DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA E INSTRUMENTACIÓN

PROYECTO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA E INSTRUMENTACIÓN

TEMA: "SISTEMA DE CONTROL DE BRAZO ROBÓTICO MEDIANTE ONDAS CEREBRALES DESARROLLADO EN SOFTWARE LIBRE PARA ASISTENCIA A PERSONAS CON CAPACIDADES ESPECIALES".

AUTORES: LEONARDO ANTONIO SOLÍS CÓRDOVA

JORGE ANDRÉS TAPIA HERRERA

DIRECTOR: PHD. VÍCTOR ANDALUZ

LATACUNGA

2015

ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA E INSTRUMENTACIÓN

CERTIFICACIÓN

Certifico que el trabajo de titulación, **"SISTEMA DE CONTROL DE BRAZO ROBÓTICO MEDIANTE ONDAS CEREBRALES DESARROLLADO EN SOFTWARE LIBRE PARA ASISTENCIA A PERSONAS CON CAPACIDADES ESPECIALES"** realizado por los señores **Leonardo Antonio Solís Córdova** y **Jorge Andrés Tapia Herrera**, ha sido revisado en su totalidad y analizado por el software anti-plagio, el mismo cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, por lo tanto me permito acreditarlo y autorizar a los señores **Leonardo Antonio Solís Córdova** y **Jorge Andrés Tapia Herrera** para que lo sustenten públicamente.

Latacunga, 03 de Diciembre del 2015

PhD. Víctor Hugo Andaluz Ortiz

DIRECTOR

ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA E INSTRUMENTACIÓN

AUTORÍA DE RESPONSABILIDAD

Nosotros, **Leonardo Antonio Solís Córdova**, con cédula de identidad N°050358732-1 y **Jorge Andrés Tapia Herrera**, con cédula de identidad N°050372729-9, declaramos que este trabajo de titulación "**SISTEMA DE CONTROL DE BRAZO ROBÓTICO MEDIANTE ONDAS CEREBRALES DESARROLLADO EN SOFTWARE LIBRE PARA ASISTENCIA A PERSONAS CON CAPACIDADES ESPECIALES**" ha sido desarrollado considerando los métodos de investigación existentes, así como también se ha respetado los derechos intelectuales de terceros considerándose en las citas bibliográficas.

Consecuentemente declaramos que este trabajo es de nuestra autoría, en virtud de ello nos declaramos responsables del contenido, veracidad y alcance de la investigación mencionada.

Latacunga, 03 de Diciembre del 2015

Leonardo Antonio Solís Córdova

C.C.: 050358732-1

Jorge Andrés Tapia Herrera

C.C.: 050372729-9

ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA E INSTRUMENTACIÓN

AUTORIZACIÓN

Nosotros, **Leonardo Antonio Solís Córdova** y **Jorge Andrés Tapia Herrera**, autorizamos a la Universidad de las Fuerzas Armadas ESPE publicar en la biblioteca Virtual de la institución el presente trabajo de titulación "**SISTEMA DE CONTROL DE BRAZO ROBÓTICO MEDIANTE ONDAS CEREBRALES DESARROLLADO EN SOFTWARE LIBRE PARA ASISTENCIA A PERSONAS CON CAPACIDADES ESPECIALES**" cuyo contenido, ideas y criterios son de nuestra autoría y responsabilidad.

Latacunga, 03 de Diciembre del 2015

Leonardo Antonio Solís Córdova
C.C.: 050358732-1

Jorge Andrés Tapia Herrera
C.C.: 050372729-9

DEDICATORIA

Dedico este proyecto a mi primo Daniel Arcos, espero que desde el cielo me apoyes en mi siguiente meta.

Leonardo

AGRADECIMIENTO

En primer lugar agradezco a Dios, por darme la vida y una gran familia.

A mi padre Miguel y a mi querida madre Mercedes, que siempre me han apoyado a lo largo de mi vida a pesar de todas las dificultades.

Leonardo

DEDICATORIA

Dedico este proyecto principalmente a Dios por brindarme de toda la fuerza necesaria para salir adelante en frente de las adversidades.

A mis padres, Jorge y Carmita por brindarme su apoyo incondicional y confianza, a mis hermanos Diego y Melany, por brindarme su comprensión cuando lo necesitaba.

Andrés

AGRADECIMIENTO

Agradezco a Dios por ser parte fundamental en mi vida y brindarme de su ayuda cuando más lo necesitaba.

Agradezco a mis padres, Jorge y Carmita por su comprensión en cada una de las decisiones que he tomado, a mis hermanos por todo el apoyo brindado.

Andrés

ÍNDICE DE CONTENIDOS

CAPÍTULO IV

4. PRUEBAS Y ANÁLISIS DE RESULTADOS

CAPÍTULO V

5. CONCLUSIONES Y RECOMENDACIONES

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

El presente proyecto describe el desarrollo y la implementación de un algoritmo de control desarrollado en lenguaje de programación de software libre como lo es Python, mediante el cual se crea una interfaz gráfica amigable con el usuario que permite ejecutar diferentes tareas establecidas para el brazo robótico, ya sea mediante la ejecución de trayectorias predeterminadas como realizar una circunferencia o una silla de montar con diferentes valores tanto de altura como de radio, adicionalmente se presenta la opción de controlar al brazo robótico mediante el uso de un joystick, el mismo que está configurado para variar la posición del extremo operativo del robot en los ejes X, Y, Z. Para la implementación del algoritmo de control del brazo robótico se utiliza el modelamiento cinemático directo, el cual mediante una serie de cálculos matemáticos permite ubicar en el espacio tridimensional al extremo operativo del robot. El algoritmo también se enfoca en responder a las señales emitidas por los dieciséis electrodos del casco Emotiv EPOC, permitiendo así a una persona con capacidades especiales simular el movimiento de una de sus extremidades superiores mediante un brazo robótico. Al establecer una interfaz invasiva cerebro máquina, el algoritmo de control desarrollado permite al brazo robótico cumplir con las órdenes establecidas por el usuario.

PALABRAS CLAVE:

- **ELECTROENCEFALOGRAMA**
- **PRÓTESIS ROBÓTICAS**
- **REDES NEURONALES**
- **EMOTIV EPOC**
- **SOFTWARE PHYTON**

ABSTRACT

This project describes the development and implementation of a control algorithm developed in the programming language of free software as it is Python, whereby a user-friendly graphical interface is created with the user can perform different tasks set for the robot arm, either by executing predetermined as making a circle or a saddle with different values in both height and radio paths, further the option of controlling the robot arm by using a joystick is shown, it is configured to varying the position of the operating end of the robot in the X, Y, Z. To implement the control algorithm of the robotic arm kinematics direct modeling, which, through a series of mathematical calculations can be placed in three-dimensional space to the operating end of the robot is used. The algorithm also focuses on responding to signals from the sixteen electrodes Emotiv EPOC hull, allowing a person with special abilities simulate the movement of his upper extremities by a robotic arm. By establishing an invasive brain machine interface, the control algorithm allows the robot arm developed comply with the orders set by the user.

KEYWORDS:

- **ELECTROENCEPHALOGRAM**
- **ROBOTIC PROSTHESIS**
- **NEURAL NETWORKS**
- **EMOTIV EPOC**
- **PYTHON SOFTWARE**

CAPÍTULO I

1. GENERALIDADES

1.1 Objetivo General

Realizar el diseño e implementación de un sistema de control de brazo robótico mediante ondas cerebrales desarrollado en software libre para asistencia a personas con capacidades especiales.

1.2 Objetivos Específicos

- Estudiar el funcionamiento de la cinemática del brazo robótico de seis grados de libertad, así como también la utilización del casco Emotiv EPOC para la captación de las señales cerebrales.
- Desarrollar en Python una plataforma que permita reconocer y adquirir las diferentes señales cerebrales proporcionadas por el casco Emotiv EPOC.
- Desarrollar un algoritmo de control que tenga como entradas las señales cerebrales emitidas por el casco Emotiv EPOC para controlar al brazo robótico de seis grados de libertad.
- Realizar pruebas de funcionamiento y verificar el correcto movimiento del brazo robótico.

1.3 Justificación e importancia

La importancia del presente proyecto consiste en ofrecer a una persona con alguna discapacidad física una innovadora alternativa a la hora de realizar en gran parte de una manera independiente sus tareas cotidianas, dando como resultado un aumento en la calidad de vida que lleva cada una de los individuos antes mencionados.

El movimiento del brazo robótico mediante ondas cerebrales tiene un gran campo de desarrollo y al iniciar el trabajo se quiere incursionar en el mismo, ya que con esto se puede ayudar a muchas personas no solo en la ciudad de Latacunga sino también en el país. Además al ser un proyecto implementado

en software libre su costo disminuye considerablemente en comparación a los dispositivos que se encuentran en el mercado actual para asistencia a personas discapacitadas, debido a que no existe costo alguno en la instalación de las licencias del software. El casco Emotiv EPOC permite la obtención del electroencefalograma generado por el cerebro humano utilizando tecnología de punta dentro del campo de la medicina, con lo que cada persona podrá colocarse el casco anteriormente mencionado con mucha facilidad y sin ningún problema.

Los trabajos que tienen relación para realizar este proyecto son: Diseño del sistema de control basado en software libre para un brazo robótico de 6 grados de libertad con funcionalidad de mecanizado y paletizado para el laboratorio de electrónicos de la Universidad de las Fuerzas Armadas - ESPE Extensión Latacunga (Rivas et al., 2015). Implementación de una interface cerebro - computador para la detección de posición con la ayuda de las señales (Ponce Jurado, 2014). Con el objetivo de desarrollar proyectos futuros afines con la robótica y el software libre se permitirá a la comunidad politécnica acceder a la información teórica y práctica implementada dentro del presente proyecto.

1.4 Introducción

Los avances tecnológicos de robótica en el campo de la discapacidad humana ha surgido por la necesidad de facilitar y mejorar la ejecución de tareas cotidianas llevadas a cabo por personas que sufren algún tipo de discapacidad, donde la asistencia de un prototipo robótico automatizado pudiera ser la solución para dichas tareas. El conocimiento previo para el entendimiento e implementación de un prototipo robótico se basa en tres campos fundamentales: la electrónica, la programación y la mecánica.

La Universidad de las Fuerzas Armadas ESPE Extensión Latacunga ha desarrollado diversos proyectos dentro del mundo de la robótica gracias a la implementación y adquisición de varios prototipos robóticos en su mayoría

brazos robóticos; sin embargo pocos son los proyectos planteados que se enfocan en la asistencia a personas discapacitadas.

Debido a las limitadas aplicaciones de la robótica para personas discapacitadas físicamente, se propone implementar una Interfaz Cerebro Computadora (BCI) basado en software libre para el control de un brazo robótico construido con servos de marca Dynamixel de seis grados de libertad previamente implementado en el laboratorio de circuitos electrónicos de la universidad de las fuerzas armadas – ESPE extensión Latacunga (Rivas et al., 2015); con lo cual la ESPE tendría la oportunidad de implantar robots asistenciales diseñados y construidos en el país con tecnología actual y de bajo costo.

1.5 Electroencefalografía

La electroencefalografía es el registro de la actividad eléctrica generada por las neuronas del cerebro, que se obtienen mediante la utilización de electrodos ubicados en la superficie del cuero cabelludo, en la Figura 1 se puede observar un tipo de casco elástico que se utiliza para obtener las señales cerebrales por medio de electrodos superficiales.

Figura 1: Casco para Electroencefalografía

Fuente: (Medical Computer System, 2013)

La captación de las ondas del cerebro posee formas muy complejas que varían mucho con la localización de los electrodos y entre individuos. Esto es debido a la gran cantidad de interconexiones que poseen las neuronas y la estructura no uniforme que presenta el encéfalo.

1.6 Historia del EEG

En 1870, fue una guerra la que permitió estudiar y explorar el cerebro humano por primera vez, Fritsch y Hitzig (Calzada Reyes, 2004), médicos militares del ejército prusiano, notaron que al estimular, mediante corriente galvánica, diferentes áreas laterales de cerebros descubiertos se producían movimientos en el lado opuesto del cuerpo. Cinco años más tarde Richard Catón (Calzada Reyes, 2004), joven médico inglés, registró una actividad eléctrica oscilante y continua procedente del cerebro de conejos y monos, ratificando que el cerebro es capaz de producir corrientes eléctricas.

Ferrier (Calzada Reyes, 2004), continuando en la misma línea experimento con corriente farádica, consiguiendo resultados suficientemente fiables para finales del siglo en donde indicaba que el cerebro de animales poseía propiedades eléctricas comparables a las encontradas en el nervio y en el músculo. En 1913, Prawdwicz-Neminski (Calzada Reyes, 2004) registró lo que llamó «electrocerebrograma» de un perro, siendo el primero en tratar de catalogar dichas observaciones. Cabe aclarar que todos estos experimentos realizados se los hacían sobre cerebros descubiertos, ya que los cambios eléctricos presentados en el cerebro eran muy pequeños y al no tener un sistema de amplificación era muy difícil captar las señales cerebrales.

En 1928 Hans Berger (Calzada Reyes, 2004) psiquiatra y neurólogo alemán, desarrollo un método que podía aportar para la investigación de la actividad cerebral, descubriendo lo que se conoció como «ritmo de Berger». Debido a la falta de conocimientos técnicos en el área, pasaron algunos años para que le dieran la importancia debida a lo que Berger había investigado. En 1929, la opción de electroencefalografía clínica se discutía por vez primera en una reunión en el Laboratorio central de Patología del Hospital Maudsley

de Londres. Un grupo de investigadores intentaba registrar el «ritmo de Berger» usando amplificadores para poder mejorar la señal, sin embargo los estudios del cerebro así como los estudios realizados por Berger no se los tomaba en serio.

En 1934, Adrian y Matthews (Calzada Reyes, 2004) comprobaron por primera vez el «Ritmo de Berger» mediante una exposición pública ante un auditorio británico en una reunión de la Sociedad de Fisiología, en Cambridge. Berger mediante los progresos que había realizado Adrian, avanzó hasta donde sus pocos conocimientos técnicos le permitían, donde observó por ejemplo que si una persona abría los ojos o resolvía algún problema de manera mental existía una alteración en el ritmo amplio y regular.

Posterior a ello fue verificado por Adrian y Matthews que al contar con mayores conocimientos científicos y técnicos los llevo a realizar muchos más avances, demostrando que el ritmo regular y amplio de diez ciclos por segundo no pertenecía a todo el cerebro si no a las áreas visuales de asociación. Posteriormente la electropatología del cerebro creció en importancia. Se avanzó mucho en este campo, comenzando a interesar, entre los investigadores del EEG, el estudio de la epilepsia y otras enfermedades mentales, poniéndose de relieve la complejidad del tema y la imposibilidad de aislamiento de funciones simples, siendo necesario estudiar al cerebro como un órgano total.

1.6.1 Captación del EEG

La actividad eléctrica que se manifiesta en el cerebro puede obtenerse mediante diferentes procedimientos:

- Sobre el cuero cabelludo.
- En la base del cráneo.
- En el cerebro expuesto.
- En localizaciones cerebrales profundas.

Para captar las señales de EEG se tienen varios tipos de electrodos:

- **Electrodos superficiales:** Se aplican sobre el cuero cabelludo.
- **Electrodos basales:** Este tipo electrodos van ubicados sobre la base del cráneo sin ser necesario un procedimiento quirúrgico.
- **Electrodos quirúrgicos:** Para la aplicación de estos electrodos es necesario de la cirugía y pueden ser corticales o intracerebrales.

La exploración de la actividad bioeléctrica cerebral recibe distintos nombres según la forma de captación:

- **Electroencefalograma (EEG):** Cuando se utilizan electrodos de superficie.
- **Electrocorticograma (ECoG):** Cuando en el proceso es preciso utilizar electrodos quirúrgicos en la superficie de la corteza cerebral.
- **Estéreo Electroencefalograma (E-EEG):** Cuando se utilizan electrodos quirúrgicos de aplicación profunda.

1.6.2 Tipos de electrodos

a) Adheridos: Son pequeños discos metálicos de 5 mm de diámetro. Se adhieren con pasta conductora y se fijan con colodión que es un tipo de aislante. Aplicados correctamente dan resistencias de contacto muy bajas (1-2 kilo ohmios).

b) De contacto: Son pequeños tubos de plata los que van roscados a soportes de plástico. Para realizar un buen contacto las almohadillas son humedecidas con solución conductora, se sostienen a la superficie del cráneo con bandas elásticas y son conectadas con pinzas que tienen forma de cocodrilo. Estos son de colocación muy fácil pero resultan incomodas para la persona, en la Figura 2 se puede observar el esquema de un electrodo de contacto así como su colocación.

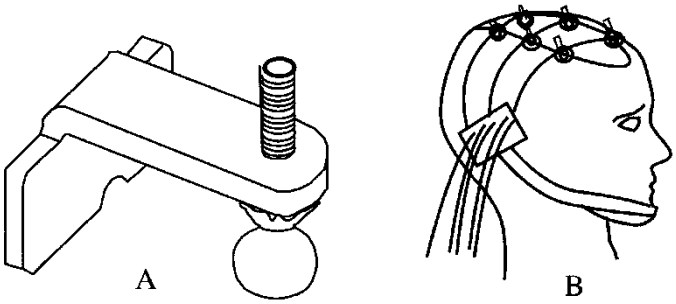


Figura 2: A) Esquema de un electrodo de contacto B) Colocación de los electrodos de contacto

Fuente: (Sánchez, 2014)

c) Casco de malla: Los electrodos van distribuidos por la superficie de una especie de casco elástico como se observa en la Figura 3 existen en diferentes tamaños para que lo utilicen personas adultas o niños. Las características principales del casco es que permiten comodidad de colocación para obtener registros de larga duración, gran inmunidad de ruido, precisión en su colocación, útiles para estudios comparativos, aunque para realizar este tipo de estudios se necesita de una técnica muy depurada.

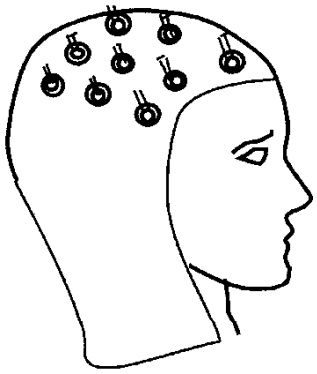


Figura 3: Principio de colocación de electrodos en casco de malla

Fuente: (Sánchez, 2014)

d) De aguja: De uso limitado, es ocupado en recién nacidos. Existen de material desechable o también de uso múltiple, es decir que se los puede reusar. La esterilización es de mucha importancia, así como el cuidado al ser manipulados. Todos los electrodos descritos anteriormente registran solamente la convexidad superior de la corteza.

e) Quirúrgicos: Se usan en eventos quirúrgicos y deben ser utilizados por médicos neurocirujanos. Pueden ser corticales o intracerebrales.

1.6.3 Ondas del EEG

La amplitud varía desde los 10mv en registros sobre el córtex hasta los 100μV en la superficie del cuero cabelludo. Las ondas en la mayoría de veces no poseen ninguna forma determinada, en ritmos normales en los cuales se puede clasificar son α, β, θ, δ como se muestra en la Figura 4.

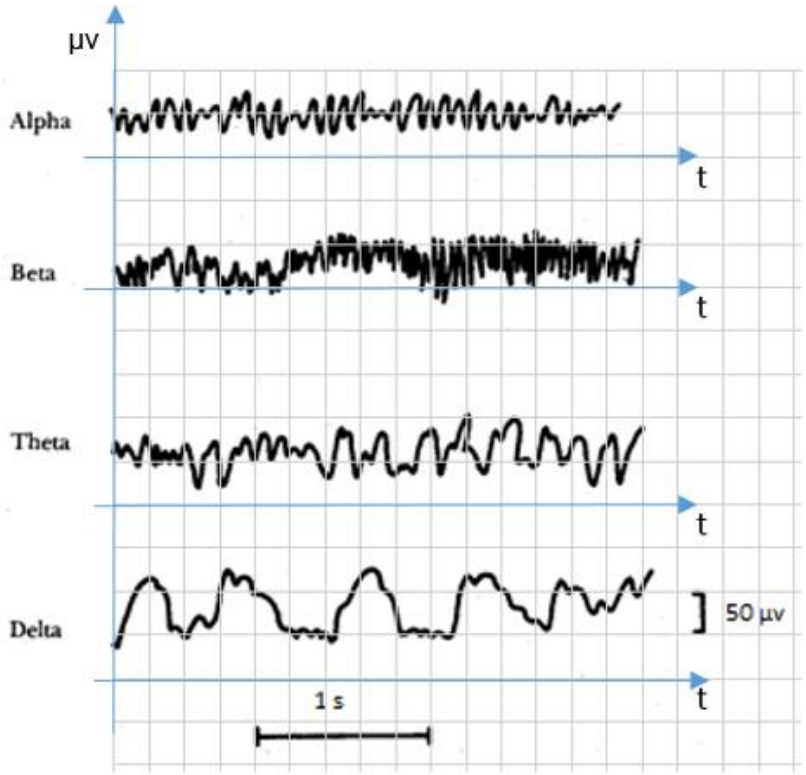


Figura 4: Ritmos normales en electroencefalografía

Fuente: (Barea, 2002)

a) Ondas Alfa

Tienen una frecuencia de 8 –12 Hz y están asociadas con estados de relajación. Se registran especialmente momentos antes de dormirse. Sus efectos característicos son: relajación agradable, pensamientos tranquilos y

despreocupados, optimismo y un sentimiento de integración de cuerpo y mente.

b) Ondas Betha

Originan un campo electromagnético con una frecuencia comprendida entre 13 y 30 Hz (vibraciones por segundo). Se registran cuando la persona se encuentra despierta y en plena actividad mental. Los sentidos se hallan volcados hacia el exterior, de manera que la irritación, inquietud y temores repentinos pueden acompañar este estado.

c) Ondas Theta

Con una frecuencia de 4-7 Hz, se producen durante el sueño (o en meditación profunda, entrenamiento autógeno, yoga), mientras actúan las formaciones del subconsciente. Las características de este estado son: memoria plástica, mayor capacidad de aprendizaje, fantasía, imaginación e inspiración creativa.

d) Ondas Delta

Con una frecuencia de 1-3 Hz, surgen principalmente en el sueño profundo y muy raras veces se pueden experimentar estando despierto. Sus estados psíquicos correspondientes son el dormir sin sueños, el trance y la hipnosis profunda. Las ondas delta resultan de gran importancia en los procesos curativos y en el fortalecimiento del sistema inmunitario.

Por medio del sueño se pueden notar cambios espectrales que indican posibles patologías cerebrales. En la Figura 5 se destacan algunas fases del sueño que corresponden sucesivamente a los estados de alerta o excitación, de relajación, de somnolencia, de sueño y, finalmente, de sueño profundo. Al hacer un análisis en el tiempo se puede observar claramente que la frecuencia de las ondas del EEG van reduciéndose progresivamente, aunque también se tienen transitorias rápidas que cambian la forma de la señal al ser analizada.

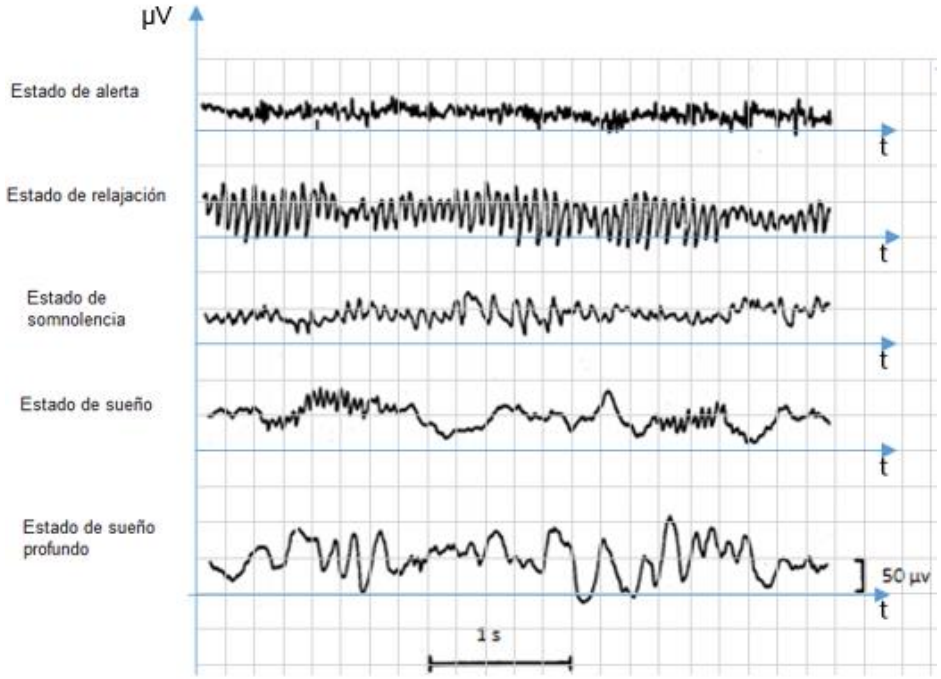


Figura 5: EEG durante diferentes fases del sueño

Fuente: (Barea, 2002)

1.7 La robótica y las personas con capacidades especiales

La discapacidad es un problema de toda la sociedad debido a que la gran mayoría de personas sufren o sufrirán algún tipo de discapacidad transitoria o permanente a lo largo de toda su vida. Según el informe presentado por la Organización Mundial de la Salud –OMS- en el año 2011 existen 600 millones de personas en todo el mundo que presentan algún tipo de discapacidad que afectan las habilidades motrices, perceptivas o intelectuales (Organización Mundial de la Salud, 2011).

El registro nacional de discapacitados presentado en septiembre del 2014 por el Consejo Nacional de Discapacitados (CONADIS) muestra que existen 397233 personas con alguna discapacidad auditiva, física, intelectual, de lenguaje, mental, psicológico y visual; dichas personas representan el 2,48% de la población ecuatoriana (Consejo Nacional para la Igualdad de Discapacitados, 2014); en la provincia de Cotopaxi existen 409.205 personas de las cuales 10.087 presentan alguna discapacidad y estas a la vez contienen

a 4.397 personas con discapacidades físicas (Instituto nacional de estadísticas y censos, 2010).

Mediante la robótica se puede mejorar la calidad de vida de un ser humano que presenta problemas de discapacidad, a través del uso de prótesis se ayuda a personas para que se desenvuelvan de una manera adecuada en la colectividad y mejoren su estado emocional, resulta preciso conocer algunas enfermedades degenerativas que deben ser analizadas como: lesión medular, parálisis cerebral, mal de parkinson, distonía muscular.

En el Ecuador se están realizando muchos aportes tecnológicos para cambiar la matriz productiva y de esta manera ayudar a combatir problemas sociales, por lo dicho anteriormente el proyecto se focaliza en asistir a personas con discapacidad física en alguna de sus extremidades superiores para mejorar su estilo de vida, contribuyendo así al desarrollo tanto tecnológico y social del país. Los avances en cuanto a robótica han ampliado su campo de estudio, ya que en sus inicios solo se enfocaban en aplicaciones industriales. Un brazo robótico hoy en día resulta de mucha ayuda para personas discapacitadas que no poseen sus extremidades superiores.

Al estar vinculada la robótica en diferentes campos de la tecnología como la electrónica, la informática, la mecánica y el control, puede dar soporte a muy diversos problemas de disminución física. De forma global, utilizando un sistema robótico completo, o bien disponiendo únicamente de simples mecanismos, dispositivos automáticos o teleoperados y equipos informáticos, se pueden diseñar y construir sistemas de ayuda.

Las primeras aplicaciones de la robótica en el campo de la discapacidad datan ya de los años 70, con la construcción de elementos prostéticos y ortéticos (brazos, piernas y manos). En estos dispositivos el control está basado en las propias señales mioeléctricas del usuario, o en elementos auxiliares adaptados a las capacidades remanentes de interacción del mismo usuario con el sistema. El robot encuentra también su aplicación como herramienta de ayuda, utilizado como un soporte externo, un asistente, bajo

el control del propio usuario. En esta línea existen ya diversas soluciones: ya sea disponer de un robot fijo, un robot adaptable a la silla de ruedas o un robot sobre una base móvil para desplazarse en su entorno, doméstico o de trabajo.

Las operaciones a realizar cotidianamente son muchas y muy diferentes, algunas de ellas son además demasiado complejas para ser programadas en los robots industriales actuales. Así pues, hoy en día no se puede pensar en disponer de un robot doméstico capaz de ayudar a una persona severamente discapacitada a efectuar todas estas funciones. Sin embargo, sí es posible utilizar robots orientados a efectuar un limitado número de funciones básicas tales como apartar y acercar objetos, ayudar a beber o comer, ayudar al usuario en su higiene personal, a pasar hojas de un libro.

En cambio, otras acciones para el control del entorno, ya resueltas actualmente, como levantar y bajar persianas, conectar-desconectar la radio, TV, son efectuadas más eficientemente mediante dispositivos específicos que van siendo cada vez más utilizados. En esta línea, la casa inteligente, creada inicialmente para aumentar la comodidad a familias acomodadas, constituye ya una potencial y gran ayuda a personas mayores y personas con distintos niveles de discapacidad. La seguridad es un factor esencial. Mientras en el entorno industrial el robot está normalmente aislado del entorno de operación humana, el robot asistencial debe en general operar cerca del usuario, e incluso en contacto con él.

Para ello será preciso disponer de dispositivos de seguridad, como sensores para la detección de proximidad o contacto, que permitan controlar el robot en consecuencia, o utilizando estructuras blandas para construir el robot, y por tanto intrínsecamente seguras. Las primeras realizaciones prácticas en robótica asistencial se basaron en la utilización de robots industriales con adaptaciones para este tipo de aplicación y principalmente para garantizar la seguridad del usuario.

1.8 Modelamiento Cinemático Directo

El álgebra vectorial y matricial se utiliza para describir la localización de un objeto en el espacio tridimensional con respecto a un sistema de referencia fijo. Un robot al estar conformado por una cadena cinemática formado por eslabones unidos entre sí por medio de articulaciones, se puede plantear un sistema de referencia fijo en la base del robot y representar la localización de cada uno de los eslabones con respecto a dicha referencia (Barrientos, Peñín , Balaguer, & Aracil, 1997).

Las relaciones que ayudan a localizar espacialmente el extremo del robot no resultan tan complicadas en robots que tienen pocos grados de libertad mediante sencillas consideraciones geométricas. Por ejemplo para el caso en específico de un robot de dos grados de libertad se tiene que las coordenadas x e y dependen de las coordenadas articulares q_1 y q_2.

La relación entre ellas define al modelo cinemático directo propiamente dicho:

$$\begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = f(q_1, q_2) \tag{1.1}$$

Se identifica de inmediato el modelo cinemático que viene dado por:

$$x = l_1 sen(q_1) + l_2 sen(q_1 + q_2) \tag{1.2}$$

$$y = -l_1 cos(q_1) - l_2 cos(q_1 + q_2) \tag{1.3}$$

A partir del modelo cinemático directo puede obtenerse también la siguiente relación de velocidades:

$$\begin{bmatrix} \dot{\boldsymbol{x}} \\ \dot{\boldsymbol{y}} \end{bmatrix} = \boldsymbol{J}(\boldsymbol{q}) \begin{bmatrix} \dot{\boldsymbol{q}}_1 \\ \dot{\boldsymbol{q}}_2 \end{bmatrix} \tag{1.4}$$

$$\begin{bmatrix} \dot{\boldsymbol{x}} \\ \dot{\boldsymbol{y}} \end{bmatrix} = \begin{bmatrix} l_1 cos(q_1) + l_2 cos(q_1 + q_2) & l_2 cos(q_1 + q_2) \\ l_1 sen(q_1) + l_2 sen(q_1 + q_2) & l_2 sen(q_1 + q_2) \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{q}}_1 \\ \dot{\boldsymbol{q}}_2 \end{bmatrix} \tag{1.5}$$

donde $\boldsymbol{J}(\boldsymbol{q})$ recibe el nombre de matriz jacobiana o simplemente jacobiana del robot. Para el caso de robots no redundantes, el jacobiano es una matriz cuadrada de dimensión n cuyos elementos dependen del vector de coordenadas articulares $\boldsymbol{q}$ (Kelly & Santibáñez, 2003).

1.9 Brazo Robótico

Un manipulador o brazo robótico se puede definir como el conjunto de elementos electromecánicos que permiten el movimiento de su extremo terminal, destinado para el agarre y desplazamiento de objetos, el cual es desarrollado para realizar múltiples funciones programables. La mayor parte de manipuladores tiene similitud con las extremidades superiores del cuerpo humano como se indica en la Figura 6.

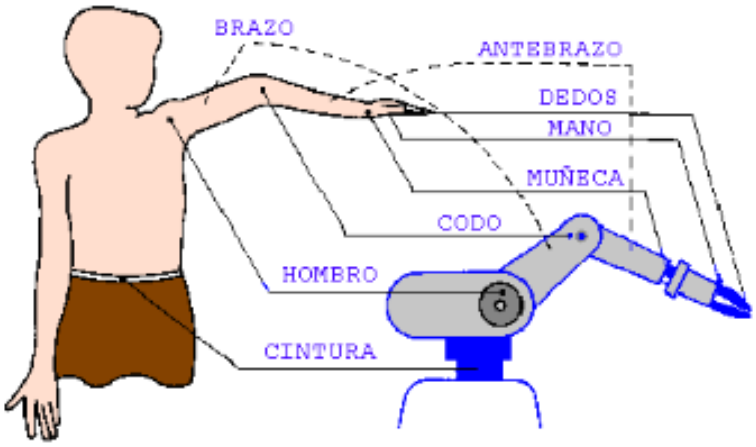


Figura 6: Comparación del brazo robótico con extremidades superiores del cuerpo humano

Fuente: (Martínez, Jáquez, Rivera, & Sandoval, 2008)

Grados de libertad: Los robots están conformados por eslabones los cuales están acoplados mediante articulaciones, en los robots industriales el número de grados de libertad equivalen al número de articulaciones siempre y cuando la articulación a la que se haga referencia tenga un solo grado de libertad.

1.10 Tipos de Brazos Robóticos

Los brazos robóticos que existen difieren unos de otros por los números de ejes o grados de libertad que poseen, el modo de control, por su estructura, tamaño y el medio donde se los emplean. Cada uno tiene una configuración que lo representa y son empleados en áreas de trabajo de todo tipo. Los diferentes brazos robóticos que se pueden destacar son los siguientes:

1.10.1 Robot Cartesiano

El robot presenta una estructura articulada como se muestra en la Figura 7, conformado por una serie de eslabones que ayudan al movimiento del mismo, los movimientos que este tipo de robot puede realizar son de forma vertical, horizontal y transversal, los ejes del robot concuerdan con los ejes cartesianos.

Sus tres articulaciones fundamentales son prismáticas, el desplazamiento sobre los ejes proporcionan las coordenadas cartesianas X, Y, Z de los puntos de trabajo. Presentan ventajas como la de ser precisos, rápidos y tener una alta capacidad de carga. Estos robots son empleados en aplicaciones donde se requiere movimientos lineales de gran precisión.

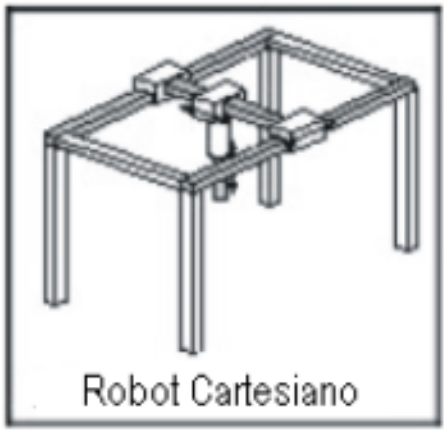


Figura 7: Robot cartesiano

Fuente: (López Segovia, Alamilla Santiago, & Domínguez Vázquez, 2007)

1.10.2 Robot cilíndrico

La estructura del robot permite un desplazamiento giratorio del eje sobre su base y dos desplazamientos perpendiculares, la posición de los puntos en el espacio se halla mediante las coordenadas cilíndricas. Al poseer un eje rotacional facilita para que el control de este robot sea fácil y rápido, así como también una mejor maniobrabilidad y velocidad. Se utiliza para casos en que no haya obstáculos en su zona de trabajo. El robot es empleado en tareas de soldadura, manipulación de piezas, pintura, mecanizado y ensamblaje. Generalmente el robot al realizar un trabajo no tiene una rotación de 360°. El robot de tipo cilíndrico se puede observar en la Figura 8.

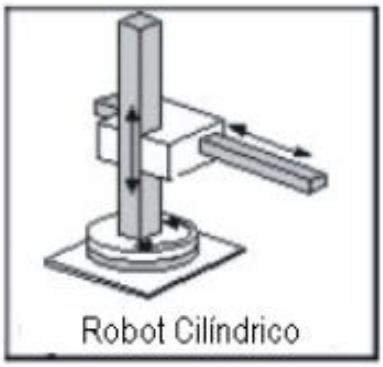


Figura 8: Robot Cilíndrico

Fuente: (López Segovia, Alamilla Santiago, & Domínguez Vázquez, 2007)

1.10.3 Robot antropomórfico

Conocido también como robot angular posee tres articulaciones principales como se observa en la Figura 9, cuenta con tres grados de libertad y cada una de ellos es una articulación de tipo rotacional, emplea las coordenadas angulares para hallar la posición del elemento terminal. Simula los movimientos de un brazo humano, comparando el primer eje con el cuerpo, el segundo eje con el brazo, el tercer eje con el antebrazo y la pinza de su extremo con la muñeca y mano. Las ventajas de este robot son la accesibilidad, maniobrabilidad y el poco espacio que ocupa en relación al campo de trabajo.

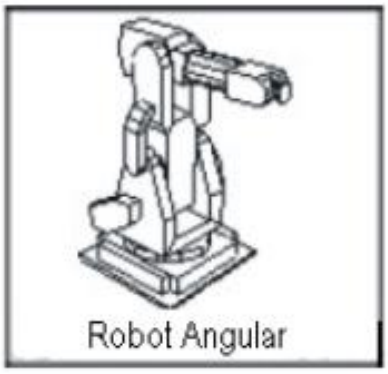


Figura 9: Robot Antropomórfico o angular

Fuente: (López Segovia, Alamilla Santiago, & Domínguez Vázquez, 2007)

1.10.4 Robot Paralelo

Un robot de configuración paralela posee la característica de tener su efector final unido a la base por más de una cadena cinemática independiente, la plataforma móvil que tiene el robot va acoplada a una base fija por medio de varios brazos. Pueden operar una mayor carga por la ventaja de tener varios brazos paralelos. Los robots paralelos según su movilidad pueden clasificarse en:

- **Robots paralelos planares:** El mecanismo más básico empleado es de cinco barras. Está conformado por cuatro eslabones y una base, unidos mediante cinco articulaciones rotacionales como se indica en la Figura 10.

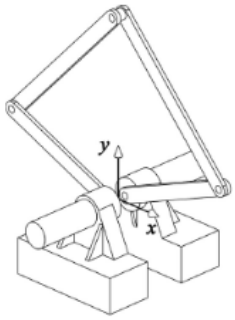


Figura 10: Robot paralelo planar

Fuente: (López Segovia, Alamilla Santiago, & Domínguez Vázquez, 2007)

- **Robots paralelos traslacional:** Tienen articulaciones esféricas en los paralelogramos que lo conforman, el manipulador Delta es un ejemplo de este tipo de robot como se muestra en la Figura 11. Posee tres eslabonamientos los cuales conectan la base fija con el triángulo equilátero del efector final, consta de tres grados de libertad.

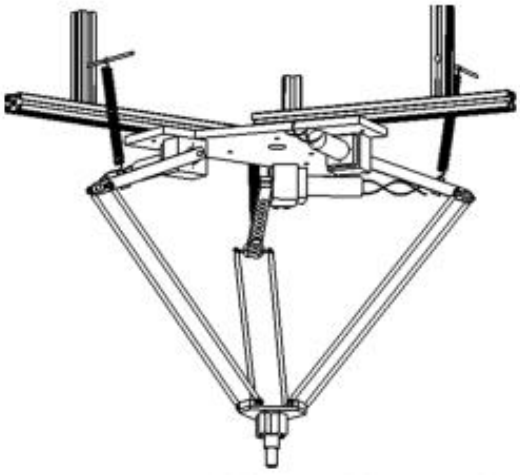


Figura 11: Manipulador paralelo tipo Delta

Fuente: (Meneses Jiménez, Méndez Canseco, & Cortés Bringas, 2007)

1.10.5 Robot Esférico o polar

Este robot posee varias articulaciones como se muestra en la Figura 12 que permiten el desplazamiento de forma rotacional, angular y lineal, por lo que sus articulaciones presentan la característica de ser prismáticas y rotacionales. La posición de un punto en el espacio se realiza mediante coordenadas polares. Los primeros robots utilizaron la configuración de este tipo, se emplean en tareas de soldadura por punto, manipulación de máquinas herramientas, movimiento de cargas elevadas en donde no se requiera de mucha exactitud ni la realización de movimientos complejos.

Figura 12: Robot Esférico o polar

Fuente: (Moya Pinta, 2010)

1.10.6 Robot SCARA

El robot SCARA como se muestra en la Figura 13 está diseñado para funciones de ensamble, posee cuatro ejes de movimiento, tres ejes que lo ayudan para realizar el posicionamiento del robot y uno para la orientación de la muñeca. SCARA que viene del acrónimo Selective Compliance Assemble Robot Arm que en español significa brazo de cumplimiento selectivo para robot de montaje. Las tareas que realiza este tipo de robot está destinada habitualmente al manejo de productos pequeños y que a la ves requieren de velocidad, el costo del mantenimiento que se realiza al robot es mínimo en comparación con otros tipos de manipuladores.

La configuración de este tipo de robots son utilizados en trabajos de paletización, ensamblaje de componentes electrónicos, realización de pruebas de calidad de componentes electrónicos, fabricación y manipulación de medicamentos, dosificación y envase de productos farmacéuticos, montaje de elementos de unión (tornillos y remaches), aplicación de adhesivos.

Figura 13: Robot SCARA

Fuente: (Martínez Castellanos, 2012)

1.10.7 Robot Articulado

Este tipo de robot tiene una estructura conformada por eslabones y articulaciones que facilitan la rotación o traslación entre dos de los eslabones, sostenidos por una base que puede ser horizontal, vertical o suspendida. Una configuración típica de este tipo de robot es el de tres grados de libertad añadiendo la posibilidad del movimiento en la muñeca, existen también robots que llegan a tener de cuatro a seis grados de libertad y robots de siete a nueve grados de libertad pero son menos comunes ya que tienen mayor complejidad. Es usado en funciones como ensamblaje, fundición a presión, soldadura por arco y pintado en spray. El área de trabajo de un robot articulado se observa en la Figura 14.

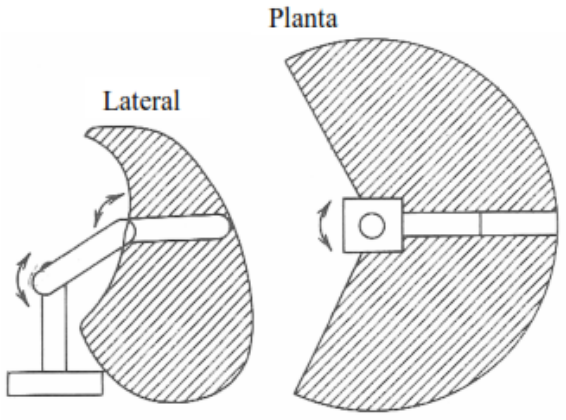


Figura 14: Área de trabajo de un robot Articulado

Fuente: (Abdalá Castillo & Ñeco Caberta, 2003)

1.11 Estructura de un robot

Un robot está constituido por la siguientes partes: eslabones y articulaciones, reductores, actuadores, sistema sensorial, sistema de control, elementos terminales, todo esto se muestra en la Figura 15.

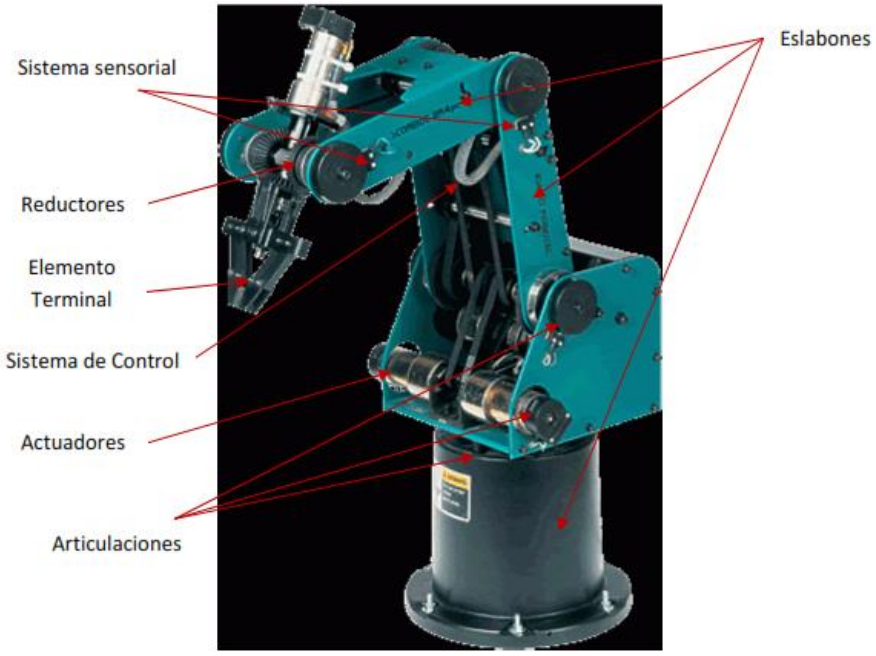


Figura 15: Estructura de un Robot

Fuente: (Moya Pinta, 2010)

1.11.1 Eslabones y articulaciones

El eslabón es la parte rígida que permite la unión entre dos de las articulaciones del robot y que por lo general representan el cuerpo del robot. El movimiento realizado por cada articulación puede ser de desplazamiento, de giro, o una combinación de ambos. Se tienen seis tipos de articulaciones como se indica en la Figura 16, en la práctica se emplean la de rotación y la prismática.

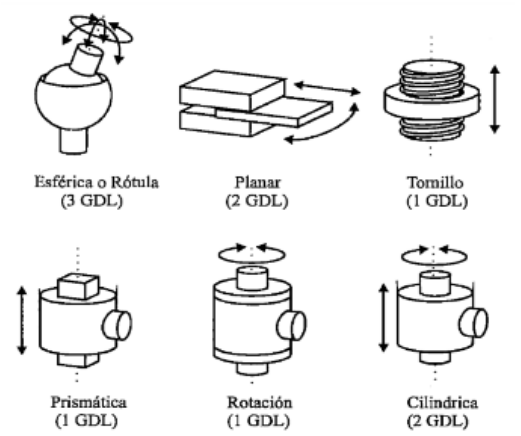


Figura 16: Tipos de articulaciones para robots

Fuente: (Moya Pinta, 2010)

1.11.2 Reductores

Su objetivo es adaptar par y velocidad de la salida del actuador para realizar el movimiento de los eslabones del robot, además se enfoca en aumentar la precisión en la medición del giro del eje sin introducir juegos mecánicos, con esto el robot puede realizar tareas o aplicaciones específicas para las que ha sido diseñado.

1.11.3 Actuadores

Dispositivos que reciben la señal desde el microcontrolador del robot manipulador, permiten el movimiento en los efectores finales del robot, estos pueden ser sistemas hidráulicos, eléctricos, neumáticos y electrónicos principalmente (Moya Pinta, 2010). Los actuadores pueden verse como

transductores, por ejemplo un motor convierte energía eléctrica (se conecta a una fuente de alimentación) en energía mecánica rotacional.

1.11.4 Sistema Sensorial

Conjunto de dispositivos electrónicos que permiten la interacción del robot con su entorno (Moya Pinta, 2010). Hay muchas maneras de clasificar a los sensores, de las cuales se tienen las siguientes:

- Directos, Indirectos
- Activos (Generadores)
- Pasivos (Moduladores)
- Resistivos, Capacitivos, Inductivos
- Locales o Remotos, Analógicos, Digitales

1.11.5 Sistema de control

Es el encargado de controlar el movimiento que se realiza en las articulaciones del robot. Resulta complejo, debido a las propiedades cinemáticas y dinámicas de los robots que varían dependiendo de la posición y a las ecuaciones resultantes que son no lineales. Existen diferentes técnicas que se pueden aplicar para controlar el robot, como son el control cinemático y control dinámico. Para el caso de la industria se utilizan microprocesadores los cuales son los encargados de realizar el control del robot, realizan funciones computacionales y también el control de pinzas o herramientas que posea el robot en su extremo operativo.

1.11.6 Elementos terminales

Son herramientas en forma de tenazas o garras, poseen la capacidad de sujetar, orientar y operar sobre las piezas manipuladas. La aplicación específica de un robot industrial que dispone de elementos terminales son el recolectar piezas, soldar, pintar, etc. Existen algunas funciones para la pinza del robot dependiendo del campo de trabajo en donde se lo utilice.

1.12 Servomotores

Un servomotor (o servo) es un motor de corriente continua que tiene la capacidad de ser controlado en posición. El mismo tiene la capacidad de ubicarse en cualquier posición dentro de un rango de operación por lo habitual de 180º y mantenerse estable en dicha posición. Los servos son utilizados en robótica, automática y modelismo (vehículos por radio-control, RC) debido a que presentan una alta precisión en el posicionamiento (Candelas Herías & Corrales Ramón, 2007).

1.13 Equipos que trabajan con señales EEG

Varias empresas han desarrollado dispositivos que permiten trabajar con señales cerebrales en los campos de la educación así como también para la implementación en videojuegos de los cuales pueden mencionar los siguientes:

1.13.1 Neurosky

Esta compañía desarrolla interfaces cerebro computadora, hace uso de la electroencefalografía, en donde se utilizan electrodos superficiales sobre la cabeza para captar los impulsos eléctricos que se producen cuando una persona está pensando o realizando algún tipo de acción.

Se han realizado mediciones de dos emociones distintas, la primera es la medida de atención que es un indicativo del grado de concentración que siente el usuario y la medida de meditación en donde se puede notar el grado de relajación que llega a tener el usuario. No es capaz de detectar pensamientos complejos como los de la competencia y tampoco detecta gestos faciales solo los interpreta como ruido.

La empresa ha creado un juego en donde permite al usuario hacer uso de la concentración para que el jugador empuje objetos como coches y utilizando la meditación permite visualizar en el juego que el objeto está levitando. Por

medio de los avances tecnológicos realizados por Neurosky cualquier persona puede hacer uso del dispositivo electrónico desde la comodidad del hogar.

El dispositivo Mindwave como el que se muestra en la Figura 17 diseñado por Neurosky, es un casco de electroencefalografía que llega al mercado después varios años de estudios científicos y utilización en ámbitos clínicos. Este instrumento permite al usuario medir la frecuencia de las ondas cerebrales, teniendo la opción de monitoreo y visualización en la pantalla del ordenador. En escasos minutos es capaz de medir niveles de atención, relajación y meditación. Este casco no es apto para captar pensamientos concretos. No tiene efectos como generar algún tipo de corriente o interferir con la actividad cerebral.

Figura 17: Casco Mindwave

Fuente: (Ponce Jurado, 2014)

Brainwave Starter Kit como se indica en la Figura 18 lleva décadas de investigación en tecnología para EEG y se pone a disposición para los usuarios. Basta con colocarse el auricular y visualizar en tiempo real las ondas cerebrales. Permite controlar los niveles de atención y relajación, aprender acerca de cómo el cerebro responde a su música favorita. Provee más de cien juegos de entretenimiento cerebral y aplicaciones educativas disponibles, dependiendo de la edad e interés personal.

Figura 18: Brainwave Starter Kit

Fuente: (Ponce Jurado, 2014)

El hardware de Neurosky, utiliza el sensor Dry Active patentado por la compañía para leer las señales cerebrales. Los electrodos estándar en la electroencefalografía médica usan un gel conductor para facilitar la lectura de las señales. El sensor Dry Active no necesita dicho gel, a diferencia de la electroencefalografía médica, que utiliza muchos electrodos para realizar las mediciones, Neurosky sólo usa uno. Esto hace que los dispositivos basados en tecnología Neurosky sean de muy bajo costo y fáciles de usar (Ponce Jurado, 2014).

1.13.2 EmSense

Es una de las compañías americanas que más destaca por su aplicación de las neurociencias a la investigación de mercados. Es una empresa ubicada en EE.UU., en la ciudad de San Francisco. Emsense ha desarrollado técnicas para unir la electroencefalografía con otras mediciones biométricas, con ello puede ofrecer a los usuarios una mejor comprensión del inconsciente de sus consumidores al analizar las ondas cerebrales que son emitidas al realizar diferentes acciones.

La empresa ha diseñado un casco ligero con tecnología EEG denominado EmGear como se indica en la Figura 19, el mismo es no invasivo. EmGear aparte de trabajar con las señales cerebrales tiene funciones adicionales

como controlar el ritmo de respiración, movimiento de la cabeza, pulsaciones, ritmo del parpadeo y la propia temperatura de la piel.

Figura 19: Casco EmGear de EmSense

Fuente: (Ponce Jurado, 2014)

Los tres conceptos que mide la metodología que utiliza EmSense para la investigación son:

- **Agrado:** Medición de las emociones positivas que la persona tiene al estar viendo y oyendo cuando está colocada el casco.
- **Pensamiento:** Medición del esfuerzo cognitivo que el usuario realiza al procesar a cada momento lo que está percibiendo.
- **Adrenalina:** Una medida de la activación o relajación de los sujetos, tomando en cuenta los latidos del corazón.

1.13.3 Neuroelectrics

Es una compañía derivada a partir de Starlab la cual tiene sede en la ciudad de Barcelona en España, tiene como objetivo el transformar la ciencia en tecnologías que conlleven a un profundo impacto en la sociedad, proporciona herramientas de medición y toma de neuromodulación asequible y disponible para todos los pacientes que lo necesitan. Los dispositivos utilizan el proceso de neuromodulación que modifica el estado del cerebro mediante la retroalimentación basándose en el entrenamiento.

Enobio es un sistema de sensores de electrofisiología portátil, con la característica de ser inalámbrico y permite la grabación de señales cerebrales. La gorra Enobio ocho como la que se muestra en la Figura 20 es ideal para aplicaciones fuera del laboratorio, viene integrada con una interfaz de usuario intuitiva y potente, capaz de reproducción de 8 canales de la señal de EEG en bruto, es de fácil configuración, se puede grabar y visualizar 24 bits de datos de EEG, incluyendo espectrograma y la posibilidad de ver en 3D las características espectrales.

Figura 20: Gorra Enobio Ocho de Neuroelectrics

Fuente: (Neuroelectris, 2011)

Además es posible enlazarse a la Nube y recoger datos experimentales, sea para investigación o uso clínico. Se tiene un acelerómetro que va recolectando los datos automáticamente, utiliza una tarjeta microSD para guardar datos en línea. Tiene un ancho de banda de muestreo de 0 a 125 Hz, permite grabar EEG en Delta, Theta, Alfa, Beta, Gamma. Los datos obtenidos de electroencefalografía pueden ser transmitidos en red para el desarrollo de aplicaciones de terceros.

El casco Enobio está disponible tanto para adultos como para niños ya que viene en diferentes tamaños. Se tiene la opción de trabajar en tiempo real con Matlab utilizando el kit de herramientas denominado MatNIC de Neuroelectrics. El casco Enobio viene también en versiones de 20 y 32 canales de captación de EEG.

1.13.4 Mindo

Mindo es una compañía que ofrece al usuario la posibilidad de captación de señales de EEG mediante la utilización de sus cascos. Los mismos no requieren de una solución para humedecer los sensores que captan las señales cerebrales, la calidad de los datos son comparables a sistemas que sí humedecen los electrodos. Los auriculares de la empresa son fáciles de usar y quitarse. Poseen una tecnología inalámbrica que permite grabaciones sin estar atado a una computadora.

Mindo incluye electrodos portátiles, inalámbricos y secos que permiten el monitoreo EEG, los equipos usados para el EEG son multicanal, permiten controlar juegos, supervisar las etapas de sueño y mejorar la concentración.

Los cascos que ofrece la empresa son los siguientes:

- **Mindo-4S Jellyfish:** Es un dispositivo con tecnología para la detección de señales cerebrales inalámbrico y de peso ligero, utiliza una banda de base de silicio, el equipo mide la actividad cerebral en la zona de la frente como se muestra en la Figura 21. Incorpora 4 sensores para detectar las señales de EEG en tiempo real, y se puede realizar un seguimiento de las señales en la tablet o PC a través de Bluetooth para traducirlos en datos significativos que el usuario pueda interpretar. Es compatible con los sistemas operativos Android y sistemas de PC. También utiliza una batería recargable de hasta 10 horas de duración.

Figura 21: Mindo-4S Jellyfish

Fuente: (Mindo, 2013)

- **Mindo-64 Coral:** Es un casco como se indica en la Figura 22 que posee sensores secos que no requieren líquidos. Tiene una alta calidad en la obtención de señales cerebrales. Sistema con comunicación inalámbrica wireless, es ligero, cómodo y recargable lo que permite la captura de datos móviles. Detección a través de 64 canales, lo que posibilita obtener una señal de mayor calidad.

Figura 22: Mindo-64 Coral

Fuente: (Mindo, 2013)

- **Mindo-32 Trilobite:** La ventaja más importante de este tipo de casco es que la calidad de los datos es comparable a la obtenida con sistemas de electrodos húmedos pero sin la necesidad de abrasión de la piel. Detección a través de 32 canales, es de tecnología inalámbrica lo que permite a la persona que utiliza el equipo moverse libremente por una habitación, oficina, o donde se esté utilizando el dispositivo, en la Figura 23 se observa la colocación del casco Mindo-32 Trilobite.

Figura 23: Mindo-32 Trilobite

Fuente: (Mindo, 2013)

1.13.5 Biosemi

Instrumentación BioSemi es la iniciativa de los ingenieros electrónicos Robert Honsbeek, Ton Kuiper y el físico Coen Metting van Rijn. Después de trabajar durante más de una década en el campo de la investigación sobre la instrumentación de investigación en el departamento de Física Médica de la Universidad de Amsterdam, decidieron comenzar BioSemi en 1998 (Biosemi, 1998).

Durante la última década, este grupo ha analizado los problemas fundamentales de las mediciones biopotencial multicanal modernas en el marco de proyectos de investigación académicos patrocinados por la Fundación Tecnología STW. Con base en los resultados de estos proyectos de investigación BioSemi ofrece una gama de equipamiento de última generación para las mediciones biopotencial más exigentes. Los principios básicos fueron descritos en diversas publicaciones científicas (Biosemi, 1998).

La empresa BioSemi ha desarrollado un casco en colaboración con el Dr. Peter Praamstra en el Centro de Ciencias del Cerebro del Comportamiento de la Universidad de Birmingham, Reino Unido (Biosemi, 1998). El casco consiste de una gorra elástica de plástico que contiene electrodos, se coloca sobre la cabeza del sujeto, y los soportes de electrodos se llenan con gel.

Se omite el uso de líquidos o pastas ya que posee electrodos activos, con ello se deja a un lado la pérdida de tiempo que representa la colocación del gel en cada uno de los electrodos. El procedimiento es rápido y fiable de 5 a 10 minutos para 32 canales que utiliza el casco (ver Figura 24) y permite la medición de EEG de alta densidad con un mínimo de tiempo de preparación. Biosemi ofrece disponibilidad de cascos que utilizan 16, 64, 128, 160 y hasta 256 canales, el casco que utiliza mayor cantidad de canales se puede observar en la Figura 25.

Figura 24: Casco Biosemi de 32 canales

Fuente: (Biosemi, 1998)

Figura 25: Casco Biosemi de 256 canales

Fuente: (Biosemi, 1998*)*

1.13.6 Emotiv

La empresa se encarga del desarrollo de interfaces cerebro-ordenador basado en la electroencefalografía, ofrece un sistema de EEG inalámbrico para la investigación que permite entretenimiento, estudios de mercado y pruebas de usabilidad. El casco Emotiv EPOC permite captar y amplificar ondas cerebrales realizadas por acciones mentales o gestos faciales, permitiendo así el control de funciones de una computadora. El equipo incluye un giroscopio que detecta los movimientos de la cabeza de una persona de manera precisa y los convierte en movimientos del mouse de la pantalla. En sus inicios el equipo fue desarrollado para su utilidad en diferentes

aplicaciones en donde la detección de acciones mentales y el control neural podrían ser útiles: investigación, biofeedback, control de videojuegos, control de funciones discretas en una computadora y accesibilidad, entre otras.

El EPOC es un aparato tipo diadema que posee dieciséis electrodos distribuidos en diferentes puntos de la cabeza como se muestra en la Figura 26. Los electrodos ocupados para leer las ondas cerebrales utilizan una solución salina que permite tener un buen contacto con la superficie del cuero cabelludo. EPOC al estar situado sobre la cabeza transmite una señal inalámbrica por radiofrecuencia a un receptor USB que va instalado en la computadora y que viene incluido con el casco Emotiv.

Figura 26: Casco Emotiv Epoc

Fuente: (Emotiv, 2009)

La empresa Emotiv cuenta con otro tipo de casco denominado Insight como el de la Figura 27, el cual permite dar seguimiento a la actividad cerebral, éste posee 5 canales de EEG más 2 referencias que ofrecen un posicionamiento óptimo. Tiene sensores de polímeros secos de vanguardia, proporcionan gran conductividad eléctrica y sin preparación. Es de tecnología inalámbrica, compatible con USB y no requiere controladores personalizados. Soporte de IOS a través de bluetooth, su batería es de litio y proporciona 4 horas de uso continuo.

Figura 27: Emotiv Insight

Fuente: (Emotiv, 2009)

1.14 Software libre

El software tiene la característica que puede ser estudiado, distribuido, modificado, copiado y usado, por lo que debe venir con el código fuente para que se puedan utilizar los beneficios que este presenta. Es importante no confundir el software libre con el software gratuito, el hecho que no cueste nada no lo convierte en software libre. Richard Stallman indica que el software libre es una cuestión de libertad y no de precio (Stallman, 2004). Las libertades que el software libre presta a los usuarios son cuatro:

- La libertad de usar el programa, con cualquier propósito.
- La libertad de estudiar cómo funciona el programa, y de adaptarlo a cualquier necesidad, para ello el acceso al código fuente es una condición necesaria.
- La libertad de distribuir copias, con lo que se pueda beneficiar otro usuario.
- La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie. El acceso al código fuente es un requisito previo para esto.

Entonces la libertad del software libre también incluye la opción de modificar y redistribuir, literal o con modificaciones, de manera gratis o

mediante una gratificación. Al utilizar un programa cualquier individuo u organización tiene la opción a ejecutarlo en cualquier sistema operativo sin la necesidad de comunicárselo ni al desarrollador ni a ninguna entidad en concreto.

1.14.1 Ventajas de software libre

- Libertad de copia.
- Libertad de uso con cualquier fin.
- El usuario no depende de la persona que realiza el software.
- Libertad para realizar modificaciones y mejoras.
- Bajo costo.
- Trabajo en conjunto con los usuarios, permitiendo una mayor innovación tecnológica.
- De rápida corrección de errores y fácil acceso a su código fuente.
- Libertad de redistribución.
- Mayor seguridad y fiabilidad.
- Independencia de un proveedor.
- Facilidad de personalizar el software dependiendo de las necesidades del usuario.
- Elimina riesgos de filtración de códigos maliciosos o de espionaje.

1.14.2 Desventajas del software libre

- No hay una garantía por parte del autor.
- Las interfaces gráficas son menos amigables.
- No se tiene un control de calidad previo.
- Hay aplicaciones específicas que no se las encuentra en software libre.
- Para su implementación se necesitan de conocimientos previos.
- Dificultad de migración de datos del usuario.

1.15 Tipos de software libre

Los softwares que se presentan a continuación poseen características como la de ser orientada a objetos, lo cual es de mucha ayuda al momento de realizar un código de programación, de igual modo poseen variedad de tipos de constructores.

1.15.1 Haskell

Es un lenguaje de programación moderno, estándar, no estricto, funcional, su logotipo se muestra en la Figura 28. Posee características avanzadas como polimorfismo de tipos, evaluación perezosa y funciones de alto orden. También soporta una forma sistemática de sobrecarga y un sistema modular. Diseñado para manejar un ancho rango de aplicaciones, tanto numéricas como simbólicas. Haskell tiene gran variedad de constructores de tipos, a parte de los tipos convencionales (enteros, punto flotante y booleanos). Los usuarios que van a empezar a usar este software tienen la opción de utilizar Hugs, el cual es un intérprete pequeño y portable de Haskell.

Figura 28: Logotipo de Haskell

Fuente: (Haskell, 1990)

Haskell es un lenguaje apropiado para programas que necesitan ser altamente modificados y mantenidos. Haskell es un lenguaje puramente funcional que ofrece:

- Incremento de la productividad de los programas.
- Código más claro y corto.
- Tiempos de computación más pequeños.

1.15.2 Smalltalk

Smalltalk es un lenguaje orientado a objetos puro, ya que las entidades que maneja son objetos. El lenguaje se basa en conceptos tales como objetos y mensajes. Integra de una manera consistente características tales como un editor, un compilador, un sistema de ventanas y un manejador de código fuente. Es un programa de tipado dinámico. Es considerado un entorno de objetos, donde incluso el propio sistema es un objeto. La programación en Smalltalk requiere de al menos los siguientes conocimientos:

- Los conceptos fundamentales del lenguaje: manejo de clases y objetos, mensajes, clases y herencia.
- La sintaxis y la semántica del lenguaje.
- Cómo interactuar con el ambiente de programación de Smalltalk para construir nuevas aplicaciones Smalltalk.
- Clases fundamentales del sistema, tales como numéricas, colecciones, gráficas y las clases de interface del usuario.

1.15.3 Java

La principal característica de Java es la de ser un lenguaje compilado e interpretado. Todo programa en Java debe compilarse y el código que se genera bytecodes es interpretado por una máquina virtual. Java es un lenguaje orientado a objetos de propósito general. Su sintaxis es parecida a la de C y C++. Permite que los desarrolladores de aplicaciones escriban el programa una vez y lo ejecuten en cualquier dispositivo, lo que nos indica que el código que es ejecutado en una plataforma no tiene que ser recompilado para correr en otra.

Existen varios niveles de seguridad en Java, desde el ámbito del programador, hasta el ámbito de la ejecución en la máquina virtual. Una fuente común de errores en programación procede del uso de punteros. En Java se han eliminado los punteros, el acceso a las instancias de clase se hace a

través de referencias. Java está preparado para la programación concurrente sin necesidad de utilizar ningún tipo de biblioteca.

1.15.4 Python

Python es un lenguaje de programación interpretado que hace énfasis en presentar una sintaxis que beneficie a realizar un código legible, el ícono del software Python se indica en la Figura 29. Es potente y fácil de aprender, posee estructuras de alto nivel con una perspectiva orientado a objetos. El tipado dinámico, junto con su naturaleza interpretada, hacen de éste un lenguaje ideal para scripting, desarrollo rápido de aplicaciones en diversas áreas y sobre la mayoría de las plataformas.

Figura 29: Icono de Python

Fuente: (Python, 1994)

El intérprete de Python puede extenderse fácilmente con nuevas funcionalidades y tipos de datos implementados en C o C++. Python también puede utilizarse como un lenguaje de extensiones para aplicaciones personalizables. Usando el lenguaje Python se puede crear todo tipo de programas; programas de propósito general y también se pueden desarrollar páginas Web.

Python al ser de Open Suorce, ha sido modificado para que pueda funcionar en diversas plataformas (Linux, Windows, Macintosh, Solaris, OS/2, Amiga, AROS, AS/400, BeOS, OS/390, z/OS, Palm OS, QNX, VMS, Psion, Acorn RISC OS, VxWorks, PlayStation, Sharp Zaurus, Windows CE y PocketPC).

Al programar en Python no hay necesidad de preocuparse por detalles de bajo nivel, como el manejo de la memoria ocupada por el programa. Python contiene una gran cantidad de librerías, tipos de datos y funciones incorporadas en el propio lenguaje, estas ayudan a realizar variedad de tareas sin la necesidad de empezar a programarlas desde cero.

CAPÍTULO II

2. ANÁLISIS DEL DISEÑO

Teniendo en cuenta la información del capítulo anterior, a continuación es necesario determinar los elementos que serán utilizados en el sistema a desarrollar para la comunicación entre el brazo robótico con el casco Emotiv EPOC, además se establecerá el algoritmo de programación tanto para las librerías como para el control del sistema.

2.1 Descripción del sistema

La Figura 30 muestra con un diagrama los elementos que serán utilizados para: la adquisición de las señales cerebrales, desarrollo del algoritmo tanto para el control del brazo robótico así como también del procesamiento de las señales EEG emitidas por el casco antes mencionado, y por supuesto el brazo robótico de 6 grados de libertad ensamblado con servomotores Dynamixel proporcionado por la Universidad de las Fuerzas Armadas Extensión Latacunga.

Figura 30: Elementos para el desarrollo del proyecto

Para realizar la interfaz cerebro computadora (BCI) se hará uso del casco Emotiv EPOC desarrollado por la empresa Emotiv Systems, el mismo que presenta una gran facilidad de uso al momento de obtener la información del electroencefalograma con la ayuda de su amigable interfaz con el usuario, determinando así un mejor desempeño para desarrollar este proyecto en específico.

Una de las principales razones al usar el casco Emotiv EPOC es que cuenta con un kit de desarrollo de software (SDK), el mismo que es un

conjunto de herramientas de desarrollo de software que le permite al programador o desarrollador de software crear aplicaciones para un sistema concreto que en este caso es Windows.Al crear una aplicación en software libre, permite al programador desarrollar proyectos donde el conocimiento de su codificación sea mucho más fácil de entender y no requiera de la compra de algún tipo de licencia que le permita avanzar en un trabajo futuro.

Teniendo en cuenta lo anterior, Python cumple con los requerimientos de software libre necesarios para la realización del presente proyecto ya que puede establecer una comunicación directa tanto con los servomotores Dynamixel así como también con el casco Emotiv EPOC. Como muestra la Figura 31, el brazo robótico de 6 grados de libertad proporcionado por la Universidad de las Fuerzas Armadas Extensión Latacunga está conformado por los servomotores Dynamixel MX-28, MX-64 y MX-106.

Figura 31: Brazo Robótico de 6 grados de libertad

2.2 Casco Emotiv EPOC

El casco Emotiv EPOC dispone de un giroscopio y de 16 electrodos para un mejor reconocimiento de las señales cerebrales emitidas por el usuario. Además el casco cuenta con su propio software llamado Emotiv Control Panel

y una serie de herramientas que facilitan el desarrollo de la codificación del algoritmo para el sistema que se desea implementar.

Para un buen contacto entre los electrodos suministrados en el kit del casco con el cuero cabelludo del usuario, es necesario humedecer a los fieltros de los 16 electrodos con la solución salina, los fieltros deben estar húmedos al tacto pero no empapados. Para la colocación del casco se requiere deslizarlo hacia abajo desde la parte superior de su cabeza teniendo cuidado de colocar los sensores con el inserto de caucho negro detrás de cada lóbulo de la oreja (ver Figura 32); la correcta colocación de la sonda de goma es fundamental para un buen funcionamiento y una mejor sincronización entre el casco y la PC.

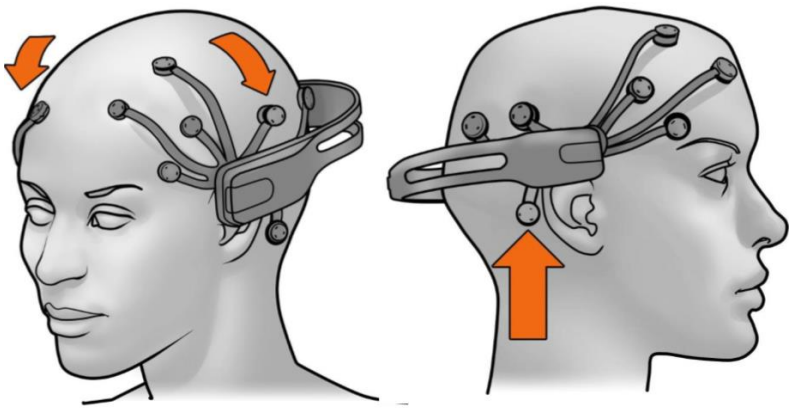


Figura 32: Colocación del casco Emotiv EPOC

Fuente: (Emotiv Systems, 2012)

El Kit de Desarrollo Emotiv consta de un conjunto de librerías que permiten el enlace y comunicación entre el casco, el EmoEngine, el programa Emotiv Control Panel, el programa EmoComposer, el programa Emokey, el manual de usuario, código fuente y la API para desarrolladores. El EmoEngine es el encargado de recibir y procesar la información del electroencefalograma enviado desde casco Emotiv EPOC, así como también de la recepción de varios parámetros como son: estado de la batería, intensidad de la señal inalámbrica, registro del tiempo. El EmoEngine también puede enlazarse para el adiestramiento tanto en el modo expresivo y cognitivo.

2.2.1 Emotiv Control Panel

Emotiv Control Panel (ver Figura 33) proporciona una interfaz gráfica de usuario (GUI) que interactúa con Emotiv EmoEngine a través de Emotiv API. La interfaz de usuario del Panel de Control muestra las capacidades del EmoEngine para descifrar las señales del cerebro y presentarlas de forma amigable para el usuario. Emotiv ofrece la Emotiv API en forma de una biblioteca de enlace dinámico llamado edk.dll.

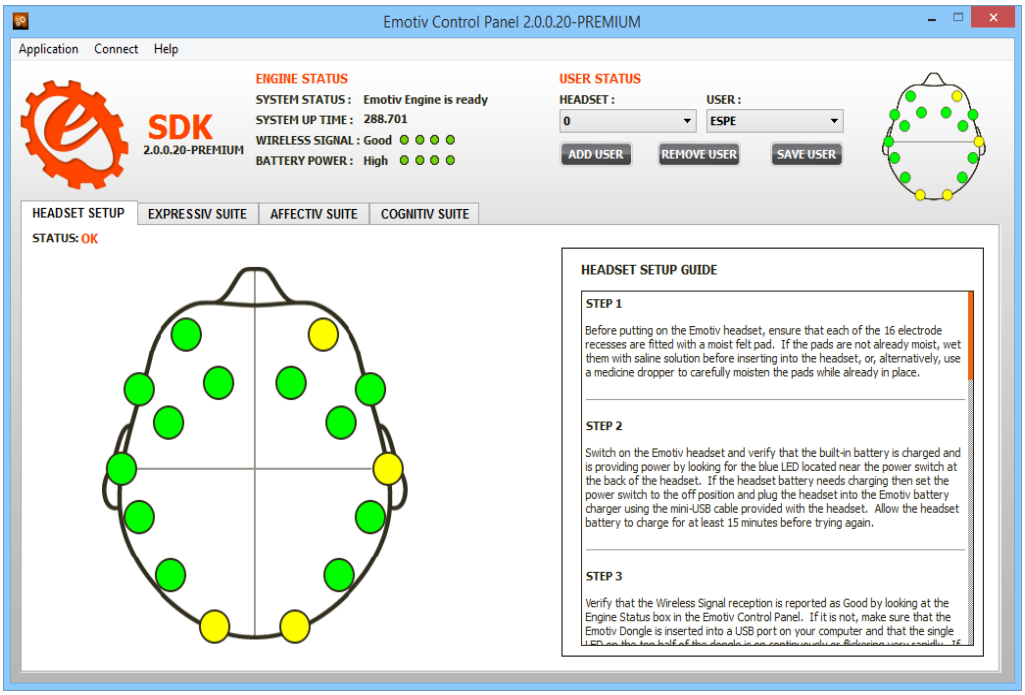


Figura 33: Ventana de Emotiv Control Panel

En la parte superior del Emotiv Control Panel se puede visualizar el estado del sistema, el tiempo de conexión, la intensidad de la señal inalámbrica, el nivel de carga de la batería, la calidad de contacto de los electrodos; además ya que se puede trabajar con dos cascos Emotiv EPOC a la vez se puede seleccionar el casco con el que se desea trabajar con su perfil de entrenamiento. La parte central del Emotiv Control Panel está compuesta por cuatro pestañas las cuales son: Headset Setup, Expressiv Suite, Affectiv Suite, Cognitiv Suite.

2.2.2 Headset Setup (Configuración del Casco)

Como se muestra en la Figura 34 esta pestaña aparece por defecto al iniciar el Emotiv Control Panel y su función principal es mostrar la calidad de contacto de los electrodos y orientar al usuario en el correcto montaje del casco, la mala calidad del contacto dará lugar a malos resultados para la detección del casco.

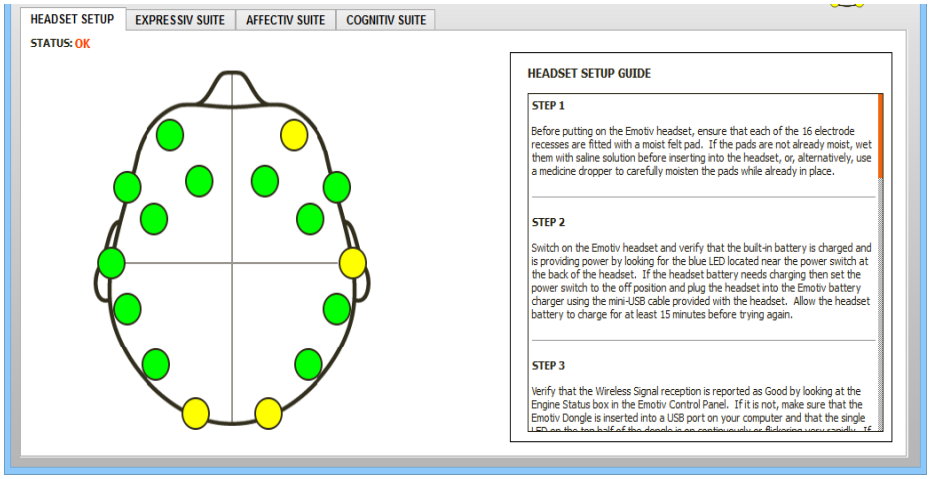


Figura 34: Headset Setup (Configuración del Casco)

La imagen de la izquierda en la Figura 34 es una representación de la ubicación de los sensores, cada círculo representa un sensor y su ubicación aproximada al usar el casco. El color del círculo es una representación de la calidad de contacto del sensor, para lograr la mejor calidad posible de contactos todos los sensores deben mostrarse de color verde. Otros colores que pueden tomar cada uno de los sensores se muestran en la Tabla 1.

Tabla 1
Representación de la calidad de contacto de los sensores del casco Emotiv EPOC

NEGRO	Sin señal
ROJO	Señal muy pobre
NARANJA	Señal pobre
AMARILLO	Señal aceptable
VERDE	Señal buena

2.2.3 Expressiv Suite (Modo Expresivo)

El Modo expresivo interpreta la información de las ondas cerebrales enviadas por el casco para determinar las expresiones faciales emitidas por el usuario. Como indica la Figura 35, en el lado izquierdo del panel se puede visualizar a un avatar, dicho avatar imita las expresiones faciales del usuario que se encuentre utilizando el casco. El centro del panel es una serie de gráficos que indican las diversas señales del evento de detección de la expresión; los gráficos deben interpretarse de la siguiente manera:

- **Pestañear (Blink):** El nivel bajo indica un estado de no parpadear, mientras que un alto nivel indica un parpadeo.

- **Guiñar el ojo derecho/izquierdo (Right/Left Wink):** Estas dos detecciones comparten una línea gráfica común. El nivel de centro indica que no hay guiño, bajo nivel indica un guiño izquierda y alto nivel indica un guiño derecha.

- **Mirar hacia la derecha/izquierda (Look Right/Left):** Estas dos detecciones comparten una línea gráfica común y un solo control deslizante de sensibilidad. A nivel de centro indica ojos mirando hacia el frente, mientras que un nivel bajo indica que los ojos miran a la izquierda, y un alto nivel indica que los ojos miran a la derecha.

- **Levantar las cejas (Raise Brow):** El nivel bajo indica que la expresión no se ha detectado, el nivel alto indica un nivel máximo de expresión detectado. El nivel gráfico aumentará o disminuirá en función del nivel de expresión detectado.

- **Fruncir las cejas (Furrow Brow):** El nivel bajo indica que la expresión no se ha detectado, de alto nivel indica un nivel máximo de expresión detectado. El nivel gráfico aumentará o disminuirá en función del nivel de expresión detectado.

- **Sonreír (Smile):** Nivel bajo indica que la expresión no se ha detectado, de alto nivel indica un nivel máximo de expresión detectado. El nivel gráfico aumentará o disminuirá en función del nivel de expresión detectado.

- **Apretar los dientes (Clench):** El nivel bajo indica que la expresión no se ha detectado, de alto nivel indica un nivel máximo de expresión detectado. El nivel gráfico aumentará o disminuirá en función del nivel de expresión detectado.

- **Mueca derecha/izquierda (Right/Left Smirk):** Estas dos detecciones comparten una línea gráfica común. A nivel de centro indica que no hay mueca, bajo nivel indica una mueca izquierda y alto nivel indica una mueca derecha.

- **Reírse (Laugh):** El nivel bajo indica que la expresión no se ha detectado, de alto nivel indica un nivel máximo de expresión detectado. El nivel gráfico aumentará o disminuirá en función del nivel de expresión detectado.

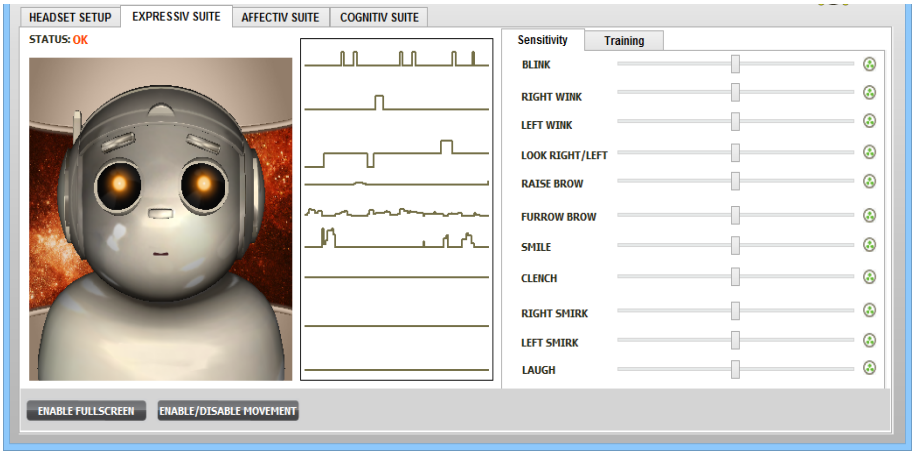


Figura 35: Expressiv Suite (Modo Expresivo)

Para una mejor apreciación de las expresiones faciales emitidas por el usuario, la pestaña de Modo Expresivo contiene dos pestañas adicionales para configurar diferentes parámetros y están ubicadas en la parte derecha, las cuales son: panel de sensibilidad y el panel de entrenamiento. El panel de

sensibilidad para el modo expresivo ofrece ajustes de sensibilidad para las detecciones de las expresiones faciales; esto se controla a través de controles deslizantes a la derecha del gráfico correspondiente. La sensibilidad puede ser aumentada o disminuida moviendo el control deslizante a la derecha o izquierda, respectivamente.

Para cada expresión facial, se debe comprobar el rendimiento de la detección, si el usuario siente que la detección no responde fácilmente a una expresión particular, es necesario aumentar la sensibilidad de esa expresión; por otro lado si el usuario siente que es demasiado fácil para desencadenar una expresión particular, o que están existiendo "falsos positivos", se debe disminuir la sensibilidad de esa expresión

Para el panel de entrenamiento del modo expresivo, se requiere que el usuario entrene al sistema mediante la ejecución de la acción deseada antes de que pueda ser detectada. A medida que el usuario proporciona más datos de entrenamiento, la exactitud de la detección mejora. Al usar este panel el sistema sólo detectará las acciones para las que el usuario ha suministrado datos de entrenamiento.

2.2.4 Affectiv Suite (Modo Afectivo)

El modo afectivo reporta cambios en tiempo real en las emociones subjetivas experimentadas por el usuario. El casco Emotiv actualmente ofrece tres detecciones afectivas distintas: concentración, entusiasmo instantáneo, y la emoción a largo plazo. Las detecciones afectivas buscan características de ondas cerebrales que son universales en la naturaleza y no requieren un entrenamiento previo por parte del usuario.

El panel de modo afectivo (ver Figura 36) contiene dos gráficos que se pueden personalizar para mostrar diferentes combinaciones de detecciones y escalas de tiempo. Por defecto, el gráfico superior está configurado para trazar los 30 segundos de datos para la concentración y las detecciones de entusiasmo instantáneos presentes en el usuario. Los valores que se dibujan

en los gráficos son los valores de salida devueltos por las detecciones afectivas que presenta el usuario.

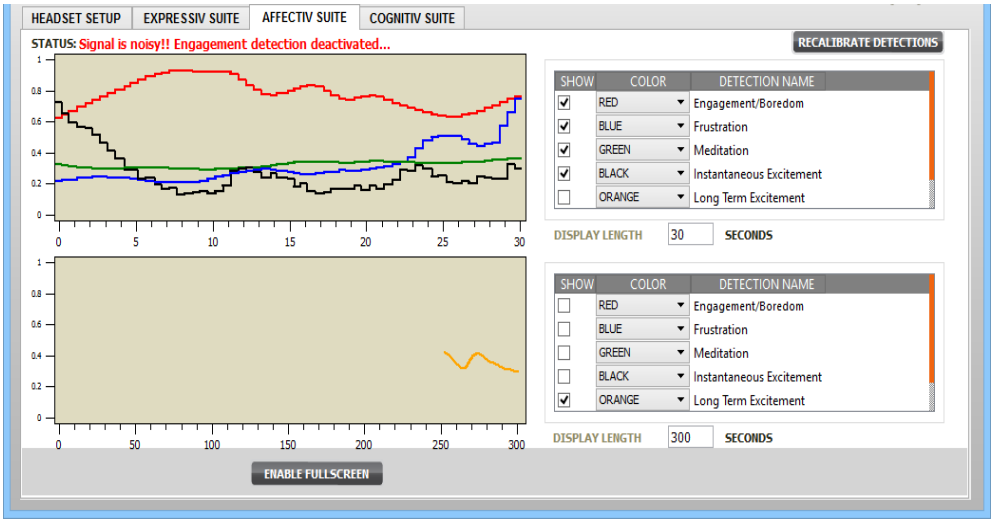


Figura 36: Affectiv Suite (Modo Afectivo)

La emoción instantánea se experimenta como una conciencia o sentimiento de excitación fisiológica con un valor positivo. La excitación se caracteriza por la activación en el sistema nervioso simpático, que se traduce en una gama de respuestas fisiológicas incluyendo la dilatación de la pupila, ensanchamiento del ojo, estimulación de la glándula de sudor, frecuencia cardiaca alta, la desviación de la sangre. En general, cuanto mayor es el aumento de la excitación fisiológica mayor será el valor de salida para la detección.

La emoción a largo plazo se experimenta y se define de la misma manera que emoción instantánea, pero la detección se ha diseñado y ajustado para ser más precisos en la medición de acuerdo a los cambios en la excitación durante períodos de tiempo más largos, por lo general se mide en minutos. La concentración se experimenta como el estado de alerta y la dirección consciente de la atención hacia los estímulos relevantes para una tarea específica. Se caracteriza por el aumento de la excitación y ondas fisiológicas beta (un tipo bien conocido de la forma de onda EEG), junto con las ondas alfa atenuadas (otro tipo de forma de onda EEG); el polo opuesto de esta detección se conoce como "aburrimiento".

2.2.5 Cognitiv Suite (Modo Cognitivo)

El modo de detección cognitivo evalúa en tiempo real la actividad de ondas cerebrales del usuario, para discernir la intención consciente del usuario para realizar acciones físicas distintas sobre un objeto real o virtual. La detección está diseñada para trabajar con 13 acciones diferentes: 6 movimientos direccionales (empujar, halar, desplazar hacia la izquierda, desplazar hacia la derecha, desplazar hacia arriba y desplazar hacia abajo), 6 rotacionales (rotar en sentido horario, rotar en sentido anti horario, rotar hacia la izquierda, rotar hacia la derecha, rotar hacia adelante, rotar hacia atrás) y por ultimo una acción adicional que sólo existe en el reino de la imaginación del usuario el cual es desaparecer.

El panel de acciones cognitivas permite al usuario elegir hasta cuatro acciones que pueden ser reconocidos en cualquier momento dado. La detección informa de una sola acción o neutral (es decir, ninguna acción) a la vez, junto con un poder de acción que representa la certeza de la detección de que el usuario ha entrado en el estado cognitivo asociado con esa acción.

Al aumentar el número de acciones concurrentes incrementa la dificultad de mantener un control consciente sobre los resultados de la detección cognitiva del usuario. Casi todos los nuevos usuarios pueden obtener fácilmente el control de una sola acción con bastante rapidez. Sin embargo aprender a controlar múltiples acciones típicamente requiere práctica y se hace cada vez más difícil a medida que se añaden acciones adicionales.

La Figura 37 indica que el modo cognitivo utiliza un cubo virtual en 3D para mostrar una representación animada para la detección cognitiva. Este cubo también se utiliza para ayudar al usuario en la visualización de la acción prevista durante el proceso de entrenamiento. El indicador de poder de la izquierda de la pantalla muestra la cantidad de concentración del usuario para realizar una acción cognitiva.

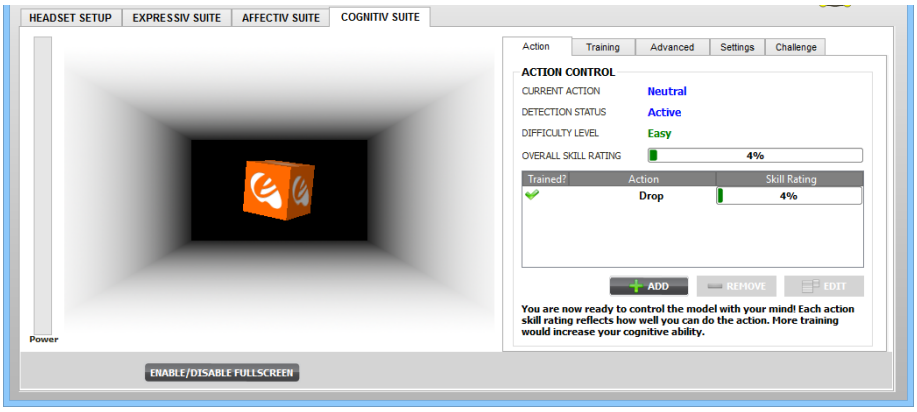


Figura 37: Cognitiv Suite (Modo Cognitivo)

La pestaña predeterminada para el modo cognitivo es la pestaña de acción (Action), la misma que muestra información sobre el estado actual de la detección cognitiva y permite al usuario definir el conjunto actual de las acciones. Junto a cada acción seleccionada existe otro indicador que muestra un índice de habilidad, esta calificación de habilidad se calcula durante el proceso de formación y proporciona una medida de cómo constantemente el usuario puede realizar mentalmente la acción prevista.

El proceso de entrenamiento (Training) para el modo cognitivo permite al EmoEngine analizar las ondas cerebrales emitidas por el casco y desarrollar un perfil personalizado correspondiente a cada acción en particular. A medida que el EmoEngine aprende y perfecciona los perfiles para cada una de las acciones las detecciones se hacen más precisas y fáciles de realizar.

El proceso de entrenamiento consta de tres pasos: en primer lugar, se debe seleccionar una acción de la lista desplegable para acciones que ya han sido capacitados se visualizan con una marca verde; acciones sin datos de entrenamiento se muestran con una “X” roja. Sólo las acciones que se han seleccionado en la pestaña de acción están disponibles para el entrenamiento.

A continuación cuando empiece a imaginar o visualizar la acción que desea entrenar es muy importante mantener su enfoque mental durante el período de entrenamiento de 8 segundos. Es importante abstenerse de hacer

movimientos de la cabeza sustanciales o expresiones faciales dramáticos durante el periodo de formación, ya que estas acciones pueden interferir con la señal del EEG registrado.

Finalmente, se le pedirá que acepte o rechace el registro del entrenamiento, para la detección ideal se consigue normalmente mediante el suministro de datos de entrenamiento constantes a través de varias sesiones de entrenamiento para cada acción habilitada. La capacidad de rechazar la última grabación de entrenamiento permite verificar si el usuario es capaz de mantenerse mentalmente enfocado en la acción apropiada durante la última sesión de entrenamiento.

El éxito del entrenamiento se basa en la coherencia y el enfoque, para obtener mejores resultados se debe realizar la acción prevista de forma continua durante el período de entrenamiento completo. Es común que los usuarios novatos se distraen en algún momento durante el período de formación y luego reiniciar mentalmente una acción, pero esta práctica se traducirá en resultados pobres de la capacitación.

La pestaña de opciones avanzadas (Advanced) expone ajustes y controles que le permiten personalizar el comportamiento de la detección cognitiva. Por defecto, la detección cognitiva está pre-configurada de una manera que produce los mejores resultados para la mayor población de usuarios.

La pestaña de ajustes (Settings) se la puede utilizar para personalizar y controlar la importación de objetos 3D, para ello es necesario importar el modelo, modificar la escala, y organizar el fondo para crear un paisaje personalizado, con el fin de controlar cualquier objeto 3D. Adicionalmente el panel de Modo Cognitivo presenta una pestaña de reto (Challenge) que le permite al usuario entrenarse con el modelo del cubo y enviar su puntuación al sitio Web de Emotiv.

2.2.6 El Programa EmoComposer

La función del programa EmoComposer es emular el funcionamiento del casco Emotiv, con el objetivo de generar actualizaciones de estados con información del estado de la batería, estado del nivel de señal inalámbrica, la calidad de contacto de cada uno de los electrodos y finalmente las acciones de los tres modos que proporciona el casco como son: el modo expresivo, el modo afectivo y modo cognitivo.

El programa EmoComposer puede establecer comunicación tanto con el Emotiv Control Panel y con el programa EmoKey, incluso la API provista por el casco dispone de una función para establecer dicha comunicación, lo cual es sumamente útil para el desarrollo del proyecto, debido que EmoComposer permite verificar el funcionamiento del programa que se está desarrollando en un entorno controlado, sin la necesidad de utilizar el casco.

El programa EmoComposer consta de dos pestañas, las cuales son: EmoScript (Guiones) e Interactive (Interactivo); dichas pestañas tienen un modo de funcionalidad diferente. La opción de trabajar con Guiones (scripts) del programa EmoComposer permite cargar un fichero con un evento predeterminado, este fichero puede ser codificado utilizando el lenguaje de modelado educativo (EML). Mediante el control deslizable que se encuentra en la parte superior de esta pestaña se podrá avanzar y retroceder los eventos del fichero, mientras se puede visualizar los cambios que se producen en el resto del emulador según las acciones establecidas en el fichero.

El modo interactivo (Interactive) permite configurar manualmente el estado Emotiv que el usuario quiera emular, además se puede modificar tanto el estado de carga de la batería y también el nivel de la señal de comunicación inalámbrica. Dentro de esta pestaña también se puede programar un envío repetitivo de datos o eventos, fijando el intervalo deseado de repetición y también el usuario que los origina. Esta pestaña está a su vez dividida en dos

sub pestañas: Contact Quality (Calidad de los Contactos) y Detection (Detección).

Como se indica en la Figura 38, la pestaña Contact Quality permite modificar la calidad de los contactos con el fin de simular situaciones donde el casco no se encuentra colocado de una manera correcta, así como también si alguno de los electrodos no tiene un buen contacto con el cuero cabelludo, pudiéndose modificar el estado de cada uno de los electrodos.

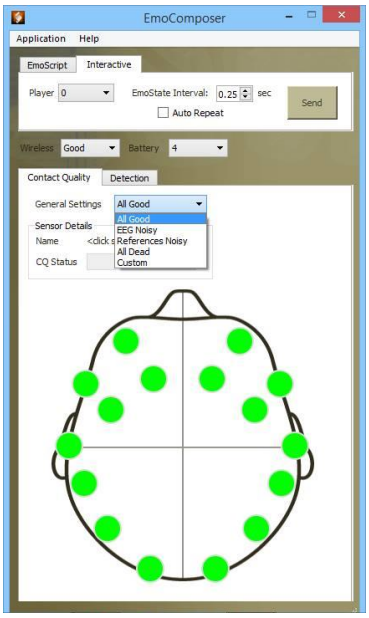


Figura 38: Pestaña de Contact Quality del programa EmoComposer

Finalmente se tiene la pestaña de Detection (Detección) donde se pueden encontrar a todas las diferentes acciones correspondientes a los tres modos de funcionamiento que posee el casco junto con su respectiva marca temporal del evento que se desea simular. Lo más importante de este modo es que permite programar un resultado para una acción de entrenamiento en específico, para los dos modos que lo admiten como son el modo expresivo y el modo cognitivo; adicionalmente esta pestaña dispone de un área de texto que indica el registro de la interacción del emulador y del programa cliente conectado al EmoComposer. La Figura 39 muestra a la pestaña anteriormente mencionada.

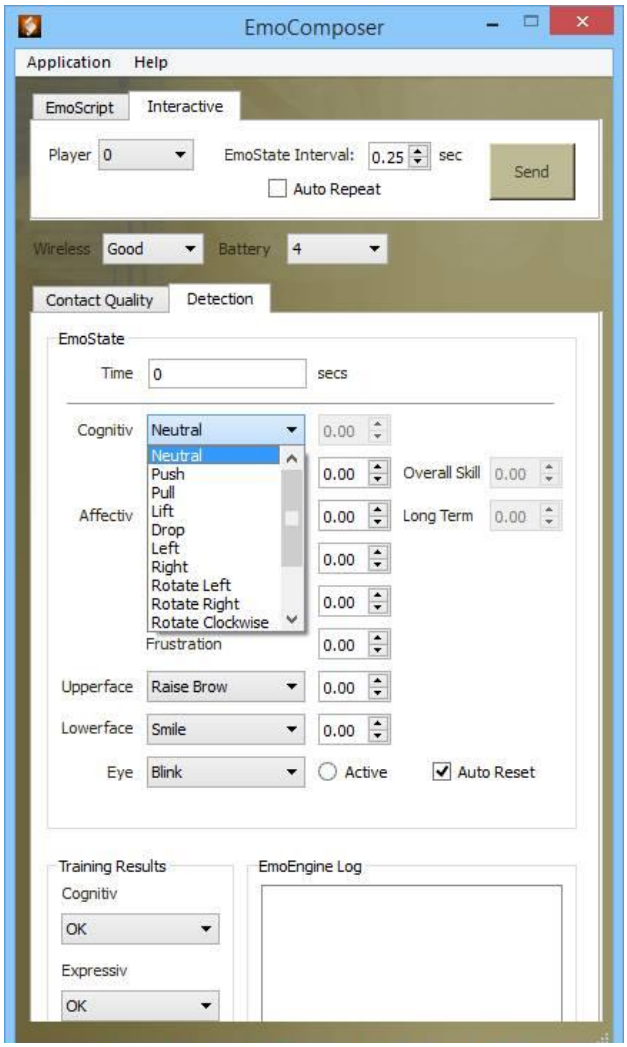


Figura 39: Pestaña Detection del programa EmoComposer

2.2.7 El Programa EmoKey

El programa EmoKey permite asignar a las acciones detectadas por el EmoEngine o bien a las acciones simuladas por el programa EmoComposer una determinada secuencia establecida por pulsaciones de teclas llamadas EmoKey Mapping (Mapeo EmoKey), las mismas que pueden ser enviadas a cualquier programa de Microsoft Windows que se encuentre en estado de ejecución.

En la Figura 40 se puede visualizar a la ventana del programa EmoKey, cabe indicar que con el fin de poder seleccionar el usuario y variar los parámetros necesarios para tener una buena detección de las acciones, la

conexión entre el programa EmoKey y el EmoEngine se la realiza mediante el Emotiv Control Panel.

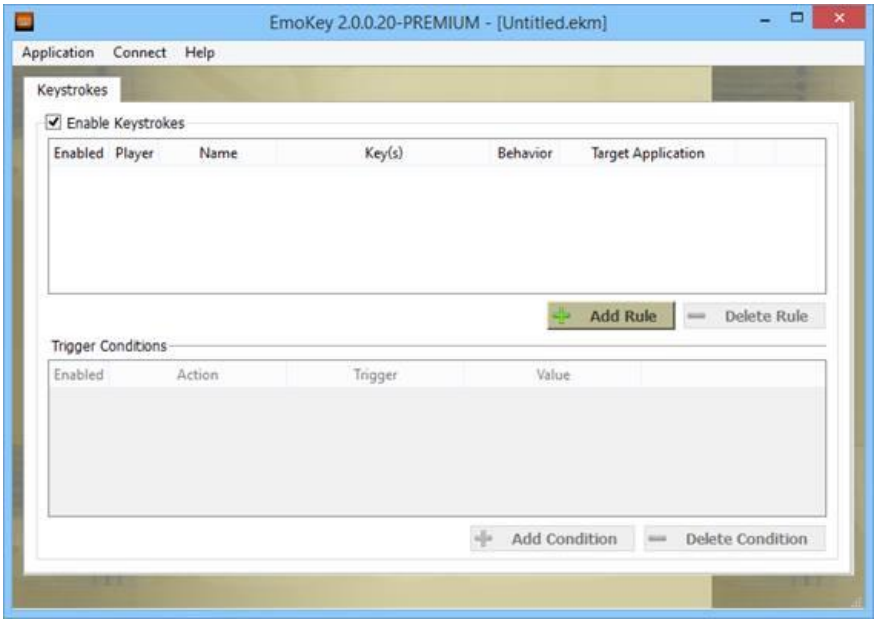


Figura 40: Programa EmoKey

2.3 Brazo robótico de 6 grados de libertad

Como se mencionó anteriormente para el desarrollo del presente proyecto se hará uso un brazo robótico de 6 grados de libertad proporcionado por la Universidad de las Fuerzas Armadas Extensión Latacunga, el mismo que esta ensamblado con servomotores Dynamixel de las series: MX-28, MX-64 y MX-106, cada uno de los servomotores que cuenta con una estructura hecha a su medida para la unión y soporte de todo el sistema. La estructura necesariamente debe ser lo suficientemente resistente para soportar el peso y fuerza de los servomotores y además tiene que ser ligera para que el peso total del brazo robótico sea el ideal para un óptimo funcionamiento del mismo.

Las piezas y uniones que conforman la estructura del brazo robótico fueron diseñadas en la Universidad de las Fuerzas Armadas ESPE Extensión Latacunga para posteriormente ser fabricadas en la ciudad de Quito-Ecuador. La Figura 41 muestra el diseño inicial del brazo robótico con las piezas manufacturas por la Universidad; después de un proceso de pruebas el diseño final para el brazo robótico es el indicado en la Figura 42.

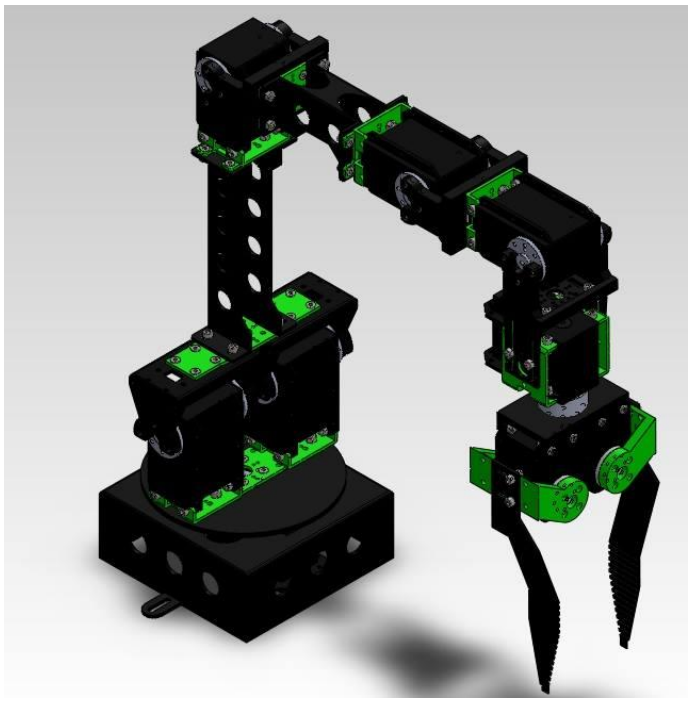
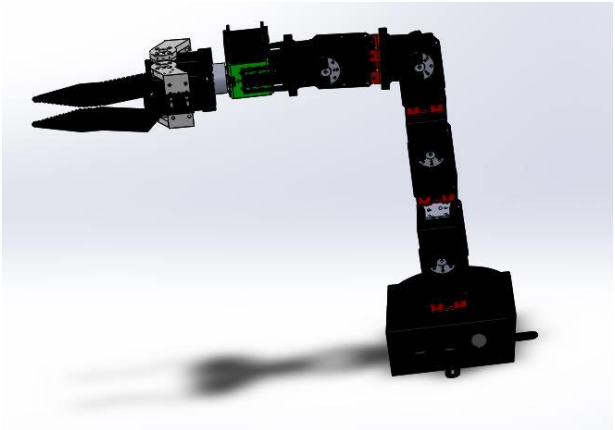


Figura 41: Diseño inicial para el brazo robótico de 6 grados de libertad

Fuente: (Velasco Sánchez & Mamarandi Quilo, 2015)



(a)

(b)

Figura 42: a) Diseño final para el brazo robótico de 6 grados de libertad; b) Estructura real del brazo robótico de 6 grados de libertad

2.3.1 Red se servomotores Dynamixel

Con el brazo robótico previamente ensamblado y fijo a una base lo suficientemente fuerte y estable, se procede a energizar adecuadamente a la red de servomotores Dynamixel, los mismos que funcionan correctamente con un voltaje de 11,1v a 14,8v y su consumo de corriente es de 1,7A, 5,2A y 6,3A para los modelos de la serie MX-28, MX-64 y MX-106 respectivamente. Por las características de consumo tanto de corriente como de voltaje de los servomotores se utilizará una fuente de voltaje marca Agilent modelo U8001A (ver Figura 43). Una de las características principales del funcionamiento de la fuente de voltaje es que permite al usuario configurar los valores tanto de voltaje como de corriente asiéndola perfecta para un óptimo desempeño de la red de servomotores.

Figura 43: Fuente AGILENT U8001A

2.3.2 Interfaz USB2Dynamixel

La comunicación entre la red de servomotores de marca Dynamixel y la PC, será mediante el uso de la interfaz USB2Dynamixel, la misma que debe ser conectada a un puerto USB del ordenador para poder establecer el enlace requerido. La interfaz está incorporada con dos conectores uno de 3P (3 pines) y otro de 4P (4 pines), con la finalidad de conectar varios dispositivos de la misma marca (ver Figura 44).

Figura 44: Interfaz USB2Dynamixel

Las principales partes presentes en la interfaz USB2Dynamixel son enumeradas a continuación:

- **Pantalla de estado LED:** Muestra el estado de la fuente de alimentación, TxD (escritura de datos), y RxD (lectura de datos).

- **Interruptor de selección de funciones:** Selecciona el modo TTL, RS-485 o RS-232.

- **Conector 3P:** Se conecta con dispositivos de la serie AX Dynamixel utilizando la red TTL.

- **Conector 4P:** Se conecta con dispositivos de la serie MX Dynamixel utilizando RS-485.

- **Conector serie:** Transforma un puerto USB en un puerto serie utilizando la red RS-232.

Para una correcta comunicación entre la red de servomotores y el PC utilizando la interfaz USB2Dynamixel se debe seguir la configuración ilustrada en la Figura 45.

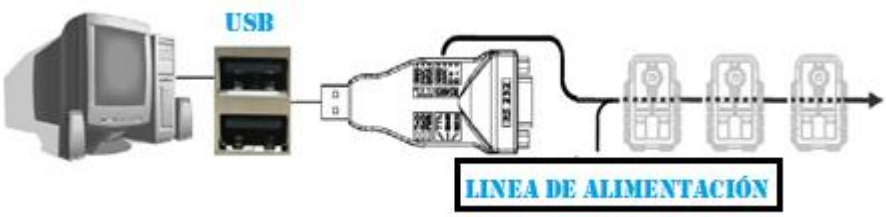


Figura 45: Comunicación mediante la interfaz USB2Dynamixel

Como indica la Figura 46, la interfaz USB2Dynamixel cuenta con un switch que permite seleccionar al usuario el protocolo de comunicación entre una computadora y uno o varios servomotores Dynamixel, de esta manera la interfaz activa el puerto de conexión seleccionado y configura los niveles de voltaje necesarios para establecer la comunicación requerida, de igual forma acondiciona a las señales emitidas por los servomotores Dynamixel para que el ordenador los pueda interpretar correctamente.

Figura 46: Switch de selección para el protocolo de comunicación del USB2Dynamixel

2.3.3 Conexión para servomotores Dynamixel serie AX

Para la conexión de los servomotores Dynamixel de la serie AX con el USB2Dynamixel, primero se debe ajustar el interruptor de selección de función para el modo TTL, a continuación se requiere conectar el cable 3P proveniente de los servomotores al conector 3P de la interfaz USB2Dynamixel. Los servomotores Dynamixel se pueden conectar en serie

mediante un cable 3P, teniendo en cuenta que se debe conectar la línea de alimentación DC al último servomotor (Velasco Sánchez & Mamarandi Quilo, 2015).

2.3.4 Conexión para servomotores Dynamixel serie MX/RX

Debido al uso de los servomotores para el desarrollo del presente proyecto, la conexión entre la red de servomotores Dynamixel y la interfaz USB2Dynamixel es la siguiente, primero se debe ajustar el interruptor de selección de función para el modo RS-485, a continuación se debe conectar el cable 4P proveniente de los servomotores al conector 4P de la interfaz USB2Dynamixel (el cable 4P se puede conectar en cualquiera de los dos conectores existentes en los servomotores). Los servomotores Dynamixel pueden ser conectados en serie mediante un cable 4P.

Finalmente se debe conectar la línea de alimentación al último servomotor Dynamixel, cabe mencionar que la fuente debe ser configurada para valores de voltaje entre 10v a 12v para los modelos RX-10, de 12v a 16v para modelos RX-28, de 15v a 18v para modelos RX-64 y de 12v a 16v para los modelos MX-117.

2.4 Comunicación de Python con el casco y el brazo

Para establecer la comunicación entre Python con el casco Emotiv EPOC y el brazo robótico ensamblado con servomotores Dynamixel es necesario la declaración de librerías al iniciar el código de programación; estas librerías son implementaciones funcionales que permiten enlazar funciones para desarrollar un determinado programa. La importación de librerías es una manera sencilla de estructurar el código de programación.

La conexión entre el brazo robótico y Python se la realiza mediante la biblioteca de enlace dinámico **windll.dynamixel**, esta librería es la encargada de acceder tanto a la memoria RAM como a la EPROM de cada uno de los servomotores Dynamixel que conforman el brazo robótico, de esta manera se puede leer o escribir una serie de datos en la celda de memoria cada

servomotor. En la Figura 47 se puede observar varios de los parámetros de la memoria RAM y le memoria EPROM de cada uno de los servomotores Dynamixel.

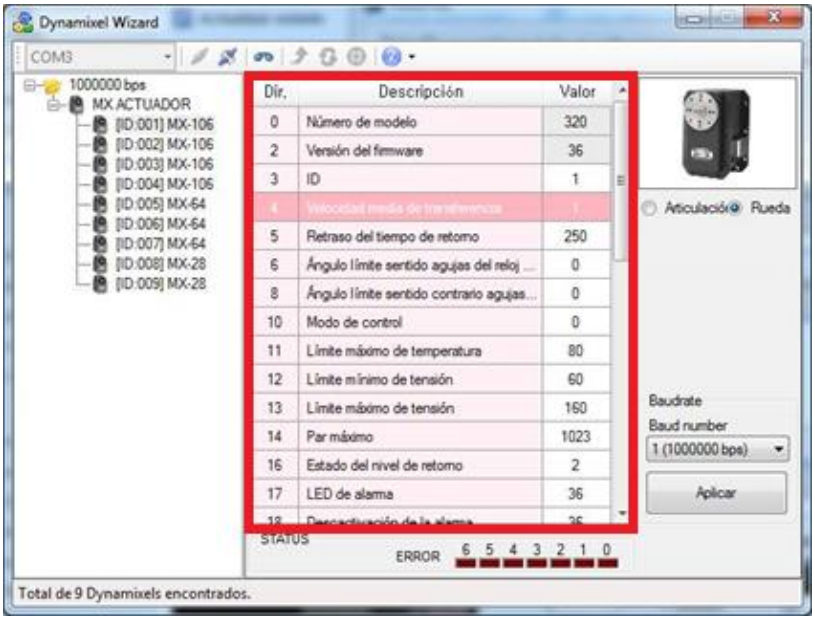


Figura 47: Parámetros de las memorias RAM y EPROM para los servomotores Dynamixel

Por otro lado, la librería encargada de enlazar a Python con el casco es: **cdll.LoadLibrary (".\\edk.dll")**, la misma que permite acceder a la API de Emotiv que a su vez consta básicamente de 3 ficheros de cabecera (edk.h, EmoStateDLL.h y edkErrorCode.h) y dos DLL's para la plataforma Microsoft Windows (edk.dll y edk_utils.dll). Los 3 ficheros mencionados anteriormente son los responsables de la comunicación o enlace entre en casco Emotiv EPOC con determinadas plataformas de Windows.

Las principales funciones para la comunicación del casco con Python se describen a continuación:

- **EE_EmoEngineEventCreate.-** Esta función devuelve un puntero a un espacio de memoria para contener un evento de EmoEngine (es el componente base para la detección e interpretación de las señales provenientes de los electrodos que se encuentran situados en el casco y que capturan la información del electroencefalograma). Este

puntero podrá ser reutilizado para obtener los eventos que se producen durante la conexión.

- **EE_EmoEngineEventGetEmoState.-** Copia un Estado Emotiv (estructura de datos que contiene información sobre la detección del estado del casco, la expresión facial, el estado emocional y el estado cognitivo del usuario) generado cuando se produce un evento **EE_EmoStateUpdate** y suministrado como primer parámetro en la memoria referenciada por el puntero pasado como segundo parámetro, que a su vez debe haber sido inicializado por una llamada a la función **EE_EmoStateCreate.**

- **EE_EmoStateCreate.-** Esta función devuelve un valor de tipo **EDK_ERROR_CODE** que toma el valor **EDK_OK** si se ha ejecutado con éxito.

- **ES_GetTimeFromStart.-** Obtiene el tiempo transcurrido desde el inicio de la conexión entre EmoEngine y el casco. Si se produce la desconexión entre el casco y el EmoEngine debido a un nivel bajo de batería o a la debilidad de la señal inalámbrica, el tiempo de conexión se inicializará a cero.

- **ES_GetWirelessSignalStatus.-** Obtiene el estado de la señal inalámbrica.

- **ES_ExpressivIsBlink.-** Consulta si el usuario se encontraba parpadeando en el momento en que el estado Emotiv fue capturado.

- **ES_ExpressivIsLeftWink.-** Consulta si el usuario se encontraba guiñando el ojo izquierdo. Devuelve un 1 si el ojo izquierdo se encontraba guiñado y un 0 en caso contrario.

- **ES_ExpressivIsRightWink.-** Consulta si el usuario se encontraba guiñando el ojo derecho. Devuelve un 1 si el ojo derecho se encontraba guiñado y un 0 en caso contrario.

- **ES_ExpressivIsLookingLeft.-** Consulta si el usuario se encontraba mirando hacia la izquierda. Devuelve un 1 si miraba hacia la izquierda y un 0 en caso contrario.

- **ES_ExpressivIsLookingRight.-** Consulta si el usuario se encontraba mirando hacia la derecha. Devuelve un 1 si miraba hacia la derecha y un 0 en caso contrario.

- **ES_ExpressivGetUpperFaceAction.-** Obtiene la acción facial en modo expresivo detectada a la región superior del rostro del usuario.

- **ES_ExpressivGetUpperFaceActionPower.-**Obtiene el nivel de potencia de la acción facial en modo expresivo de la región superior del rostro del usuario.

- **ES_ExpressivGetLowerFaceAction.-** Obtiene la acción facial en modo expresivo detectada en la región inferior del rostro del usuario.

- **ES_ExpressivGetLowerFaceActionPower.-** Obtiene el nivel de potencia de la acción facial en modo expresivo de la región inferior del rostro del usuario. Devuelve un nivel de potencia de la acción facial en modo expresivo en un rango entre 0.0 y 1.0.

- **ES_AffectivGetExcitementShortTermScore.-** Obtiene el nivel de emoción a corto plazo del usuario. Devuelve el nivel de emoción en un rango entre 0.0 y 1.0.

- **ES_AffectivGetExcitementLongTermScore.-** Obtiene el nivel de emoción a largo plazo del usuario. Devuelve el nivel de emoción en un rango entre 0.0 y 1.0.

- **ES_AffectivGetEngagementBoredomScore.-** Obtiene el nivel de atención/aburrimiento del usuario. Devuelve el nivel de atención/aburrimiento en un rango entre 0.0 a 1.0.

- **ES_CognitivGetCurrentAction.-** Obtiene la acción detectada en modo cognitivo del usuario. Devuelve el tipo de acción en modo cognitivo.

- **ES_CognitivGetCurrentActionPower.-** Obtiene el nivel de potencia de la acción en modo cognitivo detectada del usuario.

CAPÍTULO III

3. IMPLEMENTACIÓN DEL SISTEMA

El control del brazo robótico a realizar es mediante el uso de las señales cerebrales obtenidas por medio del casco Emotiv EPOC, las mismas que se transforman en posición hacia el extremo operativo del brazo y al ser enviadas al robot ejecuta el movimiento requerido por el usuario. Desde el punto de vista de control el extremo operativo del brazo robótico debe seguir la trayectoria deseada que es generada a través de señales cerebrales, obtenidas por gestos faciales.

3.1 Modelamiento cinemático del brazo robótico

El operador humano controla el brazo robótico mediante el envío de comandos de posición hacia el extremo operativo del robot: $h_l,\ h_m, h_n$, uno por cada eje con respecto al marco inercial $R(X, Y, Z)$ utilizando un dispositivo háptico. Los comandos del operador humano P_x, P_y, P_z son generados con el uso del dispositivo Emotiv EPOC como se indica en la Figura 48.

$$\boldsymbol{h_d} = [h_l \quad h_m \quad h_n]^T \tag{3.1}$$

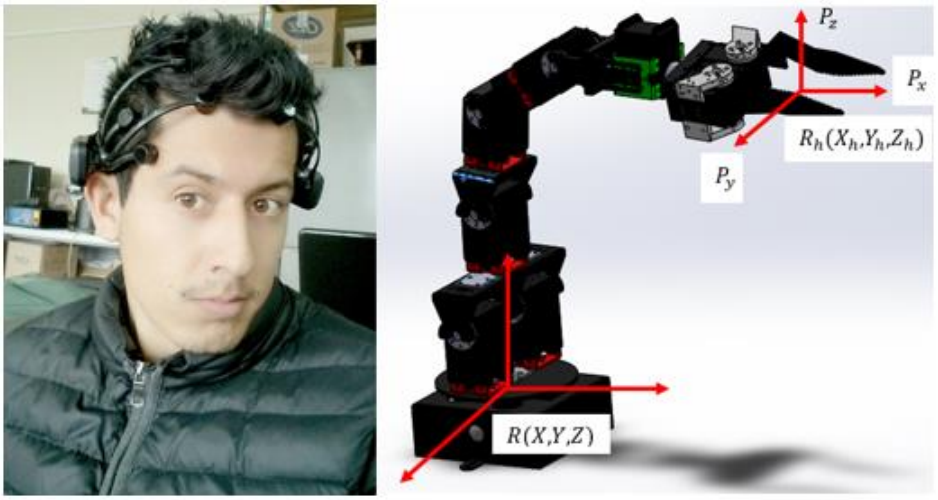


Figura 48: Comandos generados por el operador humano hacia el brazo robótico

Las posiciones P_x, P_y, P_z son traducidas en comandos de posición del extremo operativo h_l, h_m, h_n para el modo de manipulación, a través de la siguiente matriz de rotación.

$$\begin{bmatrix} h_l \\ h_m \\ h_n \end{bmatrix} = \begin{bmatrix} cos(q_1) & -sin(q_1) & 0 \\ sin(q_1) & cos(q_1) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} \tag{3.2}$$

donde q_1 representa la primera articulación del brazo robótico que gira alrededor del eje Z.

La configuración del brazo robótico es definida por el vector $\boldsymbol{q}(t)$ de n coordenadas independientes, llamadas coordenadas generalizadas del brazo robótico, donde $\boldsymbol{q} = [q_1 \quad q_2 \ \ldots \ q_n]^T$ representa las coordenadas generalizadas del brazo robótico. La ubicación del extremo operativo del brazo robótico respecto a $R(X, Y, Z)$ está dado por el vector m-dimensional donde $\boldsymbol{h} = [h_1 \quad h_2 \ \ldots \ h_m]^T$ que define la posición y la orientación del extremo operativo del brazo robótico; por lo tanto la relación entre las posiciones de las articulaciones y la posición del extremo operativo es,

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = f(\, q_1, q_2, q_3, \ q_4, q_5, \ldots . q_n) \tag{3.3}$$

A continuación se detalla el modelo de cinemática directa para un brazo robótico de seis grados de libertad que viene dado por,

$$\begin{cases} h_x = \ l_2 \cos q_2 \cos q_1 + \ l_3 \cos(q_2 + q_3) \cos q_1 + \ l_4 \cos(q_2 + q_3 \ + \ q_4) \cos q_1 + \\ \qquad l_5 \cos(q_2 + q_3 \ + \ q_4 + \ q_5) \cos q_1 \\ h_y = l_2 \cos q_2 \, sen \, q_1 + \ l_3 \cos(q_2 + q_3) \, sen \, q_1 + \ l_4 \cos(q_2 + q_3 \ + \ q_4 \,) \, sen \, q_1 + \\ \qquad l_5 \cos(q_2 + q_3 \ + \ q_4 + \ q_5 \,) \, sen \, q_1 \\ h_z = \ l_1 + \ l_2 \, sen \, q_2 + \ l_3 \, sen(q_2 + q_3) + \ l_4 \, sen(q_2 + q_3 + q_4) \ + \\ \qquad l_5 \, sen(q_2 + q_3 + q_4 + q_5) \end{cases} \tag{3.4}$$

Por otro lado, el modelo cinemático instantáneo de un brazo robótico está definido por la derivada de la ubicación del extremo operativo como una función de las derivadas de la configuración del brazo robótico.

$$\dot{\boldsymbol{h}}(t) = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}(t) \tag{3.5}$$

donde $\boldsymbol{J}(\boldsymbol{q})$ es la matriz Jacobiana que define un operador lineal entre el vector de las velocidades de las articulaciones del brazo robótico $\dot{\boldsymbol{q}}(t)$ y el vector de velocidad del extremo operativo $\dot{\boldsymbol{h}}(t)$. Por lo tanto la relación entre las velocidades para el brazo de 6DOF es,

$$\dot{\boldsymbol{h}} = \boldsymbol{J}(\boldsymbol{q}) \begin{bmatrix} \dot{q_1} \\ \dot{q_2} \\ \dot{q_3} \\ \dot{q_4} \\ \dot{q_5} \end{bmatrix}$$

con

$$\dot{\boldsymbol{h}} = \begin{bmatrix} a & b & c & d & e \\ f & g & h & i & j \\ 0 & k & l & m & n \end{bmatrix} \begin{bmatrix} \dot{q_1} \\ \dot{q_2} \\ \dot{q_3} \\ \dot{q_4} \\ \dot{q_5} \end{bmatrix}$$

$$a = -senq_1[l_2 \cos q_2 + l_3 \cos(q_2 + q_3) + l_4 \cos(q_2 + q_3 + q_4) + l_5 \operatorname{sen}(q_2 + q_3 + q_4 + q_5)]$$

$$b = -cosq_1[l_2 senq_2 + l_3 \operatorname{sen}(q_2 + q_3) + l_4 sen(q_2 + q_3 + q_4) + l_5 \operatorname{sen}(q_2 + q_3 + q_4 + q_5)]$$

$$c = -cosq_1[l_3 sen(q_2 + q_3) + l_4 sen(q_2 + q_3 + q_4) + l_5 \operatorname{sen}(q_2 + q_3 + q_4 + q_5)]$$

$$d = -cosq_1[l_4 sen(q_2 + q_3 + q_4) + l_5 \operatorname{sen}(q_2 + q_3 + q_4 + q_5)]$$

$$e = -cosq_1[l_5 \operatorname{sen}(q_2 + q_3 + q_4 + q_5)]$$

$$f = cosq_1[l_2 cosq_2 + l_3 cos(q_2 + q_3) + l_4 cos(q_2 + q_3 + q_4) + l_5 cos(q_2 + q_3 + q_4 + q_5)]$$

$$g = -senq_1[l_2 senq_2 + l_3 sen(q_2 + q_3) + l_4 sen(q_2 + q_3 + q_4) + l_5 sen(q_2 + q_3 + q_4 + q_5)]$$

$$h = -senq_1[l_3 sen(q_2 + q_3) + l_4 sen(q_2 + q_3 + q_4) + l_5 sen(q_2 + q_3 + q_4 + q_5)]$$

$$i = -senq_1[l_4 sen(q_2 + q_3 + q_4) + l_5 sen(q_2 + q_3 + q_4 + q_5)]$$

$$j = -senq_1[l_5 sen(q_2 + q_3 + q_4 + q_5)]$$

$$k = l_2 cosq_2 + l_3 cos(q_2 + q_3) + l_4 cos(q_2 + q_3 + q_4) + l_5 cos(q_2 + q_3 + q_4 + q_5)$$

$$l = l_3 cos(q_2 + q_3) + l_4 cos(q_2 + q_3 + q_4) + l_5 cos(q_2 + q_3 + q_4 + q_5)$$

$$m = l_4 cos(q_2 + q_3 + q_4) + l_5 cos(q_2 + q_3 + q_4 + q_5)$$

$$n = l_5 cos(q_2 + q_3 + q_4 + q_5)$$

3.2 Control para el brazo robótico

El diseño del controlador cinemático del brazo robótico se basa en el modelo cinemático del brazo. En la Figura 49 se muestra la proyección ortogonal de un punto deseado para el cual se propone la siguiente ley de control.

$$\dot{\boldsymbol{q}}_{\boldsymbol{c}} = \boldsymbol{J}^{\#}\left(\boldsymbol{v}_{\boldsymbol{d}} + \boldsymbol{L}_{\boldsymbol{K}}\, \boldsymbol{tanh}\left(\boldsymbol{L}_{\boldsymbol{K}}^{-1}\boldsymbol{K}\tilde{\boldsymbol{h}}\right)\right) + (\boldsymbol{I} - \boldsymbol{J}^{\#}\boldsymbol{J})\boldsymbol{L}_{\boldsymbol{B}}\boldsymbol{tanh}\left(\boldsymbol{L}_{\boldsymbol{B}}^{-1}\boldsymbol{B}\Lambda\right) \qquad (3.6)$$

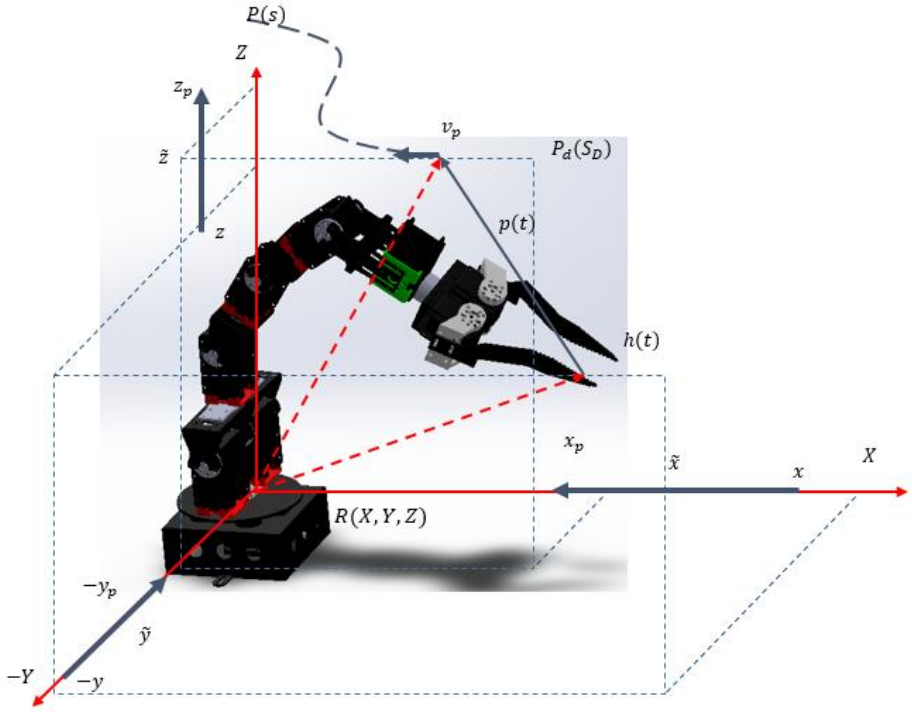


Figura 49: Proyección ortogonal del punto deseado sobre una trayectoria

donde $\boldsymbol{J^{\#} = W^{-1}J^{T}(JW^{-1}J^{T})^{-1}}$**,** siendo $\boldsymbol{W}$ una matriz positiva definida que pesa las acciones de control del sistema, $\boldsymbol{v_d}$ es el vector velocidad deseado del extremo operativo $\boldsymbol{h}$, $\boldsymbol{\tilde{h}}$ es el vector de errores de control, definido como $\boldsymbol{\tilde{h} = h_d - h}$, $\boldsymbol{B}$ y $\boldsymbol{L_B}$ son matrices diagonales positivas definidas que pesan el vector Λ. Con el fin de incluir una saturación analítica de velocidades en el brazo robótico se propone el uso de la función de $\boldsymbol{tanh}$(.), lo que limita el error en $\boldsymbol{\tilde{h}}$ y la magnitud del vector Λ. El segundo término de (3.6) representa la proyección sobre el espacio nulo de $\boldsymbol{J}$, donde Λ es un vector arbitrario que contiene las velocidades asociadas al brazo robótico. Por lo tanto, cualquier valor dado de Λ tendrá efectos solamente en la estructura interna del brazo, y no afectará el control final del extremo operativo en absoluto. Mediante el uso de este término, diferentes objetivos de control secundario se puede lograr de manera efectiva (Andaluz, Ortiz, & Sanchéz, 2015).

3.2.1 Índice de manipulabilidad

Cualquier robot dentro de su espacio de trabajo puede alcanzar cualquier punto, pero hay la posibilidad de que al tratar de alcanzar dicho punto sufra de esfuerzos mecánicos, con esto se indica que el robot puede dañar sus articulaciones al intentar alcanzar un punto aplicando mucha fuerza, estos puntos son denominados singulares. El índice de manipulabilidad ayuda a monitorear cuando el robot está próximo a alcanzar estos puntos singulares; el índice de manipulabilidad se describe como se muestra en la siguiente ecuación.

$$w = \sqrt{det(\boldsymbol{J}(\boldsymbol{q})\boldsymbol{J}^{\boldsymbol{T}}(\boldsymbol{q}))} \tag{3.7}$$

La ecuación (3.7) indica que cuando el índice de manipulabilidad se aproxima a cero es cuando el robot se acerca a los denominados puntos singulares.

3.3 Reconocimiento de servomotores Dynamixel

Para poder reconocer los servomotores se debe considerar lo siguiente:

- Es necesario tener la interfaz USB2Dynamixel conectada hacia la red de servomotores, de igual manera es muy importante que el switch que posee dicha interfaz se encuentre colocado en la comunicación RS-485 como se observa en el apartado 2.3.4.

- La fuente requerida para alimentar a la red de servomotores debe tener un voltaje de 12v de corriente continua.

El software Roboplus de la organización ROBOTIS, permite comprobar el funcionamiento de los servomotores Dynamixel, también se lo utiliza para funciones como la lectura y modificación de los datos de las memorias EPROM y RAM de cada servomotor. Además importante modificar la velocidad de transmisión de cada servomotor al valor de 1000000 bps, para que Python se pueda comunicar con la red de servomotores Dynamixel (Velasco Sánchez & Mamarandi Quilo, 2015).

3.4 Desarrollo de algoritmo de control en Python

Para facilitar la codificación tanto de la interfaz gráfica como del algoritmo de control, el código está estructurado mediante funciones las mismas que son accionadas mediante botones y checkbox ubicados estratégicamente en la interfaz gráfica desarrollada para interactuar con el usuario. La interfaz gráfica está compuesta con elementos (botones, checkbox, cajas de texto) que permiten el acceso al algoritmo de control previamente implementado para el brazo robótico, con el fin de realizar una trayectoria pre-establecida (trayectoria de una circunferencia o una silla de montar).

Además la interfaz es capaz de establecer un enlace de comunicación de Python y el casco Emotiv EPOC o Python y un Joystick; los dispositivos antes mencionados son los encargados de establecer una trayectoria mediante un control de posición para el extremo operativo del brazo. Cabe mencionar que las unidades utilizadas para la codificación del algoritmo de control del brazo robótico se encuentran en: $[m]$, $[rad]$ y $[rad/s]$; por tal razón a lo largo del programa desarrollado existen funciones con la finalidad de acondicionar a las unidades requeridas para el correcto desempeño del algoritmo.

3.4.1 Declaración de funciones

- **Comunicación ().-** Permite establecer la comunicación de la interfaz USB2Dynamixel con el software Python, de existir un error de comunicación no se ejecuta el programa y se indica con un mensaje que hubo error al abrir el USB2Dynamixel. Se iguala la variable n a la celda de memoria libc.dxl_initialize (), retorna 0 cuando no se ha establecido comunicación y 1 para cuando existió comunicación con el USB2Dynamixel. Adicionalmente sintoniza las ganancias del control PID de cada servomotor.

- **Rueda ().-** Por medio de las direcciones de memoria 6 y 8 de los servomotores Dynamixel se puede configurar para que funcionen en modo rueda (como motores) asignando a estas direcciones el valor de cero, a

continuación se configura la dirección 32 (velocidad de movimiento) para que cada uno de los servomotores se quede estático. Los valores entre 0-1023 hacen girar al servomotor en sentido antihorario y en el rango de 1024-2047 para el giro en sentido horario.

- **Arti ().-** Esta función es la encargada de configurar a los servomotores para que trabajen en modo articulación, para lo cual es necesario escribir el dato 0 en la dirección 6 y el valor de 4095 en la dirección 8 de cada servomotor Dynamixel.

- **Velocidad ().-** Asigna mediante la dirección 32 de cada uno de los servomotores un valor en el rango de 0-2047, para establecer la velocidad y sentido de giro.

- **Leer_servo ().-** Devuelve la posición en la que se encuentra cada uno de los servomotores, ingresando el nombre de la articulación de la que se requiere su posición, se debe tener en cuenta que la pinza está configurada con una apertura de cierre entre 0 y 100%.

- **Multiplicar ().-** Sirve para multiplicar matrices de n filas por m columnas.

- **Vel_servo ().-** Determina el sentido de giro y la velocidad de cada uno de los servomotores; con valores entre 0 a 1023 los servomotores giran en sentido horario, y de no ser este el caso se suma 1024 para que giren en sentido antihorario.

- **Bits ().-** Su objetivo es determinar el valor que tienen los servomotores, este valor es un dato en el rango de 0 y 4096 que permite determinar su equivalente de grados a radianes positivos en un rango de 0 a π, y de ser el caso que sobrepase el valor de π se trabaja con el mismo valor de signo negativo. La equivalencia de grados a radianes en el plano de trabajo del brazo robótico se puede notar en la Figura 50.

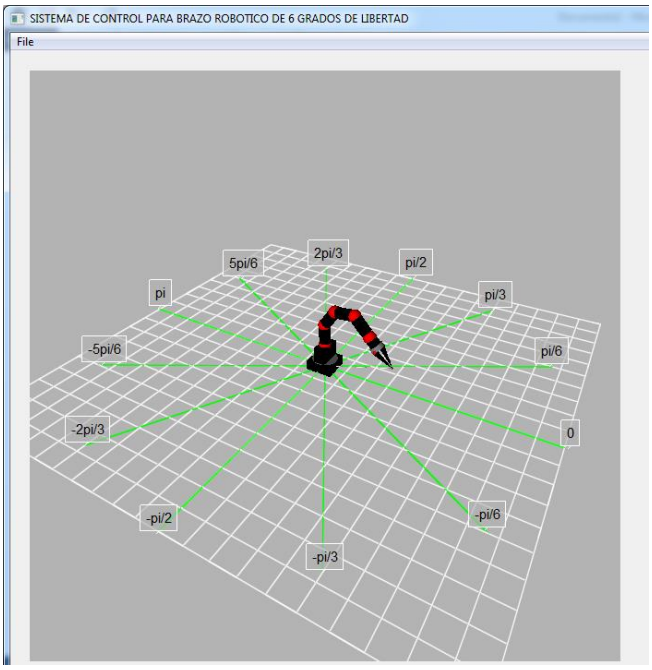


Figura 50: Grados expresados en radianes para el plano de trabajo

- **Trayectoria ().-** Cumple una determinada trayectoria, dicha trayectoria puede ser una circunferencia o una silla de montar.

- **Torque_todos ().-** Escribe en la dirección 24 de cada uno de los servomotores el valor de 1 para habilitar el torque y el valor de cero en la misma dirección para deshabilitar el torque.

- **Emoengine ().-** Habilita la comunicación con casco Emotiv, mientras que deshabilita la opción para usarlo el joystick.

- **Joys ()-** Habilita la comunicación con el joystick, mientras que deshabilita la opción para usarlo con el casco Emotiv EPOC.

- **Pinza ().-** Permite la apertura y cierre en un rango de 0 al 100% de la pinza de brazo robótico.

- **Munieca ().-** Es la encargada de que la muñeca del brazo gire en sentido horario o en sentido anti horario en un rango de 0 a 360°.

- **Circunferencia ().-** Obtiene los valores de radio y de altura para realizar la trayectoria de una circunferencia.

- **Silla ().-** Obtiene los valores de radio y de altura para realizar la trayectoria de una silla de montar.

3.4.2 Algoritmo de control desarrollado en Python

Para el algoritmo de control es necesario establecer ciertos valores de las condiciones iniciales correspondientes a los ángulos y longitudes en las que se encuentra el brazo robótico al iniciar una acción, las longitudes l_1 a la l_5 que se muestran en la Figura 51 son las medidas reales del brazo robótico las mismas que vienen dadas en metros.

```
#-------------------------------------------------------------
#ASIGNACIÓN DE CONDICIONES INICIALES
            rueda()
            k=0
            l1= 0.115
            l2= 0.095
            l3= 0.09
            l4= 0.09
            l5= 0.26
#-------------------------------------------------------------
#ALGORITMO DE CONTROL
            q1[0] = bits("Base")    # BASE DEL BRAZO
            q2[0] = bits("Art_1")   # PRIMERA ARTICULACIÓN
            q3[0] = bits("Art_2")   # SEGUNDA ARTICULACIÓN
            q4[0] = bits("Art_3")   # TERCERA ARTICULACIÓN
            q5[0] = bits("Art_4")   # CUARTA ARTICULACIÓN
```

Figura 51: Condiciones iniciales utilizadas para realizar el algoritmo de control

Como se indicó en la sección 3 para realizar el control se utiliza el modelamiento cinemático, el cual se encuentra realizado en el (Anexo B, Interfaz Final, línea 458 – línea 460), la asignación de $\mathbf{h_x[0]}$, $\mathbf{h_y[0]}$, $\mathbf{h_z[0]}$ corresponde a la posición inicial del extremo operativo, dichas ecuaciones se obtienen para cada una de las articulaciones que posee el brazo robótico.

Lo siguiente a realizar es calcular cada una de las componentes que conforman la matriz Jacobiana como indica la Figura 52, las mismas que

provienen del modelo cinemático previamente obtenido del brazo robótico (Anexo B, Interfaz Final, línea 483 - línea 487).

```
J= array([
     [J11,J12,J13,J14,J15],
     [J21,J22,J23,J24,J25],
     [J31,J32,J33,J34,J35],
     ])
```

Figura 52: Declaración de matriz Jacobiana

Como se puede visualizar en la Figura 53, es necesario de la creación de dos matrices, la una definida como **K** que en su diagonal principal posee valores de 0.5 y una matriz identidad denominada **W**. Es recomendable que los valores de la matriz **K** sean valores menores a 1.0 para un control estable del brazo robótico.

```
K = 0.5*identity(3,float)
W = identity(5,float)
```

Figura 53: Creación de matrices K y W

Las matrices **K** y **W** anteriormente mencionadas sirven para determinar una matriz **Jps**, la misma que resulta de la multiplicación de la matriz inversa de **W** por la matriz **J** transpuesta y la matriz inversa que resulta de la multiplicación entre **J**, la inversa de **W** y la traspuesta de **J**, esto se observa en las siguientes líneas de código mostradas en la Figura 54.

$$Jps = W^{-1}J^{t}(JW^{-1}J^{t})^{-1} \tag{3.8}$$

```
a1=multiplicar(3,5,J,W)
a2=multiplicar(3,3,a1,J.transpose())
a2=inv(a2)
a3=multiplicar(5,3,W,J.transpose())

jps=multiplicar(5,3,a3,a2)
```

Figura 54: Cálculo de la matriz Jps

Después de haber encontrado **Jps**, se debe hallar la matriz denominada **qp_ref** (3.9) la cual resulta de multiplicar las siguientes matrices:

$$qp\_ref = Jps\left(hd_p + Ktanh(herror)\right) + (I - JpsJ)Dtanh(n) \quad (3.9)$$

El código en Python para la obtención de la matriz $\boldsymbol{qp\_ref}$ se muestra en la Figura 55.

```
a4=multiplicar(5,5,jps,J)
a4=W-a4
a5=multiplicar(5,1,a4,tanh(n))
a6=multiplicar(3,1,K,tanh(herror))
a6=hd_p+a6

qp_ref=multiplicar(5,1,jps,a6)+a5
```

Figura 55: Cálculo de matriz $\boldsymbol{qp\_ref}$

Una vez realizado todos los cálculos mencionados anteriormente; las velocidades que son enviadas a cada servomotor que conforma una articulación del brazo robótico están detalladas en las siguientes líneas de código como se indica en la Figura 56, dichas velocidades indican el valor en $[rad/s]$ con que cada articulación debe girar para alcanzar un punto en el espacio.

```
velocidad_base = vel_servo (qp_ref[0])
velocidad("Base",velocidad_base)

velocidad_art1 = vel_servo (qp_ref[1])
velocidad("Art_1",velocidad_art1)

velocidad_art2 = vel_servo (qp_ref[2])
velocidad("Art_2",velocidad_art2)

velocidad_art3 = vel_servo (qp_ref[3])
velocidad("Art_3",velocidad_art3)

velocidad_art4 = vel_servo (qp_ref[4])
velocidad("Art_4",velocidad_art4)
```

Figura 56: Velocidades enviadas a las articulaciones del brazo robótico

3.5 Interfaz gráfica

La importancia de tener una interfaz gráfica es de gran ayuda para que cualquier persona observe los cambios que se producen en el brazo robótico. La interfaz realizada posee una gráfica donde se puede visualizar los cambios que ocurre en el brazo robótico, también existen botones y cuadros de texto donde el usuario puede modificar algunos parámetros o activar y desactivar

opciones según lo requiera. Al realizar una trayectoria la simulación del brazo robótico representa de manera gráfica dibujando en el espacio tridimensional trayectoria; la gráfica del brazo robótico presentada en la interfaz se puede visualizar en la Figura 57.

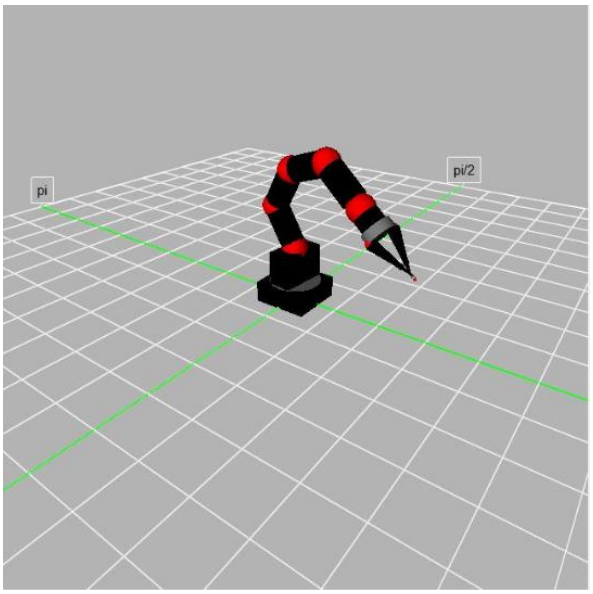


Figura 57: Simulación del brazo robótico

En la Figura 58 se muestran las opciones que posee la animación del brazo robótico para habilitar o deshabilitar las coordenadas de los ejes X, Y, Z del extremo operativo del brazo, dividir al plano en radianes en intervalos de $\pi/6$ y finalmente la opción para borrar la última trayectoria realizada por el brazo.

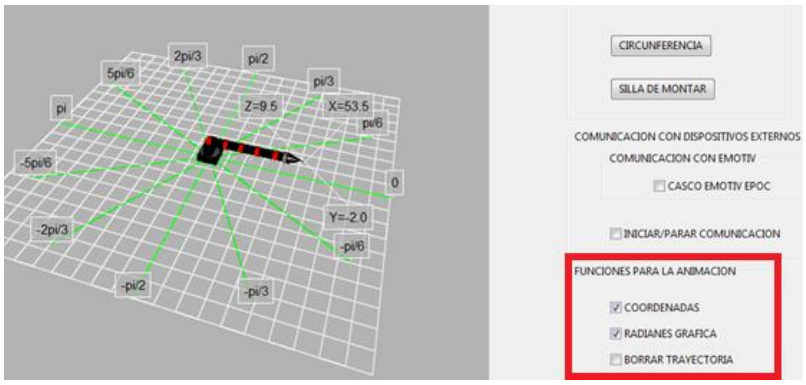


Figura 58: Checkbox de coordenadas, checkbox de radianes gráfica y checkbox para borrar la trayectoria

Como indica la Figura 59 la interfaz gráfica posee tres botones y un chekbox con funciones que se mencionan a continuación:

- **Comunicar.-** Al presionar este botón se inicia la comunicación entre el brazo robótico y Python.

- **Circunferencia.-** El brazo robótico traza una circunferencia dependiendo de la altura y radio que el usuario desee.

- **Silla de montar.-** El brazo robótico traza una silla de montar dependiendo de la altura y radio que el usuario requiera.

- **Checkbox de torque.-** Permite activar o desactivar el torque de todos los servomotores que conforman el brazo robótico.

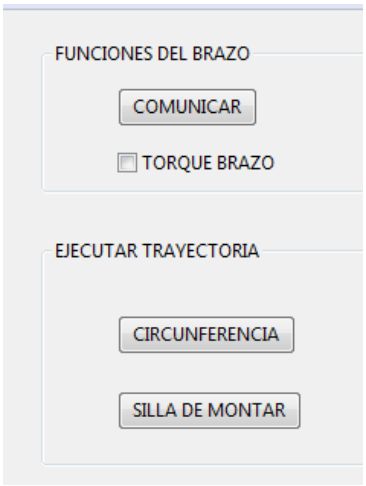


Figura 59: Botones y checkbox de torque de la Interfaz Gráfica

Finalmente la interfaz gráfica desarrollada presenta la opción para establecer la comunicación entre Python y un dispositivo externo que puede ser un joystick o el casco Emotiv EPOC (ver Figura 60); al iniciar la comunicación mediante el checkbox denominado “iniciar/parar comunicación” el joystick o el casco cambian los valores de las coordenadas en los ejes X, Y, Z para que posteriormente puedan ser visualizados.

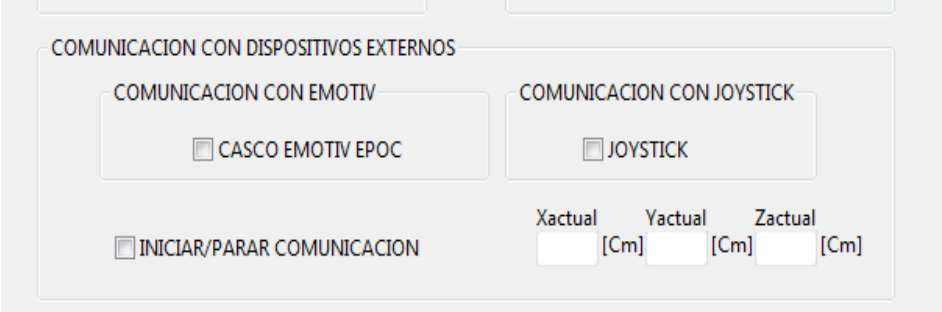


Figura 60: Elementos creados en la interfaz gráfica para la comunicación con un dispositivo externo

3.5.1 Librerías para realizar la Interfaz Gráfica

Para realizar la interfaz gráfica es necesario del uso de varias librerías adicionales como son: **visual**, **easygui**, **wx**.

- **Visual.-** Es una librería que facilita la creación y desarrollo de objetos en 3D, también se pueden crear pantallas y animaciones 3D; las mismas que se crean en la biblioteca Vpython.

- **Easygui.-** EasyGUI es una librería para la programación interfaz gráfica del usuario (GUI).

- **Wx.-** Esta librería permite el desarrollo rápido de aplicaciones gráficas multiplataforma, es compatible con Windows y Python; y es la responsable de la creación de botones, sliders, checkbox, etc.

3.5.2 Declaración de constantes para crear la pantalla de trabajo

Las constantes que se indican en la Figura 61 sirven para crear la pantalla de trabajo y los objetos tridimensionales que simulan al brazo robótico. (Anexo B, Interfaz Grafica.py, línea 112 - línea 120)

```
#-------------------------------------------------------------------------------------------------------------------
#DECLARACIÓN DE CONSTANTES PARA LA INTERFAZ
display_dimensiones=wx.GetDisplaySize()
pasos=[]
esfera=[(0,0),(0,0),(0,0),(0,0),(0,0),(0,0),(0,0)]

const_screen=display_dimensiones[0]/display_dimensiones[1]
d = 360*const_screen
d2=d/2
borde_x=12*const_screen
w = window(x=(display_dimensiones[0]-1400)/2,y=(display_dimensiones[1]-800)/2,width=1400, height=750, menus=True,
              title='SISTEMA DE CONTROL PARA BRAZO ROBOTICO DE 6 GRADOS DE LIBERTAD')

pantalla=display(window=w, x=borde_x, y=borde_x, width=d, height=d,background=(0.7,0.7,0.7))
#-------------------------------------------------------------------------------------------------------------------
```

Figura 61: Constantes declaradas para crear la pantalla de trabajo y los objetos tridimensionales

El comando **window** permite crear la ventana en la cual se va a desarrollar la interfaz gráfica, es necesario los parámetros de **X** y **Y** los cuales son las coordenadas en donde se va a ubicar la pantalla, además de **width**, **height** para determinar el alto y ancho de la pantalla, **tittle** para nombrar a la ventana (Velasco Sánchez & Mamarandi Quilo, 2015).

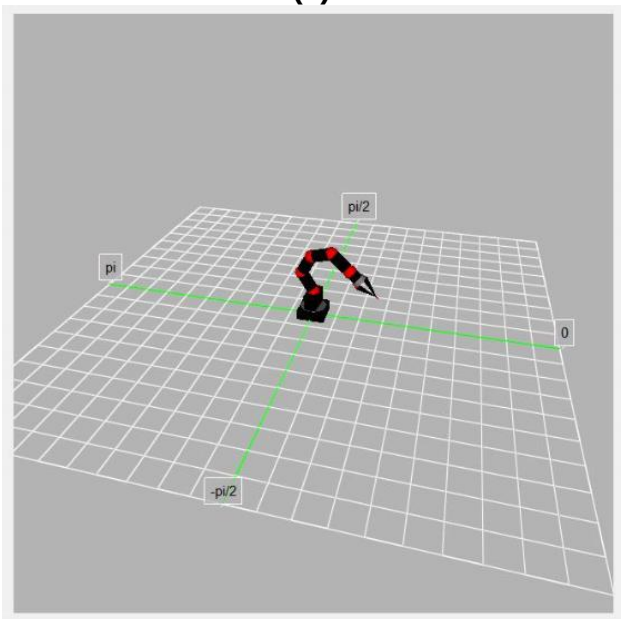
La palabra **display**, es la que crea la ventana en la cual se van a observar todos los objetos sólidos que representan el brazo robótico, los parámetros que necesitan son: **width**, **height** para determinar el alto y ancho, **X** y **Y** para ubicar a la pantalla en la ventana, **window** para saber en cual ventana se colocará y **background** que determina el color de fondo de la pantalla.

3.5.3 Creación del plano de la simulación

La matriz de coordenadas que grafica el plano en donde se encuentra la simulación del brazo robótico, crea una cuadrícula como la que se muestra en la Figura 62, esto se logra con el comando **curve**, que genera una recta de 2D uniendo los puntos en el espacio (X, Y, Z) (Velasco Sánchez & Mamarandi Quilo, 2015). (Anexo B, Interfaz Final, línea 122- línea 136)

```
plane = curve(pos=[(-100,0,100), (100,0,100), (100, 0,-100), (-100, 0, -100),
                   (-100,0,100),(-90,0,100),(-90,0,-100),(-80,0,-100),(-80,0,100),
                   (-70,0,100),(-70,0,-100),(-60,0,-100),(-60,0,100),(-50,0,100),(-50,0,-100),
                   (-40,0,-100),(-40,0,100),(-30,0,100),(-30,0,-100),(-20,0,-100),(-20,0,100),
                   (-10,0,100),(-10,0,-100),(0,0,-100),(0,0,100),(10,0,100),(10,0,-100),
                   (20,0,-100),(20,0,100),(30,0,100),(30,0,-100),(40,0,-100),(40,0,100),
                   (50,0,100),(50,0,-100),(60,0,-100),(60,0,100),(70,0,100),(70,0,-100),
                   (80,0,-100),(80,0,100),(90,0,100),(90,0,-100),(100,0,-100),(100,0,-90),(-100,0,-90),
                   (-100,0,-80),(100,0,-80),(100,0,-70),(-100,0,-70),(-100,0,-60),(100,0,-60),
                   (100,0,-50),(-100,0,-50),(-100,0,-40),(100,0,-40),(100,0,-30),(-100,0,-30),
                   (-100,0,-20),(100,0,-20),(100,0,-10),(-100,0,-10),(-100,0,0),(100,0,0),
                   (100,0,10),(-100,0,10),(-100,0,20),(100,0,20),(100,0,30),(-100,0,30),
                   (-100,0,40),(100,0,40),(100,0,50),(-100,0,50),(-100,0,60),(100,0,60),
                   (100,0,70),(-100,0,70),(-100,0,80),(100,0,80),(100,0,90),(-100,0,90)],display=pantalla)
```

(a)



(b)

Figura 62: a) Código en Python para definir un plano; b) Ventana creada mediante el código para la simulación

3.5.4 Creación de botones

Para crear un botón se tiene que tener la siguiente estructura:

wx.Button(w.panel,label=’ ’,pos=(),size=( ))

Se puede observar que una variable es igualada a una instrucción. Entre los apostrofes de **label** se escribe el nombre que se mostrará en el botón, entre los paréntesis de **pos** se ingresa las coordenadas de la posición en la que se requiere establecer y entre los paréntesis de **size** se da el tamaño que tendrá el dicho elemento (Velasco Sánchez & Mamarandi Quilo, 2015).

La Figura 63 muestra el código utilizado para la creación de los botones de circunferencia, silla de montar y comunicación que posee la interfaz gráfica,

los cuales al presionarlos permiten al brazo robótico realizar una circunferencia, una silla de montar y establecer la comunicación entre Python y la interfaz USB2Dynamixel respectivamente (Anexo B, Interfaz Final, línea 166 – línea 168).

```
#-------------------------------------------------------------------------
#DECLARACIÓN DE BOTONES
b_Comunicacion = wx.Button(w.panel,label='COMUNICAR',pos=(800,50))
b_Circunferencia = wx.Button(w.panel,label='CIRCUNFERENCIA',pos=(800,200))
b_Silla = wx.Button(w.panel,label='SILLA DE MONTAR',pos=(800,250))
#-------------------------------------------------------------------------
```

Figura 63: Codificación para crear los botones en la interfaz gráfica

3.5.5 Spinners

La declaración de un spinner es la siguiente:

wx.SpinCtrlDouble(w.panel,label=' ',pos=(),size=( ),initial,min,max)

Los spinners se utilizan para aumentar o disminuir un valor numérico, la estructura del mismo está compuesta por posición, tamaño y valor inicial, el valor mínimo se lo puede omitir ya que por defecto es 0, pero si se desea cambiar se utiliza el parámetro **min** y se lo iguala al valor que se requiera como mínimo (Velasco Sánchez & Mamarandi Quilo, 2015). (Anexo B, Interfaz Final, línea 189). En la Figura 64 se muestra el código implementado para la creación de spinners en la interfaz gráfica.

```
#-------------------------------------------------------------------------------------------
#DECLARACIÓN DE SPIN
radio= wx.SpinCtrlDouble(w.panel, pos=(1160,200), size=(50,20),initial=20,min=20,max=50,inc = 1)
altura= wx.SpinCtrlDouble(w.panel, pos=(1160,250), size=(50,20),initial=10,min=10,max=30,inc = 1)
#-------------------------------------------------------------------------------------------
```

Figura 64: Codificación para crear los Spinners en la interfaz gráfica

3.5.6 StaticBox

Los staticbox son cajas contenedores que permiten escribir títulos o referencias dentro de la interfaz gráfica. Su estructura está compuesta por posición, tamaño y etiqueta (Velasco Sánchez & Mamarandi Quilo, 2015). (Anexo B, Interfaz Final, línea 180 – línea 186)

wx.StaticBox(w.panel,label=' ',pos=(),size=())

En la Figura 65 se observa el código implementado para crear los staticbox en la interfaz gráfica.

```
#-------------------------------------------------------------------------------------------
#DECLARACIÓN DE STATICBOX
wx.StaticBox(w.panel,pos=(750,20), size=(250,100),label='FUNCIONES DEL BRAZO')
wx.StaticBox(w.panel,pos=(750,150), size=(250,150),label='EJECUTAR TRAYECTORIA')
wx.StaticBox(w.panel,pos=(1050,150), size=(250,150),label='PARAMETROS DE TRAYECTORIAS')
wx.StaticBox(w.panel,pos=(750,310), size=(550,150),label='COMUNICACION CON DISPOSITIVOS EXTERNOS')
wx.StaticBox(w.panel,pos=(750,465), size=(250,150),label='FUNCIONES PARA LA ANIMACION')
wx.StaticBox(w.panel,pos=(790,335), size=(250,58),label='COMUNICACION CON EMOTIV')
wx.StaticBox(w.panel,pos=(1050,335), size=(200,58),label='COMUNICACION CON JOYSTICK')
#-------------------------------------------------------------------------------------------
```

Figura 65: Codificación para crear los StaticBox en la interfaz gráfica

3.5.7 CheckBox

Los checkbox poseen dos estados, el de activado cuando se subraya dentro del mismo un visto y el desactivado cuando no se encuentra nada dentro del checkbox. Su estructura se compone de la posición, tamaño, etiqueta y el valor establecido siendo 1 para activado y 0 para desactivado (Velasco Sánchez & Mamarandi Quilo, 2015) (Anexo B, Interfaz Final, línea 181 – línea 187).

wx.CheckBox(w.panel,label=' ',pos=(),size=( ))

En la Figura 66 se indica la codificación para crear los checkbox en la interfaz gráfica.

```
#-------------------------------------------------------------------------------------------
#DECLARACIÓN DE CHECKBOX
cb_coordenadas = wx.CheckBox(w.panel,pos=(800,500), size=(300,30),label='COORDENADAS')
intervalo_radian = wx.CheckBox(w.panel,pos=(800,530), size=(300,30),label='RADIANES GRAFICA')
borrar_grafica = wx.CheckBox(w.panel,pos=(800,560), size=(300,30),label='BORRAR TRAYECTORIA')
comunicar = wx.CheckBox(w.panel,pos=(800,410), size=(200,40),label='INICIAR/PARAR COMUNICACION')
torque = wx.CheckBox(w.panel,pos=(800,75), size=(105,50),label='TORQUE BRAZO')
casco = wx.CheckBox(w.panel,pos=(850,350), size=(200,50),label='CASCO EMOTIV EPOC')
joystick = wx.CheckBox(w.panel,pos=(1100,350), size=(115,50),label='JOYSTICK')
#-------------------------------------------------------------------------------------------
```

Figura 66: Codificación para crear los CheckBox en la interfaz gráfica

3.5.8 StaticText

Estas son solo etiquetas que van ubicadas en una posición definida y sirven para colocar títulos, nombres, referencias. Su estructura está compuesta de posición y etiqueta. (Anexo B, Interfaz Final, línea 193 – línea 202)

wx.StaticText(w.panel,label=' ',pos=())

En la Figura 67 se puede visualizar el código realizado para crear statictext dentro de la interfaz gráfica.

```
#-------------------------------------------------------
#DECLARACIÓN DE TEXTO ESTATICO
wx.StaticText(w.panel, pos=(1100,200), label='RADIO = ')
wx.StaticText(w.panel, pos=(1090,250), label='ALTURA = ')
wx.StaticText(w.panel, pos=(1225,200), label='[Cm]')
wx.StaticText(w.panel, pos=(1225,250), label='[Cm]')
wx.StaticText(w.panel, pos=(1112,420), label='[Cm]')
wx.StaticText(w.panel, pos=(1182,420), label='[Cm]')
wx.StaticText(w.panel, pos=(1252,420), label='[Cm]')
wx.StaticText(w.panel, pos=(1070,405), label='Xactual')
wx.StaticText(w.panel, pos=(1140,405), label='Yactual')
wx.StaticText(w.panel, pos=(1210,405), label='Zactual')
#-------------------------------------------------------
```

Figura 67: Codificación para crear los StaticText en la interfaz gráfica

3.5.9 TextCtrl

Los controles de texto permiten la lectura y el ingreso de datos por teclado. Su estructura utiliza posición y tamaño. (Anexo B, Interfaz Final, línea 205- línea 207)

wx.TextCtrl(w.panel,label=' ',pos=(),size=( ),value=' ')

En la Figura 68 se muestra el código para la creación de los controles de texto que se ocupa en la interfaz gráfica.

```
#-------------------------------------------------------------
#DECLARACIÓN DE TEXTCTRL
x_actual = wx.TextCtrl(w.panel, pos=(1070,420),size=(40,20))
y_actual = wx.TextCtrl(w.panel, pos=(1140,420),size=(40,20))
z_actual = wx.TextCtrl(w.panel, pos=(1210,420),size=(40,20))
#-------------------------------------------------------------
```

Figura 68: Codificación para crear los TextCtrl en la interfaz gráfica

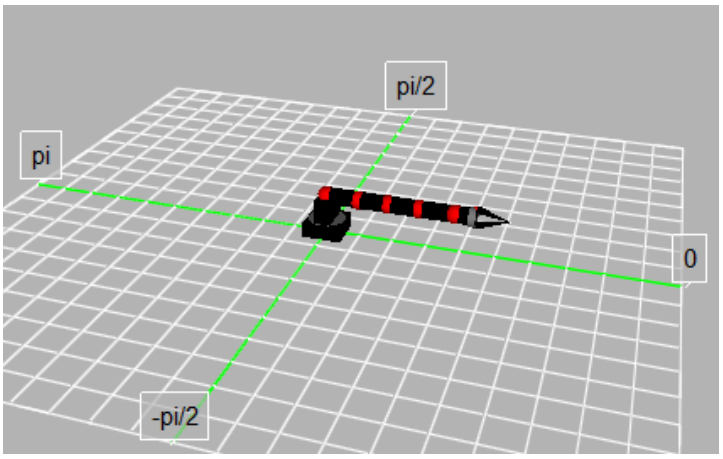
3.5.10 Creación de sólidos

Los sólidos creados para la interfaz gráfica poseen sus propios parámetros dependiendo del tipo de cuerpo geométrico. Si por ejemplo se crea un cono los parámetros a definir serían la altura y el radio de la base, si es una esfera definir su radio, si es un paralelepípedo o un cubo hay que definir su alto, ancho y espesor (Velasco Sánchez & Mamarandi Quilo, 2015). (Anexo B, Interfaz Final, línea 210 – línea 236)

En la Figura 69 se muestra el código para la creación de sólidos y la ubicación de los mismos en la interfaz gráfica.

```
#-----------------------------------------------------------------------------------------------
#DECLARACIÓN DE SOLIDOS
base = box(pos=(0,2,0),length=11.5, height=4.5, width=11.5, display=pantalla,color=color.black)
base_2=cylinder(pos=(0,4.25,0),axis=(0,1,0), radius=5.5,length=0.1,display=pantalla,color=(0.5,0.5,0.5))
pieza_1=box(pos=(0,7.85,0),length=10, height=7,  width=5 ,display=pantalla,color=color.black)
pieza_2=cylinder(length=22.5,  radius=2.5 ,color=color.black,display=pantalla)
pieza_3=cylinder(length=15.5,  radius=2.5 ,color=color.black,display=pantalla)
pieza_4=cylinder(length=9,  radius=2.5 ,color=color.black,display=pantalla)
pieza_5=cylinder(length=8.5,  radius=2.5 ,color=color.black,display=pantalla)
pieza_6=cylinder(length=6,  radius=2.5 ,color=color.black,display=pantalla)
union_1=sphere(pos=(1,11.35,0), radius=2.8, display=pantalla,color=color.red)
union_2=sphere(pos=(0,22.35,0), radius=2.8,display=pantalla,color=color.red)
union_3=sphere(pos=(0,44.85,0), radius=2.8,display=pantalla,color=color.red)
union_4=sphere(pos=(0,60.35,0), radius=2.8,display=pantalla,color=color.red)
union_5=sphere(pos=(0,67.35,0), radius=2.8,display=pantalla,color=color.red)
union_6=sphere(pos=(0,70.35,0), radius=0.3,display=pantalla,color=color.red)
union_7=sphere(pos=(0,70.35,0), radius=0.1,display=pantalla)
union_8=sphere(pos=(0,70.35,0), radius=0.1,display=pantalla)
union_9=sphere(pos=(0,70.35,0), radius=0.1,display=pantalla)
union_10=sphere(pos=(0,70.35,0), radius=0.1,display=pantalla)
union_pinza_1=sphere(pos=(-2,-4,0), radius=1,color=color.red,display=pantalla)
union_pinza_2=sphere(pos=(2,-4,0), radius=1,color=color.red,display=pantalla)
union_pinza=sphere(pos=(-2,-4,0), radius=0.2,color=color.black,display=pantalla)
pieza_pinza_1=cone(pos=(-2,-4,0),radius=1.2, color=color.black,display=pantalla)
pieza_pinza_2=cone(pos=(2,-4,0),radius=1.2, color=color.black,display=pantalla)
pieza_pinza_11=cylinder(pos=(-2,-4,0),radius=0.5,color=color.black,display=pantalla)
pieza_pinza_22=cylinder(pos=(2,-4,0),radius=0.5, color=color.black,display=pantalla)
base_pinza = cylinder( radius=3, color=(0.5,0.5,0.5),display=pantalla)
coordenadas=cylinder(pos=(0,0,0),radius=0.1,display=pantalla)
#-----------------------------------------------------------------------------------------------
```

(a)



(b)

Figura 69: a) Código para la creación de sólidos en Python; b) Sólidos creados para la interfaz gráfica

Una vez declarado y posicionado a cada uno de los elementos necesarios para el desarrollo del presente proyecto la interfaz gráfica terminada se muestra en la Figura 70.

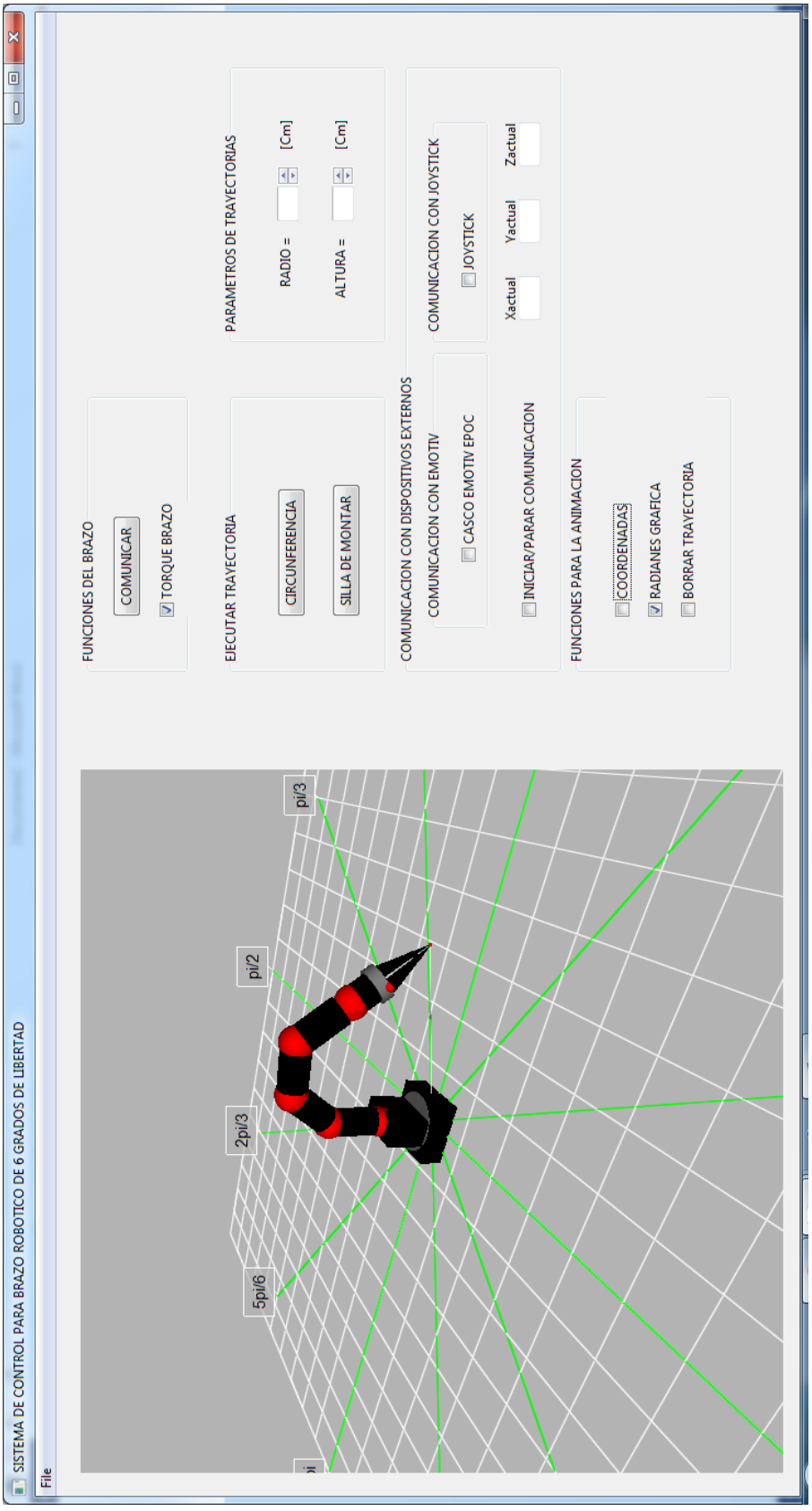


Figura 70: Interfaz Gráfica final

CAPÍTULO IV

4. PRUEBAS Y ANÁLISIS DE RESULTADOS

En los siguientes apartados se procede a realizar las pruebas de tareas específicas que se encuentran presentes en la interfaz gráfica realizada para el presente proyecto, con lo cual se logra el control del brazo robótico obteniendo los siguientes resultados mostrados a continuación.

4.1 Pruebas de funcionamiento para una trayectoria establecida

Se da la opción al usuario de poder escoger el tipo de trayectoria a realizar, ya sea esta la de una circunferencia o una silla de montar. El usuario tiene que ingresar el valor de radio y altura, estos valores sirven de referencia para que el brazo siga una trayectoria previamente establecida; por ejemplo en la Figura 71 se muestra la selección de una circunferencia la cual tiene un radio de 30 cm y altura 10 cm.

Figura 71: Selección del tipo de trayectoria e ingreso de valores en radio y altura de la interfaz gráfica

4.1.1 Trayectoria de una circunferencia

Para las pruebas con la trayectoria de una circunferencia que realizará el brazo robótico se puede escoger diferentes valores para el radio y la altura previamente limitados para evitar esfuerzos en la estructura mecánica. La Figura 72 representa la trayectoria de una circunferencia con radio de 30 cm y una altura igual a 10 cm que ejecutará el robot.

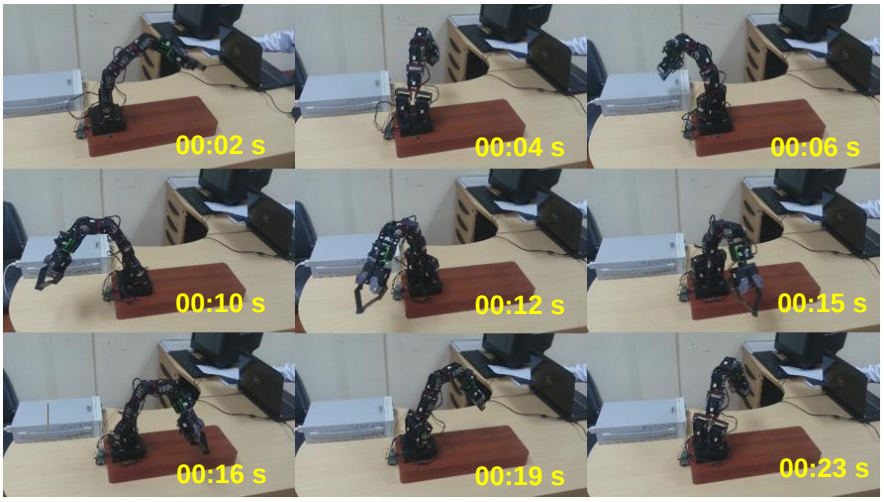


Figura 72: Trayectoria para una circunferencia de radio 30 cm y altura 10 cm

En la Figura 73 y Figura 74 se puede observar mediante la interfaz gráfica el error obtenido entre la circunferencia realizada por el brazo (color rojo) y la trayectoria ideal (color azul).

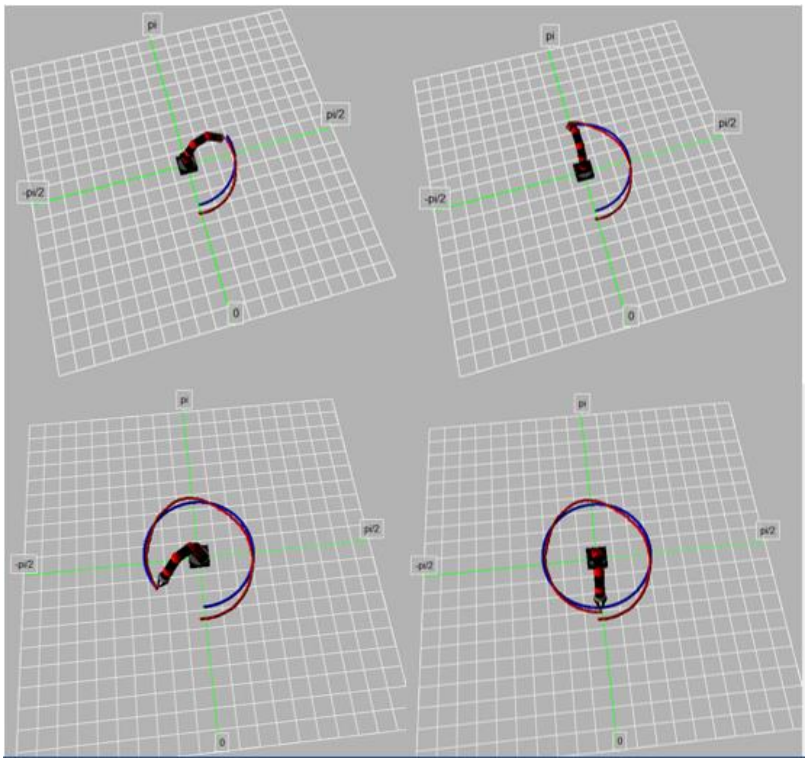


Figura 73: Circunferencia de radio 30 cm y altura 10 cm en la interfaz gráfica de Python

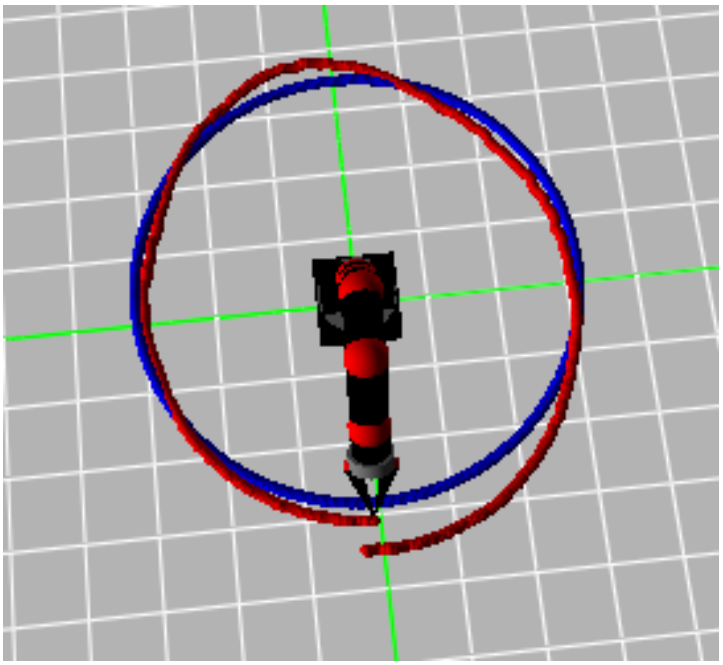


Figura 74: Resultado del error al realizar una circunferencia de radio 30 cm y altura 10 cm en la interfaz gráfica de Python

La Figura 75 representa la trayectoria de una circunferencia con radio igual a 40 cm y una altura igual a 20 cm.

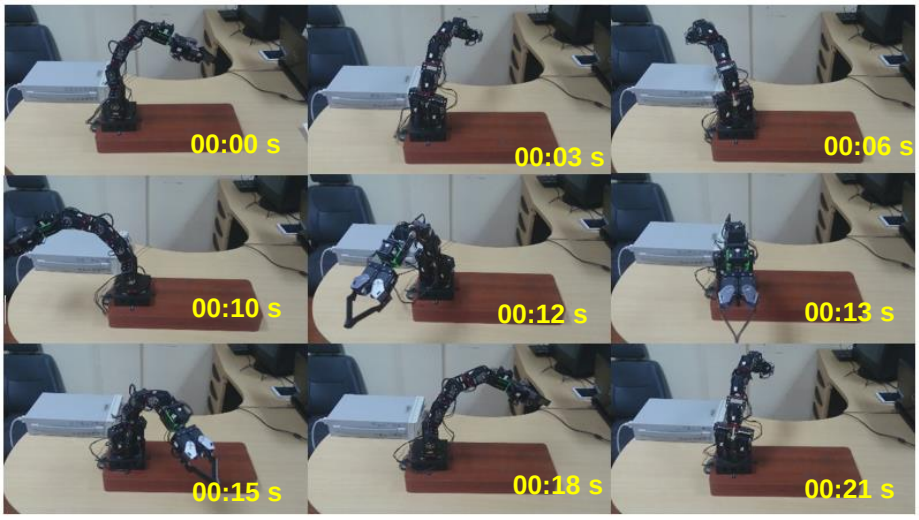


Figura 75: Trayectoria para una circunferencia de radio 40 cm y altura 20 cm

En la Figura 76 y Figura 77 se puede observar mediante la interfaz gráfica el error obtenido entre la circunferencia realizada por el brazo (color rojo) y la trayectoria ideal (color azul).

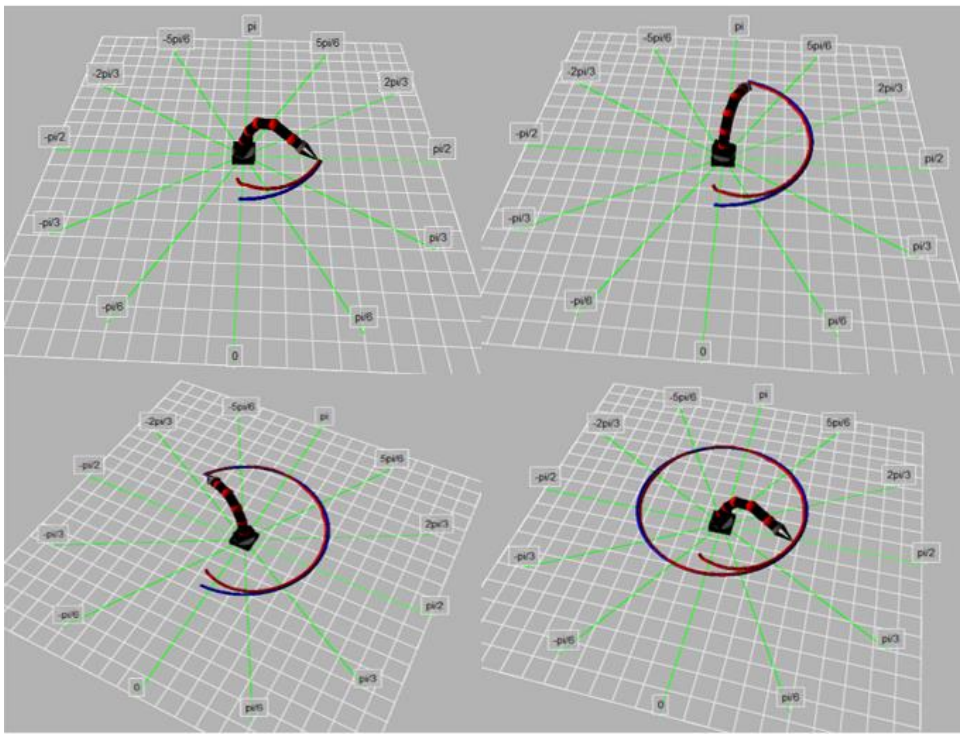


Figura 76: Circunferencia de radio 40 cm y altura 20 cm en la interfaz gráfica de Python

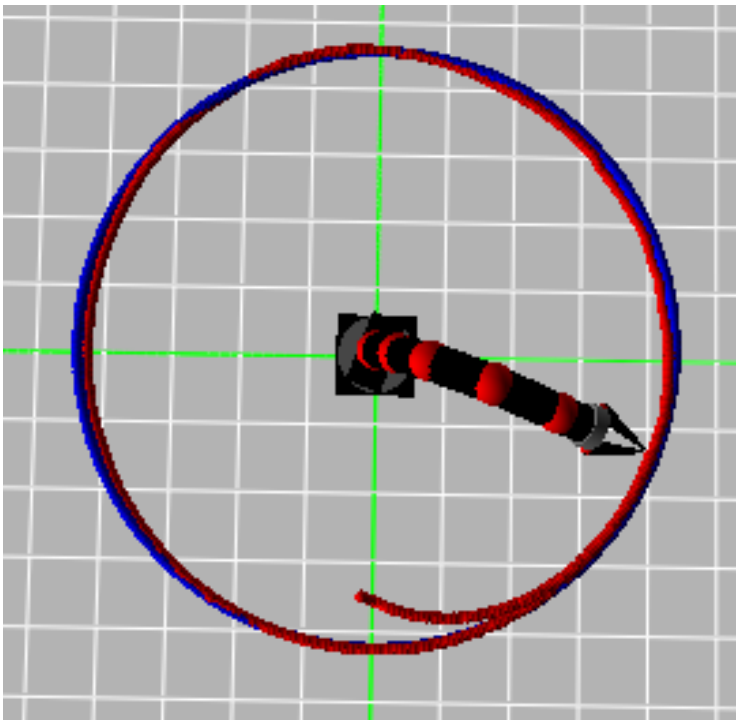


Figura 77: Resultado del error al realizar una circunferencia de radio 40 cm y altura 20 cm en la interfaz gráfica de Python

La Figura 78 representa la trayectoria de una circunferencia con radio igual a 50 cm y una altura igual a 30 cm.

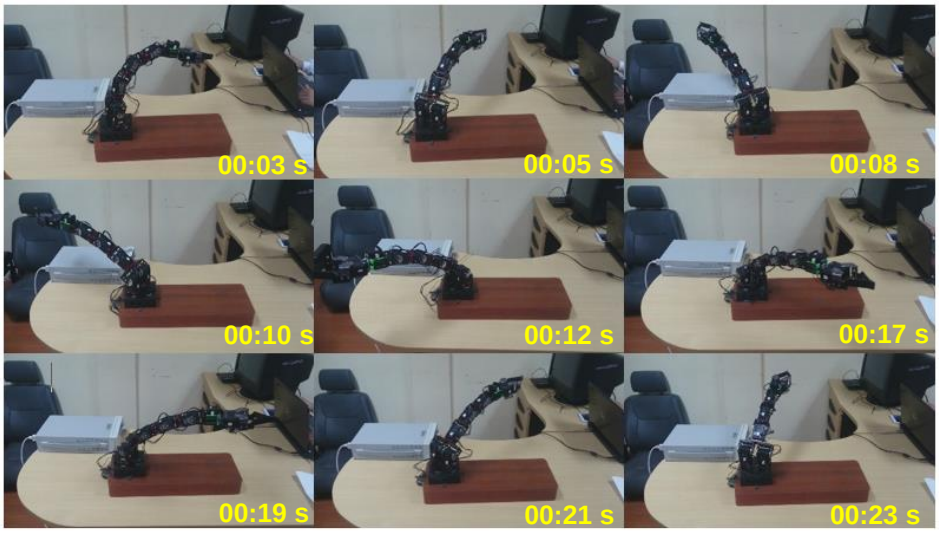


Figura 78: Trayectoria para una circunferencia de radio 50 cm y altura 30 cm

En la Figura 79 y Figura 80 se puede observar mediante la interfaz gráfica el error obtenido entre la circunferencia realizada por el brazo (color rojo) y la trayectoria ideal (color azul).

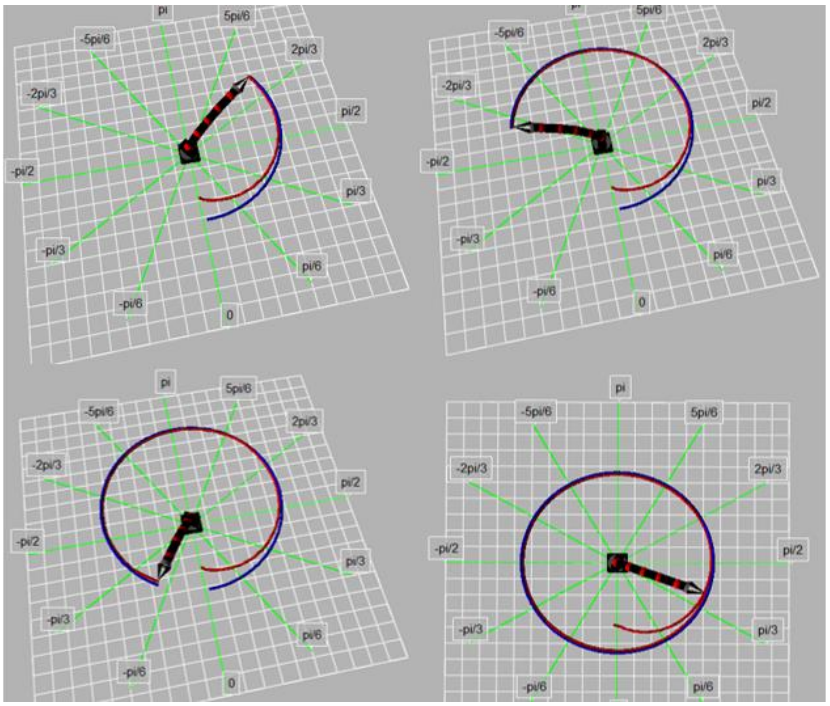


Figura 79: Circunferencia de radio 50 cm y altura 30 cm en la interfaz gráfica de Python

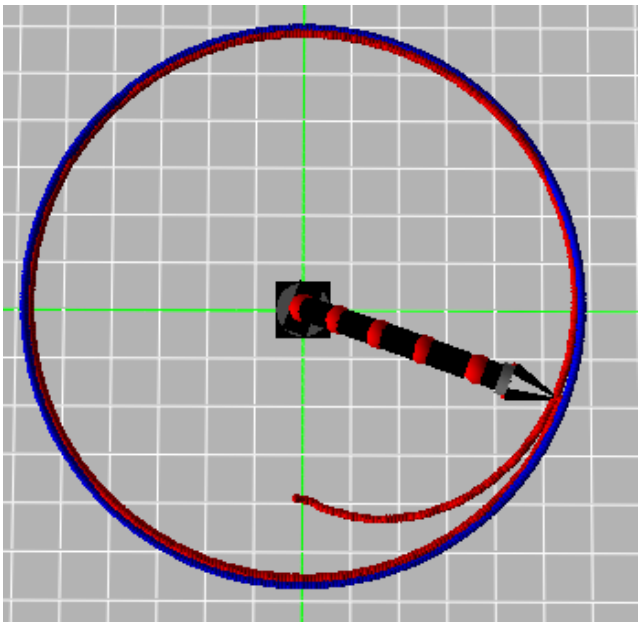


Figura 80: Resultado del error al realizar una circunferencia de radio 50 cm y altura 30 cm en la interfaz gráfica de Python

4.1.2 Trayectoria de una silla de montar

Para realizar la prueba con la trayectoria de silla de montar que cumple el brazo robótico al igual forma que al realizar una circunferencia se puede variar los valores de altura y radio. La silla de montar de radio 30 cm y de altura 10 cm, está representada en la Figura 81.

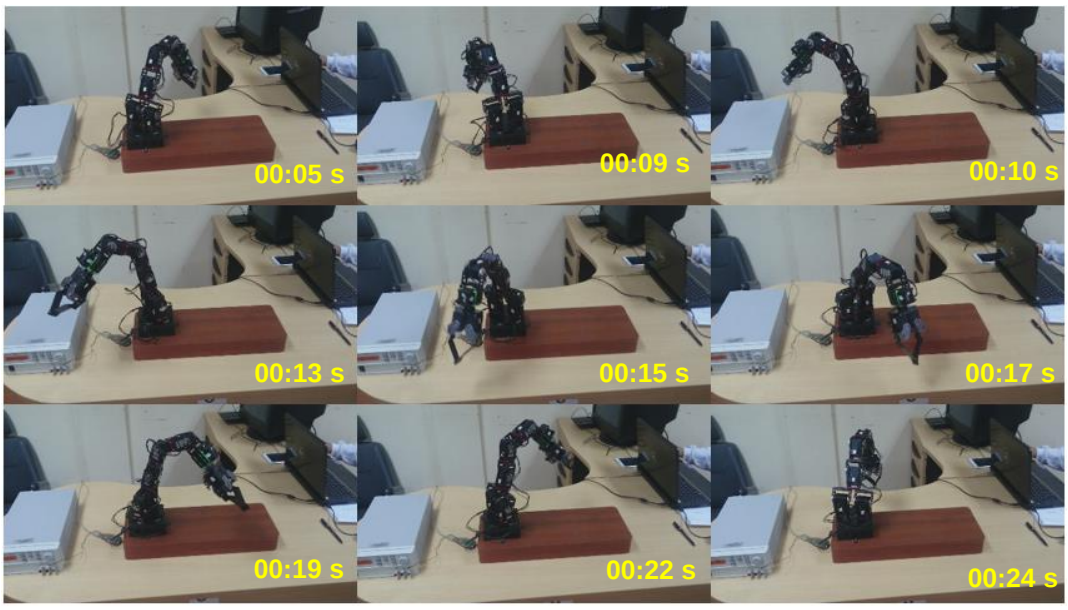


Figura 81: Trayectoria para una silla de montar con radio 30 cm y altura 10 cm

En la interfaz gráfica se puede visualizar el error que tiene la trayectoria de silla de montar en la parte real (color rojo) comparada con la trayectoria ideal (color azul), como se indica en la Figura 82 y Figura 83.

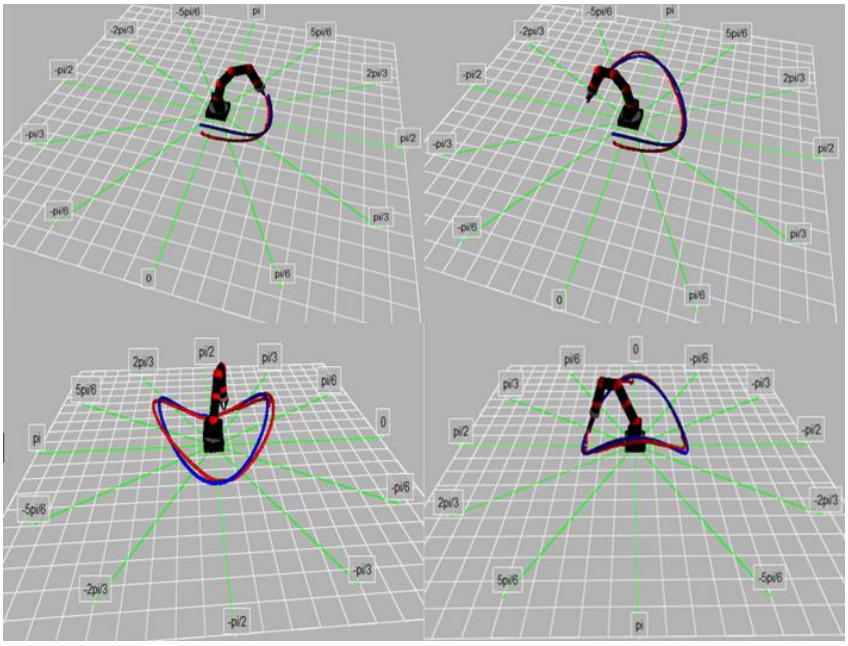


Figura 82: Silla de montar de radio 30 cm y altura 10 cm en la interfaz gráfica de Python

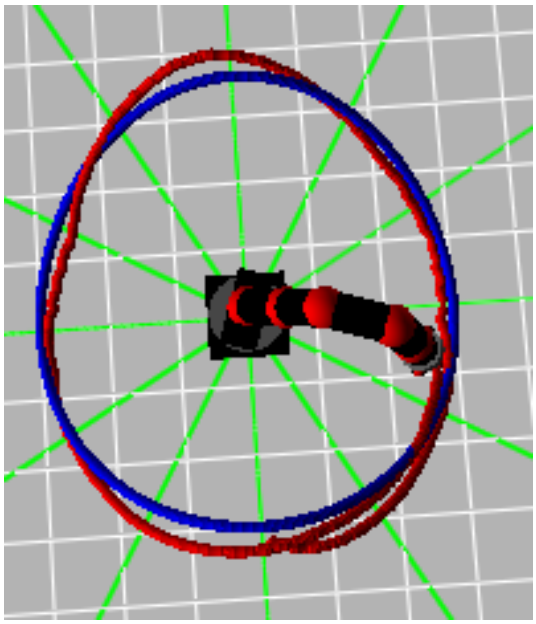


Figura 83: Resultado del error al realizar una silla de montar de radio 30 cm y altura 10 cm en la interfaz gráfica de Python

La siguiente prueba es con la trayectoria de una silla de montar con valores de radio igual a 40 cm y con una altura igual a 20 cm, esto se muestra en la Figura 84.

Figura 84: Trayectoria para una silla de montar con radio 40 cm y altura 20 cm

En la interfaz gráfica se puede visualizar el error que tiene la trayectoria de silla de montar en la parte real (color rojo) comparada con la trayectoria ideal (color azul) como se indica en la Figura 85 y Figura 86.

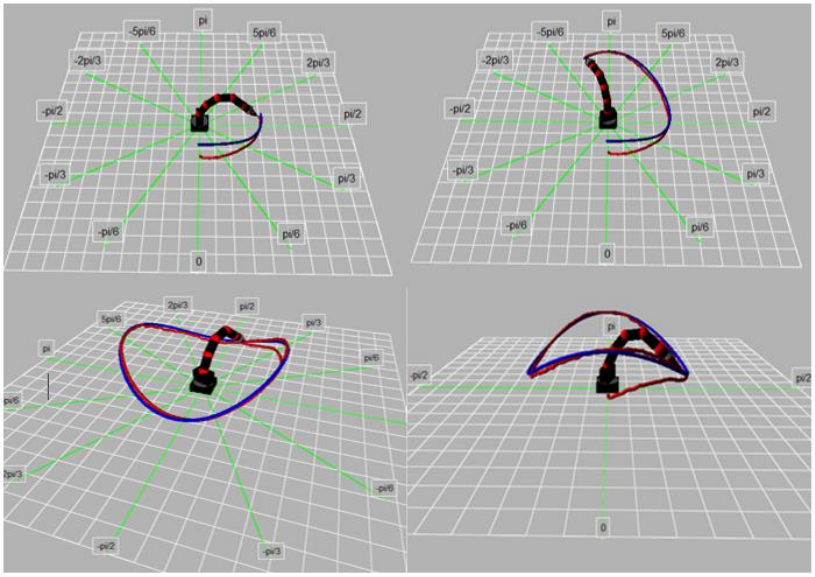


Figura 85: Silla de montar de radio 40 cm y altura 20 cm en la interfaz gráfica de Python

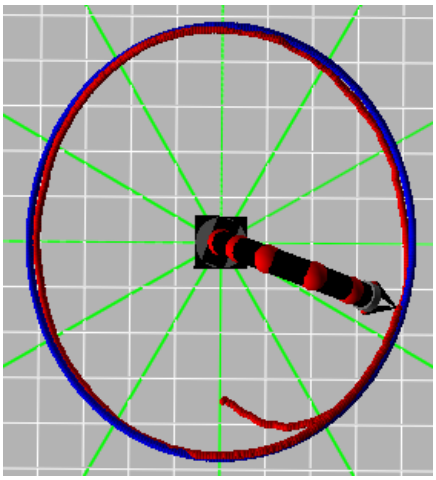


Figura 86: Resultado del error al realizar una silla de montar de radio 40 cm y altura 20 cm en la interfaz gráfica de Python

4.2 GRÁFICA DE ERRORES

Para una trayectoria de una circunferencia de radio 40 cm y altura 20 cm como se indica en la Figura 87, se tienen los siguientes errores en los ejes X, Y, Z presentes en la trayectoria como se muestra en la Figura 88. Como se puede apreciar el error existente en cada uno de los ejes es tolerable y cercano a cero, al realizar la trayectoria real con el brazo y realizar la trayectoria ideal con el software.

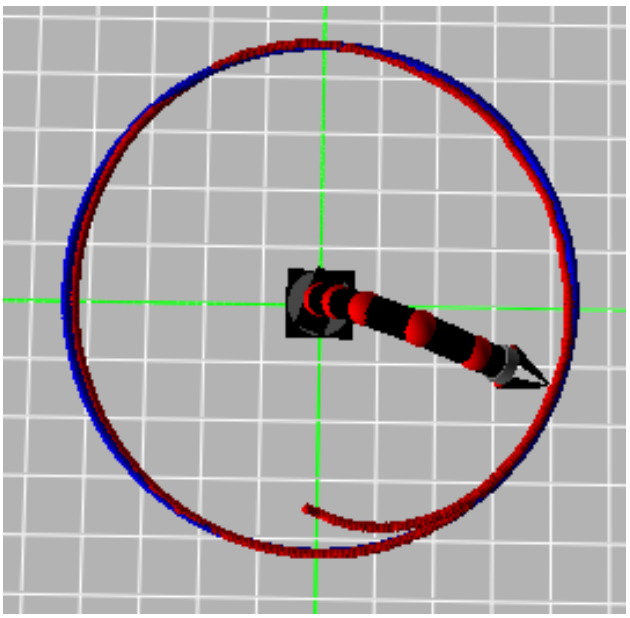
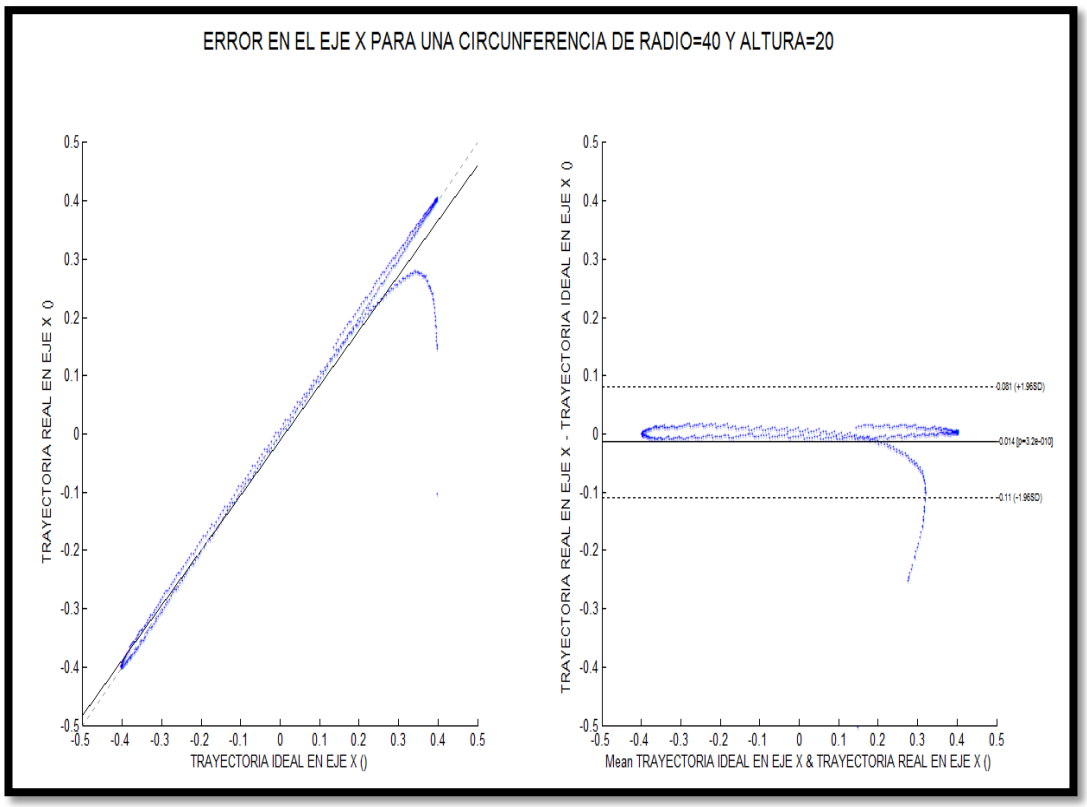
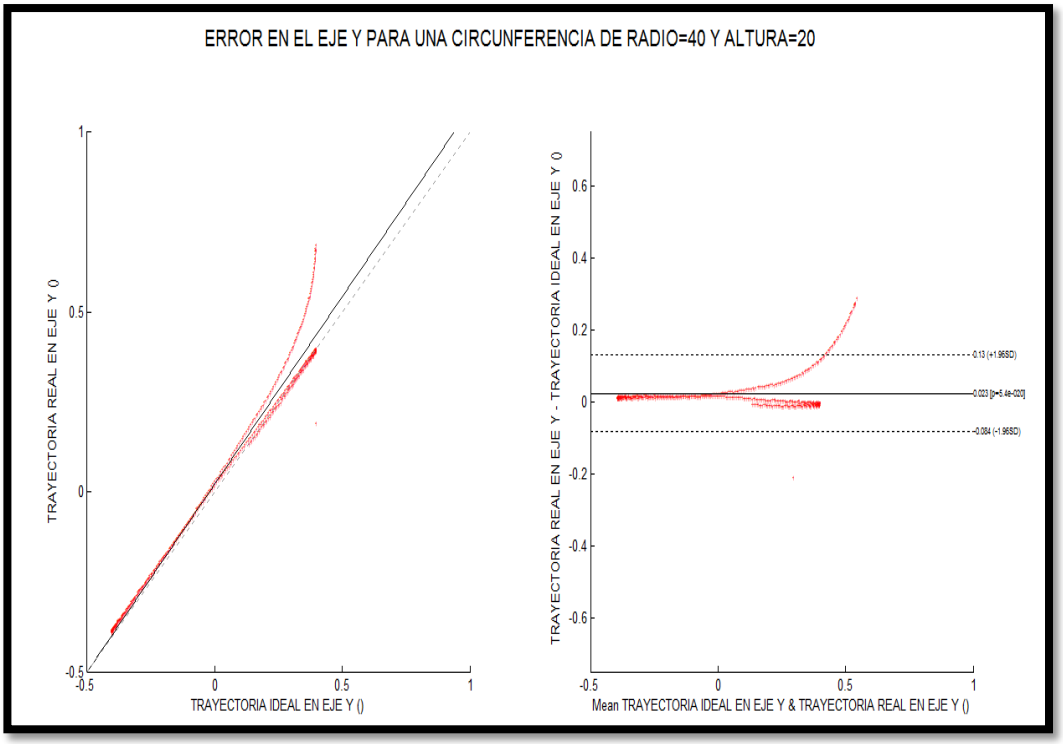


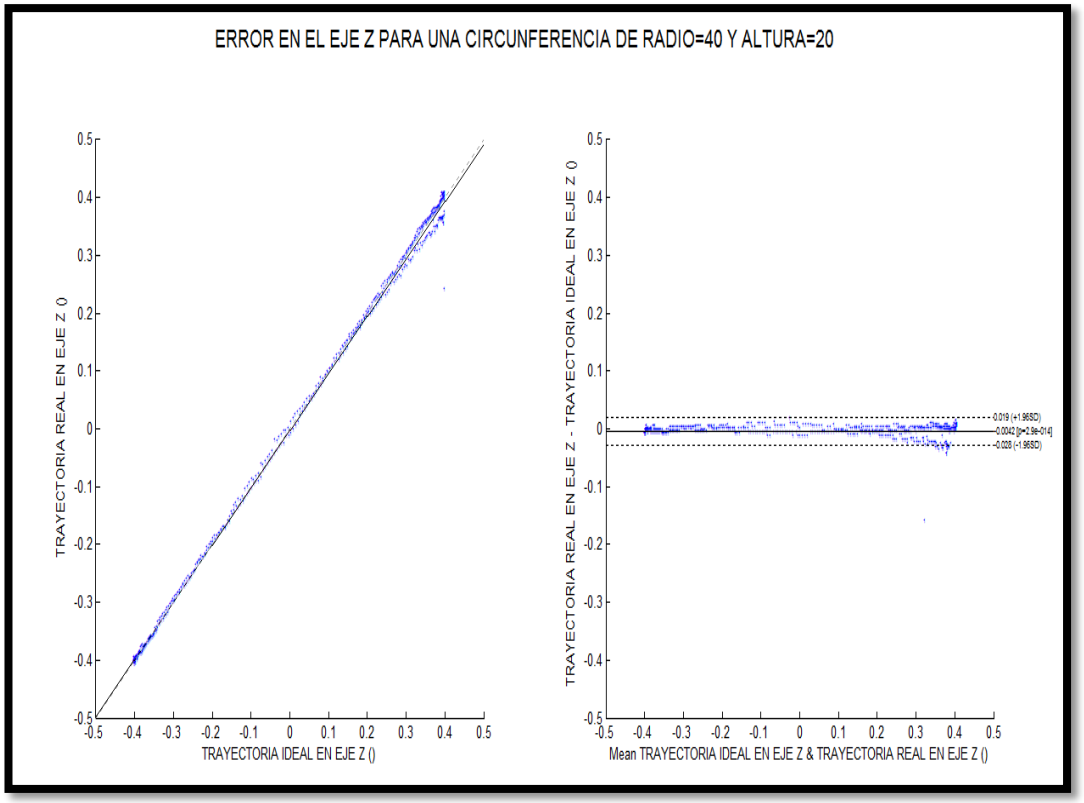
Figura 87: Realización de trayectoria de una circunferencia de radio 40 cm y altura 20 cm


ERROR EN EL EJE X PARA UNA CIRCUNFERENCIA DE RADIO=40 Y ALTURA=20
TRAYECTORIA REAL EN EJE X ()
TRAYECTORIA IDEAL EN EJE X ()
TRAYECTORIA REAL EN EJE X - TRAYECTORIA IDEAL EN EJE X ()
0.081 (+1.96SD)
0.014 [p=3.2e-010]
-0.11 (-1.96SD)
Mean TRAYECTORIA IDEAL EN EJE X & TRAYECTORIA REAL EN EJE X ()

(a)


ERROR EN EL EJE Y PARA UNA CIRCUNFERENCIA DE RADIO=40 Y ALTURA=20
TRAYECTORIA REAL EN EJE Y ()
TRAYECTORIA IDEAL EN EJE Y ()
TRAYECTORIA REAL EN EJE Y - TRAYECTORIA IDEAL EN EJE Y ()
0.13 (+1.96SD)
-0.023 [p=5.4e-020]
-0.084 (-1.96SD)
Mean TRAYECTORIA IDEAL EN EJE Y & TRAYECTORIA REAL EN EJE Y ()

(b)



(c)

Figura 88: (a) Error en el eje X; (b) Error en el eje Y; (c) Error en el eje Z, para un circunferencia de radio 40 cm y altura 20 cm

Al romperce el momento de inercia al realizar la trayectoria de una circunferencia de radio 40 cm y altura 20 cm se puede observar en cada uno de los ejes que la trayectoria enviada se ajusta a la trayectoria realizada con un error mínimo.

4.3 CONTROL DE POSICIÓN MEDIANTE JOYSTICK

La interfaz gráfica también permite seleccionar la comunicación entre Python y un joystick, de esta manera también se puede controlar al brazo robótico por medio de una palanca de juegos. El joystick está configurado convenientemente para mover al brazo en los ejes $X,,Y,Z$; y adicionalmente también puede controlar la muñeca y la pinza del robótico (ver Figura 89).

Figura 89: Configuración de botones del Joystick

El control de posición para el brazo robótico mediante el joystick permite cumplir una trayectoria definida por dicho joystick, que para este caso el brazo robótico pueda cambiar de posición a una botella, esto se muestra en la Figura 90.

Figura 90: Control del brazo robótico mediante un joystick

4.4 Control de brazo robótico utilizando el casco Emotiv EPOC

Para realizar esta prueba se utiliza acciones faciales que permiten el control del brazo robótico, las acciones obtenidas por el casco Emotiv EPOC son las siguientes: guiñar el ojo derecho/izquierdo el cual sirve para realizar el movimiento del brazo robótico en el eje X, mueca derecha/izquierda que permite desplazar al brazo robótico en el eje Y, mirar hacia la

derecha/izquierda para mover el brazo robótico en eje Z positivo. En la Figura 91 se muestra el estado de los electrodos al momento de realizar las pruebas con el casco Emotiv EPOC en el programa Emotiv Control Panel.

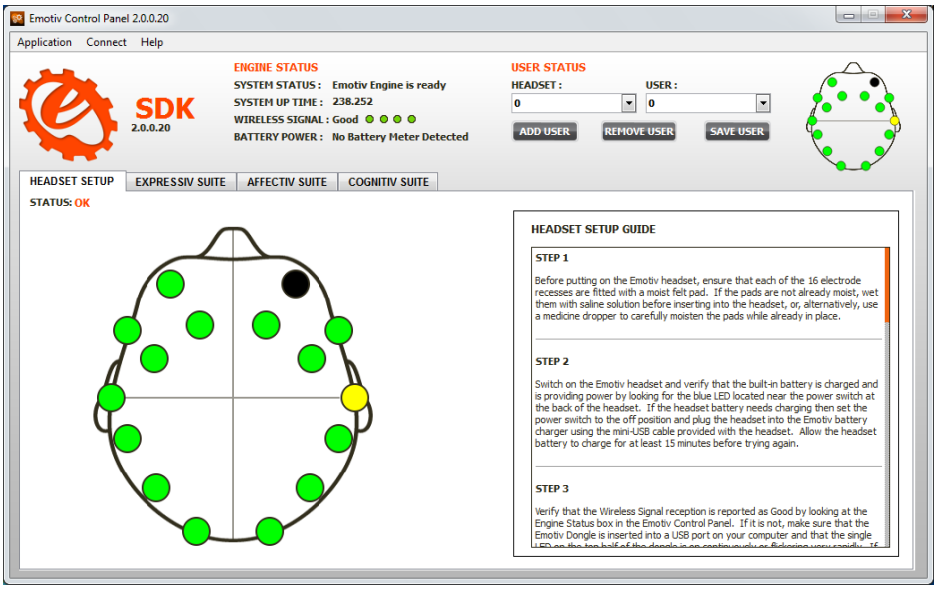


Figura 91: Estado de electrodos del casco Emotiv POC

Antes de probar el control del brazo robótico, se realizan pruebas previas de reconocimiento de gestos en el programa Emotiv Control Panel, en la Figura 92 se muestra el reconocimiento del guiño de ojo derecho, en la Figura 93 se muestra el reconocimiento del guiño del ojo izquierdo.

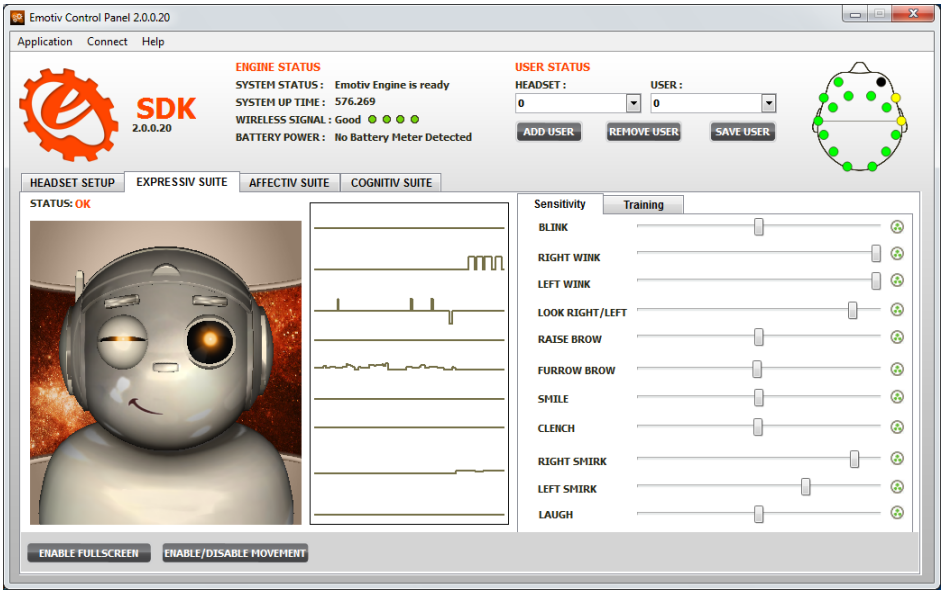


Figura 92: Guiño de ojo derecho en el programa Emotiv Control Panel

Figura 93: Guiño de ojo izquierdo en el programa Emotiv Control Panel

En el programa Emotiv Control Panel también se realizan pruebas de funcionamiento para detección de gestos como mirar hacia la derecha (ver Figura 94) y mirar hacia la izquierda (ver Figura 95).

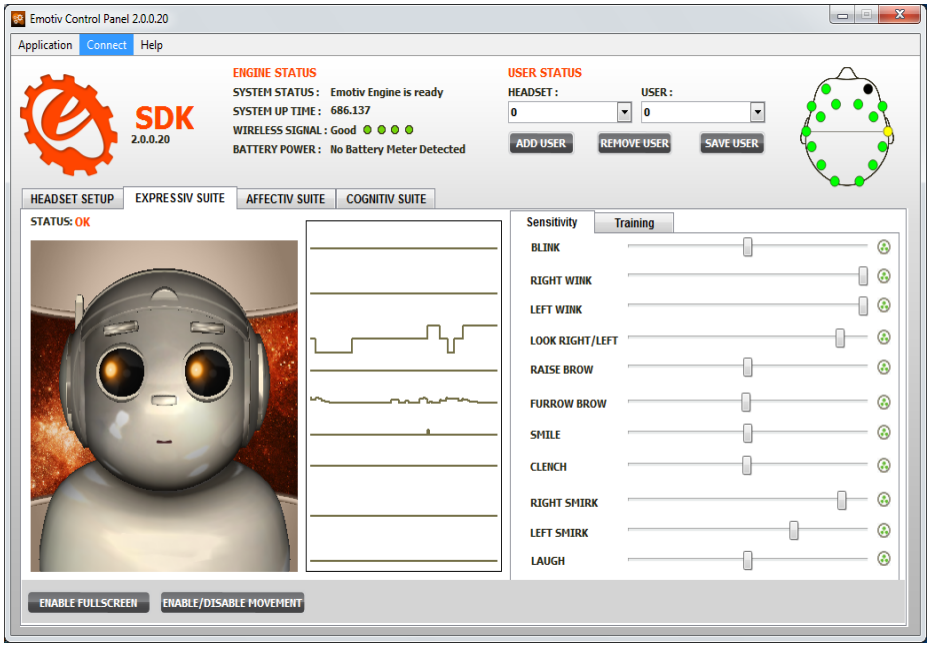


Figura 94: Mirar hacia la derecha en el programa Emotiv Control Panel

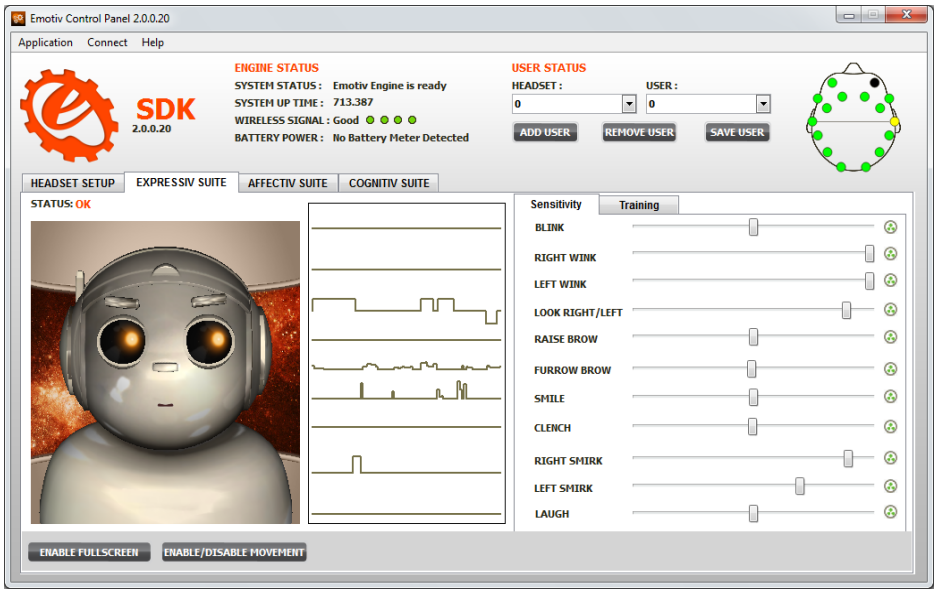


Figura 95: Mirar hacia la izquierda en el programa Emotiv Control Panel

Finalmente se realizan las pruebas de reconocimiento de mueca derecha (ver Figura 96) y mueca izquierda (ver Figura 97). Las pruebas anteriormente mencionadas son necesarias para detectar cada una de las acciones antes de ser utilizadas para el control del brazo robótico, a la vez que el usuario adquiere un entrenamiento previo para el reconocimiento de los gestos faciales a utilizar.

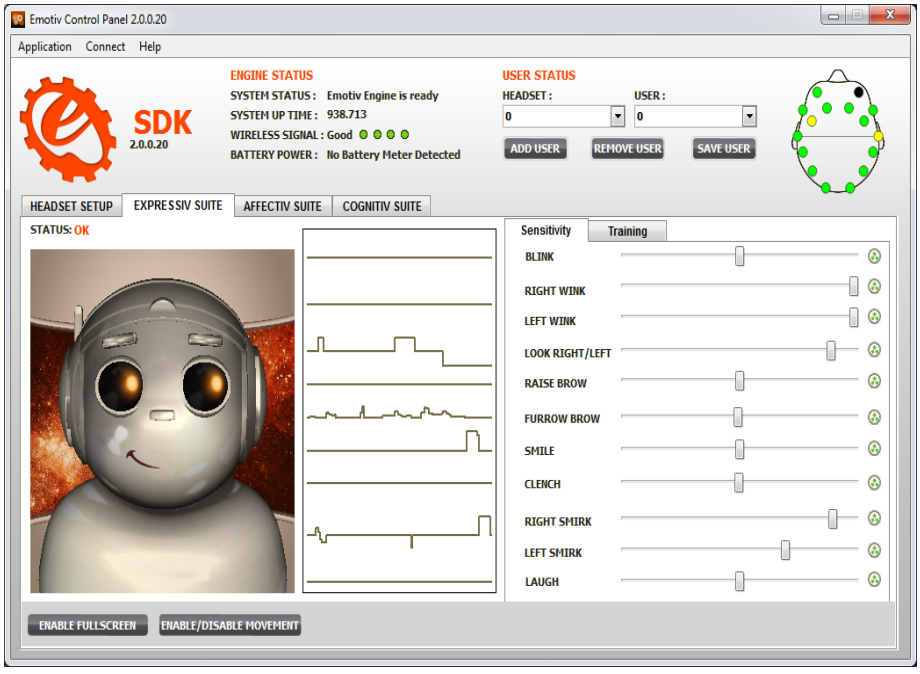


Figura 96: Mueca hacia la derecha en el programa Emotiv Control Panel

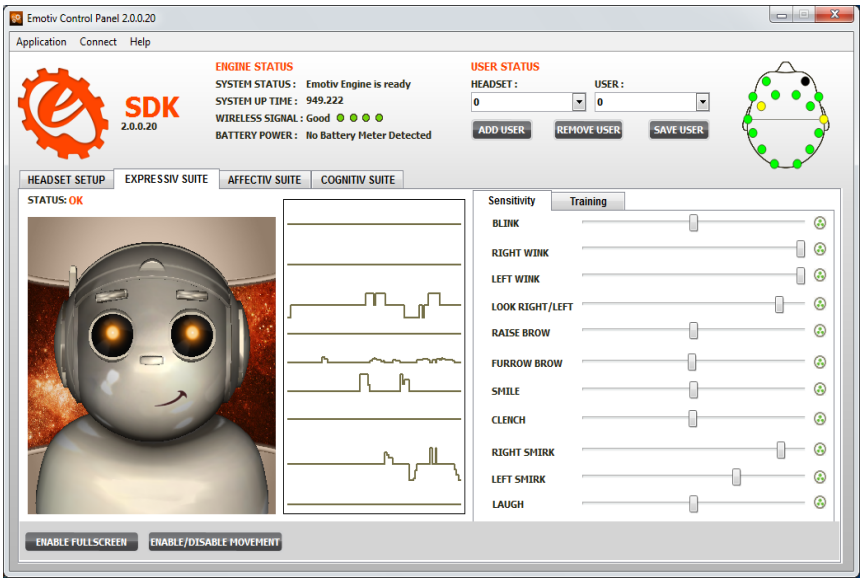


Figura 97: Mueca hacia la izquierda en el programa Emotiv Control Panel

La Figura 98 muestra el movimiento que realiza el brazo robótico hacia adelante al aumentar el valor de la coordenada en el eje X a través de un guiño con el ojo derecho utilizando el casco Emotiv EPOC.

Figura 98: Movimiento que realiza el brazo robótico hacia adelante a través de un guiño del ojo derecho

La Figura 99 muestra el movimiento que realiza el brazo robótico hacia atrás al disminuir el valor de la coordenada en el eje X a través de un guiño con el ojo izquierdo utilizando el casco Emotiv EPOC.

Figura 99: Movimiento que realiza el brazo robótico hacia atrás a través de un guiño del ojo izquierdo

El control mediante el uso del casco Emotiv EPOC permite mover al brazo robótico hacia la izquierda o derecha dependiendo del valor que posea la coordenada del eje Y; al realizar una mueca hacia la derecha el valor de su coordenada aumenta (ver Figura 100), por otro lado con una mueca hacia la izquierda el valor de su coordenada disminuye (ver Figura 101).

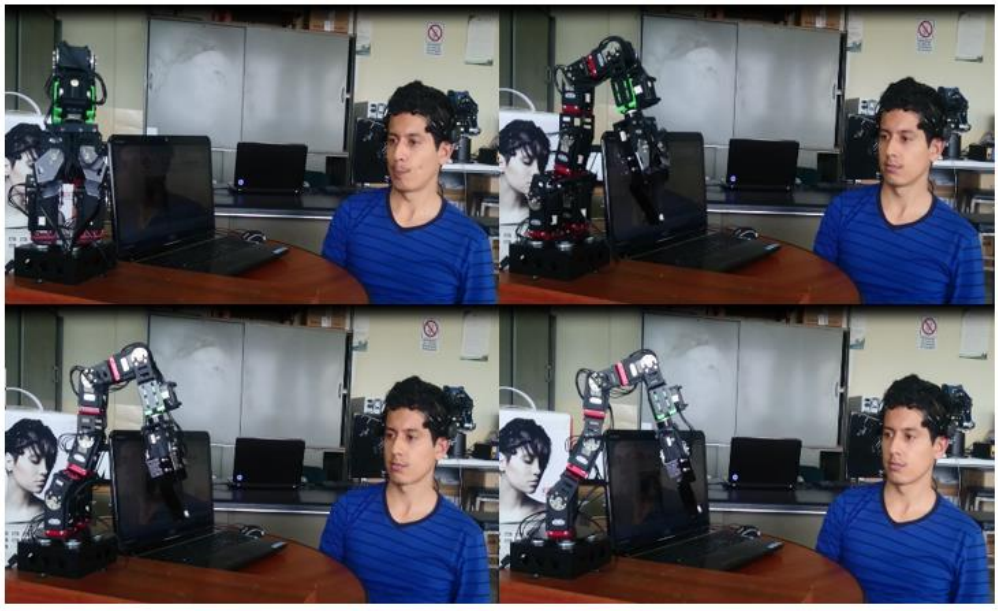


Figura 100: Movimiento que realiza el brazo robótico hacia la derecha a través de una mueca derecha

Figura 101: Movimiento que realiza el brazo robótico hacia la izquierda a través de una mueca izquierda

Para finalizar, el control utilizando el casco Emotiv EPOC permite mover al brazo robótico en el eje Z positivo al mirar hacia la derecha aumenta el valor de su coordenada (ver Figura 102), y al mirar hacia la izquierda disminuye el valor de su coordenada (ver Figura 103).

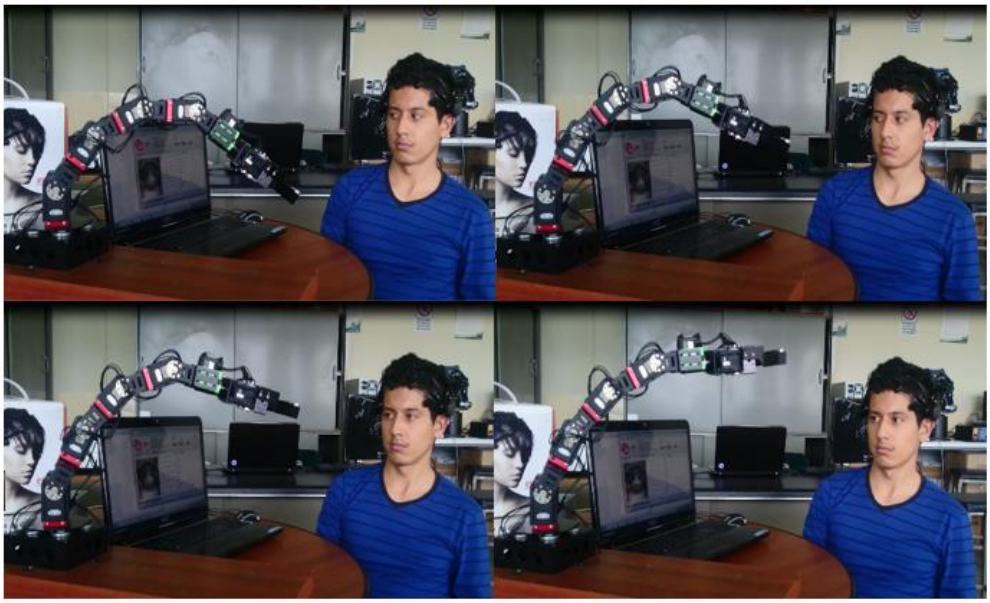


Figura 102: Movimiento que realiza el brazo robótico hacia arriba a través de mirar a la derecha

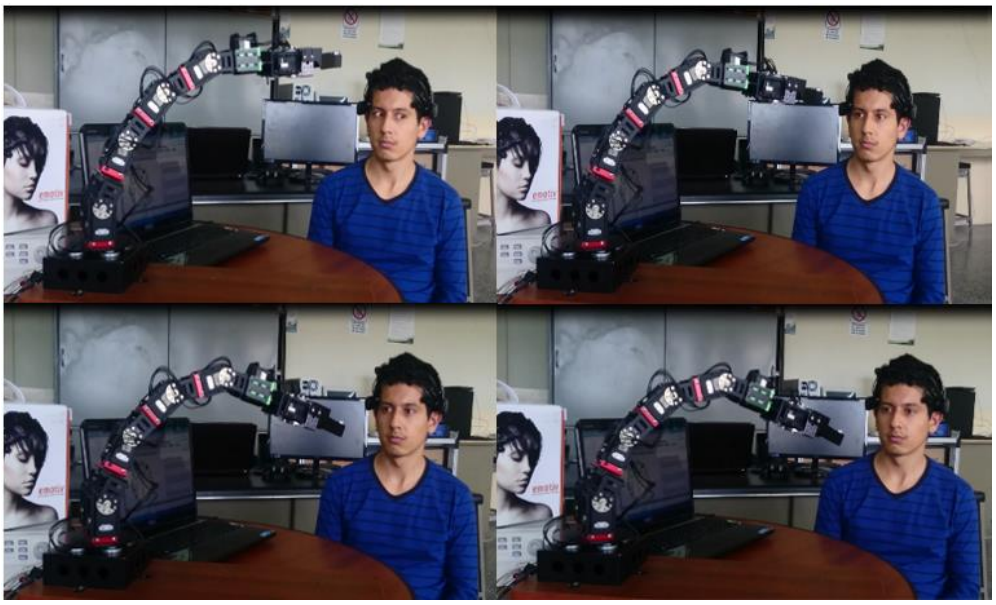


Figura 103: Movimiento que realiza el brazo robótico hacia abajo a través de mirar a la izquierda

4.5 Análisis de resultados

4.5.1 Análisis del control para las trayectorias de silla de montar y circunferencia

Mediante las pruebas prácticas de las secciones anteriores se puede notar el error que se obtiene al comparar la trayectoria realizada por el brazo robótico con la trayectoria hecha por simulación; dando como resultado un error mayor para trayectorias con un radio en el rango de 20 a 29 cm, mientras que para las trayectorias con un radio mayor a 29 cm el error fue cercano a cero, cabe mencionar que el valor de la altura tanto para la silla de montar como para la circunferencia no afecto al error obtenido.

Al momento de ejecutar una trayectoria específica, el brazo robótico la realizó con movimientos suaves y estables en cada una de sus articulaciones, de esta manera se pudo verificar el correcto funcionamiento del algoritmo de control implementado en Python. El usuario puede seleccionar los valores de radio y altura que decida siempre y cuando estos valores se encuentren dentro de los rangos de trabajo que la interfaz lo permita, con esto se asegura que el brazo robótico no sufra de esfuerzos mecánicos indebidos. Adicionalmente es

importante mencionar que la simulación de la trayectoria es realizada una vez que el brazo robótico concluya una silla de montar o una circunferencia.

4.5.2 Análisis del control de posición mediante el uso de un joystick

Para este caso el control implementado es de posición, con lo cual se permite al usuario definir una serie de puntos para completar una trayectoria específica; por ejemplo: recoger objetos utilizando el joystick. Para realizar una tarea eficiente e intuitiva, el usuario debe memorizar la configuración de cada uno de los botones del joystick para que el brazo robótico se pueda movilizar en los ejes X, Y, Z sin dificultad; se debe considerar que los objetos a levantar o recoger no deben ser muy pesados ya que los servomotores podrían entrar en falla y desactivar el torque de la red de servomotores Dynamixel.

El usuario puede utilizar la pinza para sostener objetos tan pequeños o grandes como sea permitido por la estructura mecánica, ya que la pinza tiene un rango de apertura y cierre que limita tomar objetos de distintas medidas, además se tiene la opción de girar la muñeca del brazo robótico en diferentes grados según la necesidad del usuario.

4.5.3 Análisis del control de posición mediante el uso del casco Emotiv EPOC

Al igual que para el joystick el control desarrollado con el casco Emotiv EPOC es de posición, y las acciones para el movimiento del brazo robótico son definidas por acciones faciales emitidas por el usuario; por tal razón el usuario a utilizar el casco debe tener un entrenamiento previo con las acciones establecidas para el control, además se puede hacer uso de herramientas proporcionadas por Emotiv Control Panel para editar la sensibilidad al momento de reconocer las acciones faciales a utilizar. La eficiencia de los movimientos que realiza el brazo robótico en los ejes X, Y, Z dependen directamente de la capacitad que tiene el usuario en controlar sus gestos faciales.

Para evitar que el brazo robótico realice movimientos indebidos y que estos afecten a la estructura mecánica del mismo, se recurre a limitar la distancia de alcance en cada uno de los ejes, con ello se consigue que el usuario al estar utilizando el casco Emotiv EPOC pueda tener un control óptimo y estable del robot.

4.5.4 Análisis de la interfaz gráfica

La interfaz realizada es amigable con el usuario ya que se muestra muchas opciones que pueden ser usadas e interactuar de la mejor manera consiguiendo que cualquier persona que sepa utilizar un computador pueda hacer uso de las funciones que se crearon en la interfaz. Se tiene la ayuda de la representación en 3D de los movimientos realizados por el brazo robótico que visualmente permiten entender lo que se está ejecutando.

CAPÍTULO V

5. CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

- Se estudió el funcionamiento del brazo robótico tomando en cuenta sus cinco primeras articulaciones obviando a la muñeca y pinza del robot, este proceso se lo realizó por medio de cinemática directa en donde se obtuvo su correspondiente relación de velocidades que ubican al extremo operativo del brazo robótico.

- Mediante el estudio y la experimentación se obtuvo conocimientos para la manipulación de las señales obtenidas por el casco Emotiv EPOC que permiten realizar un control de posición tridimensional para el extremo operativo del brazo robótico.

- En Python se desarrolló una plataforma amigable e intuitiva para que el usuario pueda desenvolverse de una manera fácil y sencilla a través de la misma, pudiendo escoger diferentes modos de control para el brazo robótico, entre ellas la realización de diferentes trayectorias, el control de posición por medio de un joystick o a través de las señales cerebrales emitidas por el casco Emotiv EPOC.

- El algoritmo de control implementado para el brazo robótico que tiene como entradas las señales faciales emitidas por el usuario permite tener movimientos suaves y estables en cada uno de sus eslabones, con lo cual nos aseguramos que el brazo no realice movimientos bruscos que afecten a su estructura mecánica así como también a la red de nueve servomotores Dynamixel.

- Para un óptimo control del brazo robótico se utilizó las acciones faciales que mejor resultado dieron al momento de su reconocimiento. Al aumentar

el número de acciones faciales se pueden obtener falsos positivos y con ello acciones erróneas al momento de realizar el control del robot.

- Se realizó pruebas de funcionamiento con cada uno de los modos de control disponibles en este proyecto, comprobando en cada uno de ellos el correcto funcionamiento de los movimientos que ejecutó el brazo robótico utilizado, teniendo en cada prueba un error tolerable.

5.2 Recomendaciones

- Se recomienda establecer una área de trabajo en los planos X, Y, Z para que el brazo robótico no sea afectado por ningún sobre esfuerzo mecánico en su estructura, dando como resultado un error en los servomotores afectados. La estructura mecánica del brazo robótico al estar sujeta a una base de madera no permite trabajar con valores negativos para el eje Z.

- Python puede establecer un enlace de comunicación solo con servomotores Dynamixel que tenga una velocidad de transmisión de 1000000 bps, por tal motivo es importante configurar mediante el software ROBOPLUS la velocidad de transmisión de cada uno de los servomotores.

- Es importante tomar en cuenta que los electrodos del casco Emotiv EPOC deben estar correctamente humedecidos con la solución líquida que viene en el kit de trabajo del casco, con ello se obtiene mejores resultados al momento de captar y reconocer las señales cerebrales, adicionalmente antes de realizar el control del brazo robótico la persona debe estar concentrada en los movimientos que ejecutará el robot por lo que se recomienda que el usuario tenga la mente descansada y libre de distracciones.

- Se recomienda utilizar una computadora con un sistema operativo Windows 7 de 32 bits, ya que Python es muy restrictivo a la hora de comunicarse con sistemas operativos actuales por falta de compatibilidad con drivers.

BIBLIOGRAFÍA

Barrientos, A., Peñín , L. F., Balaguer, C., & Aracil, R. (1997). *Fundamentos de Robótica*. Madrid, España: McGRAW-HILL.

Kelly, R., & Santibáñez, V. (2003). *Control de Movimiento de Robots Manipuladores.* Madrid, España: Pearson Educación.

Stallman, R. (2004). *Software libre para una sociedad libre.* España: Traficantes de sueños.

LINKOGRAFÍA

Abdalá Castillo, Salomón, & Ñeco Caberta, Raúl. (2003). *Caracterización de un robot manipulador articulado.* (Postgrado), Centro Nacional de Investigación y Desarrollo Tecnológico, Cuernavaca, México, D.F. Obtenido de http://www.cenidet.edu.mx/subaca/web-mktro/submenus/investigacion/tesis/3-4%20Salomon%20Abdala%20Castillo%20-%20Raul%20Ñeco%20Caberta.pdf (Recuperado 18/07/2015)

Andaluz, Víctor H, Ortiz, Jessica S, & Sanchéz, Jorge S. (2015). Bilateral Control of a Robotic Arm Through Brain Signals *Augmented and Virtual Reality* (pp. 355-368): Springer. Obtenido de http://link.springer.com/chapter/10.1007/978-3-319-22888-4_26 (Recuperado 20/09/2015)

Barea, RN. (2002). Electroencefalografia, Universidad de Alcalá, Tema 5, 2-24. Obtenido de http://www.bioingenieria.edu.ar/academica/catedras/bioingenieria2/archivos/apuntes/tema%205%20-%20electroencefalografia.pdf (Recuperado 05/06/2015)

Biosemi. (1998). *Products Biosemi*. Obtenido de http://www.biosemi.com/index.htm (Recuperado 03/08/2015)

Calzada Reyes, A. (2004). Técnicas Neurofisiológicas en la Psiquiatría forense. *Quinto Congreso Virtual de Psiquiatría.* Obtenido de http://www.researchgate.net/profile/Ana_Reyes7/publication/267850528_TCNICAS_NEUROFISIOLGICAS_EN_LA_PSIQUIATRA_FORENSE/links/54c12af60cf2d03405c4ed31.pdf (Recuperado 02/06/2015)

Candelas Herías, F., & Corrales Ramón, J. (20 de Septiembre de 2007). *Servomotores.* Obtenido de http://www.aurova.ua.es/previo/dpi2005/docs/publicaciones/pub09-ServoMotores/servos.pdf (Recuperado 06/06/2015)

Consejo Nacional para la Igualdad de Discapacitados. (Septiembre de 2014). *Consejo Nacional de Igualdad de Discapacidades en Ecuador*. Obtenido de http://www.consejodiscapacidades.gob.ec/ (Recuperado 20/11/2014)

Emotiv. (2009). *Products Emotiv*. Obtenido de https://emotiv.com (Recuperado 03/08/2015)

Emotiv Systems. (2012). *EPOC User Manual*. Obtenido de http://emotiv.com/developer/SDK/UserManual.pdf (Recuperado 12/08/2015)

Haskell. (1990). *Organización Haskell*. Obtenido de https://www.haskell.org (Recuperado 05/08/2015)

Instituto nacional de estadísticas y censos. (2010). *Población y Demografía en Ecuador*. Obtenido de http://www.ecuadorencifras.gob.ec/censo-de-poblacion-y-vivienda/ (Recuperado 16/11/2014)

López Segovia, José Luis, Alamilla Santiago, Misael , & Domínguez Vázquez, Juan Francisco. (2007). *Robot Cartesiano: Seguimiento de trayectorias irregulares arbitrarias mediante computadora.* (Pregrado), Universidad Autónoma del Estado de Hidalgo, Carrera de Ingeniería en Electrónica y Telecomunicaciones, Pachuca de Soto, México, D.F. Obtenido de http://www.uaeh.edu.mx/docencia/Tesis/icbi/licenciatura/documentos/Robot%20cartesiano%20seguimiento%20de%20trayectorias.pdf (Recuperado 04/07/2015)

Martínez Castellanos, Diego Alberto. (2012). *Diseño y construcción de prototipo de robot SCARA 3 DOF.* (Pregrado), Universidad Industrial de Santander, Carrera de Ingeniería Mecánica, Bucaramanga, Colombia.Obtenido de http://repositorio.uis.edu.co/jspui/bitstream/123456789/6100/2/145391.pdf (Recuperado 09/08/2015)

Martínez, G., Jáquez, S., Rivera, J., & Sandoval, R. (2008). Diseño propio y Construcción de un Brazo Robótico de 5 GDL. *REVISTA DE INGENIERÍA ELÉCTRICA, ELECTRÓNICA Y COMPUTACIÓN*, 9.Obtenido de http://www.itson.mx/publicaciones/rieeyc/Documents/v4/art2junio08.pdf (Recuperado 06/07/2015)

Medical Computer System. (2013). *Medical Expo*. Obtenido de http://www.medicalexpo.es/prod/medical-computer-systems/product-69255-565055.html (Recuperado 16/07/2015)

Meneses Jiménez, X., Méndez Canseco, M., & Cortés Bringas, E. (Noviembre de 2007). *Diseño y Control de un Robot Paralelo.* Obtenido de http://www.mecamex.net/anterior/ cong06/articulos/ 60742final.pdf (Recuperado 18/09/2015)

Mindo. (2013). *Products Mindo*. Obtenido de http://mindo.com.tw/en/goods.php (Recuperado 19/07/2015)

Moya Pinta, Diego Armando. (2010). *Modelo y análisis cinemático de un robot manipulador esférico industrial aplicando Matlab.* (Pregrado), Universidad Politécnica Nacional. Carrera de Ingeniería Mecánica, Quito,Ecuador.Obtenido de http://bibdigital.epn.edu.ec/handle/15000/2327 (Recuperado05/07/2015)

Neuroelectris. (2011). *Products Enobio*. Obtenido de http://www.neuroelectrics.com/products/enobio/ (Recuperado 19/07/2015)

Organización Mundial de la Salud. (2011). *Organización Mundial de la Salud*. Obtenido de http://www.who.int/features/factfiles/disability/es/ (Recuperado 15/11/2014)

Ponce Jurado, Jairo Vinicio. (2014). *Implementación de una interface cerebro-computador para la detección de posición con la ayuda de las señales EEG.* Universidad de las Fuerzas Armadas ESPE. Carrera de Ingeniería Electrónica en Telecomunicaciones. Obtenido de http://repositorio.espe.edu.ec/xmlui/bitstream/handle/21000/8057/T-ESPE-047748.pdf?sequence=1 (Recuperado 14/07/2015)

Python. (1994). *Organización Python*. Obtenido de https://www.python.org (Recuperado 04/08/2015)

Rivas, D., Alvarez, M., Velasco, P., Mamarandi, J., Carrillo-Medina, J. L., Bautista, V., . . . Huerta, M. (2015, 17-19 Feb. 2015). *BRACON: Control system for a robotic arm with 6 degrees of freedom for education systems.* Paper presented at the Automation, Robotics and Applications (ICARA), 2015 6th International Conference on. Obtenido de http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=7081174&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D7081174 (Recuperado 06/08/2015)

Sánchez, Jonathan González. (2014). Técnicas de toma de datos y análisis de ELECTROENCEFALOGRAFÍA. Obtenido de https://opera.eii.us.es/archivos/sinergia/entregables/2013-2014/Grupo9/Grupo9Memoria1.pdf (Recuperado 04/07/2015)

Velasco Sánchez, Edison Patricio, & Mamarandi Quilo, Javier Alexander. (2015). *Diseño del sistema de control basado en software libre para un brazo robótico de 6 grados de libertad con funcionalidad de mecanizado y paletizado para el Laboratorio de Circuitos Electrónicos de la Universidad de las Fuerzas Armadas-ESPE extensión Latacunga.* (Pregrado), Universidad de las Fuerzas Armadas ESPE Extensión Latacunga. Carrera de Ingeniería en Electrónica e Instrumentación. Obtenido de http://repositorio.espe.edu.ec/bitstream/21000/9298/1/T-ESPEL-ENI-0334.pdf (Recuperado 06/08/2015)

UNIVERSIDAD DE LAS FUERZAS ARMADAS - ESPE

INGENIERÍA EN ELECTRÓNICA E INSTRUMENTACIÓN

CERTIFICACIÓN

Se certifica que el presente trabajo fue realizado por los Sres. Leonardo Antonio Solís Córdova y Jorge Andrés Tapia Herrera.

PhD. Víctor Hugo Andaluz Ortiz

DIRECTOR

APROBADO POR:

Ing. Franklin Silva

DIRECTOR DE LA CARRERA

CERTIFICADO POR:

Dr. Rodrigo Vaca

SECRETARIO ACADÉMICO

UNIDAD ADMISIÓN Y REGISTRO

ANEXOS