Acta de Correcciones al Proyecto de Grado
Ingeniería Electrónica 362-2021

Fecha:

Autores: Juan David Holguín Díaz

Nombre del Proyecto de Grado: Algoritmo de fusión de señales de audio y vídeo para el manejo de un robot UR3

Director: Msc. José Hernando Mosquera de la Cruz
Msc. Juan David Contreras Pérez

Como indica el artículo 2.27 de las Directrices de Trabajo de Grado, he verificado que los estudiantes indicados arriba han implementado todas las correcciones que los Jurados del Proyecto de Grado definieron que se efectuaran, como consta en el Acta de Calificación correspondiente.

Firma de Director(a) del Proyecto de Grado

Firma de Codirector(a) del Proyecto de Grado

Pontificia Universidad JAVERIANA Cali

Facultad de Ingeniería y Ciencias

Nota de Aceptación

Aprobado por el Comité de Trabajo de Grado en cumplimiento de los requisitos exigidos por la Pontificia Universidad Javeriana para optar el título de Ingeniero de Sistemas y computación.

Dr. Hernán Camilo Rocha
Decano de la Facultad de Ingeniería

Dr. Luis Eduardo Tobón Llano
Director Carrera Ingeniería Electronica.

Msc. José Hernando Mosquera
Director(a) Trabajo de grado

Msc Juan David Contreras Pérez
Codirector(a) Trabajo de grado

Dr. Alexander Martínez Álvarez
Jurado 1

Dr. Jorge Finke
Jurado 2

Santiago de Cali, junio 4 de 2021

Señores
Pontifica Universidad Javeriana Cali
Dr. Luis Eduardo Tobón
Director Carrera de Ingeniería Electrónica

Cordial Saludo.

Me permito informar que he revisado el trabajo de grado "Algoritmo de fusión de señales de audio y vídeo para el manejo de un UR3" presentado por el estudiando Juan David Holguín Díaz, para optar por el título de Ingeniero Electrónico, encontrando que el estudiante reúne los requisitos que le permiten continuar con el trámite de evaluación y sustentación de su trabajo de grado.

Agradezco de antemano su atención.

Atentamente,

M. Sc. Hernando Mosquera de la Cruz
Director de trabajo de grado
Pontifica Universidad Javeriana Cali

M. Sc. Juan David Contreras Pérez
Codirector de trabajo de grado
Pontifica Universidad Javeriana Cali

Santiago de Cali, junio 4 de 2021

Señores
Pontifica Universidad Javeriana Cali
Dr. Luis Eduardo Tobón
Director Carrera de Ingeniería Electrónica

Cordial Saludo.

Me permito presentar a consideración el trabajo de grado “Algoritmo de fusión de señales de audio y vídeo para el manejo de un UR3” con el fin de cumplir con los requisitos exigidos por la universidad para optar por el título de Ingeniero Electrónico.

Agradezco de antemano la atención prestada.

Atentamente,

Juan David Holguín Díaz
c.c. 1144102354
código: 8919472

Algoritmo de fusión de señales de audio y vídeo para el manejo de un Robot UR3

Autor: Juan David Holguín Díaz

Director: M. Sc. José Hernando Mosquera de la Cruz

Codirector: M. Sc. Juan David Contreras Pérez

Pontificia Universidad Javeriana Cali
Facultad de Ingeniería y Ciencias
Ingeniería Electrónica
Proyecto de grado

Dedicatoria

Dedico este trabajo de grado a las personas que estuvieron apoyándome en mi desarrollo personal y profesional durante estos 5 años que han transcurrido de camino.

Agradecimientos

Agradecimiento principal a mis amigos cercanos que estuvieron apoyándome en las diferentes ocasiones que quise abandonar el proyecto, a mis familiares por la paciencia y entendimiento que han brindado en el proceso. A todos los que han puesto un grano de arena en mí se les dedica estos agradecimientos.

Contenido

Contenido figuras

Contenido tablas

Resumen

En el presente trabajo se realizó una revisión bibliográfica que permitió la identificación de conceptos importantes para la definición, diseño e implementación del algoritmo de fusión de señales y módulos adicionales. Donde se buscaba el cumplimiento del objetivo general que era la implementación de un algoritmo de fusión de señales de audio y vídeo que permitiera el manejo del UR3. Esto con el propósito de dotar al UR3 de una nueva forma de manejo semejante a la robots de carácter más domestico donde la interacción con estas máquinas no requiere poseer conocimiento previos, escenario que si se presenta con el robot de carácter industrial UR3. El desarrollo e implementación del sistema con el UR3 permitió el desarrollo de pruebas que consistía en el desplazamiento de cubos de una posición inicial a una posición final por medio de los comandos audiovisuales. Se realizaron las mencionadas pruebas con usuarios con pocos conocimientos afines a la robótica, los resultados obtenidos mostraron una mejoría en variables de número de instrucciones usadas en las pruebas en comparación con actividades donde no se hacía uso de la totalidad de los comandos.

Abstract

In the present work we made a bibliographic review was carried out that included the identification of important concepts for the definition, design and implementation of the signal fusion algorithm and additional modules. Where the fulfillment of the general objective was sought, which was the implementation of an algorithm of fusion of audio and video signals that allowed the handling of the UR3. This with the purpose of providing the UR3 with a new way of handling similar to the more domestic robots where interaction with these machines does not require prior knowledge, a scenario that is presented with the UR3 industrial robot. The development and implementation of the system with the UR3 includes the development of tests that consists of moving cubes from an initial position to a final position by means of audiovisual commands. The tests were carried out with users with little knowledge related to robotics, the results obtained showed an improvement in variables of the number of instructions used in the tests compared to activities where all the commands were not used.

Introducción

Hace una década aproximadamente se ha comenzado a hablar sobre la cuarta revolución industrial, esta de una forma resumida es el avance tecnológico que está teniendo la industria para ser más productiva con un factor de eficiencia mayor [1]. Uno de los sectores más involucrados en esta revolución es la robótica. Los robots han demostrado ser muy eficientes en la automatización de un proceso industrial, mejorando estándares de calidad, tiempos de producción y reducción de errores. No obstante, la interacción humano robot es un área que se busca desarrollar para tener un panorama más amplio de aplicación de los robots mejorando la calidad de vida del humano y tomando el lugar de los hombres en trabajos repetitivos o peligrosos dentro de la industria [2]. Otro ejemplo de la aplicación de los robots, es ayudando a la experiencia de usuarios dentro de centros comerciales o también, en centros hospitalarios donde se usan robots terapéuticos que ayudan a la recuperación de pacientes con condiciones especiales. Los anteriores ejemplos son algunos escenarios de las aplicaciones que puede mejorar la interacción humano robot.

El proyecto de grado presente tiene como objetivo la implementación de un algoritmo de fusión de señales audiovisuales que permita la interpretación correcta de los comandos gestuales y de voz para la realización de las tareas otorgadas por el usuario a un robot colaborativo UR3. Explicado de otra forma, el algoritmo permite que las tareas dichas por el usuario sean realizadas en la medida que use palabras o gestos que se encuentren dentro de las conocidas por el UR3. Esto con el propósito de mejorar la interacción humano robot que existe hasta ahora, la cual es limitada, ya que existe una dependencia del personal con capacidades avanzadas de programación de robots [3]. A pesar que el desarrollo de algunos robots domésticos permite que los usuarios pueden mantener una conversación básica donde se responden ciertas preguntas, esto no es un caso que se repita en la industria. Por ejemplo, el UR3 es un robot que es diseñado para realizar tareas de gran precisión y de gran repetibilidad, escenarios que se presentan en la industria donde las conversaciones o interacciones que se puedan presentar van enfocadas al cumplimiento de actividades o tareas. Por otra parte, las personas que tienen una interacción con esta máquina actualmente se ve limitada a los operarios que realicen tareas de monitoreo o calidad, y los trabajadores con capacidades de programación que puedan implementar un código para cierta tarea específica que quieran que realice el robot.

Objetivos

0.1 Objetivo general

Implementar un algoritmo de fusión de señales de audio y vídeo para el control de un brazo robótico UR3.

0.2 Objetivos específicos

- Realizar una revisión bibliográfica de antecedentes sobre algoritmos de fusión de señales de audio y vídeo aplicadas en el control de brazos robóticos.
- Definir un diccionario de comandos audiovisuales que esté asociado a las acciones primitivas del brazo robótico.
- Implementar un algoritmo para interpretar comandos audiovisuales que haga uso de acciones primitivas del brazo robótico.
- Evaluar el desempeño del algoritmo de fusión mediante un protocolo de pruebas permitiendo definir sus alcances y limitaciones.

Capítulo I

Marco Teórico

De forma resumida, las dos áreas principales que sirven como base para comprender y entender los ejes del proyecto de grado son interacción humano robot o HRI por sus siglas en inglés Human Robot Interaction y sistemas multimodales. HRI permite comprender de qué forma se podría concebir una manera correcta para interactuar con el robot y que parámetros podrían influir en la experiencia del usuario. Mientras que, los sistemas multimodales permitieron desglosar las implicaciones que se presentan al trabajar con un sistema que hace uso de dos señales de información diferente, que en este caso son de audio y vídeo.

1.1 Interacción Humano Robot

La interacción humano robot es un campo que ha ido ganando cada vez más peso y espacio dentro del mundo de la investigación [4]. Esto en parte se puede atribuir a la facilidad de acceso a robots de carácter doméstico o comercial que ha puesto retos en los diseños e implementación de sistemas que permitan llevar una experiencia más cómoda para las personas que hacen uso de estos. Antes de comenzar a hablar netamente de HRI (Human Robot Interaction) hay que definir de manera correcta a qué tipo de sistemas e interfaces se está refiriendo. Si es cierto que por medio de la inteligencia artificial se goza con mayor facilidad de experiencias donde se habla con interfaces inteligentes cómo se ha mencionado ya anteriormente el caso de Siri o Cortana donde estos son asistentes de voz [5][6]. Con ambas, puedes conversar, pedir indicaciones, soluciones, preguntas, etc. dentro de ciertas limitaciones. Sin embargo, estas no presentan un cuerpo físico, movimientos y en algunos casos, expresiones "faciales", situación que si se presenta en algunos escenarios de HRI. Por ejemplo, los robots como NAO, Pepper, Atlas o yendo un poco más lejos en el caso del humanoide Sophia si presentan cuerpo físico, movimientos, y expresiones [7][8][9]. Así que, elementos como la altura del robot, sus características físicas, su forma de movimiento, entre otras son elementos que hacen que HRI se convierta en un reto de investigación que tiene como finalidad mejorar esta experiencia teniendo presentes nuevos aspectos que no se consideraban en una interacción humano computadora [4]. Estos factores dependiendo del área en el que esté enfocado el desarrollo del robot o la interacción que se pueda tener con él, entrarán a tener diferentes reglas o consideraciones. Por ejemplo, el enfoque que recibió el robot Pepper fue el de una máquina que estará en constante relación en el día a día con el humano [8]. Aunque, el robot PARO que es un robot terapéutico también se encuentra enfocado a el diario convivir de los humanos [10]. Sin embargo, el contexto y la manera en qué se interactúa con él, juega un papel fundamental en el diseño, por ello se aprecia que en el caso de PARO tiene un diseño muy diferente que el Pepper, esto se puede evidenciar en el diseño que se ve en las figuras 1.1 y 1.2.

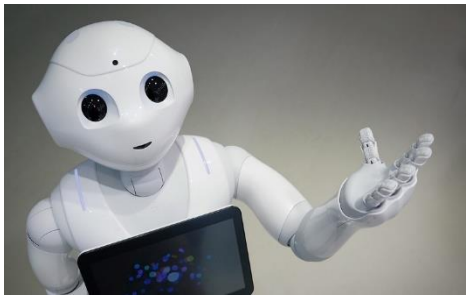


Figura 1.1: Imagen representando al robot Pepper

Figura 1.2: Imagen representando al robot PARO

HRI agrupa diferentes campos de investigación, esto lo hace un tema multidisciplinario donde se ven involucrados conocimientos de ingeniería, también disciplinas como la psicología, sociología y pedagogía, esto debido a la creciente corriente de aprendizaje de las máquinas que ha traído consigo el Machine Learning. También, se comienza a involucrar temas éticos y morales pues la interacción humano robot se ve influenciada por diferentes tipos de factores desde la cultura, el contexto socioeconómico y la etnia. Estas son variables que influyen en el desarrollo de las personas y por lo tanto en su forma de comunicación. El diseño del robot la forma en que capta las señales, su interpretación y la forma en qué responde a los estímulos propiciados por el usuario terminan siendo influenciados por los factores mencionados [4].

Por otra parte, el comenzar en el campo del HRI resulta ser también una experiencia retadora, usualmente el entendimiento de los límites del rango de movimiento y actuación de los robots pueden dificultar y convertirse en una tarea confusa a la hora de realizar respuestas naturales en relación a la interacción que tenga el humano con el robot. Esto se debe que hasta la fecha no se presentan métricas estandarizadas de evaluación cualitativa o cuantitativa de HRI [4]. Por tanto, determinar el éxito de la realización de una tarea o del entendimiento de una orden termina siendo una tarea algo confusa pues no se cuenta con herramientas que permitan determinar el grado de éxito de una interacción, comportamiento o respuesta social adecuada. Sin embargo, en el campo de HCI (Human Computer Interaction) las evaluaciones que usualmente son tomadas en cuenta la eficiencia o la manera en que la computadora logró completar la tarea o ayudar al usuario a alcanzar la meta. Esto existe en HRI donde se busca dar respuesta a preguntas sobre la eficiencia, el control, errores y mejoramiento de la calidad, pero estas métricas solo están enfocadas en medibles de límites de funcionalidad que tienen una dependencia en el contexto que se esté evaluando. A manera de ejemplo, cuando se habla de tareas en el caso del trabajo de grado se refiere a la acción de agarrar un objeto cuadrado que puede ser diferenciado por características visuales.

Si bien es cierto que no se hallaron en la búsqueda métricas estandarizadas que evalúen la interacción humano robot HRI, en el trabajo de Marta Díaz et al, se realiza una propuesta de metodología para la evaluación de la interacción persona-robot en tres diferentes escenarios de aplicación. Los escenarios son marketing, psico afectividad-rehabilitación y vida independiente. En estos tres contextos se realiza una caracterización de la plataforma robótica que está a disposición para la HRI, mirando aspectos como características de la plataforma y modalidades de interacción [11]. Esto servirá más adelante en el sector de análisis para evaluar las opiniones y datos recopilados al usuario. Además, se deberán hacer ciertos ajustes en el tema de contexto y

si existen similitudes con los que ellos plantean. Esto debido a que el ambiente en que se trabaja no es orientado a robots sociales sino por el contrario, a robots industriales, que están diseñados para llevar una relación de maestro esclavo, donde se le dice qué debe realizar y él solo lo realiza.

Entonces, HRI termina en convertirse en una rama de estudio bastante extensa que permite la integración de varios factores de forma profunda y sincronizada que permita el desarrollo de interfaces que cumplan con los requerimientos de la sociedad, cultura y contexto donde se pretende el despliegue del robot para la interacción con humanos. Para el 2010, la afinidad que podría existir entre dos disciplinas como ingeniería y la psicología seguía siendo algo incierta. Actualmente, puede que esta separación se viera reducida por los avances significados del aprendizaje de máquinas también llamado Machine Learning.

Anteriormente se mencionó el Machine Learning como uno de las razones por la cual este campo había comenzado a volverse un poco más extenso. Machine Learning de la mano con la IA (Inteligencia Artificial) ha permitido con las actividades de control y procesamiento de señales sea más sencilla. IA con sus diferentes algoritmos de regresión, clasificación y predicción permiten que la capacidad de entendimiento que puede obtener la interfaz para el manejo de un robot sea mayor. La IA es una clasificación amplia de la informática que permite que un sistema de software perciba su entorno y tome medidas que maximicen sus posibilidades de lograr sus objetivos con éxito. Un objetivo de la IA es crear un sistema de software que sea capaz de adaptarse o aprender algo por sí solo sin estar programado explícitamente para hacerlo [12].

Hay dos enfoques básicos para la IA. El primero es emplear un sistema de aprendizaje profundo que se basa en la red neuronal de la mente humana, lo que le permite descubrir, aprender y crecer a través de la experiencia. El segundo enfoque es el aprendizaje automático, una técnica de ciencia de datos que utiliza datos existentes para entrenar un modelo, probarlo y luego aplicar el modelo a nuevos datos para pronosticar comportamientos, resultados y tendencias futuras. Los pronósticos o predicciones del aprendizaje automático pueden hacer que las aplicaciones y los dispositivos sean más inteligentes. Por ejemplo, cuando compra en línea, el aprendizaje automático impulsa los sistemas de recomendación de productos que ofrecen productos adicionales en función de lo que ha comprado y las elecciones de compras de artículos similares en el pasado. Prácticamente todos los dispositivos o sistemas de software que recopilan datos textuales, visuales y de audio podrían alimentar un modelo de aprendizaje automático que haga que ese dispositivo o sistema de software sea más inteligente sobre cómo funciona en el futuro [12].

Las personas tienden a tratar de comunicarse con robots o interfaces de dialogo de forma similar a la que se comunicarían con un objeto vivo [4]. Esto terminaría por determinar rasgos en la forma en que se comunican con el robot, y la estandarización de estos puede terminar siendo una tarea difícil pues como se mencionaba anteriormente, la comunicación se ve influenciada por aspectos sociales, culturales y étnicos [4][11]. También, la interacción entre humanos es una actividad que se ve inmersa en estos aspectos mencionados que no pueden ser completamente definidos en limites o estándares específicos. Por lo tanto, la cuantificación de estas variables que tienen una fuerte influencia en la interacción tanto humana como robot debe ser tratada con cuidado para definir límites que puedan cumplir por lo menos con las condiciones básicas de una interacción aceptable.

Por último, uno de los puntos importantes a mencionar, es la interacción humano robot se ve fuertemente influenciada por el contexto en el que sucede. Por ejemplo, la HRI que puede presentarse en un centro comercial es diferente a una que se presente en un ambiente industrial. El primer ejemplo planteado, el robot podría tener un enfoque de supervisión e informar a los usuarios sobre sus preguntas frecuentes de lugares, productos, servicios, etc. Por otra parte, el segundo ejemplo, podría obedecer a una interacción de aprendizaje de una tarea específica, supervisión de un proceso o asistencia a los operadores de producción. Regresando al primer ejemplo, el robot desplegado en un centro comercial, el diseño será fuertemente influenciado al servicio y a la respuesta activa de las demandas que posean los usuarios, en este escenario, el desarrollo de interfaces gráficas que permita la muestra de emociones o gestos faciales podría ser un punto en el diseño con el fin de producir un efecto visual más agradable para los usuarios [11][13]. Por el contrario, en el segundo ejemplo, el diseño estaría enfocado más al seguimiento de ordenes con fuertes desarrollos en la parte de control, buscando una eficiencia y precisión en el cumplimiento de tareas. Con esto, se presenta que HRI está fuertemente influenciado por diferentes aspectos que entran a representar un papel importante en las etapas de concepción, diseño, implementación y evaluación de las interfaces que se desarrollen alrededor de un robot.

En el trabajo de grado, HRI está enfocado al cumplimiento de tareas demandas por el usuario. Podría asemejarse un poco a la relación maestro esclavo, donde el usuario demanda una tarea y este en colaboración con el UR3 la lleva a cabo. Esto influye en el desarrollo de control y en la captación de las órdenes. También, en la forma en que recibirá las instrucciones para realizar las tareas propuestas, después de todo las expresiones usadas en un ambiente industrial tienden a tener un lenguaje poco cotidiano a lo que se usaría en otros ambientes de la vida diaria. Aspectos como el diseño del robot, y forma de comunicación entre la interfaz y él no entrarían a considerarse, pues estas ya están diseñadas previamente por el proveedor quién da a disposición una serie de herramientas para la comunicación con la máquina. Por ello, los aspectos que componen HRI no entraría en su totalidad sino los mencionados previamente.

1.2 Sistemas Multimodales

Este trabajo es de carácter experimental y de desarrollo de software, teniendo en cuenta que lo que se busca es implementar un algoritmo de fusión de señales audiovisuales en el robot colaborativo UR3. La implementación de un algoritmo de fusión de señales audiovisuales para la interacción humano-robot requiere el manejo de dos tipos de señales análogas. Es necesario la captura de ambas señales de forma independiente para procesarlas dentro de un computador y, así poder extraer la información requerida de ellas. Luego de tener los datos de ambas señales, se debe realizar la correcta interpretación y uso de estos para generar señales de control adecuadas para el robot. Teniendo esto claro, se hablará de forma separada de ambas señales y lo que concierne la recolección y procesamiento de cada una.

1.2.1 Captación y procesamiento de señales de audio

Las señales de audio son el canal de comunicación primario para comunicarnos con otro individuo de manera presencial. Estas señales sonoras son variaciones de ondas en el aire, de otra forma dicha, son señales análogas. Para que una computadora pueda trabajar con una señal de audio esta debe ser digital así que, la digitalización de esta señal es necesaria y obligatoria. Al tener esta señal de forma digital se puede realizar un análisis en profundidad con las herramientas que provee hoy

en día la computación. Uno de los resultados de este análisis o mejor llamado procesamiento de audio termina en lo que se conoce como reconocimiento de voz o del habla, este es el campo de interés del trabajo de grado. El reconocimiento del habla o de voz, es un área que ha venido presentando diversos avances y aplicaciones. Por ejemplo, Google home, Siri y Microsoft Cortana que son sistemas de grandes compañías que hacen uso de señales de voz para mejorar la experiencia de sus usuarios [5]. Algunos de los sistemas mencionados anteriormente, se pueden ver implementados en los celulares como es el caso de Siri de Apple. Estos sistemas tienen como finalidad dotar de capacidades técnicas a una máquina que le permita el entendimiento de un lenguaje natural que resulta normal para las personas.

Para lograr esto, se debe realizar cooperación entre diferentes sistemas que permitan traducir del lenguaje natural de las personas a un lenguaje binario que comprendan las máquinas. Esto representa un reto ya que, las señales de audio por lo general al ser recolectadas presentan diferentes niveles de ruido causados por cualquier tipo de señal que fueron adicionadas a la muestra [14]. Por lo general, estas se deben a ruidos del ambiente, voces de interlocutores que no hacen parte de la señal deseada, entre otras. Sin embargo, a pesar de todas las dificultades que pueda presentar se debe llegar a un resultado aceptable que permite el entendimiento del mensaje que se recibió de la señal de audio.

La manipulación de señales de audio requiere un pre procesamiento de la señal para poder ser analizada correctamente en un computador. Al ser una señal analógica se debe primero muestrear a una frecuencia por lo menos el doble de la señal de audio, esto con el fin de respetar el teorema de Nyquist [15], de lo contrario se podría estar perdiendo información por no realizar un correcto muestreo. Posteriormente, se realiza una cuantificación de la señal, se puede realizar por medio de la Ley Mu o Ley A que son dos métodos muy conocidos [16]. A partir de aquí, se puede digitalizar la señal de audio, lo cual permite hacer uso de programas, software y diferentes técnicas de procesamiento de audio para analizar la señal recibida. Por consiguiente, portar una señal buena para lograr separar el contenido de habla del usuario, para esto existen diferentes algoritmos y técnicas que realizan esta tarea como es el caso del Reconocimiento Automático del Habla (RAH) que es una disciplina de la inteligencia artificial (IA) [17], esto por medio de fuentes de conocimiento sobre las características de la voz, lo que permite realizar esta separación. Por otra parte, existen dispositivos que permiten la captación de una señal de sonido bastante limpia, lo cual mejora en gran medida también el procesamiento de estas señales. Por ejemplo, un micrófono Yeti permitiría que la señal fuera limpia en el sonido, esto significa que los componentes de ruido se ven disminuidos, pero siguen existiendo dentro de la señal, ya que presenta diferentes configuraciones donde la captación de la fuente es diseñada para un escenario en específico. A manera de ejemplo, una de las configuraciones es “cardioide” que es el recomendado para grabar podcasts, la fuente de sonido es tomada como la que se encuentra directamente frente del micrófono [18]. Todo esto concierne a la captación de la señal análoga, luego se habla del procesamiento digital del audio o conocido como PDA.

Los dos sentidos humanos principales son la vista y el oído. En consecuencia, parte del Procesamiento Digital de Señales del cual hace parte PDA se encarga precisamente del tratamiento de la información análoga que recogen sensores como micrófonos y cámaras para cada caso respectivo. De ahí, la importancia de entender cómo se toma la señal y se digitaliza para su

respectivo procesamiento. PDS, procesamiento digital de señales, tiene un amplio desarrollo en el campo de audio y vídeo. Enfocados principalmente en PDA que es el procesamiento digital de audio, en este se tocarán dos temas que son el caso de interés del proyecto grado presente. Se comenzará con generación de voz y luego se procede con reconocimiento automático del habla. Por último, se termina con una aplicación de ambos temas tratados que es el sistema de dialogo hablado.

1.2.1.1 Generación de voz

Cortana de Microsoft es un ejemplo del uso de generación de voz. Esta herramienta le provee la capacidad al robot de hablar, de responder por medio de sonidos que resulten naturales para el humano. Existen para el habla generada por computadora: grabación digital y simulación del tracto vocal. En la grabación digital, la voz de un hablante humano se digitaliza y almacena, generalmente en forma comprimida. Durante la reproducción, los datos almacenados se descomprimen y se vuelven a convertir en una señal analógica. Una hora entera de voz grabada requiere sólo unos tres megabytes de almacenamiento, dentro de las capacidades de incluso los sistemas informáticos más pequeños. Este es el método más común de generación de voz digital que se utiliza.

Los simuladores del tracto vocal son más complicados y tratan de imitar los mecanismos físicos mediante los cuales los humanos crean el habla. El tracto vocal humano es una cavidad acústica con frecuencias de resonancia determinadas por el tamaño y la forma de las cámaras. El sonido se origina en el tracto vocal en una de dos formas básicas, llamadas sonidos sonoros y fricativos. Con los sonidos sonoros, la vibración de las cuerdas vocales produce pulsos de aire casi periódicos en las cavidades vocales. En comparación, los sonidos fricativos se originan a partir de la ruidosa turbulencia del aire en las constricciones estrechas, como los dientes y los labios. Los simuladores del tracto vocal funcionan generando señales digitales que se asemejan a estos dos tipos de excitación. Las características de la cámara resonante se simulan pasando la señal de excitación a través de un filtro digital con resonancias similares. Este enfoque se utilizó en una de las primeras historias de éxito de PDS, Speech Speel (El habla y el deletreo), una ayuda electrónica de aprendizaje para niños ampliamente vendida.

1.2.1.2 Reconocimiento automático del habla

El reconocimiento automático del habla humana es por mucho más difícil que la generación de voz. El reconocimiento del habla es un ejemplo clásico de las cosas que el cerebro humano hace bien, pero las computadoras digitales no lo hacen como quisiéramos [19]. Las computadoras digitales pueden almacenar y recuperar grandes cantidades de datos, realizar cálculos matemáticos a velocidades vertiginosas y realizar tareas repetitivas sin aburrirse o ser ineficaces. Desafortunadamente, las computadoras hoy en día presentan defectos cuando se enfrentan a datos sensoriales sin procesar, o mejor dicho deben aprender a hacerlo y nosotros debemos programarlas para que lo puedan realizar.

El PDS generalmente aborda el problema del reconocimiento del habla en dos pasos: extracción de características seguida de coincidencia de características. Cada palabra de la señal de audio entra se aísla y luego se analiza para identificar el tipo de excitación y las frecuencias de resonancia. Luego, estos parámetros se comparan con ejemplos anteriores de palabras habladas para identificar la coincidencia más cercana. A menudo, estos sistemas se limitan a unos pocos cientos de palabras, sólo puede aceptar el habla con pausas distintas entre palabras, y debe ser reentrenado para cada

orador individual. Si bien esto es adecuado para muchas aplicaciones comerciales, estas limitaciones son abrumadoras en comparación con las capacidades del oído humano.

A menudo se habla de las diferencias entre el reconocimiento del habla y el reconocimiento de la voz, estas diferencias pueden ser algo arbitrarias. Pero, básicamente la función del reconocimiento de la voz es identificar y tipificar la voz del hablante, en cambio el reconocimiento del habla distingue toda la estructura oral del usuario, palabras. Por tanto, el reconocimiento de voz permite funciones de seguridad como la biometría de la voz y el reconocimiento del habla permite transcripciones automáticas y comandos precisos. Pensando en esta cuestión se habla de reconocimiento del habla ya que la interfaz no discriminará la voz de la persona sino solo los comandos de voz asociados a la interfaz.

1.2.1.3 Sistema de diálogo hablado (Spoken Dialogue System SDS)

Un sistema de lenguaje hablado combina el reconocimiento de voz, el procesamiento del lenguaje natural y la tecnología de interfaz humana, como el entendimiento de un dialogo [20]. Su función es reconocer las palabras que la persona dice, interpretando la secuencia de palabras para conseguir un significado en términos de la aplicación y proporcionar una respuesta aceptable para el usuario. Las aplicaciones potenciales de los sistemas de lenguaje hablado van desde tareas simples, como recuperar información de una base de datos existente (informes de tráfico, horarios de las aerolíneas), hasta problemas interactivos para resolver tareas que implican una planificación y un razonamiento complejo (planificación de viajes, enrutamiento de tráfico).

Ha existido un interés renovado en el procesamiento interactivo del lenguaje natural hablado, que ya ha demostrado su utilidad para mejorar las interacciones entre humanos y computadoras [21]. Para procesar el lenguaje hablado a partir de interacciones entre humanos o entre humanos y computadoras, debemos tener en cuenta el "ruido interactivo" que implica el lenguaje hablado espontáneo. Ruido interactivo se refiere a las interjecciones, pausas, repeticiones correcciones de errores, comienzos en falso, construcciones sintácticas o semánticas poco comunes, etc. Que ocurren en el lenguaje hablado espontáneo, pero no en el lenguaje escrito.

Los reconocedores del habla, que utilizamos para identificar las palabras habladas para el procesamiento del lenguaje hablado, no pueden funcionar de manera óptima y, a menudo, crea un tipo adicional de ruido. Por lo tanto, pueden hacer hipótesis de palabras incorrectas y producir oraciones gramaticales sin sentido. Para continuar procesando el lenguaje hablado a pesar de estas limitaciones, el análisis del lenguaje debe ser robusto y tolerante a las fallas [21] [22].

Existen ocho áreas clave en las que se necesita investigación básica para producir sistemas de lenguaje hablado: 1) reconocimiento del habla, 2) entrenamiento y adaptación automáticos, 3) habla espontánea, 4) modelos de diálogo, 5) generación de respuestas en lenguaje natural, 6) síntesis de voz y generación de voz, 7) sistemas plurilingües y 8) sistemas multimodales interactivos [23]. Para la creación de la interfaz de voz se tendrán en cuenta áreas como la 1, 4, 5 y 6 que son los mínimos requerimientos para entregar una interacción humano robot aceptable.

El objetivo del reconocimiento del habla es adquirir la secuencia de palabras pronunciadas por un hablante [20] [24] [25]. Es una tarea compleja, puede haber una gran cantidad de variación en la entrada que el módulo encargado al reconocimiento debe analizar. Por ejemplo, en términos de la lingüística del enunciado, el contexto de interacción y el canal de transmisión. Una vez que el

reconocedor de voz ha proporcionado una salida, el sistema debe comprender lo que dijo el usuario. El objetivo de la comprensión del lenguaje hablado es obtener la semántica de la oración reconocida. Este proceso generalmente requiere morfológico, léxico, sintáctico, semántico, discurso y conocimiento pragmático [26] [27].

1.2.2 Captación y procesamiento de señales de vídeo

Se habla sobre la captación y procesamiento de señales o señal de vídeo. De entrada, ya se está afirmando y entendiendo que es esa señal la que se piensa usar. Sin embargo, la señal de vídeo tiene diferentes implicaciones que es necesario aclarar para conocer qué información y bajo qué condiciones se está trabajando. Esto facilita el ejercicio de elegir un correcto dispositivo para la captación de la señal y procesamiento de la misma.

Para comenzar, primero, la señal de vídeo puede ser como su nombre lo dice un vídeo que capte varios segundos de un escenario en especial, o se encuentre en vivo, entregando información todo el tiempo sobre lo que está visualizando la cámara, u otro caso es que solo capte una imagen o una toma del escenario que tiene en frente. Para los tres casos, la información y el volumen que pueda entregar va a variar pues en el caso de un vídeo en vivo se mantendría enviando información constantemente, el tamaño de los datos que debe analizar el módulo de procesamiento aumenta. Por otra parte, si es un video con una duración ya definida el tamaño también se encontraría estable, esto también aplica para una imagen, pero entendiendo que es archivo de mucho menor tamaño. Lo segundo, conocer la resolución que entrega el dispositivo que capta la señal. La precisión que pueda entregar el procesamiento de vídeo de una captura 260p a 480p es menor a la de una captura entre 720p o 1080p. Estos números hacen referencia directa a la calidad del vídeo que se está grabando. Tercero, la información que se recibe del dispositivo ¿Es 2D o 3D? Existen dispositivos actualmente como una cámara ZED o un Kinect que permite la entrega de información 3D, que en este caso sería profundidad la variable adicional para ser considerada 3D. Esto determina también el sentido del objetivo que estará persiguiendo el procesamiento de la señal de vídeo, pues esta información debe ser aprovechada totalmente. Para finalizar, el modelo de color que está manejando el vídeo, este puede ser HSV, RGB o CMYK, todos modelos usados en diferentes técnicas de procesamiento de imagen o vídeo. Teniendo claro estos conceptos y aspectos sobre la señal de vídeo se podrá elegir de manera más acertada que procesamiento se le realiza para extraer información.

Tomar información de una señal de vídeo es una tarea que diariamente se realiza en diferentes dispositivos. Por ejemplo, las cámaras de transito llamadas foto multas pueden captar la placa de registro de un vehículo, medir la velocidad a la que iba, ubicación y determinar si el conductor está cometiendo una infracción [28]. Otro ejemplo más común en los jóvenes son los tan usados filtros dentro de las aplicaciones como Instagram, Snapchat o Facebook, que por medio de un reconocimiento facial puede dibujar figuras alrededor del rostro. También, en algunos celulares el reconocimiento facial se usa como medida de seguridad para desbloquear el celular como en el caso del Apple ID del iPhone X de Apple. El procesamiento digital de señales es el que hizo esto posible. Es por medio de ellas que se digitalizó las señales análogas y que permiten ser analizadas y usadas por las computadoras. La visión por computadora es el área específica que se dedica a estudiar nuevas formas de extraer información de las imágenes.

Evaluando los ejemplos mencionados anteriormente, la forma de identificar la información que se busca en una señal de vídeo marca una perspectiva diferente en cada caso. En el caso de la foto multa, se tendrían en cuenta la identificación del automóvil, identificación de la placa de registro, velocidad del automóvil y seguimiento o tracking del auto. Mientras que, en el caso de las aplicaciones de redes sociales de dispositivos se evidencia un reconocimiento facial, correspondencia en el dibujo de efectos especiales sobre el rostro del usuario y un seguimiento o tracking del rostro. Sin embargo, estos dos solo son pequeños ejemplos de lo que puede realizar la visión por computadora, podemos ver casos donde se hable de la clasificación de semillas o de frutas, el reconocimiento y digitalización de textos, revisión de calidad de procesos o reconocimiento de obstáculos para robots móviles [29] [30] [31] [32]. Todos con un sistema de visión por computadora que permite la toma de características de la imagen y entregarla como información organizada para realizar toma de decisiones, clasificación o comprobación de estándares. Si bien, es un campo bastante extenso con aplicaciones de todos los tipos en diferentes sectores. En el trabajo de grado presente se busca reconocer características espaciales y de color de cubos que se encuentran en el espacio de trabajo del UR3.

Al igual que en el caso de las señales de audio, las de vídeo también pueden ser contaminadas por ruido. Este ruido puede verse presentado por un exceso de la fuente de luz o una inexistencia de la misma. También, por superficies reflectivas que reflejen haces de luz frente al lente que capta la imagen de vídeo. Estos son aspectos externos al dispositivo que se deben mantener en control para evitar que la señal se vea afectada en su calidad e integridad. Puesto que, una señal que no capte la información correcta del vídeo con una calidad adecuada podría presentar casos donde el procesamiento de la información termine en errores.

En la primera etapa del procesamiento de las señales de vídeo se menciona la captura de la señal, esto por medio de sensores adecuados para esta tarea cómo una cámara. Sin embargo, existen dispositivos con mejor tecnología que entregan una señal con mejor calidad y con una información más detallada como es el caso del Kinect [33]. La diferencia entre una cámara y el Kinect es el desarrollo tecnológico que existe alrededor de cada dispositivo. Sin entrar a hablar en profundidad de una cámara se entiende como un dispositivo que capta la señal de vídeo y la entrega de forma digital a la computadora. Mientras que, hablar de un dispositivo como Kinect se menciona que la información que provee no solo es la imagen sino una serie de información adicional como la profundidad del cuarto que está visualizando o el reconocimiento del cuerpo de una persona. Elegir un dispositivo correcto para la captura de vídeo se ve influenciado en gran medida por el contexto y la forma en que va a ser usado, también qué tipo de función va a cumplir. Por ejemplo, si la meta que se persigue es la captación de presencia de las manos o gestos propios de una mano tal vez los anteriores dispositivos no sean los más propios para este trabajo, existen desarrollos tecnológicos que tienen integrado hardware y software diseñado para la captación de esta información como es el caso del LEAP motion.

Luego de elegir correctamente el dispositivo que capta la señal de vídeo y manejar bien las variables externas e internas que pueden afectar la señal sigue faltando el núcleo de la visión por computadora y es precisamente el tipo de procesamiento que se le va a realizar. Dentro del mundo de la programación en diversos lenguajes existen módulos o librerías que permiten extraer la información deseada de una señal de este tipo. Hablando de un caso en particular, Python, un

lenguaje de programación de alto nivel que permite la creación de algoritmos y el uso de los mencionados módulos para obtener la información que se requiere de una señal de vídeo [34]. Librerías como Numpy, Scipy, PIL, Scikit-Image, SimpleCV, SimpleITK, Pgmagick u OpenCV son compatibles con este lenguaje de programación. Sin embargo, no todas sirven para el mismo propósito, Numpy es una librería enfocada a los arreglos y al manejo de estos, muy útil a la hora de manejar la información de vídeo dentro de Python. Pgmagick permite transformar el vídeo en diferentes formatos, creación y recorte de bordes alrededor de objetos específicos entre otras cosas que se pueden realizar con esta librería. SimpleCV es un framework de visión por computadora de manera artificial, este permite la extracción de bordes, formas, colores y diferentes análisis y aplicaciones se pueden realizar con ella. La elección del lenguaje de programación también puede variar, C, C++ o C#, incluso en algunos casos Java podrían ser opciones de lenguajes a usar, ya que en varios de estos se pueden hacer uso de las mismas librerías o encontrar nuevas. Por otra parte, la selección de la librería depende sobre la información que quieras extraer y qué procesamiento se piensa realizar.

1.2.3 Sistemas que manejan vídeo y audio como señales de entradas

En la bibliografía consultada de sistemas multimodales se logró presenciar que existen todo tipo de aplicaciones en esta área. Hay trabajos con componentes de manejo de la información multimodal muy robustos como una CNN (Convolutional neural network) o HMM (Hidden Markov models) que son algoritmos probabilísticos que permiten tener una predicción bastante aceptable, basándose en los parámetros de entrada que poseen de los sensores y un dataset que permite el entrenamiento previo a usar los sistemas. Por otra parte, también se evidenciaron propuestas menos avanzadas y con un enfoque diferente para manejar por medio de señales de vídeo y audio un robot móvil. Otros trabajos hablan de aprendizaje, esto como se mencionó anteriormente se debe al aprendizaje de máquinas o también conocido como Machine Learning que permite el aprendizaje de tareas que no tienen programas explícitamente. Además, cabe aclarar que se presentan en algunos trabajos consultados el control del robot por medio de otro tipo de comandos y no netamente de audio y vídeo, este puede ser kinestésico o tele operado. A continuación, se presentan los trabajos que muestran diferentes escenarios donde se hace uso de señales de audio y vídeo.

El trabajo de Jing Guang Han et al [35] propone un sistema recopilación de datos multimodales de interacción humano-robot, se expresa que los recursos son bastante limitados así que enfrentan diferentes retos a la hora de la construcción de este sistema. Se utilizó un robot Lego Mindstorms NXT como plataforma básica para recopilar las grabaciones de audio y visuales. El sistema Mindstorms NXT ofrece una plataforma de adquisición de datos versátil y configurable, ya que los bloques de construcción de Lego se pueden configurar de muchas maneras diferentes para montar cámaras de video y micrófonos. Por otra parte, para la programación del robot y sus algoritmos se usó el lenguaje Python, donde se usa la librería OpenCV para la detección de imágenes. En la parte audio, la interfaz de diálogo se construyó utilizando el entorno de programación visual Max / MSP que permitía una programación fácil y dinámica. Se implementa una máquina de estado finito que permite la simulación de una interacción humo-robot en fusión de cuatro estados.

El trabajo de Snejana Pleshkova et al. [36] propone el desarrollo del modelo de sistema para el control audiovisual de robots móviles con comandos de voz y gestos. En el desarrollo del trabajo proponen un sencillo diagrama de bloques, en este se evidencia la recolección de las señales en una

etapa, la siguiente seguiría la interfase, después un proceso de cómputo y luego el envío de información por medio de WiFi al robot móvil. Se propone un algoritmo en MATLAB que permita el tratamiento de ambas señales siguiendo una secuencia de pasos. Se resalta que en este algoritmo tiene una sección donde se compara los comandos de voz con la entrada y los comandos de vídeo con la entrada, luego, a partir de esto se toman las decisiones respectivas. En comparación al trabajo de grado presente, el algoritmo de fusión de señales audiovisuales propuesto en este documento es bastante útil ya que está implementado sobre un software muy conocido.

El trabajo de Juan D. S. Ortega et al. [37] propone el reconocimiento de emociones usando la fusión de funciones de audio y vídeo. Este tiene un enfoque basado en el aprendizaje de transferencia y la fusión multimodal para el problema del reconocimiento continuo de emociones en vídeo, esto con el fin de usar un conjunto de datos FER que les proporciona imágenes faciales con contenido emocional para entrenar su CNN (Convolutional Neural Network), esto les permite utilizar una CNN pre entrenada en vez de entrenar una desde cero. El conjunto de datos de destino (RECOLA) se utiliza para ajustar algunos parámetros (capas) de dicha CNN. Se usan técnicas de pre procesamiento como supresión de cuadros, selección de calidad de fotogramas y retrasar la compensación para realinear etiquetas. Para la fusión multimodal, utilizaron el conjunto extendido de parámetros acústicos minimalistas de Ginebra (eGeMAPS), utilizando en la línea de base de RECOLA. En comparación con el trabajo de grado a realizar se puede evidenciar que existen similitudes en algunos tipos de señales como lo son las de vídeo y audio, sin embargo, estos tienen parámetros adicionales a su sistema que les agrega mayor robustez. Este deja información esencial en el manejo y arquitectura de una CNN que podría ser de gran utilidad en la solución de la problemática planteada.

El trabajo de Amir Aly et al. [38] propone un modelo integrado de mapeo de gestos del habla al brazo en la interacción humano-robot. Para esto, las señales de audio o de voz tienen que pasar por tres procesos, que son un acondicionamiento de la señal, estimación de periodos candidatos y procesamiento posterior de la señal. El proceso de acondicionamiento de señal intenta eliminar los componentes de la señal interferente, como cualquier ruido extraño, mediante el uso de un filtro de paso bajo que elimina la pérdida aparente de periodicidad en el espectro de la señal sonora a frecuencias más altas, y mediante el filtrado de paso alto cuando hay componentes de CC o de muy baja frecuencia. Luego, se usa el cálculo de correlación cruzada normalizada (NCC) de dos pasadas para buscar la frecuencia fundamental. Después, se utiliza un filtrado para refinar la frecuencia fundamental. Por último, el mapeo entre el habla y los gestos del brazo se realiza mediante el uso de modelos de Markov ocultos acoplados (CHMM), que podrían verse como una colección de HMM para las transmisiones de audio y video. En relación al trabajo de grado que se presenta la similitud del uso de señales audiovisuales. Sin embargo, no son usadas con el mismo propósito que el que se plante en el proyecto de grado.

Capítulo II

Metodología de Implementación

En este segundo capítulo del proyecto de grado se presenta la metodología de implementación que se siguió para lograr completar los objetivos propuestos. Se comienza con la identificación de algoritmos o de técnicas de fusión que permitan manejar señales de vídeo y de audio para realizar un control adecuado de un robot. Dentro de este apartado se realiza la definición de criterios de selección del algoritmo que se piensa implementar, la selección del algoritmo y una explicación de este. Luego, se continúa con la definición de diferentes aspectos que deben estar enlistados para comprender el contexto en el que se está trabajando. Aquí se habla sobre las capacidades físicas del UR3, dispositivos que se piensan usar, que están integrados al robot y adecuaciones que se deberían realizar al espacio de trabajo. Parte importante de este apartado es la definición del ambiente de trabajo del UR3 y los comandos audiovisuales que están disponibles para manejar el robot. Una vez terminada la definición del contexto, se procede con la implementación de lo explicado anteriormente. Se habla sobre el módulo de vídeo y audio, revisión de comandos y coherencia de estos, la implementación del algoritmo, el control y comunicación con el UR3. Por último, pero no menos importante, dentro de la metodología de implementación se toca la evaluación del resultado final de la interfaz. En este apartado, se hace uso de encuestas cuantitativas y cualitativas para determinar aspectos técnicos y de funcionamiento del sistema.

2.1 Identificación

En este apartado se definen los criterios de selección del algoritmo que se desea implementar. Dentro de este, se explica que entradas y salidas debería tener este, entradas que son resultantes de los módulos encargados del procesamiento de audio y vídeo. Se presenta la selección del algoritmo que se desea implementar y se procede a la explicación y muestra de diferentes figuras que mejoran el entendimiento de lo que se desea implementar. Sin más explicación se comienza con la definición de los criterios de selección.

2.1.1 Definición de criterios de selección

Antes de comenzar a definir qué tipos de criterios deben tenerse en cuenta para elegir o construir un correcto "algoritmo" es pertinente definir y explicar qué es. Una definición bastante general que se encuentra hoy en internet afirma que es conjunto ordenado de operaciones sistemáticas que permite calcular o hallar la solución a un problema. También, en matemáticas, ciencias de la computación o alguna rama relacionada con estas, entienden que un algoritmo es un conjunto de instrucciones o reglas definidas, ordenadas y finitas que permite solucionar, realizar un cómputo, procesar datos y llevar a cabo otras tareas o actividades.

De forma resumida, como lo menciona Robin K. Hill en su trabajo "What an Algorithm Is" el algoritmo es lo que implementan los programas, lo que hace el procesamiento de datos y otros cálculos [39]. Por lo tanto, antes de escribir un código en algún lenguaje de programación existente se debe entender muy bien cuál problema se busca solucionar con el algoritmo que se va a implementar. Para este trabajo de grado el algoritmo debe entregar el conjunto de instrucciones necesarias para completar la actividad que se le solicita al robot por medio de comandos

audiovisuales. El conjunto de instrucciones son un conjunto de acciones primitivas que el UR3 va a replicar, al realizar en el orden estipulado que determine el algoritmo que es el más adecuado para cumplir la tarea solicitada. Hasta el momento se conoce cómo será la salida, ahora es necesario hablar sobre qué datos le llegaran al algoritmo para poder determinar el conjunto de posiciones angulares que son las responsables del movimiento y posición del robot.

En la figura 2.1 se aprecia un diagrama de bloques que muestra la estructura utilizada en diversos trabajos para la función de señales de audio y vídeo. Y es que, resulta muy apropiado tratar las señales de forma individual ya que se extraen características diferentes que permiten obtener más información acerca de la situación que se está observando. Para el caso específico del trabajo de grado, se busca principalmente el reconocimiento de comandos de audio y vídeo, también el reconocimiento de características de los objetos y del espacio de trabajo que dispone el robot. El reconocimiento de características de objetos que son invariantes en el tiempo resulta fácil después de todo no se cambiará el color, la forma u otras características de este tipo. Sin embargo, el reconocimiento de los comandos de audio y vídeo si resulta ser una tarea que representa un grado de dificultad mayor porque se debe reconocer la actividad humana y luego rasgos específicos de esta en un vídeo y del audio.

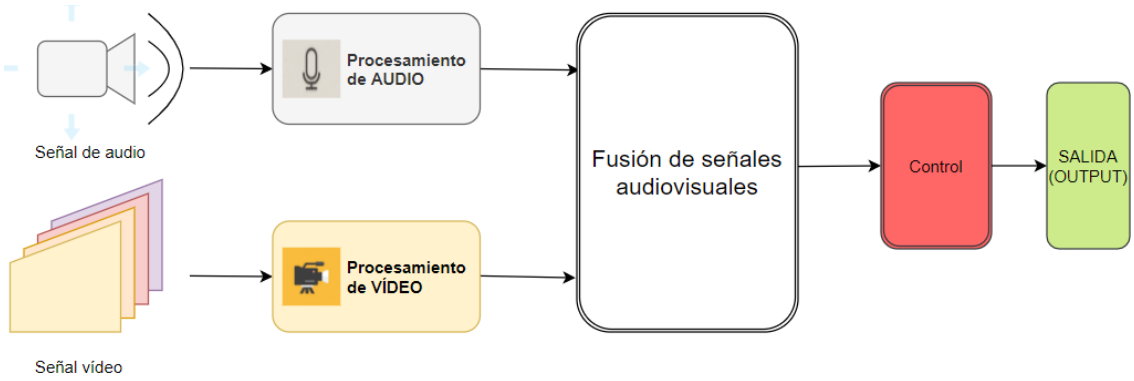


Figura 2.1: Diagrama de bloques del sistema multimodal.

El bloque del procesamiento de audio entregará un conjunto de datos que estarán ordenados en un vector que contendrá los comandos de audio proporcionados por el usuario. Esto quiere decir que, teniendo en cuenta que las pruebas consistirán en mover cubos de un lugar a otro, si el usuario luego de haber acercado al robot hasta los cubos con comandos de audio dice “Quiero que agarres el círculo rojo” la salida de la caja encargada del procesamiento de audio entregará en un vector los datos. Estos datos son los comandos de audio que son a larga verbos, sustantivos y adjetivos para dar suficiente información sobre lo finalidad de la tarea. Por otra parte, la caja del procesamiento de vídeo también entrega un vector ordenado con datos y algunos de los datos son coordenadas. Ver figura 2.2.

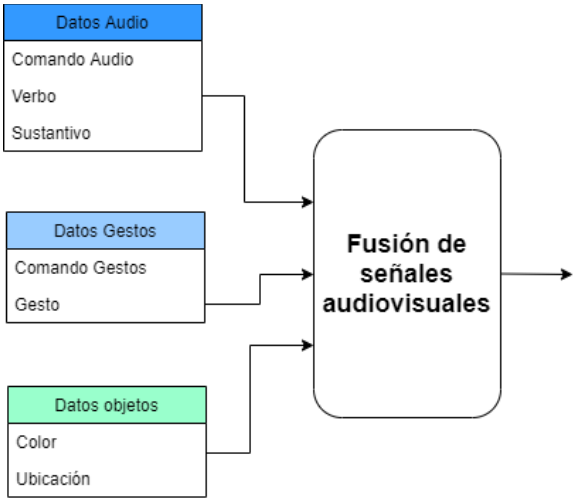


Figura 2.2: Señales de entrada al bloque de fusión señales audiovisuales.

Para la parte de control se hace uso de librerías dedicadas y pensadas en especial para este tipo de aplicaciones. Este es el caso de URScripts API que es una guía dedicada al control el UR3. Ya que, ese es el lenguaje que el maneja a nivel operativo, por lo tanto, se hace necesario una traducción de este. Cuando se hace el control de un robot a menudo es común conocer la posición final de las articulaciones, pero no el valor exacto ángulos o desplazamientos en comparación con su posición anterior. Por ello, se habla de cinemática directa e inversa cuando se trata de conocer los valores de los ángulos de las articulaciones del robot. Explicado de forma general si conoces el ángulo o el desplazamiento aplicas cinemática directa y encuentras la posición final. Mientras que, si tienes la posición final que con eso se refiere a los ángulos de cada articulación aplicando cinemática inversa se obtiene el desplazamiento que se debe realizar para llegar a dicha posición. Sin embargo, estos no son procedimientos sencillos y por ello existen librerías dedicadas a esto. Para entender esto mucho mejor ver figura 2.3.

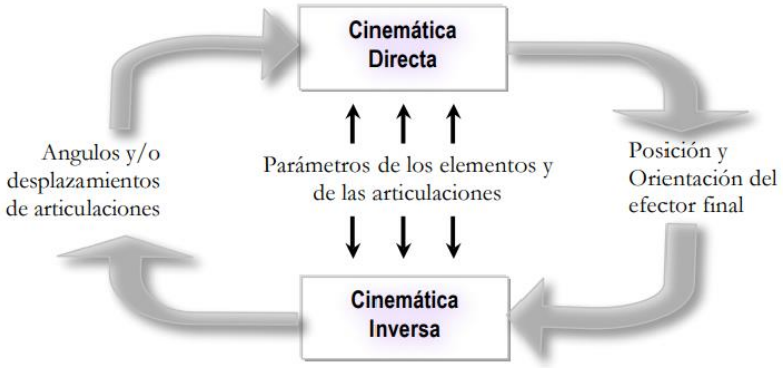


Figura 2.3: Esquema de Cinemática directa e inversa.

Los diccionarios de comandos de audio y vídeo son otro factor que se debe tener en cuenta en los criterios de selección del algoritmo. Estos son los que permiten saber si la forma en que el usuario le instruyó al UR3 está dentro de las permitidas. También, sucede para el caso del vídeo, los gestos

que realice con sus manos deben estar dentro de las permitidas de lo contrario, el robot no atendería a las instrucciones dadas. Por otra parte, los límites del área de trabajo son condiciones que deben cumplirse en todo momento, esto con el propósito que el robot no se salga en ninguna ocasión de su área permitida, en la figura 2.4 se puede apreciar esta área como el cuadrado negro donde se encontrarán ubicados los objetos que se puedan desplazar.

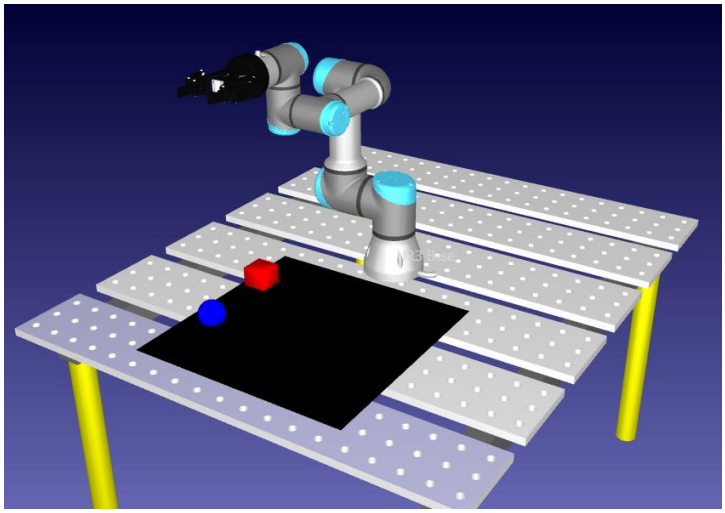


Figura 2.4: Simulación del área de trabajo del UR3.

En conclusión, el algoritmo tiene como entradas vectores con información recopilada de las señales de audio y vídeo, comparará esta información con los diccionarios que el posee de forma predeterminada para validar si es una de las instrucciones permitidas. A partir de la información obtenida tendrá que generar señales de control adecuadas que no lleven al robot a salirse de su área de trabajo respetando esta y más limitaciones que surjan en la práctica. Es necesario apreciar que existen un alto componente de tomas de decisiones que debe realizar el algoritmo, esto acota grandemente qué tipo de algoritmo se debe elegir. Por ejemplo, existen algoritmos de regresión, agrupación, bayesianos y árboles de decisión, por nombrar algunos. Si bien todos son algoritmos cumplen tareas diferentes, unos son predictivos a partir de datos suministrados, otros tienen como finalidad agrupar los datos por características y otros toman decisiones a partir de los datos suministrados y son estos últimos los llegan a cumplir los criterios.

2.1.2 Selección del algoritmo

Actualmente, la interacción humano maquina se ha convertido en una actividad que se aprecia con mayor normalidad en el diario vivir. Sistemas dedicados a recibir señales de tipo auditivas y vídeo se tienen hoy al alcance de la mano. Por ejemplo, el asistente de voz de los celulares como son el caso de Siri, también otros casos como Cortana de Microsoft que tienen un enfoque netamente de audio o se puede mirar las famosas aplicaciones como Instagram que tiene la opción de filtros en la cámara que permite a los usuarios agregar diferentes detalles a sus fotografías o vídeos. Este tipo de aplicaciones son el resultado del desarrollo de décadas descubriendo la forma correcta del tratamiento de estas señales. Comenzando por el teorema del muestreo de Nyquist que argumenta que existe una frecuencia mínima a la cual se debe muestrear la señal para no perder información

de ella, hasta las redes neuronales convolucionales por sus siglas en inglés (CNN) que permiten la extracción de algunas de las características de estas señales. Cabe resaltar, que estas solo son dos de las muchas herramientas que se usan para el tratamiento de señales audiovisuales.

La extracción de las características de las señales audiovisuales presenta un gran reto debido a la gran componente de ruido que pueden presentar. Por ejemplo, si un usuario se encuentra en medio del centro de Santiago de Cali será mucho más difícil extraer las características de su voz que si se encuentra en un ambiente con condiciones controladas si a ruido se refiere. Lo mismo sucede para el caso del vídeo, las condiciones de iluminación y velocidad de los frames son puntos muy influyentes en la calidad de características que se pueden extraer de una señal de este tipo. Esto se aprecia bien en el trabajo de Karel Palecek et al. titulado "Audio-Visual Speech Recognition in Noisy Audio Environments" donde la meta principal es mejorar el reconocimiento del habla limpiando la señal de sonido del ruido adicionado por el ambiente. Haciendo uso de Discrete Cosine Transformation, que posee un algoritmo similar al de Fast Fourier Transformation se extraen características de vídeo para ser usadas junto a las señales de audio. Por otra parte, para la señal de audio se hace uso de Mel Frequency Cepstral Coefficients (MFCCs) que es una señal de menor tamaño que contiene la información de las características que traía la señal analizada. Luego, se concluye en el trabajo de Karel que se obtiene un mejor reconocimiento del habla adicionando características visuales que eliminando el ruido de la señal de audio. Esto podría presentarse debido a que, cuando se elimina el ruido de la señal también se esté perdiendo información pertinente para la correcta interpretación.

Existen diferentes aplicaciones haciendo uso de las señales audiovisuales que tienen un enfoque a una realidad más concreta. Por ejemplo, estas herramientas también han sido objeto de estudio para el reconocimiento de escenas violentas como es el caso del trabajo de Theodoro Giannakopoulos et al. Una de las medidas usadas para la detección de escenas violentas son la sangre, medida de actividad, la presencia de armas de fuego y/o explosiones. En este escenario la señal de audio es analizada usando siete características, donde tres son interpretadas como violentas y cuatro como no violentas. Para extraer las características se hace uso de medidas estadísticas sobre la señal de audio como la media, la desviación estándar de diferentes segmentos de la grabación analizada. Por otra parte, se expone que uno de los principales retos a la hora de trabajar con las señales de vídeo es la gran variación de las escenas violentas. Para el reto anterior, se definieron tres tipos de estados en los vídeos que son; no-activity, normal-activity y high-activity donde se encontró que un gran porcentaje de las escenas de high activity correspondían a escenas violentas. Sin embargo, se afirma que no se puede asumir que las escenas calificadas como high activity corresponden solo a escenas violentas, sigue existiendo un porcentaje que pertenecen a otro tipo de actividades. Por consiguiente, medidas como el promedio del movimiento (AM Average Motion) y Varianza de la orientación del movimiento (MOV Motion Orientation Variance) se usaron para tener características que permitieran definir más acertadamente la detección de las escenas violentas en los vídeos. Por último, Theodoro et al., proponen hacer uso de Clasificador binario KNN que vienen siendo determinado por el número de características que contiene en conjunto las señales audiovisuales. Para finalmente concluir que, solo no se alcanzaron a detectar el 17% de las escenas violentas, dejando que, por lo menos 1 de 2 eventos violentos fueron detectados.

Al manejar diferentes tipos de señales la complejidad del procesamiento de información aumenta. Por lo tanto, en la literatura consultada se evidencia una tendencia de métodos que hacen uso de algoritmos, estos métodos son Convolutional Neural Networks (CNN), Hidden markov models (HMM) y Deep Bayesian Network (DBN) que permiten realizar la toma de decisiones adecuada basada en la información que pueda proveer las señales de audio y vídeo. En el caso de CNN es una neurona artificial que está organizada de manera muy semejante a las neuronas de algunos sectores de un cerebro biológico, este modelo es muy efectivo para tareas de visión artificial cuando se habla de segmentación y clasificación de imágenes. En cambio, cuando se habla de HMM se refiere a un modelo estadístico que hace uso de los parámetros conocidos para encontrar los "ocultos", esto determinando la probabilidad que existe entre la transición de estados. Por último, DBN es un modelo estadístico en un grafo acíclico dirigido que está conformado por un conjunto de variables aleatorias y sus dependencias condicionales que buscan predecir el siguiente estado. Estos solo son unos de los más conocidos, pero existen muchos más métodos que permiten realizar la relación entre características de señales diferentes. Por ejemplo, Theodoro, mencionado ya anteriormente, para esta tarea hicieron uso de un clasificador binario. La decisión de elegir el método más adecuado para la fusión de las señales audiovisuales reside en cierta medida en la forma en que procese la información de las señales de entrada.

Las señales de entrada, del bloque de fusión de señales ya están definidas en la sesión de criterios de selección. Basados en estos criterios, ver tabla 1, se decidió que es pertinente implementar un algoritmo de árbol de decisión que tienen como finalidad que por medio de diferentes datos va optando por cuáles acciones ir tomando. Métodos poderosos y robustos como CNN, HMM o DBN también hacen uso de estos algoritmos y varios adicionales para tener predicciones o decisiones más aceptables. Sin embargo, la principal razón por la cual no se implementa uno los mencionados es la carencia de un Dataset, que es un conjunto de datos que sirve para entrenar estos sistemas, debido a que son métodos basados en procesos estadísticos uno de los pilares fundamental son los datos y tener los suficientes para entrenarlos es una ventaja que en este caso no se tiene.

2.1.3 Algoritmo

El diseño del algoritmo de fusión que se presentará a continuación fue realizado por el autor de este proyecto de grado. Si bien fue planteado como un algoritmo de árbol de decisión este perdió ese diseño, pero en esencia sigue cumpliendo esta condición. Las entradas del algoritmo son los vectores con la información extraída de las señales de audio y vídeo, de estas se toman los comandos propuestos por el usuario para realizar la toma de decisiones. Hasta el momento, solo se ha mencionado que se hará uso de diccionarios para comandos de audio y comandos de vídeo. Más adelante, se hablará con mayor detalle de ambos, pero es pertinente afirmar que estos fueron fundamentales a la hora del diseño del algoritmo. La toma de decisiones estaba basada en encontrar una coherencia dentro de las palabras y gestos que pueden ser combinados de diferentes formas por el usuario, también de la exigencia de los parámetros requeridos para manejar el robot. Basados en esto, las opciones principales que podrá realizar el UR3 son dos; preguntar por más información o realizar una acción con la información que ya tiene a su disposición. Una aclaración pertinente referente a la opción de "preguntar" es, no se estará interactuando constantemente con el usuario sino revisando si aún existe información disponible de la que pueda hacer uso para completar

requerimientos para los datos de control o si por el contrario debe omitir y completarlos con valores predeterminados.

El algoritmo se puede simplificar de dos ramas principales que son la de vídeo y audio. A pesar de estar separadas dentro de sus procesos individuales hacen uso de información de la otra rama. Por ejemplo, en el caso del audio, a la hora de exigir el agarre de una pieza en especial deberá basarse en información entregada por el módulo de vídeo para conocer donde se encuentra el cubo, haciendo referencia a una ubicación espacial. Sin embargo, para el escenario del vídeo no sucede mucho esto, pues los comandos gestuales no hacen uso de la información proporcionada por el módulo de audio. Esto podría ser un punto interesante a evaluar a futuro en un próximo trabajo, pero para el caso actual y los comandos proporcionados no se encontró mucha coherencia en realizar uso del audio en la rama de vídeo. Ver figura 2.5.

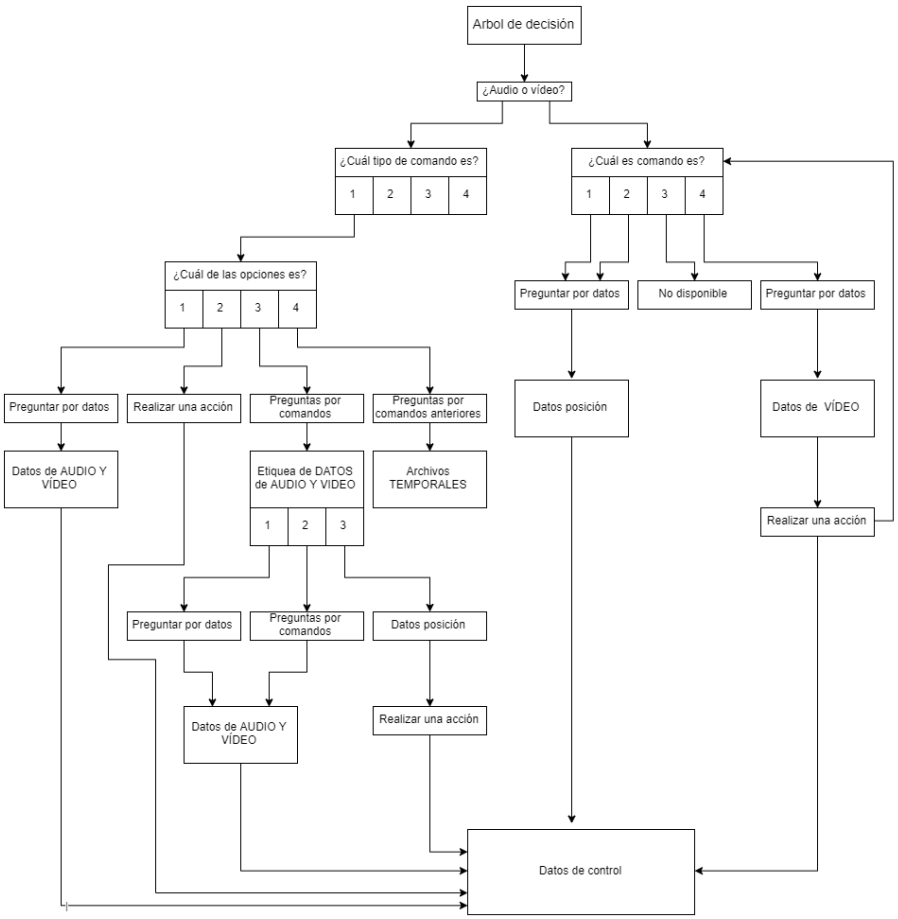


Figura 2.5: Diagrama de flujo del funcionamiento general del algoritmo de fusión de señales.

En la anterior figura podemos apreciar de forma gráfica lo explicado en los párrafos anteriores. Este es un diagrama simplificado de las decisiones y posibles opciones que tiene en sus divisiones después de una pregunta. Antes de llegar a una las opciones como asignar datos específicos de vídeo o audio, preguntar por más información o realizar una acción pueden presentarse más de dos preguntas, que terminan siendo más que las que se pueden apreciar en la anterior figura.

Si bien es cierto que se hace el uso de los diccionarios, este no es revisado continuamente en el algoritmo como se podría llegar a entender. Por el contrario, antes de hacer uso del algoritmo los datos de vídeo y de audio son usados para definir etiquetas con la información que está requiriendo el árbol de decisión para trabajar. Los diccionarios son archivos que contienen todos los comandos, revisarlos constantemente para verificar si el comando que ingresó existe, qué tipo de comando es, cuál de la lista de esos tipos corresponde, qué datos exige y demás, se terminaría convirtiendo en tareas que agregarían tiempo de procesamiento al programa. Por tal razón, se planteó el esquema que se muestra en la figura 2.6.

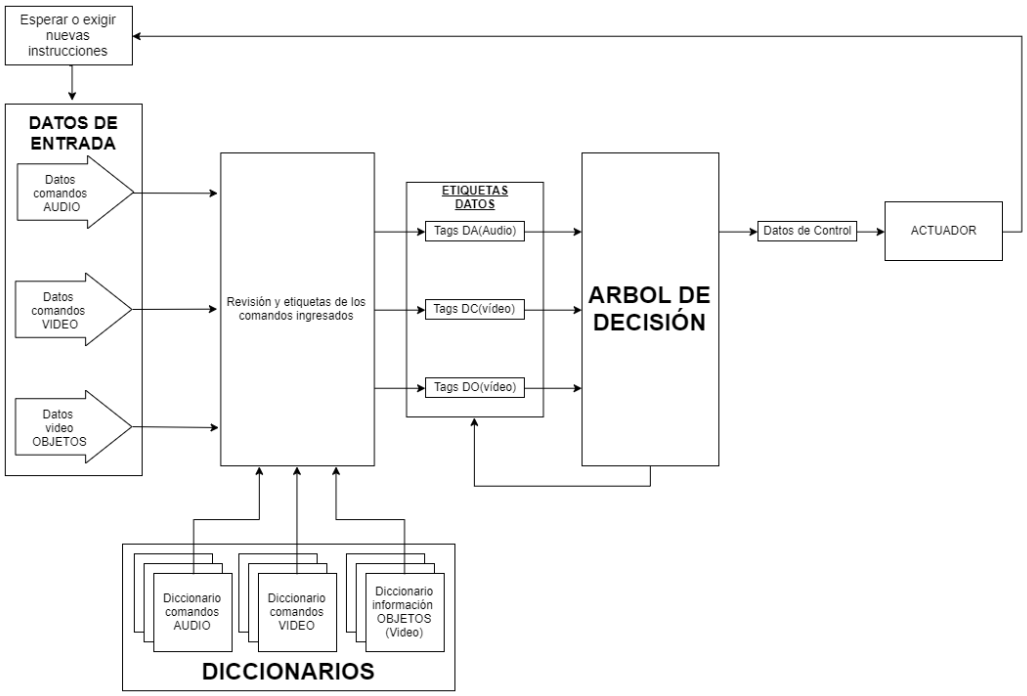


Figura 2.6: Diagrama de bloques que representa las diferentes entradas y salidas del sistema general.

El uso de las etiquetas busca facilitar la tarea de verificación y revisión de los comandos. De esta forma el árbol de decisión no deberá hacer todo el proceso de revisar los diccionarios en cada pregunta, sino las etiquetas tendrán todo lo relevante que él debe conocer. Para entender esto mejor se mira con un ejemplo. Imagine que la orden que se le entrega al UR3 es “Recoge la pelota roja”, esta instrucción es tomada por el módulo de audio y convirtiendo la señal de sonido en un string o también llamada cadena de caracteres, estos son los datos que le entran directamente al módulo de etiquetas. Este estructurará la etiqueta que se entrega al árbol de decisión. Para esto, mirar la figura 2.7.

Figura 2.7: Diagrama de bloques del funcionamiento del algoritmo.

Esta etiqueta no es para todos los casos del mismo largo, existirán casos donde serán de menor tamaño. Tomando el ejemplo de la anterior figura la etiqueta comienza con una letra 'A' que hace referencia a la rama de AUDIO y de esta forma determinando cuál será el camino a seguir. Luego, se puede apreciar que la cadena de letras y números, estas son las etiquetas que contienen la información del diccionario, una vez este hecho el proceso de verificación de los comandos resulta en eso. En el ejemplo que se tiene "M2 O1 D2" se refiere a comando de tipo movimiento y de esa lista de comandos es el número 2, lo mismo para los demás, O hace referencia a comando de tipo objeto y el número a cuál de la lista y así para el caso de D2 también. Por último, la letra que se encuentra al final de la etiqueta concierne al final de esta, que significa para el algoritmo que no hay más datos para hacer uso.

Este modelo de etiquetas se pensó siendo inspirado por la capa 3 del modelo OSI. Esta capa concierne el enrutamiento de los paquetes de datos que se envía por medio de una red de comunicación. En estas redes el direccionamiento se hace por medio de IPv4 o IPv6, es claro que aquí no se usa estos protocolos de comunicación, pero la similitud se plantea de la siguiente forma, comienza la información en un punto y por medio de estas etiquetas busca recorrer el camino adecuado para llegar al destino deseado. En el caso del proyecto, el destino deseado acción que el usuario le pide al robot y por medio del algoritmo llega al UR3.

2.2 Definición de comandos y capacidades del sistema

En este apartado se definen diferentes aspectos que deben estar enlistados para comprender el contexto en el que se está trabajando. Aquí se habla sobre las capacidades físicas del UR3, dispositivos que se piensan usar, que están integrados al robot y adecuaciones que se deberían realizar al espacio de trabajo. Parte importante de este apartado es la definición del ambiente de trabajo del UR3 y los comandos audiovisuales que están disponibles para manejar el robot.

2.2.1 Definición de los comandos de audio y vídeo

Al definición de los comandos de audio y vídeo se buscaba que fueran fácil de usar por los usuarios, así su interacción con el robot sería menos tediosa. Se procuraría utilizar un lenguaje natural y evitar llegar a un lenguaje técnico y poco usado por las personas en su diario vivir. Sin embargo, realizar una caracterización del lenguaje termina siendo difícil ya que, este está influido por diferentes componentes como el social, étnico, económico, grado de estudio y creencias, por nombrar solo algunos. Por ello, se realizó una encuesta donde se explicaba los principales puntos del proyecto de grado y que fueran los mismos usuarios los que propusieran frases, palabras y oraciones en algunos casos que permitieran estar más cercano al lenguaje que se usa en la universidad. Por lo tanto, estaría sujeto a personas de habla español con una educación entre bachiller y posgrados, con poco conocimiento sobre la robótica ya que pertenecen a carreras de la facultad de humanidades, administrativa y de la salud. Esto se hizo obedeciendo a comprender los rasgos que influyen sobre la creación de los comandos que definirán el actuar del UR3.

Conversando con algunos candidatos de sujetos de pruebas se escucharon algunas formas en las que ellos tratarían de interactuar con el UR3. A Los candidatos se les explicó con anterioridad cuál era el contexto en el que se encontraban para luego poder dar respuesta a las preguntas. Estas fueron algunas respuestas a las preguntas que se realizaron.

PREGUNTA 1: ¿Cómo le dirías al UR3 que debe colocar la pelota de color rojo en el punto A?

- Sujeto 1: “Levanta la pelota roja y ponla aquí”
- Sujeto 2: “Toma la pelota roja y colócala en el punto A”
- Sujeto 3: “Recoge la pelota roja y muévela aquí”
- Sujeto 4: “Coge la pelota roja y ponla allí”

PREGUNTA 2: ¿Cómo le pedirías lo de antes pero solo puedes darle una instrucción a la vez?

- Sujeto 1: “Coge la pelota roja, muévete hacía arriba, ponlo en el punto A”
- Sujeto 2: “Levanta la pelota, ponla aquí”
- Sujeto 3: “Levanta la pelota, levántate, coloca la pelota roja en el punto A”

PREGUNTA 3: ¿Cómo manejarías el robot para ordenar las pelotas por color (azul, rojo, verde)?

- Sujeto 1: “Coge la pelota azul y ponla aquí, coge la pelota roja y ponla aquí, coge la pelota verde y ponla aquí”
- Sujeto 2: “Recoge la pelota azul, colócala aquí, recoge la pelota roja, colócala aquí, recoge la pelota verde, colócala aquí”
- Sujeto 3: “Coloca una pelota azul aquí, a la izquierda ponme una pelota roja, a la izquierda nuevamente ponme la pelota verde”

Las anteriores respuestas se tomaron como un acercamiento con el usuario para definir una serie de comandos de audio y vídeo con la finalidad de ser cercanos al lenguaje que usarían a la hora de tener una HRI. Tomando como ejemplo las respuestas a la pregunta se comprende que los comandos de voz son más propensos a ser usados en la interacción humano robot. Además, se usan diferentes verbos para denotar acciones semejantes. Por otra parte, la señal de vídeo es usada hasta ahora como una manera de ingresar con mayor facilidad los puntos de referencias a donde se desea mover el objeto, en otros casos expresaron que usarían los gestos para arrastrar el robot hasta un punto cercano a lo que necesita recoger, o en algunos casos, solo hacían uso de los comandos de

audio. Tomando en cuenta todos estos aspectos se procedió a definir una serie de comandos de audio y vídeo, que son presentados a continuación. Sin embargo, no se pretende cumplir con todas las opciones propuestas sino con las básicas que permitan determinar si usando comandos de audio y vídeo es más fácil manejar un robot de estos y también aumentar el número de personas que pueden interactuar con él en un ambiente industrial.

2.2.4.1 Diccionario de comandos de AUDIO

Los comandos de audio fueron divididos en cuatro secciones dependiendo de su utilidad. Existe más de una palabra que va a denotar la misma acción y en lo posible se busca tener sinónimos para evitar este tipo de limitaciones. Comenzando por la tabla de comandos de movimiento que van a denotar acciones concretas para el UR3. Ver tabla 2.1

Número	Comandos de movimiento	Descripción	Código	Ejemplo de uso
1	Ir/Moverse/Muévete	Este comando permite moverse hacia alguna de las 4 direcciones dependiendo del complemento.	M1	"**Ir** hacia la **izquierda 15 centímetros**" "**Moverse** hacia **arriba 2 centímetros**"
2	Agarra/agarre/ coge/toma/levanta/ recoge/agarrar/coger/ agarre/agárralo/cógelo/ tómalo/levántalo	Este comando permite recoger uno de los objetos ya predeterminados que se tienen.	M2	"**Levanta** el **cubo verde**"
3	Poner/muévelo/ colócalo/ponlo/soltar/ Dejar/dejarlo/déjalo /ponerlo	Este comando permite colocar uno de los objetos recogidos en un lugar en específico.	M3	"Y **ponlo aquí**" "Y **muévelo** al **punto B**"
4	Girar/rotar	Este comando permite que la muñeca del brazo robótico gire sobre su eje.	M4	"**Gira** hacía la **izquierda**"
5	Bajar/descender/Baja/ Desciende/baja	Este comando permite que el brazo robótico baje (-Z)	M5	"**Baja 5 centímetros**"
6	Subir/ascender/sube/ Asciende/ascender	Este comando permite que el brazo robótico suba (+Z)	M6	"**Sube 6 centímetros**"
7	Casa/home	Este comando te permite posicionar al UR3 en una posición "casa" que es predeterminada	M7	"**volver a casa**"
8	Stop/parar/detenerse/ Salir/detener/chao/ adios	Permite que el robot se detenga	M8	"**Stop**"

9	Abrir gripper	Permite abrir el gripper del UR3	M9	"**Abrir gripper** "
10	Cerrar gripper	Permite cerrar el gripper del UR3	M10	"**Cerrar gripper**"

Tabla 2.1: Tabla de comandos de audio enfocados a movimientos.

Ahora bien, se definen otras tablas que son los complementos para los comandos de movimiento. Algunos de estos comandos se pueden evidenciar en los ejemplos de la tabla 1 como lo son "**Izquierda**", "**ponlo aquí**", "**cuadrado verde**" entre otros. En las siguientes tablas se podrán apreciar estos comandos de audio.

Número	Comandos de dirección	Descripción	Código	Ejemplos de uso
1	Derecha/ derecha	Este comando me da la dirección de derecha del usuario.	F1	"**Ir** hacía la **derecha 15 centímetros**"
2	Izquierda/izquierdo	Este comando me da la dirección de izquierda del usuario.	F2	"**Gira** hacía la **izquierda**"
3	Arriba	Este comando me da la dirección hacia arriba del usuario y robot	F3	"**Moverse** hacia **arriba 2 centímetros**"
4	Abajo	Este comando da la dirección hacia abajo del usuario y robot	F4	"**Moverse** hacia **abajo 2 centímetros**"
5	Adelante/frente/adelante	Este comando da la dirección hacia adelante del usuario	F5	"Al **frente 20 centímetros**"
6	Atrás/atrás/retroceder	Este comando da la dirección hacia atrás del usuario	F6	"**Atrás 13 centímetros**"

Tabla 2.2: Comandos de audio enfocados a la dirección

Número	Comandos de objetos	Descripción	Código	Ejemplos de uso
1	CÍrculo	Es un cubo con la imagen de un círculo en una de sus caras puede venir de color verde, rojo o azul.	O1	"**Agarra** la **pelota roja**"
2	Cuadrado/Cubo	Es un cubo que puede venir de color verde, rojo o azul.	O2	"**Levanta** el **cubo verde**"
3	Triángulo	Es un cubo que tiene una cara con la figura	O3	"**Levanta** el **cubo azul**"

Tabla 2.3: Comandos de audio enfocados a los objetos

Número	Comandos de detalles	Descripción	Código	Ejemplos de uso
1	Rojo/Azul/Verde	Comandos de colores que permite adicionarle una característica adicional a un objeto.	D1	"La **pelota roja**" "El **cubo rojo**" "El **triángulo verde**"
2	"# grados"	Permite dar un número de grados en específico para rotar a la muñeca del robot.	D2	"**Rota 45 grados** hacía la **izquierda**"
3	"# centímetros"	Permite dar un número de centímetros en específico para moverse en una dirección dada.	D3	"**Moverse** hacia **abajo 2 centímetros**"
4	X, O ó P	Ubicación predeterminada dentro del área de trabajo	D4	"**Ponerlo** en **X**"

Tabla 2.4: Comandos de audio enfocado a los detalles

2.2.4.2 Diccionario de comandos de VÍDEO

La definición de los comandos de vídeo fue difícil de pensar y se optó por la idea más sencilla haciendo uso de los recursos disponibles. Se plantearon dos comandos de vídeo que se reciben por medio del Leap Motion Y este realiza el reconocimiento o presencia de las manos. Cuando se reconoce la presencia de la mano izquierda el UR3 cierra la pinza y cuando reconoce la presencia de la mano derecha este abre el gripper. En la figura 2.8 se puede apreciar de qué forma deben ser usadas las manos.

Figura 2.8: Imagen que presenta la forma en que se deben mostrar las manos frente al sensor.

2.2.4.3 Diccionario de OBJETOS

Los objetos que van a ser usados para realizar pruebas son cubos de diferentes colores. De estos, la característica diferencial es el color. Por ende, era necesario especificar que existe también un diccionario de objetos que permite que la imagen tomada por la cámara se le realice un procesamiento por el cual se busca extraer las características de color e identificar su ubicación espacial.

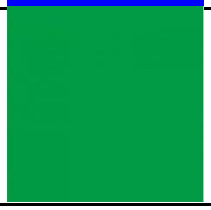
Número	Comando	Descripción	Ejemplo
1	Rojo	Color del cubo	
2	Azul	Color del cubo	
3	Verde	Color del cubo	

Tabla 2.5: Comandos de vídeo enfocado en los objetos

2.2.2 Capacidades del UR3

El robot UR3 provee una protección IP64 la cual lo escuda contra polvo y salpicaduras de líquidos, esto ayuda a ensamblarlo en diferentes ambientes de trabajo, no solo en la academia (Como es el caso del CAP de la universidad). Adicionalmente, cuenta con una certificación ISO clase 5 para áreas limpias, es decir que el robot puede ser usado en aplicaciones como biotecnología, farmacéutica, nanotecnología y diversas aplicaciones de fabricación de tecnologías limpias [40], tiene un ruido de 70dB que es el equivalente al ruido del tráfico de una autopista. Cuenta con 4 puertos digitales 2 de entrada (E) y 2 de salida (S) y tiene 2 entradas análogas. En el apartado físico del robot, este cuenta con 128mm de diámetro de la base, el cual es necesario para saber el espacio requerido en la mesa o plataforma de trabajo para poder sujetar el robot a esta. El robot está hecho de aluminio y polipropileno termoplástico, cuenta con un conector tipo M8 el cual sirve para conectar la herramienta de trabajo, este conector se usa específicamente con sensores industriales y cuenta con un tornillo impermeable de 3 clavijas, especial para entornos hostiles, en el caso del robot se pueden presentar movimientos bruscos y este conector evitaría que la herramienta se desconecte. Con el robot viene incluido un cable de 6 m/ 236 in el cual está alrededor de todo el brazo robótico una vez esté conectada la herramienta de trabajo y con la instalación del cable el robot pesaría 11 kg/ 24.3 lb. En la figura 2.9 se puede ver el robot de 6 grados de libertad.

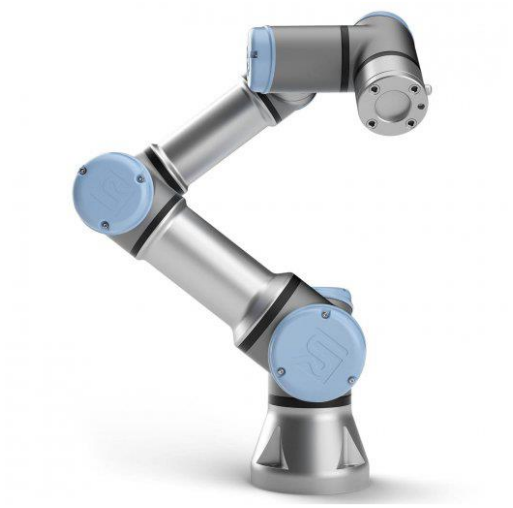


Figura 2.9: Imagen de muestra del robot colaborativo UR3.

Toda su documentación técnica se puede encontrar en la página principal de la compañía, de este lugar fue donde se extrajo la mayoría de datos que se darán a continuación del UR3 o puedes visitar este link tomado de la página también https://www.universal-robots.com/media/1801288/eng_199901_ur3_tech_spec_web_a4.pdf.

El control box es una caja de acero la cual es la encargada de alojar todo el control y la comunicación del robot UR3. Tiene un tamaño de 457mm x 423mm x 268mm y tiene un peso de 15kg / 33.1 lbs (estas características son las de fabrica). La caja de control cuenta con una protección IP20 la cual garantiza seguridad frente a solidos con diámetro superior a 12mm y cero protecciones frente a líquidos. Tiene una certificación ISO clase 6 para áreas limpias, lo cual asegura una filtración de aire de penetración de alta eficiencia de 99.9% a 0.3 micrones, tiene un ruido de trabajo menor a 65dB (10dB por encima del ruido ambiente -según OMS- [41]) lo cual equivale a un grupo de personas conversando en voz alta. Si hablamos de los puertos de entrada y de salida (I/O), tiene 16 entradas digitales, 16 salidas digitales, 2 entradas analógicas y 2 salidas analógicas, cuenta con una alimentación de 24V a 2A para estas I/O y cuenta con protocolos de comunicación TCP/IP 100Mbit, Modbus TCP, Profinet y EthernetIP. El control box funciona con una alimentación de corriente alterna de 100 V - 240 V a 50 - 60 Hz y con un rango de temperatura ambiente de 0 - 50°. En la figura 2.10 se puede ver la caja de control.

Figura 2.10: Imagen de muestra del control box del UR3.

Para el manejo del robot Universal Robots (UR) dispone de una tableta capacitiva llamada Teaching Pendant (TP) la cual es un tipo de interfaz HMI (Human-Machine Interface), cuenta con un botón de encendido del robot (Botón de Power) y un botón de emergencia (Botón Rojo) y toda su pantalla en la cual se muestra toda la interfaz gráfica para el manejo del robot. Este dispositivo viene con una protección IP20, la cual garantiza defensa frente a solidos de diámetro superior a 12mm y no provee resguardo frente a líquidos. Este dispositivo esta hecho de Aluminio y Polipropileno (PP), al tener éste materiales no tan resistentes UR provee un forro que lo protege contra golpes. El TP tiene un peso de 1.5 kg/ 3.3 lbs, lo cual no representa gran dificultad al momento de manipular el TP la primera media hora, luego ya este peso empieza a incomodar. Por último, basta nombrar que cuenta con un cable de 4.5 m/ 177 in, bastante largo para poder estar alejado del robot en alguna actividad que el robot demande espacio. En la figura 2.11 se puede ver el dispositivo antes mencionado.

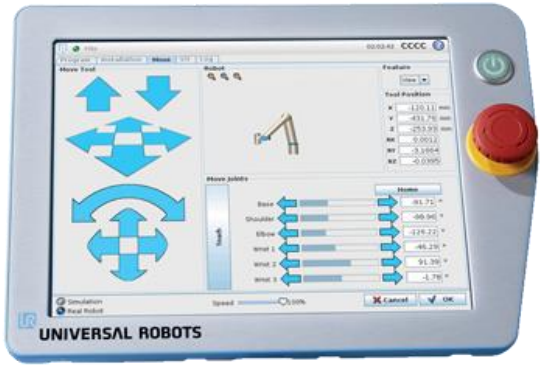


Figura 2.11: Imagen de muestra del Teachin Pendant del UR3.

En la figura 2.12 se muestra las capacidades de rendimiento del UR3. Una repetibilidad de ± 0,1 mm, la repetibilidad es tomada como la capacidad del robot de regresar a un punto en específico varias veces, un valor de 0,1 mm deja ver una precisión bastante alta para esta máquina. Puede trabajar en un rango de 0 a 50° con un consumo de energía de 90 W a 250 W como medida máxima. Cuenta con certificaciones TÜV NORD y SÜD que validan las funciones de seguridad que posee el sistema del UR3.

Rendimiento	
Repetibilidad	±0,1 mm / ±0,0039 in (4 mil.)
Intervalo de temperaturas	0-50°*
Consumo de energía	Mín. 90 W, estándar 125 W, máx. 250 W
Operación de colaboración	15 funciones avanzadas de seguridad regulables. Función de seguridad con certificación TÜV NORD Probado de acuerdo con las normas: EN ISO 13849:2008 PL d

Figura 2.12: Especificaciones técnicas de rendimiento del UR3.

En la figura 2.13 se aprecia de las especificaciones sobre la carga útil máxima que puede soportar el UR3 que son 3 Kg, un alcance de 500 mm que son equivalentes a 0,5 metros. Por otra parte, posee

seis articulaciones giratorias y una interfaz gráfica de 12” con soporte. Evidentemente el total de la carga útil no será usado, la carga que se maneja es, por mucho, menor. Por otra parte, las seis articulaciones son excelentes para versatilidad y agilidad del robot para realizar movimientos precisos y con dificultad. Por otra parte, UNIVERSAL ROBOTS recomienda que se tengan en cuenta el área volumetría de la base, ya que de estar expuesta a vibraciones o movimientos podría llegar a provocar que el cobot trabaje de forma ineficiente. Ver figura 2.14.

Especificación	
Carga útil	3 kg / 6,6 lb
Alcance	500 mm / 19,7 in
Grados de libertad	6 articulaciones giratorias
Programación	Interfaz gráfica del usuario PolyScope con pantalla táctil de 12" con soporte

Figura 2.13: Especificaciones de límites físicos del UR3.

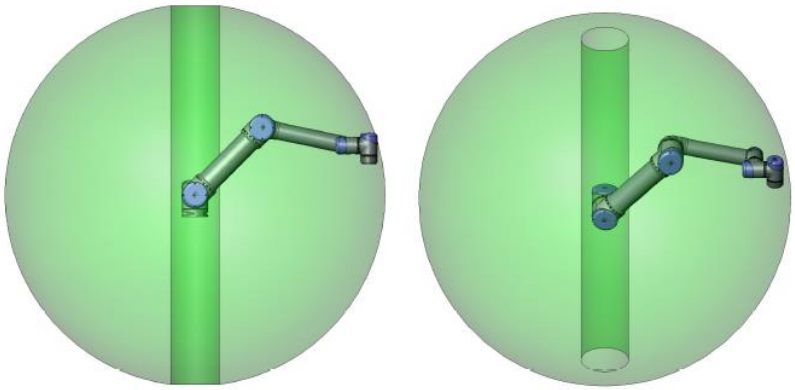


Figura 2.14: Área volumétrica recomendada del UR3.

En cuanto a las capacidades técnicas del UR3 la figura 2.15 nos muestra que sus seis articulaciones tienen nombres en específico. Desde la base hasta la muñeca tres tendríamos las seis, cada una tiene un rango de giro de ± 360°, a excepción de la muñeca tres que el rango de giro es ilimitado. La velocidad que pueden alcanzar las tres primeras articulaciones (Base, hombro y codo) es de ± 180°/s mientras que, las muñecas alcanzan velocidades de ±360°/s. Esto es de mucha importancia para conocer las limitaciones en cuestión de rotaciones del cobot.

Movimiento		
Movim. del eje del brazo robot.	Radio de acción	Velocidad máxima
Base	± 360°	± 180°/s
Hombro	± 360°	± 180°/s
Codo	± 360°	± 180°/s
Muñeca 1	± 360°	± 360°/s
Muñeca 2	± 360°	± 360°/s
Muñeca 3	Infinita	± 360°/s
Herramienta típica		1 m/s / 39,4 in/s

Figura 2.15: Especificaciones de radio de acción y velocidad del UR3.

La comunicación del robot UR3 se puede realizar por medio de protocolos de comunicación como TCP/IP 100 Mbit con el estándar IEEE 802.3u, 100BASE-TX toma ethernet y MODBUS TCP. Actualmente, el UR3 que se encuentra en el CAP goza de una comunicación TCP/IP por medio de cable ethernet conectado al robot por medio de router cuatro puertos. Esto se logra haciendo uso de las opciones de configuración de red, ver figura 2.16.

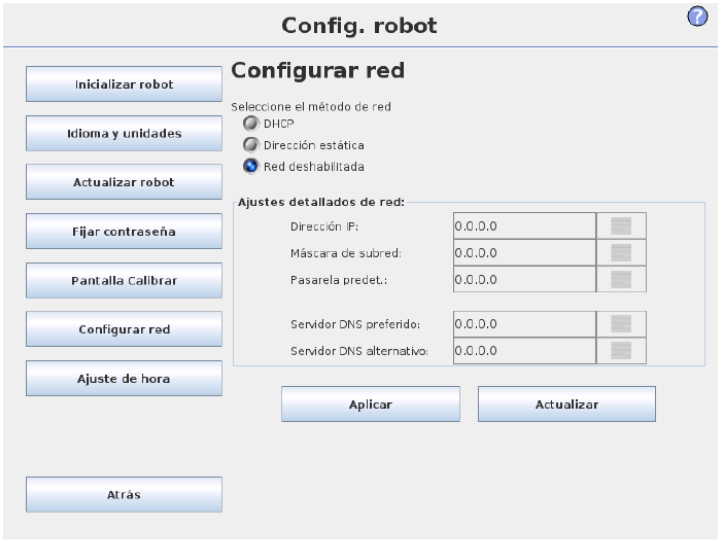


Figura 2.16: Imagen de configuración de red del UR3.

A pesar de tener datos técnicos muy positivos es necesario realizar algunas adecuaciones al espacio de trabajo para el proyecto de grado. Además, por el uso de algunas herramientas como lo son, el gripper y el sensor de torque y fuerza, se encuentran limitaciones adicionales por el cableado que usa con estas herramientas. Estas características y limitaciones se presentarán en la siguiente sección.

2.2.3 Dispositivos adicionales y limitaciones del UR3

En este apartado se explica los dispositivos o mejor llamados sensores que ya posee instalados de forma adicional el UR3. Estos no se retiran bajo ninguna circunstancia para la realización del proyecto de grado. Por lo tanto, se presenta la necesidad de documentar estos y evidenciar las limitaciones que pueden presentar a costa de su cableado. Además, dentro de este apartado, también se mencionan dispositivos adicionales que se usan para la captación de las señales de vídeo y de audio que son las encargadas de proporcionar los comandos de control para el UR3.

2.2.2.1 Dispositivos adicionales

El espacio de trabajo tiene algunas limitaciones ya incluidas por los sensores mencionados anteriormente. Los sensores con los que cuenta actualmente UR3 es un sensor FT 300-s, que se encarga de medir el torque y fuerza que ejerce el robot, una cámara Wrist y Gripper 2F-85. Las tres herramientas adicionadas al cobot son de la marca ROBOTIQ. Si bien no se usarán todos los sensores expuestos es necesario mencionarlos ya que hacen parte de la instrumentación de la que goza el UR3. El sensor que será usado principalmente será el gripper, para el desarrollo de la tesis, sin este las tareas de pick up y put down planteados en las pruebas del experimento no se podrían realizar. Por otra parte, algunas de las principales características que poseen estas herramientas son las siguientes.

Comenzando con el sensor FT-300-s, sus especificaciones presentan que puede realizar mediciones en los tres ejes cartesianos (X, Y, Z) en un rango de ± 300 N, una capacidad de sobrecarga del 500% para los tres ejes mencionado. La deflexión en la máxima carga es de 0.01 mm y tiene una masa total de 440 g. Por otra parte, es inmune a la sensibilidad del ruido exterior y tiene una tasa de salida de datos de 100Hz. Por último, usa como protocolo de comunicación Modbus RTU/Data stream (RS-485). Ver figura 2.17.

Figura 2.17: Imagen de muestra del sensor FT-300-s

Ahora, la cámara Wrist cuenta con una masa de 160 g, en cuanto a las especificaciones de sensores y óptica posee 5 MP de sensores de color y foco eléctricamente ajustable. La cámara posee algunas funciones programables como; enseñanza automatica de piezas y partes paramétricas, edición de bordes, color de objetos validación de autorización, modo avanzado y básico en el control de la cámara, permite varios parámetros como exposición, el foco, luz led, etc. Además, tiene un rango de operación de 0° a 50° C y beneficios cómo, programación en minutos de partes complejas, crear planos de trabajo solo con un click, creación automática de acción de picking con el centro de una posición entre otras muchas funciones. Ver figura 2.18

Figura 2.18: Imagen de muestra de la cámara Wrist

Ahora bien, las especificaciones del gripper son; la fuerza con la que agarra es ajustable y tiene un rango entre 20 a 235N, la carga total que puede cargar la pinza es de 5 Kg, tiene una masa de 0,9 Kg y una resolución de posición de 0.4 mm, esto lo hace bastante preciso. La velocidad con la que cierra está en un rango de 20 a 150 mm/s también es ajustable. Por otra parte, respecto a la comunicación utiliza el protocolo Modbus RTU (RS-485) con IP45. Ver figura 2.19.

Figura 2.19: Imagen de muestra del gripper.

Como se explicó en la sección del marco teórico, existen diferentes dispositivos de captación de señal de vídeo. Estos dependiendo, del desarrollo tecnológico que hayan tenido, pueden llevar a cabo de forma más completa ciertas funciones requeridas en un sistema. Recordando el ejemplo expuesto en el marco teórico, si hablamos de captar solo vídeo una cámara es suficiente, pero si necesito reconocimiento del cuerpo y profundidad, el Kinect sería el ideal. Con esto, debemos dejar claro qué requisitos se están buscando cumplir y cuáles cumplen adecuadamente sus funciones.

Tomando en consideración el diccionario de objetos y diccionario de comandos de vídeo, los requisitos necesarios es, el reconocimiento de la presencia de la mano izquierda o derecha, la captación de tres colores diferentes y la ubicación espacial o cartesiana del objeto. También se lleva a tener en consideración el lugar dónde va a estar ubicado, que posición es la recomendada o adecuada para el usuario dependiendo de las variables externas que se puedan presentar en el ambiente. Basado en lo anterior se tomaron criterios para evaluar cuál dispositivo era más apto o prometía ser más fácil trabajar con él. Además, también se presentan los dispositivos que se tuvieron en consideración. Ver la tabla 2.6

Característica	Cámara ZED	Kinect v1	LEAP MOTION	Logitech C920	Wrist
Lenguaje de programación disponible	Python, C#, C++	C#, C++, Python	Python 2, C#, C++, JavaScript, Unity, Java	Python, C#, C++	Python
Sistema operativo	Windows 7 y superiores	Windows 7 y superiores	Windows 7 y superiores	Windows 7 o superiores	Sistema UR3
Precio	1'919.620	150.000	341.692	344.900	
Tamaño	175x30x33 mm	103x39x126 mm	30x80x11.3 mm	43.3x94x71 mm	
Adecuación	Necesaria	Necesaria	Ninguna	Necesaria	Ninguna
Información 2D o 3D	3D	3D	3D	2D	2D
Documentación disponible	Pagina principal provee información sobre desarrollo de código	Algunos repositorios de código abierto y tutoriales en la red	Pagina principal provee información sobre desarrollo de código	Solo capta la señal, no es necesario para que las librerías de Python	Poca documentación de desarrollo de código
Resolución	1080p	480p	No aplica	1080p	No aplica

Tabla 2.6: Tabla comparativa de las especificaciones de los dispositivos de captación de vídeo.

El precio que se presenta ahí es a un cambio de dólar de 3.800 pesos colombianos. El lenguaje de programación es necesario conocerlo para saber si existía la posibilidad de integrarlo a los módulos que se desarrollan en este lenguaje. Se evidencia que en las cinco opciones de dispositivo todos tienen la disponibilidad de trabajar en el lenguaje necesitado. Se buscaba que el sistema operativo fuera Windows 10 ya que este era el disponible en el portátil que se usaba para el desarrollo del proyecto. El precio más elevado es el de la cámara Wrist perteneciente a Robotiq y el más bajo al Kinect. El tamaño es una variable que era necesaria consultar para evaluar si era las adecuaciones adicionales al espacio del trabajo para que estuviera la cámara ubicada en un lugar adecuado. El Leap Motion y la cámara Wrist no requieren adecuaciones, la primera opción porque existe el espacio para ubicarlo en la mesa de trabajo, la segunda porque ya se encuentra instalada directamente en el UR3. Por otra parte, las tres restantes deben ser ubicadas en un trípode o una plataforma distinta para estar a la altura de la mesa de trabajo. El Kinect en este apartado iba siendo uno de los descartados, ya que debido a sus dimensiones resultada algo difícil de ser posicionado con el trípode que se tenía a disposición. Luego, se evidencia qué información suministra cada dispositivo, la cámara C920 y la Wrist son las únicas que no suministran información 3D. En este apartado se discutía y evalúo que tan necesaria era la idea de tener información 3D con el experimento que se tenía concebido hasta el momento. El experimento consta de un ejercicio de Pick up y Put down, recoger y colocar en otra ubicación. Por lo tanto, se necesita recoger de un área definida como el espacio de recogida y otro el espacio de colocar o poner. Bajo estos requisitos sería una opción trabajar con información 3D para poder trabajar con los tres ejes cardinales XYZ, podríamos aprovechar completamente el área de trabajo que si lo hiciéramos solo con información

2D. Sin embargo, el trabajo busca implementar un algoritmo que maneje señales audiovisuales y permita evidenciar que manejar ambas señales resulta más adecuado que usar solo una. Por lo tanto, usar información 3D puede ser beneficioso y sacar grandes avances de este tipo de información, pero se elige trabajar con información 2D por simplicidad. Se continúa con la documentación disponible en la web para trabajar con estos dispositivos, se evidencia de forma muy breve que algunos poseen mayor cantidad de información accesible que otros.

En consecuencia, después de evaluar estos aspectos se eligieron dos dispositivos que se presentaron en la tabla. Estos cumplen con los requisitos presentados anteriormente. El primero es el LEAP MOTION para el reconocimiento de los comandos de vídeo. La principal razón por la que se eligió este dispositivo fue por el desarrollo tecnológico que presenta alrededor del reconocimiento los gestos que se pueden realizar con las manos. Con un dispositivo diferente al mencionado, se tendría que implementar un programa de procesamiento de imágenes de vídeo donde se reconozca la presencia de las manos, luego una adecuación adicional para que logre diferenciar la derecha de la izquierda. Con el LEAP MOTION esto ya se encuentra realizado, el reto es extraer la información que entrega al PC por medio de una entrada USB. El segundo dispositivo fue la cámara C920 de marca Logitech. El tamaño presentado es adecuado para que la cámara sea usada con el trípode disponible. Además, La diferencia de precios con la cámara ZED la hace más opcional para la captación de señal de vídeo. Se puede cuestionar porque no elegir el Kinect v1 si entrega una señal de información 3D, pero su resolución es mucho menor a la que entrega la C920 y tendría que buscarse recursos adicionales para encontrar una superficie donde estar soportada.

Las señales de vídeo serán recopiladas y procesadas por dos dispositivos. Uno estará enfocado al reconocimiento de gestos manuales y el otro será enfocado a la visualización del espacio de trabajo del UR3 y reconocimiento de las diferentes fichas que estén presentes en el espacio de trabajo. Comenzando por el sensor encargado de los gestos manuales este es el Leap Motion, un sensor que lleva décadas de desarrollo. Este sensor es diseñado especialmente para el reconocimiento de los gestos de las manos y así mismo su proveedor presenta diferentes usos y aplicaciones. El Leap Motion es alimentado a 5 Voltios DC, usa un conector USB 2.0, su zona de reconocimiento va hasta una profundidad de 60 cm, su campo de visión posee una apertura de 140x120° y según el proveedor su funcionamiento es aceptable en diferentes condiciones ambientales. Esto quiere decir que no se restringe a un solo espacio en especial. También posee dos cámaras de infrarrojo cercano de 640x240 píxeles; espaciados 40 milímetros; con ventana infrarroja transparente, opera en el rango espectral de 850 nanómetros +/- 25; normalmente funciona a 120 Hz; capaz de capturar imágenes en 1 / 2000th de un segundo. Ver figura 2.20.

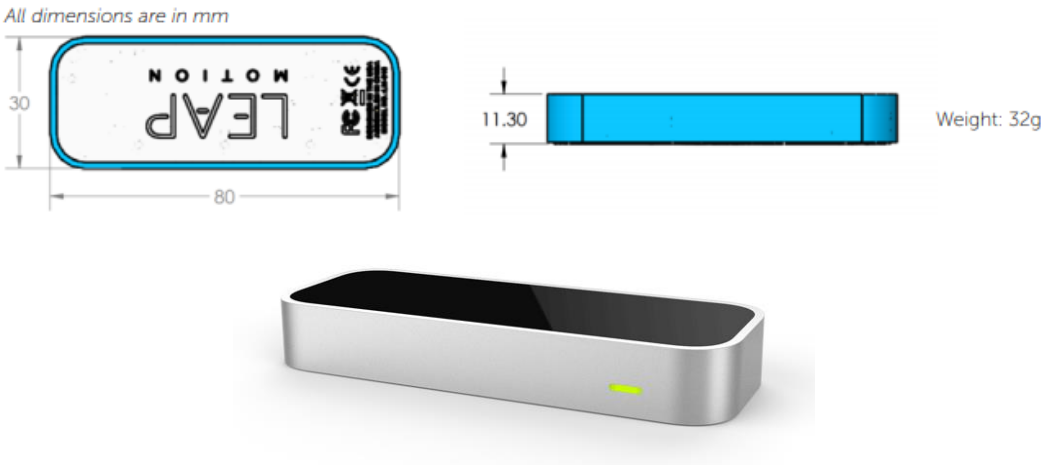


Figura 2.20: Imagen de muestra de las dimensiones del Leap Motion.

La cámara que se usa para el reconocimiento de las características de las fichas y el espacio es una Camera Logitech C920. Es una cámara que proporciona imágenes y vídeos de alta calidad en Full HD 1080p a 30 fps, teniendo una calidad de foto de 15 MP. Esta es compatible con sistemas operativos como Windows XP, Vista y 7, también para Mac OS X 10.5 o mayores. Posee un campo visual diagonal fijo de 78°. Además, C920 está dotada de corrección de iluminación automática llamado RightLight 2. Por otra parte, tiene conexión plug and play USB-A. Ver figura 2.21.

Figura 2.21: Imagen de muestra de la cámara Logitech C920

Como se explicó en la sección del marco teórico, existen diferentes dispositivos de captación de señal de audio. Estos dependiendo del desarrollo tecnológico que hayan tenido pueden llevar a cabo de forma más completa ciertas funciones requeridas en un sistema. Recordando el ejemplo expuesto en el marco teórico, si hablamos de captar señales de sonido existe una gama variada de micrófonos con diferentes prestaciones que te pueden entregar una señal bastante adecuada, donde la presencia de ruido sea poca. Por lo tanto, se hace necesario tener una serie de criterios para determinar cuál dispositivo es más apto

Tomando en consideración el diccionario de comandos de audio, y el reconocimiento del habla del usuario, logrando captar las palabras que este usa para comunicarle las ordenes al UR3. Por lo tanto, se hace necesario una señal bastante limpia de ruido. También se lleva a tener en consideración el lugar dónde va a estar ubicado, que posición es la recomendada o adecuada para el usuario dependiendo de las variables externas que se puedan presentar en el ambiente. Basado en lo

anterior se tomaron criterios para evaluar cuál dispositivo era más apto. Además, también se presentan los dispositivos que se tuvieron en consideración. Ver la tabla 2.7.

Característica	Micrófono Yeti	Diadema Logitech
Dimensiones	120x125x295 mm	188x195x87 mm
Precio	440.000	420.000
Software	No posee	Visualizador de configuración
Sistema operativo	Windows 7 o superiores	Windows 7 o superiores
Conexión PC	USB	USB

Tabla 2.7: Comparación de capacidades de los dispositivos de captación de sonido

La principal razón que influyo fuertemente en la decisión de elegir el Microfono Yeti fue el hecho que está diseñado, pensado y desarrollado para captar sonido. Además, tiene cuatro diferentes configuraciones que permite captar la señal ideal para cuatro diferentes aplicaciones, dentro de estas configuraciones se encuentra el cardiode que es la solicitada. Mientras que, por el lado de la diadema Logitech su desarrollo no está pensado netamente en la captación de la señal de sonido sino también en la reproducción de este. Por lo tanto, entrega un mayor grado de confiabilidad el hecho que el dispositivo elegido este pensado y diseñado para la tarea de se necesita.

Por consiguiente, para las señales de audio se harán uso un micrófono Yeti. Este consume una corriente de 150 mA y una tensión de 5V. Tiene una frecuencia de muestreo de 48KHz, una tasa de 16Bit. Presenta diferentes configuraciones de patrones polares los cuáles son: Cardiode que es el recomendado para grabar a una persona hablando, bidireccional, omnidireccional y estéreo. Una respuesta en frecuencia de 20Hz a 20KHz. Por último, se conecta a una computadora por medio de cable USB y es compatible con Windows 7, 8.1 y 10, también esta para macOS 10.10 en adelante. Figura 2.22.

Figura 2.22: Imagen de muestra del micrófono Yeti.

La comunicación con el UR3 se hace por medio de cable UTP Categoría 6 (CAT 6), este es una medida estándar para las comunicaciones de tipo Giga Ethernet, hay otros cables de categorías inferiores

como el CAT 5, CAT 5e y el CAT3 que son compatibles con este tipo de comunicación. Sin embargo, el cable CAT 6 posee especificaciones más estrictas para disfonía (crosstalk) y el ruido en el sistema. El estándar de cable proporciona un rendimiento de hasta 250 MHz y es adecuado para 10BASE -T / 100BASE -TX y 1000BASE -T / 1000BASE -TX (Gigabit Ethernet). Recordando que, el UR3 tiene una comunicación Modbus y 100BASE – TX, lo cuál lo hace la categoría más adecuada para la comunicación. Ver figura 2.23.

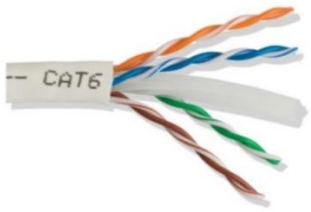


Figura 2.23: Imagen de muestra del cable UTP.

La comunicación con el UR3 no es directa entre el portátil o laptop que se esté usando. Esta es hecha por medio de un router TP link WR840N. Este router opera de forma inalámbrica y alámbrica, pero se tiene un foco en la comunicación alámbrica porque es la que ya se encuentra en funcionamiento con el UR3. Tiene un puerto 10/100Mbps WAN y 4 puertos 10/100Mbps LAN, se alimenta con una fuente de 9 voltios DC y 0.6 Amperios. Por último, tiene dimensiones 3,55 x 18,29 x 12.7 (Alto, Ancho, Profundidad) centímetros. Ver figura 2.24.

Figura 2.24: Imagen de muestra del router TP Link WR840N

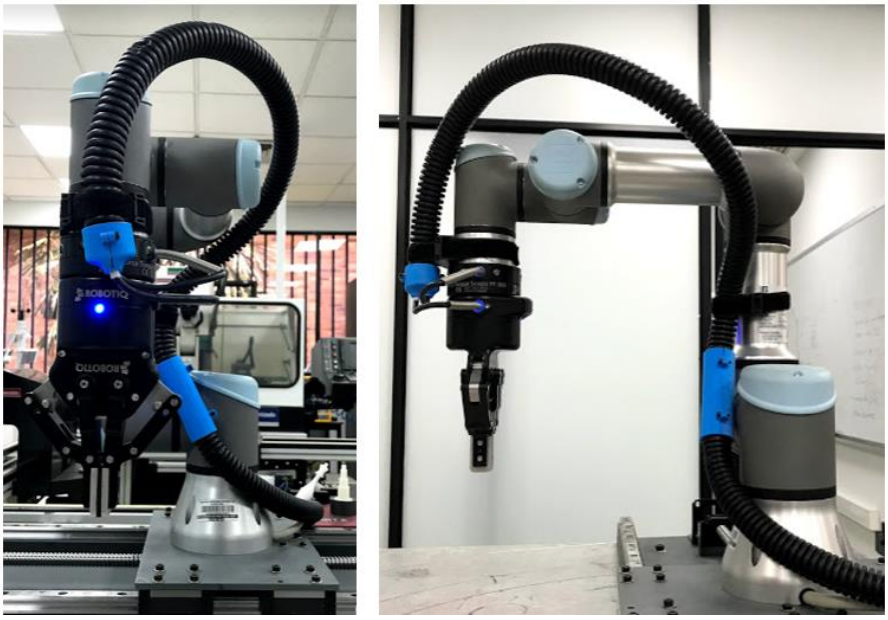
2.2.2.2 Limitaciones del UR3

Por los sensores adicionados al UR3, que son el sensor de torque y fuerza, gripper y la cámara, estos llevan consigo un cableado para el transporte de la información que se recopila. Por tal motivo, los grados de libertad poseen limitaciones diferentes a las expuestas en la sesión de documentación del cobot a utilizar en la tabla 3. Usando las funciones de movimiento libre del UR3 se procedió a determinar cuál sería los ángulos de rotación que tendría cada articulación. Se parte de una posición base. En la siguiente figura 2.27, se puede apreciar los cables que se mencionan anteriormente.

(a) (b)

Figura 2.25: a y b son imágenes de muestra del cableado del UR3.

Por otra parte, La posición base es elegida por ser la más recurrente a la hora de desarrollar actividades del estilo pick up and put down que son las propuestas a realizar en el experimento. Esto con el fin de saber qué tipo de movimientos puede realizar el UR3 en este instante, ya que podría llegar a dar giros de forma inadecuada o no permitida que termine por dañar una de estas conexiones. En la siguiente figura se puede apreciar la posición base que fue mencionada. Se tiene la intención de hacer movimientos en diferentes direcciones en cada articulación para llegar hasta un límite donde los cables se vean involucrados y corran riesgos. De esta forma, recomendaremos no realzar giros cercanos a ese valor de grados. Ver figura 2.28.



(a) (b)

Figura 2.26: a es la vista frontal de la posición llamada “Casa” o “Home”, b es la vista lateral.

La primera articulación, que en la tabla 2.8 se aprecia como base, se giró en el sentido de las manecillas del reloj teniendo un desplazamiento aproximado de 228°, se repitió el procedimiento en el sentido contrario y el valor del desplazamiento fue de 231°. La medida del desplazamiento máximo se determinaba cuando el cable estaba ya en una posición donde podría resultar dañado o

inducir al UR3 en un estado de bloqueo por sentir presión en sus articulaciones. Este mismo proceso se realizó para las demás articulaciones. Todos los datos recopilados están expuestos en la tabla 2.8.

Articulación	Radio de acción sin sensores	Radio de acción con sensores	Diferencia
Base	± 360°	± 230°	130°
Hombro	± 360°	± 200°	160°
Codo	± 360°	± 360°	0°
Muñeca 1	± 360°	± 270°	90°
Muñeca 2	± 360°	± 270°	90°
Muñeca 3	Infinita	± 270°	90°

Tabla 2.8: Limitación de grados de rotación de las articulaciones del UR3.

2.2.4 Definición del ambiente de trabajo

En este aparatado se tuvieron en cuenta aspectos como la iluminación, ruido, base del robot y mesa donde se encuentra ubicado el UR3. También, en esta sección entra la definición del tablero de trabajo del robot, los objetos que se van a usar para el experimento y sus características. La definición de estos permite tener unos límites de acción establecidos y claros. Esto permite determinar adecuadamente rangos de funcionamiento donde el UR3 se encuentra habilitado.

Las condiciones de iluminación y ruido del ambiente del CAP son adecuadas para el desarrollo de las actividades y pruebas del trabajo de grado. Inicialmente se pensaba que estas podrían llegar a afectar el desarrollo de alguna toma de señales. Sin embargo, la iluminación es suficiente para la correcta captación del vídeo y el ruido presente en el ambiente no interviene tampoco en la captación de los comandos de audio. Adicionalmente, mirando las especificaciones de funcionamiento de los dispositivos que se usarán no se encontró ningún problema, estos según sus datos técnicos trabajan en un rango de condiciones bastante amplio. La mesa de trabajo en la que se encuentra soportada el UR3 tiene como dimensiones 90 x 130 x 90 centímetros (Alto, Ancho, Profundidad). Debajo de ella se encuentra la caja de control del UR3 donde se puede apreciar las entradas y salidas de este. En el lado izquierdo está ubicada al UPS de seguridad para el UR3. Ver figura 2.27.

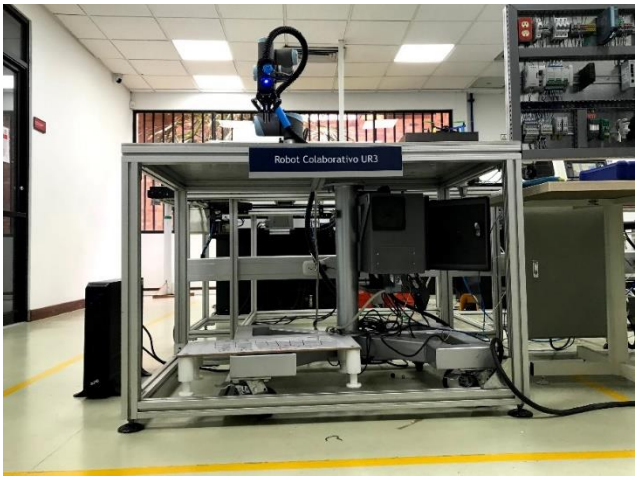


Figura 2.27: Imagen de muestra de la mesa de trabajo del UR3.

La base del robot es una plataforma móvil que hace parte de un proyecto diferente donde se pretende agregarle movimiento a la base del robot y obtener un séptimo grado de libertad. Por el momento, el proyecto no interviene de ninguna forma con el proyecto de grado y la base donde fue posicionado el UR3 no presenta ningún tipo de limitación para el desarrollo de actividades. Sin embargo, las variaciones de altura que presenta respecto a la mesa si es necesario tenerlas en cuenta. La base le agrega una altura de 7 centímetros de alto y un ancho de 130 centímetros, igual que la mesa. Ver figura 2.28.

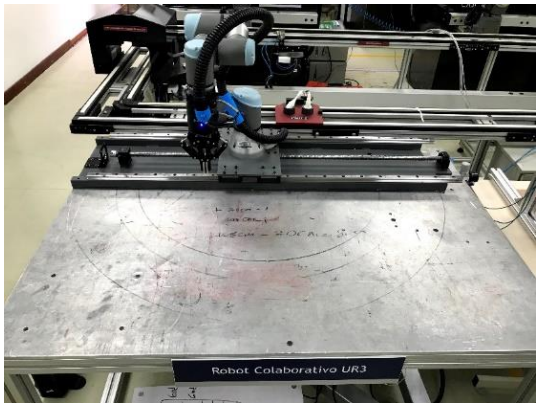


Figura 2.28: Imagen de muestra de la base y mesa del UR3.

Por otra parte, se usarán cubos para desarrollar actividades del experimento, estos cubos serán todos del mismo tamaño con las mismas dimensiones que son 5 cm, 5 cm, 5cm (Alto, Ancho, Profundidad). Cada cubo será de diferente color. La principal razón por la que se eligen este tipo de objetos es por la facilidad de agarre. Por ejemplo, Konrad Ahlin en su trabajo "Autonomous Leaf Picking Using Deep Learning and Visual-Servoing" explica que el agarre de objetos con un brazo robótico es una tarea difícil y muestra una alternativa de solución por medio del procesamiento de imágenes con una combinación de redes neuronales convolucionales (Por sus siglas en inglés Convolutional Neuronal Network CNN). Entendiendo que el enfoque del proyecto de grado no es solucionar un problema de agarre de diferentes objetos sino netamente de, la interacción humano-

robot y como esta puede ser mejorada por medio del procesamiento de señales de vídeo y de audio se realiza esta restricción. Ver figura 2.29

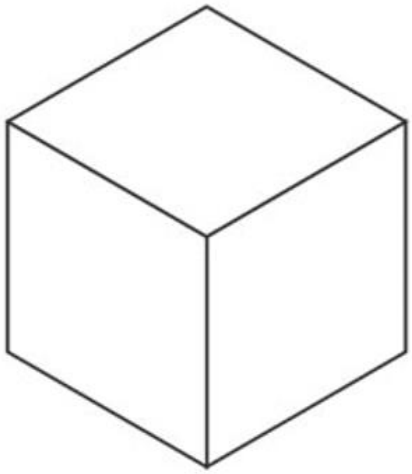


Figura 2.29: Imagen de muestra de la figura tridimensional que se va a usar.

Por último, el área de trabajo del UR3, este es un espacio limitado que es el lugar donde el cobot tiene disponibilidad de realizar las actividades que se le exijan. No se tomarán acciones por fuera de este espacio, por ejemplo, si el usuario le exige al cobot colocar una de las fichas fuera del área este no realizará la tarea. También, aplica para situaciones donde se exija recoger una ficha por fuera. El área es la representada en la figura 2.30.

Figura 2.30: Imagen de muestra del área de trabajo establecido para el UR3.

De forma general, el espacio de trabajo en conjunto con la mesa y el UR3 terminaría visualizándose de la siguiente manera, ver figura 2.31. Se resalta que, el cuadrado amarillo que se encuentra frente a la mesa es el lugar donde deberá posicionarse el usuario. El dispositivo que se presenta en el costado izquierdo de la imagen hace referencia a la cámara C920 que es usada para el reconocimiento de los objetos mencionados anteriormente. Esto con el fin, de obtener una mayor compresión del ambiente de trabajo más completo y no por separado. Ver figura 2.31.

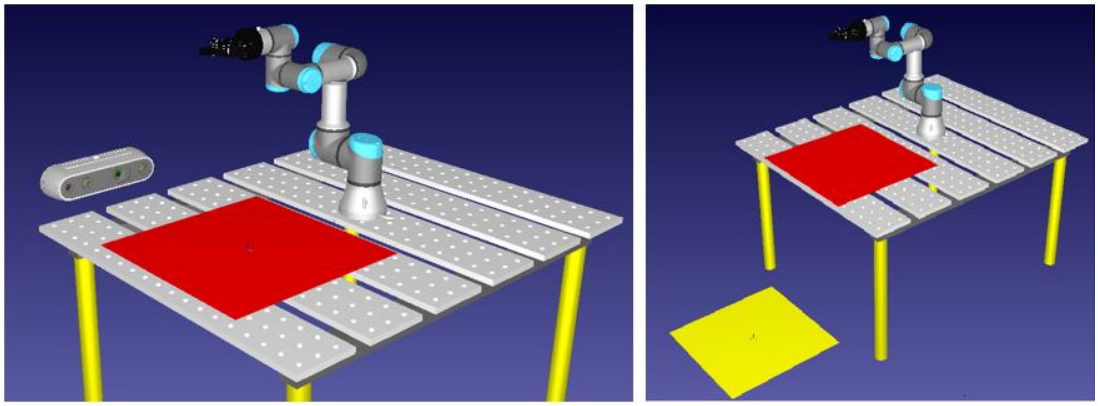


Figura 2.31: Simulación del área de trabajo del UR3 para la realización de pruebas.

2.3 Implementación

En este apartado se explicará lo implementado dentro de cada módulo del sistema en el lenguaje de programación Python, donde se hizo uso de diferentes librerías para el desarrollo de los apartados. Se comienza con el módulo de vídeo mostrando cuáles son los dispositivos elegidos para captar la señal, razones por las que se eligieron y cómo estarán ordenados en el espacio que se dispone para trabajar con el UR3. También, se explica el proceso que se llevó a cabo para tomar las decisiones de qué sería implementado como librerías y división de submódulos. Luego, se continúa con el módulo de audio donde se presenta el dispositivo elegido para la captación de dicha señal. Además, se explica la configuración del módulo como librerías utilizadas, diferentes submódulos y funciones principales. El tercer apartado presenta la revisión de comandos para la generación de etiquetas de entrada al algoritmo. El cuarto apartado evidencia la implementación del árbol de decisión como la integración de la información de las señales de vídeo y de audio. Después, se explica el funcionamiento del módulo de control en el cuarto apartado, y se finaliza con el módulo de comunicación que es el sexto apartado.

2.3.1 Módulo de vídeo

En este módulo se presenta la selección del dispositivo para la captación de la señal de vídeo. En este apartado, se presenta la forma de configuración que se realizó quedar preparado para su uso. Luego, se presenta el apartado del procesamiento de señales de vídeo, en este se muestran los diferentes módulos implementados, comparación de las librerías consultadas y cuáles fueron las elegidas, listado de librerías que son necesarias para implementar el módulo de vídeo, o lo que respecta su procesamiento de señales y las partes del código más significativas donde se evidencia en detalle la lógica que persigue para cumplir su función. Por lo tanto, se procede a la explicación de la toma de señales de vídeo.

2.3.1.1Toma de señales de vídeo (Dispositivos)

Recordando que los dispositivos para la captación de las señales de vídeo son la cámara Logitech C920 y el LEAP MOTION se presenta la explicación de cómo se configuraron y en qué lugar fueron ubicados. También, se evidencian algunos pasos que se tuvieron que realizar para tener en funcionamiento la cámara y revisar el estado del LEAP MOTION. Comenzando con la cámara Logitech C920 esta no fue comprada para realizar el trabajo de grado sino una que tenía disponible el CAP. Esta junto con el trípode se ubican al lado izquierdo del área de trabajo dispuesto para el

UR3, pero desde la perspectiva del usuario. La manera en que colocó la cámara fue horizontal. Para entender mejor mirar la figura 2.31

Figura 2.32: Imagen de muestra de la cámara C920 montada en el tripode.

La ubicación de la cámara siempre debe ser la misma, ya que de no ser así la correspondencia con lo programado en las medidas de desplazamiento estarían viendose afectadas. Una vez se tuvo el montaje de la cámara como se evidencia en la figura 2.33 se ubicó en el lado izquierdo del área de trabajo desde la perspectiva del usuario, de tal manera que la cámara estuviera viendo hacía el área de trabajo. En la mesa de trabajo se dejaron unas marcas que denotan como se debería ubicar. Ver figura 2.33.

Figura 2.33: Ubicación de la cámara C920 en la mesa de trabajo.

Luego, se conecta el cable USB a la computadora. Si fuera una computadora de mesa no habría que realizar nada más. Sin embargo, por estar trabajando en una laptop en algunas ocasiones no la reconoce como la cámara predeterminada así que hay que configurar esto. Para esto se coloca el puntero encima del icono de windows, se da click derecho, en las opciones desplegables que se ven se elige la opción de “administrador de dispositivos” buscan la cámara integrada del laptop y se

deshabilita, con esto debe ser suficiente. Adicionalmente, la cámara posee un sotfware del fabricante que se puede descargar de la pagina principal de él donde permite ver diferentes opciones de la C920.

Una vez ubicada y configurada la cámara se procede a realizar unas pequeñas pruebas donde se pueda evidenciar que estaba en el lugar correcto. Para esto, se tomó unas imágenes donde se pueda ver en su totalidad los cubos. Más adelante, esta imagen será la tomada de ejemplo para la comprabación del procesamiento del vídeo y que funcione adecuadamente. Ver figura 2.34.

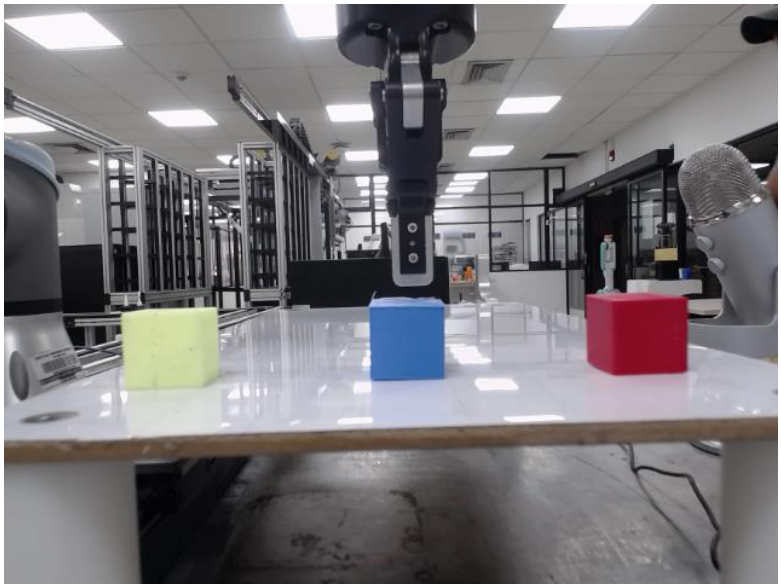


Figura 2.34: Imagen de muestra los cubos en el área de trabajo del UR3.

Terminado de instalar y ubicar la cámara C920 se prosiguío la instalación de LEAP MOTION. Este al tener las dimensiones algo pequeñas en comparación a los otros dispositivos comparados no requeria una adecuación para su uso. Este se pone en frente del área de trabajo sobre la mesa de trabajo que está soportado el UR3. Este dentro de la caja viene con dos cables a su disposición, es recomendable para este caso usar el más largo. Se conecta el puerto correspondiente con el LEAP y se deja el otro libre para ser ingresado al PC. Ver figura 2.35.

Figura 2.35: Imagen de muestra del Leap Motion

Una vez hecho esto se procede a la configuración del LEAP MOTION dentro del computador. Luego de haber descargado el Panel de Control del Leap Motion lo iniciamos y con el Leap conectado se dirige a la pestaña de resolución de problemas, aquí se aprecia el estatus del dispositivo que permite

dar seguridad que está funcionando correctamente. Si el estatus del dispositivo no es como el que se presenta a continuación no es recomendable realizar pruebas con el sistema. Para mayor claridad ver la figura 2.36.

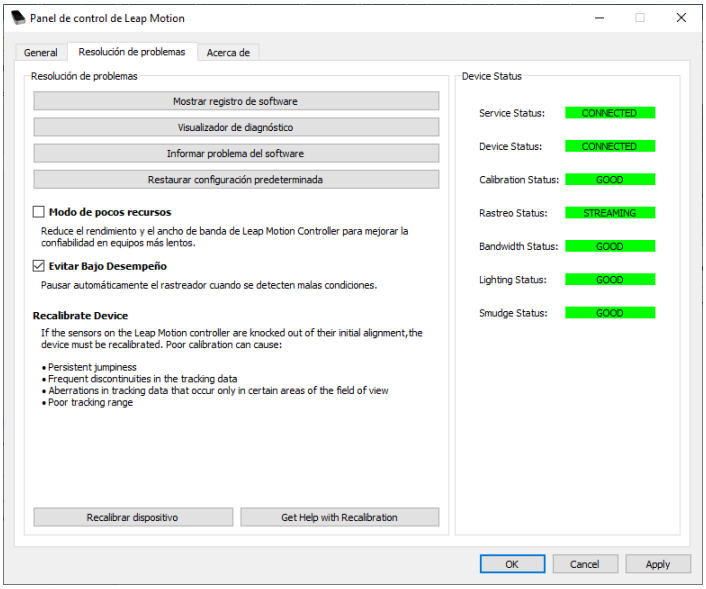


Figura 2.36: Imagen de muestra del panel de control del Leap Motion.

En resumen, la toma de señales de vídeo se realiza por medio del Leap Motion y la cámara C920. Hasta ahora se tiene la ubicación de los sensores y la configuración que se realiza en la computadora para no ir a presentar ningún tipo de inconvenientes ni errores en el desarrollo del código. Por lo tanto, se prosigue al procesamiento de las señales de vídeo donde se explica qué se hace para tomar la información que se desea de los datos captados por los dispositivos explicados.

2.3.1.2 Procesamiento de las señales de vídeo

Para comenzar de forma general se explica qué se realiza en este módulo de señales de vídeo. Primero, se explicará el procesamiento de vídeo alrededor de la captación de las características de los objetos, luego se mostrará el desarrollo de software para el reconocimiento de las manos por medio del Leap. Para el procesamiento de señales de vídeo primero se realiza la captura de estas, se obtienen los datos en formato RGB u otros de los existentes, se realiza un preprocesamiento donde se cambian de formato los datos, se hacen algunas adecuaciones, se extrae la información bajo alguna técnica o librería y se realiza una interpretación de la información obtenida. Esto se puede evidenciar mejor la figura 2.37.

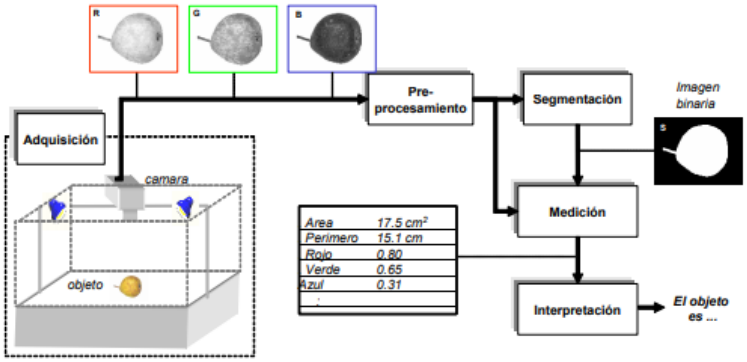


Figura 2.37: Imagen de muestra del sistema de captación y procesamiento de información de vídeo

Para realizar todo este módulo se necesitaron diferentes librerías que trabajan con Python. Se consultaron diferentes opciones como SimpleCV, Pgmagic y OpenCV. Sin embargo, se optó por OpenCV por la facilidad de programación dentro de Python y su documentación disponible en la web. Por consiguiente, es la librería primordial que se debe descargar para el procesamiento de la señal de vídeo. Por otra parte, la señal de vídeo no es un formato MP4, sino PNG, esto quiere decir que es una imagen, no un vídeo. La anterior aclaración es pertinente realizarla y está sustentada bajo la premisa que el área de trabajo y las posiciones no son variables que estén en constante cambio, por el contario estas son poco variantes en el tiempo. La única razón por la que la posición de los cubos debe cambiar es cuando el UR3 esté desplazándolos de un lugar a otro, por tal motivo, este solo debe "echar un vistazo" para saber en qué lugar está y luego tomarlo. Por lo tanto, se prefirió la captación de una imagen y no de un video.

El primer paso en la creación del archivo es importar las librerías que se van a usar dentro de este. Para el caso del reconocimiento de color objetos son OpenCV y Numpy. Con el comando *import*s se importan las librerías que van a ser usadas en el archivo. Comenzando con la cv2 que es la correspondiente a OpenCV que es la encargada del procesamiento de la imagen, luego con Numpy que nos servirá más adelante para el manejo de matrices dónde estarán alojados los datos de las imágenes o copias que debemos realizar.

Dentro de este archivo se definen tres funciones. La primera con un nombre *TomarImagen()*, luego *detectColores()* y *detectColor*. De manera general, la primera función es llamada en algunos apartados del código donde es necesario tomar una foto del espacio de trabajo, esto cumpliría la tarea de la captación de señal. En *detectColores()* se encuentra la parte fuerte del procesamiento de imagen, aquí se hace uso de tres entradas que permiten el dibujo de contornos, el cálculo del centro y así determinar el color que se encuentra en la imagen y su ubicación. Por último, *detectColor* es la función que es llamada por otros módulos a la hora del reconocimiento de color dentro del área de trabajo.

En *tomarImagen()* se toma la captura siguiendo el siguiente proceso. Se inicia la cámara por medio del comando *cv2.VideoCapture*, Luego, dentro de un *while* se toma la información del vídeo, se guarda la imagen dentro de la carpeta en la que se encuentra corriendo el programa con el comando *cv2.imwrite*, se finaliza el ciclo y se apaga la cámara.

Dentro de la función *detectColores()* posee tres entradas, mascara, color e imagFinal. Luego, de esto se inician dos variables que son el espacio donde se guardan las coordenadas del cubo. Sin embargo, se hace necesario dar una idea sobre qué es la máscara y cómo está definida para posteriormente ver cómo se implementa en la función que hace el llamado de esta y le ingresa esto como una entrada.

Las máscaras son un arreglo con la información de un color en específico que se busca reconocer dentro de una imagen. Esta información está en formato HSV, esto será explicado en mayor profundidad cuando se esté hablando de la detección del color. Sin embargo, se puede pensar la máscara como un papel en blanco que va a calcar la imagen original pero solo lee el color deseado lo cual lleva a la segunda entrada de la función que es el color. Este se entrega en formato RGB, lo que quiere decir tres números con valor mínimo de 0 y valor máximo de 255. Esto se debe a que cada color es representado con 8 bits, esto da el rango mencionado. La variación de los tres colores Red (R), Green (G) y Blue (B) permite tener una gama variada de colores. Sin embargo, el proyecto de grado está haciendo uso de los primarios, RGB. La última entrada hace referencia a la imagen captada por la cámara con un procesamiento de colores, donde se deja solo el color requerido y los demás en escalas de grises. Esto se hizo con la intención de permitir que el reconocimiento de bordes no fuera a sufrir problemas en reconocer sesiones de la imagen como un objeto cuando no lo era.

Se usa la función de OpenCV *cv2.findContours(imagen, modo, método)* que permite encontrar los contornos de los colores que estamos buscando en la imagen tomada. Esta imagen debe ser una imagen binaria a la cual se pretende encontrar los contornos. Luego, sigue la opción de modo, esta funciona por medio de jerarquía, esta determina que contorno te va retornar. El modo elegido es *cv2.RETR_EXTERNAL* que retorna el contorno mayor encontrado en la imagen binaria. El método *cv2.CHAIN_APPROX_SIMPLE* Comprime segmentos horizontales, verticales y diagonales y deja solo sus puntos finales, ahorrando memoria al no almacenar puntos redundantes. Si se eligiera la otra opción es cambiando la palabra “SIMPLE” por un “NONE” este guardaría todos los puntos del contorno, gastando memoria adicional. Se prosigue a iniciar un ciclo *for* que recorra lo guardado en la variable contornos. Esta es la aproximación hallada. Se calcula el área de dicho contorno por medio de la función *cv2.contourArea* y se evalúa el condicional donde el área deme ser mayor a 1500 para ser un área significativa y no un error. Este valor se encontró realizando varias tomas y evidenciando que era el más adecuado pues reconocía correctamente lo colores. Se usa *cv2.moments(c)* que en OpenCV, los momentos son el promedio de las intensidades de los píxeles de una imagen. Los momentos OpenCV se utilizan para describir varias propiedades de una imagen, como la intensidad de una imagen, su centroide, el área y la información sobre su orientación. Se genera un condicional para no llegar multiplicar por cero y este igualarlo a 1, para encontrar para encontrar las coordenadas en X y en Y simplemente se dividen. El restante del código son funciones que permiten dibujar el contorno sobre la imagen.

Luego, se crea un arreglo vacío donde se guardará la información recopilada, porque ya encontramos la existencia del color y sus coordenadas en X y Y. Por lo tanto, dependiendo de la entrada *Color* se agregarán las coordenadas y la letra inicial del color reconocido, luego este arreglo es retornado. Esto se repite para cada color, pero solo se presenta el caso del color R.

Antes de continuar la explicación de la forma en que funciona el módulo de reconocimiento de las características de los objetos, se debe introducir un concepto que se tuvo que aplicar para trabajar con la información de la imagen. HSV, al igual que RGB son formatos en los que el computador comprende los colores, solo que su forma de funcionar es diferente. El RGB fue explicado a grandes rasgos en uno de los párrafos anteriores. Por otra parte, HSV significa Hue, Saturation y Value, que en español sería matiz, saturación y valor. Así que, a partir de aquí, se puede presentar una semejanza con RGB y es que los rangos de valores de estas variables son de 0 a 255. La siguiente imagen ayuda a la comprensión de los cambios que puede sufrir el color que se tome dependiendo de las variables que se modifiquen en el rango permitido. Ver figura 2.38.

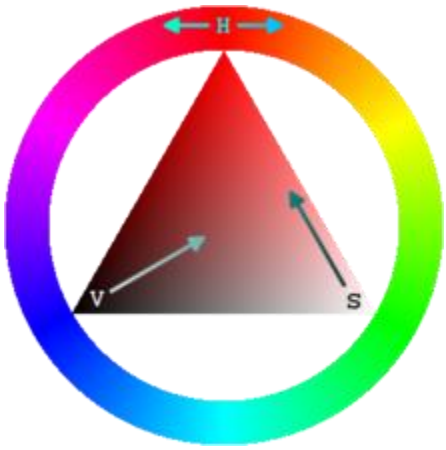


Figura 2.38: Imagen de muestra del formato de color HSV.

Conociendo como funciona HSV se debe establecer los colores que se necesitan usar para el procesamiento de la imagen. En este caso sería Rojo, Verde y Azul, recordando que estos son los colores que se permiten permanecer en la imagen. Los valores que se eligieron resultaron de realizar pruebas donde dadas las condiciones del CAP reconociera adecuadamente el color. Se comenzaron con valores de HSV cercanos al color que se requería, para comprender esto se puede visualizar las figuras 2.39 y 2.40.

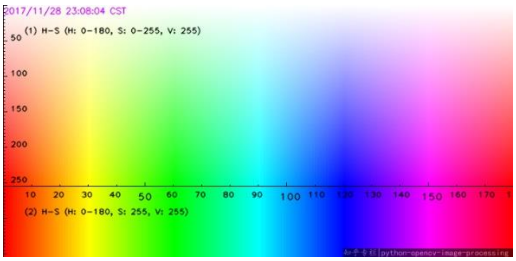


Figura 2.39: Imagen de muestra de valores HSV

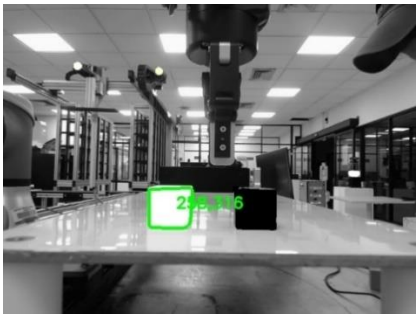


Figura 2.40: Imagen cubo verde reconocido.

Entendido esto, primero se crea un arreglo con los valores seleccionados teniendo cuando el concepto de HSV, se agrega el tipo de datos que necesitamos que en este caso es uint8 que es un entero sin signo representado en 8 bits. Esto termina dando un valor que se distribuye en el rango de [0, 255] que es justamente el requerido. Para el ejemplo se presenta la máscara de color verde pero este proceso se repite para los demás colores, cambian valores de HSV pero el resto se mantiene igual. Ver figura 2.41.

```
verdeBajo = np.array([20,50,50],np.uint8)
verdeAlto = np.array([50,255,255],np.uint8)
```

Figura 2.41: Imagen de muestra de arreglo con valores HSV.

Luego se procede a leer la imagen que se fue guardada después de la ejecución de la función *tomarImagen()*. Este está almacenada en la misma carpeta así que se usa el comando *cv2.imread*() junto con el nombre. Se usa de la librería imutils la función *resize* que permite redimensionar el tamaño de la imagen que se esté usando. Al tener la información de la imagen dentro de una variable se debe primero pasarla a escala de grises, y luego crea una imagen exactamente a la tomada por la cámara con los colores en el formato HSV.

Teniendo la imagen en formato HSV crea la máscara con el color que se quiere conocer. Para esto se hace uso de *cv2.inRange(Imagen, verdeBajo, verde Alto)*. Un punto importante que se olvidó mencionar en el apartado donde se definen los valores de HSV es que construye la máscara tomando un de inicio en el matiz que sería el Bajo y un final que es el *Alto.* En esta operación que se realiza es que queden en blanco solo los pixeles donde se reconocieron el color verde. Lo que se esperaría visualizar al imprimir esa imagen sería algo como el ejemplo de la pelota azul de la figura 2.42.

Figura 2.42: Ejemplo de una imagen binaria.

Luego, con el comando *cv2.medianBlur()* se busca mejorar la imagen disminuyendo sus componentes de ruido. Esto permite que la imagen a ser un poco más limpia permita tener una probabilidad menor de tomar valores erróneos en el reconocimiento. Para comprender de forma gráfica que realiza este comando ver la figura 2.43.

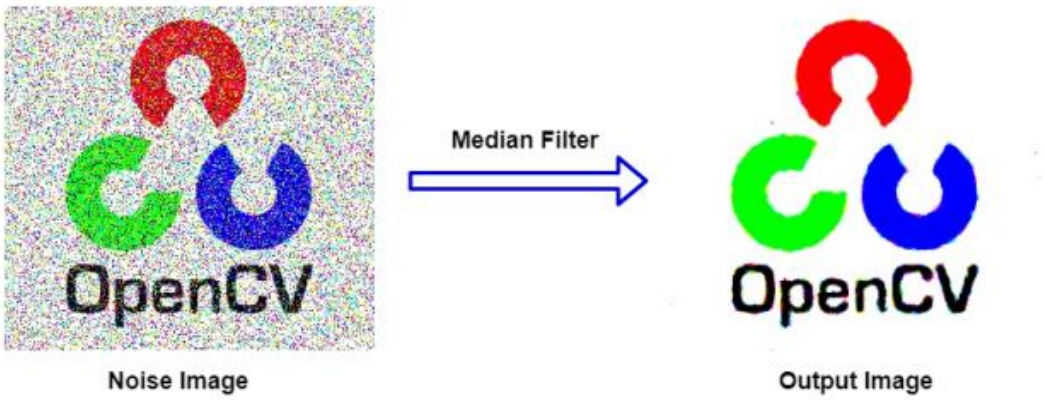


Figura 2.43: Imagen del resultado de la aplicación de la función *cv2.medianBlur().*

Se continúa creando un arreglo vacío donde se ingresarán los datos que se recopilen. Se toma la entrada de la función que se llama *color* para determinar cuál mascara se debe usar y así mismo que color detectar, para posteriormente realizar el llamado de la función de contornos que nos permite determinar la ubicación cartesiana del cubo. Por tanto, se hace uso de la función *cv2.bitwise_and* que detecta el color verde en cada pixel de la imagen que se ingresa. Se realiza el mismo proceso, pero para una imagen con el fondo en gris, buscando que el reconocimiento sea más exacto para evitar los errores donde se reconocen colores que están muy cercanos a los límites. Estas dos imágenes son adicionadas por medio del comando *cv2.add*(imag1, imag2) buscando tener una imagen más adecuada para captar la información 2D. Se llama la función *detectColores(Mascara, Color, imagFinal)* para recibir como respuesta la inicial del color y sus coordenadas. Esta nueva imagen que tiene dibujado el contorno es guardada en la carpeta del programa y esta es la respuesta que se retorna al llamado de esta función

Esto es lo que se encuentra en el archivo presentado al inicio de procesamiento de señales de vídeo. Sin embargo, cabe aclarar que se presentó las partes más relevantes de la estructurando obviando los procesos repetitivos para los otros colores. La información qué retorna es un arreglo que contiene 4 datos, la letra del color y las 3 coordenadas. Por otra parte, se puede evidenciar de qué forma se vería reconocido el color verde, para esto ver la figura 2.40.

Ahora es el turno del procesamiento de información por medio del Leap Motion. Para poder comenzar a desarrollar el código que se usa en este módulo se debió tener diferentes aspectos a consideración. Comenzando por los archivos adicionales que son librerías compatibles con Python. Estos no se integran directamente con el lenguaje, ni existe la opción de descargarla por medio del comando *pip install* en la consola del sistema. Los documentos *Leap.dll* y *Leap.lib* fueron tomadas de las carpetas donde se descargó el panel de control, se agregan a la misma carpeta donde se encuentra el código desarrollado para el proyecto. Dentro de la documentación que, tiene dispuesta los fabricantes del dispositivo se encontró un ejemplo donde hacía uso del reconocimiento de las manos. Por otra parte, Leap Motion tiene soporte para su dispositivo para ciertas versiones de

Python, en este caso la 2.7 de 32 bits, mientras que la versión que se maneja para OpenCV es la 3.6.8 de 64 bits.

Para comenzar, se importan las librerías que se van a usar a lo largo del código. La librería *os* nos va a permitir la creación de un archivo *.txt* dentro de la carpeta donde se aloja el programa. Este archivo contiene las salidas que son las entradas para el algoritmo, en otras palabras, es la información del reconocimiento de la presencia de la mano izquierda o derecha. Las librerías *Leap, sys, thread y time* son las que permiten construir la clase que nos va a permitir conectar con el Leap Motion por medio de Python. Esta clase dentro de ella tiene definida cinco funciones que permiten el uso de las funciones con este nombre dentro de los archivos agregados a la carpeta del módulo. La función de interés es la llamda *on_fram(self, controller)* que es la encargada de adquirir la información que está leyendo el Leap Motion y hacer uso de ella. Una vez llamada la función el proceso que se sigue es iniciar el controlador, tomar la información que se encuentra dentro de la función la librería y esto se hace por medio de la variable *frame.hands*. Esto fue lo que se pudo intuir de visualizar como estaba compuesto el código de dichos archivos adicionales. Luego, basta con usar un condicional que me guarde el dato sobre si la presencia que de la mano es izquierda, sino lo es seguro es la derecha. Posterior a esto se crea el archivo *.txt* con la etiqueta que luego va a ser necesaria en el siguiente módulo. Ver figura 2.44.

```
from typing import Hashable
import Leap, sys, thread, time
import os

class LeapMotionListener(Leap.Listener):

    def on_frame(self, controller):
        frame = controller.frame()
        for hand in frame.hands:
            handtype = "Left Hand" if hand.is_left else "Right Hand"
            print (handtype)

            if handtype == 'Left Hand':
                file = open('reconoManos.txt', 'w')
                file.write('L \n')


            elif handtype == 'Right Hand':
                file = open('reconoManos.txt', 'w')
                file.write('R \n')
```

Figura 2.44: Definición de las funciones e importación de librerías del módulo del Leap Motion.

La anterior figura nos permite visualizar la implementación de la clase que realiza el reconocimiento de la presencia de la mano y cuál de los dos es. Ahora, se presenta la implementación de la función denonaminada *main()* que permite hacer el llamado y uso de la clase *LeapMotionListener()*. Dentro

de esta función se inicia la clase, se define el controlador y se agrega la variable de la clase dentro del controlador. Esto son los pasos necesario para poder acceder con las funciones a los datos del Leap Motion. Luego, un condicional permite permanecer en un loop con una lectura continúa de los datos del Leap, si se recibe la interrupción del teclado este saldrá del loop y terminará el programa. Para correr el programa se debe ingresar el comando *py -2 nombreArchivo.py* recordando que este debe ser usado en una versión recomendada de Python 2.7.

Ingresando el comando recomendado se evidencia el correcto funcionamiento por el mensaje de entrada que se recibe y el mensaje de que el sensor ha sido conectado. Luego, al pasar la mano por la superficie del Leap respetando un espacio de entre 15 a 20 centímetros es reconocida la mano y se imprime cuál fue. Esto puede ser apreciado en las siguientes capturas, ver las figuras 2.45.

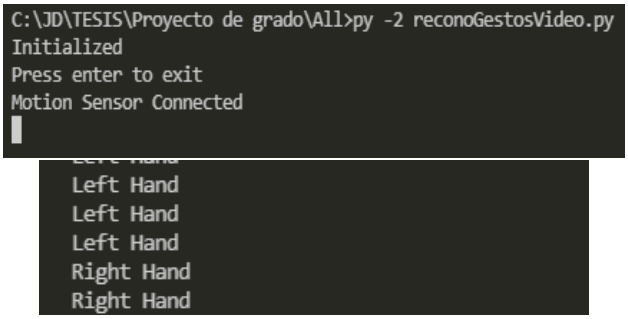


Figura 2.45: Capturas del cmd evidenciando el funcionamiento del módulo del Leap Motion.

2.3.2 Módulo de audio

En este apartado, se presenta la forma de configuración que se realizó quedar preparado y para su uso. Luego, se presenta el apartado del procesamiento de señales de audio, en este se muestran los diferentes módulos implementados, comparación de las librerías consultadas, listado de librerías que son necesarias para implementar el módulo de audio, o lo que respecta su procesamiento de señales y las partes del código más significativas donde se evidencia en detalle la lógica que persigue para cumplir su función. Por lo tanto, se procede a la explicación de la toma de señales de audio.

2.3.2.1 Toma de señales de audio (Dispositivos)

Teniendo claro que el dispositivo para la captación de la señal es el micrófono Yeti se presenta la explicación de cómo se configuró y en qué lugar fue ubicado. Este dispositivo solo contiene un cable que es el que se conecta por medio de USB al PC. Una vez conectado se elige como la entrada de sonido y se deja en las salidas de sonido los altavoces del pc. Es importante realizar esto porque luego cuando se desea reproducir sonido y el Yeti se encuentra conectado no se podrá realizar. Esto sucede debido a que el dispositivo es tomado también como salida de sonido de forma errónea en algunos casos. Ver figura 2.46.

Figura 2.46: Imagen de muestra del microfono Yeti.

Es importante recordar que se debe seleccionar la configuración cardioide del micrófono. Para esto se elige el símbolo que denota esta configuración, esta puede ser vista en el manual de configuración del micrófono. También, es necesario tener en cuenta la forma en que se posiciona el micrófono, puesto que se necesita que la fuente de sonido quede directamente en el frente del micrófono. Ver figura 2.47.

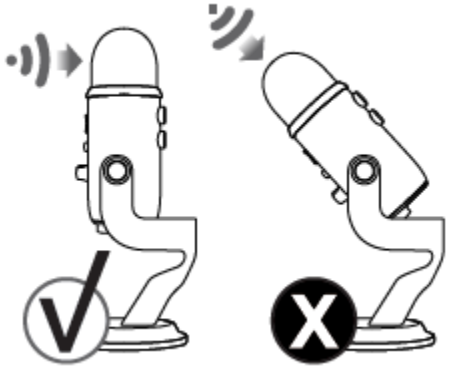


Figura 2.47: Imagen de muestra del correcto uso del micrófono Yeti.

Por otra parte, el micrófono se ubicó en frente del área de trabajo del UR3. Esto quiere decir que el usuario podrá ubicarse directamente en frente del robot y tendrá a su disposición el micrófono captando la señal como si estuviera dirigiendo la palabra el UR3. Esto resulta conveniente, después de todo da la sensación que es el robot quién escucha. Para esto ver la figura 2.48.

Figura 2.48: Imagen de muestra del lugar donde se encuentra ubicado el micrófono Yeti.

2.3.2.2 Procesamiento de la señal de audio

Primero, el título *reconoComanAudio.py* que cumple la función de tomar la señal de voz captada, extraer los datos y entregar un string con las palabras dichas por el usuario. Segundo, el archivo con nombre *sintetizador.py* que es el que permite la generación de voz para las respuestas automáticas hacía el usuario. Comenzando por el archivo *reconoComanAudio.py* se importa la librería *speech_recognition* con la abreviación *sr* para que resulte más fácil el llamado de esta. Dentro del programa se define solo una función titulada *reconocer*, se establecen variables donde se guardará la información captada por el micrófono para luego usarla. Se usa la función de *Microphone()* de la librería y se establece como la fuente principal de sonido. Posteriormente, se establecen 2 condicionales que permiten la captación del sonido para con la función *r.recognize_google*, sus entradas son el audio captado e idioma en el que se busca reconocer el habla, para este caso es *es_CO*. Si no es conocido el valor reconocido se muestra un mensaje que avisa que no entendió o sino, muestra el palabras que captó de la señal. Ver la figura 2.49.

```
import speech_recognition as sr
def reconocer(duracion):
    global r
    global source
    r = sr.Recognizer()

    with sr.Microphone() as source:

        if 'None' in duracion:
            r.adjust_for_ambient_noise(source, duration = 1)
            audio = r.listen(source, phrase_time_limit = 3 )
```

```
        else:
            r.adjust_for_ambient_noise(source, duration = 1)
            audio = r.listen(source, phrase_time_limit = int(duracion))

        try:
            response = r.recognize_google(audio, language = 'es_CO')
            print ("Entendi: " + response + '\n')
            return response

        except sr.UnknownValueError:
            return 'No te entendi'

        except sr.RequestError as e:
            print("GSR; {0}".format(e))
            return 'No logre reconocer ningun audio'
```

Figura 2.49: Imagen de muestra de función *reconocer()* del módulo de audio.

Para el archivo de *sintetizador.py* se importa la librería *pyttsx3*. Se procede a iniciar la variable *tts*, con la función *hablar()* definida se hace llamado de las funciones necesarias para que el generador de voz diga el texto ingresado. Por ejemplo, si yo le ingreso el texto "Hola mundo" eso es lo que dirá. Ver figura 2.50.

```
import pyttsx3

tts = pyttsx3.init()

def hablar(texto):
  tts.say(texto)
  tts.runAndWait()
```

Figura 2.50: Imagen de muestra del sintetizador.

2.3.3 Revisión de comandos y diccionario

Dentro de este apartado se habla sobre el módulo de las etiquetas y los comandos. Este está directamente relacionado con la implementación del código de los diccionarios de comandos en general. El funcionamiento que cumple este, es la generación de la cadena de datos que se le envía directamente al árbol de decisión donde este hará uso para su interpretación.

Para el primer archivo contiene la programación del diccionario de comandos, esto de forma general incluye los comandos de vídeo, audio y de objetos. Por lo tanto, este es el archivo que revisa el programa de *revisionComandos.py* para verificar que la información captada y entregada por los módulos de vídeo y audio si están definidos y existen dentro del programa. En la figura 2.60 se aprecia la estructura de diccionario de comandos de dirección perteneciente al diccionario de audio.

Se crea un arreglo con su nombre donde estará conformado por listas, esta guardará en la fila el comando al que corresponden y dentro de la lista los sinónimos del comando. Por esta razón, se evidencia que en la misma lista dice "adelante, frente, delante" son diferentes formas que ha sido interpretada la instrucción. Esta estructura es semejante para los demás.

En el archivo de *revisionComandos.py* se importar las librerías *diccionarioComandos, reconoComandAudio* y *reconoColorObjetos* que son los programas explicados hasta el momento donde se encuentran las funciones que nos ayudan a captar la señal de audio, la de vídeo y objetos. Se hace el llamado de las funciones ya mencionadas para captar los datos necesarios para evaluar la existencia de los comandos en el sistema. Dentro del archivo se implementan tres funciones para la revisión y confirmación de los comandos. La función *revisionComandos()* se encarga de los comandos de audio, *revisionGestos()* es la función que revisa los comandos de vídeo y *revisionColorObjetos()* las características de los objetos.

2.3.4 Árbol de decisión – Algoritmo implementado

Dentro de la carpeta del proyecto de grado se encuentra el archivo titulado *ArbolDecision.py* en este se encuentra la implementación completa de lo que fue presentado como el algoritmo elegido. Este tiene como entradas las salidas directas del módulo de revisión de comandos que fue explicado anteriormente. Con dichas entradas el algoritmo trabaja para generar las señales de salida necesarias para enviar al módulo de control y luego comunicarlo al UR3.

Dentro de este módulo se realiza el llamado de varias funciones para la articulación de información necesaria para el correcto funcionamiento. Se comienza con la que genera sus entradas que es *revisionComandos,* luego se hace uso del *sintetizador* para generar respuestas automáticas en algunos casos, *random* para la generación de números aleatorios, *controlUR3* para el envío de las señales de control y la ejecución de las acciones necesarias por el UR3, *time* para poder implementar tiempos de delay dentro del código, *comunicacionUR3* para el envío de información de control del UR3 y finalmente el archivo *reconoColorObjetos* para el reconocimiento de los objetos o el uso de alguna función dentro de él.

Una parte significativa del marco teórico se dedicó a hablar sobre HRI, Interacción humano robot. Pensando en los casos que se presentan o se podrían presentar dentro del desarrollo de las pruebas se establecieron respuestas que dependiendo del comando que se esté ejecutando la respuesta irá en coherencia con este. Esto con el fin de implementar un canal de comunicación por parte del UR3 con el usuario. Esto permite que la comunicación sea bidireccional donde se le exige al robot y este puede responderle exigiendo que le repita lo dicho. Por ejemplo, en la función *respuestaUsuario* se hace uso del programa *sintetizador* para en los momentos de ser llamada la interfaz pueda responder. Algunas de las respuestas que se presentan aquí son orientados al servicio, esperando ordenes o instrucciones para seguir. Por mencionar una de las respuestas predeterminadas se tiene "Espero la siguiente instrucción" o "¿En qué más te puedo ayudar?". Son cinco funciones diferentes con respuestas orientadas al contexto.

La lógica de esta función es bastante simple. Se genera un número aleatorio, este valor dentro de un rango definido por el número de respuestas existentes, la elegida hace el llamado del *sintetizador* y su función de *hablar* y el computador dice con voz de mujer lo que se lee entre los paréntesis.

Por otra parte, se presentan funciones adicionales que permiten solucionar uno de los puntos importantes del procesamiento de vídeo. Cuando se estaba en el proceso de la implementación no se lograba imaginar de que formar se iba a realizar una correspondencia espacial con lo captado por la cámara y la ubicación del movimiento que debería alcanzar el UR3. Por ejemplo, el módulo de reconocimiento de objetos me dice que el cubo está en la ubicación (300, 345, 200), entendiendo que son los eje X,Y y Z está ubicación hace referencia a la distribución de pixeles de la imagen captada, no del área de trabajo. Por lo tanto, es necesario que el algoritmo determine cuál es la distancia necesaria a desplazarse para alcanzar la ubicación adecuada para recoger o agarrar el cubo reconocido. Para la implementación de la solución a este problema se definieron dos funciones que fueron *cuantoMoverme()* y *cualColor()*. Por medio de la *cualColor()*, se elige el color para determinar en qué posición del vector de información deberá buscar los datos del color *y cuantoMoverme()* asigna el valor que debe desplazarse.

En el ejercicio que se realizó para solucionar la correspondencia de estas medidas se eligió un cubo de cualquier color y se tomó nota de la ubicación en pixeles que tenía en un cuadrado en la mesa de trabajo. Luego, se movió al siguiente cuadrado y se calculó cuánto varió la ubicación respecto a la anterior. Esto se repitió múltiples veces hasta que se terminaron definiendo ciertos rangos de aceptación donde el funcionamiento era el adecuado. Ver figura 2.51.

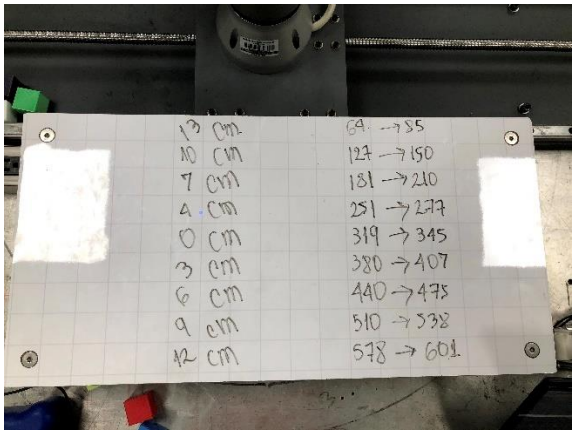


Figura 2.51: Imagen del área de trabajo donde se realizaron medidas de distancia para el movimiento del UR3.

Teniendo presente esto se explica el proceso que se lleva dentro de las funciones *cuantoMoverme()* y *cualColor()*. Se llama a la función *cualColor(),* esta con los argumentos de entrada el color determina qué coordenadas van a ser usadas para llamar la función *cuantoMoverme().* La función entra a evaluar cuál es la cantidad que debe desplazarse y en qué dirección para llegar a la ubicación correcta donde debe recoger un objeto. Por otra parte, las funciones principales son cuatro, estas están determinadas por los diccionarios de audio disponibles.

Las funciones principales en estructura en esencia son semejantes, pero atienden casos distintos, la diferencia recae en el tipo de acción o significado que posee la información que manejan. Por ejemplo, la función *comandosMovimiento()* tiene 3 entradas que hacen referencia a las señales de gestos, objetos y audio. Dentro de esta función se evalúan las diferentes opciones que presenta, como es el comando de moverse en una dirección, agarrar un objeto, soltarlo, parar, etc.

Por último, la función *arbolDecision()* que posee las mismas 3 entradas de las demás funciones principales cumple un papel importante. Este funciona a manera de orquestador, toma las señales de llegada, evalúa que caso se está presentando, qué tipo de comando es llamado primero, y basado en eso marca el inicio de la llamada función para tomar los pasos siguientes a realizar. Cabe resaltar, que la siguiente figura no contiene el código en su totalidad pues se presente solo la estructura más significativa.

2.3.5 Control

Dentro de este apartado se explicará de forma general el funcionamiento del código implementado para la tarea de control del UR3. Para realizar el control son usados seis archivos que trabajan en conjunto para cumplir con las tareas necesarias. El desarrollado es el titulado *controlUR3.py* el resto son librerías tomadas de repositorios de código. Como en todos los códigos anteriores se comienza importando las librerías necesarias para la implementación. Librerías como *sintetizador y time* ya han sido mencionadas y explicadas anteriormente. La librería *comunicacionUR3* usa el protocolo TCP para comunicarse con el UR3, esta será explicada en un apartado siguiente a este. Por otra parte, la librería *socket* nos facilitará el objeto nombrado para que la comunicación sea posible. Ver figura 2.52.

```
import time
import comunicacionUR3 as COM
import socket
import sintetizador
```

Figura 2.52: Importación de las librerías necesarias para el archivo de control del UR3.

Dentro del programa *controlUR3.py* se definen cinco funciones que permiten la realización de una acción diferente cada una. Sus nombres dan una breve idea de qué acción se puede realizar. Por ejemplo, el llamado de la función *move()* permite el desplazamiento a una posición x,y,z. Por otra parte, las funciones terminadas en la palabra Gripper son acciones que se pueden realizar con esta herramienta.

El socket que se crea para la comunicación con el robot se usa para enviarle al UR3 comandos del lenguaje de programación URScript, que es el lenguaje creado por Universal Robots para controlar el robot UR3. Se debe hacer este proceso ya que los comandos del lenguaje URScript no los puede compilar un computador, entonces se debe hacer un programa en py para luego hacer comunicación vía socket y enviar los comandos de movimiento tipo URScript. Los comandos del lenguaje URScript se pueden encontrar en el siguiente enlace: https://s3-eu-west-1.amazonaws.com/ur-support-site/32554/scriptManual-3.5.4.pdf

Algo importante a destacar es que los URScript son comandos que solo competen a funcionalidades del robot, movimientos, giros, manejo libre o freedrive, etc. Es decir, que para usar los dispositivos adicionales, el gripper, la cámara o el sensor de torque y fuerza por medio de código, no es posible con los URScript ya que estos dispositivos o accesorios son adicionales al robot y son marca Robotiq. Por consiguiente, para el manejo del gripper o la pinza se usan 4 archivos para las acciones de activar, abrir, cerrar y posición media del gripper, en estos se encuentran diferentes instrucciones que sirven para comandar el gripper, estos archivos se envían vía socket en forma binaria.

2.3.6 Comunicación

Por último, el módulo de comunicación. Este módulo representa el puente o el medio por el cuál el UR3 las instrucciones que el algoritmo le está avisando el usuario requiere.

Para el control del UR3 se usa comunicación por socket, el cual es un proceso servidor-cliente basado en un conjunto de protocolos TCP/IP, para establecer comunicación con el robot se debe saber la IP del robot y el puerto de comunicación disponible para la conexión. Las funciones implementadas aquí principalmente funcionan para transportar comandos que estén definidos dentro de los URScripts, para la comunicación de las acciones del Gripper se hace directamente en la función en el módulo de control.

2.4 Evaluación

En este apartado se va a presentar el caso de estudio que se tomó a consideración para evaluar el rendimiento del sistema implementado. Se presentan las métricas de evaluación cuantitativas que permite analizar el rendimiento general que hemos tenido en la aplicación de las pruebas. Por último, se presentan las métricas de evaluación cualitativa que nos presenta una postura no numérica sino de cualidades del algoritmo desde la perspectiva del usuario.

2.4.1 Caso de estudio

Las pruebas constaran de llevar a cabo una actividad de "Pick up and put down" como se ha mencionado a lo largo del documento. Esta consiste en tomar un objeto de una posición inicial y colocarla en una posición final. De otra forma, esta labor es la realizada por ejemplo en industrias donde manejan un volumen alto de mercancía y deben organizarlas para la distribución, lo que es conocido como paletizado, donde sólo consiste en recoger una caja de un punto A y colocarla en el punto B que es la base donde se están organizando o apilando, ver figura 2.73. Dentro de este proceso se tomarán en cuenta diferentes parámetros que serán analizados como lo son la precisión, la velocidad, el número de intentos y el número de instrucciones dadas. Con esto, además de lograr comprobar la hipótesis propuesta también se busca medir el rendimiento y capacidades del algoritmo desarrollado en el proyecto de grado.

Figura 2.53: Acción a realizar en las pruebas.

En esta primera prueba el usuario le deberá pedir por medio de los comandos audio al UR3 que mueva el cubo con el color de su preferencia al punto A. En este primer caso, se hace necesario solo el uso de los comandos de audio y ninguno del de vídeo para luego posteriormente poder comparar cuál entrega mejores condiciones para comandar el robot. Ver figura 2.74.

Figura 2.54: Imagen de ejemplo de la prueba 1.

En la segunda prueba el usuario deberá exigirle al UR3 que mueva dos objetos de diferentes colores a posiciones A y X que uno de las posiciones es conocida por el UR3 y la otra es desconocida para el robot. En este caso el usuario si puede hacer uso de los comandos de objetos y gestos. Ver figura 2.75

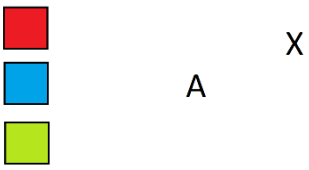


Figura 2.55: Imagen de ejemplo de la prueba 2.

Dentro de la prueba existen condiciones que no son variables dentro de ella y por el contrario deben mantenerse fijas. Una de estas es el lugar donde se recogen los cubos. Este está predispuesto como la cuarta columna de cuadros comenzando de la izquierda hacía la derecha en el área de trabajo. Las posiciones que conoce el UR3 que están dentro de los diccionarios no son modificables.

2.4.2 Métrica de evaluación cuantitativa

Estas métricas tienen un carácter numérico, esto permite hacer uso de medidas como porcentajes, promedios y otras variables estadísticas que permiten realizar diferentes análisis o definir supuestos e hipótesis. Las variables seleccionadas son las consideradas como necesarias para tener un primer acercamiento sobre la interacción humano robot, y comprender si el rendimiento y posibles mejoras que se le deben realizar al algoritmo implementado. A continuación, se mencionan las variables que pertenecen a esta métrica.

La precisión, esta será medida en dos momentos diferentes, el pick up y el put down. Para el primer momento que es el “pick up” se verá de qué forma agarra el objeto, se definirá un agarre ideal para este tipo de objetos ya que todos serán iguales. En base en el agarre ideal se medirá la precisión con la que agarra en el primer momento. En el segundo momento que es el “put down” se medirá la distancia entre el punto que el UR3 posiciona la ficha y el punto real que se ha pedido. Dado el caso la distancia sea igual cero se tomará esto como un caso de éxito completo.

El tiempo se medirá en tres momentos. Primero, en el pick up cuánto tarda desde el momento que recibe la orden hasta el momento en que agarra el cubo. Segundo, el put down que comienza desde el momento donde agarró el objeto hasta que termina colocándolo en la posición pedida por el usuario. Tercero, es la suma de ambos tiempos diferentes para tener el tiempo que se demoró en realizar toda la tarea exigida.

El número de intentos es uno de los puntos más importantes de la medición, ya que de requerir más de un intento para realizar cualquiera de las tareas propuestas por el usuario la interacción humano-robot se vería afectada. Por ejemplo, si usted estuviera sosteniendo una conversación con un compañero de trabajo y usted tiene que repetir 3 o 4 veces el mensaje que desea transmitir la comunicación que existe entre ambos sería poco efectiva. Lo mismo ocurre en el caso del experimento, en el caso ideal se debería completar la tarea propuesta por el usuario en un solo intento.

Para finalizar, el número de instrucciones se evaluará aspectos como el número de detalles o de información que el usuario debió propiciarle al UR3 para completar la tarea. Por ejemplo, si el UR3 se le pide que mueva un cubo de color rojo a un punto especifico se espera que lo haga sin tener que adicionar detalles como un poco más hacía la izquierda, derecha, arriba, etc. Por otra parte, este aspecto también influye en la situación presentada del *número de intentos*. Si un usuario debe entregar muchos detalles al robot para cumplir adecuadamente la tarea también se vería afectada y no sería muy efectiva. Por consiguiente, es uno de los aspectos principales a evaluar.

En el momento de llevar a cabo el experimento se dividirá dos partes. La primera parte se llevará a cabo esta actividad solo haciendo uso de los comandos de audio. En la segunda parte se podrán usar ambos comandos, tanto el de voz como de vídeo. Se espera después de realizar el experimento que el hacer uso de los comandos de audio y vídeo hayan presentado un mejor rendimiento. Ambos casos deberán ser evaluados con la misma métrica para ser comparados adecuadamente. Cuando se menciona "métrica" solo es el conjunto de variables mencionadas anteriormente dentro de una tabla, esto permite presentar la información de manera más ordenada y entendible. Ver tabla 2.9.

Métrica			
Parámetros	Medibles		
Precisión	Distancia del punto de referencia a donde fue colocada la ficha	¿Agarró correctamente el objeto?	¿Recogió la ficha correcta?
Tiempo	¿Cuánto se demoró haciendo toda la tarea?	¿Cuánto se demoró en el pick up?	¿Cuánto se demoró en el put down?
Número de intentos	Número de intento para el pick up	Número de intentos para el put down	Número total de intentos
Número de instrucciones	Número de instrucciones para el pick up	Número de instrucciones para el put down	Número total de instrucciones

Tabla 2.9: Variables de la métrica de evaluación cuantitativa.

2.4.3 Métrica de evaluación cualitativa

En la evaluación cualitativa de las pruebas existe una diferencia notable en la forma de calificar. La evaluación cuantitativa presentaba un enfoque de funcionalidad respecto al rendimiento que tendría el algoritmo desarrollado para el manejo del UR3. Sin embargo, en este apartado no se evalúa netamente el algoritmo sino el sistema de manera más general. También, que a diferencia

de la evaluación cuantitativa que se hace uso de variables que pueden tomar valores numéricos en la evaluación cualitativa las variables son dicotómicas, esto quiere decir que se elige una de las opciones disponibles dependiendo de la pregunta. Esto puede sonar algo confuso, pero visualizando la tabla 2.10 será más entendible.

Preguntas\Opciones	Opción 1	Opción 2
Pregunta 1		
Pregunta 2		
Pregunta 3		
Pregunta 4		

Tabla 2.10: Imagen de ejemplo de la métrica cualitativa.

En el experimento se presentan tres pruebas donde se exige realizar unas tareas específicas. Cada una de estas pruebas fue desarrollada primero haciendo uso solo de comandos de audio y una segunda ocasión con comando de audio y vídeo. Estos dos escenarios son la representación de la opción 1 y la opción 2 que se muestran en la tabla anterior. Ahora bien, las preguntas no presentarán un enfoque de funcionalidad como ya se había mencionado. Por el contrario, irán dirigidas a evaluar la experiencia del usuario con ambas condiciones de trabajo buscando que ellos respondan con cuál opción se sintieron más cómodos, fue mejor su interacción con el robot, cuál les entregó mayor facilidad para llevar a cabo la actividad y con cuál creen que se demorarían menos a la hora de hacer actividades más complejas. Por ejemplo, la tabla 2.11 para la evaluación de las pruebas sería la siguiente.

Preguntas\Opciones	Prueba 1 haciendo uso de los comandos de AUDIO	Prueba 1 haciendo uso de los comandos de AUDIO y VÍDEO
¿Cuál de las dos opciones te entregó mayor comodidad para desarrollar la actividad?		
¿Cuál de las dos opciones te permitió interactuar de la mejor forma con el robot UR3?		
¿Cuál de las dos opciones te entregó mayor facilidad de control para llevar a cabo la actividad?		
¿Con cuál de las dos opciones crees que te demorarías menos en realizar una actividad con un grado de dificultad mayor?		

Tabla 2.11: Métrica de evaluación cualitativa.

Capítulo III

Análisis y conclusiones

Dentro de este capítulo se expone los resultados obtenidos en las pruebas realizadas en el capítulo anterior. Luego, se procede a realizar un análisis de evidenciado para determinar si los objetivos planteados al inicio del proyecto fueron alcanzados en sus totalidad o si por el contrario no fueron cumplidos. Después, se continúa con las conclusiones obtenidas y se finaliza con los trabajos futuros propuestos que giran en torno a esta línea de investigación.

3.1 Análisis y resultados

Para la realización de las pruebas se contó con algunos inconvenientes que fueron difíciles de manejar como lo fue la poca afluencia de personas en la universidad por causa de la pandemia propiciada por la Covid-19, adicional a esto en las fechas que se habían previsto las pruebas se presentó el paro nacional, por tal razón para las pruebas se contó con tan solo 7 personas. A estas personas se les explicaba antes de comenzar con las pruebas en qué consistía, se les mostraba por medio de una corta presentación qué debían hacer y al final se les dejaba proyectado los comandos disponibles.

Tomando como ejemplo la prueba aplicada en Sara, una estudiante de Ingeniería Mecánica, se mostrará cuál fue el proceso de llenado de las tablas. Luego, se procede a presentar los resultados en una tabla resumida. Con esto, se procede a realizar la discusión de los resultados y analizar los datos tomados. Para comenzar, una vez explicado a Sara lo que se debía realizar ella prosiguió a comenzar el desarrollo de las pruebas. Con un metro se midió la distancia en la que terminó el objeto luego de finalizar la tarea. Se evidencio que el UR3 agarró el objeto de manera adecuada y tomp la ficha correcta. Respecto al tiempo, el total fueron 6 minutos, 4 en recoger la ficha y 2 minutos en dejarla en el lugar correspondiente. Sin embargo, se realiza la observación que el tiempo que transcurrió hablando con el UR3 era menor que estos 6 minutos, se observó que tomaba un poco de tiempo cómo hablarle al robot. También, en repetidas ocasiones se debía hablar rápido y fuerte para que el módulo de audio lograra captar la señal adecuadamente. Ver tabla 3.1.

Parámetros	Medibles		
Precisión	Distancia del punto de referencia a donde fue colocada la ficha	¿Agarró correctamente el objeto?	¿Recogió la ficha correcta?
	1.1 cm	SI	SI
Tiempo	¿Cuánto se demoró haciendo toda la tarea?	¿Cuánto se demoró en el pick up?	¿Cuánto se demoró en el put down?
	6 min	4 min	2 min
Número de intentos	Número de intento para el pick up	Número de intentos para el put down	Número total de intentos
	12	6	18
Número de instrucciones	Número de instrucciones para el pick	Número de instrucciones para el put down	Número total de instrucciones
	4	4	8

Tabla 3.1: Métrica cuantitativa aplicada al usuario.

La misma métrica fue aplicada en la prueba 2 para los dos casos diferentes que existen en esa prueba. Por otra parte, la métrica cualitativa era la última en aplicarse. Se le realizaban las preguntas al usuario buscando que respondiera basado en la experiencia que acababa de vivenciar con el manejo del UR3. Para el caso de Sara las respuestas y las preguntas se pueden evidenciar en la tabla 3.2.

Preguntas\Opciones	Prueba haciendo uso de comandos de AUDIO	Prueba haciendo uso de los comandos de AUDIO y VÍDEO
¿Cuál de las dos opciones te entregó mayor comodidad para desarrollar la actividad?		x
¿Cuál de las dos opciones te permitió interactuar de la mejor forma con el robot UR3?		x
¿Cuál de las dos opciones te entregó mayor facilidad de control para llevar a cabo la actividad?		x
¿Con cuál de las dos opciones crees que te demorarías menos en realizar una actividad con un grado de dificultad mayor?		x

Tabla 3.2: Métrica cualitativa aplicada al usuario.

Este proceso fue realizado con los siete usuarios mencionados al inicio. Con cada uno se recopilaron datos diferentes y opiniones que aportaron una perspectiva nueva al trabajo. A continuación, se presenta la tabla con todos los datos resumidos. Para la primera prueba aplicada, donde se debía usar solo los comandos de audio se evidencio que en promedio se tardaban casi 3 minutos recogiendo el cubo y 2 minutos colocándolo en su lugar. Por otra parte, el número de intentos para

recoger el cubo fueron 11, realizando la aclaración que esto se debida a que en repetidas ocasiones el módulo de vídeo no reconocía correctamente el habla, 7 intentos colocando el cubo ese es el promedio de esta prueba. Se manejó en promedio una precisión de 1,07 cm de diferencia respecto a la ubicación donde se debería encontrar. Por último, en promedio se usaron 5 instrucciones para realizar el pick y 4 para realizar el put. El resumen de los datos puede ser visualizado en la tabla 3.3.

Prueba 1							
Usuario	Precisión	Tiempo		Numero intentos		Numero instrucciones	
		Pick	Put	Pick	Put	Pick	Put
Johan	1	3	1	9	4	4	4
José	1.1	2	2	9	8	3	5
Laura	1.4	3	2	14	6	4	4
Sara	1.1	4	2	9	6	4	4
Nicole	1	2	3	11	5	5	4
Sebastian	0.9	2	1	13	8	7	4
Daniel	1	3	1	12	9	6	7

Tabla 3.3: Datos resumidos de la prueba 1.

Para la segunda prueba aplicada, donde se podía hacer uso de los comandos de audio y vídeo. Haciendo referencia al ejercicio donde la ubicación no era conocida por el UR3 se evidencio que en promedio se tardaban casi 0.5 minutos recogiendo el cubo y 2 minutos colocándolo en su lugar. Por otra parte, el número de intentos en promedio para recoger el cubo fueron 3 y 6 intentos colocando el cubo. Se manejó una precisión en promedio de 0,53 cm de diferencia respecto a la ubicación donde se debería encontrar. Por último, en promedio se usaron 1 instrucción para realizar el pick y 4 para realizar el put. El resumen de los datos puede ser visualizado en la tabla 3.4.

Prueba 2 - Ubicación 1							
Usuario	Precisión	Tiempo		Número intentos		Número instrucciones	
		Pick	Put	Pick	Put	Pick	Put
Johan	0	0.6	2	2	6	1	4
José	0.9	0.5	1	2	4	1	4
Laura	0.6	0.4	3	4	7	1	6
Sara	0.8	0.5	2	3	8	1	5
Nicole	0	0.5	3	4	5	1	4
Sebastian	0.3	0.6	1	5	3	1	3
Daniel	0.7	0.4	2	2	6	1	4

Tabla 3.4: Datos resumidos de la prueba 2 con la ubicación 1.

Para la segunda prueba aplicada, donde se podía hacer uso de los comandos de audio y vídeo. Haciendo referencia al ejercicio donde la ubicación si era conocida por el UR3 se evidenció que en promedio se tardaban casi 0.7 minutos recogiendo el cubo y 1.05 minutos colocándolo en su lugar. Por otra parte, el número de intentos en promedio para recoger el cubo fueron 3 y 4 intentos colocando el cubo. Se manejó en promedio una precisión de 0,61 cm de diferencia respecto a la ubicación donde se debería encontrar. Por último, en promedio se usaron una instrucción para realizar el pick y una para realizar el put.

Este es el punto más positivo que se evidencia dentro de una comparación entre la prueba 2 con una ubicación desconocida donde hay que guiar al robot y la segunda es una ubicación que el conoce. Tener 4 instrucciones y pasar 1 es un punto positivo a usar la captación de vídeo y una posición conocida. Sin embargo, no es muy determinante la medida de precisión, pues los movimientos de dejar el objeto son dependientes del usuario. Esto muestra que debería existir un gran desarrollo en el manejo del control con precisión del UR3. Esto sustentado desde el hecho que el UR3 es un robot con capacidad de repetitividad que maneja rangos de ±0.1mm. Puesto que, el valor de la precisión tomado en la prueba es demasiado alto en comparación con las capacidades de UR3, la diferencia es de 1 cm cuando debería optar los rangos de los milímetros. El resumen de los datos puede ser visualizado en la tabla 3.5.

Prueba 2 - Ubicación 2							
Usuario	Precisión	Tiempo		Numero intentos		Numero instrucciones	
		Pick	Put	Pick	Put	Pick	Put
Johan	0.3	1.5	0.5	4	1	1	1
José	1	0.5	0.5	1	1	1	1
Laura	0	0.5	1.5	3	4	1	1
Sara	0.4	0.5	1.5	3	4	1	1
Nicole	0.7	0.5	1	2	6	1	1
Sebastián	0.9	0.6	1.4	4	5	1	1
Daniel	1	0.8	1	3	7	1	1

Tabla 3.5: Datos resumidos de la prueba 2 con la ubicación 2.

En la comparación del promedio total de cada variable en las diferentes pruebas realizadas permiten evidenciar ciertos aspectos positivos de la integración del vídeo con el audio. Una disminución en el error de la precisión, disminución en el tiempo de ejecución de las tareas, disminución en el número de intentos y una reducción del número de instrucciones usadas para llevar a cabo la tarea. Se permite realizar esta comparación entendiendo que son el mismo ejercicio con diferentes herramientas adicionales para la interacción humano robot. La información puede verse en la tabla 3.6.

Promedio del total de las variables medidas en la métrica cuantitativa			
Variable	Prueba 1	Prueba 2 - Ubicación 1	Prueba 2 - Ubicación 2
Precisión	1,1	0,5	0,6
Tiempo	4,4	2,5	1,8
Número intentos	17,6	8,7	6,9
Número instrucciones	9,3	5,3	2,0

Tabla 3.6: Promedio de las variables en las diferentes pruebas aplicadas.

Estudiando cada variable en particular, la cantidad de instrucciones e intentos usados se ve reducido conforme se adicionan herramientas al usuario. El salto más significativo en la reducción del número de intentos fue en la adición de la señal de vídeo, esto comparando las gráficas de prueba 1 y las prueba 2 – Ubicación 1. Se puede apreciar que existe una disminución en la cantidad de intentos de hablarle al robot, esto es un 61% para la prueba final. Esto pudo verse debido a diferentes factores, mayor silencio o el usuario con las anteriores dos pruebas comprendieron mejor en qué forma hablarle. Además, también se evidencia una disminución en la cantidad de instrucciones usadas para pedir la tarea requerida en un 79%. Por otra parte, en la prueba 1 el reconocimiento del habla fue correcto en un 53%, en la prueba 2 en la ubicación 1 se reconoció el habla correctamente en un 61%. Mientras que, en la última prueba se reconoció el habla solo en un 29%. Mirar la figura 3.1.

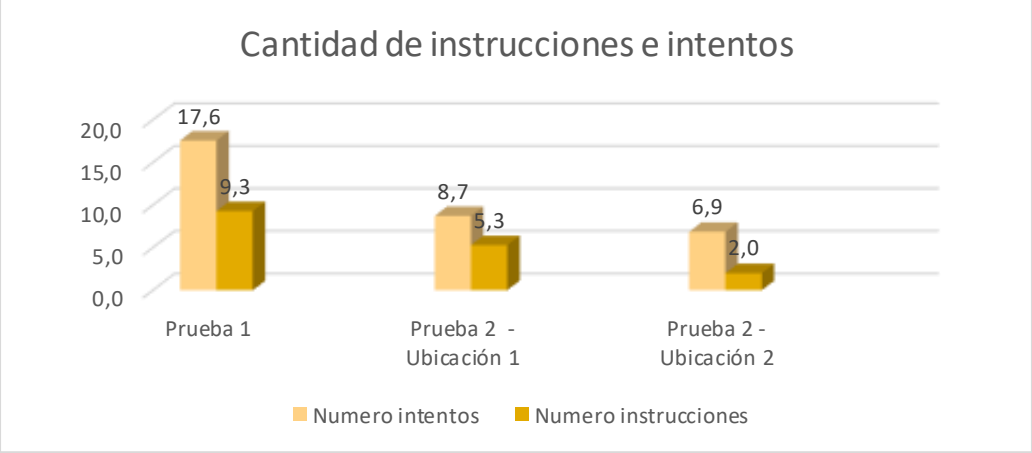


Figura 3.1: Comparación de cantidad de instrucciones e intentos en las pruebas.

Evaluando la variable de precisión se debe comenzar realizando una aclaración. Esta medida entre mayor sea el valor peor situación se está presentando. Esta medida de precisión es tomada como la diferencia entre el punto ideal y el punto real donde se ubicó el cubo. Por lo tanto, tener una precisión de 1 cm es bastante negativo. Sin embargo, cuando se le adicionan las herramientas de vídeo esta precisión mejora en un 46%. Esto se puede evidenciar en la figura 3.2.

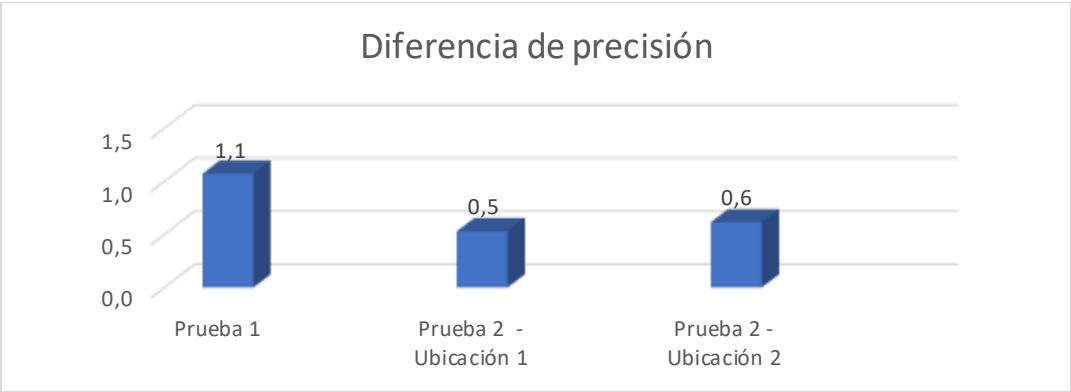


Figura 3.2: Comparación de la diferencia de precisión en las pruebas.

Por último, en cuanto a las métricas cuantitativas, se evidenció que el tiempo de realización de las pruebas disminuyó considerablemente. A pesar que, en la prueba 2 se debía mover dos cubos el tiempo disminuyó en un 59% en comparación con el tiempo inicial que se tenía en la prueba 1. Ver figura 3.3.

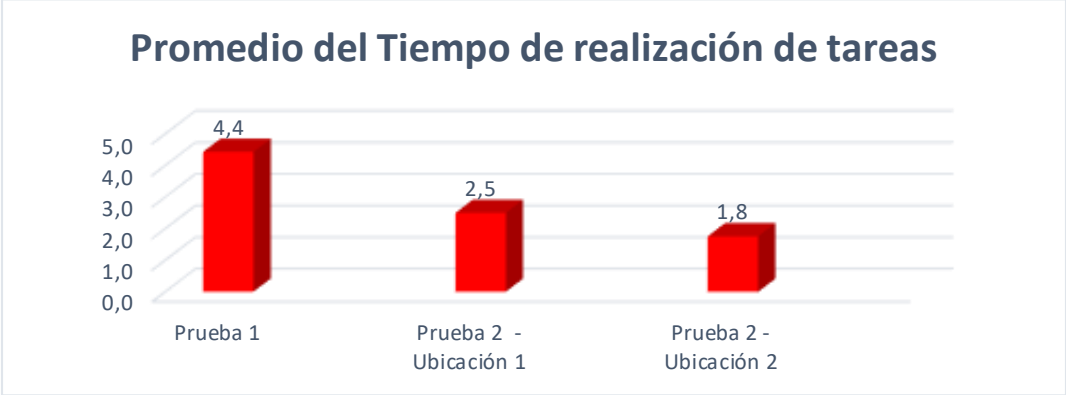


Figura 3.4: Comparación del promedio del tiempo tomado para las pruebas.

Por otra parte, las métricas cualitativas que reflejan la opinión y la experiencia del usuario, o ese fue el sentido que se estaba persiguiendo cuando se diseñaron las preguntas, fueron aplicadas justo después de terminar las pruebas. Las siete personas eligieron los comandos audiovisuales por encima de solo comandos de audio. Sin embargo, debido a que la muestra no es lo suficiente significativa frente a la población de la Universidad Javeriana no se puede determinar con certeza que esto sea completamente cierto. De igual forma, es un acercamiento positivo. Ver los datos resumidos en la tabla 3.7.

Preguntas	AUDIO	AUDIO Y VÍDEO
¿Cuál de las dos opciones te entregó mayor comodidad para desarrollar la actividad?	0	7
¿Cuál de las dos opciones te permitió interactuar de la mejor forma con el robot UR3?	0	7
¿Cuál de las dos opciones te entregó mayor facilidad de control para llevar a cabo la actividad?	0	7
¿Con cuál de las dos opciones crees que te demorarías menos en realizar una actividad con un grado de dificultad mayor?	0	7

Tabla 3.7: Datos resumidos de la métrica cualitativa.

3.2 Conclusiones

En este proyecto de grado se implementó un algoritmo de fusión de señales que permitió el reconocimiento de comandos de audio y vídeo. Este algoritmo fue usado dentro de una interfaz donde en conjunto con los módulos de procesamiento de vídeo y de audio se logró el objetivo. La interfaz está en la capacidad de recibir comandos audiovisuales soportados en un diccionario de comandos de vídeo, audio y objetos. Esto permitió dirigir al UR3 a realizar acciones específicas para cumplir una tarea de desplazamiento de ubicación de un cubo. Donde la clasificación y selección del cubo está basada en la característica visual de su color.

De la revisión bibliográfica consultada se evidenció que era necesario el procesamiento de la señales para la extracción de las características e información importante. Al tener la una mayor cantidad de comandos el canal de audio fue el principal para la comunicación de comandos al UR3. Sin embargo, este presentaba problemas para el reconocimiento de las palabras, lo cual sembró la idea de ser necesario un desarrollo más robusto y fuerte alrededor del reconocimiento automático del habla. Continuando, la implementación y la aplicación de las dos pruebas con el UR3 y los siete usuarios dejaron las siguientes conclusiones.

- Primeramente, el porcentaje del reconocimiento del habla es necesario mejorarlo para aumentar la calidad de la interacción humano robot, esto debido a que el porcentaje más bajo se le atribuye a la prueba 2 con ubicación conocida por el UR3 donde se obtuvo un reconocimiento del 29%. Durante la ejecución de las pruebas se logró evidenciar que se veía entorpecida la experiencia por las repetidas ocasiones donde el módulo de audio no reconocía correctamente el habla. Se especula que, la situación empore para los escenarios donde el ambiente en el que se desarrollan las pruebas existan varias fuentes de ruido.

- La diminución de las instrucciones que se usan para completar una tarea de desplazamiento cuando se hace uso de señales de audio y vídeo es del 79% en comparación a la situación

donde solo se usan señales de audio. Al guiar al robot haciendo uso netamente de comandos de audio resulta consumir más tiempo que cuando se le instruye y este contiene información visual. Por lo tanto, el uso de la información de vídeo termina siendo un factor positivo para la interacción humano robot.

- El 100% de las personas prefirió el manejo del robot UR3 por medio de comandos audiovisuales. Dentro de los argumentos expresados por algunos usuarios, se explicaba que permitía hablar de forma más natural con el robot, ya que podías mencionar cosas que se encuentran en el tablero de trabajo del UR3. Sin embargo, también se sugirió poder realizar la acción de señalar para ingresar coordenadas por medio de este gesto.

- La precisión es una variable que tuvo en rendimiento bastante bajo, se esperaba valores entre 0.2 y 0.5 cm de diferencia en comparación a la posición inicial donde debería ser puesta la ficha. Sin embargo, se evidenció un caso donde el promedio de las medidas tomadas en las pruebas duplica estos valores que fue 1.1 cm de diferencia. En conclusión, este es un aspecto necesario a mejorar en la programación en la tarea de posicionar el objeto en la ubicación deseada.

3.3 Trabajos futuros

Los trabajos futuros irían encaminados a potenciar el reconocimiento del habla y a los sistemas de diálogos que puede incluir la interfaz para mejorar su interacción con el usuario. Se puede generar un nuevo modelo de dialogo integrando técnicas de procesamiento del lenguaje natural, como también una integración de librerías en el programa que faciliten el planteamiento de nuevos comandos de voz, formando oraciones con variaciones en los sinónimos que se pueden usar para la conformación de cada comando de voz.

Como trabajo futuro cercano se podría pensar en la implementación de diferenciar en el reconocimiento de objetos por medio de visón artificial. Este permitiría explorar que grado de diferenciación y exactitud puede manejar. Además, que se ve extendido el campo de aplicación en el que se puede desarrollar pruebas con el UR3.

Para trabajos futuros con mayor contenido de desarrollo se pondría pensar en la implementación de chatbots que permitan tener una interacción más cercana con los usuarios, esto podría ser uno de los primeros trabajos que aporte al desarrollo de HRI dentro de la carrera de ingeniería. También, el aprendizaje de tareas por medio del aprendizaje de máquinas. Este podría ser un servicio aprovechado desde la nube, donde también podría estar disponible el programa para ser implementado en cualquier UR3 que tenga acceso a la nube. Sin embargo, todas estas opciones requieren de una inversión enorme de investigación y esfuerzo para llevarlas a cabo.

Anexos

[1]

[2] https://drive.google.com/drive/folders/10noppvKpBRxzNWYBAgXmV2x_Bg3PioBy?usp=sharing

Bibliografía

[1] Enrique Caro Márquez et al. La cuarta revolución industrial. *Depósito de investigación Universidad de Sevilla*, 2017.

[2] Hyun-Don Kim, Jong-Suk Choi, and Munsang Kim. Human-robot interaction in real environments by audio-visual integration. *International Journal of Control, Automation, and Systems*, 5(1):61–69, 2007.

[3] Zengxi Pan, Joseph Polden, Nathan Larkin, Stephen Van Duin, and John Norrish. Recent progress on programming methods for industrial robots. In *ISR 2010 (41st International Symposium on Robotics) and ROBOTIK 2010 (6th German Conference on Robotics)*, pages 1–8. VDE, 2010.

[4] Young, J. E., Sung, J. Y., Voida, A., Sharlin, E., Igarashi, T., Christensen, H. I., & Grinter, R. E. (2010). Evaluating Human-Robot Interaction. International Journal of Social Robotics, 3(1), 53–67. https://doi.org/10.1007/s12369-010-0081-8

[5] Veton Kepuska and Gamal Bohouta. Next-generation of virtual personal assistants (Microsoft cortana, apple siri, amazon alexa and google home). *In 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC),* pages 99–103. IEEE, 2018.

[6] Hoy, M. B. (2018). Alexa, Siri, Cortana, and More: An Introduction to Voice Assistants. Medical Reference Services Quarterly, 37(1), 81–88. https://doi.org/10.1080/02763869.2018.1404391

[7] Retto, J. (2017). Sophia, first citizen robot of the world. ResearchGate, URL: https://www. researchgate. net.

[8] De Santis, A., Siciliano, B., De Luca, A. y Bicchi, A. (2008). An atlas of physical human–robot interaction. Mechanism and Machine Theory, 43(3), 253–270. https://doi.org/10.1016/j.mechmachtheory.2007.03.003

[9] Pandey, A. K. y Gelin, R. (2018b). A Mass-Produced Sociable Humanoid Robot: Pepper: The First Machine of Its Kind. IEEE Robotics & Automation Magazine, 25(3), 40–48. https://doi.org/10.1109/mra.2018.2833157

[10] Kazuyoshi, W., Yousuke, I., Kaoru, I. y Reona, U. (2010). Development and preliminary evaluation of a caregiver's manual for robot therapy using the therapeutic seal robot Paro. IEEE, (19th International Symposium in Robot and Human Interactive Communication), Artículo 10.1109/ROMAN.2010.5598615.

[11] Díaz Boladeras, M., Andrés, A., Casacuberta Bagó, J., & Angulo Bahón, C. (2011). Propuesta metodológica para la evaluación de la interacción persona-robot en diversos escenarios de aplicación. In ROBOT 2011: robótica experimental: libro de actas (pp. 617-621).

[12] Exam AZ-900: Microsoft Azure Fundamentals - Learn. (s. f.). Developer tools, technical documentation and coding examples | Microsoft Docs. https://docs.microsoft.com/en-us/learn/certifications/exams/az-900

[13] van Pinxteren, M. M. E., Wetzels, R. W. H., Rüger, J., Pluymaekers, M. y Wetzels, M. (2019). Trust in humanoid robots: implications for services marketing. *Journal of Services Marketing*, *33*(4), 507 518. https://doi.org/10.1108/jsm-01-2018-0045

[14] Irena Maravic and Martin Vetterli. Sampling and reconstruction of signals with finite rate of innovation in the presence of noise. IEEE Transactions on Signal Processing, 53(8):2788–2805, 2005.

[15] Harry Nyquist. Certain topics in telegraph transmission theory. Transactions of the American Institute of Electrical Engineers, 47(2):617–644, 1928.

[16] Lewis, M., & MTSA, S. (1997). A-Law and mu-Law Companding Implementations Using the TMS320C54x.

[17] Hugo L Rufiner and Diego H Milone. , 15(28):151–177, 2004.

[18] Micrófonos Blue - Yeti. (s. f.). Blue. https://www.bluemic.com/es-es/products/yeti/

[19] Jinyu Li, Li Deng, Reinhold Haeb-Umbach, and Yifan Gong. Introduction. In Robust Automatic Speech Recognition, pages 1–7. Elsevier, 2016.

[20] Alexandros Tsilfidis, Iosif Mporas, John Mourjopoulos, and Nikos Fakotakis. Automatic speech recognition performance in different room acoustic environments with and without dereverberation preprocessing. Computer Speech & Language, 27(1):380–395, January 2013.

[21] A. Waibel, P. Geutner, L.M. Tomokiyo, T. Schultz, and M. Woszczyna. Multilinguality in speech and spoken language systems. Proceedings of the IEEE, 88(8):1297–1313, August 2000.

[22] S. Wermter and V. Weber. Interactive spoken-language processing in a hybrid connectionist system. Computer, 29(7):65–74, July 1996.

[23] R. Cole, L. Hirschman, L. Atlas, M. Beckman, A. Biermann, M. Bush, M. Clements, L. Cohen, O. Garcia, B. Hanson, H. Hermansky, S. Levinson, K. McKeown, N. Morgan, D.G. Novick, M. Ostendorf, S. Oviatt, P. Price, H. Silverman, J. Spiitz, A.Waibel, C.Weinstein, S. and V. Zue. The challenge of spoken language systems: Research directions for the nineties. IEEE Transactions on Speech and Audio Processing, 3(1):1–21, 1995.

[24] Douglas O'Shaughnessy. Invited paper: Automatic speech recognition: History, methods and challenges. Pattern Recognition, 41(10):2965–2979, October 2008.

[25] Ramón López-Cózar and Zoraida Callejas. ASR post-correction for spoken dialogue systems based on semantic, syntactic, lexical and contextual information. Speech Communication, 50(8-9):745–766, August 2008.

[26] Wei-Lin Wu, Ru-Zhan Lu, Jian-Yong Duan, Hui Liu, Feng Gao, and Yu-Quan Chen. Spoken language understanding using weakly supervised learning. Computer Speech & Language, 24(2):358–382, April 2010.

[27] Ramón López-Cózar, Zoraida Callejas, and David Griol. Using knowledge of misunderstandings to increase the robustness of spoken dialogue systems. Knowledge-Based Systems, 23(5):471– 485, July 2010.

[28] VALENCIA, Jesús, et al. Detección de infracciones y matrículas en motocicletas, mediante visión artificial, aplicado a Sistemas Inteligentes de Transporte. Revista Ibérica de Sistemas e Tecnologias de Informação, 2020, no 37, p. 1-15.

[29] MIRANDA, Juan Carlos. Clasificación automática de naranjas por tamaño y por defectos utilizando técnicas de visión por computadora. 2018.

[30] REDOLFI, Javier A., et al. Clasificación de variedades de semillas de trigo usando visión por computadora. En VIII Congreso Argentino de AgroInformática (CAI-2016)-JAIIO 45 (Tres de Febrero, 2016). 2016.

[31] MONTOYA HOLGUIN, Christian; CORTÉS OSORIO, Jimmy Alexander; CHAVES OSORIO, José Andrés. Sistema automático de reconocimiento de frutas basado en visión por computador. Ingeniare. Revista chilena de ingeniería, 2014, vol. 22, no 4, p. 504-516.

[32] VARGAS, Nestor Andrés González. Sistema de visión por computadora para la medición de distancia e inclinación de obstáculos para robots móviles. Ingeniería y universidad, 2005, vol. 9, no 2, p. 0.

[33] Kinect: desarrollo de aplicaciones de Windows. (s. f.). Microsoft Developer. https://developer.microsoft.com/es-es/windows/kinect/

[34] Welcome to Python.org. (s. f.). Python.org. https://www.python.org/

[35] Jing Guang Han, John Dalton, Brian Vaughan, Catharine Oertel, Ciaran Dougherty, Céline De Looze, and Nick Campbell. Collecting multi-modal data of human-robot interaction. In 2011 2nd International Conference on Cognitive Infocommunications (CogInfoCom), pages 1–4. IEEE, 2011.

[36] Snejana Pleshkova and Zahari Zahariev. Development of system model for audio visual control of mobile robots with voice and gesture commands. In 2017 40th International Spring Seminar on Electronics Technology (ISSE). IEEE, May 2017.

[37] Juan D. S. Ortega, Patrick Cardinal, and Alessandro L. Koerich. Emotion recognition using fusion of audio and video features. In 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC). IEEE, October 2019.

[38] Amir Aly and Adriana Tapus. An integrated model of speech to arm gestures mapping in human-robot interaction. IFAC Proceedings Volumes, 45(6):817–822, May 2012.

[39] Hill, R. K. (2015). What an Algorithm Is. Philosophy & Technology, 29(1), 35–59. https://doi.org/10.1007/s13347-014-0184-5

[40] ¿A QUÉ EQUIVALEN 65 DECIBELIOS? (s. f.). ALLPE - Empresa de Medio Ambiente - Acústica - Topografía - Ingeniería. https://www.allpe.com/acustica/ingenieria-acustica/mediciones-acusticas/a-que-equivalen-los-diferentes-niveles-de-decibelios/a-que-equivalen-65-decibelios/

[41] Konrad Ahlin, Benjamin Joffe, Ai-Ping Hu, Gary McMurray, and Nader Sadegh. Autonomous leaf picking using deep learning and visual-servoing. IFAC-PapersOnLine, 49(16):177–183, 2016.