ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE MECÁNICA

CARRERA INGENIERÍA INDUSTRIAL

“IMPLEMENTACIÓN DE UN SISTEMA INCLUSIVO BASADO EN EL USO DEL BRAZO ROBÓTICO ANNO RV624 PARA ALIMENTAR A PERSONAS CON DISCAPACIDAD MOTORA”

Trabajo de Titulación

Tipo: Propuesta Tecnológica

Presentando para optar al grado académico de:

INGENIERO INDUSTRIAL

AUTOR:

ERICK SAMIR PÁSTOR JÁCOME

Riobamba – Ecuador

2022

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE MECÁNICA

CARRERA INGENIERÍA INDUSTRIAL

"IMPLEMENTACIÓN DE UN SISTEMA INCLUSIVO BASADO EN EL USO DEL BRAZO ROBÓTICO ANNO RV624 PARA ALIMENTAR A PERSONAS CON DISCAPACIDAD MOTORA"

Trabajo de Titulación

Tipo: Propuesta Tecnológica

Presentando para optar al grado académico de:

INGENIERO INDUSTRIAL

AUTOR: ERICK SAMIR PÁSTOR JÁCOME

DIRECTOR: Ing. EDUARDO FRANCISCO GARCÍA CABEZAS

Riobamba – Ecuador

2022

© 2022, Erick Samir Pástor Jácome

Se autoriza la reproducción total o parcial, con fines académicos, por cualquier medio o procedimiento, incluyendo la cita bibliográfica del documento, siempre y cuando se reconozca el Derecho de Autor.

Yo, **Erick Samir Pástor Jácome,** declaro que el presente trabajo de titulación es de mi autoría y los resultados del mismo son auténticos. Los textos en el documento que provienen de otra fuente están debidamente citados y referenciados.

Como autor asumo la responsabilidad legal y académica de los contenidos de este trabajo de titulación; el patrimonio intelectual pertenece a la Escuela Superior Politécnica de Chimborazo.

Riobamba, 09 de Marzo del 2022.

Erick Samir Pástor Jácome

060405279-5

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE MECÁNICA

CARRERA INGENIERÍA INDUSTRIAL

El Tribunal del Trabajo de Titulación certifica que: El Trabajo de Titulación; Tipo: Propuesta Tecnológica, **"IMPLEMENTACIÓN DE UN SISTEMA INCLUSIVO BASADO EN EL USO DEL BRAZO ROBÓTICO ANNO RV624 PARA ALIMENTAR A PERSONAS CON DISCAPACIDAD MOTORA",** realizado por el señor: **ERICK SAMIR PÁSTOR JÁCOME**, ha sido minuciosamente revisado por los Miembros del Tribunal de Trabajo de Titulación, el mismo que cumple con los requisitos científicos, técnicos, legales, en tal virtud el Tribunal Autoriza su presentación.

	FIRMA	**FECHA**
Ing. Eugenia Mercedes Naranjo Vargas **PRESIDENTE DEL TRIIBUNAL**		2022-03-09
Ing. Eduardo Francisco García Cabezas **DIRECTOR DEL TRABAJO DE TITULACIÓN**		2022-03-09
Ing. Carlos Santillán Mariño **MIEMBRO DEL TRIBUNAL**		2022-03-09

DEDICATORIA

Este trabajo de titulación de pre-grado va dedicado a mi familia que me apoyó y ayudó durante toda mi vida estudiantil y especialmente mi hijo que fue el motor principal para poder cumplir este objetivo.

Samir Pástor

AGRADECIMIENTO

Un especial agradecimiento a la Escuela Superior Politécnica de Chimborazo, a la Facutad de Mecánica y a la Escuela de Ingeniería Industrial por formarme a lo largo de todo este tiempo y brindarme la oportunidad de consolidarme como Ingeniero Indistrial.

A mi familia y amigos que me dieron su apoyo y soporte en los momentos más difíciles de mi vida.

Samir Pástor

TABLA DE CONTENIDO

CAPÍTULO III

ÍNDICE DE TABLAS

ÍNDICE DE GRÁFICOS

ÍNDICE DE FIGURAS

ÍNDICE DE ANEXOS

RESUMEN

En el siguiente trabajo se planteó la implementación un sistema inclusivo basado en el uso del brazo robótico ANNO RV624 para alimentar a personas con discapacidad motora con el propósito fundamental de vincular el conocimiento académico con la sociedad, brindando una solución a grupo de personas considerado vulnerable, como primer paso se determinó el funcionamiento y constitución del brazo robótico, para comprender el mecanismo de trabajo de sus actuadores, motores, componentes y controladores, para luego diseñar un algoritmo que servirá para la utilización de la visión artificial, con el objetivo de detectar y reconocer el área facial, específicamente la boca del individuo a alimentar, en esta etapa se utilizó una cámara para la adquisición de la información y se emplearán técnicas de visión por computadora, además como software principal Python, haciendo uso sus librerías Open CV y MediaPipe Face Mesh, que se basan específicamente en el análisis de imágenes del mundo real para poder ser traducidas a un lenguaje binario que puede ser comprendido por el computador y de esta forma ejecutar las acciones programadas en el algoritmo. En cuanto a la medición de tiempos según el rango de velocidad se demostró que la velocidad al 20% no es óptima para cumplir con la función de alimentación, ya que ocasionó la pérdida del alimento que recogía la cuchara y para una trayectoria corta generaba vibraciones en la mesa y desaceleraciones abruptas, que pueden generar daños físicos en la integridad de los servomotores del brazo robótico, por último antes de utilizar el brazo robótico ANNO RV624 se recomienda setear la posición HOME o posición cero, ya que este toma como posición inicial el último movimiento que se realizó antes de apagar el equipo.

Palabras clave: <SISTEMA INCLUSIVO> <BRAZO ROBÓTICO> <VISIÓN ARTIFICIAL> <PYTHON (SOFTWARE)> <ALGORITMO DE CONTROL>.

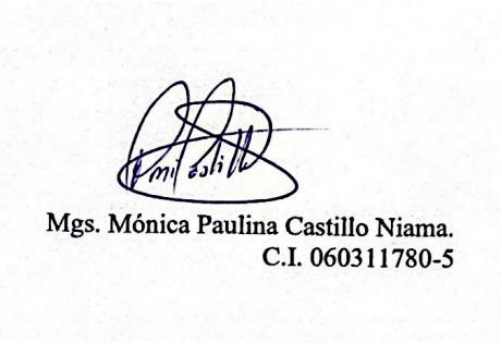
1968-DBRA-UTP-2022

ESPOCH - DBRAI
PROCESOS TÉCNICOS Y ANÁLISIS BIBLIOGRÁFICO Y DOCUMENTAL
DBRAI
03 OCT 2022
REVISIÓN DE RESUMEN Y BIBLIOGRAFÍA
Por: Rona Hora: 10:38

SUMMARY

In the present work the implementation of an inclusive system based on the use of the ANNO RV624 robotic arm to feed people with motor disabilities was proposed with the fundamental purpose of linking academic knowledge with society, providing a solution to a group of people considered vulnerable, as first step it was determined the operation and constitution of the robotic arm, to determine the working mechanism of its actuators, motors, components and controllers, and then design an algorithm that will serve for the use of artificial vision, in this stage a camera was used for the acquisition of information and computer vision techniques will be used, in addition as main software python, making use of its open cv and mediapipe face mesh databases, which are specifically based on the analysis of images of the real world to be translated into a binary language that can be understood by the computer and thus implement the actions programmed in the algorithm. As for the measurement of times according to the speed range, it was evidenced that the speed at 20% is not optimal to fulfill the feeding function, since it caused The loss of the food that the spoon picked up and for a short trajectory it generated vibrations in the table and abrupt decelerations, which can generate physical damage to the integrity of the servomotors of the robotic arm, finally before using the robotic arm ANNO RV624 it is recommended to set the HOME position or zero position, since this takes as initial position the last movement that was made before turning off the equipment.

Keywords: <INCLUSIVE SYSTEM> <ROBOTIC ARM> <ARTIFICIAL VIEW> <PYTHON (SOFTWARE)> <CONTROL ALGORITHM>.



Mgs. Mónica Paulina Castillo Niama.
C.I. 060311780-5

INTRODUCCIÓN

La alimentación es una actividad cotidiana que es indispensable para todo ser vivo en el planeta, y esta actividad puede resultar una tarea complicada para las personas con discapacidad que tienen inconvenientes de motricidad en sus extremidades, a tal punto de depender completamente de otras personas para poder ser alimentados.

De esta forma, el Consejo Nacional para la Igualdad de Discapacidades del Gobierno del Ecuador, indica que, en el país, hasta la fecha, existen alrededor de 475.166 personas con algún tipo de discapacidad motora, de las cuales más de la mitad poseen un grado de discapacidad muy alto (mayor a 50 %), estas personas por su condición deben tener un auxiliar de enfermería o personal de ayuda para realizar tareas cotidianas tales como (comer, ir al baño, cambiarse de vestimenta, cepillarse los dientes, etc.). (Consejo Nacional para la Igualdad de Discapacidades, 2021).

Así sabiendo el índice de personas con discapacidad motriz junto con el desarrollo integral de nuevas tecnologías en las áreas de Control, Automatización y Robótica, que permiten la obtención de nuevas herramientas que al ser aplicadas dan como resultados nuevos dispositivos tecnológicos como es el brazo robótico ANNO RV624, que junto a la visión artificial y un algoritmo de control, realizan el movimiento del mismo para proporcionar alimentos a dichas personas.

Basado en el principio de la visión artificial que dice que, se deben analizar imágenes del mundo real para ser traducidas a un lenguaje que sea comprendido por el controlador para luego ser procesadas y de esta forma obtener un proceso denominado reconociemito facial.

El reconocimiento facial ha sido utilizado en el mundo tecnológico durante años siendo una herramienta fundamental para la detección de movimiento, para esta trabajo se utilizó una cámara conectada al computador que realizará la captura de imágenes del usuario y para posteriormente realizar el movimiento adecuado en cada punto.

Para el desarrollo de la investigación se utilizará un procesador que se adapte a las condiciones para el funcionamiento y codificación del algoritmo de visión artificial para el brazo robótico ANNO RV624.

CAPÍTULO I

1 INTRODUCCIÓN

1.1 Antecedentes

Para la elaboración del presente trabajo de pregrado, el cual tiene por objetivo realizar un sistema inclusivo para poder alimentar a personas con discapacidad motriz utilizando el brazo ANNO RV624, es necesario citar investigaciones previamente realizadas con respecto al tema que se tiene en consideración, siendo estas las siguientes:

Según lo mencionado por (Naciones Unidas Cepal, 2012) , "en Latinoamérica cerca de 12% de la población latinoamericana viviría con al menos una discapacidad, lo que involucra aproximadamente a 66 millones de personas, según cifras recogidas de distintas fuentes estadísticas de la región."

Centro para el Control y la Prevención de Enfermedad, (2020) indica que la inclusión de personas con discapacidades en las actividades cotidianas conlleva prácticas y políticas diseñadas para identificar y eliminar barreras, como obstáculos físicos, de comunicación y de actitud, que dificultan la capacidad de las personas de tener una participación plena en la sociedad, al igual que las personas sin discapacidades.

De acuerdo con el trabajo realizado por (Carvajal Valencia, y otros, 2017 pág. 28) titulado "Empleo de la robótica en ayuda a las personas con discapacidad", mencionan que, el desarrollo de los proyectos tecnológicos con afines robóticos permite un gran avance en la sociedad, aportando de manera significativa en los sectores laborales, educativos e inclusive médicos, ayudando a personas que padezcan de alguna discapacidad.

Para el desarrollo de este proyecto tecnológico se realizó un análisis cuantitativo para recopilar información de los inventos que han sido creados a nivel mundial y se los separó según las necesidades de las personas y según el carácter regional, dando como resultado que la región con más inventos fue Europa y América del Norte, los cuales desarrollaron inventos para la discapacidad física.

En el trabajo de (Vallejo Yépez, 2018) titulado "Control bio-eléctrico de un robot para la asistencia de discapacidad de extremidades superiores" se desarrolló una interfaz para procesar, identificar, validar y controlar las señales electromiográficas superficiales receptadas, con el fin de brindar asistencia a personas con discapacidaf1d en las extremidades superiores para lograr alimentarse.

Al igual, Vallejo menciona que, "para clasificar los movimientos se utilizó una red Neuronal Artificial que permite activar los grados de libertad del robot. Para la interfaz entre el sistema y

el robot se utilizó un micro controlador, el cual genera las acciones de control para el robot, y envía datos a la herramienta Matlab Simulink para las pruebas HIL y SIL.” (Vallejo Yépez, 2018)

En el trabajo de titulación “Integración de un sistema robótico asistencial controlado mediante una interfaz cerebro computador para personas con discapacidad motora” realizado por (Jener, 2019 pág. 12), se menciona que, el empleo de un sistema robótico puede lograr devolver la autonomía parcial a una persona que tenga algún padecimiento de discapacidad motora que no le permite desarrollar las actividades cotidianas.

Este proyecto tuvo como objetivo desarrollar una interfaz cerebro computador que se integre con un sistema robótico, sin embargo, este tipo de estudio demanda de tiempo, por lo cual, se lo ejecutó en individuos sanos, obteniendo resultados satisfactorios indicando que con el empleo de estos sistemas robóticos se puede brindar la ayuda necesaria para personas con discapacidad y de esta manera ellos puedan realizar las tareas diarias.

Según (Triviño, 2018 pág. 14) en su trabajo titulado “Diseño e implementación de un manipulador robot de asistencia a personas con disfunción motora”, menciona que, el empleo de equipos especiales podría ser útil para la manipulación de objetos, de manera voluntaria y controlada, sin embargo, puede que existan dificultades al momento de levantar el brazo, esto debido a la precisión con la debería moverse.

Dentro de este proyecto se pueden encontrar dos procesos de estudio como es el desarrollo del software que analiza las imágenes del ojo y las reproduce por medio de un actuador, y el segundo proceso se encuentra enfocado a la mecatrónica del proyecto con el empleo de actuadores capaces de reproducir ciertas acciones.

En el trabajo de (Cushpa, 2020 pág. 60) titulado “Comunicación y virtualización de procesos industriales basados en Industria 4.0” describe que, muchas de las empresas requieren mejoras en la productividad, maximizando la eficiencia en los procesos y tener así una mayor rentabilidad, para lo cual emplean el uso de la tecnología, sin embargo, en Latinoamérica aún existen muchas deficiencias dentro de esta área.

Este proyecto tuvo como objetivo el desarrollo de un sistema virtual de comunicación para los procesos de carácter industrial, el cual está formado por Dockers, que a su vez contienen un sistema operativo virtual, en donde se ejecuta la aplicación Forte. Al igual, con el control de un Arduino, se construyó un prototipo de brazo robótico que envía los datos desde la Raspberry Pi para comprobar la funcionalidad de la aplicación.

Dentro del proyecto elaborado por (Sánchez Frías, y otros, 2016 pág. 24) titulada “Diseño y construcción de un brazo robótico antropomórfico de siete grados de libertad con análisis cinemático y

dinámico mediante algoritmos genéticos" se demuestra que el uso de algoritmos genéticos en un caso de ingeniería optimizan las aplicaciones, además el empleo de dichos algoritmos son adaptables para sistemas de control, como es el caso del brazo robótico.

Este proyecto tuvo como objetivo lograr la mayor flexibilidad, partiendo desde la configuración tradicional de los robots, ofreciendo mayores grados de libertad y con la ayuda de los algoritmos genéticos se puede solucionar las dificultades presentes en la dinámica y cinemática del brazo.

1.2 Planteamiento del problema

El principal problema que aqueja a las personas con discapacidad motora es la completa dependencia del factor humano para poder cumplir con sus necesidades básicas, como resulta ser la alimentación, además de esto el escaso apoyo de organizaciones gubernamentales en el área de ciencia y tecnología. Representando un gran obstáculo para el desarrollo de nuevas tecnologías que ayuden a facilitar la vida de estas personas, por lo tanto, es prioridad dar solución a estos problemas y como parte de esta solución es implementar un sistema inclusivo basado en el uso del brazo robótico ANNO RV624 para alimentar a personas con discapacidad motora.

El Grupo de Investigación de la Facultad de Mecánica – ESPOCH "Tecnologías de la información, comunicación y procesos industriales AUTOPRO" cuenta con un brazo robótico ANNO RV624 con 6 grados de libertad, el cual será proporcionado para el presente trabajo de investigación.

Es por ello, se considera importante investigar, desarrollar e implementar un sistema inclusivo para personas con discapacidad motora, este proyecto proporcionará un gran paso en la integración de este grupo vulnerable identificado en la sociedad.

En este estudio se aplicarán todos los parámetros necesarios que garantizarán la funcionabilidad optima del brazo robótico.

1.3 Justificación del proyecto

En la actualidad, el desarrollo tecnológico se ha centrado en combatir y eliminar problemas de carácter social y ayudar a personas con discapacidad motora a formar parte de este nuevo espacio tecnológico, cabe señalar que a lo largo del tiempo se han logrado grandes avances en esta área, pero aún se necesita una investigación más exhaustiva para que estas personas puedan mejorar su estilo de vida. Es por esto que el Estado ecuatoriano ha creado el Plan Nacional de Desarrollo 2017-2021 – Toda una Vida, el cual apuesta por el fortalecimiento y la institucionalización de políticas públicas y servicios que respondan a derechos fundamentales de las personas, en particular de grupos de atención prioritaria y en situación de vulnerabilidad (Senplades, 2017).
Al complementarla con la robótica representa una gran oportunidad para mejorar la calidad de vida de las personas y más aún aquellas personas que por cualquier motivo sufren de alguna discapacidad motriz que les impide realizar las actividades cotidianas con normalidad.

En base a lo antes expuesto, el presente trabajo de pregrado se lo realizara con el objetivo principal de implementar un sistema inclusivo basado en el manejo del brazo robótico ANNO RV624, mediante el diseño de algoritmos y la utilización de visión artificial para adquirir, procesar y analizar imágenes que facilitará el reconocimiento facial, con la finalidad de obtener datos que puedan ser traducidos y comprendidos por el sistema principal, para poder controlar dicho brazo y de esta forma poder cumplir con las funciones estipuladas en la investigación.

El sistema inclusivo basado en el brazo robótico ANNO RV624 dará solución a situaciones cotidianas, en las cuales el personal encargado de alimentar al paciente no pueda estar presente en el horario en el cual el paciente deba alimentarse, pudiendo generar algún tipo de trastorno alimenticio o enfermedad crónica.

Es necesario recordar que este sistema inclusivo no pretende reemplazar a un auxiliar de enfermería calificado, por el contrario, este sistema representa una herramienta de asistencia para mejorar el rendimiento del personal de la salud, en dónde la alimentación de un paciente se pueda hacer de forma remota desde alguna sala de control o de forma simultánea a dos o más pacientes.

1.4 Objetivos

1.4.1 Objetivo general

Implementar un sistema inclusivo basado en el uso del brazo robótico ANNO RV624 para alimentar a personas con discapacidad motora

1.4.2 Objetivos específicos

- Determinar el funcionamiento y constitución del brazo robótico ANNO RV624.
- Codificar un algoritmo de visión artificial para la identificación del área de la boca de la persona a alimentar.
- Vincular a un algoritmo de control la sección de visión artificial y el controlador del brazo robótico para el manejo del posicionamiento de este.
- Realizar pruebas de funcionalidad y determinar la eficiencia del sistema.

1.5 Alcance

La siguiente propuesta tecnológica va dirigida específicamente para las personas que cuentan con algún tipo de discapacidad motriz de sus extremidades, por ende la implementación del sistema inclusivo basado en el uso del brazo robótico ANNO RV624 para alimentar a personas con discapacidad motora tiene como propósito recopilar información del medio que rodea a los usuarios, permitiéndolo desarrollar una interacción entre las dos entidades (usuario-brazo robótico) y proponer asistencia a las personas con discapacidad que no pueden realizar movimientos en sus extremidades superiores e inferiores.

En dónde las funciones principales de la visión artificial son: realizar el rococimiento facial mediante es uso de una cámara, enfocándose específicamente de la zona boca mientras realiza los movimientos básicos de alimentación como son abrirla y cerrarla, además de reconocer el posicionamiento del brazo robótico mediante las distancias entre ejes centrales, usando círculos de colores en las principales articulaciones del mismo, para posteriormete con el uso de una interfaz diferente generar una secuancia que moverá el brazo para llevar el alimento del plato hacia la boca. Posteriormente en la etapa final se realizará pruebas de funcionalidad para determinar la eficiencia de los movimientos del brazo robótico para la alimentación del usuario.

CAPITULO II

2 MARCO TEÓRICO

2.1 Historia del robot industrial

La invención de los robots trae a consideración la antigua Grecia, en donde se presentaban máquinas móviles que presentaban objetos en movimiento, conocidos como autómatas. Sin embargo, no fue hasta el año 1937, cuando se fabrica el primer robot industrial, este facilitaba los trabajos de ubicación de productos, conocido como robot gargantúa, por su similitud con el brazo de una grúa.

Con este inicio y para el año de 1940, William Grey Walter, crea el primer robot autónomo, con este concepto George Davol, en 1950, fabrica y patenta el primer brazo programable; 6 años después, en 1956, Davol junto con Joseph Engelberger, abren paso a la comercialización de los robots industriales con la creación de la empresa Unimation.

Poco a poco la industria de la robótica fue creciendo, se incorporan los brazos robóticos en la empresa General Motors en el año de 1961, Europa instala el primer robot industrial en 1967, precisamente en Suecia.

El brazo robótico tuvo una gran evolución en 1969, volviéndolo articulado y controlado por computadora, gracias a investigadores de la universidad de Stanford, y lo bautizaron como brazo manipulador universal programable.

A principios de los años 70, la robótica había acaparado los mercados de las grandes potencias mundiales como Estados Unidos y Japón.

A finales de los años 80, Honda Motor Company y Mitsubishi Group, lideraban el mercado con la utilización de robots industriales en los procesos.

Desde los años 90 en adelante, la robótica ha ido creciendo de manera inmensurable, tanto, que para el año 2020 ya existan alrededor de 3 millones de robots en el mundo, con fabricantes como KAWASAKI, FANUC, EPSON, ABB.

2.2 Robot industrial

Dentro del campo industrial existen varios autores que tienen su propio concepto de lo que es un robot industrial, sin embargo, todos llegan a la misma conclusión, que los robots son máquinas que tienen la habilidad de mover elementos, por lo que se las considera como multifuncionales reprogramables.

Y de acuerdo a la Asociación de Industrial Robóticas (RIA) y la Organización Internacional de Estándares (ISO), un robot industrial es "un manipulador multifuncional reprogramables con varios grados de libertad, capaz de manipular materias, piezas, herramientas o dispositivos especiales según trayectorias variables programadas para realizar tareas diversas." (López Martínez, y otros, 2019 pág. 42)

Esta definición tiene una modificación por parte de la Federación Internacional de la Robótica (FIR) donde lo distingue como, "robot industrial se entiende a una máquina de manipulación automática, reprogramable y multifuncional con tres o más ejes que puede posicionar y orientar materias, piezas, herramientaso dispositivos especiales para ejecución de trabajos diversos en las diferentes etapas de la producción industrial, ya sea en una posición fija o en movimiento" (López Martínez, y otros, 2019 pág. 43)

2.2.1 Clasificación de los Robots Industriales

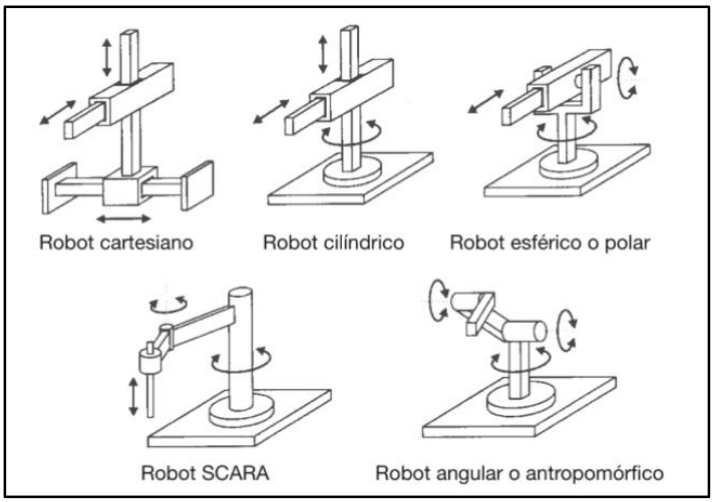


Figura 1-2: Clasificación de robots industriales
Fuente: (Navarro Piña, 2021 pág. 4)

2.2.1.1 Robot cartesiano

De acuerdo con Hernández & otros, el robot cartesiano "contiene tres grados de libertad y sus articulaciones son prismáticas en los tres ejes". (Hernández Ordoñez, y otros, 2015 pág. 47)

2.2.1.2 Robot cilíndrico

La característica principal, según Pérez & otros, de este tipo de robots es que "permiten movimientos en un perímetro circular alrededor de su base que se complementan con movimientos prismáticos para alcanzar cualquier posición dentro de un área de extensión circular". (Pérez Cisneros, y otros, 2014 pág. 144)

2.2.1.3 Robot esférico o polar

Este robot, a pesar de tener similitud en la configuración de los ejes, se diferencia del robot scara, debido a que este genera un movimiento esférico. De acuerdo con Romero (2018), "los robots esféricos disponen de dos ejes con movimiento de rotación perpendiculares entre sí y un tercer eje con movimiento lineal". (Romero Carrillo, 2018 pág. 139)

2.2.1.4 Robot SCARA

En base a lo dicho por Romero (2018) "la configuración que poseen estos robots es de dos juntas rotacionales y una prismática, sus movimientos se generan de manera rotacional y lineal, los ejes de rotación son paralelos entre sí, con lo que se consigue un espacio de trabajo cilíndrico en vez de esférico". (Romero Carrillo, 2018 pág. 139)

2.2.1.5 Robot articulado

Hernández & otros (2015), establecen que, este tipo de robots "se caracteriza por ser similar a un robot antropomórfico. Posee tres grados de libertad rotacional y un espacio de trabajo en tres dimensiones". (Hernández Ordoñez, y otros, 2015 pág. 148)

Como se menciona, los dos primeros movimientos son giratorios, pero únicamente sobre un plano, el primer movimiento dado por una columna que se apoya en la base, y el segundo movimiento en la articulación del brazo; por último, en el extremo del brazo se encuentra un eje deslizante que se mueve en el eje Z.

2.2.1.6 Robot paralelo

Según lo establecido por, un robot paralelo cuenta con "6 grados de libertad, que está formado por 6 actuadores rotatorios que proporcionan un movimiento rotacional a 6 manivelas unidas mediante untas esféricas a una barra de cada una, transmitiendo las barras el movimiento al elemento terminal mediante juntas cardan". (Asociación Española de Ingeniería Mecánica, 2016 pág. 30)

2.3 Brazo robótico

Navarro (2021), establece que este robot "tiene un estructura y movimiento similares a los de un brazo humano. Un brazo robótico antropomórfico se compone de una serie de eslabones o elementos que se mueven por medio de juntas giratorias." (Navarro Piña, 2021 pág. 4)

“Un brazo robótico es un dispositivo electromecánico diseñado con el fin de mover, agarrar o sostener objetos mediante el control y la automatización de los movimientos y sus articulaciones y eslabones”. (Alvarado Carrillo, y otros, 2019 pág. 26)

2.4 Discapacidad – paraplejia

De acuerdo con Kaplan & otros (2019), la paraplejia “se define como una debilidad motora de la extremidad inferior con una fuerza muscular más débil que la fuerza de la gravedad”. (Kaplan, y otros, 2019 pág. 418)

2.5 Python

En base a lo escrito por Guagliano (2019), “Python es un lenguaje de programación multiplataforma, consistente y maduro, utilizado por numerosas empresas internacionales”. (Guagliano, 2019)

El lenguaje de Python tiene varias características, lo que la hacen funcional en comparación con otras, este puede ser aplicado en diferentes plataformas, el código legible con el que se escribe permite la rápida interpretación y ejecución del mismo, y es multiparadigma, debido a que su lenguaje es imperativo y está orientado a objetos.

Existen más de 15 librerías empleadas dentro de esta plataforma, sin embargo, para el desarrollo de este proyecto se han tomado en consideración dos de ellas:

2.5.1 Open CV

La librería OpenCV, de acuerdo con Cea (2018), “es una librería de código abierto, que simplifica la implementación de prototipos apoyados por visión artificial, OpenCV cuenta con ciertas funciones principales que son la base para la utilización de la librería, tales como la lectura y escritura de imágenes, acceso a sus propiedades, etc. Y que estas características en conjunto con el lenguaje Python se utilizan para efectuar operaciones más complejas”. (Cea Hidalgo, 2018 pág. 15)

2.5.2 MediaPipe

Según Díaz (2021), MediaPipe “se trata de una solución actual para la detección de diversos objetos de los cuales se destacan caras y manos, utilizando técnicas más avanzadas de DL, esta emplea nodos que indica cuales son las entradas, salidas”. (Díaz Álvarez, 2021 pág. 20)

Muñoz (2021), establece que, “Media Pipe, es un marco de desarrollo (traducido del inglés, framework) que ofrece soluciones con aprendizaje automático y profundo en aplicaciones de

visión por computador y en tiempo real. Es multiplataforma y perfeccionado que no requiere un hardware de gran escala para correr sus modelos". (Muñoz Vega, 2021 pág. 80)

2.6 Visual Studio

"El IDE de Visual Studio es un panel de inicio creativo que se puede usar para editar, depurar y compilar código y, después, publicar una aplicación. Aparte del editor y el depurador estándar que proporcionan la mayoría de IDE, Visual Studio incluye compiladores, herramientas de finalización de código, diseñadores gráficos y muchas más características para facilitar el proceso de desarrollo de software". (Microsoft, 2022)

2.7 Controladores

Microsoft (2021), establece que "un controlador es un componente de software que permite que el sistema operativo y un dispositivo se comuniquen entre sí". (Microsoft, 2021)

Henríquez & Martínez (2019), menciona que, "los controladores ofrecen al usuario la capacidad de programar una determinada operación de modo que se realice de forma regular, un sistema que haya sido correctamente preparado hará que el sistema sea independiente de las perturbaciones externas". (Henríquez Novoa, y otros, 2019 pág. 38)

2.8 Sensores

2.8.1 Sensores de proximidad

Según lo establecido por Vallejo & Arias (2022), "este tipo de sensores no está diseñado para medir un valor específico de distancia, sino que, simplemente se activan ante la presencia cercana da un objeto. Esto se logra mediante un capacitor cuya capacitancia se ve afectada cuando un material penetra el campo eléctrico que este genera, lo cual es posible cuando el objeto que se acerca tiene una constante dieléctrica superior a la del aire". (Vallejo Valencia, y otros, 2022 pág. 33)

2.8.2 Final de carrera

"Este sensor es conocido también como sensor de contacto, el cual se lo considera como un interruptor que se acciona cuando un elemento móvil entra en contacto, por esta razón, se lo coloca en lugares estratégicos para determinar la posición exacta de dicho elemento". (Ingeniería Mecafenix, 2021)

2.9 Actuadores

De acuerdo con (Corona Ramírez, y otros, 2014 pág. 23) "un actuador es un dispositivo con la capacidad de generar una fuerza que ejerce un cambio de posición, velocidad o estado de algún tipo sobre un elemento mecánico, a partir de la transformación de energía."

Los actuadores se clasifican en dos grupos, que a su vez tienen sub clasificaciones:

- Por el tipo de energía utilizada
  - Neumático
  - Hidráulico
  - Eléctrico

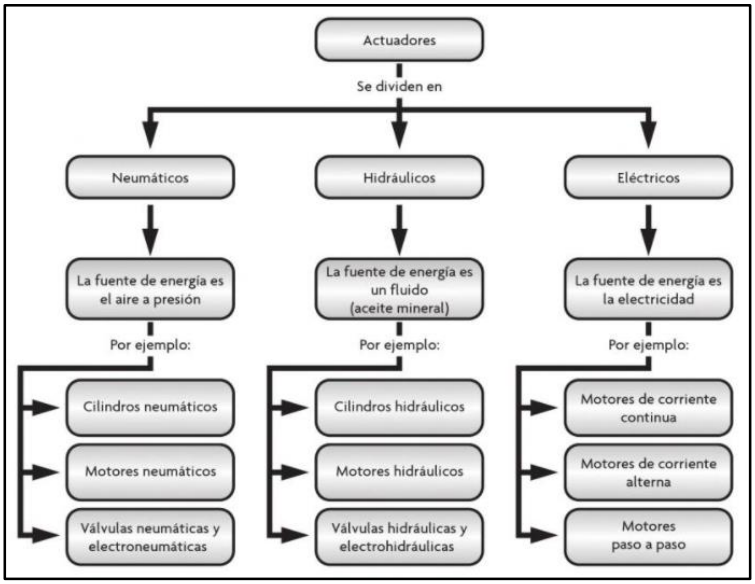


Gráfico 1-2: Clasificación de los actuadores

Fuente: (Corona Ramírez, y otros, 2014 pág. 26)

2.9.1 Actuadores eléctricos

2.9.1.1 Motor paso a paso:

Estos motores son capaces de convertir la energía eléctrica en movimiento angular, tiene el mismo principio de los motores de corriente alterna y directa, a diferencia que, "este tipo de actuador convierte una serie de impulsos eléctricos en desplazamientos angulares." (Corona Ramírez, y otros, 2014 pág. 33)

Los motores paso a paso, se caracterizan por su capacidad de mantener la posición indicada, esto gracias a la fabricación del mismo motor, donde el rotor está constituido por un imán permanente y el estator, está construido por bobinas.

2.9.1.2 Servomotores

De acuerdo con Zabala, el servomotor "es un pequeño, pero potente dispositivo que dispone en su interior de un pequeño con un reductor de velocidad y un multiplicador de fuerza". (Zabala, 2007 pág. 19)

Por otra parte, también se puede decir que es un actuador que posee en su interior un motor que cuenta con un redactor de velocidades y multiplicador de fuerza. En la gran mayoría de estos se cuenta con un ángulo de giro de 180º sin embargo también hay con giros de 360º.

2.10 Visión Artificial

La visión artificial, también conocida como visión por computadora o visión por ordenador, es una "disciplina que engloba todos los procesos y elementos que proporcionan ojos a una máquina, y que, mediante un proceso computacional, permite analizar e interpretar dichas imágenes" (González Marcos, y otros, 2006 pág. 11)

2.10.1 Historia

La visión artificial surge en los años 60, con la necesidad de analizar estructuras mediante el procesamiento de imágenes, por esta razón, se crea un prototipo automatizado, dividido en cámaras de visión y sistemas de procesamiento de imágenes. Sin embargo, la visión artificial toma impulso en los años 80, con el desarrollo de la informática, así, es como se ha ido desarrollando diferentes sistemas de visión artificial, cada vez más sofisticados, incluyendo la caracterización e interpretación de los objetos existentes dentro de una imagen.

2.10.2 Aplicaciones de la visión artificial

La visión artificial, en la actualidad, se ha visto incluida en numerosos campos, en donde facilitan la resolución de problemas, sin embargo, pueden existir limitaciones, estas varían por formas, texturas, inclusive técnicas de captación de imagen.

- **Agricultura:** Con el objetivo de reducir la aplicación de químicos en los cultivos se puede implementar aplicaciones tecnológicas, en donde la visión artificial cumple su papel detectando la maleza y plagas presentes en los cultivos.
- **Biología:** Mediante la captación de imágenes se pueden diferenciar las texturas, colores, tamaños, y así clasificar a las diferentes especies vegetales y animales.
- **Inspección y control de calidad:** Con la visión artificial se pueden identificar posibles fallas en la producción, detectando impurezas, apariencia, colores, y así, incrementar la calidad de los productos elaborados.

- **Medicina:** Se pueden trabajar con las imágenes resultantes de ecografías, radiografías, resonancia magnética, mamografías, tomografía, entre otros; con el fin de diagnóstico de ciertas patologías.
- **Meteorología:** Las condiciones ambientales se pueden analizar con claridad gracias al uso de la visión artificial, puesto que esta capta las variaciones de colores, dando a conocer los posibles fenómenos naturales, por medio de imágenes satelitales.
- **Modelado y visualización en 3D:** Se logra extraer la estructura geométrica y semántica de imágenes para reconstruir y crear automáticamente modelos 3D y sistemas de visualización interactiva.
- **Reconocimiento y clasificación:** En este ámbito se pueden reconocer objetos inmersos en las imágenes y su posterior clasificación, por ejemplo, el reconocimiento de rostros y la clasificación por edades, identificación de placas de vehículos, etc.
- **Robótica:** Este sistema de visión permite la navegación o el guiado automática de máquinas.
- **Seguridad:** A través de secuencias de imágenes se logra ejecutar vigilancia para detectar presencia y movimiento de cuerpos extraños, reconocimiento dactilar y ocular.

(La visión artificial y los campos de aplicación, 2015)

2.10.3 La visión artificial aplicada a la robótica

No se debe confundir al procesamiento digital de imágenes con la visión artificial, "cuando el PDI termina de optimizar la información de la imagen, la visión artificial entra en acción"

Los objetivos típicos de la visión artificial incluyen:

- Reconocimiento de ciertos objetos en imágenes, por ejemplo, rostros humanos.
- La evaluación de los resultados; por ejemplo, obtener las dimensiones de un objeto proveniente de una máquina CNC y comprobar que cumpla con las especificaciones de diseño.
- Registro de diferentes imágenes de una misma escena u objeto; por ejemplo, hacer concordar un mismo objeto en diversas tomas.
- Seguimiento de un objeto en una secuencia de imágenes. Se suele utilizar para medir posición, velocidad y aceleración.
- Mapeo de una escena para generar un modelo tridimensional; tal modelo puede ser usado por un robot para navegar por la escena.
- Estimación de las posturas tridimensionales de humanos.

(La Visión Artificial en la Robótica, 2007 pág. 26)

2.11 Etapas de un sistema de visión artificial

Como lo menciona Muñoz (2021), para la captación de las imágenes mediante visión artificial se emplean 8 etapas o procesos:

- Escenario analizar: Es el área que se desea capturar, donde se encuentra la información que busca analizar y procesar.
- Adquisición y digitalización de la imagen: En esta etapa se captura una proyección en dos dimensiones de la luz reflejada por los objetos de la escena, pasando a un formato digital comprensible para la pc.
- Preprocesamiento: Se realizan tareas de eliminación de ruido y/o realce de la imagen.
- Segmentación: detección de bordes y regiones. Permite separar los diferentes elementos de la escena. Extracción de características: Se obtiene una representación formal de los elementos segmentados en la etapa anterior.
- Reconocimiento y localización: Mediante técnicas, como la triangulación, se localiza al objeto en el espacio 3D.
- Interpretación: A partir de la información obtenida en las etapas previas y del conocimiento acerca del entorno se interpreta la escena y se de una aplicación de acuerdo a la información analizada.
- Aplicación: Es el objeto final que se controlará de acuerdo a la información que se procesó.

(Muñoz Vega, 2021 págs. 51-52)

2.12 Métodos para programar un robot industrial

2.12.1 Método de programación guiada

La programación guiada permite que el robot ejecute las actividades mediante la movilidad de las articulaciones, para posterior, realizar la actividad de manera automática, sin hacer uso de códigos.

De acuerdo con Guaraca & Ochoa (2015), "la programación consiste en la manipulación directa de los actuadores del robot mediante un control o maqueta del sistema, que permite posicionar al robot en diferentes puntos". (Guaraca Medina, y otros, 2015 pág. 16)

2.12.2 Método de programación textual

Esta programación hace uso de los diferentes lenguajes de programación, por medio de la repetición y uso de comandos verbales. Esta programación implica el "uso de órdenes dictados al sistema y ejecutados por el robot". (Guaraca Medina, y otros, 2015 pág. 18)

2.13 Marco legal

2.13.1 Constitución de la República del Ecuador

Art. 16.- Todas las personas, en forma individual o colectiva, tienen derecho a: 4. El acceso y uso de todas las formas de comunicación visual, auditiva, sensorial y a otras que permitan la inclusión de personas con discapacidad.

Art. 35.- Las personas adultas mayores, niñas, niños y adolescentes, mujeres embarazadas, personas con **discapacidad**, personas privadas de libertad y quienes adolezcan de enfermedades catastróficas o de alta complejidad, recibirán atención prioritaria y especializada en los ámbitos público y privado.

Art. 46.- El Estado adoptará, entre otras, las siguientes medidas que aseguren a las niñas, niños y adolescentes: 3. Atención preferente para la plena integración social de quienes tengan discapacidad. El Estado garantizará su incorporación en el sistema de educación regular y en la sociedad.

Art. 47.- El Estado garantizará políticas de prevención de las discapacidades y, de manera conjunta con la sociedad y la familia, procurará la equiparación de oportunidades para las personas con discapacidad y su integración social. Se reconoce a las personas con discapacidad, los derechos a: 2. La rehabilitación integral y la asistencia permanente, que incluirán las correspondientes ayudas técnicas.

Art. 48.- El Estado adoptará a favor de las personas con discapacidad medidas que aseguren: 1. La inclusión social, mediante planes y programas estatales y privados coordinados, que fomenten su participación política, social, cultural, educativa y económica.

2.13.2 Ley orgánica de discapacidades

Artículo 1.- Objeto. - La presente Ley tiene por objeto asegurar la prevención, detección oportuna, habilitación y rehabilitación de la discapacidad y garantizar la plena vigencia, difusión y ejercicio de los derechos de las personas con discapacidad, establecidos en la Constitución de la República, los tratados e instrumentos internacionales; así como, aquellos que se derivaren de leyes conexas, con enfoque de género, generacional e intercultural.

2.13.3 Reglamento de Régimen Académico Institucional

Artículo 2. Objetivos. a) Garantizar una formación de alta calidad que propenda a la excelencia y pertinencia de la educación politécnica, mediante su articulación a las necesidades de la transformación y participación social, fundamentales para alcanzar el Buen Vivir. b) Regular la gestión académica-formativa en todos los niveles de formación y modalidades de aprendizaje de la educación superior, con miras a fortalecer la investigación, la formación académica y profesional, y la vinculación con la sociedad.

CAPÍTULO III

3 METODOLOGÍA

Para la realización de este sistema inclusivo para alimentar a personas con discapacidad motriz utilizando el brazo robótico ANNO RV624 se cumplió con la siguiente metodología, primero se definen los requerimientos de hardware y software, luego se describe el proceso de diseño del algoritmo de visión artificial y el enlace con la secuencia del brazo robótico y por último se describen las pruebas para la obtención y análisis de los resultados.

En el gráfico 1-3 se muestra el diagrama de flujo del algoritmo metodológico que se va a seguir para la realización de este sistema inclusivo.

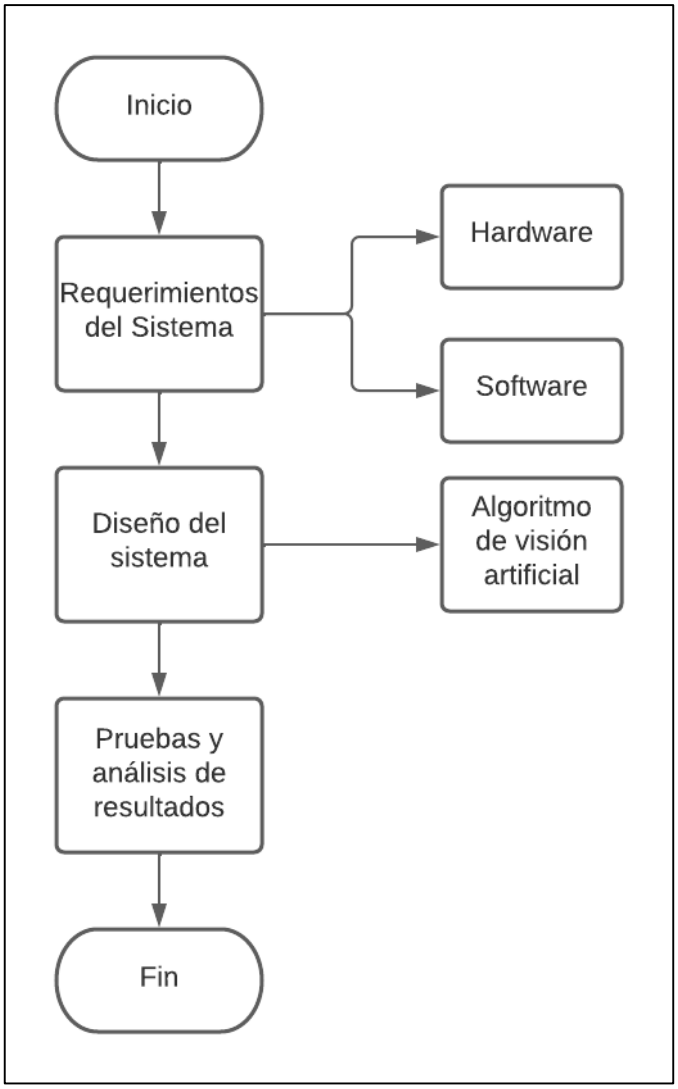


Gráfico 1-3: Diagrama de flujo del algoritmo metodológico

Realizado por: Pástor, S. (2021)

3.1 Definición de requerimientos de hardware y software

Se requiere brindar las facilidades necesarias en términos de control del sistema inclusivo para que el auxiliar o persona esté encargada de la alimentación del paciente pueda acceder de manera rápida y sencilla a la manipulación de esta. Para ello es necesario identificar los requerimientos del hardware y del software.

Para poder alimentar al paciente con discapacidad motriz se establece que debe haber una posición de reposo y una secuencia de alimentación, la secuencia de alimentación se encuentra plasmada en las tres dimensiones, para ello es necesario la creación manual de las secuencias que posteriormente se acoplan al sistema para alimentación del usuario que posee discapacidad motriz.
Las secuencias que se generan son las siguientes:

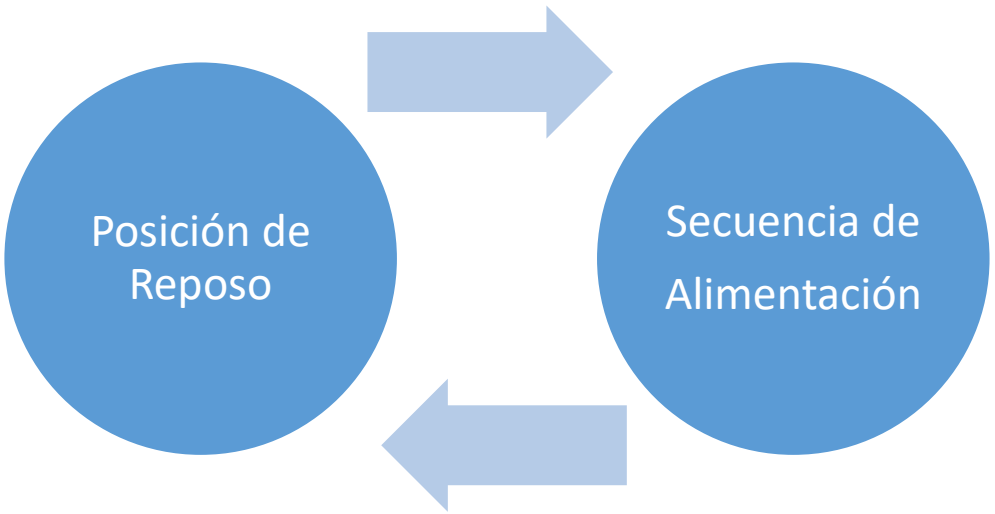


Gráfico 2-3: Estado de reposo y secuencia de alimentación.

Realizado por: Pástor, S. (2021)

Para la creación de las secuencias se utilizó el simulador Anno-VR que nos proporcionan las coordenadas del brazo, el programa de Robot AnnoSTM32.V2, que permite manipular el brazo robótico real.

La Secuencia de alimentación del individuo es fundamental conocer la ubicación de la boca del paciente con discapacidad motriz. Si el paciente decide parar o iniciar la secuencia el sistema de visión artificial detectara el patrón establecido y ordenara al brazo seguir la orden del usuario.

3.1.1 Hardware

Entendiendo que el proyecto está basado en el uso de un algoritmo de visión artificial se procedió a detallar los requerimientos de hardware para su correcto funcionamiento:

- Conocer los elementos que componen el brazo robótico ANNO RV624.
- Determinar cuál es el dispositivo para la obtención de imágenes para el reconocimiento facial.
- Detallar el tipo de controlador que se usará y los recursos necesarios para su funcionamiento.

3.1.2 Brazo robótico ANNO RV 624

El brazo robótico ANNO RV 624 es un robot de la línea industrial liviana que consta con seis grados de libertad, con un alcance de extensión máximo de 690 mm que está encargado de realizar el proceso de alimentación de un paciente con discapacidad motriz, para mejorar el entendimiento se puede observar en la figura 1-3 los motores y servomotores que constituyen el brazo robótico.

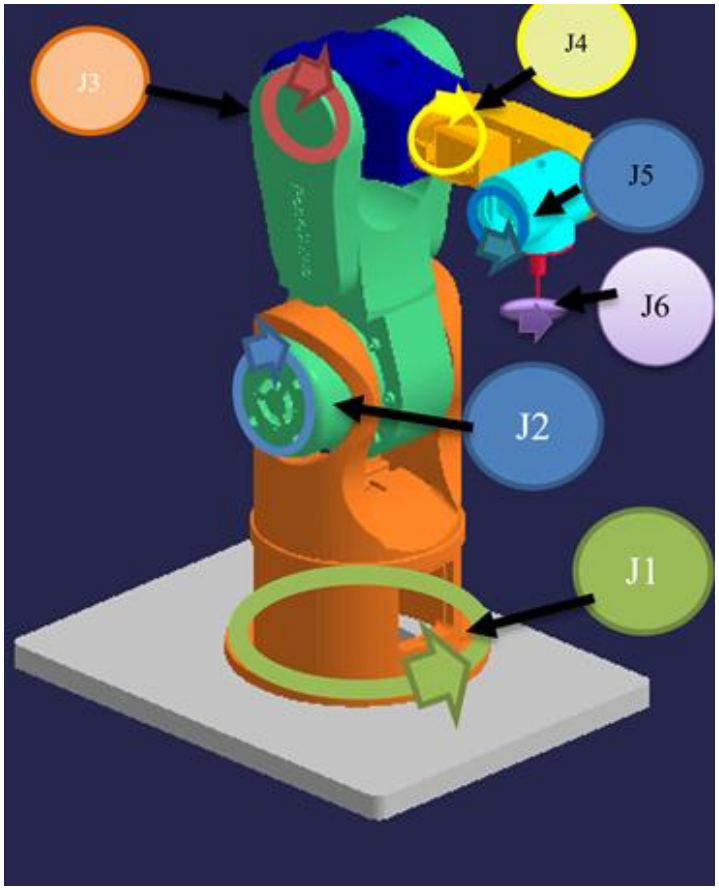


Figura 1-3: Ubicación de los componentes del brazo.

Realizado por: Pástor, S. (2021)

El brazo robótico está compuesto por seis motores paso a paso que le dan movimiento, también cuenta con un gripper que cumple con la función de sujetar objetos mediante una pinza, el material de construcción de los demás elementos es de aluminio.

A continuación como se observa en la tabla 1-3 se detalla cada uno de los motores paso a paso:

Tabla 1-3: Descripción de los motores paso a paso que contiene el brazo robótico

Motores paso a paso	Denominación	Características
Motor Servo integrado 57AIM30	J1	50W, 2.2-4.4A, 24-36v, 500KHz
Motor Servo integrado 60AIM25	J2	7A, 36V, 200 W
Motor Servo integrado 57AIM30	J3	50W, 2.2-4.4A, 24-36v, 500KHz
Motor Servo integrado 57AIM30	J4	50W, 2.2-4.4A, 24-36v, 500KHz
Motor Servo integrado 60AIM25	J5	Nema 34, 50w, 1.5-4.4 A, 24V, 500 KHz
Motor Servo integrado 42AIM15	J6	Nema 34, 100w, 1.5-4.4 A, 24V, 500 KHz

Fuente: Manual del robot ANNO

Realizado por: Pástor, S., (2021)

3.1.3 Descripciones técnicas de los servos motores integrados

3.1.3.1 Motor Servo Integrado 57AIM30

Como se puede observar en la tabla 1-3, el servo motor 57AIM30 está presente en el brazo robótico en 3 diferentes junturas como son: juntura 1 (J1), juntura 3 (J3) y juntura 4 (J4), a continuación se representa en la tabla 2 - 3 de especificaciones técnicas y en la figura 2-3 su estado físico.

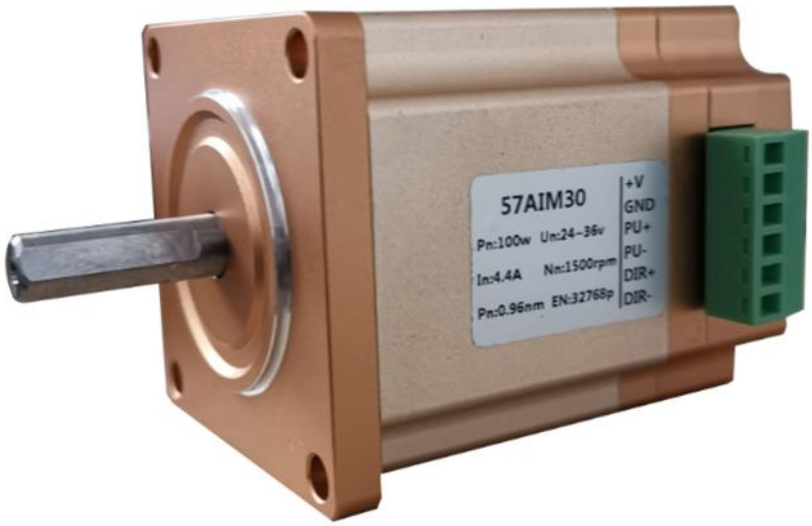


Figura 2-3: Motor Servo Integrado 57AIM30

Fuente: *(RobotAnno, 2021)*

Tabla 2-3: Especificaciones técnicas del servo motor 57AIM30

Fuente de Alimentación	**Voltaje**	**24VDC ±10%**
	Amperaje	**2.2A**
Parámetros del Motor	Esfuerzo de Torsión	**0.48NM**
	Velocidad Nominal	**1000RPM**
	Velocidad Máxima	**1500RPM**
	Potencia	**50W**
Método de Enfriamiento		**Enfriamiento natural**
Posición Modelo de control	Frecuencia máxima de pulso de entrada	**500KHz**
	Modo de comando de pulso	**Pulso + dirección, A + B**
	Relación engranaje electrónico	**Rango de Juego 1-65535:1-65535**
	Posición frecuencia de muestreo	**2KHz**
Función Protectora		**Alarma de Parada**
Puerto de Comunicación		**Easycan (CANBus de juguete,Velocidad 1M) Puerto Serial TTL(19200,8,N,1)(estado del motor del Monitor y parámetros de modificación)**
Medio Ambiente de uso	Temperatura Ambiente	**0~40º**
	Temperatura máxima del motor	**85º**
	Filtro HSensor de umidez	**5~95%**

Fuente: (Aliexpress, 2021)

Realizado por: Pástor S., 2021

3.1.3.2 Motor Servo Integrado 60AIM25

Como se puede observar en la tabla 1-3, el servo motor 60AIM25 está presente en el brazo robótico en 2 diferentes junturas como son: juntura 2 (J2) y juntura 5 (J5), a continuación se representa en la tabla 3-3 de especificaciones técnicas y en la figura 3-3 su estado físico.

Figura 3-3: Servo motor integrado 60AIM25

Fuente: (RobotAnno, 2021)

Tabla 3-3: Especificaciones técnicas del servo motor 60AIM25

Fuente de Alimentación	**Voltaje**	**36VDC ±10%**
	Amperaje	**7A**
Reductor	Relación de reducción	**50**
	Par nominal	**51NM**
	Ver par	**127NM**
Parámetros del Motor	Esfuerzo de Torsión	**1.91NM**
	Velocidad Nominal	**1000RPM**
	Velocidad Máxima	**1500RPM**
	Potencia	**200W**
Señal de retorno	**15 codificador de valor absoluto multigiro** **Una vuelta 15 Bits, varias vueltas 9 Bits, total 24 BIts (32768 pulsos porvuelta)**	
	De entrada máxima frecuencia de pulso	**500KHz**

Posición **Modelo de control**	Modo de comando de pulso	**Pulso + dirección, A + B**
Función Protectora		**Alarma de Bloqueo**
Puerto de Comunicación		**Easycan (CANBus de juguete,Velocidad 1M)**
Medio Ambiente de uso	Temperatura Ambiente	**0~40º**
	Temperatura máxima admisible	**85º**
	Humedad	**5~95%**

Fuente: (Aliexpress, 2021)

Realizado por: Pástor, S., 2021

3.1.3.3 Motor Servo Integrado 42AIM15

Como se puede observar en la tabla 1-3, el servo motor 42AIM15 está presente en el brazo robótico en la juntura 6 (J6), a continuación se representa en la tabla 4-3 de especificaciones técnicas y en la figura 4-3 su estado físico.

Figura 4-3: Servo motor integrado 42AIM15

Fuente: (RobotAnno, 2021)

Tabla 4-3: Especificaciones técnicas del servo motor 42AIM15

Fuente de Alimentación	**Voltaje**	**24~36VDC**
	Amperaje	**2.2A**
Parámetros del Motor	Esfuerzo de Torsión	**0.48NM**
	Velocidad Nominal	**1000RPM**
	Velocidad Máxima	**1500RPM**
	Potencia	**50W**
	Resistencia	**2.65Ω**
Método de Enfriamiento		**Enfriamiento natural**
Posición Modelo de control	De entrada máxima frecuencia de pulso	**500KHz**
	Modo de comando de pulso	**Pulso + dirección, A + B**
	Equipo electrónico relación	**Rango de Juego 1-65535:1-65535**
	Posición frecuencia de muestreo	**2KHz**
Función Protectora		**Alarma de Bloqueo**
Puerto de Comunicación		**Easycan (CANBus de juguete,Velocidad 1M)**
Medio Ambiente de uso	Temperatura Ambiente	**0~40º**
	Temperatura máxima admisible	**85º**
	Humedad	**5~95%**

Fuente: (Aliexpress, 2021)

Realizado por: Pástor S., 2021

3.1.4 Conjunto del cajetín de control

En el cajetín de control se dispone de distintos componentes electrónicos que al trabajar de manera unísona controlan el movimiento de los seis grados de libertad que tiene el robot ANNO RV 624, mediante la recepción de señales digitales o analógicas que accionan los motores paso a paso y actuadores para de esta forma obtener dicho movimiento.

Como se puede observar en la figura 5-3, los componentes que contiene la caja de control son los siguientes:

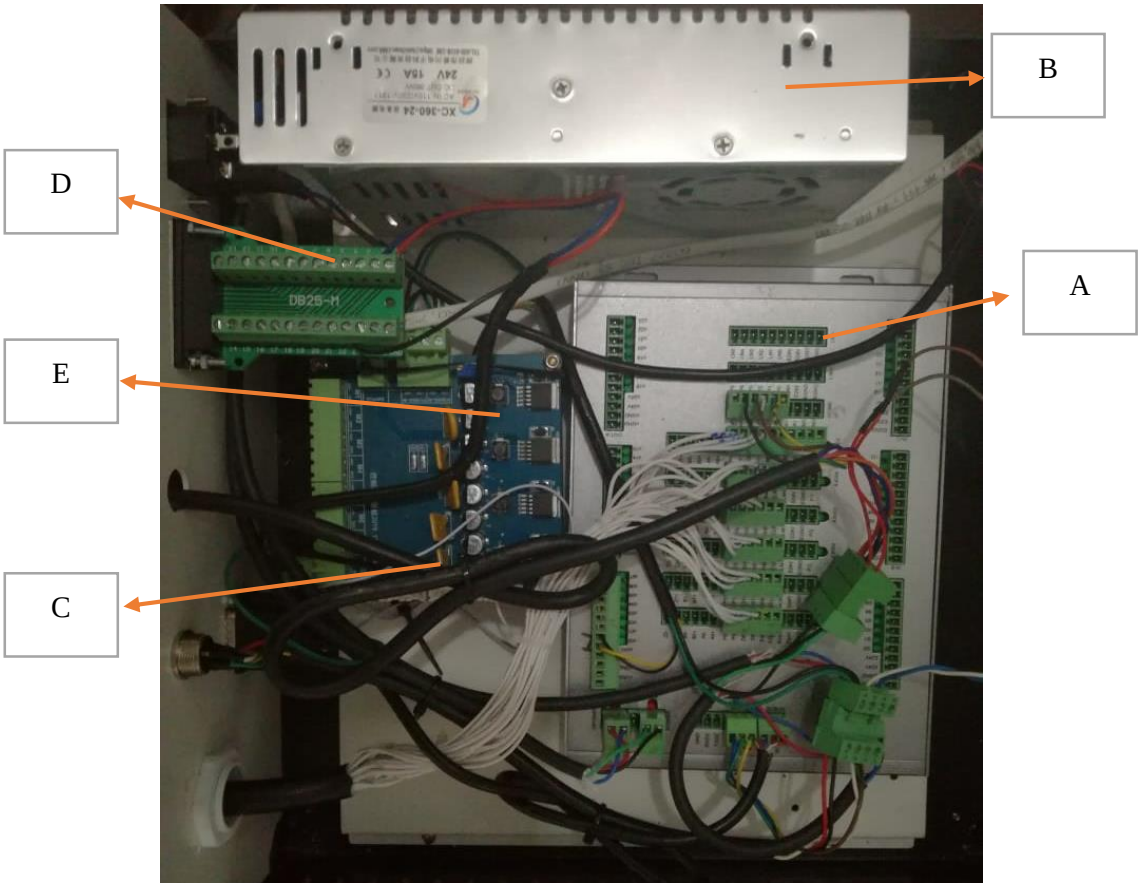


Figura 5-3: Cajetín de control

Realizado por: Pástor, S., (2021)

A. Controlador STM-32
B. Fuente de poder
C. Placa de relés
D. Placa de conexión para puerto paralelo
E. Regulador de voltaje

3.1.4.1 Controlador STM-32

El controlador STM-32 nos brinda un algoritmo de control que manipula los seis grados de libertad que posee en brazo ANNO RV 624, también incluye una interfaz API que nos permite enlazar las instrucciones y las muestras predeterminadas con nuevos algoritmos de control creados por el usuario, de esta forma se desarrolló un nuevo algoritmo de control para poder alimentar a personas con discapacidad motriz.

En la figura 6-3, se evidencia la constitución del controlador:

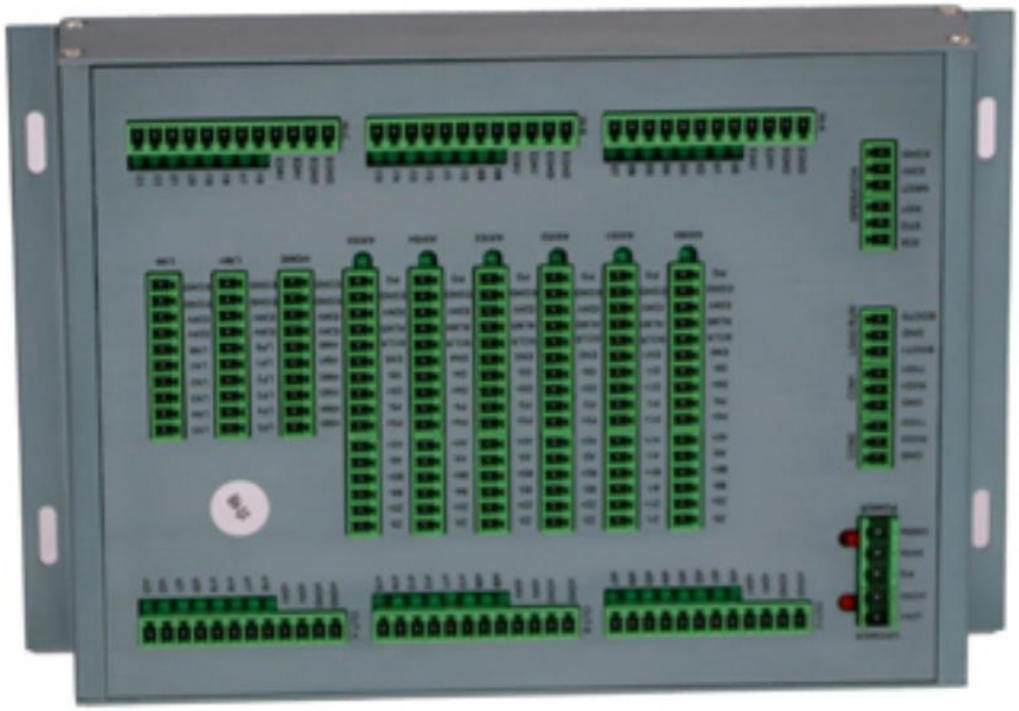


Figura 6-3: Controlador STM-32

Fuente: (RobotAnno, 2021)

3.1.4.2 Componentes del cajetín de control

Como ya se mencionó el cajetín de control dispone de varios elementos de control, que son una parte fundamental del robot ANNO Rv624 debido a que aquí llega toda la información para ser procesada y traducida a un lenguaje que entienda el brazo, por ende en la síguete tabla 5-3 se detalla los componentes electrónicos:

Tabla 5-3: Detalles de los elementos del cajetín de control

Elemento	Detalle
Fuente de poder	La fuente de poder es la que provee de energía al cajetín para que funcionen todos los elementos del cajetín de control y trabaja a 220 V y 360 W.
Regulador de voltaje	El regulador mantiene estables los picos límites del voltaje y asegura el correcto funcionamiento de todos los componentes.
Placa de relés	La placa de relés nos permite trabajar y controlar varias cargas con un mismo procesador, estas debido a su tamaño son compactas y muy útiles.
Placa de conexión para puerto en paralelo	Esta placa es la encargada de controlar la electroválvula que acciona al gripper, de esta manera las pinzas se pueden abrir y cerrar.

Fuente: Manual del Robot ANNO

Realizado por: Pástor, S., (2021)

3.1.5 Electroválvula Airtac 5/2

Esta electroválvula es una airtac 5/2 – 1/4 de simple solenoide con accionamiento y retorno eléctrico o por muelle, que ayuda a dirigir el flujo de aire que envía la mini bomba de una cámara a otra, esto permite abrir y cerrar el actuador del gripper, además de esto tiene posibilidad de montaje manifold, en la figura 7-3 para mejor comprensión se puede evidencia la constitución de la misma:

Figura 7-3: Electroválvula Airtac 5/2

Fuente: (Pneumatic, 2019)

Cabe recalcar que para el uso de la electroválvula está debe ir conectada al cajetín de control, para poder generar la apertura y cierre de la pinza.

3.1.6 Mini bomba ZR553PM

La función de la mini bomba es enviar aire que va a ser controlado por la electroválvula Airtac 5/2 por medio de dos mangueras plásticas hacia el doble actuador del gripper para mover las pinzas, en la figura 8-3 se muestra la mini bomba:

Figura 8-3: Mini bomba ZR3335PM

Realizado por: Pástor, S., (2021)

3.1.7 Software

Como parte de los objetivos fundamentales de este trabajo se propuso crear un algoritmo de visión artificial que reconozca el rostro de los pacientes y ubique las zonas de interés como lo son la boca y los labios.

Para el desarrollo del algoritmo de visón artificial se del sistema inclusivo para alimentar a personas con discapacidad motriz, se empleó Python con sus librerías Open CV y Media Pipe.

En este punto se describen las tareas realizadas para el desarrollo del algoritmo reconocimiento facial mediante la visión artificial que permita según los requerimientos planteados programar la secuencia de alimentación del paciente.

3.1.8 Lenguajes de programación y librerías

3.1.8.1 Python

Se consideró que el uso de Python como método de programación porque es un lenguaje de programación con un manejo sencillo, es el más popular y cuenta con un repositorio de librerías extenso que presenta compatibilidad entre sí, es de licencia libre, posee un gran apoyo por parte de la comunidad y se centra en un avance tecnológico presentando diferentes funciones.

3.1.8.2 Open CV

OpenCV es la librería más completa y tiene compatibilidad con Python, es el principal motor de la librería de MediaPipe porque trabajan conjuntamente con esta, además incluye las funciones de dibujo y segmentación dentro de la imagen, que permite trabajar con una cámara obteniendo imágenes en tiempo real, con videos guardados o imágenes.

3.1.8.3 MediaPipe

Los grandes avances que han realizado Google junto con otras grandes empresas han permitido perfeccionar la inteligencia artificial y la rama en que se fundamenta este proyecto es la Visión Artificial, por eso de esta forma el uso de esta librería se vuelve indispensable, ya que se encuentra centrado en dos de sus principales aplicaciones que son:

- MediaPipe Face Mesh : Alimentación y Condicionamiento
- MediaPipe Pose: Entrenamiento

MediaPipe Mesh trabaja con mallas triangulares y hace un cálculo de profundidad obteniendo valores en los tres ejes, en la figura 9-3 se presentan todas las funciones de MediaPipe:

	Android	iOS	C++	Python	JS
Face Detection	✅	✅	✅	✅	✅
Face Mesh	✅	✅	✅	✅	✅
Iris	✅	✅	✅		
Hands	✅	✅	✅	✅	✅
Pose	✅	✅	✅	✅	✅
Holistic	✅	✅	✅	✅	✅
Selfie Segmentation	✅	✅	✅	✅	✅
Hair Segmentation	✅		✅		
Object Detection	✅	✅	✅		
Box Tracking	✅	✅	✅		
Instant Motion Tracking	✅				
Objectron	✅		✅	✅	✅
KNIFT	✅				
AutoFlip			✅		
MediaSequence			✅		
YouTube 8M			✅		

Figura 9-3: Funciones de MediaPipe

Fuente: *(MediaPipe, 2020)*

3.1.9 Visión artificial

3.1.9.1 Visón artificial - alimentación

Para el desarrollo del algoritmo de reconocimiento facial se definió las zonas de interés que se encuentran involucradas en el momento de alimentar a una persona para luego poder analizarlas y obtener resultados.

La primera zona de interés que se reconoce es la de los labios puesto que nos permite visualizar si el paciente está masticando la comida, como se puede observar en la figura 10-3.

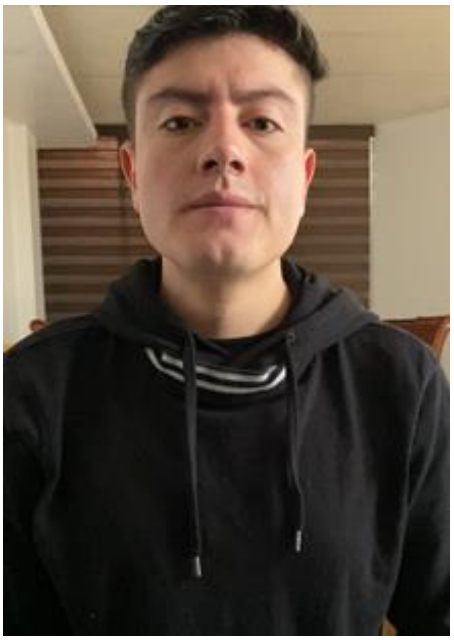


Figura 10-3: Reconocimiento de la zona de interés – labios

Realizado por: Pástor, S., (2021)

La segunda zona de interés que se reconoce es el interior de la boca debido a que permite visualizar si el paciente posee o no comida para poder seguir con la secuencia de alimentación o parar, como se observa en la figura 11-3:

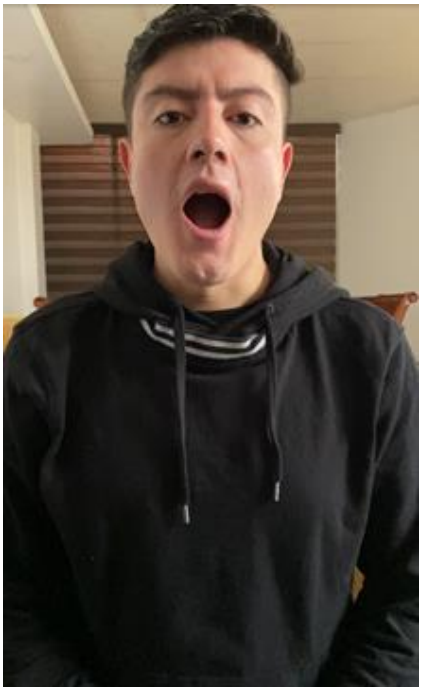


Figura 11-3: Reconocimiento zona de interés - Interior boca

Realizado por: Pástor, S., (2021)

3.1.9.2 Visión artificial – condicional

Una vez que se definió las zonas de interés labios y boca también se determinó un área mayor que corresponde al rostro completo, con la finalidad de realizar un pequeño condicional que muestre al algoritmo que el usuario se encuentra satisfecho, estos condicionales se representaron con el movimiento de la cabeza cuando afirmamos y negamos, es decir movimientos de arriba hacia abajo y movimientos de derecha a izquierda.

Figura 12-3: Condicional - Movimientos de afirmación y negación

Realizado por: Pástor, S., (2021)

Esta zona facial involucra todos los contornos del rostro ya que esto nos permitió mostrar las inclinaciones de la cabeza al momento de ejecutar los movimientos.
Es importante mencionar que el inicio y la pausa del proceso se lo ejecutan de esta manera porque este proceso es inclusivo para personas con capacidades especiales y discapacidades.

3.1.9.3 Visión artificial – entrenamiento

Este trabajo se basa en el movimiento de los brazos de manera natural, por ello se toma de referencia a como una persona que se alimenta independiente, analizando los movimientos de cada articulación al momento de tomar los alimentos y llevarlos hacia la boca, evitando que la extremidad superior bloquee la vista del paciente.

Figura 13-3: Autor alimentándose de manera independiente

Realizado por: Pástor, S., (2021)

3.1.10 Pasos para la generación de las zonas de interés

Para realizar la obtención de la zona del interior de la boca conjuntamente con la zona de los labios, se debe seguir la siguiente secuencia como lo determina el gráfico:

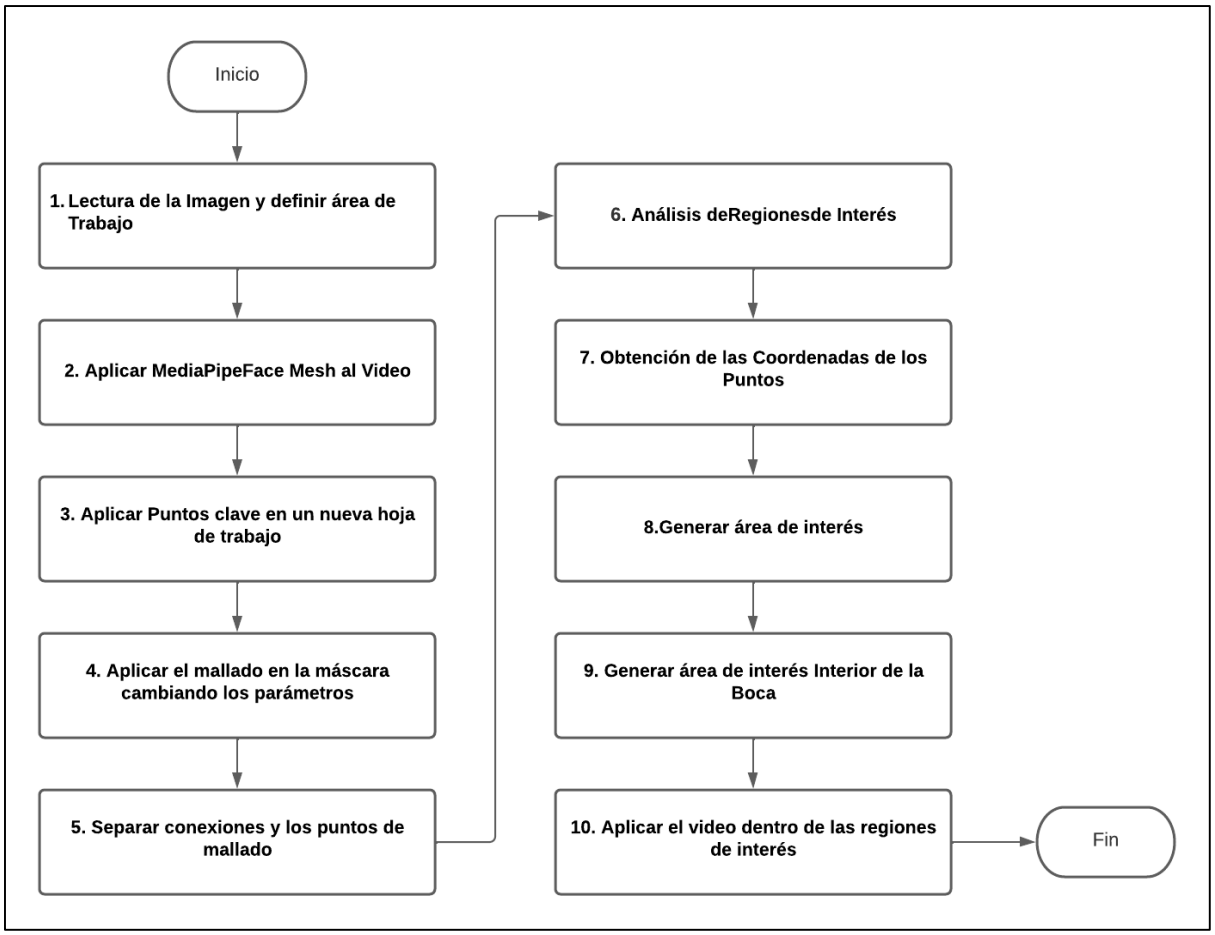


Gráfico 3-3: Diagrama de flujo de los pasos para generar zonas de interés

Realizado por: Pástor, S., (2021)

3.1.10.1 Lectura de la imagen y definir el área de trabajo

Controlamos que el formato de entrada del video es el adecuado, visualizamos el video, en el caso de que el área del trabajo es mayor a la resolución se recomienda realizar un escalamiento de la imagen para divisar los resultados de una mejor manera.

Figura 14-3: Área de trabajo

Realizado por: Pástor, S., (2021)

Luego de esto se realizó una revisión de la calidad de la imagen, obteniendo el Alto y Ancho de la imagen, lo definimos como el área de trabajo, es importante recordar que se trabaja con coordenadas y no con vectores, de la siguiente forma, para ver el código completo revisar el anexo F:

“#LECTURA DE LA IMAGEN
image = cv2.imread("Vid/M2.png")
#DEFINIR AREA DE TRABAJO
height, width, _ = image.shape
#VISUALIZACION DE LA IMAGEN
cv2.imshow("Image", image)”

3.1.10.2 Aplicar MediaPipe face mesh al video

MediaPipe Face Mesh trabaja con un mallado de 468 Puntos clave, y la configuración se la describe así:

“static_image_mode=True,
max_num_faces=1,
min_detection_confidence=0.5)”

La configuración de esta línea de código nos dice que cuando static_image_mode=true el detector de rostros será aplicado a todas las imágenes que se procesen dentro del video, de la misma forma solo se va a detectar un máximo de un solo rostro por imagen analizada y con el min_detection_confidence=0.5 controlamos que ese es el valor mínimo de confianza para que el modelo de detección facial sea considerado como exitoso.

Posteriormente activamos la visualización tanto de los puntos de la malla como de las conexiones, definimos los colores y el espesor de los puntos.

“for face_landmarks in results.multi_face_landmarks:
mp_drawing.draw_landmarks(image, face_landmarks,
mp_face_mesh.FACE_CONNECTIONS,
mp_drawing.DrawingSpec(color=(255, 0, 0), thickness=-1, circle_radius=1),
mp_drawing.DrawingSpec(color=(0, 255, 0), thickness=1))”

Después de este proceso de activación se observa los siguientes resultados:

Figura 15-3: Puntos de malla y líneas de contorno

Realizado por: Pástor, S., (2021)

3.1.10.3 Aplicar los puntos clave en una nueva hoja de trabajo

Se creó una hoja de trabajo con las características dimensionales (alto, ancho y canales), respetando las proporciones de la imagen del video de entrada mediante el comando np.zeros_like, de la siguiente manera:

```
“mascara = np.zeros_like(frame)
  cv2.imshow("mascara", mascara )
  cv2.imshow("Frame", frame)”
```

De esta forma así se visualiza conjuntamente con la imagen antes obtenida, como lo muestra la figura 16-3:

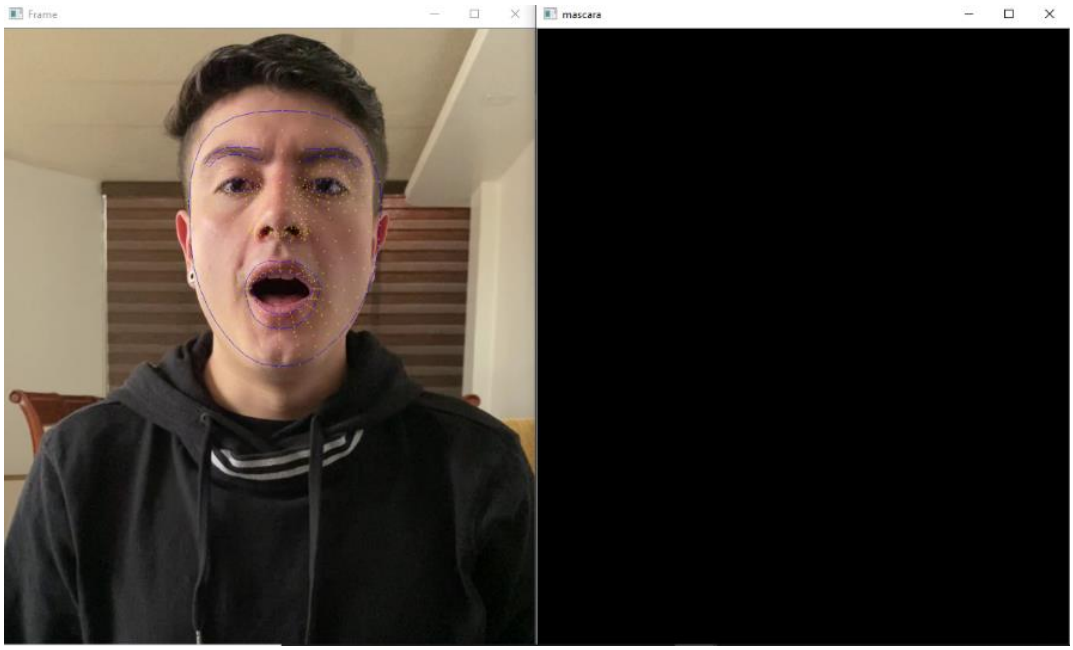


Figura 16-3: Imagen original y nueva hoja de trabajo

Realizado por: Pástor, S., (2021)

3.1.10.4 Aplicar el mallado en la máscara cambiando los parámetros de dibujo

En la nueva hoja de trabajo cambiamos los parámetros de dibujo que se encuentran directamente en el video de entrada, delimitando que los bordes serán de color azul y los puntos clave están en color amarillo:

"for face_landmarks in results.multi_face_landmarks:
mp_drawing.draw_landmarks(mascara, face_landmarks,
mp_face_mesh.FACE_CONNECTIONS,
mp_drawing.DrawingSpec(color=(0, 0, 0), thickness=1, circle_radius=1),
mp_drawing.DrawingSpec(color=(255, 0, 0), thickness=1))"

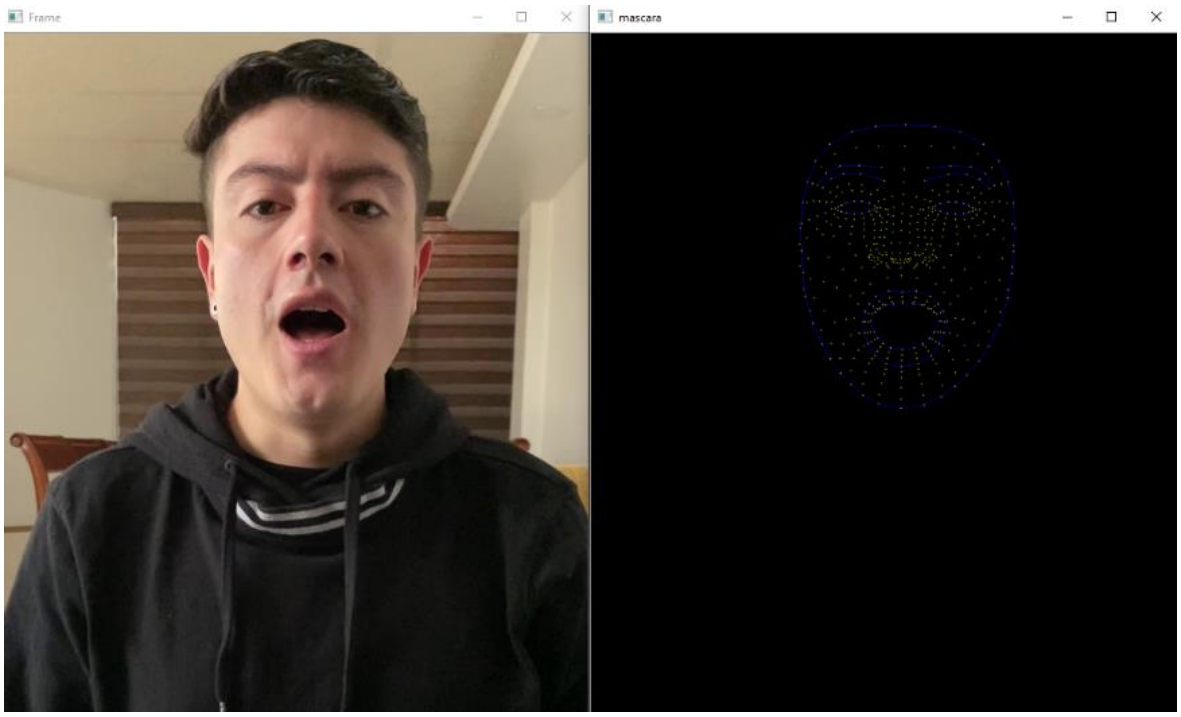


Figura 17-3: Líneas de borde y puntos clave en nueva hoja de trabajo

Realizado por: Pástor, S., (2021)

3.1.10.5 Separar las conexiones y los putos de mallado

Mediapipe nos ofrece visualizar las conexiones de los puntos clave, estas áreas que se generan son la de las cejas, los ojos y los labios (borde de bermellón) y la zona interior de los labios, para este caso en particular se utilizar solo las zonas del interior de la boca y los labios.

De esta manera al separar los puntos clave de los bordes se puede analizar las zonas de interés antes mencionadas, para mejor entendimiento se detalla este proceso en las figuras 18-3 y 19-3:

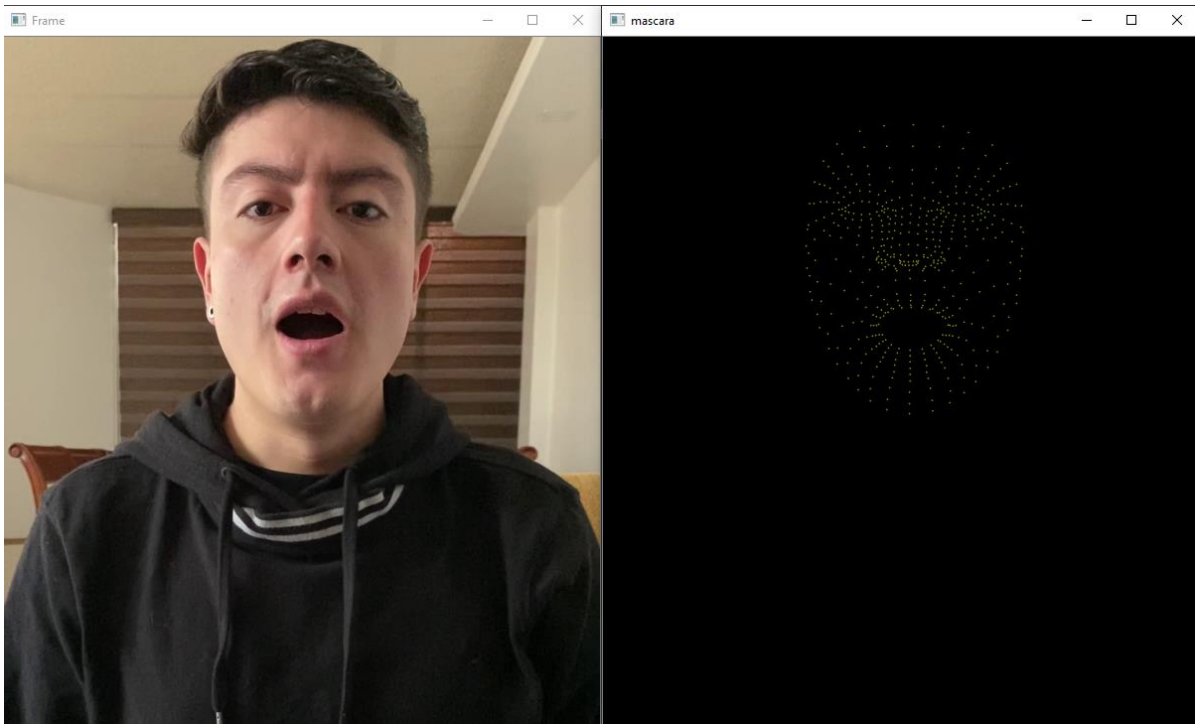


Figura 18-3: Visualización de los 468 puntos clave

Realizado por: Pástor, S., (2021)

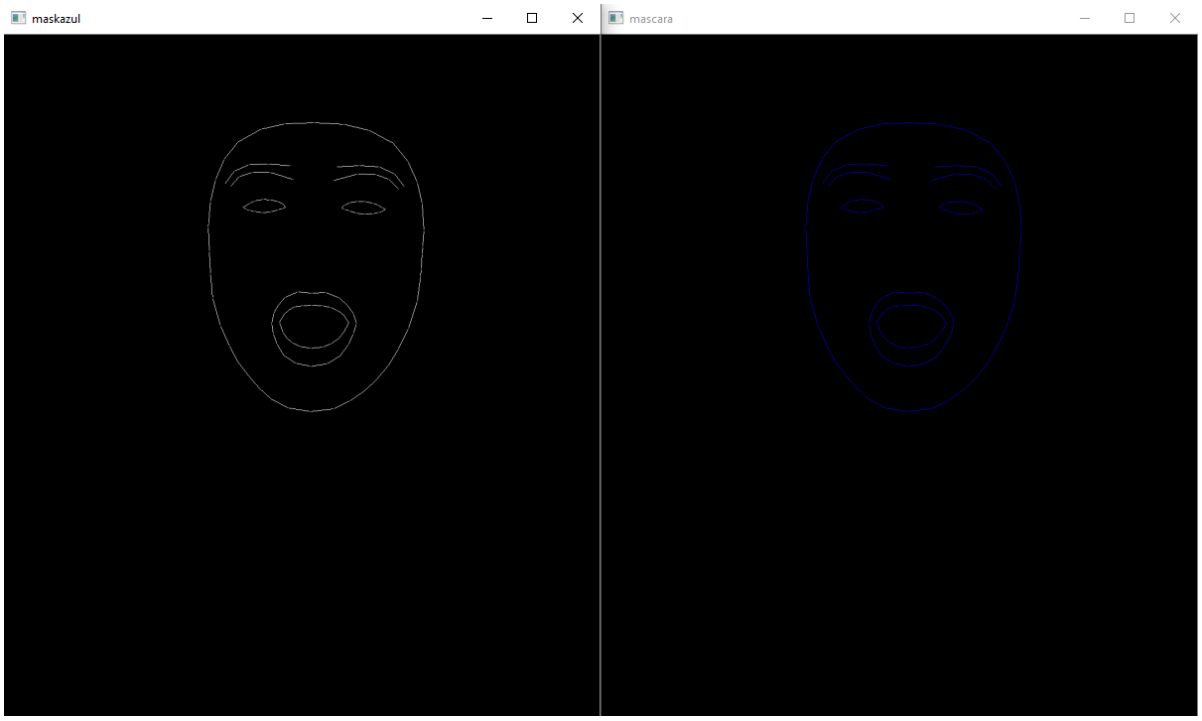


Figura 19- 3: Vizualización de los bordes

Realizado por: Pástor, S., (2021)

3.1.10.6 Análisis de las regiones de interés

Una vez que se separan los puntos clave y los bordes se estudian las conexiones que nos proporciona el programa, que son la base en la que se fundamentan los puntos claves, por ello se utiliza la función mascara, que incluye únicamente las conexiones de las zonas de interés que son necesarias.

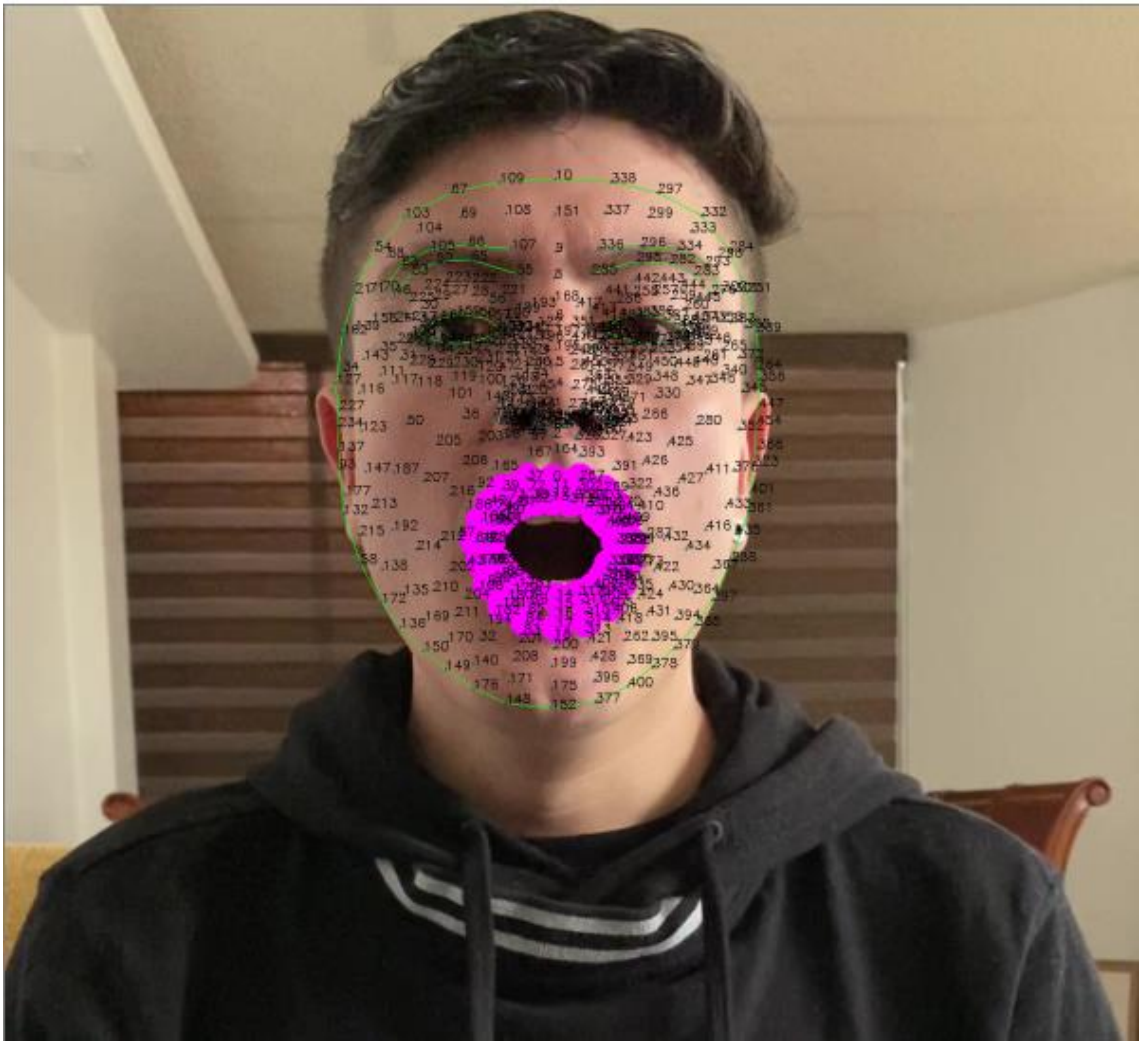


Figura 20-3: Enumeración de los puntos clave

Realizado por: Pástor, S., (2021)

Las limitaciones serán las mismas que nos brinda el programa, para el caso de las zonas laterales de la boca se debe delimitar el contorno de la misma para proceder a la obtención de los puntos en dónde se ubican estas zonas. Según la figura 20-3, el mallado que nos suministra Mediapipe, se observa que los puntos claves se encuentran numerados

El siguiente paso es tomar los puntos que intervienen en las zonas de estudio o de interés que van a ser analizadas para el proceso de alimentación de una persona con discapacidad motriz, como lo expresan las figuras 21-3 y 22-3:

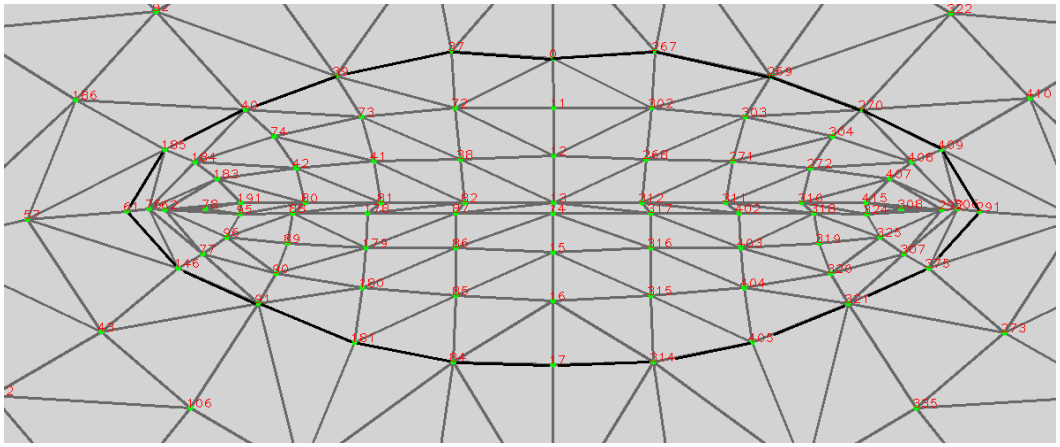


Figura 21-3: Contorno de los labios

Realizado por: Pástor, S., (2021)

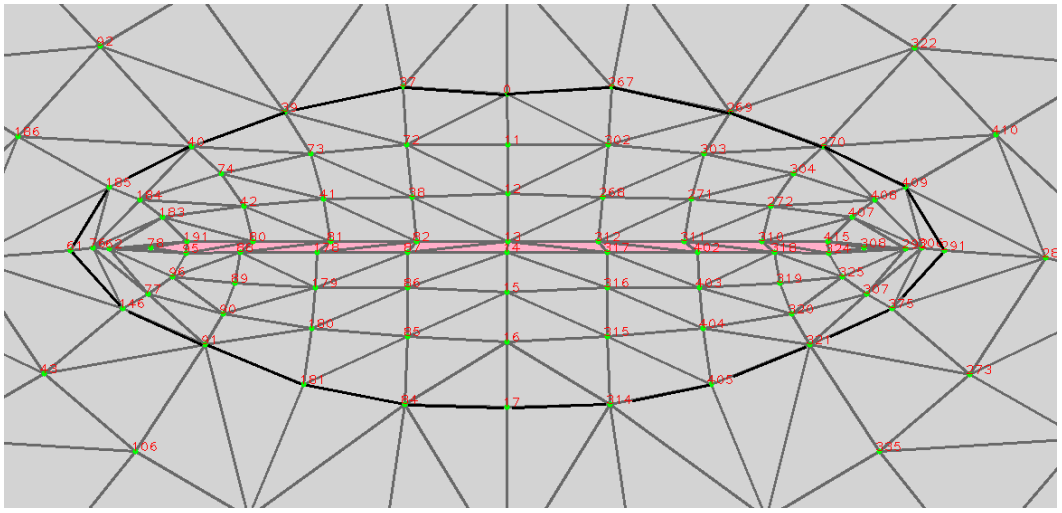


Figura 22-3: Contorno interior de la boca

Realizado por: Pástor, S., (2021)

Cuando ya se obtienen los puntos en dónde se ubican las zonas de interés se los almacena en una lista, y como resultado obtenemos que hay un total de 20 elementos para cada lista, en la siguiente figura 23-3 se observan las listas de los puntos:

```
Lista_labios = [291, 409, 270, 269, 267, 0 ,37 , 39 , 40 , 185  , 61, 146 , 91, 1
81,84 , 17,314,405,321, 375]
```

```
Lista_boca = [  415,310,311, 312,13,82,81,80,191,78 , 95 ,88 ,178 ,87 ,14,317,402
,318,324,308]
```

Figura 23-3: Lista de puntos del contorno de labios e interior de la boca

Realizado por: Pástor, S., (2021)

3.1.10.7 Obtención de las coordenadas de los puntos

La función "landmark", nos proporciona los valores de las coordenadas en los tres ejes tomando como referencia la altura, el ancho y la profundidad como unidad, de la siguiente manera como indica la figura 24-3:

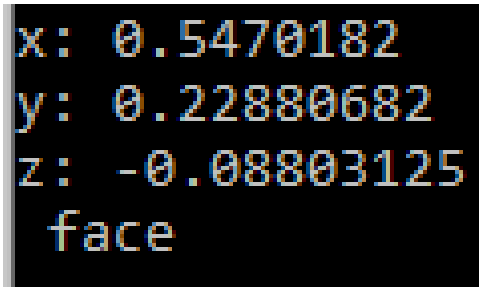


Figura 24-3: Coordenadas de ancho, altura y profundidad

Realizado por: Pástor, S., (2021)

Luego de obtener las coordenadas se crea una lista vacía al inicio del ciclo infinito, permitiendo que esta almacene los datos en tiempo real y se actualice constantemente con el comando:
"Lista2= []"

MediaPipe genera los datos en función de las proporciones de la imagen, ofreciendo un valor decimal, que se lo multiplica por el valor de la propiedad correspondiente, obteniendo un resultado en coordenadas dentro de la imagen, consecutivamente estos resultados se almacenan únicamente de las coordenadas en x, y, en 4l siguiente extracto se muestra la codificación para la obtención de coordenadas:

" for index in index_list2:
    x = int(face_landmarks.landmark[index].x * width)
    y = int(face_landmarks.landmark[index].y * height)
    qp2 = x,y
    Lista2.append(qp2)"

3.1.10.8 Generación de áreas de interés

Después de determinar los puntos de interés, generamos una zona binaria que sigue la sifuiente codificación, utilizando las listas con las coordenadas (x, y) y obteniendo los siguientes resultados expresados en las figuras 25-3 y 36-3:

```
#GENERACION DE UN ARRAY
area_pts = np.array([Lista2])
#CREACION DE UNA MASCARA NUEVA
imAux = np.zeros(shape=(frame.shape[:2]), dtype=np.uint8)
#DIBUJAR CONTORNOS EN LA MASCARA NUEVA
imAux = cv2.drawContours(imAux, [area_pts], -1, (255), -1)
```

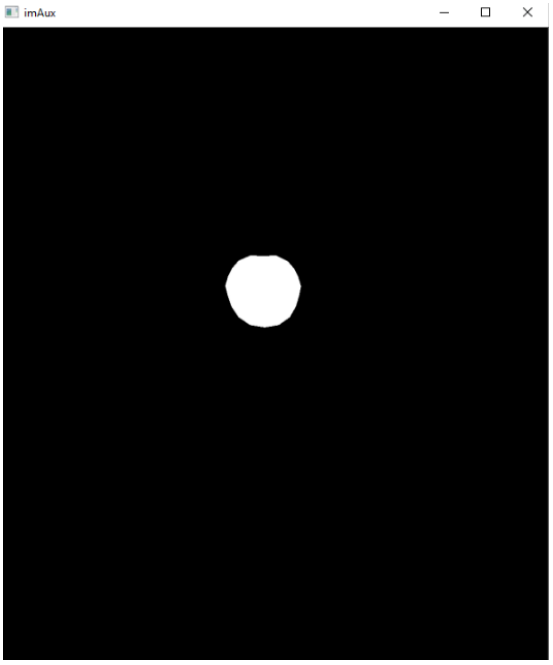


Figura 25-3: Zona de interés - contorno labios

Realizado por: Pástor, S., (2021)

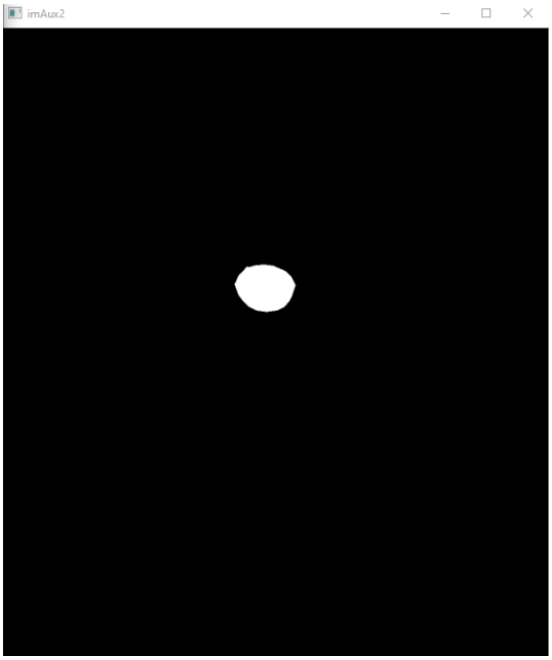


Figura 26-3: Zona de interés - interior boca

Realizado por: Pástor, S., (2021)

3.1.10.9 Generar área de interés del interior de la boca

Para generar en la zona de interés interior boca, se posee dos áreas, la primera área incluye los labios y la zona del interior de la boca, mientras que la segunda solo el interior de la boca.

Recordando que la primera área no incluye a la segundo área, ya que ambas son motivos de estudio de manera individual.

Como tenemos imágenes binarias (blanco y negro), se puede realizar operaciones lógicas entre ellas, delimitando que:

- A= zona de interés – contorno labios (figura 25-3)
- B= zonda interés – interior boca(figura 26-3)

En las siguientes figuras se demuestra el proceso de operaciones lógicas con las imágenes binarias, sabiendo que la figura 27-3 represneta a la zona de interés (A) que comprende los labios y el interior de la boca, la figura 28-3 es la negación de la zona B, onteniendo como resultado solo el interior de la boca (Not B) y por último la figura 29-3 es la operación lógica de la zona A y la negación de B, resultanco así el contorno de los labios (C)

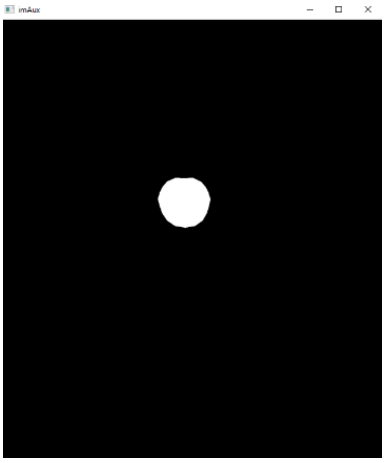


Figura 27-3: A

Realizado por: Pástor, S., (2021)

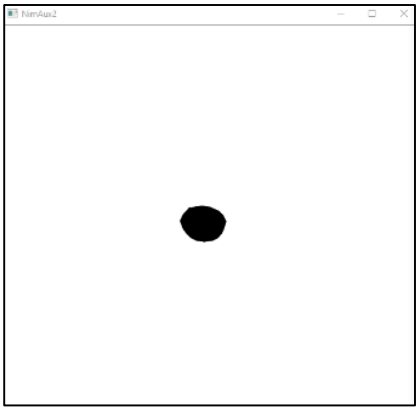


Figura 28-3: NOT B

Realizado por: Pástor, S., (2021)

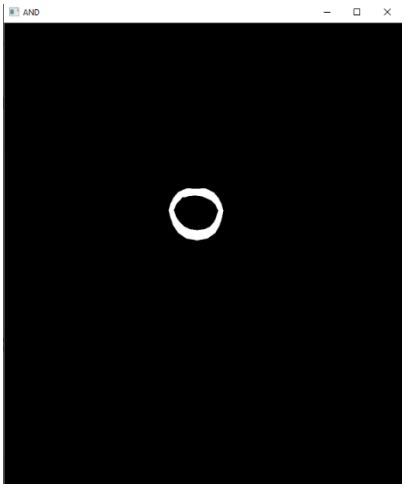


Figura 29-3: C = A and (NOT B)

Realizado por: Pástor, S., (2021)

3.1.10.10 Aplicar el video dentro de las regiones de interés

Se realizó una operación lógica con el uso de una máscara como indica la codificación, que la función de esta es transformar las imágenes binarias (figuras 28-3 y 29-3) en imágenes de la entrada original del video y se obtiene los siguientes resultados mostrados en la figura 30-3:
imAux2 = cv2.bitwise_and(frame, frame, mask=imAux2)

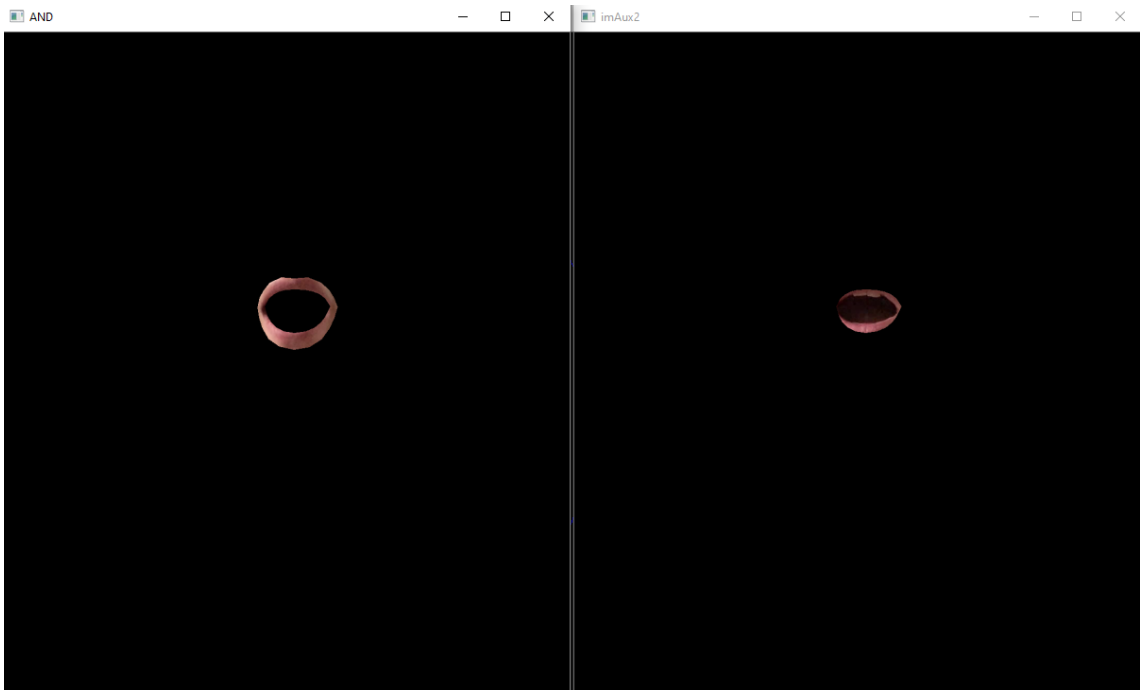


Figura 30-3: Resultado final de la extracción de las zonas de interés

Realizado por: Pástor, S., (2021)

3.2 Cinemática del Brazo ANNO RV624

La creación de la secuencia de alimentación de la persona se la realiza mediante el simulador para posteriormente trabajar directamente en el brazo. El brazo robótico la empresa RobotAnno se encuentra limitado por las especificaciones de los creadores por ello se trabaja con la información que nos proporcionan los fabricantes, entre estas nos proporciona los controladores y una guía de usuario en el manejo del brazo. También ofrecen un simulador que nos permite imitar los movimientos del brazo y observar su posición.

Con la finalidad de determinar el funcionamiento del brazo robótico ANNO RV624, se utilizó el simulador como una aplicación de introducción a la cinemática del brazo robótico y conjuntamente conocer las limitaciones físicas del brazo con respecto a su manipulación en los distintos ejes , además realizar las pruebas pertinentes sin poner en riesgo el estado físico del de brazo robótico.

El programa denominado ANNO-VRSimulater es el encargado de simular los movimientos del brazo y la interfaz gráfica que ofrece se representa en la figura 31-3:

Figura 31-3: Interfaz gráfica ANNO-VRSimulater

Realizado por: Pástor, S., (2021)

Como se visualiza dentro de la interfaz los idiomas disponibles son el idioma chino mandarín e inglés. Para una mejor comprensión de la aplicación que nos ofrece RobotAnno se detallara el funcionamiento del brazo mediante la interfaz.

La interfaz posee dos pestañas en la cual la primera sirve para mover el brazo mediante coordenadas y en la segunda permite la movilización mediante las articulaciones mecánicas del brazo.

A continuación se describe la primera pestaña que nos ofrece el simulador mostrado las interacciones posibles como lo representa la figura 32-3:

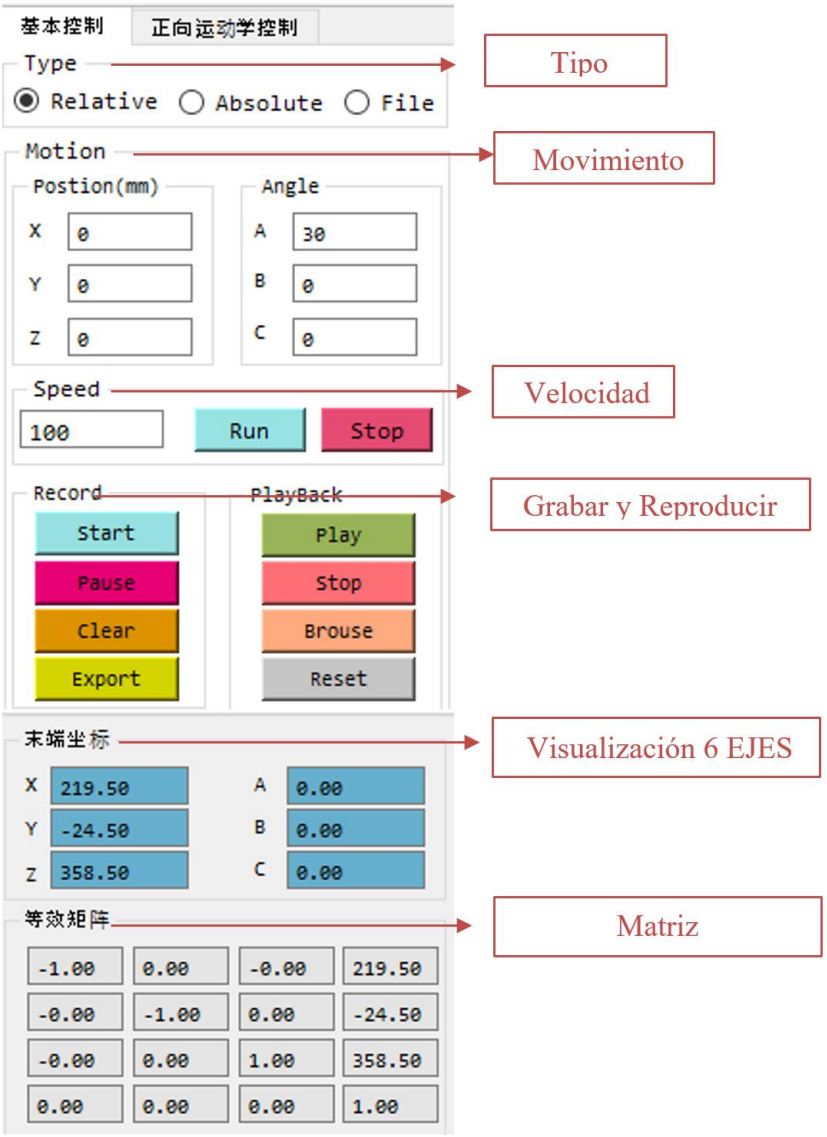


Figura 32-3: Primera pestaña de la interfaz del ANNO-VRSimulater

Realizado por: Pástor, S., (2021)

3.2.1 Posición inicial

Como se puede observar las coordenadas iniciales de todos los ejes del brazo están dados por las siguientes coordenadas, expresadas en la figura 33-3:

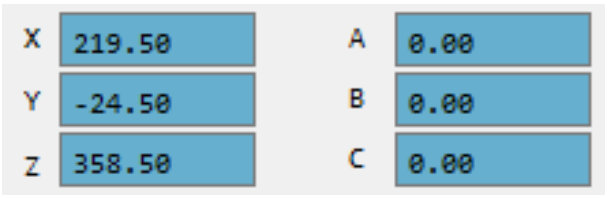


Figura 33-3: Coordenadas de posición inicial

Realizado por: Pástor, S., (2021)

Cabe recalcar que al momento de encender el robot se debe tomar en cuenta las situaciones en las que se encuentra el Robot ya que si se realiza un movimiento brusco generará un daño físico en el equipo, además antes de manipular el equipo se debe setear la posición inicial para que cumpla con las funciones requeridas.

3.2.2 Tipos de movimientos

Las opciones de movimiento con las que cuenta el brazo robótico ANNO RV 624 se clasificaron en 4 grupos que se demuestran en el gráfico 4-3:

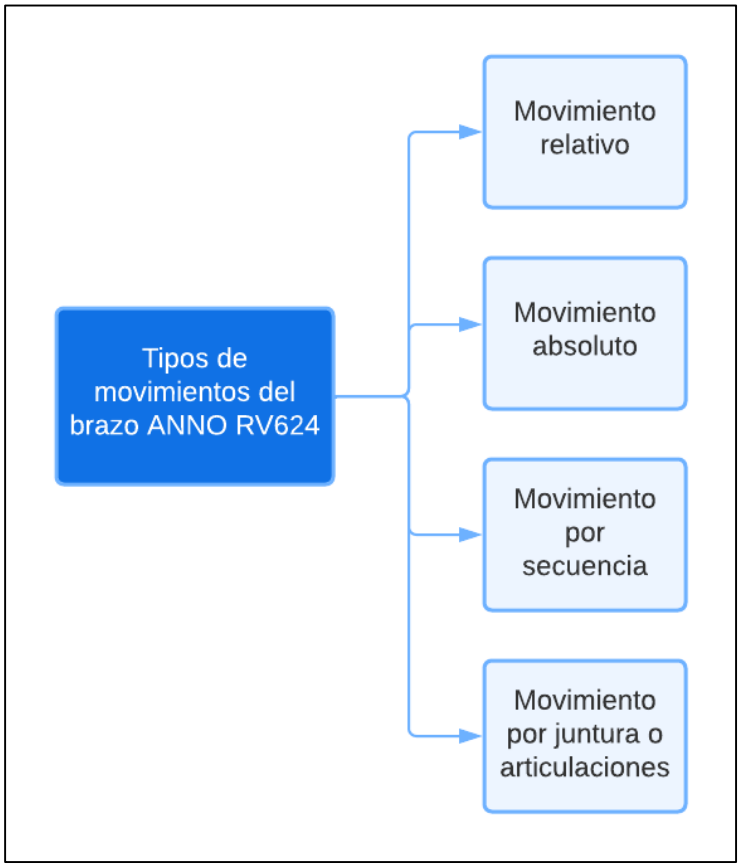


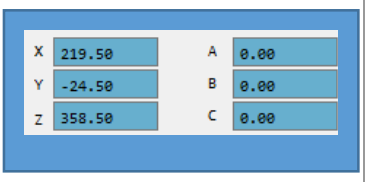
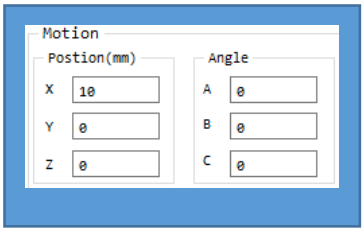
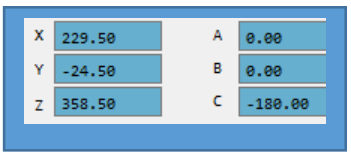
Gráfico 4-3: Diagrama del tipo de movimientos

Realizado por: Pástor, S., (2021)

3.2.2.1 Movimiento relativo

El brazo ANNO RV 624 se desplazará el valor exacto que se ingrese en los cuadros de dialogo del apartado Motion teniendo la siguiente interacción expresada en la tabla 6-3:

Tabla 6-3: Ingreso de datos para la interacción en el apartado Motion

Posición Inicial	Acción	Posición Final
		

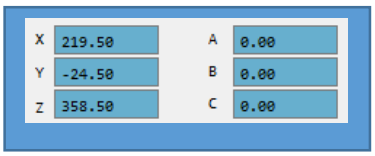
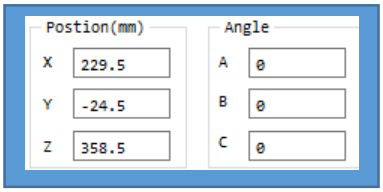
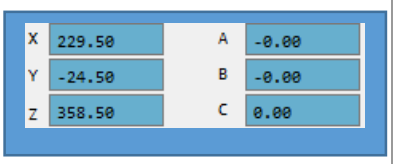
Realizado por: Pástor, S., (2021)

Siguiendo los parámetros ingresados en los cuadros de diálogo podemos observar que la acción a ejecutar es desplazar el brazo robótico 10 mm en el eje X con un valor positivo y la posición final concuerda con la acción ejecutada.

3.2.2.2 Movimiento absoluto

Para utilizar la función de movimiento absoluto, el brazo robótico se desplazará a la posición final que se ingrese en el los cuadros de diálogo del apartado de Position, como se detalla en la tabla 7-3:

Tabla 7-3: Ingreso de datos para la ejecución del movimiento absoluto

Posición Inicial	Acción	Posición Final
		

Realizado por: Pástor, S., (2021)

3.2.2.3 Movimiento por secuencia

El brazo robótico ANNO RV624, reproducirá una secuencia almacenada en lenguaje de programación Gcode y su respectivo archivo debe contar con un tipo de extensión ".nc, .gcode o .csv" para que el brazo puede ejecutar los movimientos programados.

La opción de grabar y reproducir dentro del apartado del simulador también ayuda a crear una secuencia para posteriormente reproducirla.

3.2.2.4 Movimiento por junturas o articulaciones

El simulador *ANNO-VRSimulater* brinda otra pestaña donde se puede manipular el brazo por cada una de las articulaciones o junturas de los elementos que lo constituyen, con el uso barras de seguimiento como se observa en la siguiente figura 34-3:

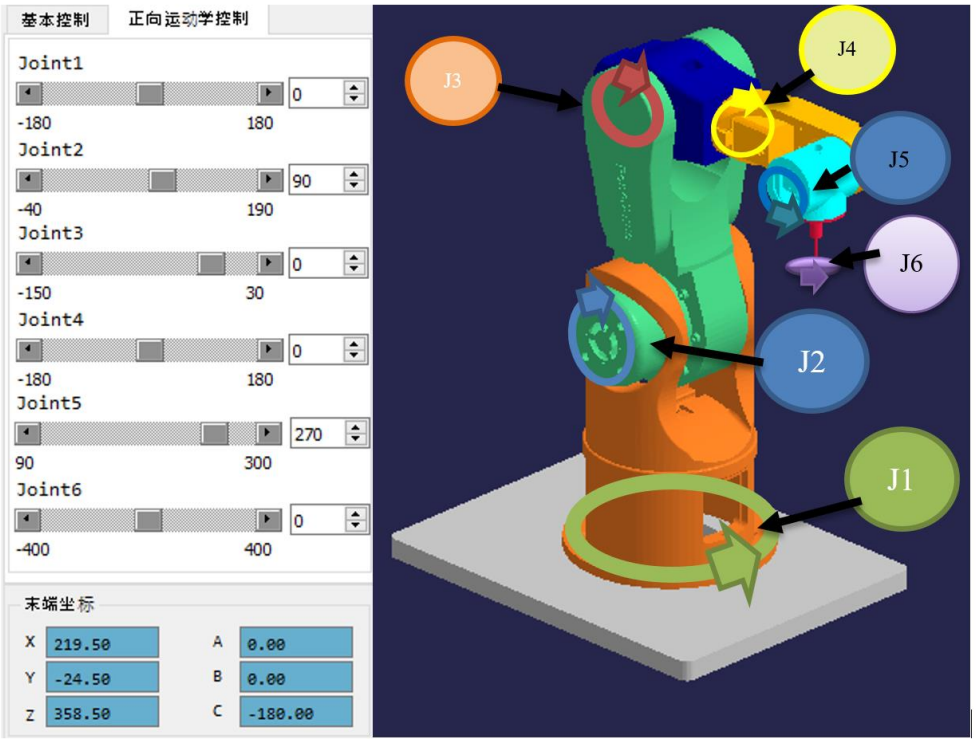


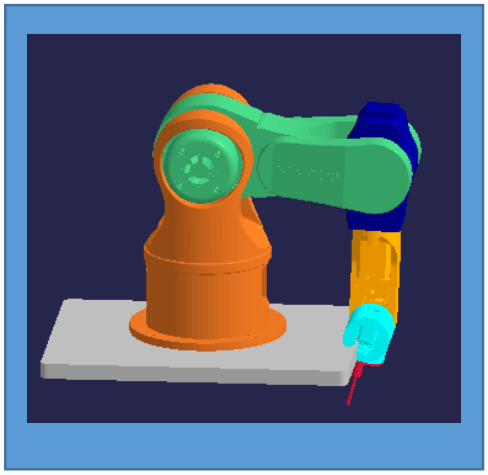
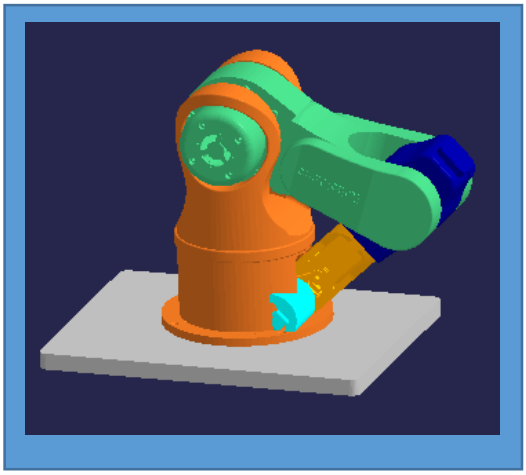
Figura 34-3: Interfaz de movimiento por articulaciones

Realizado por: Pástor, S., (2021)

3.2.3 Creación de limitación en el movimiento del brazo robótico ANNO RV624

La cinemática del brazo permitió realizar varias combinaciones de movimientos pero algunas de estas pueden ocasionar que se comprometa el estado físico del brazo y la seguridad integral del usuario, el simulador nos muestra gráficamente cuál es el resultado en el caso de una mala manipulación del brazo robótico y se lo especifica en la tabla 8-3:

Tabla 8-3: Colisiones del brazo robótico

Brazo colisiona contra la mesa	Brazo colisiona contra sí mismo
	

Realizado por: Pástor, S., (2021)

3.3 Visión artificial de la cinemática del brazo

3.3.1 Posición del brazo en sus dimensiones

Después de definir el funcionamiento y constitución del brazo robótico ANNO RV 624, se procede a encontrar las posiciones que maneja cada articulación para ello se procedió a agregar marcas de colores en las articulaciones, que ayudan a determinar la posición espacial y sus ángulos en lo que se encuentra el brazo, a continuación se presenta un diagrama de flujo de proceso de dimensionamiento del brazo en el gráfico:

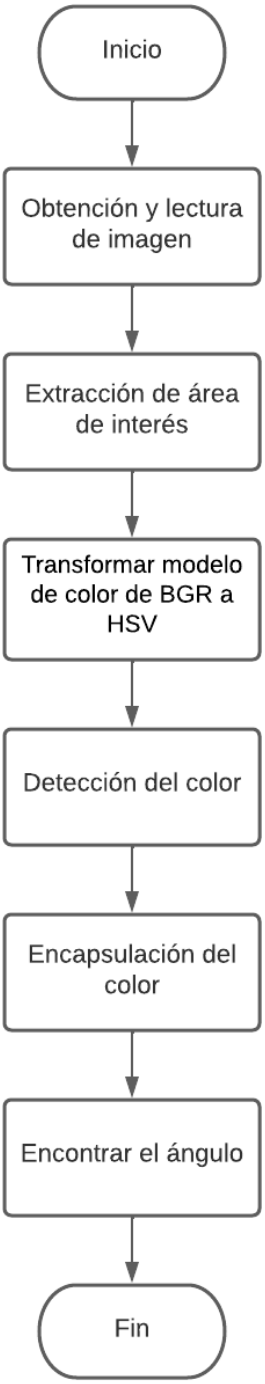


Gráfico 5-3: Diagrama de flujo de proceso de dimensionamiento del brazo

Realizado por: Pástor, S., (2021)

3.3.1.1 Obtención y lectura de la imagen

La imagen fue captada en el laboratorio de "Autopro" facultad de mecánica, mediante una cámara de alta definición a una distancia de 2 metros del brazo.

Figura 35-3: Brazo robótico ANNO RV624

Realizado por: Pástor, S., (2021)

Mediante la librería de OpenCV se procede a lectura de la imagen, para las pruebas realizadas se utilizó una imagen estática y posteriormente se trabajó en tiempo real mediante un video.

3.3.1.2 Extracción de áreas de interés

La imagen capturada es de un gran tamaño con respecto a su alto y ancho además la zona de interés es parte de toda la imagen para ello se procede a recortar la sección de la imagen que se necesita, la zona marcada de color azul es la imagen resultante caber mencionar que se definieron lugares fijos para la posición de la cámara por lo que el proceso se realiza una única vez.

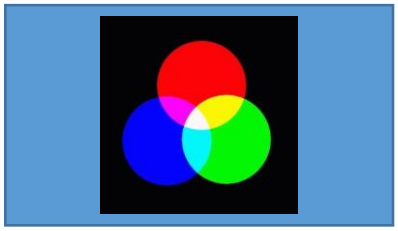
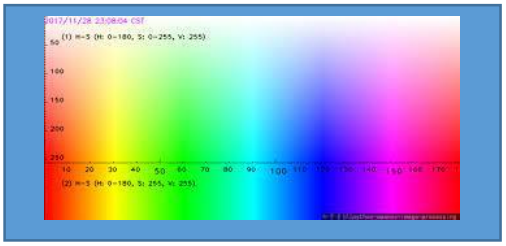
Figura 36-3: Ajuste de imagen para extracción del área de interés

Realizado por: Pástor, S., (2021)

3.3.1.3 Transformar modelo de color de BGR a HSV

Por defecto la librería realiza la lectura en el modelo de color BGR, con la finalidad de la detección de marcas de color, por lo cual se procedió a cambiar al modelo HSV, detallando la razón en la tabla 9-3:

Tabla 9-3: Transformación de modelo de color BGR a HSV

Modelo BGR	**Modelo HSV**
	
Un tono se encuentra categorizado por la proporción de color Rojo, Azul y Verde.	Un tono se encuentra categorizado por la proporción de matiz (rango de colores), Saturacion y Brillo.

Realizado por: Pástor, S., (2021)

La matiz con la que trabaja el modelo de color HSV es la que permite la detección de un intervalo de color por ello se cambia el modelo, su diferencia visual no se ve afectada pero la librería realiza la conversión correspondiente.

3.3.1.4 Detección de color

Las características del modelo permiten la extracción del color por medio de intervalos para obtener el área de interés, para ello se define un punto inicial y un punto final del intervalo considerando los rangos que manejo el modelo de color HSV junto con la librería OpenCV, que son los siguientes:

- Para la matiz valor mínimo desde el cero hasta el 180.
- Para la saturación y el brillo valores mínimos desde 0 hasta 255.

Obteniendo el siguiente resultado:

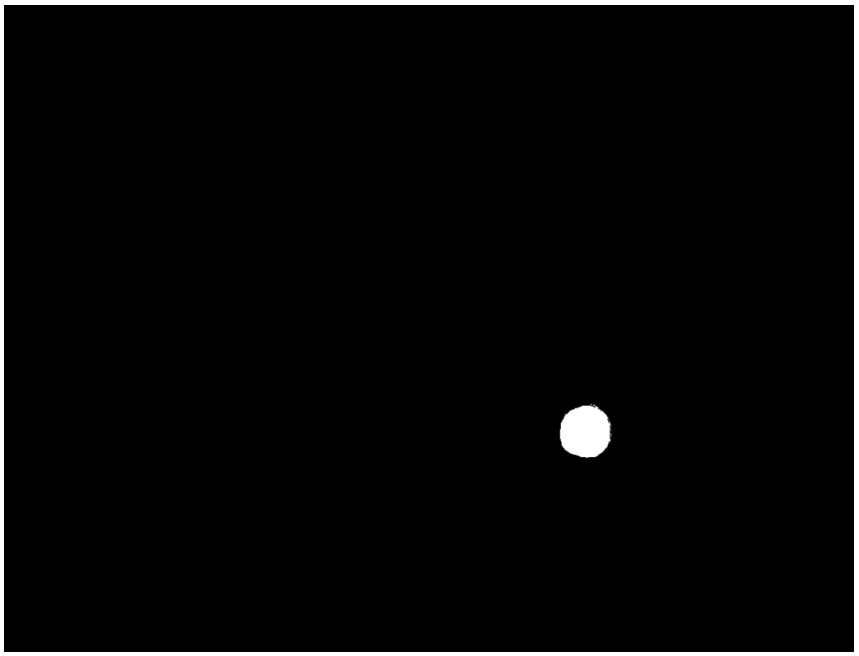


Figura 37-3: Área de interés del brazo robótico

Realizado por: Pástor, S., (2021)

3.3.1.5 Encapsulación del color

El color se encuentra limitado por una forma amorfa similar a un círculo, esta forma posee un área y un sinfín de posiciones, para la detección de un único punto que represente a todo el circulo se procede a encapsular todo el contorno en un rectángulo de color rojo, para seguir con el proceso se toma los vértices opuesto del cuadrilátero y se realiza una línea de color azul obteniendo así el baricentro de la figura simple.

La librería junto con el lenguaje de programación nos brinda las coordenadas del punto en sus dimensiones x e y que se visualiza el punto de color verde.

Figura 38-3: Encapsulamiento de la zona de interés

Realizado por: Pástor, S., (2021)

3.3.1.6 Encontrar el ángulo

El proceso se lo realiza para todas las marcas de color que se posicionaron en el brazo, de igual manera se obtiene el baricentro de la otra, obteniendo así dos puntos junto con sus coordenadas correspondientes que mediante funciones algebraicas se procede a encontrar el ángulo, sabiendo que cada valor de ángulo se lo va a visualizar en la pantalla.

Figura 39-3: Determinación del ángulo entre las dos marcas de color

Realizado por: Pástor, S., (2021)

3.4 Interfaz de visión artificial

Una vez demostrado las funciones de la visión artificial se recopila todas las funciones en una sola interfaz gráfica que permita el manejo práctico de la detección del área de interés del brazo robótico, como del reconocimiento de la zona facial del usuario como se representa en la figura 40-3.

Figura 40-3: Interfaz gráfica de visión artificial

Realizado por: Pástor, S., (2021)

3.5 Diseño del software de la nueva interfaz

Para la programación de la nueva interfaz se seleccionó Visual Studio, debido a su compatibilidad con varios lenguajes de programación como se determinó en el marco teórico.

En el software Visual estudio se realizó la programación para ejecutar el movimiento de cada juntura o articulación del brazo, recalcando que se decidió trabajar con este software debido a práctico manejo que tiene, además de la posible vinculación del lenguaje C con el lenguaje del software Python usado para el proceso de visión artificial, detallado en la figura 41-3, también otorga la facilidad de comunicación con el controlador STM-32 por medio del uso de las librerías DLL con instrucciones como: “openPort” para abrir la comunicación serial, “closePort” que terminar la comunicación serial, entre otras instrucciones las cuáles permiten el control de los movimientos del brazo robótico.

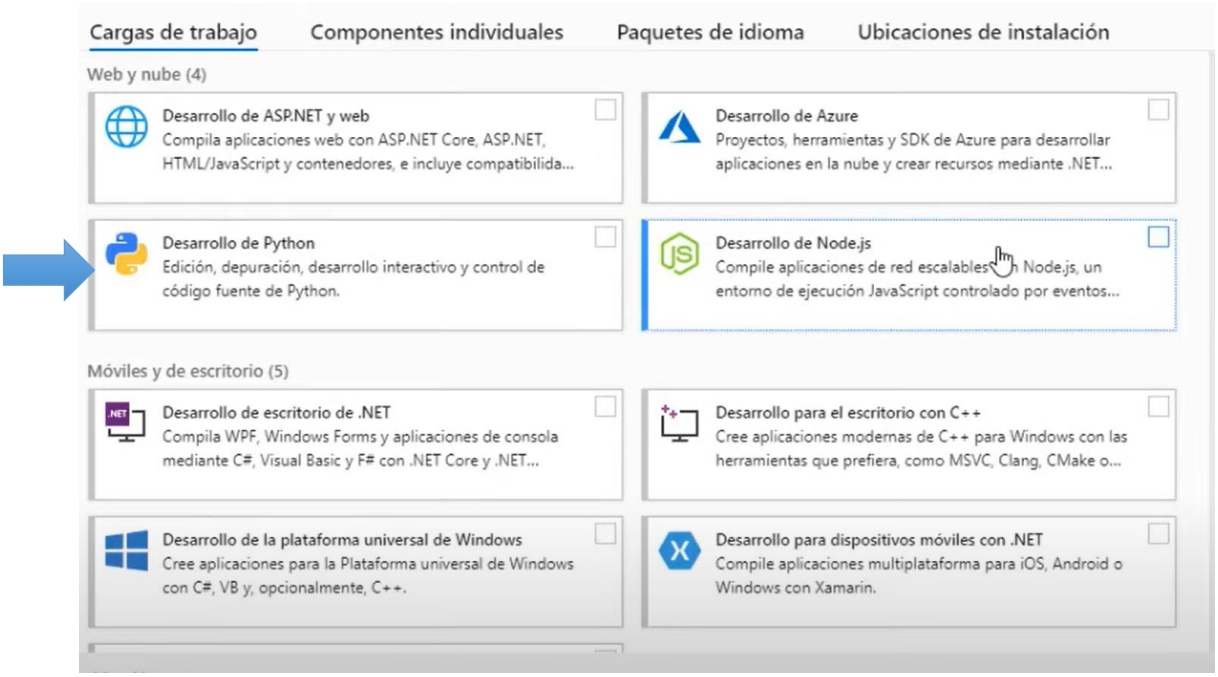


Figura 41-3: Opciones de compatibilidad de Visual Studio

Realizado por: Pástor, S., (2021)

3.5.1 Programación de la interfaz

Para una mejor comprensión de la programación se expone las partes más importantes de la codificación de la interfaz para el movimiento del brazo robótico.

Para establecer la comunicación controlador - brazo robótico, primero se declararon las librerías DLL, para posteriormente, una vez que estén escritas en las línea de programación se guarden y se copien en la ubicación en dónde se encuentra el proyecto, caso contrario no va a realizar el llamado de las instrucciones de las librerías y va a generar un error cuando se requiera compilar la codificación.

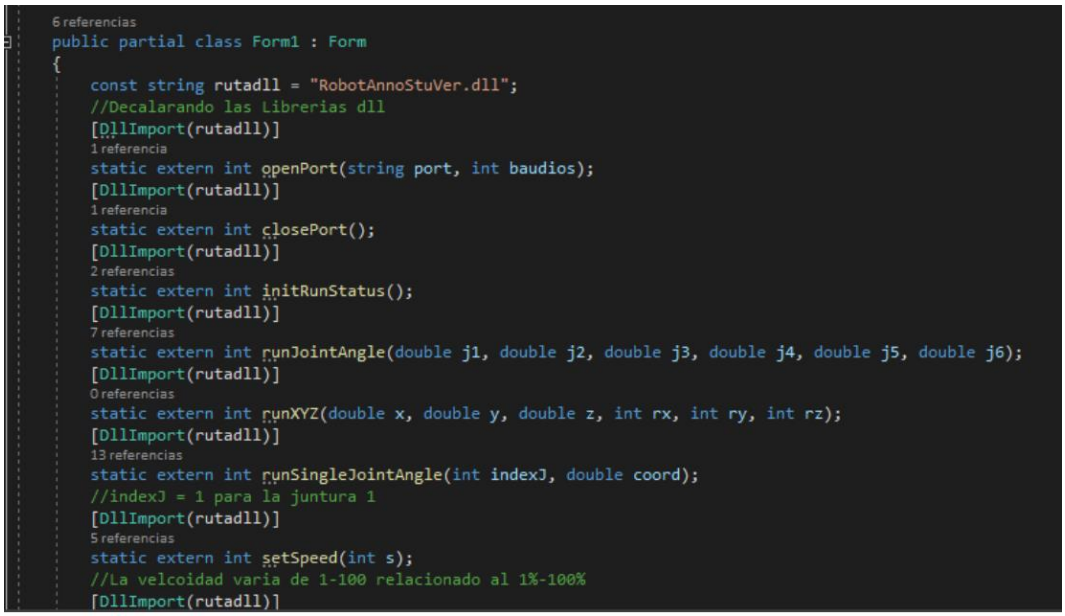


Figura 42-3: Declaración de librerías DLL en visual studio

Realizado por: Pástor, S., (2021)

Cuando ya se declararon las librerías DLL, se debe lograr que los servomotores se muevan de manera individual y para esto se programó por separado, para esto se muestra el código dónde la articulación aumente de grado, para realizar esto se utiliza la instrucción de la librería DLL denominada "runSingleJointAngle", esta instrucción va a permitir que solo se mueva un ángulo de las junturas o articulaciones.

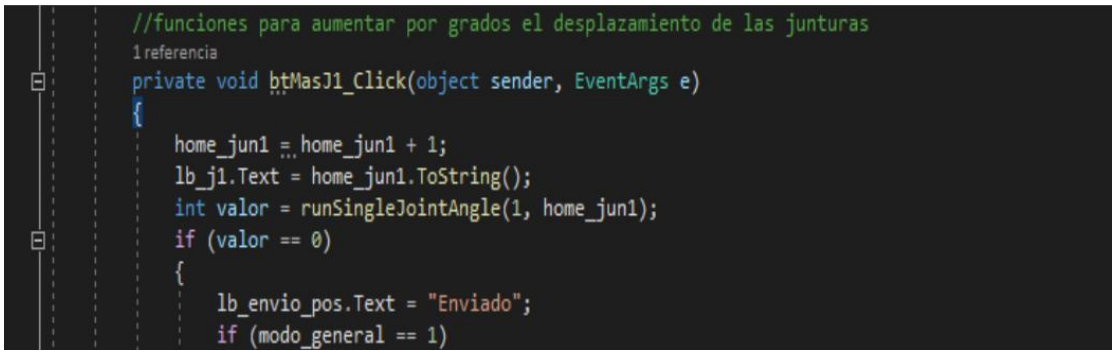


Figura 43-3: Codificación para el movimiento individual de junturas o articulaciones

Realizado por: Pástor, S., (2021)

3.5.2 Presentación y descripción de la interfaz

El la figura 51-3 se presenta el diseño de la interfaz para la calibración de la trayectoria del brazo robótico ANNO RV624, en dónde se detallaron todas las funciones que cumple cada elemento que constituyen parámetros como control de movimiento por juntas, comunicación serial, entre otros.

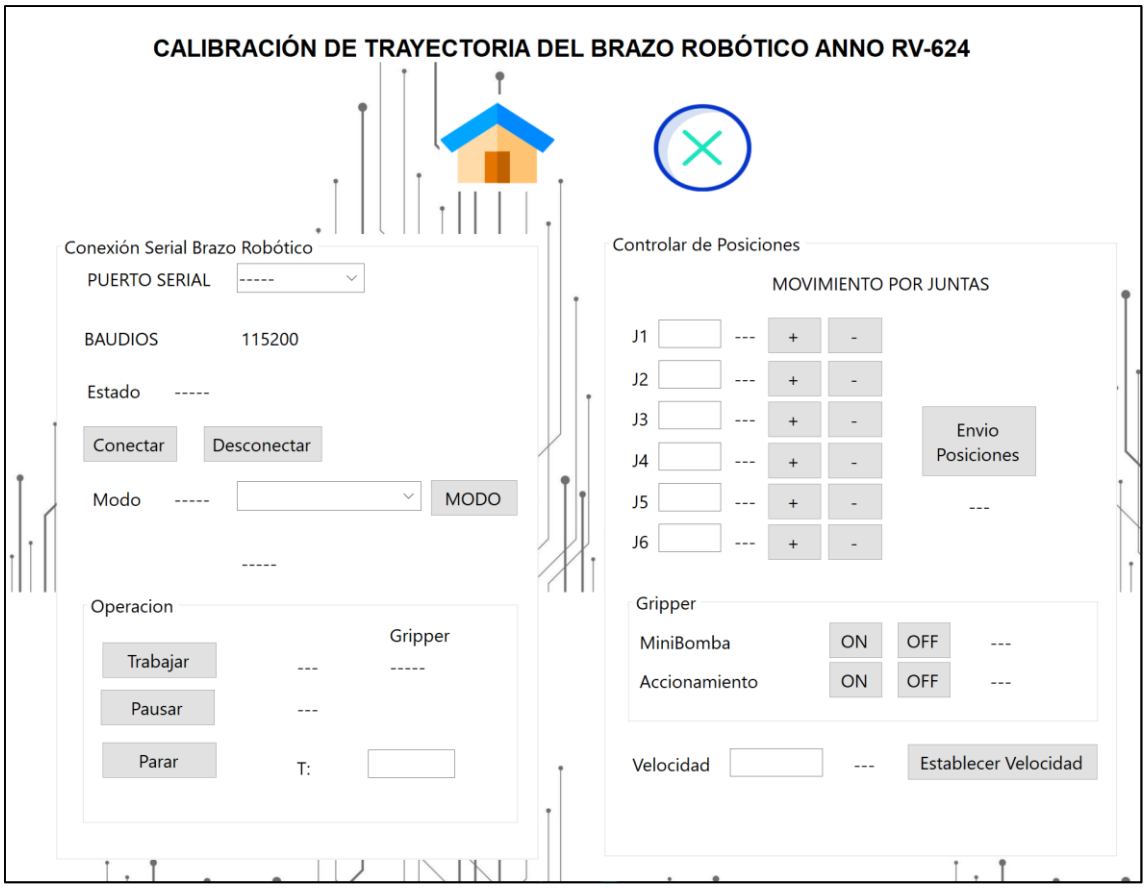


Figura 44-3: Interfaz de calibración de trayectoria del brazo robótico

Realizado por: Pástor, S., (2021)

3.5.2.1 Movimientos de articulaciones por juntas

Como se observa en la figura 45-3, esta parte de la interfaz permite controlar los movimientos del brazo de forma manual, dando clic sobre los botones de “+” y “-”, para moverlos grado por grado o se puede ingresar los ángulos a dónde se quiere mover.

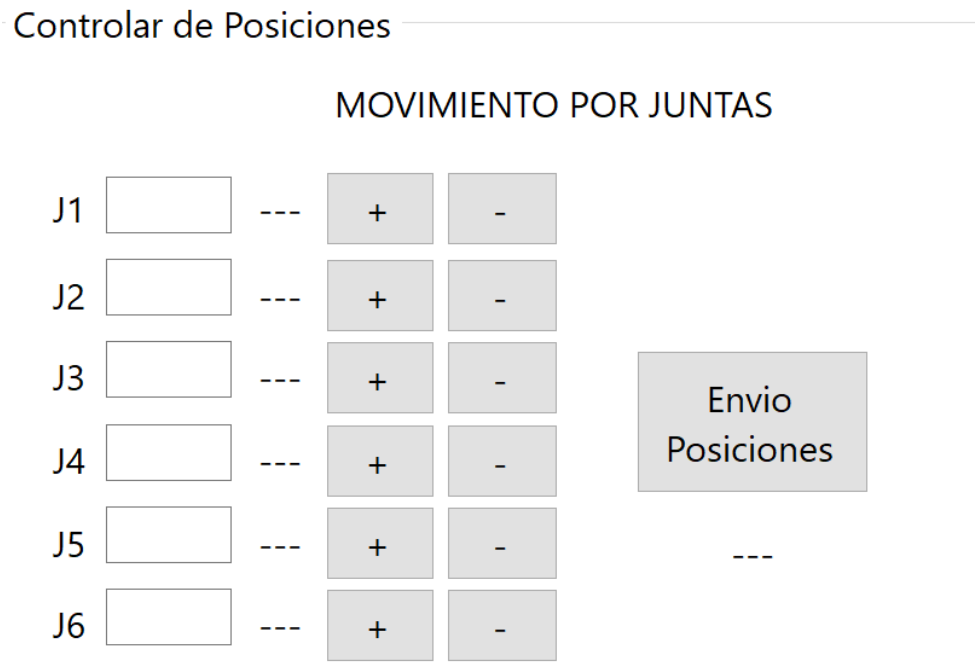


Figura 45-3: Control de movimiento de las articulaciones

Realizado por: Pástor, S., (2021)

3.5.2.2 Activación gripper

Cuándo se requiera encender el gripper para que realice su movimiento de apertura y cierre se debe primero accionar la mini bomba y posteriormente encender la electroválvula.

Figura 46-3: Control de mini bomba y gripper

Realizado por: Pástor, S., (2021)

3.5.2.3 Selección de velocidad

En este apartado nos permite seleccionar o establecer la velocidad a la que se requiere que las articulaciones del brazo robótico trabajen.

Figura 47-3: Selector de velocidad

Realizado por: Pástor, S., (2021)

3.5.2.4 Operación

En esta sección se encuentras los botones para que el brazo robótico realice movimientos antes prestablecidos en la sección de control de movimiento, además cuenta con las dos opciones para pausar o para finalizar la secuencia establecida.

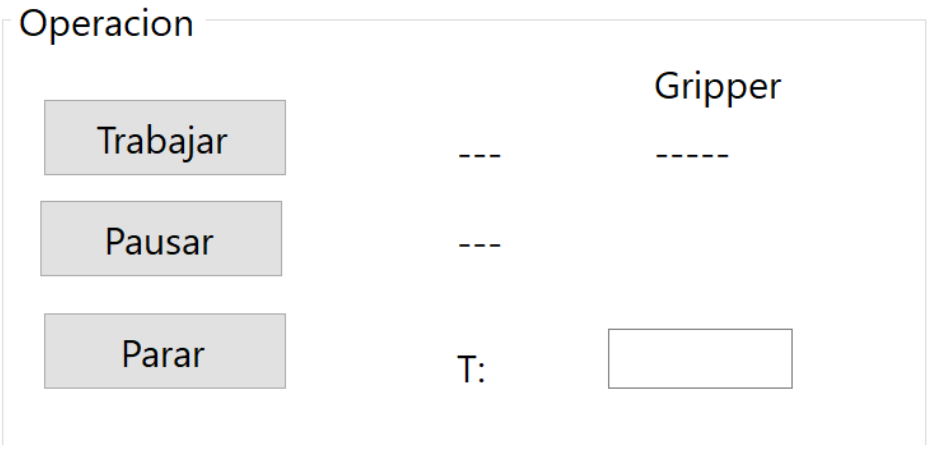


Figura 48-3: Sección de accionamiento del brazo robótico

Realizado por: Pástor, S., (2021)

3.5.2.5 Modo de trabajo

En el modo de trabajo se debe seleccionar control HMI, ya que este modo nos permite que el operador del brazo robótico lo pueda manipular desde la interfaz, pudiendo manejarlo por los movimientos descritos en el apartado de "movimiento de articulaciones por juntas", siguiendo los pasos del gráfico 6-3:

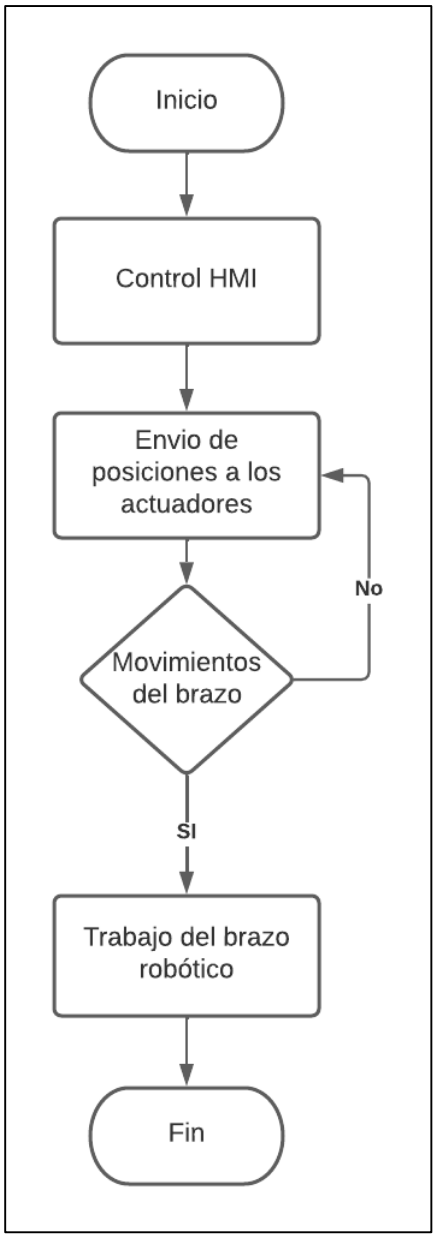


Gráfico 6-3: Diagrama de flujo del control HMI

Realizado por: Pástor, S., (2021)

3.5.2.6 Puerto de comunicación serial

En la sección de puerto serial el operario tiene la opción de desplegar el cuadro de diálogo para poder seleccionar al puerto al que esté conectado el controlador del brazo robótico:

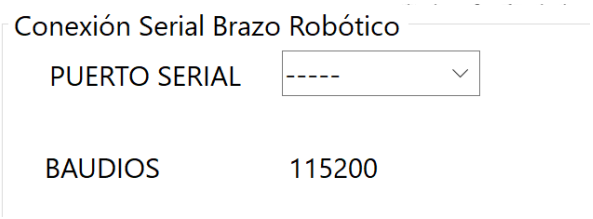


Figura 49-3: Conexión serial del brazo robótico

Realizado por: Pástor, S., (2021)

3.5.2.7 Estado

Como última sección se encuentra la de estado y esta nos permite acceder al uso de la interfaz después de haber hecho la comunicación serial mediante el botón "Conectar", si no se realiza este procedimiento el operario no podrá manipular la interfaz, un vez que haya concluido la conexión en el apartado "Estado" debe aparecer la palabra Conectado.

Figura 50-3: Estado de la conexión

Realizado por: Pástor, S., (2021)

3.6 Pruebas de funcionalidad - interfaz

3.6.1 Arranque de la interfaz de calibración de trayectoria del brazo robótico

La interfaz tiene un tiempo de apertura rápido y no tiene presenta ninguna complicación al momento de ejecutarla, cabe notar que solo se abrirá siempre y cuando se use el software Visual studio.

Figura 51-3: Interfaz ejecutada

Realizado por: Pástor, S., (2021)

3.6.2 Conexión inicial

Se puede notar en la figura 52-3 que la conexión serial del puerto USB funciona sin ningún problema, arrojando el estado de “conectado” y el modo de control HMI se puede evidenciar que está activo, entonces la interfaz esta lista para recibir instrucciones.

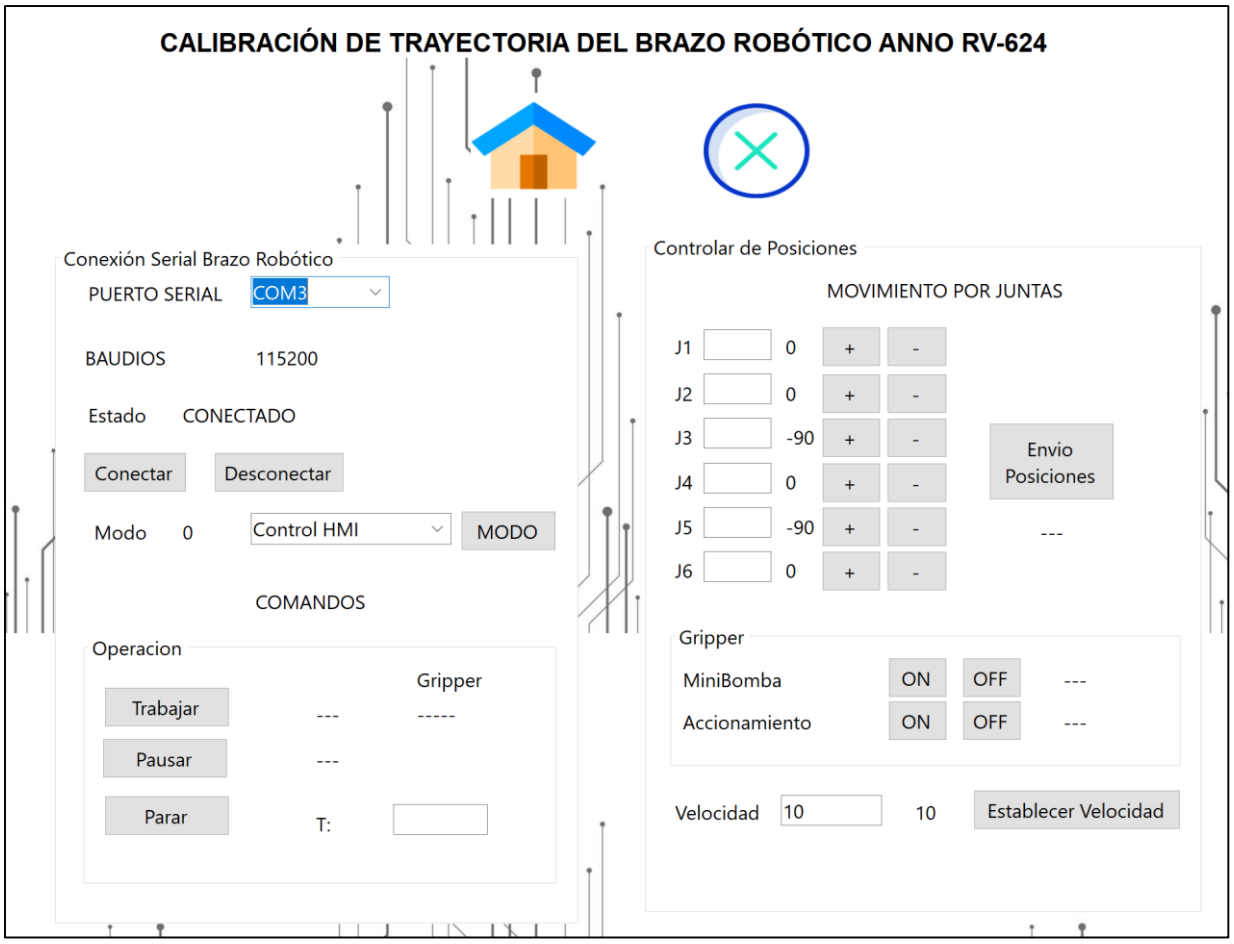


Figura 52-3: Puertos de conexión del brazo

Realizado por: Pástor, S., (2021)

Una vez que el puerto serial ya fue reconocido por la interfaz, esto nos quiere decir que la comunicación con el controlador y el computador está abierta, y se espera instrucciones para iniciar con el movimiento.

3.7 Pruebas de funcionalidad – secuencia de alimentación

En las siguientes figuras 53-3 hasta la 56-3, se demuestra las diferentes posiciones de la secuencia de alimentación de una persona con discapacidad motriz, utilizadas para la medición de tiempos mínimos y máximos en los cuales se puede cumplir la secuencia.

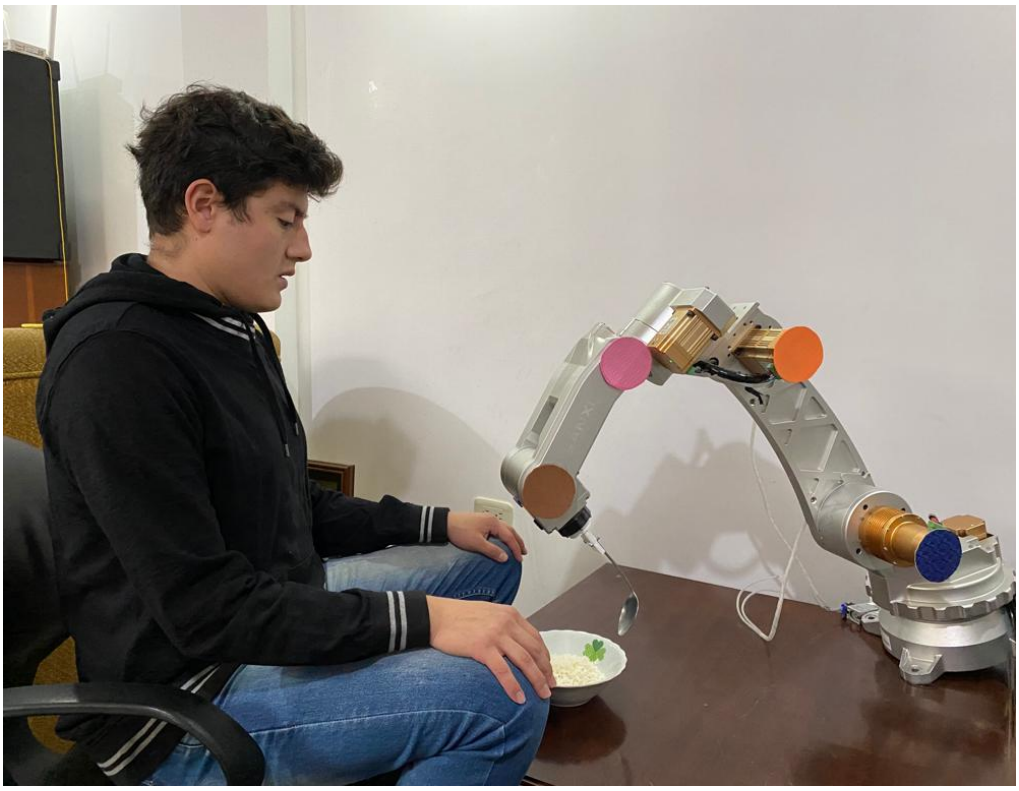


Figura 53-3: Acercamiento de la cuchara al plato

Realizado por: Pástor, S., (2021)

Figura 54-3: Recolección de comida

Realizado por: Pástor, S., (2021)

Figura 55-3: Acercamiento del plato hacia el mentón

Realizado por: Pástor, S., (2021)

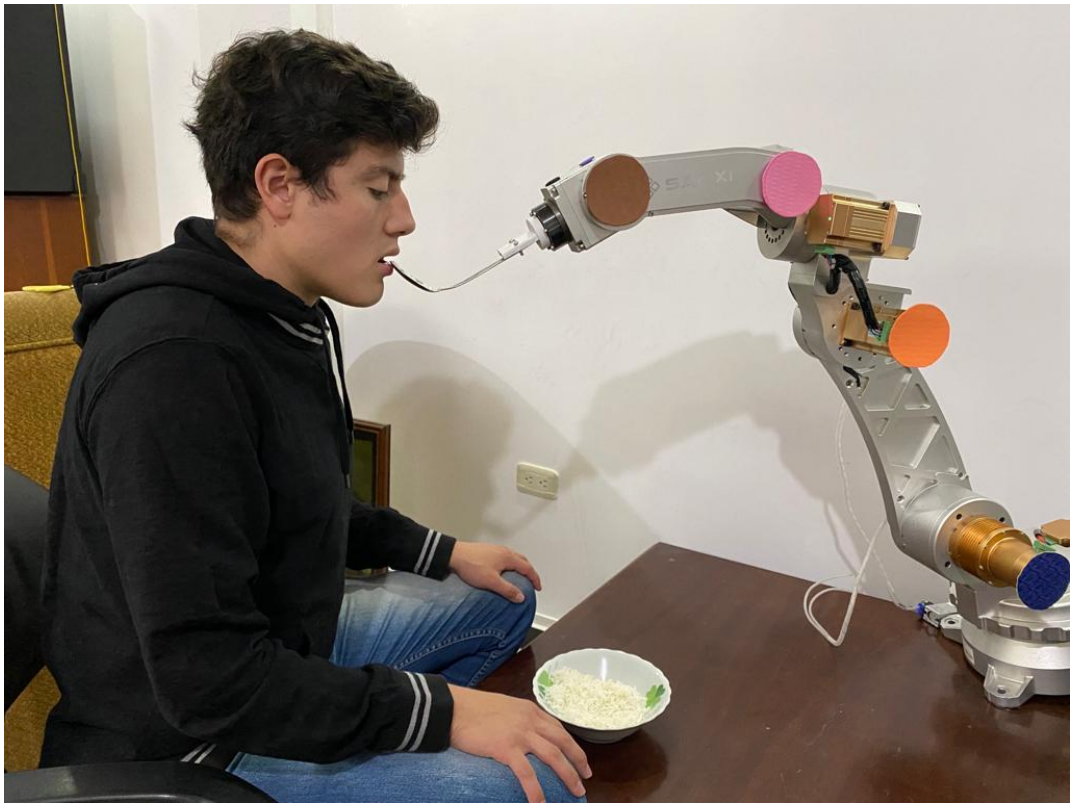


Figura 56-3: Acercamiento del mentón hacia la boca

Realizado por: Pástor, S., (2021)

3.7.1 Tablas de tiempos resultantes de la secuencia de alimentación

Después de haber realizado las pruebas de funcionalidad en la secuencia de alimentación y en el uso de la interfaz, se procedió a tomar los tiempos en segundos, según el rango de velocidad especificado (revisar anexos A, B y C) como se demuestra en las siguientes tablas.

Tabla 10-3: Tiempos del primer movimiento de la secuencia de alimentación

Movimientos	Velocidad	Tiempo promedio
1. Acercamiento de la cuchara al plato	5%	17,97
	10%	13,01
	15%	10,04
	20%	____

Realizado por: Pástor, S., (2021)

Tabla 11-3: Tiempos del segundo movimiento de la secuencia de alimentación

Movimientos	Velocidad	Tiempo promedio
2. Recolección de comida	5%	13,05
	10%	8,04
	15%	5,48
	20%	____

Realizado por: Pástor, S., (2021)

Tabla 12-3: Tiempos del tercer movimiento de la secuencia de alimentación

Movimientos	Velocidad	Tiempo promedio
3. Acercamiento del plato hacia el mentón	5%	15,53
	10%	11,45
	15%	7,53
	20%	____

Realizado por: Pástor, S., (2021)

Tabla 13-3: Tiempos del cuarto movimiento de la secuencia de alimentación

Movimientos	Velocidad	Tiempo promedio
4. Acercamiento del menton hacia la boca	5%	11,03
	10%	6,38
	15%	4,06
	20%	____

Realizado por: Pástor, S., (2021)

Como se puede observar en las tablas 10-3 hasta la 13-3, la velocidad al 20% no fue medida, ya que el exceso de velocidad para una trayectoria corta generaba vibraciones en la mesa y desaceleraciones abruptas, que ocasionaron la pérdida del alimento que recogía la cuchara, así también esto puede generar daños físicos en la integridad de los servomotores del brazo robótico.

Tabla 14-3: Tiempos totales para la ejecución de las secuencias

Velocidad	Tiempos Totales de ejecución de secuencias
5%	57,59
10%	38,89
15%	27,11
20%	____

Realizado por: Pástor, S., (2021)

La tabla 14-3 nos indica el tiempo total de ejecución para la secuencia de alimentación según la selección de velocidad, dando como velocidades óptimas del 5 al 10%, ya que no presenta vibraciones en la mesa, ni paros agresivos que dañen al equipo o arrojen comida.

3.8 Pruebas de funcionalidad – Visión artifical

Para realizar las pruebas de visión artifical se han dividido en dos grandes grupos como son el reconocimiento facial y el reconocimiento de distancias mediante ejes centrales del brazo robótico ANNO RV 624.

3.8.1 Reconocimiento facial

Esta prueba se realizó para determiner que en el proceso de visión artificial la detección facial funcione de una manera acertada, midiendo el tiempo que se demora en procesar y reconocer los siguientes condicionales: el rostro del usuario, el condicional SI y NO, boca abierta y boca cerrada y rostro fuera de rango.

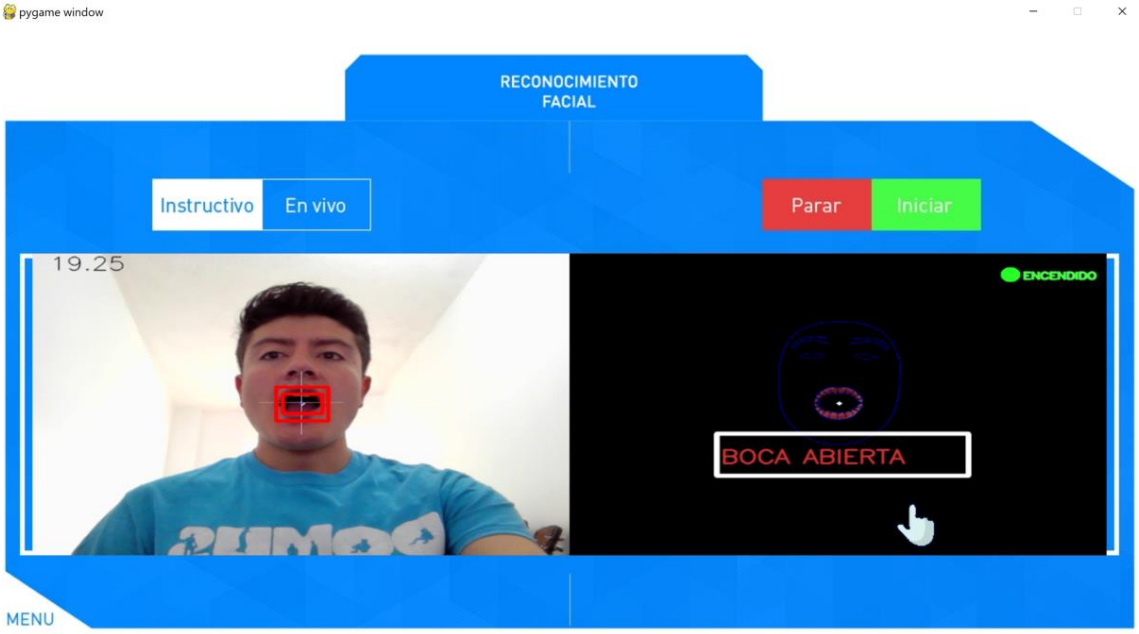


Figura 57-3: Reconocimiento zona de interés boca

Realizado por: Pástor, S., (2021)

En la tabla 15-3 se muestra un resumen de los resultados de cada uno de los condicionales mencionados para evaluar el tiempo de respuesta del reconocimiento facial, para ver la tabla completa revisar Anexo D.

Tabla 15-3: Tabla resumen tiempos de visión artificial

Reconocimiento facial		Condicional SI	Boca cerrada	Boca Abierta	Rostro fuera de rango	Condicional NO
N° Datos	Tiempo(S)	Tiempo(S)	Tiempo(S)	Tiempo(S)	Tiempo(S)	Tiempo(S)
1	0,00338	0,35616	0,44167	0,03134	0,31101	0,13184
2	0,13709	0,86687	0,13564	0,87362	0,50848	0,31927
3	0,86429	0,26833	0,40139	0,84924	0,04111	0,26400
4	0,87074	0,30023	0,73198	0,88728	0,27918	0,16666
5	0,95214	0,84918	0,08425	0,77409	0,09547	0,63069

26	0,51170	0,66495	0,19953	0,58432	0,82491	0,81424
27	0,30847	0,36874	0,22242	0,27850	0,15624	0,18972
28	0,01913	0,29894	0,30115	0,11000	0,83106	0,45415
29	0,89179	0,12954	0,98359	0,61602	0,37296	0,24453
30	0,15054	0,48916	0,83459	0,52214	0,95509	0,30046
Promedio =	0,35370	0,56835	0,48799	0,52011	0,42197	0,44476

Realizado por: Pástor, S., (2021)

Del anális de la tabla 15-3, se determinó el promedio de los tiempos de una muestra de 30 datos para determinar el tiempo de respuesta de reconcimiento facial del usuario, dando como resutado una media 0,46615 cienmilésimos de segundo, indicándonos que el reconocimiento facial es eficiente y eficaz.

3.8.2 Reconocimiento de distancias mediante ejes centrales del brazo robótico

Esta prueba se realizó para medir el tiempo que demora en reconocer la visión artificial los puntos medios de las circunferencias de colores para medir la distancia entre ejes centrales.

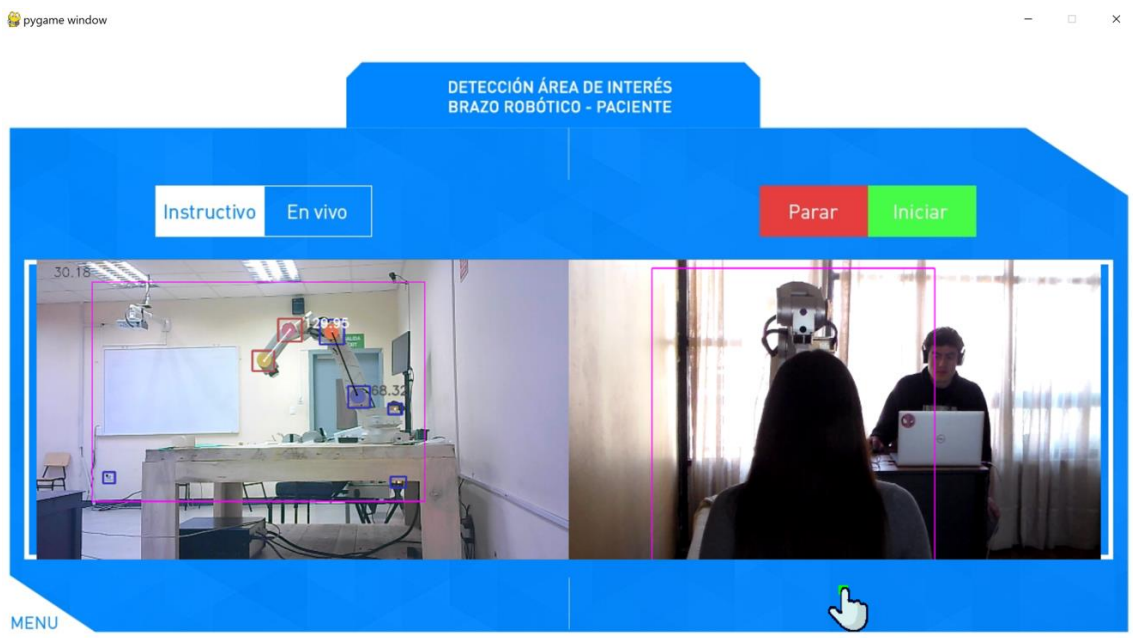


Figura 58-3: Reconocimiento de distancias

Realizado por: Pástor, S., (2021)

Del análisis de la tabla 16-3, se muestra un resumen de los resultados de cada uno de los tiempos que tardó la visión artifical en reconocer los ejes centrales para medir las distancias, para ver la tabla completa revisar Anexo E.

Tabla 16-3: Tabla de resumen de tiempos

Ejes centrales	
N° Datos	Tiempo
1	0,92779
2	0,98168
3	0,49964
4	0,45032
5	0,38312
6	0,05508
7	0,60605
8	0,78405
9	0,28093
10	0,33645
26	0,70272
27	0,75753
28	0,73645
29	0,62569
30	0,78765
Promedio =	0,50068

Realizado por: Pástor, S., (2021)

Del anális de la tabla 16-3, se determinó el promedio de los tiempos de una muestra de 30 datos para determinar cuanto tarde en reconocer los ejes centrales de las circunferencias de colores, dando como resutado una media 0,50068 cienmilésimos de segundo, indicándonos que el reconocimiento trabaja de una manera óptima.

4 GESTIÓN DEL PROYECTO

4.1 Cronograma

El itinerario de actividades se dividió de la siguiente manera como lo indica el siguiente cronograma:

Tabla 1-4: Cronograma de actividades

ACTIVIDADES	MESES															
	MES 1				MES 2				MES 3				MES 4			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Capacitación sobre el manejo de la interfaz predeterminada del brazo robótico ANNO RV624																
Análisis bibliográfico y recolección de información referente al tema de estudio.																
Codificaión del algoritmo de visión artificial																
Diseño de la nueva interfaz para la calibración de la trayectoria de los movimientos del brazo.																
Implementación del sistema inclusivo para alimentar a personas con discapacidad motriz.																
Obtención de datos para el aná lisis de resultados																
Etapa de redacción del informe final																

Realizado por: Pástor, S., (2021)

4.1.1 Costos

A continuación, se indican los costos totales necesarios para la implementacióndel sistema inclusivo para alimentar a personas con discapacidad motriz usando el brazo robótico ANNO RV624

Tabla 2-4: Costos totales para la implementación de un sistema inclusivo

N°	Detalle	Cantidad	Valor unitario($)	Valor total ($)
1	Impresiones en 3D del soporte de Gripper	2	7,00	14,00
2	Cámara web 4k	1	48,00	48,00
3	Juego de llaves hexagonales	1	4,75	4,75
4	Trípode para cámara	2	16,50	33,00
5	Trasnformador de 110 a 220 V, 360W	1	27,00	27,00
			Total (USD)	126,75

Realizado por: Pástor, S., (2021)

4.1.2 Recurso humano

El principal recurso humano es Erick Samir Pástor Jácome, autor del presente trabajo de pregrado, que junto a la metodología aplicada y el compromiso con el trabajo permitieron cumplir los objetivos planteados y próximamente la titulación.

CONCLUSIONES

Se consiguió la implementación de un sistema inclusivo para alimentar a personas con discapacidad motriz utilizando en brazo robótico ANNO RV624 que incorpora recursos tecnológicos para la detección facial, procesamiento de datos y ejecución del movimiento del brazo, en la denominada secuencia de alimentación, así también como el desarrollo de una nueva interfaz.

Se utilizó como software principal Python, haciendo uso sus librerías Open CV y MediaPipe Face Mesh, recalcando que esta última librería trabajó con un mallado de 468 puntos clave que permitieron el desarrollo del reconocimiento facial de los pacientes, específicamente las zonas de interés denominadas, zona A “contorno labios” y zona B “interior boca”, que permitieron la detección de alimento durante y después de la secuencia de alimentación, así también como la creación de los movimientos condicionales de la cabeza que inician o paran el proceso.

Integrando el software Visual Studio al proyecto se logró desarrollar una interfaz para la calibración de la trayectoria del brazo robótico ANNO RV624, que se basa en el movimiento por junturas o articulaciones por las cuáles está conformando el mismo, mediante el uso de las librerías DLL que permiten la manipulación del movimiento estableciendo una comunicación entre el controlador STM-32 y el computador, mediante la comunicación serial por un puerto USB.

Mediante las pruebas de funcionalidad de la interfaz, se determinó que en el momento de ejecución de la aplicación, esta no presenta ningún tipo de dificultad, además de esto la comunicación serial se logra en por alto porcentaje del 100%, permitiendo la manipulación total de todas sus funciones.

En cuanto a la medición de tiempos según el rango de velocidad se demostró que la velocidad al 20% no es óptima para cumplir con la función de alimentar a una persona con discapacidad motriz, ya que ocasionó la pérdida del alimento que recogía la cuchara, además que el exceso de velocidad, para una trayectoria corta generaba vibraciones en la mesa y desaceleraciones abruptas, que pueden generar daños físicos en la integridad de los servomotores del brazo robótico, dándonos como resultado que la velocidad adecuada para esta función tiene un rango del 5 al 10% de velocidad, ya que no presentaba ninguna dificultad en la recolección de comida, ni en el movimiento del brazo.

RECOMENDACIONES

Para correcto funcionamiento del controlador SMT-32, es obligatorio instalar el driver actualizado denominado PL203 para que de esta manera se eviten problemas de compatibilidad con el sistema operativo, ya que los drivers de fábrica solo abarcan versiones de Windows 7 o inferiores.

Antes de utilizar el brazo robótico ANNO RV624 se debe setear la posición HOME o posición cero, ya que este toma como posición inicial el último movimiento que se realizó antes de apagar el equipo.

Para la obtención de imágenes de visión artificial, se recomienda el uso de una cámara de alta calidad, debido a que se trabaja con píxeles, y mientras mayor calidad de imagen, mayor número de píxeles, y la detección de zonas de interés y puntos clave en el mallado, se encuentra de manera más precisa.

GLOSARIO

Controlador: Henríquez & Martínez (2019), menciona que, "los controladores ofrecen al usuario la capacidad de programar una determinada operación de modo que se realice de forma regular, un sistema que haya sido correctamente preparado hará que el sistema sea independiente de las perturbaciones externas". (Henríquez Novoa, y otros, 2019)

DLL: Es una biblioteca de códigos que pueden ser llamados o cargados por otros programas, y estos puedes ser utilizados por software distintos a la vez.

Python: En base a lo escrito por Gualiano (2019), "Python es un lenguaje de programación multiplataforma, consistente y maduro, utilizado por numerosas empresas internacionales". (Guagliano, 2019)

Visión Artificial: La visión artificial, también conocida como visión por computadora o visión por ordenador, es una "disciplina que engloba todos los procesos y elementos que proporcionan ojos a una máquina, y que, mediante un proceso computacional, permite analizar e interpretar dichas imágenes". (González Marcos, y otros, 2006)

BIBLIOGRAFÍA

ALVARADO CARRILLO, N., & GUALTEROS MATIZ, Y. E. *Diseño de un brazo robótico para utilizar en un laboratorio de automatización.* (2019). Tesis ingenieril, Universidad de América, Ingeniería mecánica, Bogotá. Recuperado el 28 de Julio de 2021, de https://repository.uamerica.edu.co/bitstream/20.500.11839/7440/1/4131684_2019-2-IM.pdf

ASOCIACIÓN ARGENTINA DE CONTROL AUTOMÁTICO. Servomotores: control, precisión y velocidad. *AADECa*(4), 22. (2017). Recuperado el 5 de Julio de 2021, de https://editores-srl.com.ar/revistas/ie/318/automacion_servomotores

ASOCIACIÓN ESPAÑOLA DE INGENIERÍA MECÁNICA. *XXI Congreso Nacional de Ingeniería Mecánica.* Alicante, España: Universidad Miguel Herández de Elche. (2016). Recuperado el 4 de Agosto de 2021, de https://innovacionumh.es/editorial/XXI%20Congreso%20Nacional%20Ingenieria%20Mecanica.pdf

CANTOS SERRANO, J., & PÉREZ LLORENS, J. *Instalaciones eléctricas básicas* (2018). (Primera ed.). Madrid, España: Ediciones Paraninfo, S.A. Recuperado el Junio de 2021, de https://books.google.com.ec/books?id=qE9tDwAAQBAJ&printsec=frontcover#v=onepage&q&f=true

CARDOSO, E., & FERNÁNDEZ, A. *Modelos cinemático y dinámico de un robot de cuatro grados de libertad. SciELO Analytics, 38(3).* (2017). Recuperado el 20 de Julio de 2021, de http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1815-59282017000300006

CARVAJAL VALENCIA, B., & HUARACA BUÑAY, B. *Empleo de la robótica en ayuda a las personas con discapacidad.* (2017). Tesis Ingenieril, Universidad Estatal de Milagro, Facultad Ciencia de la Ingeniería, Milagro. Recuperado el 3 de Junio de 2021, de http://repositorio.unemi.edu.ec/bitstream/123456789/3594/1/CARVAJAL%20VALENCIA%20y%20HUARACA%20BU%c3%91AY%20EMPLEO%20DE%20LA%20ROB%c3%93TICA%20EN%20AYUDA%20A%20LAS%20PERSONAS%20CON%20DISCAPACIDAD.pdf

CASTILLO GARCÉS, D. S. *Diseño e implementación de luces de Xenón e iluminación Led en faros de vehículos.* (2014). Tesis Ingenieril, Universidad San Francisco de Quito, Ciencias e Ingeniería, Quito. Recuperado el Julio de 2021, de https://repositorio.usfq.edu.ec/bitstream/23000/4603/1/113636.pdf

CASTRO GUAMÁN , M. P., & POSLIGUA MURILLO, N. C. *Diseño de iluminación con luminarias tipo led basado en el concepto eficiencia energética y confot visual, implementación de estructura para pruebas.* (2015). Tesis Ingenieril, Universidad

Politécnica Salesiana Sede Guayaquil, Facultad de Ingenierías, Guayaquil. Recuperado el 2 de Agosto de 2021, de https://dspace.ups.edu.ec/bitstream/123456789/10253/1/UPS-GT001344.pdf

CEA HIDALGO, E. M. *Estudio y aplicación de la librería OpenCv sobre la arquitectura ARM, para el control de agentes robóticos móviles usando visión artificial.* (2018). Tesis ingenieril, Universidad del BÍO - BÍO , Ingeniería civil en informática, Chillán. Recuperado el 27 de Julio de 2021, de http://repobib.ubiobio.cl/jspui/bitstream/123456789/2586/1/Cea%20Hidalgo%2C%20Esteban%20Mitchel.pdf

CERDÁ FILIU, L. M. *Automatismos neumáticos e hidráulicos.* (2018). (Primera ed.). Madrid , España: Ediciones Paraninfo. Recuperado el Junio de 2021, de https://books.google.com.ec/books?id=4_p6DwAAQBAJ&printsec=frontcover#v=onepage&q&f=true

CONSEJO NACIONAL PARA LA IGUALDAD DE DISCAPACIDADES. *Consejo Nacional para la Igualdad de Discapacidades*. (2021). Recuperado el 01 de Octubre de 2021, de Estadísticas de Discapacidad: https://www.consejodiscapacidades.gob.ec/estadisticas-de-discapacidad/

CORONA RAMÍREZ, L., ABARCA JIMÉNEZ, G., & MARES CARREÑO, J. *Sensores y actuadores.* (2014). Azcapotzalco, México: Grupo Editorial Patria, S.A. de C.V. Recuperado el 4 de Agosto de 2021, de https://books.google.com.ec/books?id=wMm3BgAAQBAJ&printsec=frontcover#v=onepage&q&f=true

CREUS SOLÉ, A. *Neumática e hidráulica* (Segunda ed.). Barcelona, España: ediciones técnicas marcombo. (2011). Recuperado el Julio de 2021, de https://books.google.com.ec/books?id=CXtRrbhr9bYC&printsec=frontcover#v=onepage&q&f=true

CUSHPA, P. *Comunicación y virtualización de procesos industriales basados en industria 4.0.* (2020). Tesis Ingenieril, Universidad Técnica de Ambato , Ingeniería en electrónica y comunicaciones , Ambato . Recuperado el 28 de Agosto de 2021, de https://repositorio.uta.edu.ec/jspui/handle/123456789/30704

DE LA CRUZ, Y. G., & TINOCO YAMUNAQUÉ, J. *Determinación de la calidad de granos de arroz pulito utilizando algoritmos de procesamiento digital de imágenes.* (2019). Tesis Ingenieril, Universidad Nacional Pedro Ruiz Gallo, Facultad de ciencias físicas y matemáticas, Lambayeque. Recuperado el Julio de 2021, de https://repositorio.unprg.edu.pe/bitstream/handle/20.500.12893/7990/BC4380%20DE%20LA%20CRUZ%20MORALES-TINOCO%20YAMUNAQUE.pdf?sequence=1&isAllowed=y

DÍAZ ÁLVAREZ, J. *Visión por computador para el uso de realidad aumentada en Unity3D.* (2021). Tesisi ingenieril, Universidad Politécnica de Madrid, Ingeniería informática, Madrid. Recuperado el 25 de Julio de 2021, de https://oa.upm.es/68007/1/TFG_JORGE_DIAZ_ALVAREZ.pdf

DIRECCIÓN DE EDUCACIÓN TÉCNICA. *Morfología básica de un robot industrial.* (2020). Cuadernillo de actividades , Buenos Aires. Recuperado el 15 de Agosto de 2021, de https://cdn.continuemosestudiando.abc.gob.ar/uploads/e5644eb9-713f-45a0-9be2-552c4945bf33.pdf

FERNÁNDEZ, J. L. *FisicaLab*. (2020). Recuperado el 30 de Julio de 2021, de Óptica Geométrica: https://www.fisicalab.com/apartado/camara-fotos

GAGO , A., & FRAILE, J. *Iluminación con tecnología LED.* (2012). España: Ediciones Parainfo, S.A. Recuperado el Julio de 2021, de https://books.google.com.ec/books?id=8FN1mCQVzrIC&printsec=frontcover&dq=iluminaci%C3%B3n+led+pdf&hl=es-419&sa=X&redir_esc=y#v=onepage&q&f=true

GARCÍA, I., & CARANQUI, V. *La visión artificial y los campos de aplicación.* (2015). *Tierra Infinita, I*(1), 94-103. doi:https://doi.org/10.32645/26028131.76

GONZÁLEZ MARCOS, A., MARTÍNEZ DE PISÓN, F., PERNÍA ELÍAS, A., & CATEJÓN LIMAS, M. *Técnicas y algoritmos básicos de visión artificial.* (2006).La Rioja, España: Universidad de La Rioja. Recuperado el 25 de Julio de 2021, de https://publicaciones.unirioja.es/catalogo/online/VisionArtificial.pdf

GUAGLIANO, C. *Programación en Python* (Primera ed., Vol. II). Buenos Aires, Argentina: Six Ediciones. (2019). Recuperado el 20 de Agosto de 2021, de https://books.google.com.ec/books?id=y1yzDwAAQBAJ&printsec=frontcover&dq=Que+es+Python+y+para+que+sirve&hl=es419&sa=X&ved=2ahUKEwip2O_Ouab2AhVlRTABHffADyYQ6AF6BAgCEAI#v=onepage&q&f=true

GUARACA MEDINA, P. J., & OCHOA OCHOA, J. L. *Estudio de la programación y operación de los robots industriales KUKA KR 16-2 y KR 5-2 ARC HW.* (2015). Tesis ingenieril, Universidad Politécnica Saleciana Sede Cuenca, Ingeniería electrónica, Cuenca. Recuperado el 15 de Agosto de 2021, de https://dspace.ups.edu.ec/bitstream/123456789/7744/1/UPS-CT004593.pdf

HENRÍQUEZ NOVOA, J. A., & MARTÍNEZ RODRÍGUEZ, W. J. *Identificación y sintonización de controladores PID para procesos de integración*. (2019). Tesisi ingenieril, Universidad de la costa - CUC , Ciencias de la computación y electrónica, Barranquilla. Recuperado el 9 de Julio de 2021, de https://repositorio.cuc.edu.co/bitstream/handle/11323/5557/Identificaci%C3%B3n%20y%20sintonizaci%C3%B3n%20de%20controladores%20PID%20para%20procesos%20de%20integraci%C3%B3n.pdf?sequence=1&isAllowed=y

HERNÁNDEZ ORDOÑEZ, M., ORTÍZ MOCTEZUNA, M. B., CALLES ARRIAGA, C. A., & RODRÍGUEZ PORTILLO, J. C. *Robótica. Análisis, modelado, control e implementación.* (2015). Ciudad de México, México: Omina Publisher SL. doi:https://dx.doig.org/10.3926/oss.18

INGENIERÍA MECAFENIX. *Ingeniería Mecafenix*. (2021). Recuperado el 14 de Julio de 2021, de ¿Qué es un fina de carrera y para que sirve?: https://www.ingmecafenix.com/electronica/final-de-carrera/

JENER, P. C. *Integración de un sistema robótico asistencial controlado mediate una interfaz cerebro computador para personas con discapacidad motora.* (2019). Tesis Ingenieril, Pontífica Universidad Católica del Perú, Facultad de ciencias e ingeniería, Lima. Recuperado el 29 de 09 de 2021, de http://hdl.handle.net/20.500.12404/15547

KAPLAN, J., CRONIN, B., & MAUS, T. *Anestesia en cirugía cardiaca* (Segunda ed.). (2019). Barcelona, España: Elsevier España, S.L.U. Recuperado el 15 de Agosto de 2021, de https://books.google.com.ec/books?id=8nSDwAAQBAJ&pg=PA418&dq=paraplejia&hl=es419&sa=X&ved=2ahUKEwjn_vCOtqb2AhVpSjABHY5LAfwQ6AF6BAgJEAI#=onepage&q&f=true

LÓPEZ MARTÍNEZ, A., GÓMEZ GALÁN, M., SÁNCHEZ SALINAS, S., & MARTÍNEZ LAO, J. *Tecnología de la Fabricación.* (2019). Almería, España: Editorial Universidad de Almería. Recuperado el 2 de Junio de 2021, de https://books.google.com.ec/books?id=U2GUDwAAQBAJ&pg=PA42&dq=definici%C3%B3n+de+robot+industrial&hl=es419&sa=X&ved=2ahUKEwjmnNWAw9fzAhUdVTABHVI-D04Q6AF6BAgJEAI#v=onepage&q&f=true

MANDADO PÉREZ, E., ACEVEDO, J., FERNÁNDEZ, C., & ARMESTO, J. *Autómatas programables y sistemas de automatización* (Segunda ed.). (2009). Barcelona, España: ediciones técnicas marcombo. Recuperado el 1 de Septiembre de 2021, de https://books.google.com.ec/books?id=5jp3bforBB8C&pg=PR24&dq=clasificaci%C3%B3n+de+los+sensores&hl=es419&sa=X&ved=2ahUKEwjukOyHsOHzAhWjRjABHSleCyUQ6AF6BAgHEAI#v=onepage&q&f=true

MEDIAPIPE, G. *Mediapipe.* (2020). Recuperado el 20 de 11 de 2021, de https://google.github.io/mediapipe/

MENDOZA AUSTRIA, L., CASTELÁN , S., ISLAS , A., & ZUVIRIE, E. Internet de las cosas para generar una cocina segura. *Revist de sistemas computacionales y TIC´s, III*(1021), 23. (2017). Recuperado el 6 de Junio de 2021, de https://www.ecorfan.org/spain/researchjournals/Sistemas_Computacionales_y_TICs/vol3num10/Revista_de_Sistemas_Computacionales_y_TICS_V3_N10_3.pdf

MICROSOFT. *Microsoft*. (2021). Recuperado el 7 de Enero de 2022, de ¿Qué es un controlador?: https://docs.microsoft.com/es-es/windows-hardware/drivers/gettingstarted/what-is-a-driver-

MICROSOFT. *Microsoft*. (2022). Recuperado el 26 de Febrero de 2022, de Le damos la bienvenida al IDE de Visual Studio: https://docs.microsoft.com/es-es/visualstudio/get-started/visual-studio-ide?view=vs-2022

MUÑOZ VEGA, E. P. *Desarrollo de un sistema de control de acceso de personal empleando reconocimiento facial respaldado con técnicas de aprendizaje profundo.* (2021). Tesis ingenieril, Universidad de las Fuerzas Armadas, Eléctrica y elctrónica y telecomunicaciones, Sangolquí. Recuperado el 26 de Agosto de 2021, de http://repositorio.espe.edu.ec/bitstream/21000/25302/1/T-ESPE-044623.pdf

NACIONES UNIDAS CEPAL. *Notas de la Cepal.* (N. U. CEPAL, Ed.). (2012). Recuperado el 13 de Junio de 2021, de Discapacidad en América Latina y el Caribe, desafíos para las políticas públicas: https://www.cepal.org/notas/74/Titulares2.html

NAVARRO PIÑA, A. *Robot Industrial* (Primera ed.). (2021). (M. J. López Raso, Ed.) Madrid, España: Paraninfo. Recuperado el 25 de Julio de 2021, de https://books.google.com.ec/books?id=Q_APEAAAQBAJ&pg=PA4&dq=clasificaci%C3%B3n+de+robots+industriales&hl=es419&sa=X&ved=2ahUKEwi4tanvhd_zAhX_RTABHXmZD5EQ6AF6BAgLEAI#v=onepage&q&f=true

ORGANIZACIÓN MUNDIAL DE LA PROPIEDAD INTELECTUAL. *Informe mundial sobre la propiedad intelectual en 2015: La innovación revolucionaria y el crecimiento económico.* (2015). Ginebra, Suiza: OMPI. Recuperado el 25 de Octubre de 2021, de https://books.google.com.ec/books?id=M_OdDwAAQBAJ&printsec=frontcover#v=onepage&q&f=true

PARDO ALONSO, J. L. *Montaje y puesta en marcha de sistemas robóticos y sistemas de visión en bienes de equipo y maquinaria industrial.* (2012). Antequera, Málaga, España: IC Editorial. Recuperado el 25 de Junio de 2021, de https://books.google.com.ec/books?id=6VQpEAAAQBAJ&pg=PT291&dq=adquisici%C3%B3n+de+la+imagen+en+visi%C3%B3n+artificial&hl=es419&sa=X&ved=2ahUKEwj326qw9uPzAhUgSTABHUTrBsUQ6AF6BAgHEAI#v=onepage&q=adquisici%C3%B3n%20de%20la%20imagen%20en%20visi%C3%B3n%20arti

PÉREZ CISNEROS, M., CUEVAS JIMÉNEZ, E., & ZALDÍVAR NAVARRO, D. *Fundamentos de robótica y mecatrónica con Matlab y Simulink.* (2014). Madrid, España: RA-MA, S.A. Recuperado el 4 de Agosto de 2021, de https://books.google.com.ec/books?id=oo2fDwAAQBAJ&printsec=frontcover#v=onepage&q&f=true

PNEUMATIC. *Pneumatic-service.* (2019). Recuperado el 02 de 11 de 2021, de https://www.pneumatic-service.com.ar/wp-content/uploads/2017/01/4V200.pdf

PUBLICACIONES VÉRTICE S.L. *Implantación de espacios comerciales.* (2010). Málaga, España: Editorial Vértice. Recuperado el 2 de Julio de 2021, de https://books.google.com.ec/books?id=HWqXRig8HEYC&pg=PA154&dq=l%C3%A1mparas+fluorescentes&hl=es419&sa=X&ved=2ahUKEwiwg6TG9fDzAhUxSzABHRLwCCI4ChDoAXoECAIQAg#v=onepage&q=l%C3%A1mparas%20fluorescentes&f=true

QUISHPE, D. *Estudio de la nuemática y sus aplicaciones en diferentes campos de la industria.* (2018). Tesis Ingenieril, Universidad Central del Ecuador, Carrera de tecnología electromecánica, Quito. Recuperado el 17 de Junio de 2021, de http://www.dspace.uce.edu.ec/bitstream/25000/16881/1/T-UCE-0010-FIL-145.pdf

ROBOTANNO. *Robot Anno*. (2021). Recuperado el 25 de 10 de 2021

ROJAS HERNÁNDEZ , R., SILVA ORTIGOZA, R., & MOLINA VILCHIS, M. La Visión Artificial en la Robótica. *Polibits*(35), 22-28. (2007). Recuperado el 14 de Junio de 2021, de https://www.redalyc.org/pdf/4026/402640448005.pdf

ROJO, L. *La fibra óptica en la iluminación. iluminet.* (2015). Obtenido de https://www.iluminet.com/fibra-optica-iluminacion/

ROMERO CARRILLO, P. *Montaje y mantenimiento de líneas automatizadas* (Primera ed.). (2018). Madrid, España: Ediciones Paraninfo, SA. Recuperado el 29 de Julio de 2021, de https://books.google.com.ec/books?id=YoNZDwAAQBAJ&pg=PA139&dq=robots+scara&hl=es419&sa=X&ved=2ahUKEwj00fjor5v2AhWzRjABHVnsC0UQ6AF6BAgCEAI#v=onepage&q=robots%20scara&f=true

SÁNCHEZ DEL POZO, A. J., GÓMEZ JIMÉNEZ, J., & GÓMEZ JIMÉNEZ, J. *Simulación de sistemas mecatrónicos* (Primera ed.). (2021). Madrid, España: Ediciones Paraninfo. Recuperado el 24 de Julio de 2021, de https://books.google.com.ec/books?id=enI-EAAAQBAJ&pg=PA68&dq=adquisici%C3%B3n+de+la+imagen+en+visi%C3%B3n+artificial&hl=es-419&sa=X&ved=2ahUKEwj326qw9uPzAhUgSTABHUTrBsUQ6AF6BAgJEAI#v=onepage&q=adquisici%C3%B3n%20de%20la%20imagen%20en%20visi%C3%B3n%20artif

SÁNCHEZ FRÍAS, A. P., & TERÁN GORDILLO, A. F. *Diseño y construcción de un brazo robótico antropomórfico de siete grados de libertad con análisis cinemático y dinámico mediante algoritmos genéticos.* (2016). Tesis Ingenieril, Universidad de las Fuerzas Armadas, Departamento de ciencias de la energía y mecánica, Sangolquí . Recuperado el 25 de Junio de 2021, de http://repositorio.espe.edu.ec/handle/21000/11635

TRIVIÑO, M. *Diseño e implementación de un maniulador robot de asistencia a personas con disfunción motora.* (2018). Tesis Ingenieril, Universidad Politécnica de Madrid,

Ingeniería Eléctrica, Electrónica Automática y Física Aplicada, Madrid. Recuperado el 5 de Agosto de 2021, de https://oa.upm.es/49704/

VALLEJO VALENCIA, M., & ARIAS LONDOÑO, A. *Introducción a la adquisición y acondicionamiento de señales.* (2022). Medellín, Colombia: Instituto Tecnlógico Metropolitano. Recuperado el 26 de Febrero de 2022, de https://books.google.com.ec/books?id=KI5dEAAAQBAJ&pg=PA34&dq=sensores+pdf&hl=es419&sa=X&ved=2ahUKEwitq5j9yKb2AhXPQTABHb8sCkwQ6AF6BAgKEAI#v=onepage&q=sensores%20pdf&f=true

VALLEJO YÉPEZ, J. M. *Control bioeléctrico de un robot para la asistencia de discapacidad de extremidades superiores.* (2018). Tesis ingenieril, Escuela Superior Politécnica de Chimborazo, Riobamba. Recuperado el 20 de Agosto de 2021, de http://dspace.espoch.edu.ec/bitstream/123456789/8380/1/20T01025.pdf

ZABALA, G. *ROBÓTICA. Guía teórica y práctica.* (2007). Costa Rica: USERS Express. Recuperado el 25 de Octubre de 2021, de https://books.google.com.ec/books?id=JPgyRgn-j1YC&pg=PA19&dq=historia+del+robot+industrial&hl=es419&sa=X&ved=2ahUKEwjF6Zfko5v2AhWaSjABHWzWAU04ChDoAXoECAUQAg#v=onepage&q=historia%20del%20robot%20industrial&f=true

ANEXOS

Anexo A: Tabla de toma de tiempos para la velocidad al 5 %

Velocidad al 5 %							
Acercamiento cuchara-plato		Recolección de comida		Acercamiento plato-mentón		Acercamiento boca mentón	
N° Datos	Tiempo	N° Datos	Tiempo	N° Datos	Tiempo	N° Datos	Tiempo
1	18,3	1	12,14	1	15,75	1	11,6
2	17,93	2	13,3	2	15,41	2	11,42
3	17,59	3	12,03	3	15,35	3	10,18
4	18,32	4	13,54	4	15,29	4	11,5
5	17,63	5	12,12	5	15,35	5	10,79
6	17,61	6	12,6	6	15,10	6	10,21
7	17,75	7	12,57	7	15,13	7	11,81
8	18,9	8	13,67	8	15,57	8	10,78
9	18,56	9	12,45	9	15,89	9	11,06
10	18,5	10	12,06	10	15,16	10	11,09
11	17,26	11	13,09	11	15,42	11	10,11
12	17,77	12	13,15	12	15,41	12	11,14
13	17,07	13	13,33	13	15,60	13	11,33
14	17,99	14	13,97	14	15,82	14	11,29
15	18,86	15	12,57	15	15,98	15	11,44
16	17,52	16	13,37	16	15,42	16	11,48
17	17,12	17	12,3	17	15,82	17	11,8
18	18,47	18	13,58	18	15,88	18	10,73
19	18,22	19	13,28	19	15,42	19	11,38
20	17,98	20	12,47	20	15,34	20	11,81
21	17,79	21	13,25	21	15,79	21	10,65
22	18,34	22	13,85	22	15,14	22	10,17
23	17,17	23	13,1	23	15,09	23	11,04
24	17,99	24	13,48	24	15,34	24	11,08
25	18,13	25	13,31	25	15,85	25	10,1
26	17,85	26	12,5	26	15,39	26	11,48
27	17,65	27	13,9	27	15,88	27	10,87
28	17,74	28	13,66	28	15,68	28	10,44
29	18,36	29	13,03	29	15,64	29	11,41
30	18,82	30	13,9	30	15,99	30	10,76
Promedio =	17,97		13,05		15,53		11,03

Anexo B: Tabla de toma de tiempos para la velocidad al 10 %

Velocidad al 10 %							
Acercamiento cuchara-plato		Recolección de comida		Acercamiento plato-mentón		Acercamiento boca mentón	
N° Datos	Tiempo	N° Datos	Tiempo	N° Datos	Tiempo	N° Datos	Tiempo
1	13,01	1	7,79	1	11,68	1	6,17
2	13,84	2	8,79	2	11,13	2	6,23
3	12,39	3	8,52	3	11,83	3	6,51
4	12,79	4	8,42	4	11,76	4	6,57
5	12,75	5	7,67	5	11,23	5	6,91
6	13,52	6	7,71	6	11,19	6	6,14
7	13	7	8,82	7	11,46	7	6,01
8	13,22	8	8,83	8	11,95	8	6,87
9	13,52	9	7,78	9	12,09	9	6,34
10	12,22	10	8,61	10	11,15	10	6,71
11	12,19	11	7,96	11	11,09	11	6,31
12	12,85	12	8,66	12	11,54	12	6,11
13	12,41	13	7,66	13	11,51	13	6,9
14	13,66	14	7,78	14	11,64	14	6,25
15	13,49	15	7,59	15	12,03	15	6,1
16	13,32	16	8,64	16	11,42	16	6,4
17	12,56	17	7,83	17	11,08	17	6,42
18	12,56	18	7,39	18	11,22	18	6,37
19	12,38	19	8,49	19	11,11	19	6,04
20	13,59	20	8,59	20	11,25	20	6,03
21	13,95	21	8,07	21	11,38	21	6,05
22	12,23	22	7,83	22	11,32	22	6,82
23	13,7	23	8,91	23	12,03	23	6,32
24	13,2	24	7,29	24	11,49	24	6,41
25	12,94	25	7,13	25	11,53	25	6,76
26	12,06	26	7,37	26	11,15	26	6,01
27	13,92	27	7,42	27	11,07	27	6,18
28	13,58	28	7,39	28	11,27	28	6,91
29	12,87	29	8,88	29	11,28	29	6,34
30	12,55	30	7,51	30	11,68	30	6,31
Promedio =	13,01		8,04		11,45		6,38

Anexo C: Tabla de toma de tiempos para la velocidad al 15 %

Velocidad al 15 %							
Acercamiento cuchara-plato		Recolección de comida		Acercamiento plato-mentón		Acercamiento boca mentón	
N° Datos	Tiempo	N° Datos	Tiempo	N° Datos	Tiempo	N° Datos	Tiempo
1	10,91	1	5,98	1	7,66	1	3,61
2	10,45	2	5,03	2	7,33	2	4,51
3	10,23	3	5,31	3	7,50	3	4,76
4	9,52	4	5,61	4	7,53	4	4,32
5	9,11	5	5,72	5	7,45	5	3,8
6	10,12	6	5,49	6	7,80	6	4,41
7	10,02	7	5,39	7	7,18	7	4,4
8	10,01	8	5,8	8	7,21	8	4,18
9	9,95	9	5,04	9	7,84	9	3,94
10	10,03	10	5,39	10	7,39	10	4,89
11	10,72	11	5,97	11	7,21	11	3,47
12	10,28	12	5,29	12	7,55	12	4,99
13	10,41	13	5,29	13	7,54	13	3,75
14	10,84	14	5,83	14	7,05	14	3,2
15	9,3	15	5,06	15	7,98	15	3,24
16	10,83	16	5,18	16	7,72	16	3,77
17	9,33	17	5,75	17	7,72	17	4,41
18	10,44	18	5,69	18	7,63	18	4,48
19	9,26	19	5,89	19	7,98	19	3,54
20	10,35	20	5,42	20	7,07	20	3,17
21	10,31	21	5,65	21	7,17	21	3,87
22	10,38	22	5,45	22	7,34	22	4,46
23	9,98	23	5,98	23	7,64	23	4,57
24	9,88	24	5,25	24	7,50	24	3,69
25	9,19	25	5,22	25	7,60	25	3,43
26	9,32	26	5,68	26	7,54	26	3,57
27	9,26	27	5,22	27	7,79	27	4,12
28	10,8	28	5,43	28	8,00	28	4,48
29	10,76	29	5,04	29	7,82	29	4,54
30	9,13	30	5,27	30	7,28	30	4,29
Promedio =	10,04		5,48		7,53		4,06

Anexo D: Tabla de tiempos de reconocimiento facial

Reconocimiento facial		Condicional SI		Boca cerrada		Boca Abierta		Rostro fuera de rango		Condicional NO	
N° Datos	Tiempo	N° Datos	Tiempo	N° Datos	Tiempo	N° Datos	Tiempo	N° Datos	Tiempo	N° Datos	Tiempo
1	0,00338248	1	0,35616007	1	0,4416737	1	0,03133652	1	0,31100709	1	0,13184117
2	0,137089326	2	0,86686734	2	0,1356383	2	0,87361795	2	0,50848196	2	0,31926687
3	0,864286006	3	0,26833152	3	0,4013857	3	0,84924194	3	0,04111213	3	0,26399799
4	0,870743298	4	0,30023422	4	0,7319827	4	0,88728441	4	0,27917703	4	0,16666332
5	0,952140435	5	0,84917729	5	0,0842536	5	0,77408901	5	0,09546653	5	0,63069484
6	0,270587963	6	0,63478213	6	0,9370253	6	0,19285505	6	0,7314538	6	0,23506475
7	0,188287599	7	0,35656856	7	0,9656743	7	0,55262786	7	0,08867747	7	0,95320156
8	0,067037995	8	0,98482103	8	0,1207278	8	0,60641847	8	0,18837453	8	0,67565823
9	0,05481449	9	0,9229395	9	0,4989874	9	0,16921913	9	0,73091139	9	0,03550822
10	0,969161025	10	0,7085732	10	0,7181204	10	0,43032854	10	0,90695638	10	0,53182011
11	0,399968574	11	0,95548115	11	0,4626735	11	0,60163647	11	0,12801806	11	0,61470918
12	0,061761588	12	0,67790968	12	0,4850193	12	0,28832312	12	0,64895704	12	0,30061785
13	0,101307104	13	0,79407744	13	0,3983006	13	0,69857268	13	0,08676976	13	0,65435976
14	0,075840567	14	0,47847561	14	0,2852771	14	0,10031198	14	0,53265232	14	0,72930827
15	0,505559893	15	0,23332066	15	0,7063895	15	0,49989521	15	0,23687309	15	0,70458666
16	0,377954833	16	0,26082495	16	0,1321144	16	0,64231673	16	0,59392773	16	0,5712293
17	0,406084079	17	0,30829998	17	0,8176677	17	0,1819961	17	0,13510371	17	0,87492062
18	0,651945135	18	0,90325321	18	0,7374048	18	0,25260809	18	0,20007306	18	0,93629321
19	0,058566471	19	0,71539995	19	0,2632029	19	0,98967007	19	0,25190562	19	0,00490336
20	0,611442601	20	0,86456596	20	0,2002526	20	0,72527088	20	0,47718143	20	0,00660904
21	0,245862555	21	0,08398983	21	0,5062195	21	0,88266205	21	0,36152563	21	0,01892296
22	0,274526837	22	0,82953221	22	0,7155742	22	0,50038136	22	0,30418309	22	0,50577525
23	0,271788648	23	0,31096902	23	0,3640076	23	0,88254717	23	0,38807677	23	0,55565141
24	0,183390047	24	0,56582641	24	0,9671353	24	0,77319355	24	0,43619546	24	0,60959344
25	0,125949112	25	0,86880377	25	0,0215845	25	0,10579686	25	0,85580289	25	0,30836826
26	0,511695016	26	0,66495362	26	0,1995302	26	0,58432419	26	0,82490857	26	0,81423885
27	0,308466252	27	0,36874129	27	0,2224154	27	0,27849847	27	0,15623803	27	0,18972369
28	0,019125101	28	0,29893853	28	0,3011504	28	0,11000006	28	0,83106224	28	0,45414824
29	0,891787626	29	0,12953967	29	0,9835875	29	0,61602324	29	0,37295925	29	0,24453464
30	0,15053518	30	0,48915867	30	0,8345888	30	0,52213905	30	0,9550949	30	0,30045828
Promedio =	0,3537029		0,568351		0,4879855		0,5201062		0,4219709		0,4447556

Anexo E: Tabla de tiempos de reconocimiento entre ejes centrales

Ejes centrales	
N° Datos	Tiempo
1	0,92779
2	0,98168
3	0,49964
4	0,45032
5	0,38312
6	0,05508
7	0,60605
8	0,78405
9	0,28093
10	0,33645
11	0,07634
12	0,15011
13	0,32531
14	0,00541
15	0,77352
16	0,96167
17	0,19445
18	0,13729
19	0,07732
20	0,84570
21	0,70777
22	0,36238
23	0,85691
24	0,39673
25	0,23424
26	0,70272
27	0,75753
28	0,73645
29	0,62569
30	0,78765
Promedio =	0,50068

Anexo F: Codificación de la interfaz en python

```
import pygame
import cv2
import numpy as np
import mediapipe as mp
import time
import math
## VISION ROSTRO
mp_face_mesh = mp.solutions.face_mesh
mp_drawing = mp.solutions.drawing_utils
index_list =  [291, 409, 270, 269, 267, 0 ,37 , 39 , 40 , 185  , 61, 146 , 91, 181,84 ,
17,314,405,321, 375]
index_list2 = [415,310,311, 312,13,82,81,80,191,78 , 95 ,88 ,178 ,87 ,14,317,402,318,324,308]
prev_frame_time = 0
new_frame_time = 0
RosadoB1 = np.array([160, 0,0], np.uint8)
RosadoA1 = np.array([180, 255,255], np.uint8)
RosadoB2 = np.array([0, 50,150], np.uint8)
RosadoA2 = np.array([10, 255,255], np.uint8)
PROSX = 0
PROSY = 0
h = 0
w = 0
w2 = 0
h2 = 0
CENTRO_X = []
CENTRO_Y = []
ESTADO = []
CONDICIONAL_SI = 0
CONTADOR_SI    = 0
A_RRIBA = 0
A_BAJO  = 0
CONDICIONAL_NO = 0
CONTADOR_NO    = 0
D_ERECHA    = 0
I_ZQUIERDA  = 0
```

```
ENCENDIDO = 0
INICIO = 0
TXT = ""
COLOR =(0,0,0)
N_fps = 0
###________
def angle_between(p1, p2):
        if len(p1) != 2 or len(p2) != 2 :
                P2 = 0
                return P2
        elif (p2[0]-p1[0]) != 0:
                Ang = (p2[1]-p1[1])/(p2[0]-p1[0])
                P = math.atan(Ang)
                P2 = P*(180/3.141592653589793)
                return P2
        else :
                P2 = 0
                return P2
cap =  cv2.VideoCapture('Videos/S4.mp4')
cap2 = cv2.VideoCapture('Videos/8008.mp4')
cap0 = cv2.VideoCapture("Videos/3003.mp4")
cap1 =  cv2.VideoCapture(1)
cap21 = cv2.VideoCapture(2)
cap01 = cv2.VideoCapture(0)
"""
cap1 =  cv2.VideoCapture('Videos/2022A.mp4')
cap21 = cv2.VideoCapture('Videos/8008A.mp4')
cap01 = cv2.VideoCapture("Videos/3003A.mp4")
"""
##BRAZO
X11 = 1920
Y11 = 1080
Y21 = 1280
X21 = 720
medidas1 = 0
medidas2 = 0
```

```
prev_frame_time2 = 0
new_frame_time2 = 0
BlancoB = np.array([0, 0,0], np.uint8)
BlancoA = np.array([180, 255,10], np.uint8)
AzulB = np.array([112, 110,135], np.uint8)
AzulA = np.array([120, 170,205], np.uint8)
CafeB = np.array([20, 80,140], np.uint8)
CafeA = np.array([26, 180,220], np.uint8)
NaranjaB = np.array([0, 60,160], np.uint8)
NaranjaA = np.array([12, 220,255], np.uint8)
RosadoB = np.array([164, 80,140], np.uint8)
RosadoA = np.array([180, 200,255], np.uint8)
NaranjaB2 = np.array([112, 100,180], np.uint8)
NaranjaA2 = np.array([120, 140,240], np.uint8)
PROS = (0, 0)
PNAR = (0, 0)
PCAF = (0, 0)
PAZU = (0, 0)
PBLA = (0, 0)
PNAR2 = (0, 0)
MedidasLateral = 0
A = 1
B = 1
frame_counter = 0
N_fps2 = 0
#cap01 = cv2.VideoCapture("Videos/3003A.mp4")
"""
video_brazo = cv2.VideoCapture("Videos/Brazo Frontal.mp4")
video_brazo_lateral = cv2.VideoCapture("Videos/Brazo Lateral.MP4")
video_rostro = cv2.VideoCapture("Videos/Rostro.mp4")
"""
##INICIAR PYGAME
pygame.font.init()
pygame.init()
```

```
##CLASES
class mano(pygame.sprite.Sprite):
	def __init__(self):
		super().__init__()
		self.image = pygame.image.load("INTERFAZ/MANOX.png").convert()
		self.image.set_colorkey(NEGRO)
		self.rect = self.image.get_rect()
class pguia1(pygame.sprite.Sprite):
	def __init__(self):
		super().__init__()
		self.image = pygame.image.load("INTERFAZ/Guia1.png").convert()
		self.image.set_colorkey(BLANCO)
		self.rect = self.image.get_rect()
class pguia2(pygame.sprite.Sprite):
	def __init__(self):
		super().__init__()
		self.image = pygame.image.load("INTERFAZ/Guia2.png").convert()
		self.image.set_colorkey(NEGRO)
		self.rect = self.image.get_rect()
###COLORES
CELESTE = (0,162,232)
NARANJA = (255,127,39)
VERDE1 = (181,230,30)
MORADO = (112,146,190)
ROSA =  (245,143,139)
NEGRO   = (0,0,0)
GRIS1   = (128,128,128)
GRIS2   = (64,64,64)
AMARILLO = (255,255,0)
VERDE2 = (0,255,0)
VERDE = (25,150,25)
ROJO =  (255,0,0)
AZUL =          (0,0,255)
BLANCO = (255,255,255)
COLORCELESTE =     (153,217,234)
COLORCREMA =    (239,228,176)
```

```
### CREAR HOJA DE TRABAJO
screen = pygame.display.set_mode([1200, 640])
FONDO = pygame.image.load("INTERFAZ/FONDOSAMIR2.png").convert()
### RELOJ
clock = pygame.time.Clock()
## CREACION DE GRUPOS
all_sprite_list = pygame.sprite.Group()
## AGREGAR
Mano = mano()
Pguia1 = pguia1()
Pguia2 = pguia2()
all_sprite_list.add(Mano)
all_sprite_list.add(Pguia1)
all_sprite_list.add(Pguia2)
pygame.mouse.set_visible(1)
## VARIABLES UNIVERSAL
done = False
THE_X = 0
CONFIGURACION = 1
ESTADO_PULSO = 0
PG2 = 1200
PG3 = 1200*2
PG4 = 1200*3
COMENZAR = 0
COMENZAR2 = 0
frame_counter2 = 0
X_R = 1
Y_R = 1
##VISION ROSTRO
GUIA22 = 0
GUIA33 = 0
CAMARA22 = 0
CAMARA33 = 0
with mp_face_mesh.FaceMesh(static_image_mode=False,
max_num_faces=1,min_detection_confidence=0.5) as face_mesh:
    while not done:
        for event in pygame.event.get():
```

```
            if event.type == pygame.QUIT:
                done = True
            if event.type == pygame.KEYDOWN :
                if event.key == pygame.K_ESCAPE :
                    done = True
        ##MOUSE
        mouse_pos = pygame.mouse.get_pos()
        Mano.rect.x = mouse_pos[0] - 12
        Mano.rect.y = mouse_pos[1]
        if GUIA22 == 1 :
            Pguia1.rect.x = 40 + THE_X + PG2
            Pguia1.rect.y = 237
        else:
            Pguia1.rect.y = 700
        if GUIA33 == 1 :
            Pguia2.rect.x = 40 + THE_X + PG3
            Pguia2.rect.y = 237
        else:
            Pguia2.rect.y = 700
        ## INTER PAG 1
        CUADRO1 = pygame.draw.rect(screen, (80,160,80)    ,(433+ THE_X,518,80,55) )
        CUADRO2 = pygame.draw.rect(screen, (160,80,80)    ,(899+ THE_X,518,80,55) )
        ## INTER PAG 2
        CAMARA2 = pygame.draw.rect(screen,  (180,40,180),(280+ THE_X + PG2  ,160,115,55)
)
        GUIA2    = pygame.draw.rect(screen,  (70,0,150),(165+ THE_X + PG2  ,160,115,55) )
        INICIAR2 = pygame.draw.rect(screen, (40,180,180),(915+ THE_X + PG2  ,160,115,55) )
        DETENER2 = pygame.draw.rect(screen, (200,40,40) ,(800+ THE_X + PG2  ,160,115,55) )
        ## INTER PAG 3
        CAMARA3 = pygame.draw.rect(screen,  (180,40,180),(280+ THE_X + PG3  ,160,115,55)
)
        GUIA3   = pygame.draw.rect(screen,  (70,0,150),(165+ THE_X + PG3  ,160,115,55) )
        INICIAR3 = pygame.draw.rect(screen, (40,180,180),(915+ THE_X + PG3  ,160,115,55) )
        DETENER3 = pygame.draw.rect(screen, (200,40,40) ,(800+ THE_X + PG3  ,160,115,55) )
        ##UNIVERSAL
        CUADROIZQ  = pygame.draw.rect(screen, (0,0,0) ,(1130 ,0,70,92))
        CUADRODER  = pygame.draw.polygon(screen, (255,0,0), [ (0,540),  (0,640), (100,640)])
```

```
        ##VER
        screen.blit(FONDO, [THE_X, 0])
        ##MOUSE POS
        MAN_O = pygame.draw.rect(screen, (0,255,0) ,(mouse_pos[0],mouse_pos[1],10,10) )
        if CONFIGURACION == 1 :
            THE_X = 0
            if MAN_O.colliderect(CUADRO1)  and event.type ==
pygame.MOUSEBUTTONDOWN and ESTADO_PULSO == 0:
                CONFIGURACION = 2
                ESTADO_PULSO = 1
            if MAN_O.colliderect(CUADRO2)  and event.type ==
pygame.MOUSEBUTTONDOWN and ESTADO_PULSO == 0:
                CONFIGURACION = 3
                ESTADO_PULSO = 1
        if CONFIGURACION == 2 :
            THE_X = -1200
            if MAN_O.colliderect(INICIAR2)  and event.type ==
pygame.MOUSEBUTTONDOWN and ESTADO_PULSO == 0:
                COMENZAR = 1
                ESTADO_PULSO = 1
            if MAN_O.colliderect(DETENER2)  and event.type ==
pygame.MOUSEBUTTONDOWN and ESTADO_PULSO == 0:
                COMENZAR = 0
                ESTADO_PULSO = 1
            if MAN_O.colliderect(CAMARA2)  and event.type ==
pygame.MOUSEBUTTONDOWN and ESTADO_PULSO == 0 and CAMARA22 == 0 :
                print("CAMARA 2 A")
                CAMARA22 = 1
                medidas1 = 0
                ESTADO_PULSO = 1
            if MAN_O.colliderect(CAMARA2)  and event.type ==
pygame.MOUSEBUTTONDOWN and ESTADO_PULSO == 0 and CAMARA22 == 1 :
                print("CAMARA 2 B")
                CAMARA22 = 0
                medidas1 = 0
                ESTADO_PULSO = 1
```

```
        if MAN_O.colliderect(GUIA2)  and event.type == pygame.MOUSEBUTTONDOWN
and ESTADO_PULSO == 0 and GUIA22 == 0:
            print("GUIA 2 A")
            GUIA22 = 1
            ESTADO_PULSO = 1
        if MAN_O.colliderect(GUIA2)  and event.type == pygame.MOUSEBUTTONDOWN
and ESTADO_PULSO == 0 and GUIA22 == 1:
            print("GUIA 2 B")
            GUIA22 = 0
            ESTADO_PULSO = 1
    else:
        COMENZAR = 0
    if CONFIGURACION == 3 :
        THE_X = -2400
        if MAN_O.colliderect(INICIAR3)  and event.type ==
pygame.MOUSEBUTTONDOWN and ESTADO_PULSO == 0:
            COMENZAR2 = 1
            ESTADO_PULSO = 1
        if MAN_O.colliderect(DETENER3)  and event.type ==
pygame.MOUSEBUTTONDOWN and ESTADO_PULSO == 0:
            COMENZAR2 = 0
            ESTADO_PULSO = 1
        if MAN_O.colliderect(CAMARA3)  and event.type ==
pygame.MOUSEBUTTONDOWN and ESTADO_PULSO == 0 and CAMARA33 == 0:
            print("CAMARA 3 A ")
            CAMARA33 = 1
            medidas2 = 0
            ESTADO_PULSO = 1
        if MAN_O.colliderect(CAMARA3)  and event.type ==
pygame.MOUSEBUTTONDOWN and ESTADO_PULSO == 0 and CAMARA33 == 1:
            print("CAMARA 3 B")
            CAMARA33 = 0
            medidas2 = 0
            ESTADO_PULSO = 1

        if MAN_O.colliderect(GUIA3)  and event.type == pygame.MOUSEBUTTONDOWN
and ESTADO_PULSO == 0 and GUIA33 == 0:
```

```
            print("GUIA 3 A")
            GUIA33 = 1
            ESTADO_PULSO = 1

        if MAN_O.collliderect(GUIA3)  and event.type == pygame.MOUSEBUTTONDOWN
and ESTADO_PULSO == 0 and GUIA33 == 1:
            print("GUIA 3 B")
            GUIA33 = 0
            ESTADO_PULSO = 1

    else:
        COMENZAR2 = 0



    clock.tick(30)

    ##UNIVERSAL
    if ( MAN_O.collliderect(CUADRODER) or MAN_O.collliderect(CUADROIZQ) ) and
event.type == pygame.MOUSEBUTTONDOWN and ESTADO_PULSO == 0:
        CONFIGURACION = 1
        ESTADO_PULSO = 1

    if event.type == pygame.KEYUP or event.type == pygame.MOUSEBUTTONUP :
        ESTADO_PULSO = 0
        #print("levantado")
    else:
        #print("click")
        pass



    if CONFIGURACION == 2 and COMENZAR == 1 :
        if CAMARA22 == 1 :
            success, frame = cap.read()
            ret2, frame2 = cap2.read()
        else:
            success, frame = cap1.read()
            ret2, frame2 = cap21.read()
```

```
        if medidas1 == 0 :
            alto1, ancho1, _ = frame.shape
            X11 = alto1/1080
            Y11 = ancho1/1920
            alto2, ancho2, _ = frame2.shape
            Y21 = alto2/720
            X21 = ancho2/1280
            print(X11,"X11")
            print(Y11,"Y11" )
            medidas1 = 1


        if success:
            #RESET VIDEO
            frame_counter += 1
            if frame_counter == int(cap.get(cv2.CAP_PROP_FRAME_COUNT)):
                frame_counter = 0 #Or whatever as long as it is the same as next line
                cap.set(cv2.CAP_PROP_POS_FRAMES, 0)
                cap2.set(cv2.CAP_PROP_POS_FRAMES, 0)

            Simp2 = frame2[int(60*Y21):int(250*Y21),int(450*X21):int(620*X21)]
            cv2.rectangle(frame2, (int(200*X21),int(20*Y21)), (int(880*X21),int(750*Y21)),
(255, 0, 255), 2)
            SimpHSV2 = cv2.cvtColor(Simp2, cv2.COLOR_BGR2HSV)

            #_____________________CENTRO_____________________#
            SimpColorNaranja2 = cv2.inRange(SimpHSV2, NaranjaB2 , NaranjaA2)
            SimpColorNaranja2 = cv2.dilate(SimpColorNaranja2, np.ones((9,9),np.uint8),
iterations=2)
            cntsNARA2,_  =
cv2.findContours(SimpColorNaranja2,cv2.RETR_LIST,cv2.CHAIN_APPROX_SIMPLE)[-2:]

            for cntx in cntsNARA2:
                x,y,w,h = cv2.boundingRect(cntx)
                if w > int(20*X21) and h > int(20*Y21) :
                    cv2.rectangle(frame2,(x+int(450*X21)-5,y+int(60*Y21)-
5),(x+w+int(450*X21)+5,y+h+int(60*Y21)+5),(180,50,50),3)
```

```
                PROSX = x + int(0.5*w)
                PROSY = y + int(0.5*h)
                PNAR2 = (PROSX+int(450*X21) , PROSY +int(60*Y21))
                cv2.circle(frame2, PNAR2, 1, (255,255,255), 4)

            ###FPS
            new_frame_time2 = time.time()
            fps = 1/(new_frame_time2-prev_frame_time2)
            prev_frame_time2 = new_frame_time2
            N_fps2 = int((fps*100))/100



            Simp = frame[int(80*Y11):int(870*Y11),int(200*X11):int(1400*X11)]

            cv2.rectangle(frame, (int(200*Y11),int(80*Y11)), (int(1400*X11),int(870*Y11)),
(255, 0, 255), 2)
            SimpHSV = cv2.cvtColor(Simp, cv2.COLOR_BGR2HSV)



            #____________________CAFE____________________#
            SimpColorCafe = cv2.inRange(SimpHSV, CafeB , CafeA)
            SimpColorCafe  = cv2.dilate(SimpColorCafe, np.ones(( int(9*X11)   ,   int(9*Y11)
),np.uint8), iterations=3)
            cntsCAFE,_  =
cv2.findContours(SimpColorCafe,cv2.RETR_LIST,cv2.CHAIN_APPROX_SIMPLE)[-2:]

            for cntx in cntsCAFE:
               x,y,w,h = cv2.boundingRect(cntx)
               if w > int(X11*30) and h > int(Y11*30) :
                  cv2.rectangle(frame,(x+int(200*X11)-
5,y+int(80*Y11)),(x+w+int(200*X11),y+h+int(80*Y11)),(80,80,180),3)
                  PROSX = x + int(0.5*w)
                  PROSY = y + int(0.5*h)
                  PCAF = (PROSX+int(200*X11) , PROSY +int(80*Y11))
                  cv2.circle(frame, PCAF, 1, (255,255,255), 4)

            #____________________ROSADO____________________#
```

```
        SimpColorRosado = cv2.inRange(SimpHSV, RosadoB , RosadoA)
        SimpColorRosado  = cv2.dilate(SimpColorRosado, np.ones(( int(9*X11)   ,
int(9*Y11) ),np.uint8), iterations=3)
        cntsROSA,_  =
cv2.findContours(SimpColorRosado,cv2.RETR_LIST,cv2.CHAIN_APPROX_SIMPLE)[-2:]

        for cntx in cntsROSA:
           x,y,w,h = cv2.boundingRect(cntx)
           if w > int(X11*40) and h > int(Y11*40) :
              cv2.rectangle(frame,(x+int(200*X11)-5,y+int(80*Y11)-
5),(x+w+int(200*X11)+5,y+h+int(80*Y11)+5),(80,80,180),3)
              PROSX = x + int(0.5*w)
              PROSY = y + int(0.5*h)
              PROS = (PROSX+int(200*X11) , PROSY +int(80*Y11))
              cv2.circle(frame, PROS, 1, (255,255,255), 4)

        #____________________NARANJA____________________#
        SimpColorNaranja = cv2.inRange(SimpHSV, NaranjaB , NaranjaA)
        SimpColorNaranja = cv2.dilate(SimpColorNaranja, np.ones(( int(9*X11)   ,
int(9*Y11) ),np.uint8),iterations=2)
        cntsNARA,_  =
cv2.findContours(SimpColorNaranja,cv2.RETR_LIST,cv2.CHAIN_APPROX_SIMPLE)[-2:]

        for cntx in cntsNARA:
           x,y,w,h = cv2.boundingRect(cntx)
           if w > int(X11*20) and h > int(Y11*20) :
              cv2.rectangle(frame,(x+int(200*X11)-5,y+int(80*Y11)-
5),(x+w+int(200*X11)+5,y+h+int(80*Y11)+5),(180,50,50),3)
              PROSX = x + int(0.5*w)
              PROSY = y + int(0.5*h)
              PNAR = (PROSX+int(200*X11) , PROSY +int(80*Y11))
              cv2.circle(frame, PNAR, 1, (255,255,255), 4)

        #____________________AZUL____________________#
        SimpColorAzul = cv2.inRange(SimpHSV, AzulB , AzulA)
        SimpColorAzul  = cv2.dilate(SimpColorAzul, np.ones(( int(9*X11)   ,   int(9*Y11)
),np.uint8), iterations=2)
```

```
        cntsAZUL,_  =
cv2.findContours(SimpColorAzul,cv2.RETR_LIST,cv2.CHAIN_APPROX_SIMPLE)[-2:]

        for cntx in cntsAZUL:
            x,y,w,h = cv2.boundingRect(cntx)
            if w > int(X11*20) and h > int(Y11*20) :
                cv2.rectangle(frame,(x+int(200*X11)-5,y+int(80*Y11)-
5),(x+w+int(200*X11)+5,y+h+int(80*Y11)+5),(180,50,50),3)
                PROSX = x + int(0.5*w)
                PROSY = y + int(0.5*h)
                PAZU = (PROSX+int(200*X11)-5 , PROSY +int(80*Y11)-5)
                cv2.circle(frame, PAZU, 1, (0,0,0), 4)



        ANGULOnegro = angle_between(PAZU, PNAR)
        cv2.putText(frame, str(int(ANGULOnegro*100)/100), (PAZU[0]+ int(X11*50),
PAZU[1]), cv2.FONT_HERSHEY_SIMPLEX, round(2*X11), (80, 80, 80), 2, cv2.LINE_AA)
        cv2.line(frame, PAZU, PNAR, (0,0,0), int(3*X11)  )

        ANGULOblanco = angle_between( PROS  ,PCAF )
        cv2.putText(frame, str(int((ANGULOblanco+180)*100)/100), (PROS[0]+
int(X11*50), PROS[1]- int(Y11*10)), cv2.FONT_HERSHEY_SIMPLEX, round(2*X11), (255,
255, 255),2, cv2.LINE_AA)
        cv2.line(frame, PCAF, PROS, (255,255,255), int(3*X11) )



        cv2.putText(frame, str(N_fps2), (int(40), int(40)), cv2.FONT_HERSHEY_SIMPLEX,
1, (0,0,0), 1, cv2.LINE_AA)

        cv2.imshow('frame', SimpColorCafe)
        #cv2.imshow('frame2', frame2)
        k = cv2.waitKey(1) & 0xFF
        if k == 27:
            break

        video_image  = cv2.resize(frame, (560, 315))
```

```
            video_image2  = cv2.resize(frame2, (560, 315))

            video_surf = pygame.image.frombuffer(video_image.tobytes(),
video_image.shape[1::-1], "BGR")
            video_surf2 = pygame.image.frombuffer(video_image2.tobytes(),
video_image2.shape[1::-1], "BGR")
            screen.blit(video_surf, (40, 237))
            screen.blit(video_surf2, (40+560, 237))

        if CONFIGURACION == 3  and   COMENZAR2 == 1  :
          if CAMARA33 == 1 :
            success, frame = cap0.read()
          else:
            success, frame = cap01.read()

          if medidas2 == 0 :
            Alto3, Ancho3, _ = frame.shape
            Y_R = Alto3/720
            X_R = Ancho3/1280
            print(Y_R,"X11")
            print(X_R,"Y11" )
            medidas2 = 1
          if success:
            frame_counter2 += 1
            ##VISION ROSTRO
            #FPS
            new_frame_time = time.time()
            fps = 1/(new_frame_time-prev_frame_time)
            prev_frame_time = new_frame_time
            N_fps = int((fps*100))/100
            frame = cv2.flip(frame,1)
            height, width, _ = frame.shape
            frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
            results = face_mesh.process(frame_rgb)
            mascara = np.zeros_like(frame)
```

```
            cv2.putText(frame, str(N_fps), (int(40*X_R), int(40*Y_R)),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0,0,0), 1, cv2.LINE_AA)

            Lista = []
            Lista2 = []

            if results.multi_face_landmarks is not None:
               INICIO = 1
               for face_landmarks in results.multi_face_landmarks:
                  mp_drawing.draw_landmarks(mascara,
face_landmarks,mp_face_mesh.FACE_CONNECTIONS)
            else:
               INICIO = 0

            if INICIO == 0:
               cv2.rectangle(mascara, (int(250*X_R), int(430*Y_R)), (int(1055*X_R),
int(530*Y_R)), (176, 228, 239), -1)
               cv2.rectangle(mascara, (int(255*X_R), int(435*Y_R)), (int(1050*X_R),
int(525*Y_R)), (255, 255, 255), 1)
               cv2.putText(mascara, str("NO SE DETECTA ROSTRO"), (int(280*X_R),
int(500*Y_R)), cv2.FONT_HERSHEY_SIMPLEX, 2*X_R, (0, 0, 0), round(3*X_R),
cv2.LINE_AA)

            if INICIO == 1 :
               for index in index_list:
                  x = int(face_landmarks.landmark[index].x * width)
                  y = int(face_landmarks.landmark[index].y * height)
                  qp = x,y
                  Lista.append(qp)
               for index in index_list2:
                  x = int(face_landmarks.landmark[index].x * width)
                  y = int(face_landmarks.landmark[index].y * height)
                  qp2 = x,y
                  Lista2.append(qp2)

               if len(Lista) > 0 and len(Lista2) > 0  :
                  area_pts = np.array([Lista])
```

```
                imAux = np.zeros(shape=(frame.shape[:2]), dtype=np.uint8)
                imAux = cv2.drawContours(imAux, [area_pts], -1, (255), -1)

                area_pts2 = np.array([Lista2])
                imAux2 = np.zeros(shape=(frame.shape[:2]), dtype=np.uint8)
                imAux2P = cv2.drawContours(imAux2, [area_pts2], -1, (255), -1)

                NimAux2 = cv2.bitwise_not(imAux2)
                AND2 = cv2.bitwise_and(imAux,NimAux2)
                mascara = cv2.bitwise_and(frame, frame, mask=AND2)
                ANDO = mascara
                imAux2 = cv2.bitwise_and(frame, frame, mask=imAux2P)

            cntsNARA,_  =
cv2.findContours(imAux2P,cv2.RETR_LIST,cv2.CHAIN_APPROX_SIMPLE)[-2:]
            for cntx in cntsNARA:
                x,y,w,h = cv2.boundingRect(cntx)
                PROSX = x + int(0.5*w)
                PROSY = y + int(0.5*h)
                PNAR22 = (PROSX , PROSY )
                cv2.circle(frame, PNAR22, 1, (255,255,255), 4)
                cv2.circle(mascara, PNAR22, 1, (255,255,255), 4)

            for face_landmarks in results.multi_face_landmarks:
                mp_drawing.draw_landmarks(mascara, face_landmarks,
                    mp_face_mesh.FACE_CONNECTIONS,
                    mp_drawing.DrawingSpec(color=(0, 0, 0), thickness=1, circle_radius=1),
                    mp_drawing.DrawingSpec(color=(255, 0, 0), thickness=1))

            ##CONDIONAL SI
            CENTRO_X.append(PROSX)
            C_X= np.mean(CENTRO_X)

            CENTRO_Y.append(PROSY)
            C_Y= np.mean(CENTRO_Y)

            C_X = int(C_X)
```

```
            C_Y = int(C_Y)

            if len(CENTRO_X) == 21 :
               CENTRO_X.pop(0)
            if len(CENTRO_Y) == 21 :
               CENTRO_Y.pop(0)

            if len(ESTADO) == 6 :
               ESTADO.pop(0)


            if sum(ESTADO) == 5000 :
               TXT = "BOCA CERRADA"
               COLOR = (0,160,40)
            elif sum(ESTADO) == 500   :
               TXT = "BOCA ABIERTA"
               COLOR = (50,50,200)
            elif sum(ESTADO) == 50 :
               TXT = "VISION BLOQUEADA"
               COLOR = (151,69,165)


            cv2.circle(frame, (C_X,C_Y), 1, (155,40,40), 4)
            cv2.line(frame, (C_X+50,C_Y), (C_X-50,C_Y), (180,180,180), 1)
            cv2.line(frame, (C_X,C_Y+50), (C_X,C_Y-50), (180,180,180), 1)

            ##CONDICIONAL SI
            if PROSY+int(30*Y_R) < C_Y and (PROSX+int(5*X_R)  > C_X or PROSX-
int(5*X_R) < C_X)  and A_RRIBA == 0 and CONDICIONAL_SI == 0 :
               print("ARRIBA")
               A_RRIBA = 1
            if PROSY-int(30*Y_R) > C_Y and (PROSX+int(5*X_R) > C_X or PROSX-
int(5*X_R) < C_X) and A_BAJO == 0  and CONDICIONAL_SI == 0 :
               print("ABAJO")
               A_BAJO  = 1

            if A_RRIBA == 1 and A_BAJO == 1 and CONDICIONAL_SI == 0   :
```

```
            CONTADOR_SI = CONTADOR_SI + 1
            A_RRIBA = 0
            A_BAJO = 0

        if CONTADOR_SI == 3 :
            CONDICIONAL_SI = 1
            ENCENDIDO = 1

        ##CONDICIONAL NO

        if PROSX+int(60*X_R) < C_X and (PROSY+int(5*Y_R)> C_Y or PROSY-
int(5*Y_R) < C_Y)  and I_ZQUIERDA == 0  and CONDICIONAL_NO == 0 :
            print("IZQUIERDA")
            I_ZQUIERDA  = 1

        if PROSX-int(60*X_R) > C_X and (PROSY+int(5*Y_R) > C_Y or PROSY-
int(5*Y_R) < C_Y) and D_ERECHA == 0 and CONDICIONAL_NO == 0 :
            print("DERECHA")
            D_ERECHA = 1

        if I_ZQUIERDA == 1 and D_ERECHA == 1 and CONDICIONAL_NO == 0   :
            CONTADOR_NO = CONTADOR_NO + 1
            I_ZQUIERDA = 0
            D_ERECHA = 0

        if CONTADOR_NO == 2 :
            CONDICIONAL_NO = 1
            ENCENDIDO = 0
            print("FUNCIONANDO NO ")


        #print(CONTADOR_NO , "CONTADOR_NO")

        if   ENCENDIDO == 0 :
            cv2.circle(mascara, (int(1050*X_R), int(50*Y_R)), 12, (40, 40, 255), -1)
            cv2.putText(mascara, str("APAGADO"), (int(1080*X_R), int(60*Y_R)),
cv2.FONT_HERSHEY_SIMPLEX, 1*X_R , (80,80,255), round(3*X_R), cv2.LINE_AA)
```

```
                if CONTADOR_SI == 0 :
                    cv2.rectangle(mascara, (int(350*X_R), int(430*Y_R)), (int(950*X_R), int(530*Y_R)), (255, 255, 255), 4)
                    cv2.putText(mascara, str("MODO DE ESPERA"), (int(380*X_R), int(500*Y_R)), cv2.FONT_HERSHEY_SIMPLEX, round(2*X_R)   , (176, 228, 239), round(3*X_R) , cv2.LINE_AA)
                if CONTADOR_SI > 0 :
                    cv2.rectangle(mascara, (350,430), (int(950*X_R), int(530*Y_R)), (255, 255, 255), 4)
                    cv2.putText(mascara, str("CONDICIONAL SI : " + str(CONTADOR_SI)), (int(380*X_R), int(500*Y_R)), cv2.FONT_HERSHEY_SIMPLEX, round(1.5*X_R) , (231, 240, 174), round(3*X_R), cv2.LINE_AA)


            elif ENCENDIDO == 1 :
                cv2.circle(mascara, (int(1050*X_R), int(50*Y_R)), 12, (40, 255, 40), -1)
                cv2.putText(mascara, str("ENCENDIDO"), (int(1080*X_R), int(60*Y_R)), cv2.FONT_HERSHEY_SIMPLEX, 1*X_R, (80,255,80), round(3*X_R), cv2.LINE_AA)
                ## NO ZONA
                SimpHSV = cv2.cvtColor(ANDO, cv2.COLOR_BGR2HSV)

                SimpColorRosado1 = cv2.inRange(SimpHSV, RosadoB1 , RosadoA1)
                SimpColorRosado2 = cv2.inRange(SimpHSV, RosadoB2 , RosadoA2)
                SimpColorRosado2 =  cv2.add(SimpColorRosado1, SimpColorRosado2)

                AND3 = cv2.bitwise_and(SimpColorRosado2, AND2)
                AND3  = cv2.dilate(AND3, np.ones((int(9*X_R) ,int(9*Y_R)),np.uint8), iterations=1)
                cntsAND  = cv2.findContours(AND3,cv2.RETR_LIST,cv2.CHAIN_APPROX_SIMPLE)[-2:]

                for cntx in cntsAND[0]:
                    P = cntx[0]
                    x,y,w2,h2 = cv2.boundingRect(cntx)
                    cv2.rectangle(frame,(x,y),(x+w2,y+h2),(0,0,255),3)
                """
                if  w2 < int(75*X_R) or  h2 < int(36*Y_R) :
```

```
                    ESTADO.append(10)
                """
                if h > int(30*Y_R) :
                    ESTADO.append(100)
                else :
                    ESTADO.append(1000)
                cv2.rectangle(mascara, (int(350 * X_R),int(430*Y_R)) , (int(950 *
X_R),int(530*Y_R)), (255, 255, 255), 5)
                cv2.putText(mascara, str(TXT), (int(360 * X_R),int(500*Y_R)),
cv2.FONT_HERSHEY_SIMPLEX, round(2*X_R) , COLOR, round(3*X_R) , cv2.LINE_AA)
                #cv2.imshow("AND3", AND3)

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

            video_image  = cv2.resize(frame, (560, 315))
            video_image2  = cv2.resize(mascara, (560, 315))

            video_surf = pygame.image.frombuffer(video_image.tobytes(),
video_image.shape[1::-1], "BGR")
            video_surf2 = pygame.image.frombuffer(video_image2.tobytes(),
video_image.shape[1::-1], "BGR")
            if frame_counter2 == ( int(cap0.get(cv2.CAP_PROP_FRAME_COUNT)) or
int(cap01.get(cv2.CAP_PROP_FRAME_COUNT))     ):
                frame_counter2 = 0
                cap0.set(cv2.CAP_PROP_POS_FRAMES, 0)
        else:
            run = False
        screen.blit(video_surf, (40, 237))
        screen.blit(video_surf2, (40+560, 237))
    #RESET
    """
    ##VER
    screen.blit(FONDO, [THE_X, 0])
    """
    ##DIBUJAR
```

```
        all_sprite_list.draw(screen)
        ##FINAL
        pygame.display.flip()
pygame.quit()
```