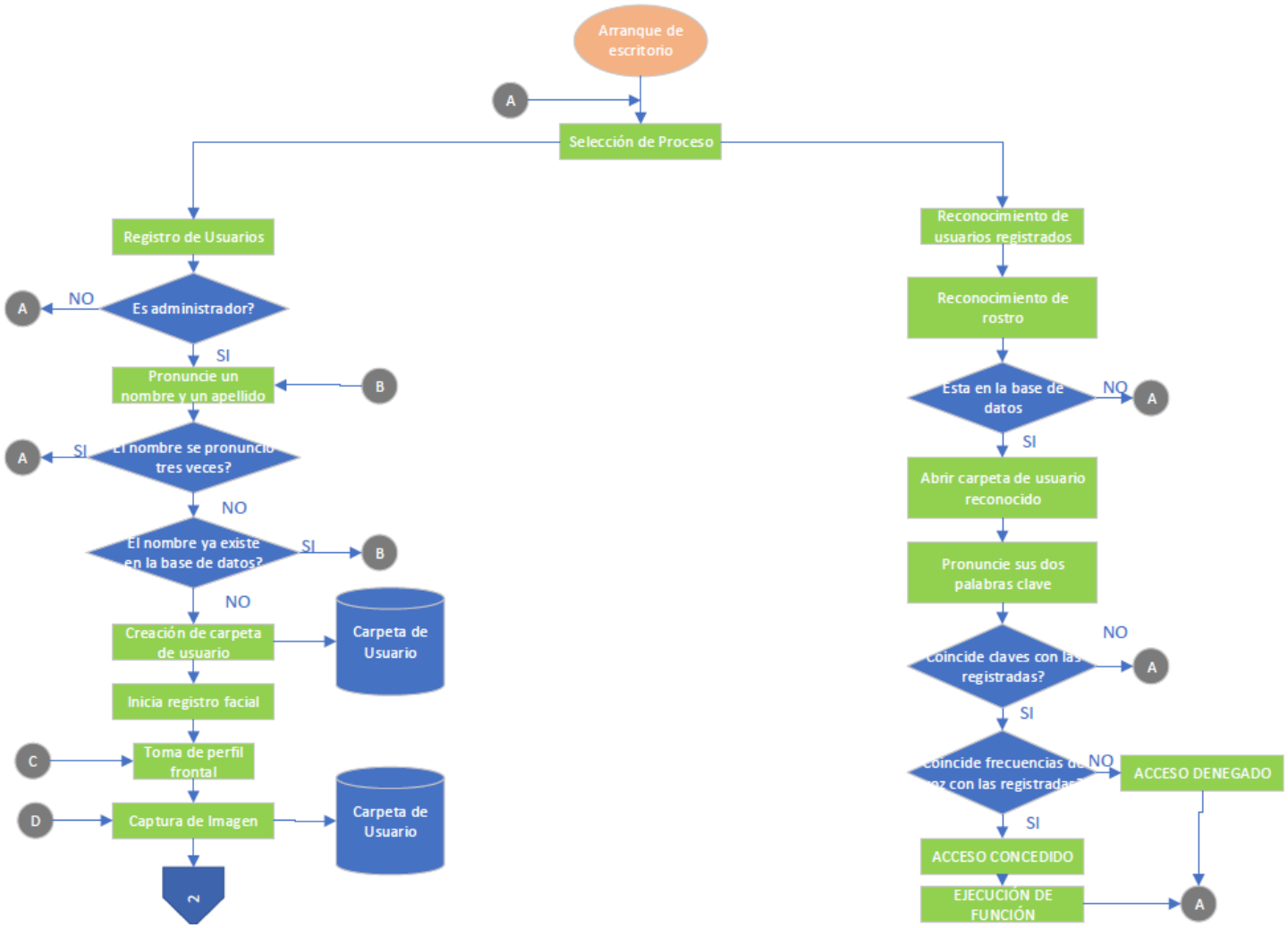
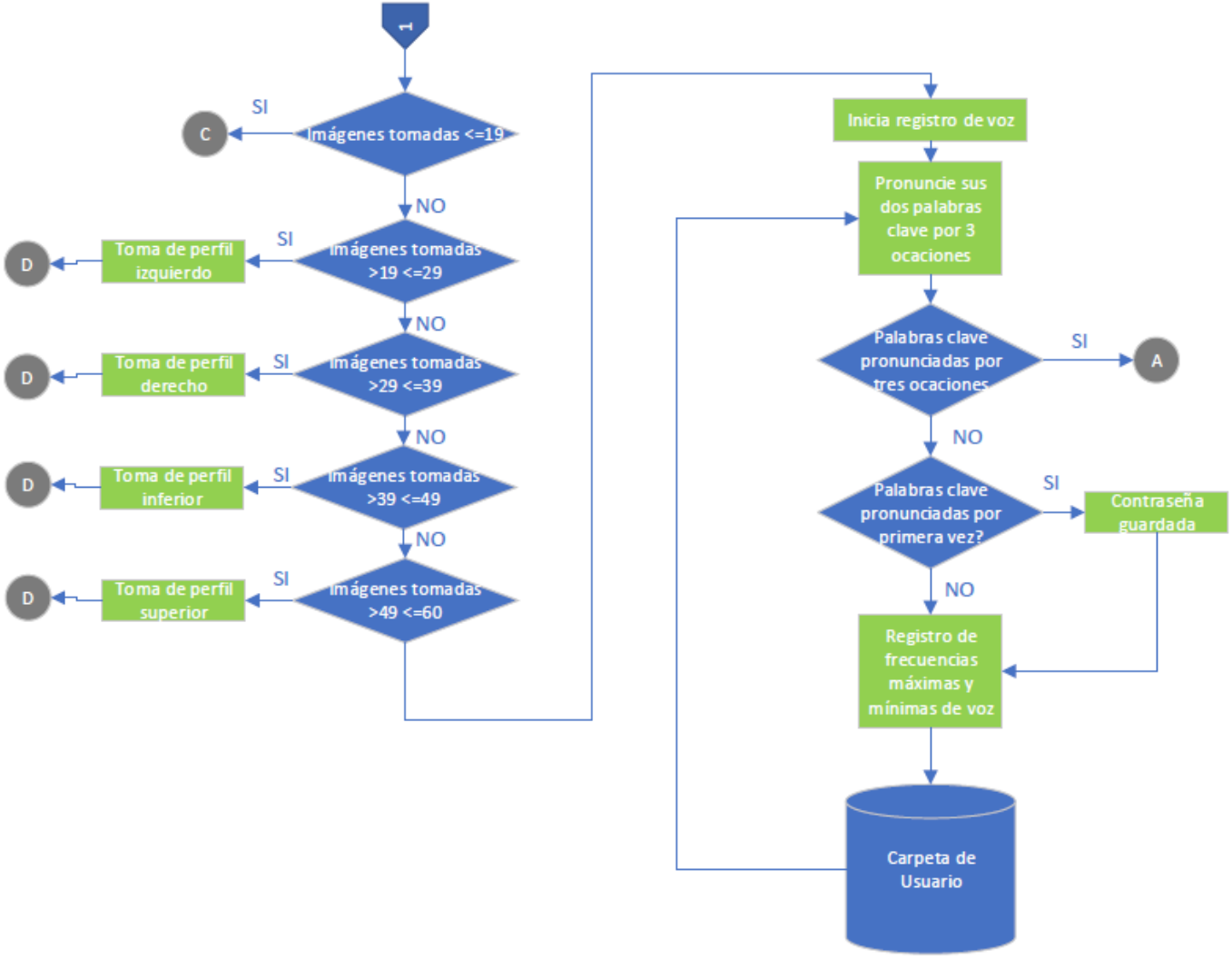


Figura 17

Diagrama de flujo parte 2



Explicación del aporte

El sistema realizará la adquisición de las biometrías facial y de voz, con su respectivo análisis ambos aspectos serán efectuados en una Raspberry Pi 4B, el programa se ensamblará en Python, el proceso contará con 2 modalidades, de registro y reconocimiento, ambas solicitadas por comando de voz.

Acondicionamiento del hardware. Para poder desarrollar el proyecto en curso será necesario contar con el hardware necesario para la ejecución del programa.

Instalación del sistema operativo. Se procederá con la instalación de un sistema operativo de 64 bits “Raspberry Pi OS with desktop” en una Raspberry PI 4 B de 4 GB de RAM y una memoria de 252 GB.

Instalación de accesorios para Raspberry PI. Se instalará vía cable USB el micrófono externo con capacidad de captura del sonido en modo estéreo y filtro de ruido, configurado con una ganancia media, esto debido a que una ganancia alta produce un aumento en la sensibilidad, lo cual conlleva a una amplificación del ruido residual, en contraparte una ganancia baja causa poca recepción del audio. Para la captura de imágenes se usará una cámara web N8 HD de 1080P, se adicionará una pantalla LCD touch de 3.5 pulgadas diseñada para la Raspberry Pi, esto para la visualización del escritorio e imágenes de aviso.

Se implementará un UPS Plus SKU, complementado con unas baterías 18650 de litio, brindando una autonomía de una hora aproximadamente, lo cual es esencial para un sistema de acceso, su diseño permite la optimización del espacio como se ve en la Figura 18.

Figura 18

Instalación de UPS Plus SKU

Diseño de una estructura para el sistema. Para poder acoplar los elementos externos es necesario diseñar una estructura la cual es mostrada en la Figura 19, con una base para colocar

la cámara, adicionar ventiladores para el enfriamiento del procesador y acceso a los puertos mini HDMI, carga, apagado del UPS y Raspberry PI como se observa en la Figura 20.

Figura 19

Diseño de estructura para el acoplamiento de los dispositivos

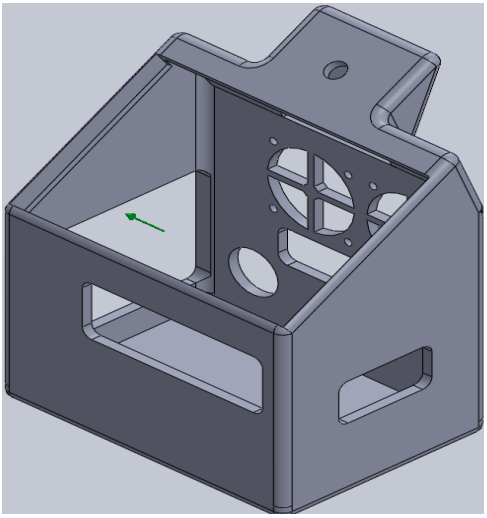
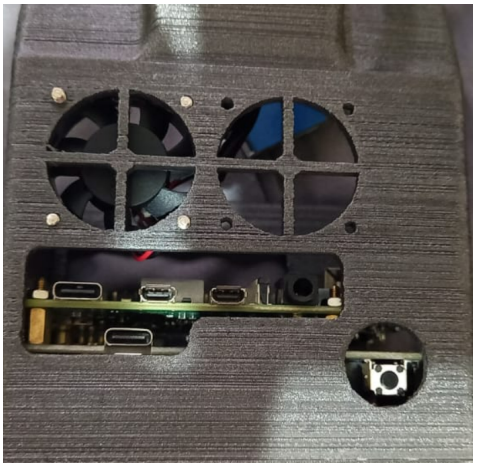


Figura 20

Estructura para acoplamiento de dispositivos



Registro de patrón biométrico facial. Para el registro facial se usa la librería Haar Cascade de OpenCV, esto para agilizar la adquisición del registro de imágenes pues su evaluación se realiza de forma rápida, en este punto no se necesita de un análisis profundo, solamente un identificador de rostros para la captura de la imagen y posterior conversión a escala de grises, formando una base de datos del sistema de forma ágil y bajo almacenamiento, permitiendo mayor cantidad de usuarios registrados.

Registro de patrón biométrico de la voz. El registro de un sistema multimodal con el uso de las biometrías facial y de voz brindan un fortalecimiento en la seguridad, esto se lograría

mediante el uso de un modo de autorizaciones lineales es decir solo si accede al primer reconocimiento llegaría a tener pasó al segundo y de la misma forma al tercero, contando con tres filtros previo a la autorización final.

Implementación de filtros de ruido para limpieza de la señal. En el caso de la adquisición de la biometría de voz se procurará un enfoque direccionado a mejorar la calidad de la adquisición del patrón, tomando en cuenta la supresión del ruido ambiental que causa problemas en la lectura correcta de la voz, para esto se procederá con el uso de filtros de ruido tanto físicos como digitales, mediante la combinación de estos dos filtros se obtendrá una muestra con mayor claridad para el análisis, permitiendo mayor seguridad al momento de generar la base de datos.

Incremento en la adquisición de datos. Un punto de suma importancia será la adquisición de los datos, para contar con una muestra significativa de cada usuario se usará un micrófono con una transmisión de datos de 24 bits, esto permitirá contar con una mayor cantidad de información, logrando obtener más versatilidad de frecuencias predominantes, brindando una base de datos exacta en cuanto a sus frecuencias más significativas.

Espacio de almacenamiento. Como punto esencial para disminuir el espacio ocupado en el dispositivo se almacenarán solamente los datos generados por el algoritmo, es decir los más significativos en formato de texto. Al igual que los parámetros de frecuencia significativa, se almacenarán las dos claves generadas por el usuario como una etapa adicional de verificación, mejorando la seguridad ya que de un mismo padrón biométrico se obtienen dos datos para la verificación.

Reconocimiento de patrón biométrico facial. Para esto se usará el método LBP con un sistema de entrenamiento del mismo tipo para acelerar el reconocimiento, el script de entrenamiento se ejecutará cada vez que exista un nuevo registro para que el modelo siempre se encuentre actualizado, brindando agilidad al reconocimiento de usuarios registrados.

Reconocimiento de patrón biométrico de la voz. El sistema tomará una muestra de voz cuando el usuario pronuncie sus contraseñas y si previamente se reconoció su biometría facial, esto brindará un paquete de datos similar al registrado. El paquete de datos brinda dos seguridades, el reconocimiento de la clave pronunciada y del patrón, permitiendo acceder al reconocimiento biométrico del patrón si la clave pronunciada fue la correcta fortaleciendo la seguridad en el sistema en este punto de análisis.

Interacción con el sistema. Para esto se implementará un script que permitirá la interacción por comando de voz tanto para el registro como el reconocimiento, de igual manera para fortalecer la seguridad el registro de usuarios solo se dará si es validado por un administrador para evitar que un usuario no deseado se registre.

Sistema de energía auxiliar o UPS. Ya que los sistemas de seguridad están diseñados para permitir o restringir el acceso, es necesario que estos cuenten con sistemas auxiliares de energía que permitan realizar su función en el caso de un apagado inesperado, para salvaguardar la seguridad de las personas y los bienes es necesario que el hardware cuente con un sistema que lo provea de energía durante un tiempo definido.

Métricas de rendimiento para sistemas biométricos. Los métodos para la comprobación del sistema biométrico serán el TAR, FAR, FRR y TRR, estos se obtendrán al realizar las pruebas correspondientes, analizados mediante porcentajes obtenidos de cada uno.

TAR o Tasa de aceptación real, se medirá la cantidad de accesos y autenticaciones reales realizados.

FAR o Tasa de aceptación falsa, en esta se medirá la cantidad de accesos falsos que el sistema permite, es decir la cantidad de personas cuyo acceso fue concedido, pero no son los usuarios reales.

FRR o Tasa de rechazos falsos, con esta tasa se medirá la cantidad de rechazos a personas cuyas identidades han sido registradas, es decir cuyo acceso debió ser concedido, pero se rechazó.

TRR o Tasa de rechazo real, analizará la cantidad de rechazos verdaderos, cuando sean usuarios no registrados y los reconozca como tales.

Con el uso de estos métodos y elementos se espera conseguir un sistema de seguridad confiable, fácil de usar y con una alto TAR junto con su respectivo TRR y bajos FAR y FRR, para comprobar esto se procederá a realizar pruebas con 6 personas de las cuales se analizará los resultados de la interacción con el sistema para comprobar si cumple con lo esperado.

El sistema será implementado en una plataforma Raspberry PI, esta contará con su propia estructura la cual integrará los accesorios cuyo tamaño lo permita, como una pantalla para poder visualizar las lecturas faciales, la cámara y el resto de accesorios como el parlante y micrófono serán elementos externos.

Para enfriar el procesador se usará un ventilador pues un sobrecalentamiento puede dañar seriamente los equipos.

Estrategias y/o técnicas

Registro de patrones biométricos faciales. Para cada patrón biométrico se necesitará un instrumento que adquiera la señal, en este caso los dispositivos deberán ser compatibles para trabajar con Linux ya que el proyecto se realiza en un ambiente de Raspbian basado en Ubuntu.

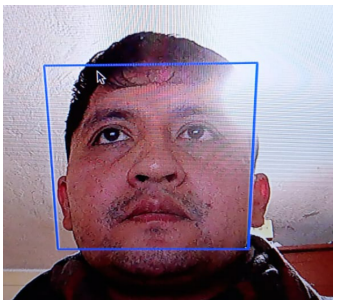
Adquisición de imágenes. Para la adquisición de esta señal se tomará en consideración los criterios de análisis de imágenes, ya que se va a utilizar Haar Cascade para el reconocimiento

del rostro no es necesario una cámara de alta resolución, en este caso se tomará una cámara web N8 de 1080P.

Detección del rostro. Se generará un script (M_REGISTRO_ROSTRO.py) para la adquisición de imágenes con base en Haar Cascade en Python, esto para la detección del rostro. Serán configuradas las características para la detección, como la conversión a escala de grises para su respectivo análisis (esto ayuda a un menor consumo de espacio de memoria), un scaleFactor de 1.07 para un análisis profundo pero sin forzar el consumo de la RAM, un minNeighbors de 4 para disminuir el número de detecciones falsas, un minSize de (250, 250) y un maxSize de (900, 900) para detectar los rostros hasta una distancia de 2 metro, debido a que se trata de un sistema de acceso y los usuarios tienen que ubicarse frente a la cámara no amerita mayor amplitud en este rango, se puede ver el resultado en la Figura 21.

Figura 21

Detección de rostro con configuración implementada



Almacenamiento de imágenes. Luego que el sistema detecte el rostro con la configuración realizada, se procede al almacenamiento de la imagen, para esto es necesario que se solicite una etiqueta o un identificador para designar un espacio de memoria para los registros tanto facial como de voz.

En el script (M_REGISTRO_ROSTRO.py) se solicitará al usuario un nombre y un apellido, usados como etiqueta para poder generar una carpeta con el nombre del usuario, en el caso de que el usuario ya exista, el sistema le solicitará cambiar a otro nombre para reconocerlo y si no existe se generará una carpeta en la dirección denominada como base de datos.

Captura de imágenes. Tomando las recomendaciones de variabilidad en los rostros, el sistema sugerirá realizar gestos variados, esto mientras prosigue la captura de imágenes, se realizará en 6 posiciones diferentes. Para iniciar, el sistema toma 20 capturas frontales, prosigue con 10 capturas del lado izquierdo del rostro, 10 capturas del lado derecho, 10 capturas de la parte inferior y finaliza con 10 capturas de la parte superior.

Esta versatilidad en las imágenes permite una amplia gama del reconocimiento en diferentes posturas del usuario, para esto se recomienda el registro en un ambiente con luz estable, como se puede observar en la Figura 22, transformada a escala de grises.

Figura 22

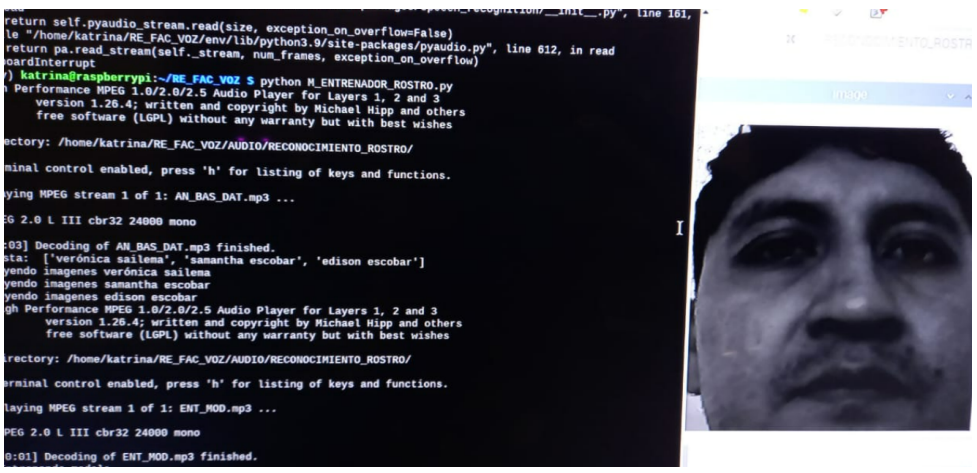
Conversión de imagen escala de grises

Se completará el almacenamiento de las imágenes con un total de 60 capturas con diferentes ángulos del rostro, cada una en una carpeta personal identificada con el nombre del usuario.

Entrenamiento de modelo. Para acelerar el reconocimiento facial es necesario contar con un algoritmo que agilite este proceso por lo cual se elaborará un script (M_ENTRENADOR_ROSTRO.py), el cual contendrá el modelo de entrenamiento con esto se evita realizar la comparación una imagen a la vez, para ello se usa el entrenamiento de modelo "LBPHFaceRecognizer" con el cual genera una matriz que guarda las características binarias de cada captura, el tiempo de análisis varía dependiendo de las cantidad de usuarios registrados y es necesario realizarla cada vez que se añada un usuario al grupo, en la Figura 23, se puede observar la ejecución del entrenador.

Figura 23

Ejecución de entrenador



Ya finalizado el registro, el sistema anunciará que se guardaron los datos biométricos exitosamente.

Registro de patrones biométricos de la voz.

Adquisición de patrones de voz. Para la adquisición de este patrón es necesario tomar en cuenta los proyectos anteriores, estos sugieren contar con un sistema de supresión de ruido sea físico o digital, por tal motivo se selecciona el micrófono cardioide (PYLE-PDMIUSBMT300), cuenta con varias modalidades para adquisición de sonidos entre ellas el estéreo que presenta mayor claridad al momento de adquirir las señales, un envío de paquetes de 24 bits que adquiere la mayor cantidad de información de cada grabación y por su diseño capacitivo suprime en gran cantidad los ruidos externos.

Se genera un script (M_LEC_REC_VOZ.py) en el cual se configura la adquisición del sonido en paquetes de 24 bits para aprovechar la mayor cantidad de información en la muestra de sonido, esto se guarda en un formato WAV, debido a que este formato puede almacenar la mayor cantidad de información con un mínimo de pérdida de datos.

Se importará la librería “speech_recognition” y “OS” para implementar un filtro de ruido digital, en el caso de que el físico no logre suprimir todo el ruido externo o se genere ruido en el proceso de conversión de la señal, con “OS” guardar la grabación de voz en un formato WAV como se puede observar en la Figura 24.

Figura 24

Implementación de filtro de ruido y conversión a formato WAV

```
        audio = r.listen(source)
        try:
            Contr_audio =r.recognize_google(audio, language="es-EC")
            Contr_audio = Contr_audio.lower()
            Contr_audio = Contr_audio.split()
            #print(Contr_audio[0],"Y",Contr_audio[1])
        except LookupError:
            Contr_audio = "Baja conexion de red"

    with open("/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_VOZ/ANALISIS/MUESTR
        file.write(audio.get_wav_data())
        print('Audio listo para procesar')
    return Contr_audio
```

Tratamiento de la señal. Al obtener la señal de voz filtrada, libre de ruido se tiene lo siguiente imagen en la Figura 25, esto se debe a que se encontrara en el dominio del tiempo, en este estado es difícil su análisis por lo cual es necesario su conversión en el dominio de la frecuencia para obtener sus frecuencias más relevantes, esto se logra aplicando la FFT mediante la librería scipy de Python y sus sub librerías como se observa en la Figura 26. Programado en el script (M_ANALISIS_FFT_VOZ.py).

Figura 25

Señal en función del tiempo

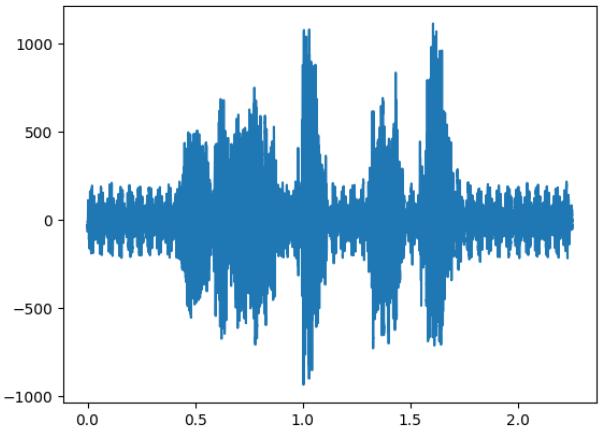


Figura 26

Librerías necesarias para la Transformada Rápida de Fourier

```
import scipy.fftpack as fourier
import scipy.io.wavfile as waves
import scipy.signal
import scipy.fft
```

Al ejecutarse la FFT se obtendrá una muestra simétrica en el dominio de la frecuencia como se observa en la Figura 27, para omitir estos datos duplicados se dividirá la muestra, obteniendo así un conjunto de datos más específicos como se observa en la Figura 28.

Figura 27

Señal en función de la frecuencia completa

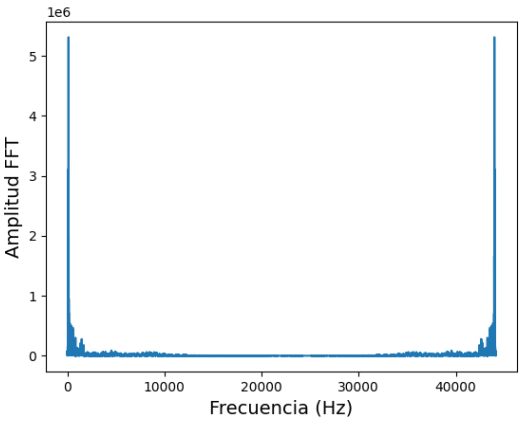
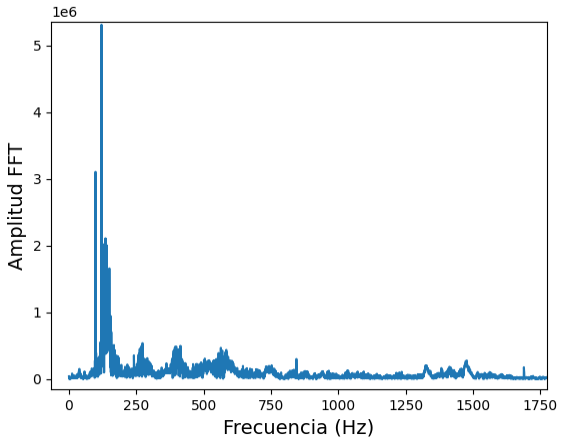


Figura 28

Señal en función de la frecuencia completa



En este punto se obtendrán dos factores uno es la contraseña y otra es la frecuencia de trabajo, ejecutados en el script (IDENTIFICACION_VOZ.py), es necesario tomar una muestra del patrón de voz el cual se deberá registrar y reconocer en las mismas condiciones. Debido a que la pronunciación o la tonalidad de voz difiere en cada palabra, llega a evitar un reconocimiento exitoso, para esto se decide seleccionar dos claves de lo cual se hablará a continuación:

Clave de usuario. El sistema solicitará dos claves las cuales pueden ser cualquier tipo de palabra sin adjetivos o conjunciones, serán guardadas en la carpeta de cada usuario, deben ser pronunciadas correctamente para el registro y repetidas de la misma forma para el reconocimiento.

Para la conversión de voz a texto se utilizará la librería GTTS de Google la cual permite convertir un fragmento de voz en texto, en conjunto con un algoritmo de separación de palabras y comparación de las mismas, permite evaluar las contraseña, este aspecto brinda una seguridad adicional, debido a que el texto resultante varía según la pronunciación y el idioma configurado, es decir el texto traducido de un usuario no será igual a otro, en la Figura 29 podemos observar un fragmento del código.

Figura 29

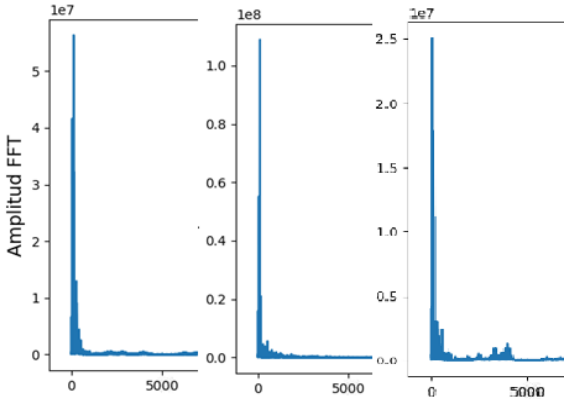
Uso de GTTS para la conversión de voz a texto

```
r = sr.Recognizer()
mic = sr.Microphone()
def escuchar(SUJETO):
    with mic as source:
        print("Separando Ruido")
        os.system("mpg123 /home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_VOZ/FI
        os.system("mpg123 /home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_VOZ/PR
        r.adjust_for_ambient_noise(source, duration=9)
        print("Puede hablar")
        audio = r.listen(source)
        try:
            Contr_audio =r.recognize_google(audio, language="es-EC")
            Contr_audio = Contr_audio.lower()
            Contr_audio = Contr_audio.split()
            #print(Contr_audio[0],"Y",Contr_audio[1])
        except LookupError:
            Contr_audio = "Baja conexion de red"
```

Registro de voz. Al contar con una muestra de voz legible, en este caso la misma en la cual se pronuncia la contraseña, ahora se la convierte en función de la frecuencia luego de haber aplicado la FFT, es necesario que el sistema seleccione la mejor forma de diferenciar un usuario de otro, esto se lo realizará con la extracción de la frecuencia máxima de señal. Para contar con un rango de operación más exacto se solicitará al usuario la pronunciación de las claves por tres ocasiones, con esto se realizará el promedio y se obtiene el rango final de operación en el cual se encuentra la frecuencia predominante. En ciertos casos estos difieren levemente como se puede observar en la Figura 30. La amplitud varía según el volumen de la voz, más aún la frecuencia permanece sin una variación considerable las tres veces en que la contraseña es pronunciada.

Figura 30

Diferencia entre fragmentos de voz

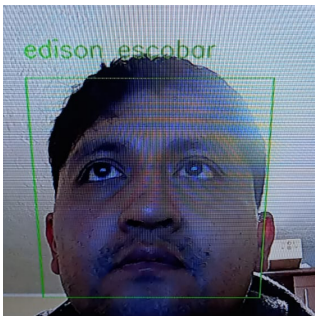


Reconocimiento de patrones biométricos. Para acceder a este sistema es necesario tener en cuenta las claves registradas.

Reconocimiento facial. En el reconocimiento se utilizará el método LBP, ejecutado en el script (RECONOCIMIENTO_ROSTRO.py), este método se caracteriza por su baja sensibilidad a las sombras y la variación de luz. De igual forma que en el registro, primero detecta un rostro y lo identifica, mediante la comparación con el modelo pre entrenado, como se puede observar en la Figura 31.

Figura 31

Reconocimiento de rostro mediante el uso del modelo pre entrenado



Reconocimiento de voz. En la fase de reconocimiento de voz será necesario tener presente las dos claves registradas, el sistema solicitará las claves y al mismo tiempo analizará el rango de frecuencia en el cual se ubica la voz del usuario.

Se utilizará la librería GTTS de Google para convertir el texto a voz y realizará una comparación entre las claves pronunciadas con las guardadas en la base de datos correspondientes al usuario que se encuentra en proceso de reconocimiento.

Posteriormente se ejecuta un script (M_LEC_REC_VOZ.py) para guardar el fragmento de audio con las claves en un formato WAV, se realiza la FFT para obtener el valor representativo de la frecuencia y posteriormente compararlo con los datos guardados, para que el sistema lo considere como correcto deberá situarse entre los valores de rangos registrados, se puede observar el fragmento de código en la Figura 32.

Figura 32

Fragmento de script para el reconocimiento

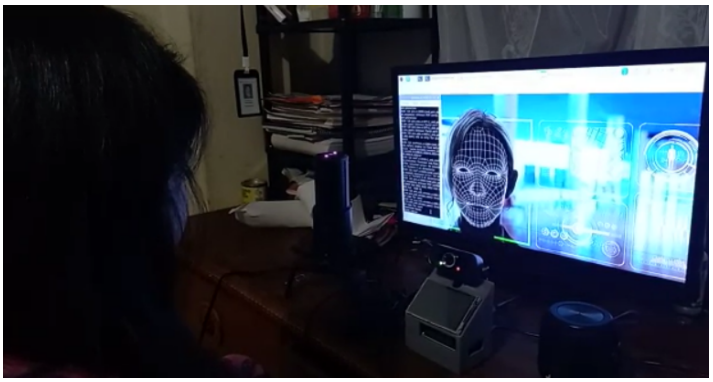
```
if RG_RC==3:
   REG_A.append(REG_A1/3)
   REG_A.append(REG_A2/3)
   print(REG_A)
   np.savetxt("/home/katrina/RE_FAC_VOZ/RecoFacial/Registro_voz/"+SUJETO+".t

elif RG_RC==0:
    REC_A=VALORESF[2]
    REC_B=np.loadtxt("/home/katrina/RE_FAC_VOZ/RecoFacial/Registro_voz/"+SU
    print(REC_A,REC_B)
    if REC_A <=REC_B[0] and REC_A >=REC_B[1]:
        COM_ACCESS_OK= "Contraseña y patrón de voz correctos, acceso conced
        print("Acceso concedido "+SUJETO)
        ACCESS_OK = gTTS(text=COM_ACCESS_OK, lang=idioma,slow=False)
        ACCESS_OK.save("/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_VOZ/RI
        os.system("mpg123 /home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_VOZ/
```

Método de fusión para reconocimiento de usuario. El método de fusión usado para este proyecto será el secuencial, es decir el acceso se dará mediante la confirmación del usuario en primera instancia por la biometría facial, seguido por la comparación de claves y finalmente la frecuencia de voz, el reconocimiento facial se puede observar en la Figura 33.

Figura 33

Primera etapa del reconocimiento, biometría facial



Posterior a la confirmación del usuario por vía facial, el sistema le concederá el acceso al reconocimiento de la clave, en este paso tendrá que pronunciar la clave de forma correcta, tal y como se registró en la etapa anterior, si no concuerda con la clave registrada o no pronuncio el número correcto de palabras podrá contar con un intento más, luego de superar los dos intentos el sistema abortará el reconocimiento y regresará a su modo de espera.

Si es correcta la clave, el registro de voz con la clave se guardarán en un archivo de audio temporal WAV, para ser procesado mediante FFT y comparado con los valores de frecuencia registrados, si no cumple el sistema aborta el proceso y regresará a su modo de espera, caso contrario que la clave sea la correcta el sistema le concederá el acceso.

Sistema interactivo por comandos de voz. Para poder ejecutar esto se utilizará el conversor de voz a texto de Google “GTTS” y en el caso de texto a voz “Speech”, para ambos el uso de internet será necesario. Las traductoras offline generan voces robotizadas por lo cual no serían mayormente aceptadas por el usuario.

El sistema contará con una matriz de saludos y respuestas (M_MATRIZ_RESP.py), a la cual se puede acceder vía editores para la adición de mayor amplitud en las respuestas, se muestra un fragmento de su código en la Figura 34.

Figura 34

Matriz de saludos y respuestas

```
import os
import numpy as np
import statistics
saludo=[["hola","none","none"],["estimado","none","none"],["estimada","none","none"],["encantado","de","conoc
resp_saludo=("hola, cómo estás","hola, cómo le va","hola, que tal","de igual manera","de igual forma","buenos
matriz=np.array(saludo)
def Ma(ki):
    i=j=m=suma=0
    numeros=[]
    numeros2=[]
    unico=[]
    rep1=[]
    palabras=(ki.split())
    try:
        for k in palabras:
            for i in range (17):
                for j in range (3):
                    if matriz[i][j] == k:
                        #if matri
                        #print(i,j)
                        numeros.append(i)
                        numeros2.append(j)
```

Los saludos serán parte de su sistema, otro aditamento es que siempre se encontrará a la escucha esperando una instrucción para realizar, esto se encontrará programado en su script principal denominado “RESP_INTERACT”.

Gracias a esta cualidad el usuario podrá activar los modos de reconocimiento y registro mediante la voz, para añadir seguridad en la fase del registro se solicitará cada vez que alguien quiera registrarse que el administrador se coloque frente a la cámara y el sistema lo reconozca, caso contrario no permite los registros, esto se encuentra programado en el script “RESP_INTERACT”.

El sistema se comunicará con el usuario mediante el micrófono y se lo escuchará por medio del parlante, sea esto para responder a comandos o solicitar nuevos intentos en el caso de ingresar mal una clave.

Se implementará una pantalla micro LCD para monitorear la ejecución del programa o revisar las fallas emitidas, de ser necesario incluso se podría activar la salida micro HDMI de la plataforma para conectarla a una pantalla, permitiendo mejorar la experiencia del usuario con el sistema, en la figura 35 se observa el sistema con los dispositivos previamente instalados.

Figura 35

Sistema con dispositivos conectados

Validación de la propuesta

Tabla 1

Descripción de perfil de validadores

Nombres y Apellidos	Años de experiencia	Titulación Académica	Cargo
Edisson Fernando Herrera Nuñes	5	Master Universitario en Industria 4,0	Docente del Instituto Tecnológico Particular Tsachila
Paúl Isaías Tinizaray Romero	4	Master en Computación con mención en sistemas inteligentes	Técnico de investigación en la Escuela Politécnica Nacional
Pamela Monserrath Espejo Velasco	7	Master en sistemas de Control y Automatización Industrial	Docente SENESCYT Instituto Superior Tecnológico Tungurahua

Tabla 2

Escala de evaluación del validador 1

CRITERIOS	EVALUACIÓN SEGÚN IMPORTANCIA Y REPRESENTATIVIDAD				
	En Total Desacuerdo	En Desacuerdo	Ni de Acuerdo Ni en Desacuerdo	De Acuerdo	Totalmente Acuerdo
Impacto					x
Aplicabilidad				x	
Conceptualización				x	
Actualidad					x
Calidad Técnica					x
Factibilidad				x	
Pertinencia					x

Nota. Elaborada por: *Ing. Wilmer Fabian Albarracín Guarochico MBA.*

Tabla 3

Escala de evaluación del validador 2

CRITERIOS	EVALUACIÓN SEGÚN IMPORTANCIA Y REPRESENTATIVIDAD				
	En Total Desacuerdo	En Desacuerdo	Ni de Acuerdo Ni en Desacuerdo	De Acuerdo	Totalmente Acuerdo
Impacto					x
Aplicabilidad					x
Conceptualización				x	
Actualidad					x
Calidad Técnica				x	
Factibilidad					x
Pertinencia				x	

Nota. Elaborada por: *Ing. Wilmer Fabian Albarracín Guarochico MBA.*

Tabla 4

Escala de evaluación del validador 3

CRITERIOS	EVALUACIÓN SEGÚN IMPORTANCIA Y REPRESENTATIVIDAD				
	En Total Desacuerdo	En Desacuerdo	Ni de Acuerdo Ni en Desacuerdo	De Acuerdo	Totalmente Acuerdo
Impacto					x
Aplicabilidad				x	
Conceptualización				x	
Actualidad					x
Calidad Técnica				x	
Factibilidad					x
Pertinencia				x	

Nota. Elaborada por: *Ing. Wilmer Fabian Albarracín Guarochico MBA*

Matriz de articulación de la propuesta

Tabla 5

Matriz de articulación

Ejes o partes principales del proyecto		Breve descripción de los resultados de cada parte	Sustento teórico que se aplicó en la construcción del proyecto	Metodologías, herramientas técnicas y tecnológicas que se emplearon
1	Micrófono cardioide MT300	Suprime el ruido, modo estéreo permite mejor claridad de voz, salida de 24 bits permite obtener mayor información de cada señal.	Análisis de sonido, problemas del ruido, métodos de supresión.	
	Cámara web HD 1080 P	Debido al uso de Haar Cascade para la adquisición de registro biométrico no fue necesario el uso de una cámara de mayor resolución.	Métodos de procesamiento de imágenes con Haar Cascade.	Método cuantitativo, análisis experimental de pruebas y de resultados.
	Raspberry PI 4 B	Esta plataforma permite la programación de manera fácil y su procesador el análisis de variables.	Sistemas operativos y hardware libre	
2	Reconocimiento de rostro	Se aplicó un script en Python mediante el uso de OpenCV y sus múltiples librerías.	Librerías OpenCV y características para el reconocimiento facial.	Se usan las librerías en python ya que el proyecto se enfoca en la programación en este entorno.
	Reconocimiento de voz	Librerías scipy y numpy de python que permite la transformada y el análisis de las señales.	Métodos de análisis de ondas y su procesamiento	

3	Sistema interactivo	Uso de script con conversores de txt a voz y viceversa para combinar comando e instrucciones.	Uso de conversores de texto a voz	GTTS y Speech son los mejores en cuanto a conversión se trata y son libres de pago.

Análisis de resultados. Presentación y discusión

Se realizó el registro y posterior reconocimiento de 6 personas como se observa en la Tabla 6, se efectuó 10 pruebas a cada uno obteniendo lo siguiente:

Tabla 6

Usuarios de prueba

Nombre	Número de Usuario
Verónica Sailema	Usuario 1
Samantha Escobar	Usuario 2
Edisson Escobar	Usuario 3
Luis Sailema	Usuario 4
Doménica Sailema	Usuario 5
Mayra Sailema	Usuario 6

Se nota una variación en la lectura al realizarla con luz ambiental y artificial, el LBP ayuda a un mejor reconocimiento, pero aún existe el problema de la variación de luz generada por el ambiente.

Presentación de resultados del primer filtro-reconocimiento facial

Se realizaron pruebas en las cuales se analiza la TAR, FAR y FRR para determinar el mejor índice de confiabilidad para el reconocimiento con LBP, los resultados de estas pruebas se pueden observar en las Tablas 7, 8 y 9.

En la Tabla 7 se puede evidenciar que con el índice LBP de 40, existe un TAR de 11,67% es decir el reconocimiento correcto de una persona disminuye radicalmente y aumenta el rechazo falso FRR a un 88,33%.

Tabla 7

Índice de confiabilidad LBP de 40

Índice de confiabilidad LBP de 40											Tasa de aceptación real - TAR	Tasa de aceptación falsa - FAR	Tasa de Rechazo Falso - FRR
	1	2	3	4	5	6	7	8	9	10			
Usuario 1	X	X	X	X	X	X	X	X	X	√	1	0	9
Usuario 2	√	X	X	X	X	X	X	√	X	X	2	0	8
Usuario 3	X	X	X	X	X	X	X	X	X	X	0	0	10
Usuario 4	X	X	X	X	X	X	X	√	X	X	1	0	9
Usuario 5	X	√	X	X	X	√	X	X	√	X	3	0	7
Usuario 6	X	X	X	X	X	X	X	X	X	X	0	0	10
											7	0	53
Resultados											11,67%	0,00%	88,33%

Nota. Las casillas en rojo son los FRR, los verdes los TAR y los FAR en café.

En la Tabla 8 se evidencia un aumento en el reconocimiento real y una notoria disminución del reconocimiento falso, en esta prueba aparece un porcentaje bajo de aceptación falsa es decir se confundió a un usuario con otro.

Tabla 8

Índice de confiabilidad LBP de 52

Índice de confiabilidad LBP de 52											Tasa de aceptación real - TAR	Tasa de aceptación falsa - FAR	Tasa de Rechazo Falso - FRR
	1	2	3	4	5	6	7	8	9	10			
Usuario 1	√	C	√	√	√	√	√	√	√	√	9	1	0
Usuario 2	√	√	√	√	√	√	√	√	X	√	9	0	1
Usuario 3	√	√	√	√	√	√	√	√	√	√	10	0	0
Usuario 4	√	√	√	X	√	√	√	C	√	√	8	1	1
Usuario 5	√	√	√	√	√	√	√	√	√	√	10	0	0
Usuario 6	√	√	√	√	√	√	√	X	√	√	9	0	1
											55	2	3
Resultados											91,67%	3,33%	5,00%

Nota. Las casillas en rojo son los FRR, los verdes los TAR y los FAR en café.

En la Tabla 9 se evidencia un aumento muy significativo en las aceptaciones falsas, es decir confunde a los usuarios unos con otros, mientras que las aceptaciones verdaderas disminuyen radicalmente.

Tabla 9

Índice de confiabilidad LBP de 60

Índice de confiabilidad LBP de 60											Tasa de aceptación real - TAR	Tasa de aceptación falsa - FAR	Tasa de Rechazo Falso - FRR
	1	2	3	4	5	6	7	8	9	10			
Usuario 1	C	C	√	X	C	C	X	C	C	X	1	6	3
Usuario 2	√	C	X	C	C	C	C	C	C	√	2	7	1
Usuario 3	C	C	X	C	X	√	X	C	C	C	1	6	3
Usuario 4	C	C	C	C	C	X	C	X	C	C	0	8	2
Usuario 5	C	√	C	X	C	C	C	√	C	C	2	7	1
Usuario 6	C	C	X	C	C	C	C	C	C	C	0	9	1
											6	43	11
Resultados											10,0%	71,7%	18,3%

Nota. Las casillas en rojo son los FRR, los verdes los TAR y los FAR en café.

Presentación del tercer filtro-reconocimiento de voz

Para el análisis de frecuencias predominantes se realizaron tres medidas de esta frecuencia para encontrar el máximo y mínimo valor en el cual oscila esta señal en cada usuario.

Tabla 10

Valores máximos y mínimos de frecuencias

	Frecuencia predominante promedio (Hz)
Usuario 1	192-200
Usuario 2	230-232
Usuario 3	95-100
Usuario 4	150-158
Usuario 5	238-245
Usuario 6	185-190

Presentación de resultados del reconocimiento en el entorno del registro

En las pruebas realizadas fue necesario usar un factor de confiabilidad de 52, esto para disminuir el índice de reconocimientos erróneos y confusión de biometrías faciales entre usuarios, dando como resultado una disminución de la TAR. En la Tabla 11 se puede observar un TAR de 91.67 %, es decir de 60 pruebas realizadas 55 reconocimientos faciales fueron exitosos y los mismo coincidieron en el reconocimiento de contraseña y se encontraron dentro del rango de frecuencias registradas.

Además, existieron 3 rechazos falsos, es decir a tres usuarios registrados no se los reconoció fácilmente dando un FRR de 5 %.

La FAR obtuvo un 3.33 %, esto debido a que el reconocimiento facial confundió a los usuarios 1 y 4 con los 2 y 3 respectivamente, esto le sucedió una vez a cada usuario, lo cual no se repitió en el resto de pruebas, este factor al complementarse con el reconocimiento de voz redujo el porcentaje a 0%, debido a que estos usuarios no conocían las contraseñas, para comprobar la tercera barrera en la misma prueba se les facilitaron las contraseñas y en este caso no coincidió su frecuencia registrada.

Tabla 11

Tabla de resumen de resultados con igual iluminación del registro

Tabla de resultados del reconocimiento multimodal														
Reconocimiento Facial														
Usuario	Pruebas										Lecturas totales	Tasa de aceptación real - TAR	Tasa de aceptación falsa - FAR	Tasa de rechazo falso - FRR	Tasa de rechazo real - TRR
	1	2	3	4	5	6	7	8	9	10					
Usuario 1	√	C	√	√	√	√	√	√	√	√	10	9	1	0	0
Usuario 2	√	√	√	√	√	√	√	√	X	√	10	9	0	1	0
Usuario 3	√	√	√	√	√	√	√	√	√	√	10	10	0	0	0
Usuario 4	√	√	√	X	√	√	√	C	√	√	10	8	1	1	0
Usuario 5	√	√	√	√	√	√	√	√	√	√	10	10	0	0	0
Usuario 6	√	√	√	√	√	√	√	X	√	√	10	9	0	1	0
Lecturas en biometría facial											60	55	2	3	0
Reconocimiento de voz															
Usuario 1	√	X	√	√	√	√	√	√	√	√	10	9	0	0	1
Usuario 2	√	√	√	√	√	√	√	√		√	9	9	0	0	0
Usuario 3	√	√	√	√	√	√	√	√	√	√	10	10	0	0	0
Usuario 4	√	√	√		√	√	√	X	√	√	9	8	0	0	1
Usuario 5	√	√	√	√	√	√	√	√	√	√	10	10	0	0	0
Usuario 6	√	√	√	√	√	√	√		√	√	9	9	0	0	0
Lecturas en biometría de voz											57	55	0	0	2
Resultados												91,67%	3,33%	5,00%	3,33%

Nota. Los vistos significan pruebas exitosas, las C reconocimientos fallidos y las C el reconocimiento de otra persona perteneciente al grupo de registro.

Presentación de resultados del reconocimiento con variaciones de iluminación

En este caso se realizó la prueba a las personas ya registradas, en un entorno bajo una luz de diferente intensidad. Los resultados se reflejan en la Tabla 12, se puede ver que al utilizar una confiabilidad LBP de 50 reduce la FAR, pero se vuelve muy sensible a los cambios de luz, con un TAR del 11.67%, un FRR de 88.33% pero generando un FAR de 0%.

El reconocimiento de voz y contraseña complementarios no sufren variación y son coincidentes con los usuarios reales, ya que no se presenta un FAR, es muy restrictivo y no permite el acceso a usuarios no identificados.

Tabla 12

Tabla de resumen de resultados con variación de iluminación

Tabla de resultados del reconocimiento multimodal														
Reconocimiento Facial														
Usuario	Pruebas										Lecturas Totales	Tasa de aceptación real - TAR	Tasa de aceptación falsa - FAR	Tasa de Rechazo Falso - FRR	Tasa de Rechazo Real - TRR
	1	2	3	4	5	6	7	8	9	10					
Usuario 1	X	X	X	X	√	X	X	X	X	√	10	2	0	8	0
Usuario 2	X	X	X	X	X	X	X	X	X	√	10	1	0	9	0
Usuario 3	X	X	X	X	X	X	X	X	X	X	10	0	0	10	0
Usuario 4	X	X	X	√	X	X	X	√	X	X	10	2	0	8	0
Usuario 5	X	X	X	X	X	√	X	X	X	X	10	1	0	9	0
Usuario 6	X	X	X	X	√	X	X	X	X	X	10	1	0	9	0
Lecturas en biometría facial											60	7	0	53	0
Reconocimiento de voz															
Usuario 1					√					√	2	2	0	0	0
Usuario 2										√	1	1	0	0	0
Usuario 3											0	0	0	0	0
Usuario 4				√				√			2	2	0	0	0
Usuario 5						√					1	1	0	0	0
Usuario 6					√						1	1	0	0	0
Lecturas en biometría de voz											7	7	0	0	0
Resultados												11,67%	0,00%	88,33%	

Análisis de resultados obtenidos

Con los resultados obtenidos en las pruebas para determinar el LBP como método de reconocimiento complementario al reconocimiento por discriminación de frecuencias de voz, se evidencia que un índice de confiabilidad de 52 permite un TAR de 91,67%, un FRR de 5%, un FAR de 3,33% y un TRR de 3,33%, esto significa que el índice de reconocimiento real es alto, con un rechazo falso bajo, el porcentaje de confusión entre usuarios es de 3,33% pero el reconocimiento de frecuencias y la solicitud de clave disminuyen este factor a cero, brindando protección en el caso de reconocer falsamente un rostro.

Este análisis se realizó en el mismo ambiente luminoso en el cual se registraron, es decir sin cambios en la luz, sean éstas luz natural o artificial.

De igual manera se efectuó la prueba en un ambiente con una luz variable, obteniendo un TAR de 11.67% y un FRR de 88,33%, es decir que existe un reconocimiento exitoso muy bajo y un rechazo falso alto, el sistema rechaza a gran parte de usuarios verdaderos. Ya que el índice de reconocimientos verdaderos bajó, también disminuyó a 0% el reconocimiento erróneo de usuarios, esto evidencia que el grado de confusión depende del índice de reconocimientos reales.

En este caso el sistema diseñado puede ser óptimamente utilizado en ambientes con luces artificiales fijas y sin ninguna variación lumínica natural. Comparándolo con sistemas como (Moreno, 2021), cuenta con un TAR bajo de 91,67% frente al 99,51% del proyecto citado, pero redujo su FAR de 1,08% del mismo proyecto a 0% en el proyecto actual, es necesario considerar que el proyecto de mencionado se evaluó con una muestra mayor a la utilizada en el proyecto realizado.

En 2020, Zhang et al. centró su investigación en aumentar el TAR tanto en el reconocimiento facial como en el de voz, su sistema se enfoca en el reconocimiento del rostro pese al uso de disfraces o máscaras, esto aumenta la tasa de reconocimiento falsos, no es óptimo para un control de acceso pues se debe limitar a reconocer específicamente a la persona correcta por lo cual en este proyecto se redujo los reconocimientos falsos a un 3.33% y se los redujo comparándolos con el registro de la contraseña hablada y los rangos de frecuencia de trabajo del usuario.

Adicionalmente se mejoró la interfaz de un sistema táctil presentado en el proyecto mencionado a uno controlado por voz, con el aditamento de una micro LCD para visualizar el progreso del reconocimiento y de requerirse vía HDMI se puede conectar a una pantalla externa para mejorar la interacción con el usuario.

CONCLUSIONES

La información obtenida de los estudios e investigaciones precedentes relacionados, son una excelente base de datos para una comparativa entre los índices de evaluación TAR, FAR, FRR y TRR usados para calificar los sistemas de identificación.

La plataforma Raspberry PI 4 B utilizada en este proyecto cumple con las exigencias de procesamiento de imágenes, pese a tener 4 GB de RAM no presenta problemas al ejecutar script de análisis de datos como en las FFT o el análisis facial LBP.

El tamaño de la plataforma la convierte en una herramienta ideal para su implementación como sistema de acceso en edificios y domicilios, debido a que no ocupa gran espacio de instalación.

El UPS utilizado como complemento en conjunto con las baterías 18650 permiten una autonomía de uso de 1 hora 20 min, esto es esencial en un sistema de acceso pues debe permitir el ingreso o salida incluso al existir variaciones de energía.

El Script elaborado en Python al tener una filosofía secuencial, fortalece el sistema de acceso debido a que es un requisito coincidir con la primera biometría antes de acceder a la siguiente.

Al programar sistemas de acceso, en este caso con la biometría facial usando LBP como sistema de reconocimiento, permite una FAR baja, pero al mismo tiempo la limita a trabajar en el entorno de iluminación en el cual fue registrado, presenta una mejora no muy significativa al usar una iluminación artificial enfocada al rostro.

Se evidencia un aumento en el reconocimiento facial al usar iluminación variable, sólo si existe un registro que contenga capturas del rostro tomadas en presencia de esas variaciones.

El script de interacción, comunica satisfactoriamente al usuario con el sistema, debido a su librería Speech de Google es necesario el uso de una conexión de internet estable, lo cual lo convierte dependiente a la misma.

El programa es fácilmente usado por usuarios de diferentes edades, esto se evidenció al momento de la interacción de cada usuario, pues no fue necesario una explicación extensa para su uso.

Se obtiene una mejora al combinar los sistemas de reconocimiento fácil con el de voz esto se puede evidenciar en la tasa de aceptación falsa (FAR), la cual se genera en el reconocimiento facial, pero se anula al llegar al reconocimiento de contraseña y frecuencia de voz.

La interacción con el sistema es estable y la velocidad de respuesta a las instrucciones depende de la rapidez de la red.

RECOMENDACIONES

Es recomendable en el caso de sistemas de acceso elaborados en Python, investigar no solo en artículos pues en materia de programación GitHub cuenta con varios foros y programadores que brindan soporte en el caso de presentarse fallas en el sistema.

Es recomendable el uso de la plataforma en conjunto con un UPS, pues brinda protección al Hardware y respaldo en el caso de un corte súbito de energía, pues los sistemas de acceso deben permitir tanto el ingreso como la salida en el caso de una emergencia.

Al implementar el sistema puede programarse las salidas GPIO provistas en Raspberry PI para dar accionamiento a los actuadores necesarios.

Se recomienda el uso del sistema de reconocimiento, en lugares con iluminación controlada y estable, en este caso cabinas, como se utilizó en proyectos anteriores o al interior de edificios e instalaciones donde la luz ambiental interfiere en el entorno.

Es recomendable la instalación de un sistema de iluminación artificial enfocada al rostro, esto ayuda levemente al aumento del TAR.

La variabilidad en la toma de imágenes con diferentes fuentes de luz, es esencial para un buen reconocimiento.

Es necesaria la mejora de los sistemas de conversión TXT a voz “offline”, dotándolas de mejor fluidez para una interacción más amigable con el usuario, esta característica la poseen las aplicaciones Online, convirtiendo cualquier sistema que las use en dependientes de una conexión de red.

Es necesario generar un script que permita la interacción del sistema en varios idiomas, ya que en este caso se lo generó netamente en español.

Se recomienda el uso del sistema con una conexión de internet estable, pues la etapa de interacción es dependiente de la misma.

BIBLIOGRAFÍA

Adamovik. (1 de Julio de 2022). *Índice de Criminalidad 2022 Mitad de año*. https://es.numbeo.com/criminalidad/clasificaciones.

Ahmed, N., Hemayed, E., & Fayek, M. (2020). Hybrid Siamese Network for Unconstrained Face Verification and Clustering under Limited Resources, *BDCC,* 4(19), 1-21, https://www.mdpi.com/2504-2289/4/3/19.

Afrin, N., Sai, G., Krishna, K., & Rathan, K. (2022). Language Convertor Using Python. *International Research Journal of Modernization in Engineering Technology and Science,* 5 (6), 733-735, https://www.irjmets.com/uploadedfiles/paper//issue_6_june_2022/25567/final/fin_irjmets1654697583.pdf.

Athey, T., Liu, T., Pedigo, B., & Vogelstein. (2017). AutoGMM: Automatic and Hierarchical Gaussian Mixture Modeling in Python. *Journal of XYZ*, 5(1), 1-16, https://doi.org/10.48550/arXiv.1909.02688.

Baggio, D., Emami, S., Escrivá, D., Levgen, K., Mahmood, N., Saragih, J., & Shilkrot, R. (2012). *Mastering OpenCV with Practical Computer Vision Projects*. PACKT.

Balaka-Ramesh, N., & Prasad, R. (2019). Fusion of Face and Voice for a Multimodal Biometric Recognition System. *International Journal of Engineering and Advanced Technology (IJEAT)*, 8(3), 505-515, https://www.ijeat.org/wp-content/uploads/papers/v8i3/C5976028319.pdf.

Buvinic, M., Morrisson, A., & Orlando, M. (2005). Violencia, crimen y desarrollo social en América Latina y el Caribe. *Scielo*, 11 (43), https://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S1405-74252005000100008.

EP-0136. (5 de mayo de 2022). 52Pi Wiki. Retrieved from https://wiki.52pi.com/index.php?title=EP-0136&oldid=12046.

Hasan, R., & Sallow, A. (2021). Face Detection and Recognition Using OpenCV. *Journal of Soft Computing and Data Mining*, 2(2), 86-97, https://publisher.uthm.edu.my/ojs/index.php/jscdm/article/view/8791.

INEC. (2022). Estadísticas de Seguridad Integral: Delitos de mayor connotación psicosocial. Comisión Estadística de Seguridad Ciudadana y Justicia, https://www.ecuadorencifras.gob.ec/justicia-y-crimen/.

Juárez, U. (2018). *Reconocimiento de objetos y rostros con técnicas de visión por computadora e inteligencia artificial*, [Informe de práctica de entrenamiento Industrial, Centro de Ingeniería y desarrollo Industrial]. https://cidesi.repositorioinstitucional.mx/jspui/bitstream/1024/357/.

Martínez, J. (2016). *Diseño e implementación de un sistema de reconocimiento de voz mediante Raspberry PI*, [Tesis de Grado, Universidad Tecnológica de Pereira]. https://repositorio.utp.edu.co/items/eb9d1002-b495-4779-8ee0-192414b0a9d0.

Meng, W., Wong, D., Furnell, S., & Zhou, J. (2015). Surveying the Development of Biometric User Authentication on Mobile Phones, *IEEE*, 17(3), 1268 – 1293, https://ieeexplore.ieee.org/document/7000543.

Merchán, F., Galeano, S., & Poveda, H. (2016). Mejoras en el Entrenamiento de Esquemas de Detección de Sonrisas Basados en AdaBoost. *Revista Académica, UDP*, 10(2), 17-30, https://revistas.utp.ac.pa/index.php/id-tecnologico/article/view/21.

Motta, P., & Rogers, D. (2002*). Freeopen source software: an alternative for engineering and student*. [Archivo PDF]. http://laris.fesb.hr/digitalno_vodjenje/download/free-open-source.pdf.

Moreno, J., Atenco, J., Ramirez, J., Martinez, R., Gomez, P., & Fonseca, A. (2021). BIOMEX-DB: A Cognitive Audiovisual Dataset for Unimodal and Multimodal Biometric Systems. *IEEE,* 9(1), 111267-111276, https://ieeexplore.ieee.org/document/9496677.

Muños, E. (2021). *Desarrollo de un sistema de control de acceso de personal empleando reconocimiento facial respaldado con técnicas de aprendizaje profundo*, [Tesis de Grado, Universidad de las Fuerzas Armadas-ESPE]. http://repositorio.espe.edu.ec/bitstream/21000/25302.

NTi Audio Inc. (10 agosto del 2022). *Transformación rápida de Fourier FFT - Conceptos básicos*. https://www.nti-audio.com/es/servicio/conocimientos/transformacion-rapida-de-fourier-fft#:~:text=La%20%22Transformaci%C3%B3n%20r%C3%A1pida%20de%20Fourier,proporciona%20informaci%C3%B3n%20sobre%20su%20composici%C3%B3n.

Oloyede, M., & Hancke, G. (2016). Unimodal and Multimodal Biometric Sensing Systems: A Review. *IEEE*, 4(1), 7532 – 7555, https://ieeexplore.ieee.org/abstract/document/7580649.

Pardo, J. (2020). *Reconocimiento facial en tiempo real orientado a video llamadas o live stream para autenticar identidades durante una audiencia legal*, [Tesis de Grado, Universidad Santo Tomás Seccional Tunja]. https://repository.usta.edu.co/handle/11634/30508.

Parzibite. (1 de Julio del 2022). Conversión de texto a voz (TTS) con Python y gTTS. https://parzibyte.me/blog/2019/07/06/conversion-texto-voz-tts-python-gtts/.

Platero, D. (2015). *Reconocimiento de imágenes faciales orientado a controles de acceso y sistemas de seguridad.* [Tesis de Grado, Universidad Distrital "Francisco José de Caldas"]. https://repository.udistrital.edu.co/bitstream/handle/11349/7359/.

Raspberry PI. (2021). Raspberry Pi 4 Computer Model B [Archivo PDF]. https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-product-brief.pdf.

Santander, D., Rosero, K., & Libreros, M. (2020). *Reconocimiento de voz para un sistema de interacción humano máquina orientado a personas con limitaciones motrices*, [Tesis de Grado, FUNDACIÓN UNIVERSITARIA CATÓLICA LUMEN GENTIUM]. https://repository.unicatolica.edu.co/handle/20.500.12237/2102.

Sampieri, R. E., Collado, C., Baptista, M., Valencia, S., & Mendoza, C. (6 Eds.). (2014). *Metodología de la Investigación*. McGraw Hill.

Singh, N., Khan, R. (2015). Speaker Recognition and Fast Fourier Transform. *International Journal of Advanced Research in Computer Science and Software Engineering*, 5(7), 530-534, https://www.researchgate.net/profile/Nilu-Singh/publication/281843840_Speaker_Recognition_and_Fast_Fourier_Transform/links/55fae30208aeba1d9f3a0970/Speaker-Recognition-and-Fast-Fourier-Transform.pdf.

Spilsbury, M., & Euceda, A. (2017). Trasformada rápida de Fourier utilizando Python. *REVISTA DE LA ESCUELA DE FÍSICA, UNAH,* 5(1), 6-10, https://fisica.unah.edu.hn/assets/Revista/Volumen-V-N1/REF-UNAH-51-6.pdf.

Zhang, X., Cheng, D., Jia, P., & Xu, X. (2020). An Efficient Android-Based Multimodal Biometric Authentication System with Face and Voice. *IEEE,* 8, 102757 – 102772, https://ieeexplore.ieee.org/abstract/document/9104992/.

Shetty, A., Deeksha, B., Rebeiro, J., & Ramyashree. (2021). Facial recognition using Haar cascade and LBP classifiers. *Global Transitions Proceedings,* 2(1), 330-335, https://doi.org/10.1016/j.gltp.2021.08.044.

ANEXOS

ANEXO 1

RESP_INTERACT

```
#Respuestas interactivas katrina siempre escucha
import speech_recognition as sr
import pyttsx3
from gtts import gTTS
import os
import M_MATRIZ_RESP as MTRZR
import time
import RECONOCIMIENTO_ROSTRO as RERO
import M_REGISTRO_ROSTRO as REGIS
import IDENTIFICACION_VOZ as ID_V
idioma="es"
#go=0
r=sr.Recognizer()

def SpeakText(command):
    engine = pyttsx3.init()
    engine.say(command)
    engine.runAndWait()
while (True):
    with sr.Microphone() as source2:
        try:
```

```
        print("Separando Ruido")
        r.adjust_for_ambient_noise(source2, duration=2)
        print("Puede hablar")
        while (True):
            audio2 = r.listen(source2)
            try:
                Mytext =r.recognize_google(audio2, language="es-EC")
            except:
                Mytext= "Baja conexion de red"
            Mytext = Mytext.lower()
            print(Mytext)
            Enable = Mytext.split()
            print (Enable)
            go=0
            go1=0
            for ena in Enable:
                if ena=="catrina" or ena=="catrinas":
                    go=1
                    for ena2 in Enable:
                        if ena2=="activar" or ena2=="activa" or ena2=="activando":
                            for ena3 in Enable:
                                if ena3=="reconocimiento":
                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/INTERACCION/INI_RECON.mp3")
                                    Nom=RERO.RECON()
                                    print(Nom)
                                    ID_V.VOZ_ID(Nom,"Reconocer")
                                    go1=1
                                elif ena3=="registro":
                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/INTERACCION/INI_REGIS.mp3")
                                    Nom=RERO.RECON()
                                    go1=1
                                    if Nom=="edison escobar":
```

```
                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/INTERACCION/ADMIN.mp3")
                    NM_USR=REGIS.REG_ROS()
                    print(NM_USR)
                    ID_V.VOZ_ID(NM_USR,"Registrar")
                else:
                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/INTERACCION/NO_ADMIN.mp3")
            elif ena3=="descripción" or "identificación":
                os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/INTERACCION/DSCRP.mp3")
                go1=1
        if go==1 and go1==0:
            Nombre=MTRZR.Ma(Mytext)
            Mytext.split()
            print("Respuesta",Nombre)
            #Nombre="aprendiendo Edisson"
            myobj = gTTS(text=Nombre, lang=idioma,slow=False)

myobj.save("/home/katrina/RE_FAC_VOZ/AUDIO/INTERACCION/INTERAC_US.mp3")
            os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/INTERACCION/INTERAC_US.mp3")
            print(Nombre)
            time.sleep(0.5)
    except LookupError:
        print("No te entiendo")
```

ANEXO 2

M_MATRIZ_RESP

```
import os
import numpy as np
import statistics
saludo=[["hola","none","none"],["estimado","none","none"],["estimada","none","none"],["
encantado","de","conocerte"],["encantado","de","conocerla"],["que","gusto","verte"],["que","
```

```
gusto","verla"],["buenos","dias","none"],["buenas","tardes","none"],["buenas","noches","non
e"],["cómo","está","none"],["cómo","estás","none"],["cómo","vas","none"],["querida","none",
"none"],["todo","bien","none"],["qué","tal","none"],["aló","none","none"]]
    resp_saludo=("hola, cómo estás","hola, cómo le va","hola, que tal","de igual manera","de
igual forma","buenos dias","buenas tardes","buenas noches","bien","muy
bien","excelente","estimado","aló")
    matriz=np.array(saludo)
    def Ma(ki):
        i=j=m=suma=0
        numeros=[]
        numeros2=[]
        unico=[]
        rep1=[]
        palabras=(ki.split())
        try:
            for k in palabras:
                for i in range (17):
                    for j in range (3):
                        if matriz[i][j] == k:
                            #if matri
                            #print(i,j)
                            numeros.append(i)
                            numeros2.append(j)
    #print(numeros)
    #print(numeros2)
            suma=statistics.mean(numeros2)
    #print(suma)
    #for y in numeros2:
            if suma > 0:
                for x in numeros:
                    if x not in unico:
                        unico.append(x)
                    else:
                        if x not in rep1:
                            rep1.append(x)
```

```
            rep=rep1[0]
            for y in rep1:
                if numeros.count(y)>m:
                    m=numeros.count(y)
                    rep=y
                #print(y,numeros.count(y))
        else:
            rep=numeros[0]
        #print(rep)
        if rep==0 or rep==16:
            cont=resp_saludo[0]
        if rep==1:
            cont=resp_saludo[1]
        if rep==2:
            cont=resp_saludo[2]
        elif rep==3 or rep==5:
            cont=resp_saludo[3]
        elif rep==4 or rep==6 :
            cont=resp_saludo[4]
        elif rep==7:
            cont=resp_saludo[5]
        elif rep==8:
            cont=resp_saludo[6]
        elif rep==9:
            cont=resp_saludo[7]
        elif rep==10:
            cont=resp_saludo[8]
        elif rep==11 or rep==14:
            cont=resp_saludo[9]
        elif rep==12 or rep==15:
            cont=resp_saludo[10]
        elif rep==13:
            cont=resp_saludo[11]
    except ValueError:
        cont="Frase no reconocida"
```

```
        #print(cont)
        return cont
```

ANEXO 3

RECONOCIMIENTO_ROSTRO

```
#RECONOCIMIENTO DE ROSTROS
def RECON():

    import numpy as np
    import cv2
    import os
    from gtts import gTTS

    dataPath = "/home/katrina/RE_FAC_VOZ/RecoFacial/BaseDatos"
    imagePaths = os.listdir(dataPath)
    print ("imagePaths=" ,imagePaths)

    face_recognizer = cv2.face.LBPHFaceRecognizer_create()

face_recognizer.read("/home/katrina/RE_FAC_VOZ/RecoFacial/Entrenador/modeloLBPHFace.y
ml")
    faceCascade =
cv2.CascadeClassifier("/home/katrina/RE_FAC_VOZ/env/lib/python3.9/site-packages/cv2/data/
haarcascade_frontalface_default.xml")

    cam= cv2.VideoCapture(0)
    cam.set(3, 1280)
    cam.set(4, 720)

    minW= 0.1*cam.get(3)
    minH= 0.1*cam.get(4)
    print (minW)
    print (minH)
    conReg=0
```

```
    Regisro= []
    idioma1="es"
    while (True):
        ret, img =cam.read()
        img= cv2.flip(img, 1)
        gray= cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        auxFrame = gray.copy()
        faces= faceCascade.detectMultiScale(gray,
        scaleFactor= 1.07,
        minNeighbors= 4,
        minSize= (250, 250),
        maxSize= (900, 900))


        for(x,y,w,h) in faces:
            rostro = auxFrame [y:y+h,x:x+w]
            rostro = cv2.resize(rostro,(150,150), interpolation= cv2.INTER_CUBIC)
            result = face_recognizer.predict(rostro)
            fin= imagePaths[result[0]]

            if result[1] < 78 :
                cv2.putText (img,fin,(x,y-25),cv2.FONT_HERSHEY_SIMPLEX,1,(0,255,0),1,cv2.LINE_AA)
                cv2.rectangle(img, (x,y),(x+w,y+h),(0,255,0),1)
                conReg=Regisro.count(fin)

                if conReg == 0:
                    Regisro.append(fin)
                    myobj = gTTS(text=fin, lang=idioma1,slow=False)

myobj.save("/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/VARIABLE/USU
ARIO.mp3")
                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/BIENV.mp3 ,
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/VARIABLE/USUARIO.mp3")
                    return fin
```

```
        else:
            cv2.putText(img, "Desconocido",(x,y-20),2,0.8,(0,0,255),1,cv2.LINE_AA)
            cv2.rectangle(img, (x,y),(x+w,y+h),(0,0,255),2)

    cv2.imshow("img",img)
    k= cv2.waitKey(100) & 0xff
    if k== 27:
        print ("Reconocimiento abortado")
        break
cam.release()
cv2.destroyAllWindows()
```

ANEXO 4

M_REGISTRO_ROSTRO

```
    #REGISTRO DE ROSTRO
 def REG_ROS():
    import cv2
    import numpy as np
    import gc
    import os
    import time
    from shutil import rmtree
    from gtts import gTTS
    import M_ENTRENADOR_ROSTRO as ENROS
    import speech_recognition as sr
    import pyttsx3

    captur= cv2.VideoCapture(0)
    captur.set(3, 1280)
    captur.set(4, 720)
    #Direccionar a base de datos para lectura
                                                        face_detector=
cv2.CascadeClassifier("/home/katrina/RE_FAC_VOZ/env/lib/python3.9/site-packages/cv2/data/
haarcascade_frontalface_default.xml")
```

```
        #LINEAS PARA HABILITAR MAS USUARIOS
                                                                    #os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/NUM_USR_REG.mp3")
        #Numero= int(input('\n ¿Numero de usuarios que se registran?: '))
        Numero=1
        Dir="/home/katrina/RE_FAC_VOZ/RecoFacial/BaseDatos"
        User=1
        conNodet=0
        MICF=True
        Nodetect=[]
        enable_camera_led=1
        idioma="es"
        r=sr.Recognizer()
        def SpeakText(command):
           engine = pyttsx3.init()
           engine.say(command)
           engine.runAndWait()
                                                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/INI_REG_FAC.mp3")
        while (True):
           count=0
           print("\n  Usuario",User)
           while (MICF):
              with sr.Microphone() as source3:
                                                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/PRO_NOM.mp3")
                 r.adjust_for_ambient_noise(source3, duration=5)
                 #time.sleep(8)
                 print("Aislando Ruido")
                 print("Puede hablar")
                 audio3 = r.listen(source3)
                 MICF=False
                 try:
                    NOM_USR =r.recognize_google(audio3, language="es-EC")
                 except:
```

```
                                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/ERR_LEC.mp3")
                MICF=True
            NOM_USR = NOM_USR.lower()
            print(NOM_USR)
            NOM_USR1 = len(NOM_USR.split())
            print (NOM_USR1)
            if NOM_USR1==2:
              MICF=False
            else:
              MICF=True

        myobj = gTTS(text=NOM_USR, lang=idioma,slow=False)

myobj.save("/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/VARIABLE/USU
ARIO.mp3")
        Dirpers = Dir + "/" + NOM_USR

        if not os.path.exists(Dirpers):
          #myobj = gTTS(text=Nombre, lang=idioma,slow=False)
          #myobj.save("USUARIO.mp3")
                                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/EL_NOM.mp3          ,
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/VARIABLE/USUARIO.mp3  ,
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/NO_DT_CNT.mp3")
          print ("\n El nombre", NOM_USR, "no existe en la base de datos, puede continuar")
                                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/VARIABLE/USUARIO.mp3  ,
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/GEN_AR_USR.mp3")
          os.makedirs(Dirpers)
          print("\n   Inicializando mire directamente a la camara")

          while (True):
            try:
              ret, img= captur.read()
```

```
                img= cv2.flip(img, 1)
                #print(img)
                Escgris= cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
                Rostro= face_detector.detectMultiScale(Escgris,
                scaleFactor= 1.05,
                minNeighbors= 4,
                minSize= (250, 250),
                maxSize= (900, 900))
                k= cv2.waitKey(100) & 0xff
                if k== 27:
                   print ("Ejecucion abortada")
                   rmtree(Dirpers)
                   break
                cv2.imshow("img",img)
                for (x,y,w,h) in Rostro:
                   cv2.rectangle(img, (x,y),(x+w,y+h),(255,0,0),2)
                   count += 1
                   print ("\n    Registro", count,"Tomado")
                            cv2.imwrite(Dirpers + "/" + str(NOM_USR) + "." + str(count)  + ".jpg",
Escgris[y:y+h,x:x+w])
                   cv2.imshow("image", img)
                   if k== 27:
                      print ("Ejecucion abortada")
                      rmtree(Dirpers)
                      break
                   time.sleep(0.4)
                if count >= 1 and count <=59:
                   Nodetect.append(count)
                   conNodet= Nodetect.count(count)
                   print (conNodet)
                   if count == 20 and conNodet == 1:
                                                                        os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/MIR_DR.mp3")
                      print ("\n Mire ligeramente hacia la derecha")
                      time.sleep(0.3)
```

```
                    if count == 30 and conNodet == 1:
                                                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/MIR_IZ.mp3")
                        print ("\n Mire ligeramente hacia la izquierda")
                        time.sleep(0.3)
                    if count == 40 and conNodet == 1:
                                                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/MIR_AR.mp3")
                        print ("\n Mire ligeramente hacia arriba")
                        time.sleep(0.3)
                    if count == 50 and conNodet == 1:
                                                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/MIR_AB.mp3")
                        print ("\n Mire ligeramente hacia abajo")
                        time.sleep(0.3)
                    if conNodet > 15:
                                                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/RET_MIR.mp3")
                        print ("\n Retorne su mirada ligeramente hacia la cámara")
                        conNodet==0
                        Nodetect.remove(count)
                        #time.sleep(8)
                elif count >= 60:
                                                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/CAP_TERM.mp3")
                    print ("\n Toma de imaguenes terminada")
                    User+=1
                    time.sleep(0.2)
                    break

            except ValueError:
                print ("Ejecucion abortada por falla")
                rmtree(Dirpers)
                break
        else:
```

```
            print ("El nombre", NOM_USR, "esta ocupado")
                                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/EL_NOM.mp3 ,
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/VARIABLE/USUARIO.mp3 ,
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/NOM_OCUP.mp3")
            MICF=True
            time.sleep(1)

        k= cv2.waitKey(100) & 0xff
        if k== 27:
            print ("Ejecucion abortada")
            rmtree(Dirpers)
            break
        elif User > Numero:
            print ("\n Finaliza toma a usuarios")
            ENROS.END_MOD()
            break

    gc.collect(generation=0)
    gc.collect(generation=1)
    gc.collect(generation=2)
    captur.release()
    cv2.destroyAllWindows()
    return (NOM_USR)
```

ANEXO 5

M_ENTRENADOR_ROSTRO

```
#ENTRENADOR DE FUNCION DE RECONOCIMIENTO
def END_MOD():
    import cv2
    import os
    import numpy as np
```

```
                                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/AN_BAS_DAT.mp3")

    BaseDat = "/home/katrina/RE_FAC_VOZ/RecoFacial/BaseDatos"
    Lista = os.listdir(BaseDat)
    print("Lista: ", Lista)

                                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/ACT_ENT.mp3")

    labels = []
    facesData = []
    label = 0

    for nameDir in Lista:
      Identi = BaseDat + "/" + nameDir
      print ("Leyendo imagenes " +nameDir)

      for fileName in os.listdir(Identi) :
        labels.append(label)
        facesData.append(cv2.imread(Identi + "/" + fileName,0))
        image = cv2.imread(Identi + "/" + fileName,0)
        cv2.imshow("image",image)
        cv2.waitKey (10)
      label = label + 1

    face_recognizer = cv2.face.LBPHFaceRecognizer_create()

                                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/ENT_MOD.mp3")

    print ("Entrenando modelo.....")
    face_recognizer.train (facesData, np.array(labels))
                                                    face_recognizer.write
("/home/katrina/RE_FAC_VOZ/RecoFacial/Entrenador/modeloLBPHFace.yml")
```

```
        print ("Modelo almacenado....")

                                                                os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_ROSTRO/TER_ENT_MOD.mp3")
        cv2.destroyAllWindows()
```

ANEXO 6

IDENTIFICACIÓN_VOZ

```
       #Programa principal para el reconocimiento de voz
    def VOZ_ID(NM_USR,FUNCION):
        import M_LEC_REC_VOZ as LRV
        import M_ANALISIS_FFT_VOZ as av
        import numpy as np
        from gtts import gTTS
        import os

        idioma="es"
        REG_A=[]
        R_CTR=[]
        REG_A1=0
        REG_A2=0
        TOL=15
        conctr=0
        RG_RC=0
        ON2=True
        ON3=True
        ON4=True
        ON5=True

        SUJETO=NM_USR
          #FUNCION=input("Que desea hacer Registrar o Reconocer: "  ) #Pregunta si registrar o
reconocer
        print(FUNCION)
        #LRV.escuchar(SUJETO)   #Graba el audio para registro o reconocimiento.
```

```
        if FUNCION == "Registrar":
            RG_RC=3
                                                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_VOZ/TRE_AUD.mp3")
        elif FUNCION == "Reconocer":
            RG_RC=0
                                                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_VOZ/REC_AUD.mp3")

        print(RG_RC)

        while (ON4):
            CTRS=LRV.escuchar(SUJETO)
            print(CTRS)
            long_Ctr=len(CTRS)
            if long_Ctr==2:
                ON4=False
            else:
                print("Número de palabras incorrectas")
                                                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_VOZ/NUM_CONTR.mp3")

        if FUNCION=="Registrar":
            R_CTR.append(CTRS[0])
            R_CTR.append(CTRS[1])
            print(R_CTR)

np.savetxt("/home/katrina/RE_FAC_VOZ/RecoFacial/Contr_voz/"+SUJETO+".txt",R_CTR,fmt="%
s")
        elif FUNCION=="Reconocer":
            while (ON3):
                with open("/home/katrina/RE_FAC_VOZ/RecoFacial/Contr_voz/"+SUJETO+".txt") as
ctr:
                R_CTR2 = ctr.read()
```

```
            R_CTR2=R_CTR2.split()
            print(R_CTR2)
            if CTRS[0]==R_CTR2[0] and CTRS[1]==R_CTR2[1]:
               print("Contraseña correcta")
               ON3=False
            else:
               if conctr <1:
                  conctr+=1
                  print("Contraseña incorrecta")
                  print("Puede intentarlo una vez mas")
                                                                  os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_VOZ/Un_intento_mas.mp3")
                  while (ON5):
                     CTRS=LRV.escuchar(SUJETO)
                     print(CTRS)
                     long_Ctr=len(CTRS)
                     if long_Ctr== 2:
                        ON5=False
                     else:
                        print("Número de palabras incorrectas")
                                                                  os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_VOZ/NUM_CONTR.mp3")
               else:
                                                                  os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_VOZ/Contr_Inc_FIN.mp3")
                  exit()

         VALORESF=av.Analisisv() #Realiza Transformada de Fourier para obtener frecuencias
significativas
      print(VALORESF)


      for x in range(RG_RC):
         ON2=True
         if x>=1:
```

```
            LRV.escuchar(SUJETO)
            VALORESF=av.Analisisv()
            print(VALORESF)
          else:
            print("Primer ingreso")
          while (ON2):
            if VALORESF[0]==VALORESF[1]:
              LRV.escuchar(SUJETO)
                    VALORESF=av.Analisisv() #Realiza Transformada de Fourier para obtener
frecuencias significativas
              print(VALORESF)
            else:
              ON2=False
          REG_A1=REG_A1+VALORESF[0]
          REG_A2=REG_A2+VALORESF[1]
        #REGISTRAR PROMEDIO DE MAXIMOS Y MINIMOS
        if RG_RC==3:
          REG_A.append(REG_A1/3)
          REG_A.append(REG_A2/3)
          print(REG_A)

np.savetxt("/home/katrina/RE_FAC_VOZ/RecoFacial/Registro_voz/"+SUJETO+".txt",REG_A,fmt
="%d")

        elif RG_RC==0:
          REC_A=VALORESF[2]

REC_B=np.loadtxt("/home/katrina/RE_FAC_VOZ/RecoFacial/Registro_voz/"+SUJETO+".txt")
          print(REC_A,REC_B)
          if REC_A <=REC_B[0] and REC_A >=REC_B[1]:
                COM_ACCESS_OK= "Contraseña y patrón de voz correctos, acceso concedido
"+SUJETO
            print("Acceso concedido "+SUJETO)
            ACCESS_OK = gTTS(text=COM_ACCESS_OK, lang=idioma,slow=False)
```

```
ACCESS_OK.save("/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_VOZ/RESPUESTA_VAR/ACCESS_OK.mp3")
                                                                os.system("mpg123 /home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_VOZ/RESPUESTA_VAR/ACCESS_OK.mp3")
        else:
            print("Acceso denegado usted no es "+SUJETO)
                                                                os.system("mpg123 /home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_VOZ/RESPUESTA_VAR/ACCESS_NO.mp3")
```

ANEXO 7

M_LEC_REC_VOZ

```
#LECTURA Y GRABACION DE VOZ
import speech_recognition as sr
from gtts import gTTS
import gc
import os
import numpy as np
import time

#voz1="Un momento hasta filtrar el ruido de fondo"
#voz2="Acérquese al micrófono y pronuncie sus 2 palabras clave con claridad"
idioma="es"

r = sr.Recognizer()
mic = sr.Microphone()
def escuchar(SUJETO):
   with mic as source:
      print("Separando Ruido")
                                                                os.system("mpg123 /home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_VOZ/FILT_FONDO.mp3")
```

```
                                                    os.system("mpg123
/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_VOZ/PRON_CLV.mp3")
        r.adjust_for_ambient_noise(source, duration=9)
        print("Puede hablar")
        audio = r.listen(source)
        try:
          Contr_audio =r.recognize_google(audio, language="es-EC")
          Contr_audio = Contr_audio.lower()
          Contr_audio = Contr_audio.split()
          #print(Contr_audio[0],"Y",Contr_audio[1])
        except LookupError:
          Contr_audio = "Baja conexion de red"

                                                                    with
open("/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_VOZ/ANALISIS/MUESTREO.wav
", "wb") as file:
        file.write(audio.get_wav_data())
        print('Audio listo para procesar')
      return Contr_audio
```

ANEXO 8

M_ANALISIS_FTT_VOZ

```
  # ANALISIS DE VOZ POR FFT
  def Analisisv():
    from IPython.display import Image

    from pydub.playback import play
    from pydub import AudioSegment

    import scipy.fftpack as fourier
    import scipy.io.wavfile as waves
    import scipy.signal
    import scipy.fft
    import M_LEC_REC_VOZ
```

```
        import numpy as np
        import matplotlib.pyplot as plt
        import os

        Nummax=3
        Nummax2=3
        ConPosm=0
        i=0

        MtzFA=np.empty((3,2))
        Posmls=[]
        LSTF=[]


Audio='/home/katrina/RE_FAC_VOZ/AUDIO/RECONOCIMIENTO_VOZ/ANALISIS/MUESTREO.wa
v'    #Lectura de audio tomado        # Origen del audio
        #song=AudioSegment.from_wav(Audio)                  #Lectura de audio wav
        #play(song)                                  #Reproduccion de audio wav
        Fs, data = waves.read(Audio)                       # Análisis del archivo de audio

        Audio_t = data[:]
        L = len(Audio_t)                           # Longitud del paquete de datos
        n = np.arange(0,L)/Fs                        # Vector de tiempo vs longitud de la señal
        FFT = fourier.fft(Audio_t)                  # Transformada de Fourier
        MFFT = abs(FFT)                           # Magnitud de la FFT
         MFFT = MFFT[0:L//2]                          #  Por la simetría de la transformada tomamos las
mitad de los datos
    #PASA   BANDA#h=scipy.signal.firwin(L,   (fc1,   fc2),   window="hamming",   pass_zero=False,
fs=MFFT)
        ANGFFT = np.angle(FFT)
        F = Fs*np.arange(0, L//2)/L
    #Determina los 3 valores mas altos
        PMFFT=np.argsort(MFFT)
        SMFFT=MFFT[PMFFT]
```

```
Act=True
while (Act):
   maxval = SMFFT[-Nummax2:]
   for n in maxval:
      Posm = np.where(MFFT==n)          # Posicion de la magnitud maxima
      F_fund = F[Posm]
      if F_fund > 62:
         Posmls.append(Posm)
         ConPosm+=1
         if ConPosm >= 3:
            Act=False
            break
      else:
         Nummax2+=1

redo=np.round(maxval,2)
for h in Posmls:
   F_fund = F[h]
   Frec = np.round(F_fund,2)
   M_gk_fund = MFFT[Posm]
   Ampl = np.format_float_scientific(M_gk_fund,2)
   for j in range (2):
      if j==0:
         MtzFA[i][j]=int(Frec)
         LSTF.append(int(Frec))
      elif j==1:
         MtzFA[i][j]=Ampl
         i+=1
#print(MtzFA)
MAXF=np.max(LSTF)
MINF=np.min(LSTF)
PROMF=(MAXF+MINF)/2
#plt.plot(F, MFFT)
#plt.xlabel('Frecuencia (Hz)', fontsize='14')
#plt.ylabel('Amplitud FFT', fontsize='14')
```

```
        #plt.show()

        return [MAXF,MINF,PROMF]

        #print(MAXF,MINF,PROMF)

#np.savetxt("/home/katrina/RecoFacial/Registro_voz/Amp_Frec_USR.txt",MtzFA,fmt="%d")
```

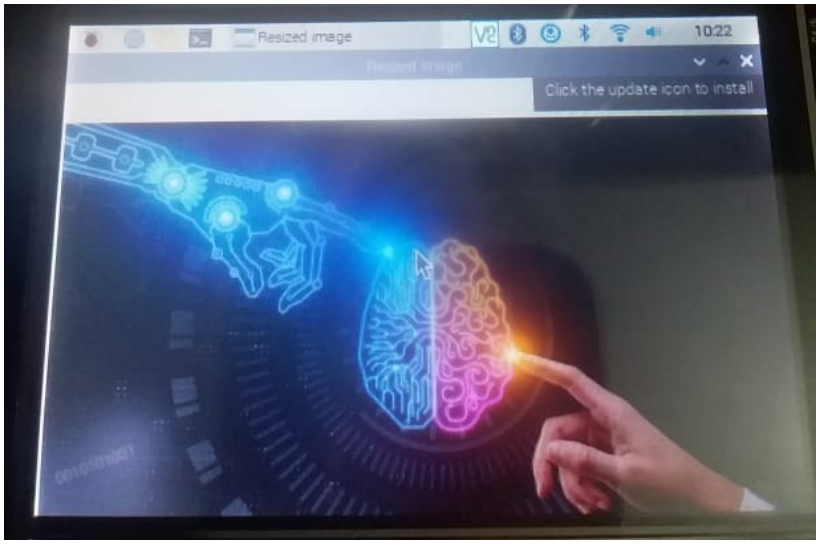
ANEXO 5

Registro fotográfico

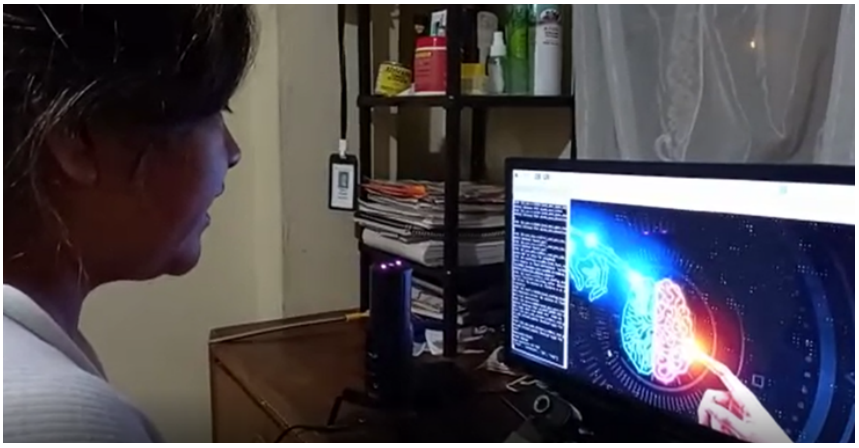
Sistema con sus dispositivos conectados

Compuertas de acceso a pines de salida y tarjeta micro SD

Pantalla mini LCD en modo de inicio



Verónica Sailema – Usuario 1

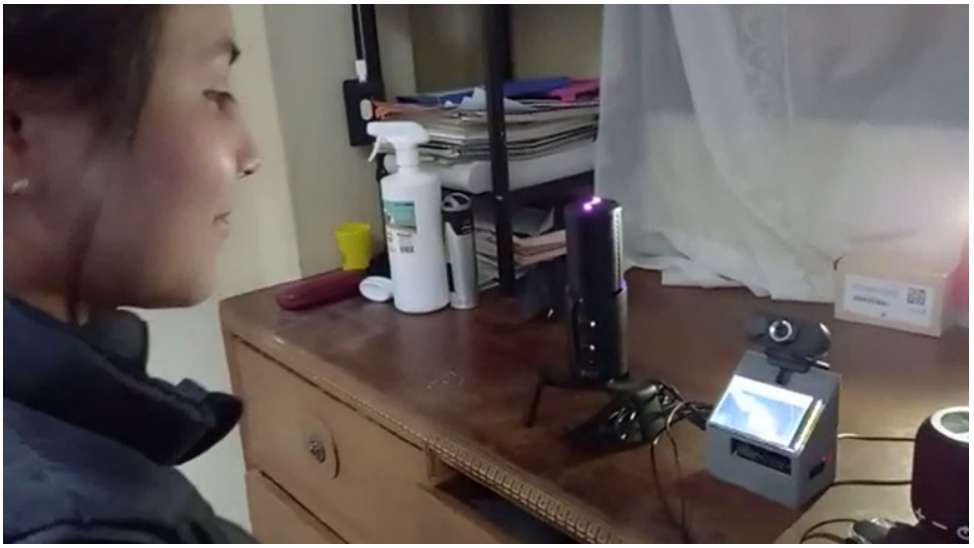


Samantha Escobar – Usuario 2

Edisson Escobar – Usuario 3, Administrador

Luis Sailema – Usuario 4

Doménica Sailema – Usuario 5



Mayra Sailema – Usuario 6

ANEXO 6

Manual de usuario

MANUAL DE USUARIO

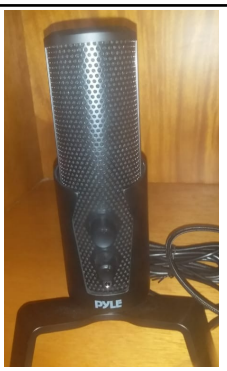
SISTEMA DE CONTROL DE ACCESO POR RECONOCIMIENTO FACIAL Y COMANDO DE VOZ EN PYTHON

Autor: Edisson Escobar

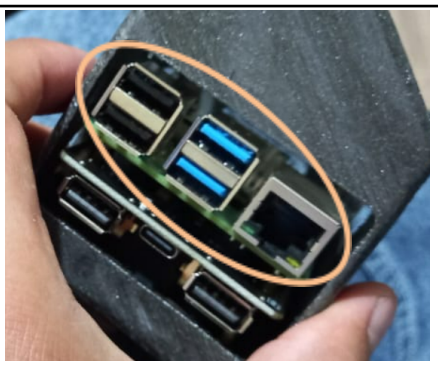
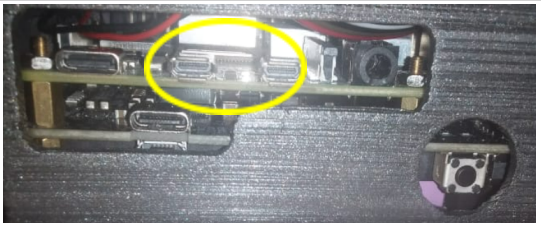
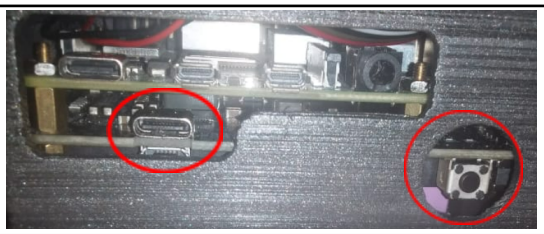
Contenido

Elementos del Sistema

El sistema está conformado por los siguientes elementos:

Micrófono Cardioide Pyle	
Altavoz bluetooth	
Cámara articulada empotrada a la carcasa	 
Raspberry PI 4B 4GB integrada en la carcasa	
UPS Plus con dos baterías 16850 capacidad 1h	 

Puertos para dispositivos

Dos puertos USB 2.0, dos puertos 3.0, un puerto Ethernet, wifi, bluetooth.	
Dos puertos mini-hdmi.	
Puerto de carga y botón de encendido del sistema.	
Puertos de carga de dispositivos auxiliares	

Preparación del sistema

Previo al uso del sistema es necesario comprobar los siguientes puntos:

Paso 1

La cámara y el micrófono se encuentran conectados a los puertos USB 3.0 (color azul), como se observa en la Figura 1.

Figura 1

Conexión de micrófono y cámara

Paso 2

Encender el dispositivo presionando el botón de encendido como se ve en la Figura 2.

Figura 2

Botón de encendido

Paso 3

Una vez arrancado el dispositivo encender el altavoz, al encontrarse encendido el altavoz destella una luz azul como se observa en la Figura 3.

Figura 3

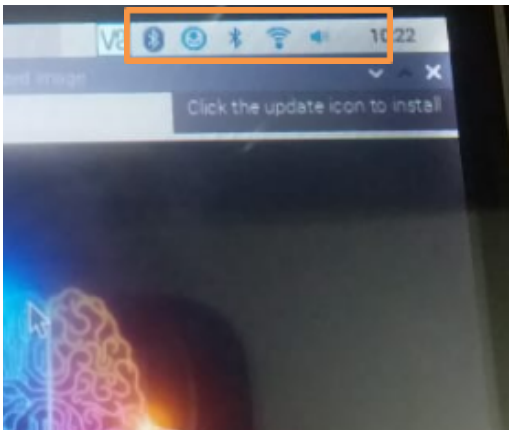
Encendido de altavoz

Paso 4

Configurar la red inalámbrica o conectarse al puerto ethernet de ser el caso. Adicional vincular el altavoz por vía bluetooth, esto se realiza atreves de la pantalla mini LCD como se observa en la Figura 4.

Figura 4

Pantalla mini LCD



Paso 5

Seleccionar el modo estéreo en el micrófono, esto se realiza presionando el botón de la parte superior hasta que se encienda la luz frontal y posterior, como se observa en la Figura 5.

Figura 5

Modo estéreo

Paso 6

Una vez desplegada la imagen de la Figura 6 en la pantalla, el sistema se encuentra habilitado para su uso.

Figura 6

Sistema habilitado

Nota: *El paso 4 se debe realizar solo cuando se usa por primera vez, posterior a eso el sistema guarda la contraseña y el dispositivo.*

Modo de Uso

El nombre del sistema es “Katrina”, responde a los comandos cuyas frases integren este nombre, se encuentra en continua escucha hasta que se pronuncie una de las siguientes instrucciones y posterior a culminar la instrucción retorna nuevamente a su modo de escucha:

Comandos de activación de funciones

El orden de las palabras puede cambiar, mas es necesario que éstas integren la frase que el usuario pronuncie.

Nombre del sistema. Katrina.

Descripción del producto. “Katrina activa la descripción”.

Inicializa la descripción audible del sistema, esto como un modo de presentación del mismo.

Registro de un nuevo usuario. “Katrina activa el registro”.

Ejecuta el modo de registro de usuarios.

Nota: *Para poder iniciar el registro de usuarios el sistema solicita identificar facialmente al administrador.*

Reconocimiento de usuarios. “Katrina activa el reconocimiento”.

Inicializa el modo de reconocimiento de usuarios.

Saludos variados. Los saludos deben necesariamente estar acompañarlos con el nombre del sistema para que este lo reconozca.

Hola, buenos días, buenas tardes, buenas noches, como estas, aló, como esta, un gusto etc.

Registro de usuarios

Para registrar un nuevo usuario, es necesario que el sistema detecte en primera instancia la presencia del **“administrador”**, esto para que exista un control de los usuarios en el registro.

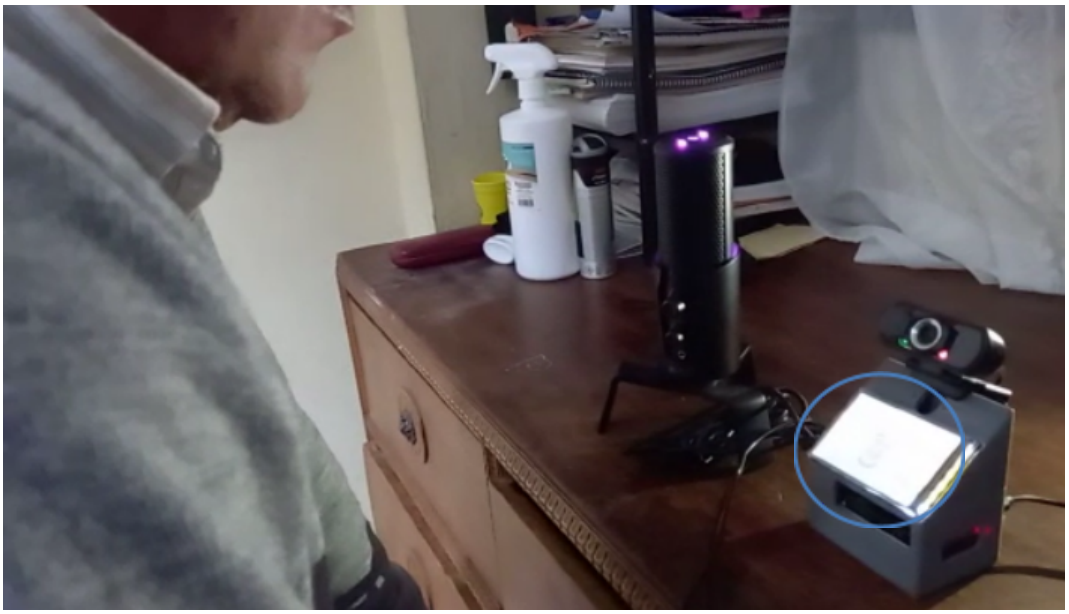
El modo de registro inicia con el comando “Katrina activa el registro”, puede variar el orden de las palabras, pero la frase debe estar integrada con las mismas.

Primera etapa del registro. El sistema solicitará colocarse frente a la cámara y pronunciar un nombre y un apellido, esto para proporcionar una etiqueta específica al usuario.

Como se observa en la Figura 7, la imagen del rostro es visualizada en la mini LCD montada sobre la carcasa del dispositivo, para que el usuario pueda posicionarse de mejor manera frente a la cámara.

Figura 7

Solicitud de un nombre y apellido



El sistema comprobará la pronunciación de dos palabras, si se pronuncian menos o más de dos el sistema advertirá esto y las solicitara nuevamente. Este proceso se realizará hasta contar con el número de palabras correctas.

La existencia del nombre y apellido serán comprobados en la base de datos, en el caso de que exista otro usuario llamado de la misma forma el sistema informará que el usuario se encuentra registrado y le solicitará cambiar su nombre y apellido por otro.

En el caso de no existir otro usuario con el mismo nombre se procederá a crear un archivo dentro del cual se guardarán los datos que serán recolectados posteriormente.

Nota: *La pronunciación del nombre y apellido deben ser correctas ya que será con este nombre con el cual el sistema lo identificará, en el caso de no pronunciar correctamente su nombre la edición del mismo solo se podrá realizar por el administrador.*

Segunda etapa del registro. En esta se realizará la toma de imágenes, se tomarán 60 imágenes del rostro, solicitando variedad en los gestos, esto ayuda a mejorar el reconocimiento del usuario. Se indicará realizar movimientos del rostro hacia la derecha, izquierda, arriba y abajo, esto crea una base de datos variada como se puede observar en la Figura 8.

Figura 8

Toma de imágenes variadas

Finalizada la toma de imágenes el sistema informará si se realizaron exitosamente.

Nota: *En el caso de que alguno de estos movimientos posicione al rostro fuera del campo visual de la cámara, se informará que retorne el rostro ligeramente hacia la cámara.*

Tercera etapa del registro. Luego de realizar la toma de imágenes prosigue el registro de la muestra de voz, el sistema solicita una clave de dos palabras, de las cuales se extraerán dos caracteres, la contraseña y el rango de frecuencia. No es necesario acercarse al micrófono, el sistema puede captar la voz a la distancia en la cual el rostro fue reconocido como se observa en la Figura 9.

Figura 9

Pronunciación de contraseñas

El sistema solicitará las claves por tres ocasiones, por lo tanto, se deben pronunciar exactamente igual las tres veces.

Reconocimiento de usuarios

El sistema accede al registro de usuarios por medio de la frase “Katrina activa el reconocimiento”, puede ser activado por cualquier usuario, el sistema de reconocimiento cuenta con las siguientes etapas:

Primera etapa de reconocimiento. El sistema solicita al usuario se posicione frente a la cámara, si se lo reconoce inmediatamente se pronunciará el nombre del usuario reconocido, en el caso de que no se reconoce el sistema intentara identificarlo por 10 ocasiones, de no encontrar coincidencia en estos intentos el sistema lo anunciara como usuario desconocido y regresará al modo de escucha, como se observa en la Figura 10.

Figura 10

Reconocimiento facial

Segunda etapa de reconocimiento. Luego de reconocer al usuario como uno registrado, prosigue la toma del patrón de voz, se le recuerda que debe conocer las dos claves registradas, al iniciar el sistema de reconocimiento de voz se desplegará en la pantalla un icono indicando que se está ejecutando como se visualiza en la Figura 11.

Figura 11

Inicia Reconocimiento de voz

En el caso de pronunciar menos o más de dos claves el sistema le pedirá ingresar nuevamente el número correcto de palabras y en el caso de pronunciar mal la palabra sólo le permitirá un total de dos intentos.

Si la contraseña es correcta, se procede a analizar el espectro de la voz, si este coincide con el registro se brinda el acceso, caso contrario lo rechaza automáticamente sin intentos adicionales.