UNIVERSIDAD ESTATAL PENÍNSULA DE SANTA ELENA

FACULTAD DE SISTEMAS Y TELECOMUNICACIONES

CARRERA DE TELECOMUNICACIONES

TRABAJO DE INTEGRACIÓN CURRICULAR

Previo a la obtención del título de

INGENIERO EN TELECOMUNICACIONES

Implementar algoritmos de alto nivel para la detección de comportamientos sospechosos y reconocimiento facial

AUTOR

Joan Steven Baño Sandoval

CARRERA

Telecomunicaciones

PROFESOR TUTOR

Ing. Luis Miguel Amaya Fariño, MSc.

LA LIBERTAD - ECUADOR

2024-2

AGRADECIMIENTO

Quiero dedicar estas líneas a expresar mi más sincero agradecimiento a las personas que han sido fundamentales para la culminación de esta tesis.

A mis padres y a mi hermano por su apoyo incondicional desde el principio de la carrera. Gracias por sus sacrificios, las incontables palabras de apoyo que me motivaron en cada paso que di en este camino. A Emely por su inquebrantable apoyo moral, paciencia y comprensión que me han apoyado en la culminación de este proyecto y me ha ayudado en todo lo posible.

DECLARACIÓN AUTORÍA DEL ESTUDIANTE

El presente trabajo de Integración Curricular, con el titulo **"Implementar algoritmos de alto nivel para la detección de comportamientos sospechosos y reconocimiento facial",** declaro que la concepción, análisis y resultados son originales y aportan a la actividad educativa en el área de Telecomunicaciones

Atentamente,

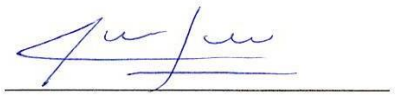
Baño Sandoval Joan Steven

C.I. 1803634086

DECLARATORIA DE RESPONSABILIDAD

Quien suscribe, Baño Sandoval Joan Steven con C.I. 1803634086, estudiante de la carrera de Telecomunicaciones declaro que el Trabajo de Titulación presentado a la unidad de Integración Curricular cuyo tema es **"Implementar algoritmos de alto nivel para la detección de comportamientos sospechosos y reconocimiento facial"** corresponde y es de exclusiva responsabilidad de los autores y pertenece al patrimonio intelectual de la Universidad Estatal Península de Santa Elena.

Atentamente,



Baño Sandoval Joan Steven

C.I. 1803634086

DECLARACIÓN DE DOCENTE ESPACIALISTA

En mi calidad de Docente especialista del trabajo de Integración Curricular, **"Implementar algoritmos de alto nivel para la detección de comportamientos sospechosos y reconocimiento facial"**, elaborado por **Baño Sandoval Joan Steven**, estudiante de la Carrera de Telecomunicaciones, Facultad de Sistemas y Telecomunicaciones de la Universidad Estatal Península de Santa Elena, previo a la obtención del título de Ingeniero en Telecomunicaciones, me permito declarar que tras supervisar el desarrollo y estructura final del trabajo, este cumple y se ajusta a los estándares académicos. En consecuencia, lo considero apto en todos los aspectos y listo para la sustentación del trabajo.

Atentamente

ING. FERNANDO CHAMBA MACAS, MGTR.

DOCENTE ESPECIALISTA

DECLARACIÓN DE DOCENTE TUTOR

En mi calidad de Docente Tutor de integración Curricular, **"Implementar algoritmos de alto nivel para la detección de comportamientos sospechosos y reconocimiento facial"**, elaborado por **Baño Sandoval Joan Steven**, estudiante de la Carrera de Telecomunicaciones, Facultad de Sistemas y Telecomunicaciones de la Universidad Estatal Península de Santa Elena, previo a la obtención del título de Ingeniero en Telecomunicaciones, me permito declarar que tras supervisar el desarrollo y estructura final del trabajo, este cumple y se ajusta a los estándares académicos. En consecuencia, lo considero apto en todos los aspectos y listo para ser evaluado por el docente especialista.

Atentamente

ING. LUIS MIGUEL AMAYA FARIÑO, MGTR.

DOCENTE TUTOR

TRIBUNAL DE SUSTENTACIÓN

Ing. Ronald Rovira Jurado, Ph.D.

DIRECTOR DE LA CARRERA

Ing. Fernando Chamba Macas, Mgtr.

DOCENTE ESPECIALISTA

Ing. Luis Amaya Fariño, Mgtr.

DOCENTE TUTOR GUÍA

Ing. Corina Gonzabay De la A, Mgtr.

SECRETARIA

RESUMEN

Los sistemas de seguridad y monitoreo de movimientos en entornos controlados han mejorado confirme las necesidades de los usuarios, con el robot se propone utilizar un modelo de aprendizaje con varias herramientas de reconocimiento facial y de postura. Los componentes del robot y herramientas como MediaPipe para obtener mayor precisión en cuestión de percepción y captura de imagen así mismo como su procesamiento. Con las capacidades de visualización del robot junto con la correcta detección y reconocimiento facial y de movimientos generan resultados más que satisfactorios para este proyecto.

La aplicación de MediaPipe y Opencv generan una correcta detección de rostros y de movimientos, así poder identificar patrones dentro de la detección junto con los sensores del robot Rosmaster X3 proporcionan un sistema efectivo de detección y autentificación de rostros y movimientos dentro de un entorno controlado.

Este proyecto tiene el enfoque dirigido a la seguridad y control de un entorno con modelos de aprendizaje, así como el poder identificar y verificar la estabilidad del mismo. Esto se aplicará en un futuro cuando se demuestre su funcionamiento en la Facultad de Sistemas y Telecomunicaciones de la Universidad Estatal Península de Santa Elena.

Palabras claves: tecnologías digitales, seguridad, control, detección, reconocimiento fácil.

ABSTRACT

Security and movement monitoring systems in controlled environments have improved confirm the needs of users, with the robot it is proposed to use a learning model with several facial and posture recognition tools. The robot components and tools such as MediaPipe to obtain greater precision in terms of perception and image capture as well as their processing. With the visualization capabilities of the robot together with the correct facial and movement detection and recognition, they generate more than satisfactory results for this project.

The MediaPipe and Opencv application generate correct detection of faces and movements, thus being able to identify patterns within the detection together with the sensors of the Rosmaster X3 robot, providing an effective system for detecting and authenticating faces and movements within a controlled environment.

This project has a focus aimed at the security and control of an environment with learning models, as well as being able to identify and verify its stability. This will be applied in the future when its operation is demonstrated in the Faculty of Systems and Telecommunications of the Santa Elena Peninsula State University.

Keywords: digital technologies, security, control, detection, easy recognition.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Introducción

Los modelos de machine learning nació desde la necesidad de clasificar datos de forma eficiente y rápida, para el cual los modelos se desarrollan en diferentes etapas, como primer punto evaluamos el problema y conforme a eso escribimos reglas explicitas para resolver el mismo, evaluamos el modelo ya establecido con las reglas y analizamos sus errores para su lanzamiento.

Según los conceptos ya mencionados se intentan desarrollar modelos de aprendizaje según las necesidades del usuario. En este caso usamos herramientas como MediaPipe de manera conjunta con modelo de aprendizaje para elaborar un sistema sofisticado dirigido en el apartado de autentificación y reconocimiento en movimientos específicos.

El robot Rosmaster X3 en conjunto con la placa jetson nano tiene la facilidad de generar una plataforma con una interfaz intuitiva para usar modelos de aprendizaje ya sea de machine learning como de Deep learning. También se enfoca en la aplicación práctica dentro de la Facultad de Sistemas y Telecomunicaciones de la Universidad Estatal Península de Santa Elena, con este enfoque se enfatizará en el seguridad y control en un ambiente específico.

CAPITULO 1

1.1 GENERALIDADES DEL PROYECTO

1.2 Planteamiento del Proyecto

Desde que las primeras computadoras fueron fabricadas se tuvo la idea de si pueden llegar a tener la capacidad de aprender, al inicio se implementó sistemas clasificación de datos como por ejemplo el apartado de spam de los correos electrónicos donde de forma autónoma se clasifica los correos. Este tipo de clasificación se comenzó a dominar inteligencia artificial.

Para que una computadora tenga comportamientos inteligentes debería ser capaz de resolver problemas con forme a una experiencia, con esta base se realiza la recolección y modelado de los datos para poder tener resultados factibles y efectivos. Para esto se desarrolló algoritmos de alto nivel de machine learning, donde usamos algoritmos supervisados ya que elaboramos etiquetas con los datos recolectados y corregir errores que se detecten en el modelo de aprendizaje y alcanzar un nivel alto en precisión.

La investigación actual tratamos los datos adquiridos y lo procesamos para el entrenamiento de nuestro sistema de reconocimiento de manera que logramos un nivel de predicción y clasificación eficiente.

1.3 Antecedentes

Los sistemas de mejora de seguridad han ido mejorando conforme van evolucionando para asegurar la confiabilidad de los usuarios, se ha convertido en el estudio más importante para la seguridad de la red de telecomunicaciones (Jaramillo, 2021). A pesar de la importancia que tienen los datos para los usuarios existen muchos delitos informáticos y daños a las redes de telecomunicaciones, como puede ser el desfalco de información o la alteración de la información de los usuarios (Muñoz, 2021).

Para moderar estos riesgos se debe realizar un análisis de la infraestructura de la red de telecomunicaciones, preparando una prueba en la infraestructura con un método de análisis preventivo en contra de la penetración de nuestro sistema, con esta técnica se busca fallos en la seguridad de la red de telecomunicaciones, de los cuales se clasifican según el nivel de amenaza para el usuario y las consecuencias que pueden existir (Jaramillo, 2021).

La inteligencia artificial ha permitido un avance y sofisticación a la detección, reconocimiento y seguimiento de objetos, estos principios se les aplica en un sinnúmero de aplicaciones como lo es el reconocimiento facial[1], dichas aplicaciones se usan para facilitar la tarea de el control de acceso en sinnúmero de empresas tanto en el sector privado como en el sector público.

La inteligencia artificial en cuanto a acceso y detección de comportamientos sospechosos es efectiva, tiene mucha notoriedad para disminuir los errores en detección de objetos y rostros, tiene una aplicación efectiva para el acceso de personal autorizado en una oficina privada o cualquier entorno que necesite una seguridad de acceso.

Para la detección de rostros y movimientos se procesa los fotogramas captados en tiempo real, donde los mismos son almacenados para el procesamiento y reconociendo efectivo de los rostros, así comparando los fotogramas mejorando el nivel de efectividad del sistema de control de acceso (Flores, Fernando, Mariño, & Fabricio, 2022).

1.4 Descripción del Proyecto

En este proyecto se empleará un método de control de acceso utilizando técnicas de programación avanzada como lo es reconocimiento facial, con esto se autentifica los usuarios aceptados, también opera un código de detección de movimientos sospechosos

para clasificar el nivel de amenaza que puede llegar a existir, todo esto se implementara en el laboratorio de telecomunicaciones.

Figura 1

Sistema de acceso en el laboratorio de telecomunicaciones.

Fuente: Elaborado por el autor.

1.5 Objetivos

1.5.1 Objetivo General

Implementar un algoritmo basado en reconocimiento facial y detección de movimientos para autentificar usuarios y detectar movimientos sospechosos en el laboratorio de telecomunicaciones.

1.5.2 Objetivos específicos

- Emplear un sistema de control de acceso que use técnicas de reconocimiento facial para autentificar a usuarios.
- Desarrollar un algoritmo de detección de movimiento así determinar comportamientos sospechosos.
- Determinar la eficiencia en la autentificación de usuarios, relacionando las pruebas que se empleen en el sistema de

acceso.

- Emplear el programa de reconocimiento facial y detección de movimiento sospechosos para el aprendizaje.

1.6 Justificación

La seguridad en las redes de telecomunicaciones es un aspecto crucial en la actualidad, especialmente debido al creciente uso de tecnologías digitales y la transmisión de información confidencial a través de estas redes. Los sistemas de control de acceso y seguridad son fundamentales para garantizar la confidencialidad, integridad y disponibilidad de la información transmitida, así como para proteger los recursos y activos de las organizaciones.

Los sistemas de control de acceso en dispositivos electrónicos se usan con mayor regularidad gracias a la evolución de la tecnología, dentro de los cuales los sistemas de reconocimiento facial es una técnicas más fiables y usadas en la actualidad para el reconocimiento personas, con esta tecnología se obtiene buenos resultados en varias empresas que lo usan, pero la eficiencia del mismo está ligado a la tecnología e instrumentos que estén usando (Roca, 2022).

Cuando hablamos de reconocimiento facial la técnica más común y fácil en cuanto a esta tecnología es la identificación frontal de la cara de la persona en cuestión, que es una técnica muy usada en pequeñas empresas donde no amerita un alto nivel de seguridad, pero si un buen sistema de control de acceso y salida de personal (Zambrano, Orlando, & Lara, 2020).

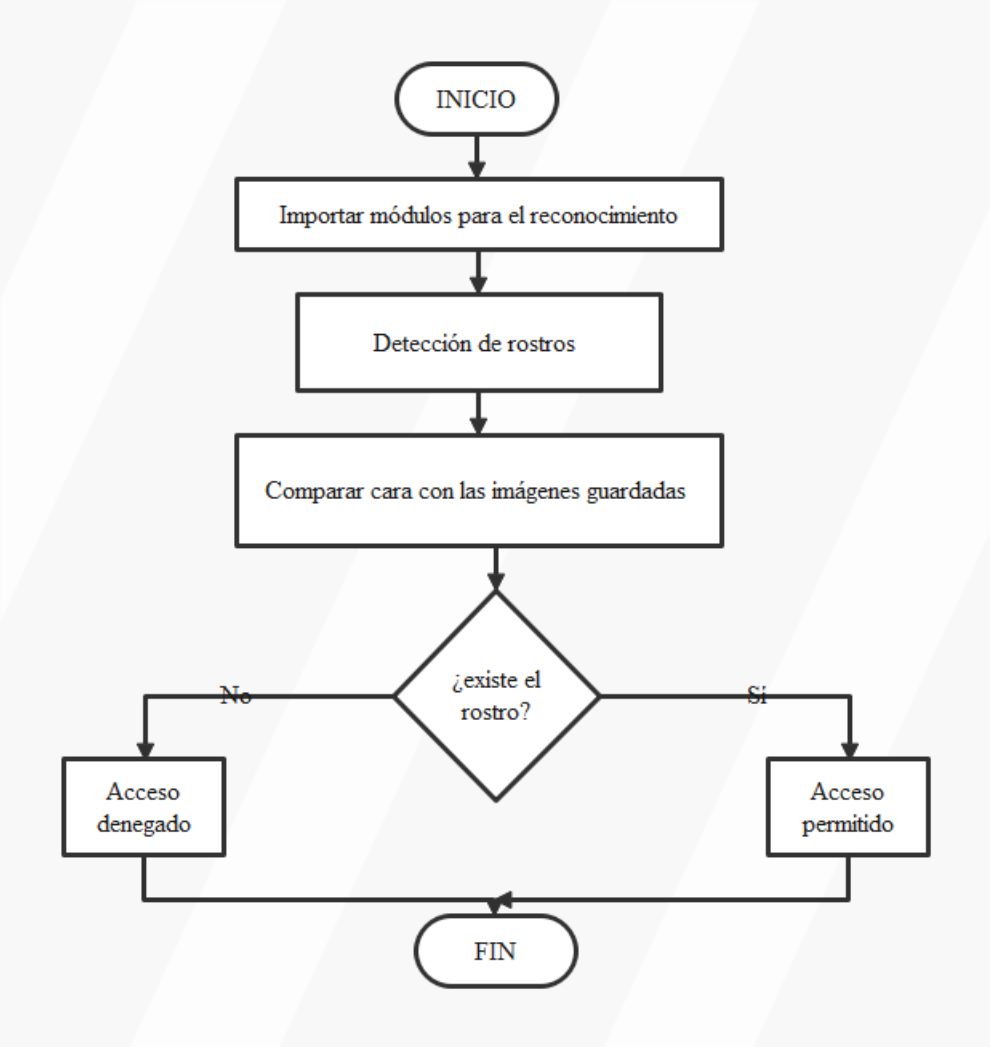
Para optimizar el sistema de control de acceso en el laboratorio de telecomunicaciones se ha propuesto implementar la tecnología de reconocimiento facial

y movimiento de objetos para mejorar la seguridad interna, donde se usa algoritmos de alto nivel de MediaPipe mitigando riesgos y posibles amenazas (Zhang, 2022).

El algoritmo de detección facial en tiempo real que se va a implementar tiene como inicio capturar imagen del rostro y luego comparar con imágenes anteriores (Singh & Neelima, 2022), si no existe un rastro de coincidencia denegar el acceso y si es el caso contrario conceder el acceso como se puede observar en la figura 2.

Figura 2

Diagrama de flujo de detección facial.



Fuente: Elaborado por el autor.

MediaPipe nos ofrece varias funciones de detección facial una de ellas es FaceMesh, esta función nos permite detectar movimientos faciales, aunque la detección

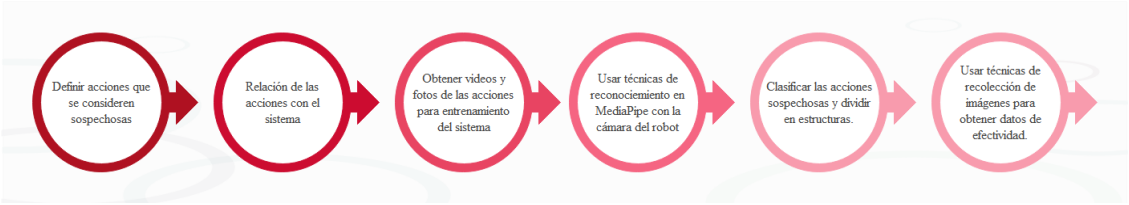
de rostros es un proceso complicado por la variabilidad que cambia el rostro humano o la iluminación del entorno, pero a pesar de estar variables FaceMesh tiene buenos resultados (Ryan León, Carranza, & Lozano, 2022).

Otra de las funciones es MediaPipe Face Detection donde no tiene el nivel de detección de movimientos faciales, solo se concentra a la detección de donde se encuentra el rostro de dicha persona y la enmarca con un recuadro alrededor de la cara del objetivo, es una manera más fácil pero más deficiente de la detección facial (Abolmaali, Hafeez, Mohsin, Hassan, & Thakur, 2021).

Para el apartado de reconocimiento de actitudes sospechosas se describe diferentes puntos a seguir para la implementación de técnicas de reconocimiento mediante MediaPipe para detectar un comportamiento que puede perjudicar la seguridad del laboratorio de telecomunicaciones (Cedano, 2020), como lo vemos en la figura 3.

Figura 3

Comportamientos sospechosos.



Fuente: Elaborado por el autor.

En la clasificación de comportamientos sospechosos se divide en dos niveles de amenazas que podría afectar al laboratorio, como primer aspecto tenemos la extracción de equipos del laboratorio y como segundo aspecto es el daño que podrían hacer a los equipos del laboratorio.

Para la predicción de este nivel de amenaza nos enfocamos en las funciones que nos ofrece MediaPipe, una de esta es MediaPipe pose que recolecta imágenes en 2D y hace una estimación de movimientos humanos, tiene limitaciones como lo son los detalles

del eje, pero aun así es capaz de comprender la estructura del cuerpo humano (Quiñonez, Lizarraga, & Aguayo, 2022), otra función de MediaPipe es MediaPipe Holistic donde está a diferencia de la otra función nos ofrece tener una percepción facial de manera simultánea donde se puede monitorear los movimientos faciales, tiene mayor complejidad por la cantidad de datos que recolecta en tiempo real (Ciudad Real Villaescusa, 2022).

1.7 Alcance del proyecto

En el presente proyecto tiene como finalidad la implementación de sistema de control de acceso y seguridad en el laboratorio de telecomunicaciones.

Con el uso de algoritmos de MediaPipe se podrá identificar y detectar personas mediante técnicas de reconocimiento facial, para esto se usa información de artículos y tesis ya realizada. Una vez construido el código se lo implementa en el laboratorio para corregir detalles, evaluando la precisión y eficiencia del mismo, así validando el sistema en entornos reales y demostrando su efectividad.

Al culminar con la implementación se procede a demostrar los resultados y el nivel de eficiencia obtenido, así mismo como las conclusiones y recomendaciones.

1.8 Metodología

- Investigación documental

Tomando como base los artículos y resultados obtenidos en algoritmos de detección facial y detección de movimientos, beneficiar el desarrollo de los algoritmos y comandos a usar en nuestro sistema de control de acceso.

- Investigación aplicada

El proyecto demuestra los comportamientos de los algoritmos de MediaPipe para los conocimientos de control y acceso en entornos reales.

1.9 Resultados esperados

Los resultados esperados para el desarrollo del proyecto son los siguientes:

- Implementar técnicas de reconocimiento facial para el desarrollo de un sistema de control de acceso.
- Evaluar los datos obtenidos para medir la eficacia del sistema.
- Implementar técnicas de movimientos de objetos para la detección de comportamientos no autorizados en el laboratorio de telecomunicaciones.
- Mostrar el sistema de manera interactiva para aprendizaje dentro del laboratorio de telecomunicaciones.

CAPITULO 2

2. MARCO TEÓRICO

Para esta investigación se tomará los temas que involucran en la implementación de algoritmos de alto nivel para la detección de movimientos sospechosos y reconocimiento facial.

2.1 Definición de reconocimiento facial

Es una nueva tecnología que recolecta datos faciales para el acceso a un lugar o un sistema, usualmente se usa en la actualidad para acceder a un sistema y la forma más común es a través de nuestro teléfono ya que la mayoría cuenta con reconocimiento facial. Cada persona tiene un patrón facial diferente y gracias a la recolección y comparación de esos datos se puede verificar a una persona (Montes, y otros, 2021).

2.2 Definición de biometría facial

La biometría facial es la técnica que más se usa para el reconocimiento facial, para usar esta técnica se necesita un dispositivo de detección facial que suele ser ligado a una tecnología digital como lo es una cámara, con esto en cuenta se toma imágenes a la persona a reconocer con diferentes poses faciales y se guardan en una base de datos.

Gracias a todo este proceso ya mencionado esta tecnología es mucho más segura que una contraseña o pin o confirmación por email ya que se usa un algoritmo único que tiene varios procesos matemáticos y la dificultad que se realiza en tiempo real que lo hacen muy eficiente (Gutiérrez, 2022).

2.3 Usos del reconocimiento facial

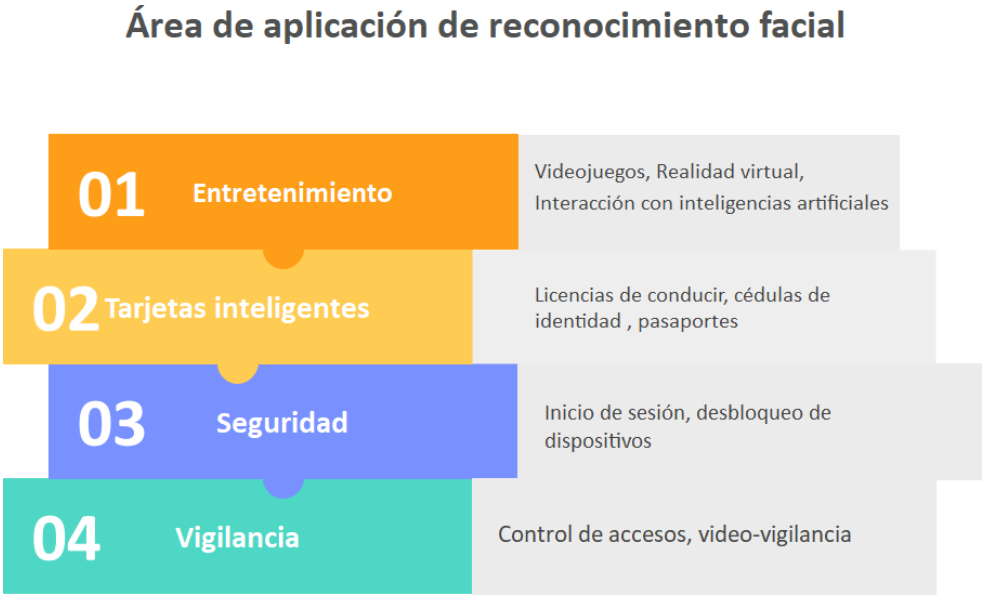
El reconocimiento facial tiene varios usos entre los cuales los que más destacan son los siguientes:

1. Para acceso a aplicaciones móviles
2. Acceso a una sala o cuarto restringido para el publico
3. Para verificación de usuario que comúnmente es método de pago
4. Proceso de verificación como puede ser los check-in

En la figura 4 podemos observar el área específica de aplicación de esta tecnología.

Figura 4

Área de aplicación de reconocimiento facial.



Fuente: Elaborado por el autor.

2.4 Beneficios de usar el reconocimiento facial

Es una tecnología que tiene un proceso muy rápido por lo cual muchas empresas como Apple han cambiado su tecnología de reconocimiento dactilar a reconocimiento facial por su eficacia y veracidad.

La experiencia que tiene el cliente con esta tecnología es más satisfactoria, al solo necesitar mirar tu rostro lo hace más cómoda para el usuario, ya que tecnologías como

reconocimiento dactilar suele ser más incomoda por la sudoración natural de las manos que hace un proceso un poco más tardado.

Tiene un porcentaje de error muy bajo sobre todo si es tiempo real, ya que al tener cada usuario características faciales únicas lo hace muy eficaz

Es muy útil para necesidades a larga distancia como por ejemplo autentificar que la llegada de un empleado a su trabajo de manera remota.

2.5 Métodos de reconocimiento facial

Para el reconocimiento facial existe varios métodos para su respectivo reconocimiento de los cuales se dividen en estas tres categorías:

1. Reconocimiento basado en imágenes 2D.
2. Reconocimiento facial 3D
3. Reconocimiento hibrido

La seguridad que nos brindan las características biométricas es de los más seguros que existen, tienen varias aplicaciones desde las militares hasta las más básicas como el FaceID que tiene la marca Apple, las características biométricas han evolucionado de manera gigantesca para aumentar su precisión ya que mientras más sensores se aplique más efectivo es el sistema incluso llega a ser más seguro que una contraseña.

2.5.1 Reconocimiento basado en imágenes 2D

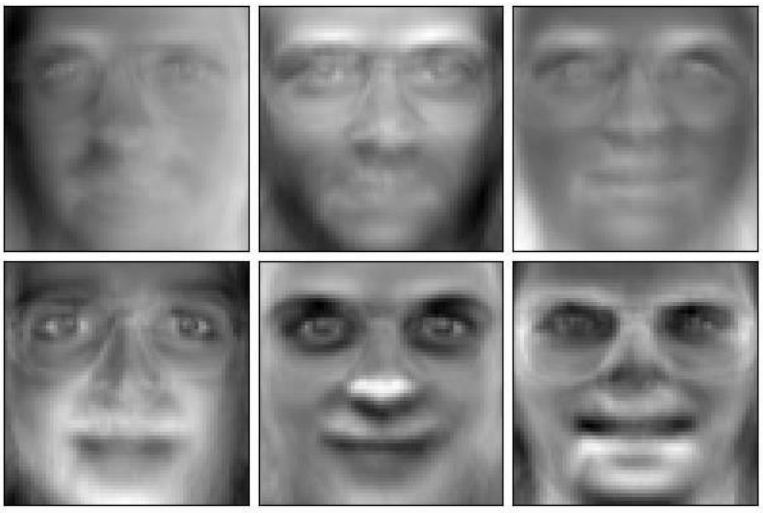
El reconocimiento facial en sus inicios fue presentado como reconocimiento de patrones en imágenes estáticas, lo cual abrió a varios métodos de reconocimiento en dos dimensiones, los cuales son por características faciales como son la posición de los ojos o la nariz o la longitud de la boca, también por geometría que es a base de métodos geométricos o por vectores.

Uno de los métodos más usados fue presentado por Kirby y Sirovich que implementa un modelo matemático basado en el análisis de componentes principales (PCA), Este modelo matemático permite simplificar la complejidad de los datos obtenidos en una muestra que en este caso toma los pixeles de NxN y crea un nuevo vector de una longitud de N^2, con las demás muestras obtenidas se obtiene el valor de un rostro medio, con esto se calcula la matriz de covarianza para encontrar los vectores y valores diferentes de 0 (Filio, 2021).

El mayor beneficio de usar el método anteriormente mencionado es la reducción de las dimensiones de las muestras ya que se toma los vectores más grandes para crear un espacio de dimensión, debido a la reducción de complejidad se toma de las características del sujeto y se compara con las del nuevo rostro y si la distancia es menor que el parámetro obtenido da como resultado que es el mismo rostro.

Figura 5

Eigenrostros calculados por PCA.



Fuente: (Gárate, Hajjam, & Andrès, 2020)

Uno de los problemas de este método es la alta tasa de error del mismo en cierto tipo de lugares que tienen poca o mucha iluminación y también la pose facial del usuario para resolver esta clase de problemas se propuso otro método que es a través del

discriminante lineal o también nombrado LDA por sus siglas en ingles (Gárate, Hajjam, & Andrès, 2020).

Otro método también usado es a través de la discriminante lineal, que este método utilizado también en estadística, reconoce patrones y aprende de manera automática en una combinación lineal. En el caso de reconocimiento facial se basa en detectar las características de la cara para maximizar sus diferencias y así reconocer diferentes rostros.

Figura 6

Rostros calculados por LDA

Fuente: (Mohammed, Abbas, & Abood, 2023).

Otra solución para estas problemáticas son las redes neuronales donde se enfocan en un modelo de aprendizaje donde según las probabilidades se toma las decisiones, por lo general se basa en tres módulos que son detector de cara, localizador de rostro humano y localizador de ojos y compararlos de una base de datos previamente creada (Mohammed, Abbas, & Abood, 2023).

2.5.2 Reconocimiento basado en imágenes 3D

En el caso del reconocimiento facial en 3D tienes menos tasa de error y no tiene el problema de variación de pose ya que se toma la imagen en 3 puntos establecidos que son x, y, z donde el punto z representa la profundidad dándole más valor a los pixeles que están x, y. También una imagen se toma en el rango de 2.5D Donde se toma la imagen en

un solo punto de vista y se modela la superficie de la cara en 3 dimensiones para dar lugar a la cabeza entera.

Figura 7

Imagen tomada del sensor Kinect.

Fuente: (Cadena, 2021).

Para la adquisición de la superficie facial el método más efectivo y económico es a través de un láser o una luz estructural donde podemos encontrar modelos geométricos y con puntos de referencia que resuelvan nuestro problema. Entonces con este punto de correspondencia se adquiere la distorsión de patrón.

También existen varias formas de llegar al mismo resultado con el mismo instrumento, como por ejemplo Blanz et da un método de deformación de la superficie facial para lograr coincidencias, también se puede digitalizar los rasgos faciales y extrapolar los puntos faciales y comparar la distancia o el ángulo de dichos puntos (Cadena, 2021).

2.5.3 Método Hibrido

Para esta técnica se combina imágenes 2D y 3D haciéndolo más preciso un ejemplo de esto es el sistema DeepFace que es la combinación de redes neuronales e imágenes 3D dando como resultado una precisión de 91.4%, si comparamos esta técnica con las dos mencionadas anteriormente nos da resultados más confiables ya que su precisión es mayor.

Gracias a esto se realizó una evaluación biométrica en Vendor Test 2006 donde se midió la precisión de el reconocimiento facial con sistema DeepFace y un sistema de seguridad a base de reconocimiento de iris, donde tuvo resultados más favorables el sistema de reconocimiento facial y con las pruebas se logró mejorar los algoritmos de reconocimiento facial e iris (Barbosa, Velázquez, & Jiménez, 2023).

2.6 Técnicas de reconocimiento facial

Para realizar una correcta verificación de puntos biométricos se usan varias técnicas de reconocimiento dentro de las cuales se destacan dos, estas técnicas usan el análisis de los componentes y la comprensión de imágenes para tener una comparación de los puntos biométricos.

2.6.1 EigenFaces

Esta técnica usa el método de análisis de componentes principales (PCA) para la comprensión de imágenes o reconocimiento facial, para usar esta técnica la dividimos en dos fases una de entrenamiento y otra de clasificación. En la primera fase se recolecta las imágenes y por medio de PCA se recolecta valores de espacio de las facciones del rostro donde se lo conoce como EigenSpace.

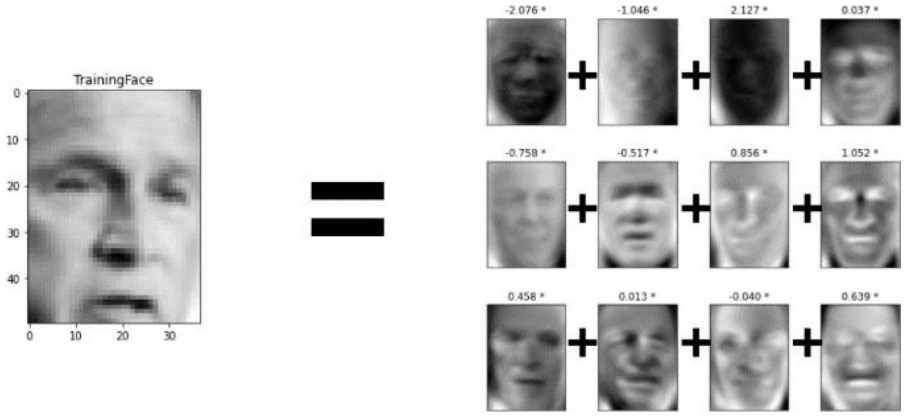
Con el EigenSpace se obtiene una matriz con vectores EigenVectores y EigenFaces donde tiene los valores de cada pixel de la imagen del usuario, para un mayor

número de matrices se le aumenta el número de imágenes con diferentes tipos de iluminación. Cada cara se considera un conjunto de estándares diferentes donde se suman los diferentes espacios dentro las características faciales.

En la siguiente fase se proyecta la nueva imagen contra el espacio que se creó en la fase de entrenamiento y por medio de la distancia en las características faciales se buscan coincidencias, sin embargo, esta técnica al estar tan ligada a la variable de la iluminación la hace muy dependiente al brillo de la imagen (Cardona, Ceballos, Torres, Mejia, & Boada, 2022).

Figura 8

Espacio creado en la primera y segunda fase.



Fuente: (Cardona, Ceballos, Torres, Mejia, & Boada, 2022).

2.6.2 Fisher Faces

Esta técnica acumula los puntos positivos de EigenFaces y agrega nuevos métodos para que se independicé de la iluminación por ejemplo en EigenFaces es dependiente de una foto frontal y una buena iluminación, pero en Fisher Faces no es tan sensible a estas variaciones para poder reconocer diferentes tipos de rostros.

Además, busca mejorar el número de muestras entre personas y tener una muestra más pequeña para una sola persona dando como resultado tener una base de datos con más usuarios a detectar, obviamente para esto es necesario tomar varias fotos con distintas

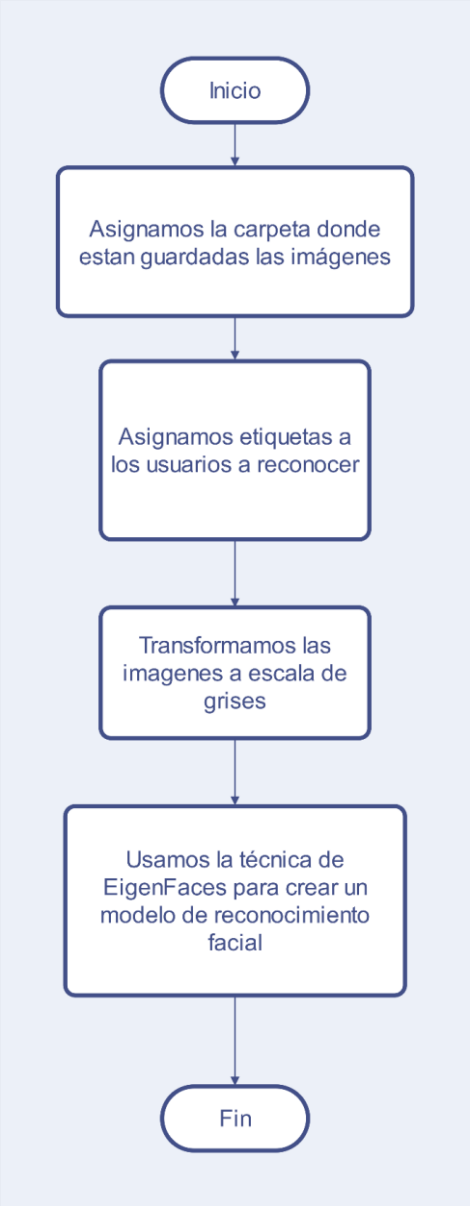
condiciones de iluminación de cada usuario para poder tener buenos resultados (Esparza, Tarazona, Sanabria, & Velazco, 2020).

2.7 Esquema del reconocimiento facial

Para el funcionamiento del sistema de reconocimiento facial se tiene dos pasos, para el primero usamos un código basado en EigenFaces donde va a capturar y almacenar los rostros de las personas para proceder un entrenamiento en torno al área de reconocimiento como lo puede apreciar en la figura 9.

Figura 9

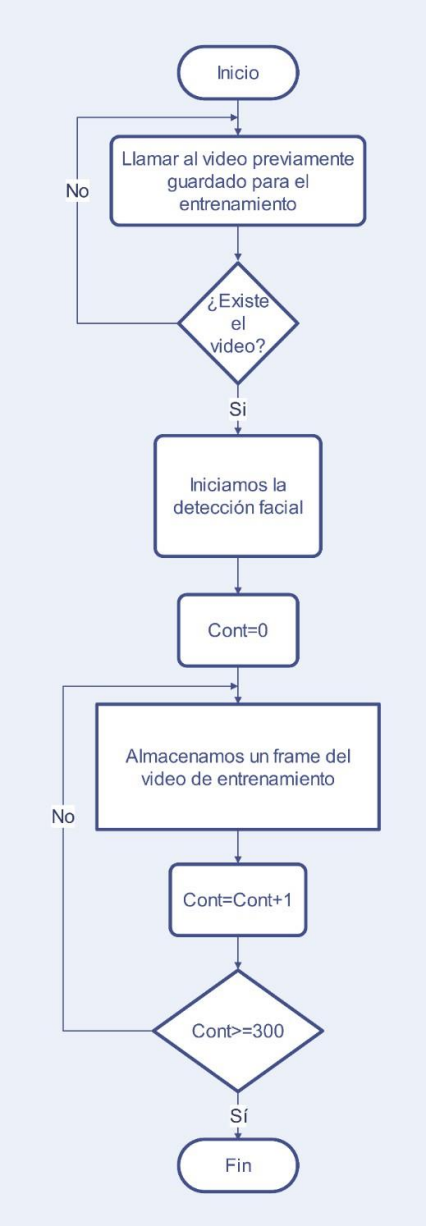
Diagrama de flujo de captura de imágenes.



Fuente: Elaborado por el autor.

Figura 10

Flujograma de entrenamiento del sistema.



Fuente: Elaborado por el autor.

El segundo paso después de almacenar las fotos es entrenar al código con las fotos guardadas en este caso se almaceno 300 fotos para comprender este paso podemos apreciar el siguiente flujograma.

Con el modelado del reconocimiento podemos comenzar las pruebas en base a videos e imágenes, cuando tengamos una buena fiabilidad comenzamos con la prueba con la cámara en vivo, Redimensionamos la base de datos a 400 imágenes para mayor fiabilidad y con diferentes expresiones faciales.

2.8 Base de datos de un sistema de reconocimiento facial

Uno de los puntos más importante para realizar el sistema es una buena de datos, ya que hay se guarda las imágenes del usuario y de esas imágenes se logra obtener los resultados que se van a comparar para reconocer al usuario. Por lo general se guarda imágenes en 3D con una base de 300 imágenes con diferentes pose y calidad de iluminación para una mayor tasa de precisión.

Con el avance de la tecnología muchos usuarios han creado una base de datos con fotos para el reconocimiento facial donde se pueden comparar y aportar nuevas imágenes referente al tema, vamos a nombrar algunos de estas bases de datos.

2.8.1 Base de rostros del sitio web YALE

Esta base de datos dispone de 165 imágenes en escala de grises donde hay 15 personas con diferentes expresiones faciales y en diferentes entornos, como la luz central, con gafas, con diferentes emociones en el rostro, luz derecha o imágenes de personas sorprendidas como se aprecia la figura 11.

Figura 11

Base de rostros del sitio web YALE.

Fuente: (Granja, Moreno, Cabrera, & Valle, 2020)

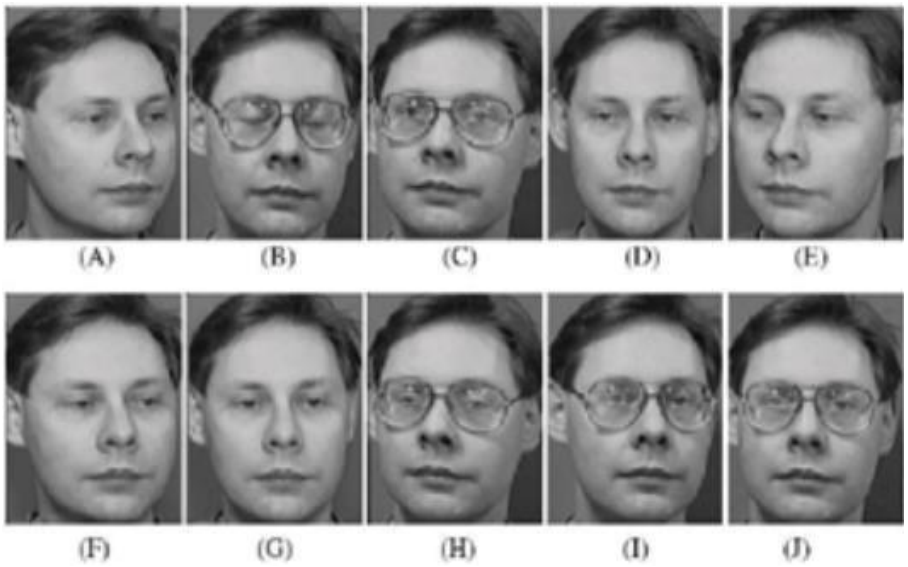
2.8.2 Base de rostros del sitio web ORL

Esta es una base de datos más grande que la anterior ya que cuenta con 400 imágenes de 40 personas distintas, con el objetico de hacer pruebas en códigos de reconocimiento de personas, ampliando el número de muestras de la base de datos YALE.

Pero al igual que la base de datos del sitio web YALE cada individuo tiene expresiones faciales diferentes, se le agregan jafas, con diferentes condiciones de luz, pero todo en un espacio homogéneo como se puede observar en la figura 12.

Figura 12

Base de rostros ORL.



Fuente: (Granja, Moreno, Cabrera, & Valle, 2020).

2.8.3 Base de datos del sitio web AR

Es una base que se creó en 1998 en Barcelona por centro de visión por computador y contiene imágenes de diferentes personas, las imágenes tienen mirada frontal y cuentan con diferentes entornos de luz.

2.8.4 Base de rostros del sitio web Feret

Para esta base de datos se clasifica los rostros en base al tiempo, a la iluminación, a la pose y a la expresión. Las imágenes fueron tomadas entre los años de 1993 y 1996

en la universidad de George Manson y el laboratorio de investigación de la armada de los estados unidos.

2.9 Reconocimiento de objetos

Con el desarrollo de la tecnología se han desarrollado distintas técnicas para reconocer imágenes específicas, las cuales sirven en muchas áreas ya sean con bases comerciales o de seguridad.

Para el desarrollo de esta tecnología se usa una cámara que emula el ojo humano y este proceso divide y procesa la imagen para poder capturar y reconocer el objeto en cuestión. Desde el desarrollo de esta tecnología se han ido aplicando en diferentes ámbitos de los cuales los principales son los que mencionamos en la figura 13.

Figura 13

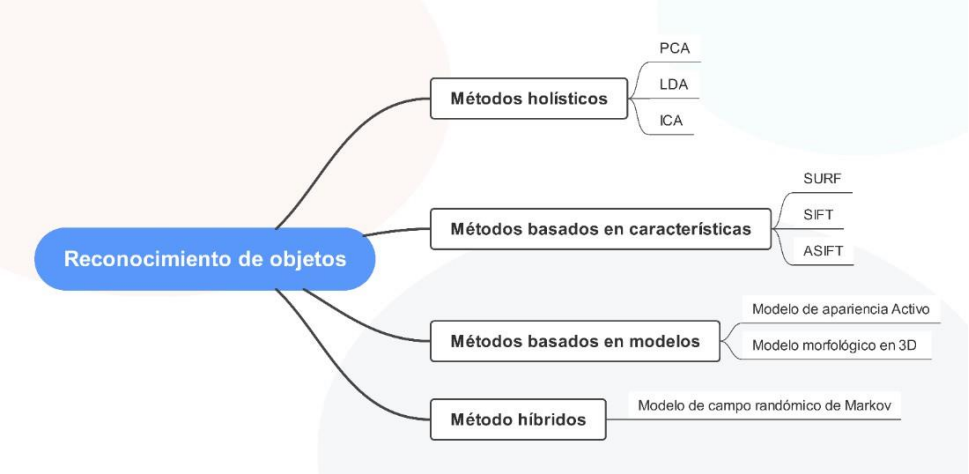
Área de aplicación de reconocimiento de objetos.

Fuente: Elaborado por el autor.

Gracias a estos campos de aplicación se puede clasificar los métodos más eficientes para el reconocimiento de objetos, de los cuales se ven en la figura 14.

Figura 14

Clasificación de métodos.



Fuente: Elaborado por el autor.

2.9.1 Método basado en Kernels

Este método es la fusión del método discriminante lineal (LDA) y el método de análisis de componentes principales (PCA), donde gracias a esta fusión se puede mapear vectores de entrenamiento con una función no lineal (Singh & Neelima, 2022).

Visto de manera más sencilla este método usa un análisis de un conjunto de datos que en este caso vienen de los vectores mapeados en las imágenes previamente guardadas y tratadas. Usa varios fundamentos matemáticos pero el fundamental es la investigación continua de la selección de parámetros para su debido ajuste.

2.9.2 Support Vector Machine (SVM)

Es un método que se usa para separar patrones en un espacio determinado por ejemplo sacar al sacar los puntos faciales de todas las imágenes se aplica SVM donde encuentra un hiperplano y separa los puntos de la misma clase, al maximizar cada clase se puede evaluar por conjunto a los que se les llama vectores de soportes (Singh & Neelima, 2022).

2.9.3 Métodos locales o geométricos

Estos métodos usan como datos las características principales del rostro como la boca, ojos, cejas, nariz etc. De los cuales las técnicas más destacables de este método son las siguientes:

1. Correspondencia entre Agrupaciones de grafos elásticos

Para este método las variables más importantes son la pose facial del usuario y la iluminación del entorno, ya que se toma esta pose facial y se coloca grafos etiquetados que guardan la información geométrica de la persona y se usa la transformación de Gabor para el procesamiento de imágenes basados en fenómenos biológicos. En la figura 15 se muestra una malla sobre el rostro, donde están los grafos y colocado un filtro grabo para extraer los datos del rostro.

Figura 15

Imagen tomada del sensor Kinect.

Fuente: (Alvarez, Marañon, & Orozco, 2022).

2. Modelado oculto de Márkov

Este modelo utiliza las propiedades de una señal donde determina sus parámetros para obtener un reconocimiento no solo facial si no del habla o escritura, pero lo malo es que es de forma temporal.

2.9.4 Métodos holísticos

Para obtener un reconocimiento de objetos podemos enfocarnos en todos los aspectos de las imágenes y no concentrarnos en una característica en específico, esto nos permite los métodos holísticos ya que se enfocan en un rango de características mucho más amplio dándonos diferentes perspectivas de un problema para dar una mejor solución en el apartado de reconocimiento. Para lo cual tenemos los siguientes:

1. Principal Components Analysis (PCA)

Es una técnica donde se utiliza un análisis de una proyección en el subespacio, donde se tiene unas imágenes iniciales de las cuales se computa la imagen promedio y se las resta con las imágenes que se usa en el entrenamiento, obteniendo un grupo de datos.

Luego con estos datos se compone una matriz con esto cada columna es una imagen, se evalúa la matriz de entrenamiento para obtener una covarianza y los componentes principales se computan si se resuelve la siguiente ecuación (Granja, Moreno, Cabrera, & Valle, 2020).

$$R^T(XX^T)R = \mathbf{\Lambda}$$

Donde $\mathbf{\Lambda}$ es la matriz diagonal de los valores propios, R es la matriz de los vectores ortogonales y X es la matriz de covarianza de entrenamiento.

Los vectores con mayor valor asociado son separados, los parámetros de comprensión de la imagen original indican su dimensión y cada una de estas son proyectadas sobre los vectores que fueron separados anteriormente, con esos valores se obtiene el reconocimiento.

2. Independent Components Analysis (ICA)

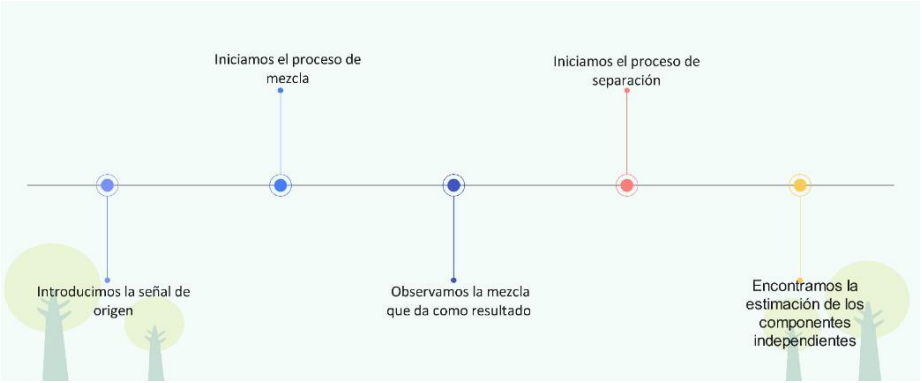
Esta técnica descompone una señal, en este caso en la imagen de un objeto, lo descompone en una combinación lineal de fuentes independientes, esta técnica surge de otra técnica que es Blind Sepparation Source y lo que hace obtener fuentes independientes de una determinada combinación.

Mientras que PCA proyecta en el subespacio y así minimiza el error cuadrático, ICA minimiza mayores ordenes de dependencia, para lo cual el número de imágenes de entrenamiento debe ser mayor o igual al número de imágenes originales.

Se crea matrices independientes y matrices de observación, con estos datos se puede crear un modelo de combinación. Asumiendo que las señales son independientes, el algoritmo de ICA intentara encontrar una matriz de separación para crear una estimación como lo podemos ver en la figura 16.

Figura 8

Modelo de ICA.



Fuente: (Kpushik, 2022).

3. Linear Discriminant Analysis (LDA)

Esta es una técnica de clasificación de datos con un aprendizaje supervisado, su idea central es tener una proyección en el espacio, pero a diferencia de PCA lo hace en un espacio mucho más pequeño que los datos de entrada, esta técnica busca patrones en las imágenes para un entrenamiento con etiquetas. También es importante aclarar es que

esta técnica no busca minimizar el error de representación como sí lo hace el método de PCA (Mohammed, Abbas, & Abood, 2023).

2.9.5 Métodos basados en características

También podemos encontrar características únicas para tener un enfoque exclusivo como pueden ser características biométricas específicas como la de los ojos, la nariz o incluso de los labios, a esto se lo conoce como extracción de características de los cuales tenemos los siguientes métodos:

1. Scale Invariant Feature Transform (SIFT)

Este método convierte una imagen en coordenadas que son invariantes en escala y rotación, Como esta técnica trabaja con las escalas de las imágenes no tiene un coste computacional tan alto como los métodos holísticos.

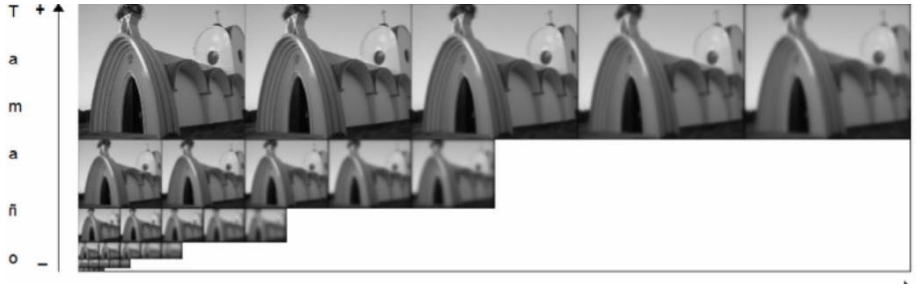
Para usar el método SIFT primero se debe seguir una serie de pasos para tratar la imagen de los cuales son los siguientes:

2. Función Scale-Space

Esta función realiza la búsqueda de los puntos en todas las escalas, para lo cual usamos una función continua, también hacemos una convolución en la imagen y una conversión gaussiana de la misma como se muestra en la figura 17.

Figura 9

Pirámide gaussiana.



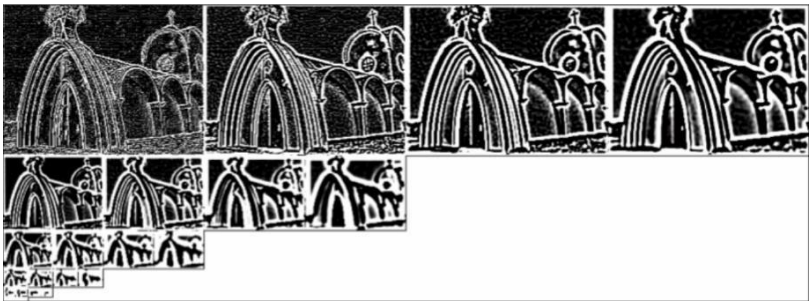
Fuente: (Jawad, Sabat, & Fatlawi, 2023).

3. Función Difference of Gaussian

En esta etapa se busca obtener puntos estables para esto se debe restar imágenes vecinas que es lo que se trabaja en Difference of Gaussian como se observa en la figura 18.

Figura 10

Pirámide de diferencia gaussiana.

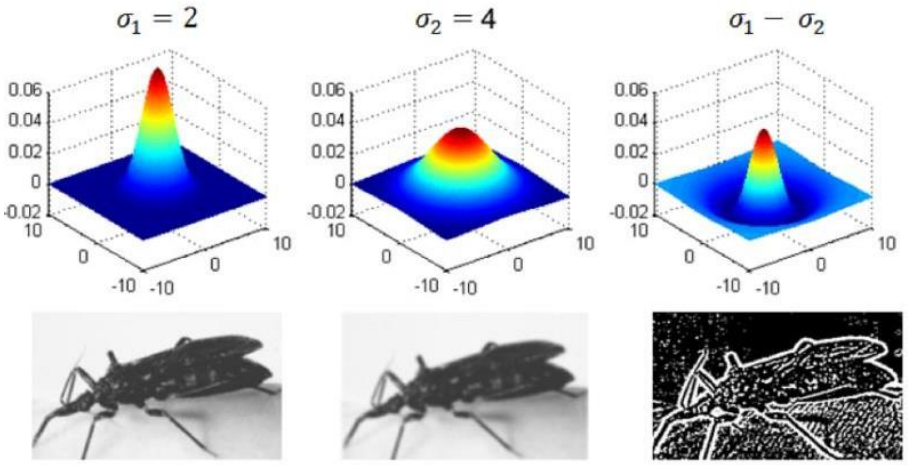


Fuente: (Jawad, Sabat, & Fatlawi, 2023).

También podemos entender mejor el proceso de esta resta de imágenes con la figura 19.

Figura 11

Resta de imágenes.



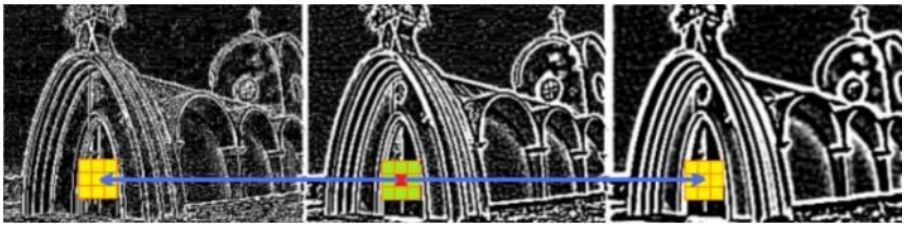
Fuente: (Jawad, Sabat, & Fatlawi, 2023).

4. Local extrema detection

Para este paso se busca buscar los puntos mínimos y máximos, para esto comparamos cada pixel obtenido en las imágenes, en este caso será comparados los 9 pixeles de la escala posterior y anterior, a continuación, los 8 pixeles de la escala anterior y posterior como lo muestra la figura 20.

Figura 12

Comparación de imágenes.

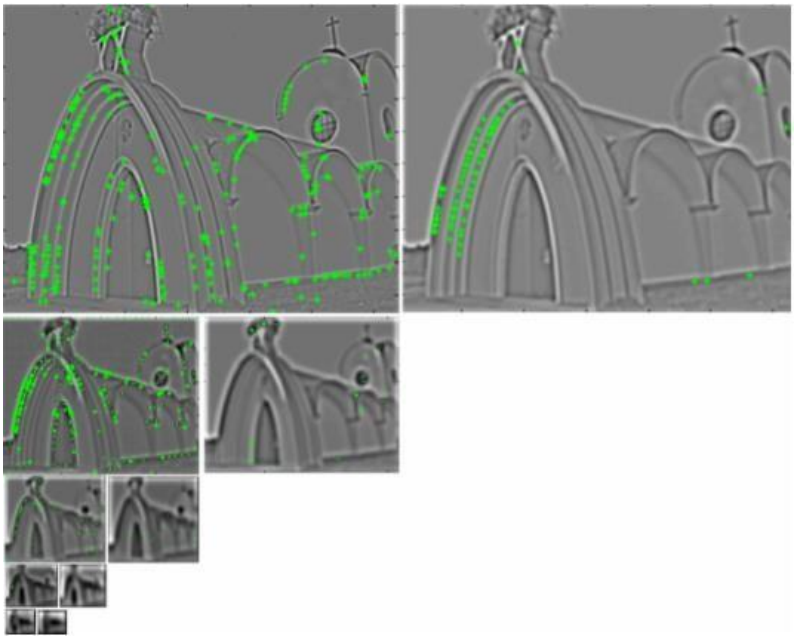


Fuente: (Jawad, Sabat, & Fatlawi, 2023).

Los puntos clave que sobrevivan serán los que son menores a 26 como lo muestra la figura 21.

Figura 13

Puntos clave obtenidos.



Fuente: (Jawad, Sabat, & Fatlawi, 2023).

Con esta serie de pasos obtuvimos los puntos clave, pero ahora debemos refinarlos para eso se eliminarán los puntos clave que no cuentan con las mejores características que en este caso son:

1. Puntos clave con bajo contraste

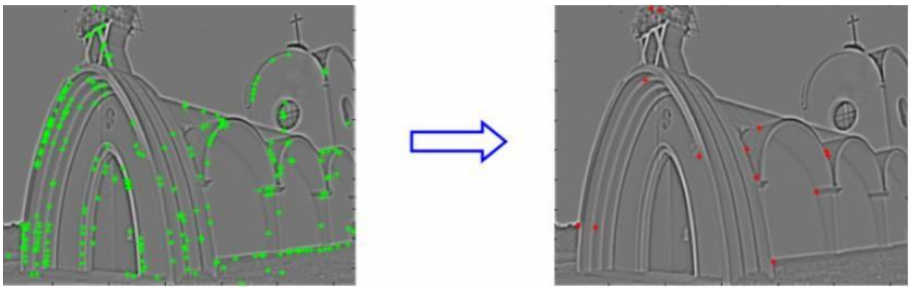
Para eliminar estos puntos se usa la expresión de Taylor, donde se calcula el extremo y refina un valor accesible en este caso si el valor es mayor a 0.5 se deberá recalcular y si es menor se hace un desplazamiento al punto inicial o de muestreo.

2. Supresión de puntos situados a lo largo de los bordes

Después de eliminar los puntos de bajo contraste se debe eliminar los puntos que estén al lado de los bordes para eso usaremos el método Hessiano, también se aumenta el umbral de 10 para eliminar por completo los puntos de los bordes como lo vemos en la figura 22.

Figura 14

Eliminación de los puntos de los bordes.



Fuente: (Jawad, Sabat, & Fatlawi, 2023).

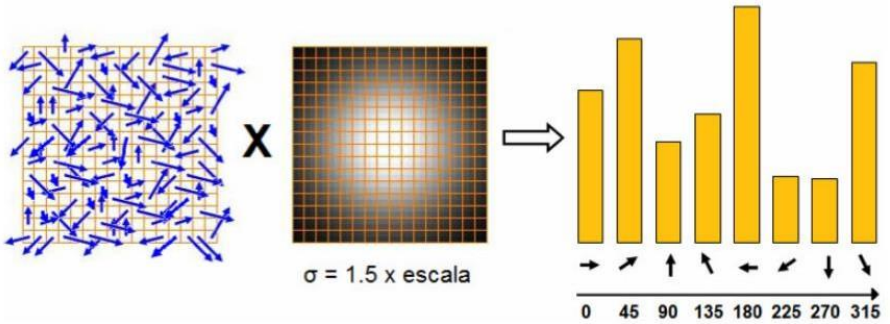
3. Orientation Assignment

Ya obteniendo los puntos clave debemos tomar un pixel y determinar la orientación para lo cual seleccionamos un pixel de 16x16 alrededor de la central y calculamos su gradiente usando su inclinación y módulo.

Para cada punto clave se crea un histograma de 36 orientaciones para lo cual ocupamos valores de orientación con el módulo y una ventana circular gaussiana con una escala de 1.5. Ya con el histograma vemos la orientación de los puntos clave a menos que la altura sea mayor a 80% para lo cual usaremos una parábola entre los puntos vecinos como lo vemos en la figura 23.

Figura 15

Histograma de los puntos clave.



Fuente: (Jawad, Sabat, & Fatlawi, 2023).

4. Keypoint Descriptor

Ya con el histograma debemos hacer un descriptor para lo cual usamos una región de 16x16 para rotarla, también usamos una región de 4x4 pixeles alrededor de la central, se obtiene 16 regiones y 8 orientaciones para usar una venta circular gaussiana. Para asegurar el resultado se normaliza la unidad y se especifica el umbral dependiendo de las imágenes en este caso es 0.2 a los histogramas.

5. Matching

Después de obtener el descriptor se observa la similitud de las imágenes para lo cual se multiplica los histogramas por las rotaciones.

Affine Scale Invariant Feature Transform (ASIFT)

Este método a diferencia del método anterior simula tres parámetros que son zoom, ángulo de la cámara tanto en latitud como en longitud. También normaliza tres parámetros que son la traslación, rotación y la escala para lo cual se realiza los siguientes pasos:

1. Afinación de la aproximación local

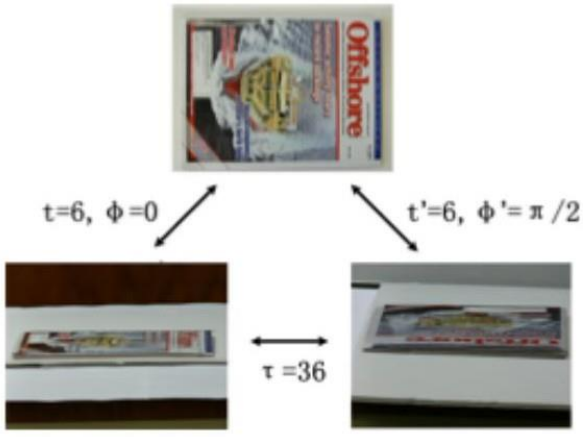
En este paso tomamos la imagen y usamos proyecciones como por ejemplo el piso se transforma en una forma rectangular mientras se calcula distancias del objeto con la fórmula de Taylor.

2. Simulación de distorsiones

En este caso hay que transformar la imagen en todas las posiciones posibles es decir tener diferentes orientaciones en longitud y latitud como lo vemos en la figura 24.

Figura 16

Rotación de la imagen.



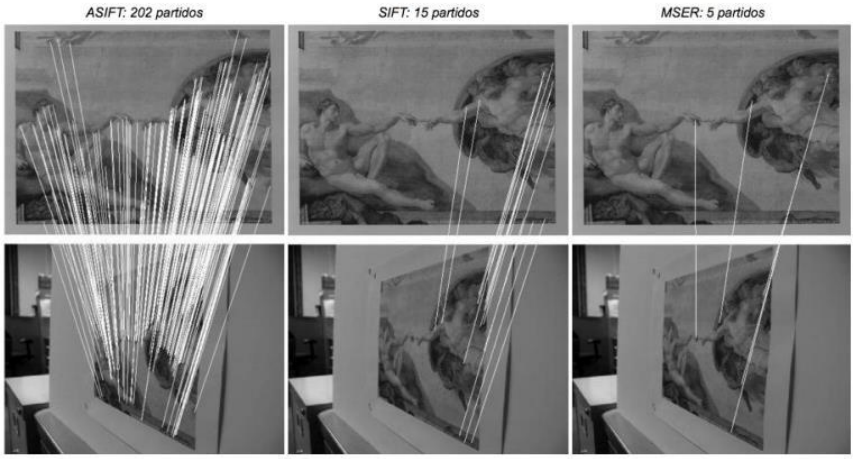
Fuente: (Jawad, Sabat, & Fatlawi, 2023).

3. Matching

Al final comparamos las imágenes obtenidas como lo vemos en la figura 25, el problema de este método que no elimina algunos puntos clave lo cual hace que tenga menos confiabilidad al momento de usarlo, pero es una forma de reconocer objetos.

Figura 17

Comparación de imágenes ASIFT.



Fuente: (Jawad, Sabat, & Fatlawi, 2023).

SPEEDED UP ROBUST FEATURES (SURF)

Este método está inspirado en el método de SIFT, ya que también toma puntos clave solo que diferencia del método antes mencionado este usa la matriz de Hessian, este método puede calcular de forma más rápida cuando se usa una imagen integral.

1. Detección de puntos de interés

Utilizamos la matriz Hessian para obtener los puntos de interés dependiendo de la curvatura local para lo cual se hace los siguientes pasos:

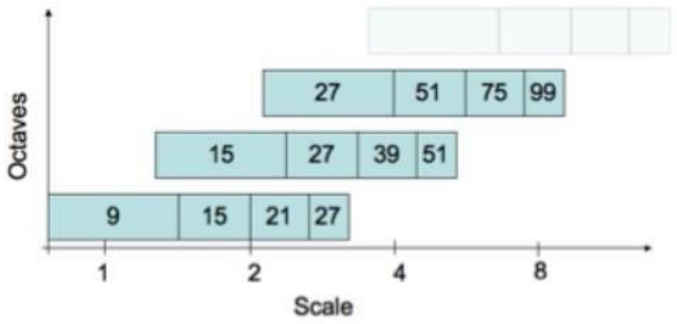
1. Se iguala a 0
2. Obtener los puntos críticos
3. Construimos la matriz Hessian
4. Evaluamos la matriz para saber si es máximo o mínimo o si esta indefinida.

Para determinar los puntos de interés se usa una pirámide suavizando cada pirámide de las diferentes escalas que se use, usamos la diferencia Gaussiana para

eliminar los puntos que no nos sirven. Dividimos los filtros para clasificar en 9x9, 15x15, 21x21 como se ve en la figura 26 (Jawad, Sabat, & Fatlawi, 2023).

Figura 18

Escalas de SURF.



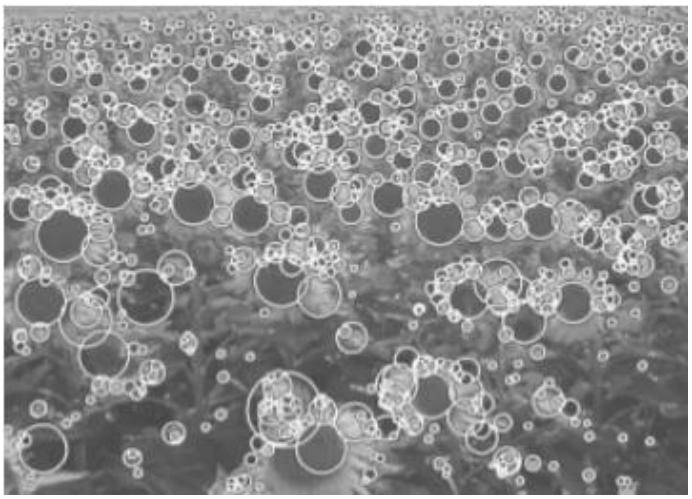
Fuente: (Jawad, Sabat, & Fatlawi, 2023).

1. Localización de puntos clave

Para localizar estos puntos clave usamos Fast-Hessian que es una técnica de puntos de interés como se ve en la figura 27.

Figura 19

Técnica Fast-Hessian.



Fuente: (Jawad, Sabat, & Fatlawi, 2023).

2. Asignamos la orientación

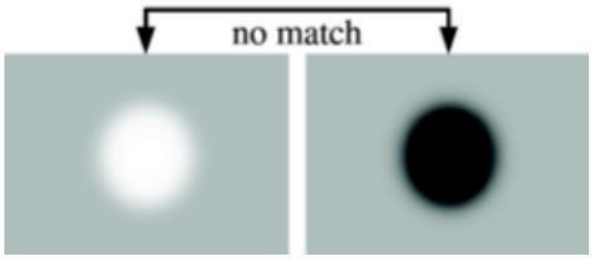
Esta técnica al ser no variante a la rotación se le debe de dar una orientación específica para lo cual tomaremos un punto clave como centro para así usar una ponderación gaussiana.

3. Matching

Usamos un método Laplaciano para distinguir manchas blancas y oscuras lo cual dependiendo el signo de respuesta de la ponderación gaussiana usada anteriormente se mira la comparación y no implica un costo computacional como se ve en la figura 28.

Figura 20

Resultado de la técnica.



Fuente: (López, y otros, 2021).

2.10 OpenCV

Para comenzar con uno de nuestros objetivos que es el reconocimiento facial vamos a usar la librería que es OpenCV. Esta librería nos permite usar estructuras y procesamiento de datos en tiempo real (Mínguez, 2021).

Esta librería es una biblioteca libre, tiene usos en varias plataformas, es dedicada para seguimiento de objetos o reconocimiento de objetos. Se ha mostrado mejores resultados en el lenguaje de C++, pero en el lenguaje de Python ha demostrado grandes avances y aplicaciones.

Para el lenguaje de Python se utiliza en la extracción de características de imágenes, reconocimiento facial, segmentación, robótica móvil entre otros. Para la

vinculación con el entorno virtual ROSMASTER ha demostrado un alto nivel de eficiencia por los conectores de lenguaje que existe dentro de esta librería.

2.11 MediaPipe

MediaPipe es una librería usada para la extracción de datos de video o imágenes y procesarlos, esta librería nos permite extraer varias características como lo son:

1. Extracción de las características de las manos
2. Extracción de las características de la cara
3. Extracción de las características del cuerpo

Para la extracción de los datos biométricos se ubican varios puntos clave que en este caso el numero aumenta dependiendo de la zona a extraer. Si es de la cara se extraen 33 puntos clave que nos ayudan para el reconocimiento o entrenamiento de un modelo de aprendizaje.

Cabe destacar que esta librería es dependiente de la calibración de la cámara ya que extrae puntos tridimensionales y el campo de visión de la cámara juega un papel crucial al momento de extraer y procesar datos. Una mala calibración puede provocar el aumento en la tasa de error, así como crear una distorsión dentro de la imagen.

CAPÍTULO 3

3 PROPUESTA DEL PROYECTO

3.1 Descripción del proyecto

El siguiente proyecto tiene como finalidad fortalecer la eficiencia de control de acceso para lo cual se utiliza varias tecnologías clave que son.

El robot Rosmaster X3, técnicas de Fisher Faces y MediaPipe Development.

La aplicación de técnicas como Fisher Faces y MediaPipe Development se enfocará en la detección, alineación, normalización y extracción de características faciales en tiempo real, lo cual es parte fundamental para tener control sobre el ingreso de personal y estudiantes dentro del laboratorio de telecomunicaciones asegurando el correcto uso de los equipos dentro del laboratorio.

El uso del robot Rosmaster X3 asegura la constante vigilancia dentro del laboratorio de telecomunicaciones ya que dicho dispositivo tiene la capacidad de moverse, así como un mayor control gracias a la detección de comportamientos inadecuados dentro del laboratorio de telecomunicaciones como lo son las señas obscenas.

Todas estas tecnologías no solo ayudan al control dentro del laboratorio, sino que también se adapta a los cambios que puede realizarse dentro del mismo. Para el correcto entendimiento del proyecto se ha desglosado en 2 etapas las cuales son:

Ensamblaje y componentes del robot ROSMASTER X3: detallamos los componentes físicos, software de aplicación, ensamble de equipos, es decir todo el proceso físico que se realizó para el desarrollo de este proyecto.

Desarrollo algorítmico y configuración del robot: Desarrollo del algoritmo que vamos a implementar, así mismo una base de datos dentro de la tarjeta Jetson Nano donde estarán varios usuarios que pueden ingresar dentro del laboratorio de telecomunicaciones, El código se lograra mediante el uso de MediaPipe Development donde se ejecutara dentro de la tarjeta Jetson Nano haciendo uso del lenguaje de Python, una vez acabado el código podemos hacer pruebas con el robot Rosmaster X3.

3.2 Ensamblaje y componentes del robot ROSMASTER X3

Introducción

Los componentes físicos a destacar que se ocupó en este proyecto se los va a detallar a continuación.

SLAM A1 Lidar

Equipo que combina dos tecnologías, que son la tecnología lidar con la tecnología de los algoritmos SLAM generalmente usado para el mapeo y localización en un entorno. Es comúnmente instalado en drones, vehículos autónomos y robots. Es capaz de medir distancias con una gran precisión, así como escanear y capturar detalles precisos en un entorno con una probabilidad de error muy baja, un detalle a tener en cuenta es que mientras más cerca este el objeto a escanear o capturar más es su precisión.

Cámara Astra Pro Depth

Cámara desarrollada por Orbbec la cual está diseñada para capturar imágenes 3D, ayudando a obtener datos con una presión alta, inicialmente fue pensada para aplicaciones de realidad aumentada, pero los usas en la robótica son amplios. Con su capacidad de capturar datos a una alta resolución y un sensor de profundidad integrado puede usarse en aplicaciones en tiempo real como lo son la detección facial.

Jetson Nano 4GB

Jetson Nano es una plataforma de computación de inteligencia artificial desarrollada por NVIDIA, su principal característica es que puede ejecutar aplicaciones de aprendizaje automático a bajo consumo eléctrico lo cual lo hace un producto esencial para los desarrolladores e investigadores que buscan realizar proyectos a bajo costo y al ser de bajo consumo eléctrico es accesible la implementación en varios equipos de robótica, a diferencia del Jetson Nano 12GB donde te ofrece mayor memoria pero a un costo demasiado elevado, no solo en el apartado monetario sino que también en el apartado energético haciendo difícil de implementar en equipos de robótica, el gasto energético seria enorme para mantener el Jetson, usualmente este equipo es usado en proyectos comerciales a gran escala donde la memoria es necesaria pero para la realización y comprobación de un proyecto el costo es excesivo y en un entorno móvil como lo es el Rosmaster X3 no es posible de implementar.

Antena M2

La antena M2 es desarrollada por Ubiquiti Networks y es usada para aplicaciones de red inalámbrica, en este caso la usamos para tener conectividad con un dispositivo móvil.

USB Hub Expansion Board

Es una placa de expansión donde agrega puertos que en este caso son 4 puertos USB y un puerto micro USB. Estos son usados para las respectivas conexiones entre las placas de expansión y la tarjeta Jetson Nano.

Robot Expansion Board

Es para la expansión de interfaces donde mejora las capacidades de un robot, tiene la capacidad de mejorar la conectividad entre varios dispositivos que en este caso son Jetson Nano, SLAM A1, cámara Astra Pro.

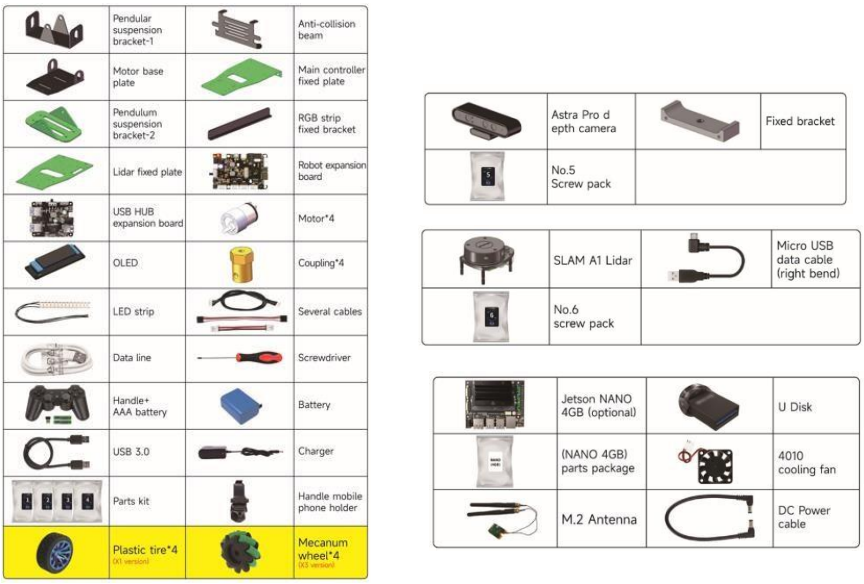
Ensamble del Robot Rosmaster X3

Para iniciar el ensamble del Robot Rosmaster X3 contamos con un manual que nos va a ayudar a colocar los elementos de forma correcta.

Como primer punto dividimos los equipos y materiales que vamos a usar como lo vemos en la figura 29:

Figura 21

Componentes del robot Rosmaster X3.

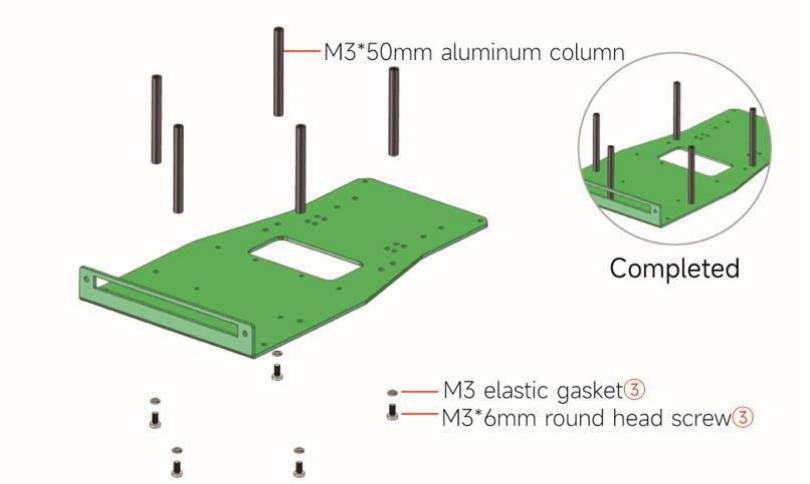


Fuente: (ROSMASTER-X3, 2022)

Comenzamos instalando las columnas de aluminio de 50mm con los tornillos de 60 mm, todo en la placa main controler como lo observamos en la figura 30.

Figura 22

Montaje de las columnas de aluminio.

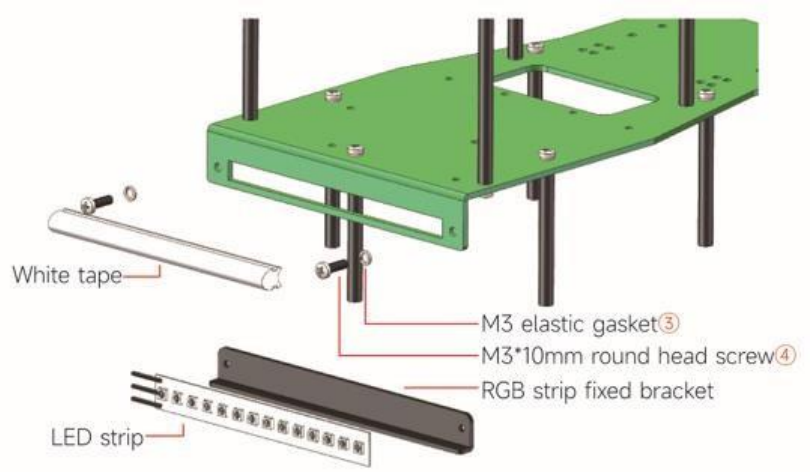


Fuente: (ROSMASTER-X3, 2022)

Comenzamos a instalar la tira led del robot para lo cual usamos el RGB strip y la tapa blanca que usamos como recubrimiento de la tira led como lo vemos en la figura 31.

Figura 23

Montaje de las luces LED.

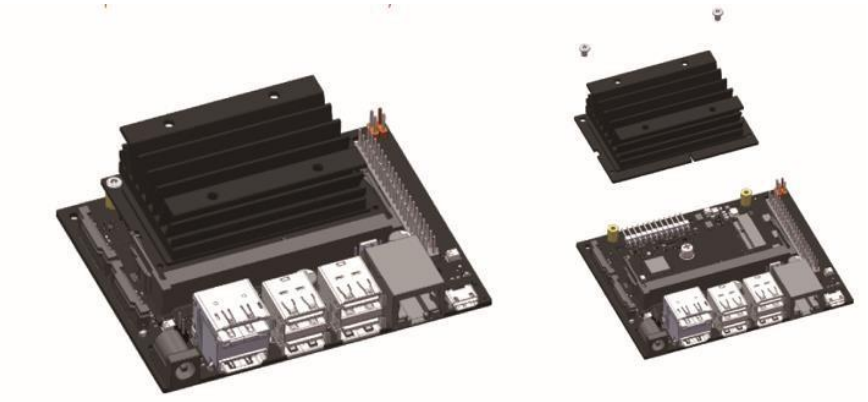


Fuente: (ROSMASTER-X3, 2022)

Ahora vamos a instalar la tarjeta de red dentro de la tarjeta Jetson Nano para su correcto funcionamiento para lo cual abrimos el core board como se muestra en la figura 32.

Figura 24

Abertura de la core board.

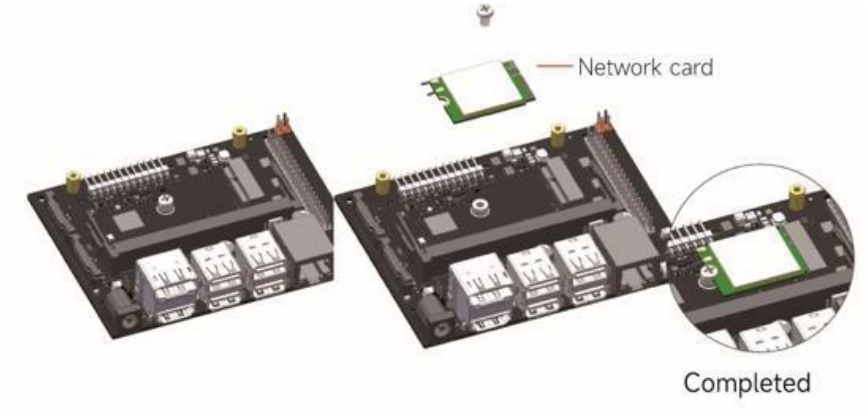


Fuente: (ROSMASTER-X3, 2022)

Instalamos la tarjeta de res y volvemos a colocar la core board asegurando de colocar los tornillos correspondientes como se muestra en la figura 33 y 34.

Figura 25

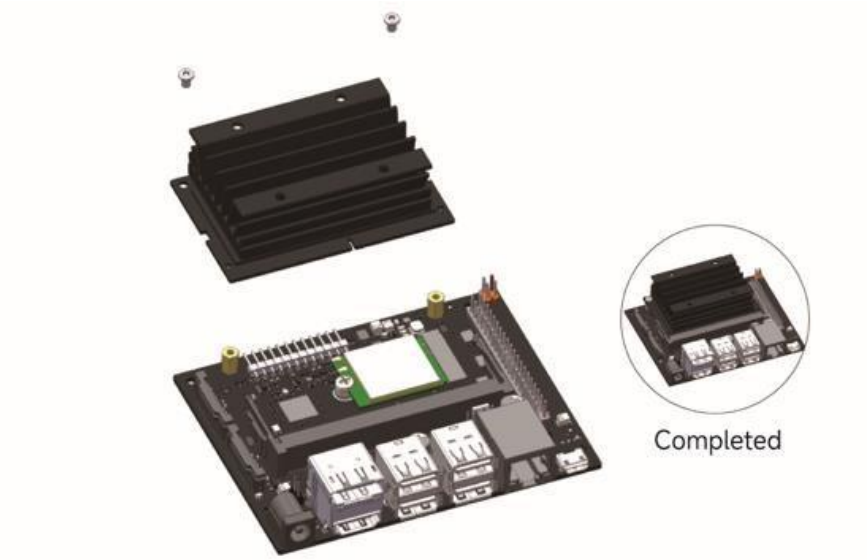
Instalación de la tarjeta de red.



Fuente: (ROSMASTER-X3, 2022)

Figura 26

Instalación de la core board.



Fuente: (ROSMASTER-X3, 2022)

Ahora instalamos el ventilador de la placa Jetson Nano y colocamos la placa dentro de la estructura ya armada anteriormente como lo vemos en la figura 35.

Figura 27

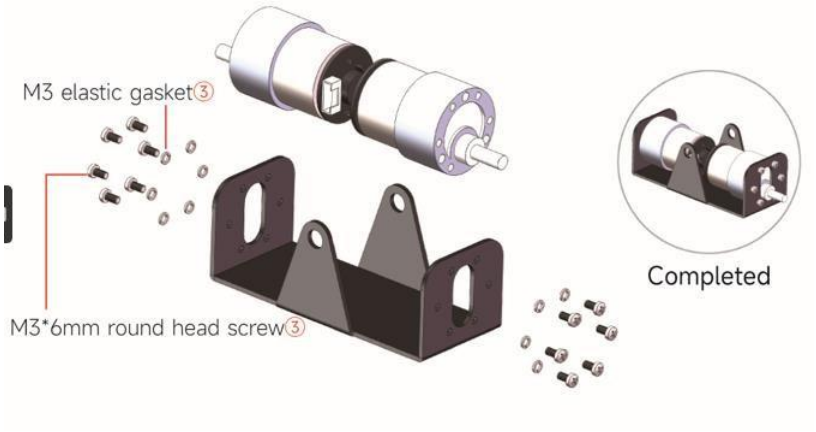
Instalación del ventilador y la placa.

Fuente: (ROSMASTER-X3, 2022)

Para la instalación de los motores del robot usamos la suspensión pendular donde lo atornillamos como se ve en la figura 36.

Figura 28

Instalación de los motores.

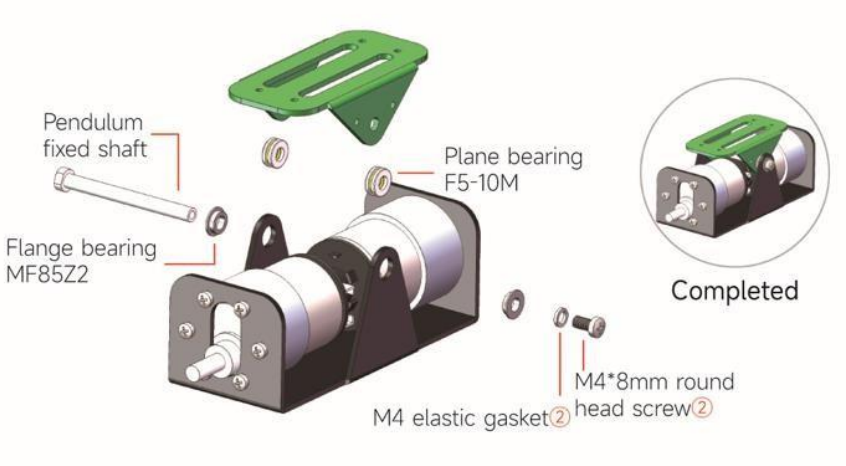


Fuente: (ROSMASTER-X3, 2022)

Instalamos la segunda suspensión pendular para dar el correcto movimiento a los motores cuando se ponga en marcha el robot como lo vemos en la figura 37.

Figura 29

Instalación de la suspensión pendular.

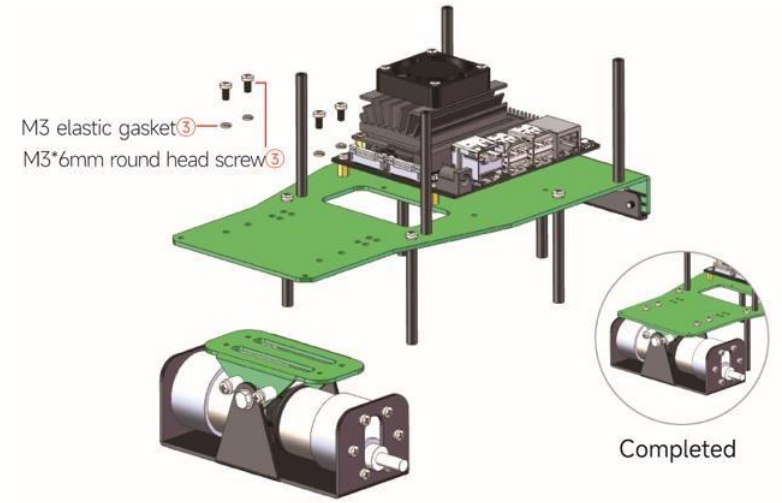


Fuente: (ROSMASTER-X3, 2022)

Colocamos el motor en la estructura anteriormente armada como lo vemos en la figura 38.

Figura 30

Montaje del motor en la estructura.



Fuente: (ROSMASTER-X3, 2022)

Para la correcta instalación de la cámara usamos el soporte fijo y el tornillo hexagonal como lo observamos en la figura 39.

Figura 31

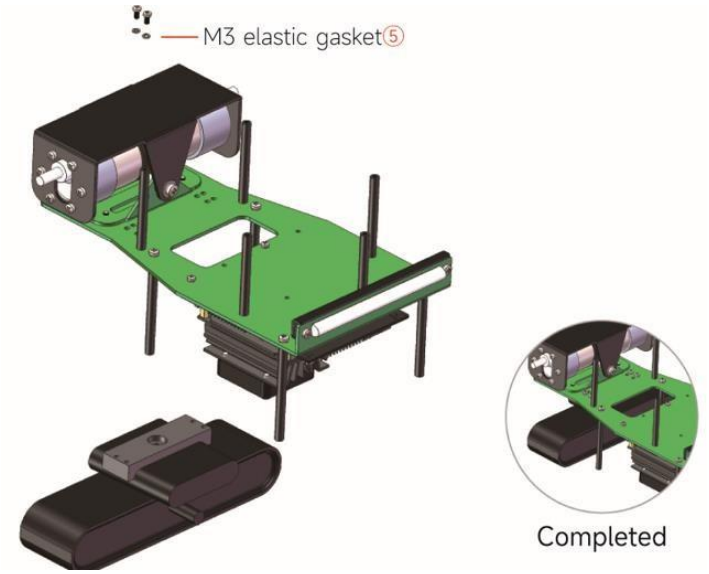
Montaje del soporte de la cámara.

Fuente: (ROSMASTER-X3, 2022)

Agregamos la cámara en la estructura previamente armada como lo vemos en la figura 40.

Figura 32

Montaje de la cámara en el soporte.

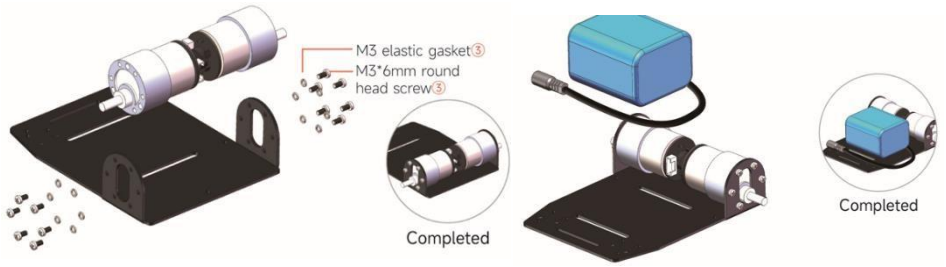


Fuente: (ROSMASTER-X3, 2022)

Usamos el otro soporte fijo para colocar los otros dos motores junto con la batería integrada del robot como lo vemos en la figura 41.

Figura 33

Instalación de la batería y motores.

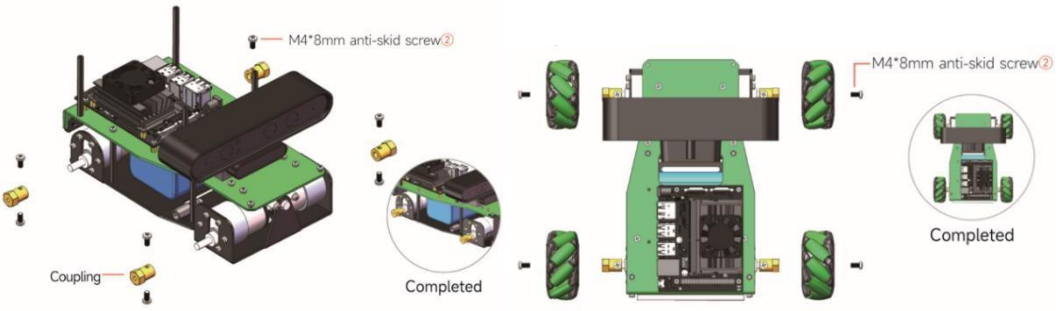


Fuente: (ROSMASTER-X3, 2022)

Vamos a colocar las placas del motor para poder colocar las ruedas del robot como lo vemos en la figura 42.

Figura 34

Montaje de las ruedas.

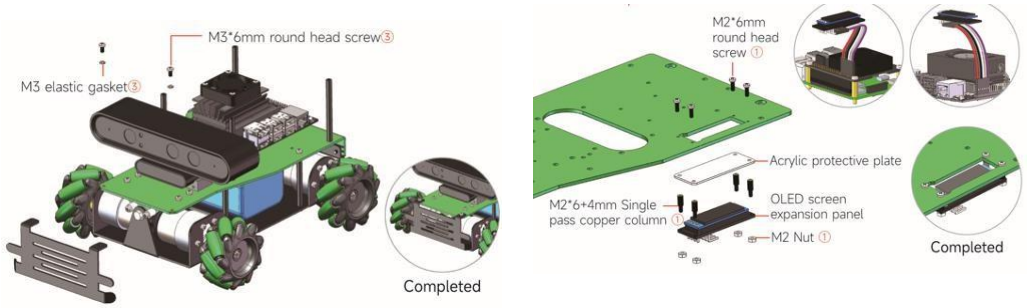


Fuente: (ROSMASTER-X3, 2022)

A continuación, vamos a instalar el antichoque del robot junto con el panel oled del robot para lo cual usamos los tornillos de 6 mm como lo vemos en la figura 42.

Figura 35

Montaje del Panel OLED.

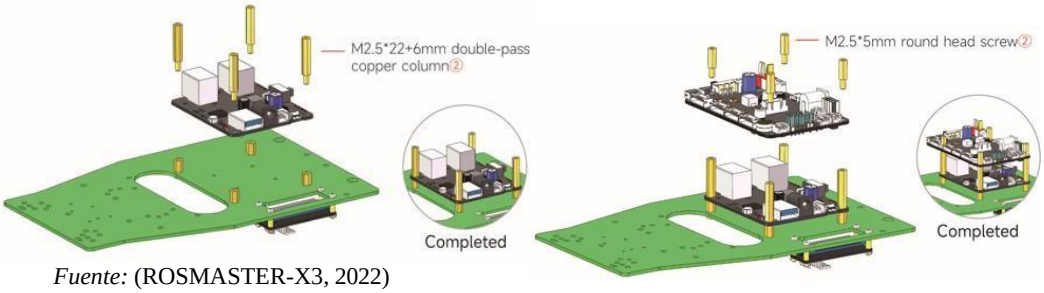


Fuente: (ROSMASTER-X3, 2022)

Para instalar las placas de expansión usamos las columnas de aluminio de 22 mm * 6mm como lo vemos en la figura 44.

Figura 36

Montaje de las placas de expansión.

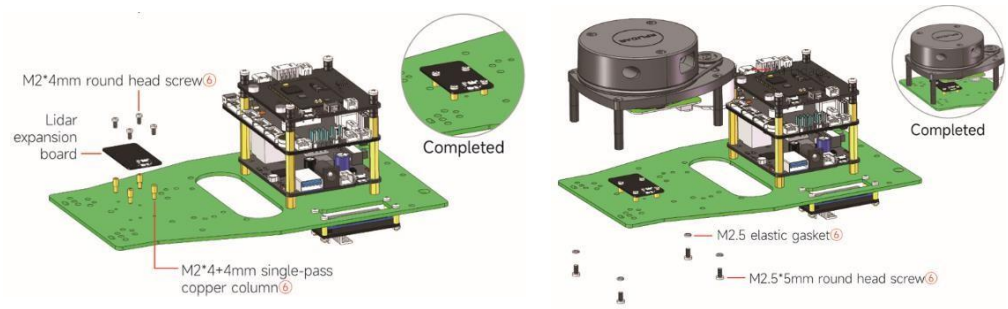


Fuente: (ROSMASTER-X3, 2022)

Para instalar el A1 lidar primero tenemos que instalar la placa de expansión junto con los tornillos de 5mm como lo vemos en la figura 45.

Figura 37

Montaje del LIDAR.

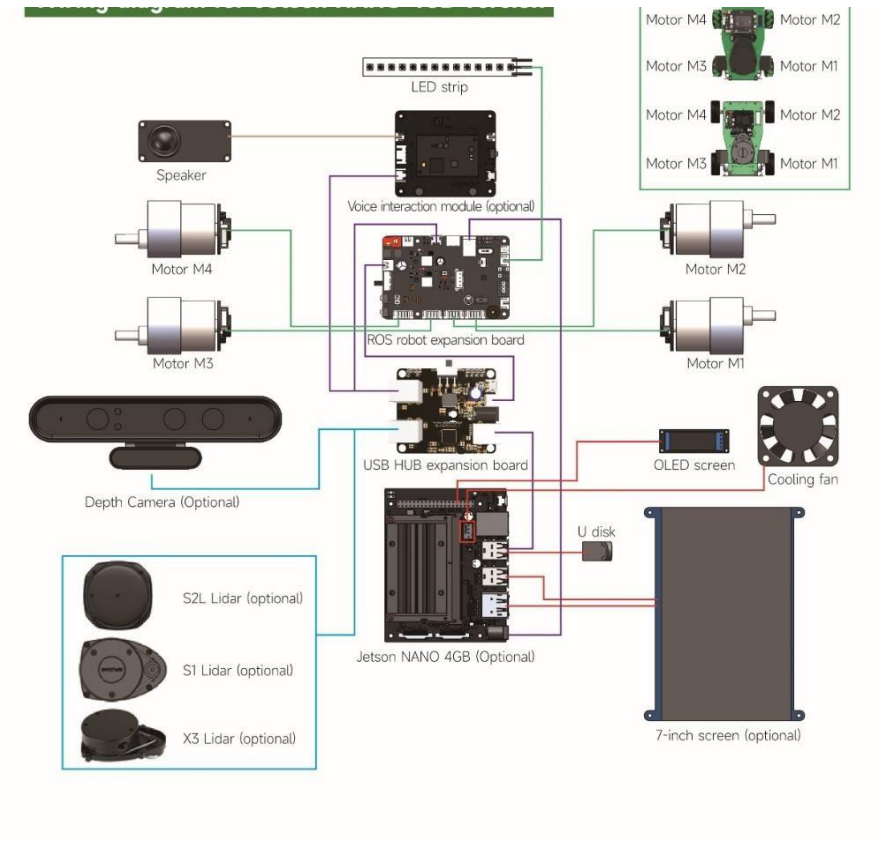


Fuente: (ROSMASTER-X3, 2022)

Ahora ya instalado todos los equipos correspondientes tenemos que encargarnos de las conexiones para lo cual seguimos las siguientes instrucciones que se pueden ver en la figura 46.

Figura 38

Conexiones del Robot Rosmaster X3.



Fuente: (ROSMASTER-X3, 2022)

3.3 Desarrollo algorítmico y configuración del robot

3.3.1 Configuración del robot

Como primer punto tenemos que instalar los softwares necesarios para el funcionamiento del robot, así como las librerías que se va usar para el código.

Para que funcione el entorno descargaremos Real VNC y Control Maker para controlar mediante el celular, usamos la dirección ip del robot que es 192.168.1.1 y su respectiva contraseña que en este caso es 123456789, también para la compilación del código usaremos el entorno de CONDA.

Las librerías usadas son Opencv, Imutils, openni, Numpy y Pillow. La librería de OpenCV nos ayuda en el apartado de detección facial y la librería Openni para la apertura de la cámara y tomar las respectivas imágenes para la base de datos, para la instalación de las librerías usamos las respetivas líneas de código.

Figura 39

Librerías para la codificación.

```
pip install numpy
pip install opencv-python
pip install openni
```

Fuente: Elaborado por el autor.

Como primer paso necesitamos realizar la base de datos de imágenes de los usuarios admitidos para la cual necesitamos tomar un número determinado de imágenes del rostro que en este caso son 300 imágenes en formato .jpg

Para lo cual llamamos las librerías y creamos una capeta con el nombre del sujeto que en este caso es Joan, usamos una sentencia para comprobar la correcta creación de la carpeta donde se van a almacenar las imágenes.

Figura 40

Creación de la base de datos.

```
import os
import cv2
import numpy as np
from openni import openni2
from PIL import Image

personaName = 'Joan'
datapath = r'C:tesis\Data'
personPath = os.path.join(datapath, personaName)

if not os.path.exists(personPath):
    print('Carpeta creada: ', personPath)
    os.makedirs(personPath)
else:
    print('Carpeta ya existe: ', personPath)
```

Fuente: Elaborado por el autor.

Descargamos los drivers para el infrarrojo de la cámara para poder optimizar la eficiencia del reconocimiento en la noche, para lo cual llamamos a NiViewer y lo iniciamos con la librería openni.

Figura 41

Iniciación de la cámara.

```
openni2_path = r'C:\Program Files\Orbbec\tools\NiViewer'
openni2.initialize(openni2_path)
```

Fuente: Elaborado por el autor.

Usamos harcascade para obtener una detección de rostro y tomar la imagen correspondiente, para la optimación de la captura de imágenes segmente el código en varias etapas de confirmación, esto ayuda para tomar una captura de imagen de manera óptima y no sobrecargar el procesador con la toma de imágenes.

Figura 42

Técnica Harcascade.

```
cascade_path = cv2.data.haarcascades + 'haarcascade_frontalface_default.xml'
if not os.path.exists(cascade_path):
    print(f"El archivo del clasificador en cascada no se encuentra en la ruta: {cascade_path}")
    exit()

faceClassif = cv2.CascadeClassifier(cascade_path)
if faceClassif.empty():
    raise IOError("El archivo del clasificador en cascada está vacío o corrupto")
print("Clasificador en cascada cargado correctamente")

count = 0
```

Fuente: Elaborado por el autor.

Normalizamos las imágenes y la convertimos cada Frame en una matriz con la librería Numpy, también verificamos el contenido de la imagen para poder guardar en entorno a la forma de rostro y los valores máximos y mínimos.

Figura 43

Normalizar los datos.

```
ir_image_normalized = cv2.normalize(ir_image, None, 0, 255, cv2.NORM_MINMAX)
ir_image_normalized = np.uint8(ir_image_normalized)
auxFrame = ir_image_normalized.copy()
```

Fuente: Elaborado por el autor.

Figura 44

Verificación del contenido.

```
try:
    ir_image = np.frombuffer(ir_frame.get_buffer_as_uint16(), dtype=np.uint16).reshape(480, 640)
except Exception as e:
    print(f"Error al convertir el frame de infrarrojos: {e}")
    break
```

Fuente: Elaborado por el autor.

Usamos una sentencia para confirmar el guardado de la imagen y evitar errores o archivos dañados al guardar y agregamos un contador para finalizar la sentencia en 300 imágenes tomadas.

Figura 45

Mensaje de confirmación.

```
try:
    rostro_image.save(file_path_jpg)
    print(f"Imagen guardada en formato JPG: {file_path_jpg}")
except Exception as e:
    print(f"Error durante el guardado de la imagen: {e}")

count += 1
```

Fuente: Elaborado por el autor.

Creamos varios bancos de imágenes en nuestra data para desarrollar un modelo de aprendizaje más preciso. Para realizar el reconocimiento creamos un modelo de aprendizaje con una red neuronal con CMake.

Figura 46

Bases de datos creadas.

Fuente: Elaborado por el autor.

Llamamos al banco de datos colocando la ubicación y haciendo un dataPath, para que nuestro modelo identifique el nombre de la persona creamos un listado de nombre con la variable peopleList que va a leer directamente de nuestra base de datos para saber de quien se trata. Usamos la variable labels para hacer el conteo de las imágenes leídas y tener una seguridad de que se trataron todos los datos.

Figura 47

Lectura de la base de datos.

```
import cv2
import os
import numpy as np

dataPath = r'C:tesis_programacion\Data'
peopleList = os.listdir(dataPath)
print('Lista de personas: ', peopleList)

labels = []
facesData = []
label = 0
```

Fuente: Elaborado por el autor.

Usamos sentencias para segmentar la codificación en varias partes, como primer apartado la correcta lectura de las imágenes y el mensaje de comprobación del mismo. Como final miramos el número de etiquetas obtenidos así comprobamos que asignamos un valor a cada imagen ya sea 0 o 1.

Figura 48

Comprobación de la correcta lectura.

```
for nameDir in peopleList:
    personPath = os.path.join(dataPath, nameDir)
    print(f'Leyendo las imágenes de {personPath}')

    for fileName in os.listdir(personPath):
        imagePath = os.path.join(personPath, fileName)
        print(f'Intentando leer la imagen en: {imagePath}')
        image = cv2.imread(imagePath, 0)
        if image is None:
            print(f"Error al leer la imagen: {imagePath}")
        else:
            facesData.append(image)
            labels.append(label)

    label += 1

print('labels= ', labels)
print('Número de etiquetas 0: ', np.count_nonzero(np.array(labels) == 0))
print('Número de etiquetas 1: ', np.count_nonzero(np.array(labels) == 1))
```

Fuente: Elaborado por el autor.

Usamos el método de modelado de LBPHF para realizar el reconocimiento, usamos una sentencia para comprobar si los datos son suficientes para realizar el correcto modelado del mismo.

Figura 49

Confirmación de la lectura de las imágenes.

```
if len(facesData) == 0 or len(labels) == 0:
    print("No hay datos suficientes para entrenar el modelo. Asegúrate de que las imágenes estén correctamente leídas.")
    exit()
```

Fuente: Elaborado por el autor.

Figura 50

Técnica LBPHF.

```
face_recognizer = cv2.face.LBPHFaceRecognizer_create()
```

Fuente: Elaborado por el autor.

Usamos el comando de train para entrenar los datos y usamos el comando write para guardar el modelo que próximamente vamos a utilizar.

Figura 51

Entrando el modelo.

```
print("Entrenando...")
face_recognizer.train(facesData, np.array(labels))
```

Fuente: Elaborado por el autor.

Figura 52

Mensaje de confirmación de guardado de modelo.

```
face_recognizer.write('modeloLBPHFace.xml')
print("Modelo almacenado...")
```

Fuente: Elaborado por el autor.

Hacemos el llamado del modelo que creamos en formato .xml para comenzar con el reconocimiento y el procedimiento de autentificación de personas.

Figura 53

Llamamiento del modelo entrenado.

```
face_recognizer = cv2.face.LBPHFaceRecognizer_create()
face_recognizer.read('modeloLBPHFace.xml')
```

Fuente: Elaborado por el autor.

Usamos una segmentación para comprobar la apertura de la cámara del robot y capturamos cada Frame para poder trasladar a escala de grises e iniciar el reconocimiento con harcascade.

Figura 54

Segmentación de apertura.

```
if not cap.isOpened():
    print("Error al abrir el dispositivo de la cámara.")
    exit()

print("Dispositivo abierto correctamente")

while True:
    ret, frame = cap.read()
    if not ret:
        print("Error al capturar la imagen.")
        break

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
    faces = face_cascade.detectMultiScale(gray, 1.3, 5)
```

Fuente: Elaborado por el autor.

Figura 55

Tratamiento de los Frame.

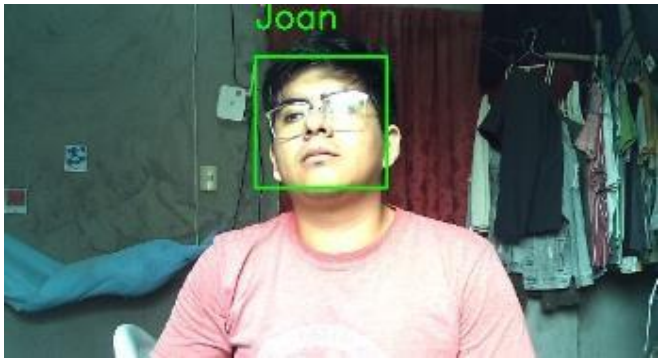
```
for (x, y, w, h) in faces:
    face = gray[y:y + h, x:x + w]
    face = cv2.resize(face, (150, 150), interpolation=cv2.INTER_CUBIC)

    result = face_recognizer.predict(face)

    cv2.putText(frame, peopleList[result[0]], (x, y - 25), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)
    cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)

cv2.imshow('frame', frame)

k = cv2.waitKey(1) & 0xFF
if k == 27:
    break
```

Fuente: Elaborado por el autor.

Interpolamos la imagen a un cubo de 150 x 150 y usamos un análisis de predicción con el modelo de entrenamiento realizado anteriormente para después mostrar el resultado de la imagen.

Figura 56

Reconocimiento facial.



Fuente: Elaborado por el autor.

Para completar este apartado designamos dos carpetas una donde se guarde un registro de entrada y la otra donde se guarda alas personas que entraron sin autorización así completamos el sistema de control de acceso.

Vamos a instalar MediaPipe para lo cual usamos los comandos pip que son los siguientes que vemos en la siguiente imagen.

Figura 57

Instalación de MediaPipe.

```
pip install mediapipe
```

Fuente: Elaborado por el autor.

Ahora con las funciones de MediaPipe Pose y MediaPipe Hands podemos reconocer poses y características en las manos lo cual usaremos para la detección de movimientos sospechosos, estos movimientos los reducimos en conductas inadecuados dentro del laboratorio las cuales son las señas obscenas y actos violentos dentro del laboratorio. Para lo cual desarrollamos un modelado de entrenamiento con imágenes y como primer paso realizamos la creación de una base de datos con dichas señas obscenas y conductas violentas.

Para iniciar el código primero llamamos a MediaPipe Pose y dibujamos los resultados de la detección dentro del video previamente grabado para poder detectar este movimiento violento como lo podemos observar en la siguiente imagen.

Figura 58

Llamamiento de MediaPipe.

```
mp_pose = mp.solutions.pose
pose = mp_pose.Pose()
mp_drawing = mp.solutions.drawing_utils
```

Fuente: Elaborado por el autor.

Lo siguiente a realizar es subir el video y crear una carpeta para descargar cada Frame en formato .png usamos las siguientes líneas de codificación.

Figura 59

Llamamiento del video.

```
video_path = 'comportamientos_violento.mp4'
cap = cv2.VideoCapture(video_path)

if not cap.isOpened():
    print("Error al abrir el archivo de video")
    exit()

output_folder = 'imagenes_extraidas'
if not os.path.exists(output_folder):
    os.makedirs(output_folder)
```

Fuente: Elaborado por el autor.

Ahora hacemos un contador para poder guardar un número determinado de imágenes que en este caso es 300. También cabe destacar que hay que convertir en formato RGB para poder usar MediaPipe Pose y guardarlo en la carpeta designada, se agregar un apartado para poder mostrar el Frame en pantalla como lo vemos en la siguiente imagen.

Figura 60

Codificación del tratado de los Frame.

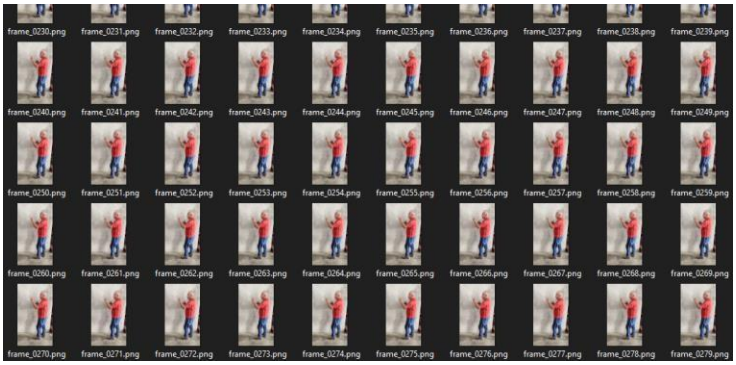
```
image_count = 0
max_images = 300

while cap.isOpened() and image_count < max_images:
    ret, frame = cap.read()
    if not ret:
        print("No se puede recibir frame (final del video?). Saliendo ...")
        break

    frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

    results = pose.process(frame_rgb)

    if results.pose_landmarks:
        mp_drawing.draw_landmarks(frame, results.pose_landmarks, mp_pose.POSE_CONNECTIONS)

    image_filename = os.path.join(output_folder, f'frame_{image_count:04d}.png')
    cv2.imwrite(image_filename, frame)
    image_count += 1

    cv2.imshow('MediaPipe Pose', frame)
```

Fuente: Elaborado por el autor.

Después de seguir los pasos de la codificación podemos realizar la base de datos, este código fue replicado para guardar todas las imágenes necesarias para el modelado así mismo como todos los movimientos destinados a detectar. Podemos ver el guardado de la primera base de datos en la siguiente imagen.

Figura 61

Base de datos movimientos sospechosos.



Fuente: Elaborado por el autor.

Ahora con la extracción de las características comenzamos con el entrenamiento del modelo para lo cual usamos Tensor Flow. Como tenemos varios tipos de movimientos incluyendo las características previamente entrenadas hacemos validación y entrenamiento de dichas características como lo vemos en la siguiente imagen.

Figura 62

Modelado con TensorFlow.

```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from sklearn.model_selection import train_test_split

X = np.load('features.npy')
y = np.load('labels.npy')

X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2, random_state=42)
```

Fuente: Elaborado por el autor.

Para finalizar creamos el modelo y comenzamos su respectivo entrenamiento para poder guardar en formato .h5 como lo vemos en la siguiente imagen.

Figura 63

Creación del modelo.

```
history = model.fit(X_train, y_train, epochs=20, batch_size=32, validation_data=(X_val, y_val))

model.save('model_obscene_violent.h5')
```

Fuente: Elaborado por el autor.

Con el modelo de entrenamiento ya podemos darle uso al mismo, para lo cual llamamos al modelo y comenzamos a captar las imágenes captada por la cámara del robot Rosmaster X3 como lo vemos en la siguiente imagen.

Figura 64

Uso del modelo entrenado.

```
model = tf.keras.models.load_model('model_obscene_violent.h5')

mp_drawing = mp.solutions.drawing_utils
mp_pose = mp.solutions.pose
mp_hands = mp.solutions.hands

cap = cv2.VideoCapture(0)

while cap.isOpened():
    ret, frame = cap.read()
    if not ret:
        break

    features = extract_features(frame)
```

Fuente: Elaborado por el autor.

Usamos el bucle while para confirmar la correcta captación de los Frame de la cámara en vivo, después dimensionamos el modelado de predicción con las características ya previamente extraídas para poder detectar si es movimiento violento o movimiento obsceno mostrándolo en labels como lo vemos en la siguiente imagen.

Figura 65

Bucle principal para el reconocimiento de datos.

```
if features.size > 0:
    features = np.expand_dims(features, axis=0)
    prediction = model.predict(features)

    label = 'Violent' if prediction > 0.5 else 'Obscene'
    cv2.putText(frame, label, (50, 50), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2, cv2.LINE_AA)

cv2.imshow('frame', frame)

if cv2.waitKey(1) & 0xFF == 27:
    break
```

Fuente: Elaborado por el autor.

CAPÍTULO 4

4.1 Verificación experimental

4.1.1 Reconocimiento facial en un entorno controlado

La cámara Astra Pro Depth tiene varios sensores que usamos en este entorno para tener la mayor eficiencia en las técnicas aplicadas en este entorno. El sensor infrarrojo se comportó de manera óptima en varios escenarios de diferentes tipos de iluminación, el paquete de función dentro del robot y la cámara Astra Pro toma la imagen y la convierte en datos que podemos tratar para con el comando roslaunch yahboomcar_mediapipe 04_FaceMesh.lanzamiento.

Figura 66

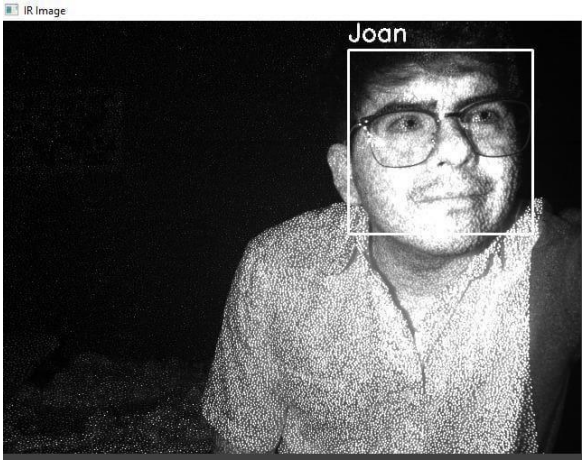
Uso del sensor de proximidad e infrarrojo.

Fuente: Elaborado por el autor.

Para un ambiente nocturno tomamos imágenes en infrarrojo y usamos el modelo previamente entrenado, para lo cual obtuvimos dos diferentes resultados en cuestión eficacia.

Figura 67

Reconocimiento en baja iluminación.



Fuente: Elaborado por el autor.

4.1.2 Reconocimiento de comportamientos sospechosos

Para este apartado lo dividimos en dos diferentes procesos, el primero es detección de señas obscenas y la segunda es detección de movimientos violentos donde creamos un modelo de aprendizaje con Deep learning.

Para el apartado de señas obscenas nos limitamos en las pocas señas que comúnmente son usadas por la gente de rango de edad 18 a 28 años, que es la común seña de levantar el dedo del corazón. Para lo cual activamos MediaPipe Hands en el robot con el siguiente comando.

Figura 68

Activación de Hand Detection.

```
roslaunch yahboomcar_mediapipe 01_HandDetector.launch # Hand detection
```

Fuente: Elaborado por el autor.

Figura 69

Detección de manos con MediaPipe.

Fuente: Elaborado por el autor.

Una vez encendido la parte de detección por medio de MediaPipe Hands se puede extraer 21 puntos de localización de las manos para tener un reconocimiento de gestos obscenos con gran precisión. Para el cual nos enfocamos en los 3 puntos del dedo del medio o corazón, estudiamos esos tres puntos y lo planteamos en el modelo para detectar la posición de los otros puntos cuando los tres puntos están por encima de los demás y así tener una detección de manera fiable.

Figura 70

Detección de seña obscena.

Fuente: Elaborado por el autor.

Inmediatamente detecta la superposición de los tres puntos del dedo corazón o del medio el robot reconoce una señal obscena manda un mensaje de alerta y guarda la imagen captada en una carpeta que se encuentra dentro de la tarjeta jetson nano. Donde podremos verlo conectado un monitor aparte en la tarjeta jetson nano o conectarlo a un computador o sobrescribir dentro de un pendrive para que se almacenen dentro del mismo y poder verlo en cualquier computador.

Para el apartado de reconocimiento de movimientos violentos dividimos en 5 tipos de guardias y golpes comúnmente usadas para iniciar una pelea o dar un golpe. Lo guardamos definiéndolas como poses de pelea del 1 al 5 donde están 300 imágenes por cada movimiento para obtener los datos suficientes para el modelo de entrenamiento.

Figura 71

Pose de pelea 3.

Fuente: Elaborado por el autor.

Para la detección y extracción de puntos usamos MediaPipe pose donde usamos 32 puntos de detección lo llamamos dentro del robot con el siguiente comando.

Figura 72

Comando para MediaPipe Pose.

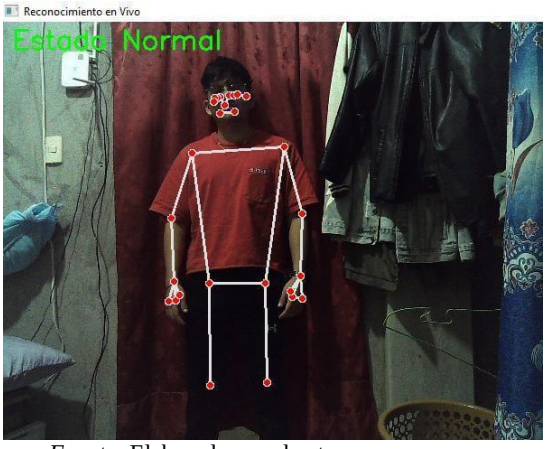
```
roslaunch yahboomcar_mediapipe 02_PoseDetector.launch # Pose detection
```

Fuente: Elaborado por el autor.

Una vez extraído los puntos comparamos con los datos que usamos para entrenar el modelo así poner un mensaje cuando estemos en un estado normal o sin anomalías y cuando tengamos una comparación efectiva dentro de los movimientos captados en la cámara del robot ponemos un mensaje de alerta y guardamos la imagen dentro de la carpeta designada.

Figura 73

Reconocimiento de movimientos violentos.

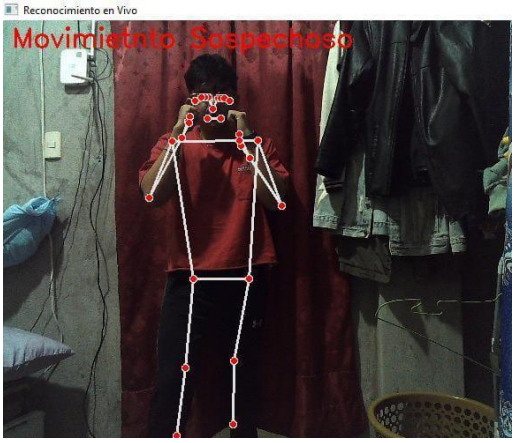


Fuente: Elaborado por el autor.

Pusimos a prueba el código de diferentes maneras, como primera estancia lo usamos con una sola persona dentro del rango de visión de la cámara a una distancia de 4 metros lo cual nos dio resultados más que efectivos ya que demostró su funcionalidad en diferentes campos de iluminación y detecto de manera óptima.

Figura 74

Reconocimiento de movimiento sospechoso.



Fuente: Elaborado por el autor.

Cuando hay más de una persona dentro del rango de visión de la cámara se le da prioridad al atacante, este se realizó para optimizar el apartado de rendimiento del procesador del jetson nano que por su capacidad tiene limitantes y puede tener problemas velocidad y rendimiento por todas las acciones que se ha entrenado en el modelado.

Figura 75

Reconocimiento de movimientos sospechosos.

Fuente: Elaborado por el autor.

4.2 Resultados Operativos de Validación

4.2.1 Análisis del ensamblaje y componentes del robot

El análisis de los componentes del robot nos sirvió para relacionarnos con el equipo que se utilizó en esta tesis, los resultados fueron satisfactorios ya que entendimos que componentes usar según que escenario de iluminación ahorrándonos tiempo de investigación. También nos facilitó el armado y ejecución del robot, como podemos ver en la figura 45 para tener todas las conexiones sin ningún inconveniente.

Después de armar el robot de manera correcta comenzamos a instalar la aplicación en nuestro teléfono, en este caso tenemos un código QR donde nos dirige a la Play Store y poder descargar la aplicación.

Figura 76

QR de la aplicación del robot.

Fuente: (ROSMASTER-X3, 2022)

Descargando la aplicación podemos conectarnos con el robot ya que la tarjeta de red del jetson nano nos brinda un punto de acceso con el nombre de ROSMASTER y la contraseña es 12345678. Al conectarnos a la red podemos abrir la aplicación con el nombre MakerControl y nos aparece la pantalla de inicio de la aplicación.

Figura 77

Interfaz de inicio de la aplicación.

Fuente: Elaborado por el autor.

Dentro de la interfaz elegimos el robot que vamos a utilizar en este caso es el Rosmaster X3, al seleccionar el robot a utilizar podemos iniciar los controles colocando la ip designada del robot y los parámetros para los puertos y hacemos clic en conectar.

Figura 78

Configuraciones WIFI.

Fuente: Elaborado por el autor.

Una vez conectado nos muestran varias funciones dentro de la interfaz la cual nosotros elegimos la opción de control remoto para dar como finalizado el apartado del funcionamiento para poder programar y ejecutar los modelos de aprendizaje que creamos.

Figura 79

Interfaz de las funciones del robot Rosmaster X3.

Fuente: Elaborado por el autor.

1. Análisis del desarrollo algorítmico y configuración del robot

Fue la parte más complicada ya que aquí desarrollamos el código para los modelos de entrenamiento y probarlos en diferentes tipos de iluminación y probar su eficacia. Lo primero que hicimos fue guardar todas las librerías correspondientes, entramos Anaconda Prompt para Linux para poder usar el código dentro de la jetson nano y no tener problemas con la ejecución del código, usamos el software de Anaconda por su versatilidad en varios lenguajes de programación, así como lo fácil que fue vincular el robot con un entorno virtual dentro de nuestra PC.

Ya instalado todas las librerías necesarias procedemos a instalar NiViewer de la página oficial de Orbbec para poder explorar todas las funciones y herramientas que nos brinda la cámara de nuestro robot, este paso es importante ya que nos permite pasar del sensor de la cámara principal a el sensor infrarrojo para poder usarla en diferentes tipos de iluminación.

Figura 80

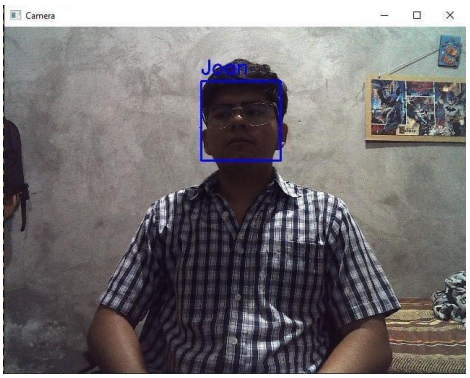
Reconocimiento facial con baja iluminación.

Fuente: Elaborado por el autor.

Para la baja o nula iluminación usamos el sensor infrarrojo lo cual nos dio resultados favorables en todas las pruebas que se realizaron, en cambio con el sensor principal también se realizaron varias pruebas con diferentes tipos de iluminación y diferentes tipos de rostros lo cual nos dio resultados variables, en el caso de buena iluminación nos dio resultados favorables en todas las pruebas, pero cuando la fuente de luz es precaria o nula sus resultados fueron ineficientes.

Figura 81

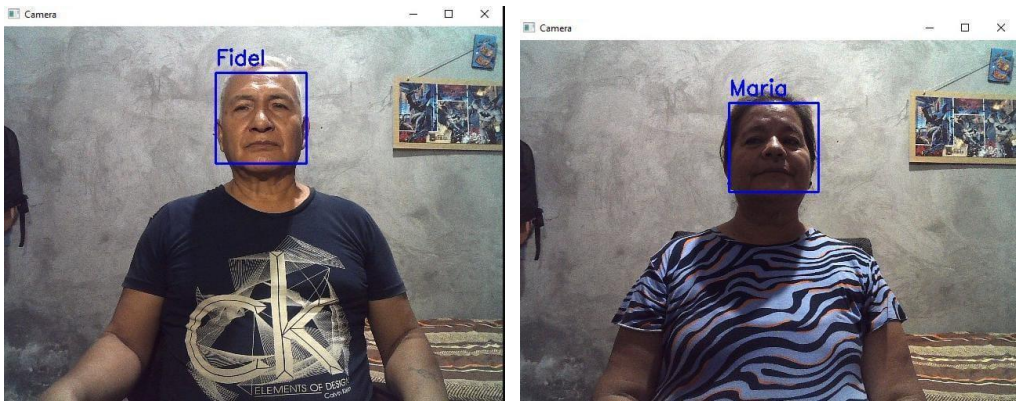
Reconocimiento facial con buena iluminación.



Fuente: Elaborado por el autor.

Figura 82

Prueba del modelo de autentificación con varias personas.

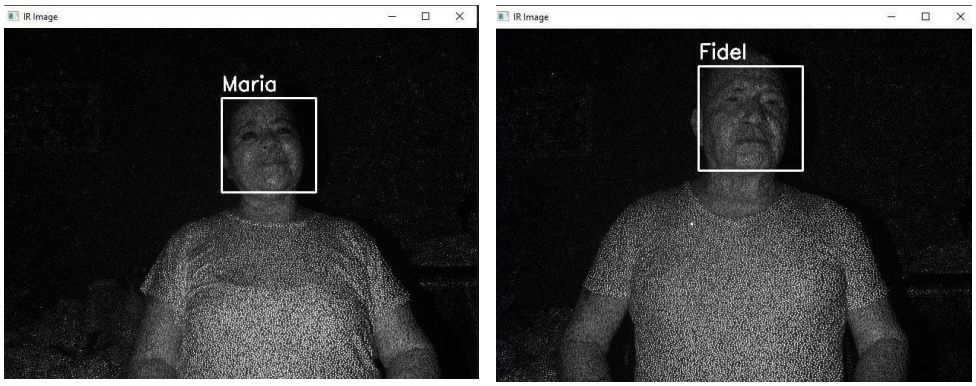


Fuente: Elaborado por el autor.

Ya con modelo entrenado para el reconocimiento de varias personas podemos obtener un sistema de autentificación monitoreando la entrada a un área controlada, así como su salida, para el mismo almacenamos la información de entrada en una carpeta con la descripción de entrada al laboratorio dentro de la jetson nano donde se guardará la imagen en formato .jpg con la descripción de la hora que fue tomado la imagen junto con el nombre y la imagen de la persona reconocida.

Figura 83

Prueba de modelo de autentificación en baja iluminación.



Fuente: Elaborado por el autor.

Para validar nuestro modelo usamos un sistema de THRESHOLD donde usamos varias técnicas de aprendizaje, en este caso empleamos 3 métodos de aprendizaje los cuales son EigenFaces, Fisher Faces y LBPH (Histogramas de patrones binarios locales).

En la prueba THRESHOLD se tomaron 120 frames que son sacados de la base de datos que ya tenemos guardado, lo cual lo definimos en varios estándares que en este caso son en torno al color, morfología y ecualización de imágenes. Para la respectiva prueba de aciertos con los modelos de aprendizaje tenemos los siguientes resultados.

Tabla 1

Resultados de la prueba de aciertos del modelo LBPH.

Método valorado	**Porcentaje de acierto**
Color	95.5%
Morfología	81.75%
Ecualización	81%

Fuente: Elaborado por el autor.

Tabla 2

Resultados de la prueba de aciertos del modelo EigenFaces.

Método valorado	**Porcentaje de acierto**
Color	70.75%
Morfología	36%
Ecualización	68.25%

Fuente: Elaborado por el autor.

Tabla 3

Resultados de la prueba de aciertos del modelo Fisher Faces.

Método valorado	**Porcentaje de acierto**
Color	59.75%
Morfología	34.70%
Ecualización	64.5%

Fuente: Elaborado por el autor.

Como podemos ver el método de LBPH es el que mejor resultados nos dio y es el modelo de entrenamiento que usamos para nuestro sistema de reconocimiento y autentificación.

Para el apartado de detección de movimientos sospechosos tenemos un modelo de aprendizaje basado en SKlearn lo cual fue elegido por la variedad de usos que se le puede dar a este modelo de aprendizaje que pueden ser desde la clasificación, regresión hasta análisis de grupo y bosques aleatorios.

Figura 84

Prueba de detección de señas obscenas.

Fuente: Elaborado por el autor.

Para todas las pruebas realizadas para la detección de señas obscenas los resultados fueron más que satisfactorios porque logramos grandes resultados en recatadas condiciones de luz. Así mismo con el reconocimiento del apartado de movimientos violentos los resultados fueron satisfactorios tanto en el reconocimiento como en el almacenado de las imágenes con la acción sospechosa dentro de la jetson nano.

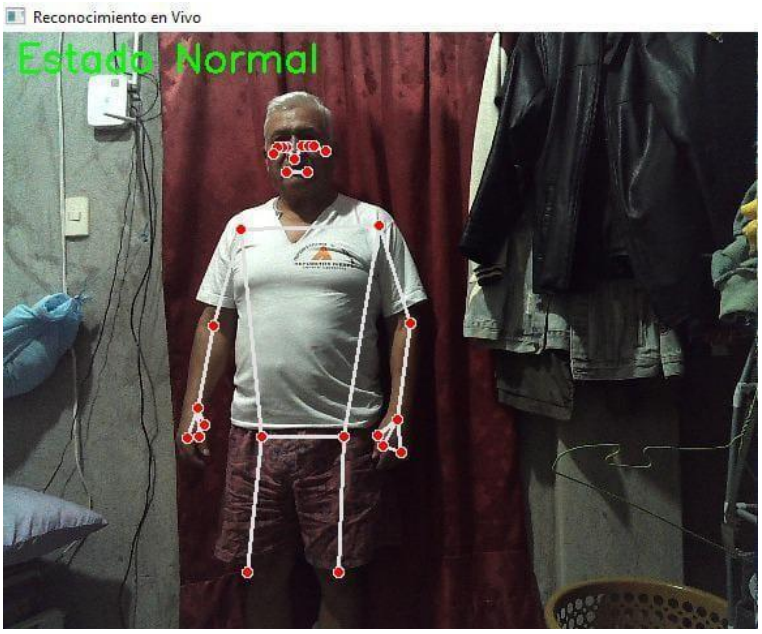
2. Análisis de la verificación experimental

Para un área controlada como puede ser el laboratorio de telecomunicaciones los resultados fueron más que satisfactorios ya que en todas las acciones no existió un error al momento del reconocimiento sin importar la persona que realice el movimiento,

también se reconoce los puntos clave para que el sistema en MediaPipe no confunda una persona con ropa, mientras este en el área de visión de la cámara es muy eficiente el sistema de reconocimiento.

Figura 85

Resultados del modelo de detección de movimientos sospechoso.



Fuente: Elaborado por el autor.

CONCLUSIONES

- La implementación y desarrollo del sistema tuvo resultados más que satisfactorios para un entorno controlado, se obtuvo resultados muy altos en entorno de eficiencia cabe destacar las limitaciones del robot, pero su funcionalidad en el entorno de monitoreo de acceso es más que satisfactorio.
- Los modelos de entrenamiento combinado en conjunto con técnicas como MediaPipe Pose nos da resultados más que favorables en el apartado de reconocimiento de movimientos y autentificación de usuarios.
- La eficiencia de cada modelo de aprendizaje es un apartado que varía según el tipo de modelo de aprendizaje donde los resultados se inclinan al modelo de LBPH con un 95.5% de efectividad.
- La combinación de las técnicas de detección con el Robot Rosmaster hace que la comprensión de estas técnicas sea muy intuitivas y fáciles para el entorno del aprendizaje.

RECOMENDACIONES

- Para el usar el control de acceso lo que se debe tomar en cuenta es habilitar el sensor infrarrojo de la cámara como sensor principal de la cámara para que el factor de la iluminación no altere la eficiencia del mismo.
- Para el correcto funcionamiento del sistema asegurarse que todas las librerías estén instaladas dentro de la placa jetson nano, esto podría generar la mala ejecución de los modelos que ya han sido entrenados o problemas en la apertura de la cámara.
- La distancia del robot con el objetivo a reconocer juega un papel crucial, no se debe estar a más de 10 metros de distancia ya que esto puede afectar en la efectividad del sistema.
- Revisar las conexiones del robot Rosmaster X3 ya que siempre suelen moverse cuando es manipulado, asegurarse que todas las conexiones estén bien conectadas puede ser vital para el correcto funcionamiento del mismo.

BIBLIOGRAFÍA

Abolmaali, Hafeez, Mohsin, Hassan, & Thakur. (2021). *Pill Ingestion Action Recognition using MediaPipe Holistic to monitor elderly patients.* International Supply Chain Technology Journal.

Alvarez, A., Marañon, E. J., & Orozco, R. (2022). *Revisión de los métodos de reconocimiento facial en imágenes RGB-D adquiridas mediante un sensor Kinect.* Santiago de Cuba: Revista Cubana de Ciencias Informáticas.

Barbosa, R. C., Velázquez, A. T., & Jiménez, C. (2023). *Reconocimiento híbrido de emociones a partir del análisis de voz y de expresiones faciales.* Oaxaca: Universidad Tecnológica de la Mixteca.

Cadena, J. (2021). *Técnica eficiente para reconocimiento facial global.* Lima: Universidad Nacional Mayor de San Marcos.

Cardona, D., Ceballos, J., Torres, J., Mejia, M., & Boada, A. (2022). *Face Recognition—Eigenfaces.* Intelligent Systems Reference Library.

Cedano, M. (2020). *Reconocimiento de la agresión física con Deep Learning y visión artificial utilizando cámaras de video, caso observado Institución Educativa Rosa Suárez Rafael N°20436.* Piura: Universidad César Vallejo.

Ciudad Real Villaescusa, A. (2022). *Diseño del control automatizado para el seguimiento de personas por reconocimiento facial en tiempo real.* Valéncia: Universidad Politécnica de valéncia.

Esparza, C., Tarazona, C., Sanabria, E., & Velazco, D. (2020). *RECONOCIMIENTO FACIAL BASADO EN EIGENFACES, LBHP Y FISHERFACES EN LA BEAGLEBOARD-xM.* Revista Colombiana de Tecnologías de Avanzada.

Filio, E. (2021). *Diseño de un sistema de seguridad física mediante Reconocimiento Facial a través del flujo de video, siguiendo las mejores prácticas de las normas ISO 80601, 13154, 19794 y el NISTIR 8238, para el área de seguridad de una empresa minera.* Cataluña: UPC.

Flores, A., Fernando, C., Mariño, G., & Fabricio, R. (2022). *Sistema automatizado para la gestión de asistencia y control de temperatura con reconocimiento facial de los empleados de la empresa eléctrica Ambato EEASA.* Ambato: Universidad Técnica de Ambato. Facultad de Ingeniería en Sistemas.

Gárate, A., Hajjam, A., & Andrès, E. (2020). *Classification models for heart disease prediction using feature selection and PCA.*

Granja, I., Moreno, D., Cabrera, F., & Valle, P. (2020). *Procesamiento De Imágenes Para La Identificación De Personas Como Sistema De Seguridad En Zonas Domiciliarias/Image Processing for identification of people as a security system in domiciliary zones.* Chimborazo: VI Congreso Internacional De La Ciencia, Tecnología, Emprendimiento E Innovación 2019.

Gutiérrez, A. (2022). *Estado del arte de registros biométricos estáticos usados para autenticar la identidad de una persona.* Repositorio Institucional de la Universidad Politécnica Salesiana.

Jaramillo, C. (2021, enero 13). *Utilización del sistema de reconocimiento facial para preservar la seguridad ciudadana*. Retrieved from El Criminalista digital: http://revistaseug.ugr.es/index.php/cridi/article/view/

Jawad, M., Sabat, H., & Fatlawi, A. (2023). *Hybrid approach for optimizing the face recognition based on SIFT, SURF and HOG features.* AIP Conference Proceedings.

Kpushik, D. (2022). *3D face recognition using a fusion of PCA and ICA convolution descriptors.* Neural Processing Letters.

López, Q., Alexandra, X., Yucta, H., Margoth, J., Amaguaya, V., & Bladimir, E. (2021). *Desarrollo de una aplicación móvil de reconocimiento facial utilizando el algoritmo Elastic Bunch Graph Matching para la Liga Deportiva Parroquial-Lican.* Chimborazo: Riobamba Universidad Nacional de Chimborazo.

Mínguez, T. D. (2021). *Visión artificial: Aplicaciones prácticas con OpenCV - Python.* Marcombo.

Mohammed, J., Abbas, E., & Abood, Z. (2023). *A facial recognition using a combination of a novel one dimension deep CNN and LDA.* Baghdad: Electrical Engineering Dept., University of Technology, Baghdad, Iraq.

Montes, A., Quetzalli, Salas, L., Orozco, S., Quintana, R., & Maricela. (2021). *Detección automatizada de emociones a través del rostro, una revisión del estado del arte.* Universidad Autónoma del Estado de México.

Muñoz, E. (2021). *Desarrollo de un sistema de control de acceso de personal empleando reconocimiento facial respaldado con técnicas de aprendizaje profundo.* Sangolquí: Universidad de las Fuerzas Armadas ESPE. Carrera de Ingeniería en Electrónica, Automatización y Control.

Quiñonez, Y., Lizarraga, C., & Aguayo, R. (2022). Machine Learning solutions with MediaPipe. *2022 11th International Conference On Software Process Improvement (CIMPS)* (pp. 212-215). Acapulco, Guerrero, Mexico: IEEE.

Roca, V. (2022). *Diseño del control automatizado para el seguimiento de personas por reconocimiento facial en tiempo real.* Valencia: Universitat Politècnica de València.

ROSMASTER-X3. (2022). *Yahboom*. Retrieved from ROSMASTER X3 repository: http://www.yahboom.net/study/ROSMASTER-X3

Ryan León, D. A., Carranza, R., & Lozano, C. (2022). *Development of a face detection system with or without a mask using software.* Trujillo: Universidad Privada del Norte.

Singh, S. H., & Neelima. (2022). *Face Detection and Recognition Using Face Mesh and Deep Neural Network.* Bengaluru: Department of Computer science and Engineering, Amrita School of Engineering,.

Zambrano, A., Orlando, L., & Lara, S. (2020). *Prototipo de reconocimiento facial para mejorar el control de asistencia de estudiantes en UNIANDES, Quevedo.* Quevedo: Uniandes.

Zhang, Y. (2022). *Applications of Google MediaPipe Pose Estimation Using a Single Camera.* Pomona: California State Polytechnic University, Pomona.