ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA, Y ELECTRÓNICA Y TELECOMUNICACIONES

CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

TRABAJO DE TITULACIÓN, PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

TEMA: DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE IMITACIÓN CORPORAL COMPLETA EN TIEMPO REAL CON UN ROBOT HUMANOIDE NAO

AUTOR: CONTERÓN MORALES, BRAYAN JOSELITO

DIRECTOR: ING. IBARRA JÁCOME, OSWALDO ALEXANDER MGs.

SANGOLQUÍ

2020

DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELEOMUNICACIONES

CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

CERTIFICACIÓN

Certifico que el trabajo de titulación, ***"DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE IMITACIÓN CORPORAL COMPLETA EN TIEMPO REAL CON UN ROBOT HUMANOIDE NAO"*** realizado por el señor ***Conterón Morales, Brayan Joselito*** el mismo que ha sido revisado en su totalidad, analizado por la herramienta de similitud de contenido; por lo tanto cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, razón por la cual me permito acreditar y autorizar para que lo sustente públicamente.

Sangolquí, 17 de enero de 2020.

Firma:

.....

Ing. Ibarra Jácome Oswaldo Alexander MGs.

C.C.: 1727296418

DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES
CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

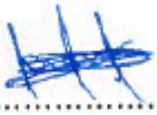
AUTORÍA DE RESPONSABILIDAD

Yo, ***Conterón Morales, Brayan Joselito***, declaro que este trabajo de titulación ***"DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE IMITACIÓN CORPORAL COMPLETA EN TIEMPO REAL CON UN ROBOT HUMANOIDE NAO"*** es de mi autoría y responsabilidad, cumpliendo con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Consecuentemente el contenido de la investigación mencionada es veraz.

Sangolquí, 17 de enero del 2020

Firma:



.....

Conterón Morales Brayan Joselito.

CC: 1003658711

DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES
CARRERA DE INGENIERÍA EN ELECTRÓNICA, AUTOMATIZACIÓN Y CONTROL

AUTORIZACIÓN

Yo, ***Conterón Morales, Brayan Joselito*** autorizo a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de titulación: ***"DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE IMITACIÓN CORPORAL COMPLETA EN TIEMPO REAL CON UN ROBOT HUMANOIDE NAO"*** en el repositorio Institucional, cuyo contenido, ideas y criterios son de mi responsabilidad.

Sangolquí, 17 de enero del 2020

Firma:

.....

Conterón Morales Brayan Joselito.

CC: 1003658711

DEDICATORIA

A mis queridos padres Joselito y Rosa Elena, que me brindaron su amor y apoyo incondicional al observar en mí el potencial de estudiar esta carrera desde que era muy pequeño.

Y en especial a mis hermanas Shira, Mery y Diana, que fueron un ejemplo a seguir para alcanzar los sueños de ser Ingenieros sin importar las dificultades.

AGRADECIMIENTO

En primer lugar, quiero dar gracias a Dios por haberme dado la vida y la sabiduría de haber culminado este proyecto.

A todo el personal docente y administrativo del Departamento de Eléctrica, Electrónica y Telecomunicaciones de la Carrera de Ingeniería en Electrónica, Automatización y Control, por haberme brindado la oportunidad y facilidades de estudiar y trabajar con el ***ROBOT HUMANOIDE NAO.***

Un agradecimiento especial al Ing. Ibarra Jácome Oswaldo Alexander MGs., actual tutor de tesis y al Ing. Andrés Sebastián Erazo Sosa M.Sc., mi tutor anterior, por las guías recibidas en la elaboración del mismo.

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

Actualmente la teleoperación ha incorporado grandes avances tecnológicos, aplicada principalmente en control de robots por humanos, ubicados en diferentes áreas geográficas, en la que la zona en la que se ubique el robot, sea de riesgo o insegura para humanos. En el presente documento se describe el diseño e implementación de un sistema de imitación de movimientos humanos, aplicado a un robot humanoide NAO. Para la captación de los movimientos humanos se diseñó una red de 16 sensores inerciales, los cuales se ubican en posiciones específicas de un humano, la red de sensores inerciales envía la orientación de cada sensor de forma inalámbrica a un software. En el software se recibe los datos de orientación de cada sensor, los distribuye y los asigna en cada articulación del robot NAO según corresponda, seguidamente se los condiciona de acuerdo a las limitaciones angulares de cada articulación, para luego proceder a calcular la posición de los efectores de cada cadena cinemática, calcula el centro de masa de cada eslabón para finalmente calcular el centro de masa total del robot, finalmente se calcula la cinemática inversa de las cadenas cinemáticas de la cabeza, brazo izquierdo y pierna izquierda. Por último, envía los valores angulares de cada articulación a un entorno de simulación del robot NAO y a un programa que controla los movimientos del robot NAO real, para que puedan reproducirlos en tiempo real.

Palabras clave:

- **SENSORES INERCIALES**
- **TELEOPERACIÓN DE ROBOTS HUMANOIDES**
- **CINEMÁTICA DEL ROBOT NAO**

ABSTRACT

Currently, teleoperation has incorporated great technological advances, mainly applied in robot control by humans, located in different geographical areas, in which the area where the robot is located, is at risk or unsafe for humans. This document describes the design and implementation of a human movement imitation system, applied to a NAO humanoid robot. For the capture of human movements, a network of 16 inertial sensors was designed, which are located in specific positions of a human, the inertial sensor network sends the orientation of each sensor wirelessly to a software. In the software, the orientation data of each sensor is received, distributed and assigned in each joint of the NAO robot as appropriate, then it is conditioned according to the angular limitations of each joint, then proceed to calculate the position of the effectors of each kinematic chain, calculate the center of mass of each link to finally calculate the center of mass of the robot, finally calculate the inverse kinematics of the kinematic chains of the head, left arm and left leg. Finally, it sends the angular values of each joint to a simulation environment of the NAO robot and to a program that controls the movements of the real NAO robot, so that they can reproduce them in real time.

Keywords:

- **INERTIAL SENSORS**
- **TELEOPERATION OF HUMANOID ROBOTS**
- **ROBOT NAO KINEMATICS**

CAPÍTULO I

INTRODUCCIÓN

1.1. Antecedentes

La caracterización del movimiento humano tiene un extenso campo de aplicaciones, tales como el análisis del entrenamiento de un atleta, aplicaciones biomédicas, posicionamiento humano, cuantificación del movimiento humano, etc. Se han desarrollado sistemas de análisis y seguimiento del movimiento humano en base al uso de sensores inerciales, ópticos y los goniómetros, los cuales proporcionan información del estado y los movimientos que realizan los humanos, para poder ser analizados y caracterizados (Chen, 2013).

Los sistemas que captan el movimiento de los humanos usan principalmente sensores inerciales, los cuales están conformados por giroscopios, acelerómetros y magnetómetros que permiten estimar la orientación de un cuerpo sólido (Wenk, 2016). Actualmente se ha desarrollado los siguientes sistemas:

- El Sistema MVN de Xsens consiste en un conjunto de 17 de sensores inerciales ubicados en todo el cuerpo, los cuales permiten estimar la orientación de cada segmento del cuerpo, funciona en tiempo real a una frecuencia de muestreo de 120 Hz, consta con un software MVN Studio, en el cual se procesan los datos y genera movimientos en 3D (Roetenberg, Luinge, & Slycke, 2009).
- El sistema desarrollado por TECHNAID consiste en un conjunto de 16 sensores inerciales ubicados en todo el cuerpo, usa un concentrador con el

cual se obtiene una frecuencia de muestreo de 500 Hz y envía los datos de forma inalámbrica, consta con un software Tech MCS Studio que permite visualizar y capturar los datos en tiempo real y genera gráficas mediante avatares en 3D (Technaid, 2016).

- Se desarrolló una red de sensores de hasta 15 sensores IMU's, con una frecuencia de muestreo de 100 Hz, se tiene como referencia un sistema de coordenadas del sensor y un sistema de coordenadas conjuntas, con el cual se mejora la precisión de seguimiento de orientación, finalmente se usa el filtro de Kalman extendido para los datos obtenidos por el acelerómetro y el giroscopio (Prayudi & Kim, 2012).

Los robots humanoides han sido desarrollados principalmente con fines de investigación en robótica, principalmente se centran en realizar movimientos similares a los que realizan los humanos de forma natural utilizando altos grados de libertad. En base al uso de sistemas de seguimiento del movimiento humano se puede captar las posturas y transferirlas al robot, siendo este proceso muy complejo debido a la diferencia de la cinemática humana y humanoide; además, las configuraciones de los movimientos que pueda realizar el robot deben tener en cuenta las limitaciones mecánicas que tiene el robot (Jin, Dai, Liu, & Wang, 2016). La imitación de los movimientos de todo el cuerpo con un robot humanoide requieren tomar en cuenta el caminar dinámico, controlando el centro de masa del robot, para lo cual se requiere el cálculo de trayectorias conjuntas que controlen el movimiento del robot en base a la imposición de restricciones de estabilidad y respeten sus limitaciones mecánicas. El movimiento del robot tiende a ser más similar si se utilizan todos los grados de libertad (DOF) posibles, aunque esto implique una mayor

cantidad de cálculos para el control de sus movimientos; los criterios de estabilidad utilizan el Zero Moment Point (ZMP) y el Center of Mass (CoM) (Teachasrisaksakul, Zhang, Lo, & Yang, 2015).

Actualmente se han realizado diferentes trabajos de investigación en la captura de movimiento de segmentos del cuerpo humano para controlar un robot humanoide, en los cuales se presenta los diferentes tipos de captura y técnicas usadas para el control del robot, teniendo como principal meta la estabilidad del robot. Entre las investigaciones que destacan están las siguientes:

- Se realizó un enfoque que permite imitar los movimientos en tiempo real con un robot humanoide NAO, en el cual se consideró las posiciones de los en efectores, tomando en cuenta el centro de maso (CoM), usando la cinemática inversa para generar los ángulos de cada articulación, para las cuatro cadenas cinemáticas del robot, tomando en cuenta el centro de masa y estabilidad del robot, se usó como sistema de captura de movimientos el sistema MVN de sensores inerciales desarrollado por Xsens (Koenemann, Burget, & Bennewitz, 2014).
- Se desarrolló un sistema de aprendizaje con el uso de redes neuronales implementado en Matlab, un sistema de captura de movimiento del cuerpo humano mediante el uso de un sensor Microsoft Kinect y se realizó el cálculo de la cinemática inversa para obtener los ángulos de cada articulación, usando un robot humanoide NAO (Montalvo López, 2017).

- Se presentó una estrategia que permite el cambio de soporte durante la imitación en línea mientras a su vez se mantienen las matrices del movimiento humano, se calcula de forma paralela la cinemática inversa, para poder así colocar a los pies en diferentes posiciones con cada cambio de soporte (Poubel, Sakka, Ćehajić, & Creusot, 2014).
- Se proporcionó un marco de optimización para la generación de movimientos de la parte superior del robot humanoide de movimientos captados en un humano, y la generación del movimiento de la parte inferior dando primitivas de la pierna derecha, que a su vez tiene en cuenta las limitaciones mecánicas del robot (Suleiman, Yoshida, Kanehiro, Laumond, & Monin, 2008).

Tomando en cuenta estos antecedentes, y con el objetivo de superar ciertas dificultades expuestas y los desafíos en la imitación del movimiento corporal con un robot humanoide, se propone en esta investigación realizar un sistema de captura de movimientos de todo el cuerpo humano y que realice el control de movimiento de un robot humanoide NAO.

1.2. Justificación e importancia

Los sistemas más usados de obtención del movimiento humano, son los sistemas ópticos y los sistemas con sensores inerciales, de los cuales mediante una comparación por el método de Dominic se obtuvo que los sensores inerciales destacan por su precio, movilidad, integración, precisión, robustez y facilidad de uso en comparación con los demás sistemas (Vivas Mateos, 2016). Por lo tanto, se optará por usar sensores

inerciales de movimiento, a los cuales se los configurará como una red de 16 sensores ubicados en todo el cuerpo.

El aprendizaje de los robots principalmente recae en la imitación de los movimientos realizados por los humanos, así como la amplia gama de comportamientos, desde la interacción social hasta el uso de herramientas teleoperadas. Esto se logra con la captura de movimientos humanos y transformarlos a información cinemática de todo el cuerpo a través de sistemas de captura de movimiento, en este caso con el uso de sensores inerciales; aunque imitar este movimiento con una dinámica de “robot estable” es un problema mucho más difícil (Chalodhorn & Rao, 2009).

La locomoción de los robots humanoides específicamente los movimientos que requiere para caminar que son los de marcha bípedo, tienden a imitar los movimientos de marcha humanos ya que se consideran los más eficientes, para lo cual requieren de un paso estable y postura correcta, esto se logra mediante un control de movimiento que tenga en cuenta la estabilidad del robot ya sea un caminar estático o dinámico, calculando los movimientos que debe realizar en base a su centro de masa y su cinemática inversa (Kah & Nasiruddin, 2017).

Esta investigación se presenta con la finalidad de realizar un sistema que imite los movimientos corporales de un humano con un robot humanoide, el cual contribuya al desarrollo de sistemas con patrones de movimientos humanos con el uso de robots humanoides que puedan imitar movimientos de una forma estable y eficiente.

1.3. Alcance del proyecto

El presente trabajo de investigación, tiene como objetivo la realización de un sistema de imitación de movimientos del cuerpo humano con el uso de una red de 16 sensores inerciales para capturar los movimientos corporales. Los datos adquiridos por la red de sensores se procesarán con el uso de un controlador programable, el cual además también enviará los datos procesados por comunicación inalámbrica a un computador, en el cual con el uso de un software, el cual procesará los datos para obtener los ángulos de cada articulación del cuerpo humano, para obtener los ángulos de movimiento que debe realizar el robot humanoide.

El proyecto a realizar está dividido en tres etapas las cuales se detallan a continuación:

- En la primera etapa del proyecto, se realizará el diseño e implementación de una red de 16 sensores inerciales y el procesamiento para la obtención de la orientación y posición de cada segmento del cuerpo y la cinemática de un humano; lo cual compete: la adquisición, transmisión y procesamiento adecuado de los sensores inerciales (IMUs) y el diseño estructural para la red de sensores que se ubicarán en todo el cuerpo.
- En la segunda etapa se realizará el cálculo de la cinemática directa, cinemática inversa y centro de masa, en base a los datos obtenidos de orientación y posición de cada segmento del cuerpo humano.

- En la tercera etapa se realizará la unión de la red de sensores inerciales con la simulación del robot Nao, el cálculo de la cinemática directa, centro de masa y cinemática inversa, y, por último, con el robot NAO real.
- La última parte del proyecto se realiza las pruebas de funcionamiento del robot en tiempo real, en donde se medirá la rapidez de seguimiento de los movimientos del robot en comparación con los del cuerpo humano.

1.4. Objetivos

1.4.1. Objetivo General

- Desarrollar un sistema capaz de imitar los movimientos corporales usando un robot humanoide NAO en tiempo real.

1.4.2. Objetivos Específicos

- Analizar e identificar los movimientos mecánicos que puede realizar el robot NAO, tomando en cuenta su centro de masa y su estabilidad.
- Diseñar e implementar una red de 16 sensores inerciales que permitan la obtención de datos angulares y una comunicación inalámbrica.
- Adquirir y procesar los datos de los sensores inerciales para obtener la orientación y posición de cada segmento del cuerpo humano.
- Diseñar e implementar un sistema que controle los movimientos de un robot humanoide NAO, capaz de imitar los movimientos corporales de un humano, tomando en cuenta sus restricciones mecánicas y estabilidad.

CAPÍTULO II

ESTADO DEL ARTE

2.1. Introducción

En el año 1948 se desarrolló el primer telemanipulador por R. C. Goertz del Argonne National Laboratory, capaz de manipular elementos radioactivos en zonas peligrosas y así evitar cualquier riesgo al operador, se encontraba formado por un dispositivo mecánico el cual permitía que el operador mediante el uso de transmisiones mecánicas pueda controlar los movimientos del telemanipulador desde una zona segura, seguidamente se desarrollaron avances significativos en el año 1954 con el reemplazo de la transmisión mecánica por una eléctrica, sus aplicaciones en ese entonces se encontraban en la industria nuclear, submarina y espacial. Un telemanipulador requiere de un control continuo por parte de un operador, que realice un control supervisado o por medio de la telepresencia realice un control supervisado desde otra zona geográfica. El reemplazo del operador por un programa de ordenador que controle las acciones y movimientos del manipulador dio paso al desarrollo de los robots. (Barrientos, Penin, Balaguer, & Santoja, 2007).

La teleoperación consiste en que un robot pueda ser controlado por un humano ubicado en una zona diferente en la que se encuentre el robot, sea una zona que presente algún riesgo para el humano o insegura, y que esta conlleve un alto grado de programación para que el robot pueda trabajar de forma autónoma, para lo cual se realiza un control de tipo maestro – esclavo, en la que el robot trabaja como esclavo y el humano como maestro. (Turner, Findley, Griffin, Cutkosky, & Gomez, 2000)

Figura 1. Imitación del movimiento del cuerpo con un robot humanoide.
Fuente: **(Koenemann, Burget, & Bennewitz, 2014)**

Actualmente los sistemas de teleoperación incorporan grandes avances tecnológicos, llegando a usar sistemas que puedan censar los movimientos del operador usando visión artificial, sensores inerciales y exoesqueletos, para aplicaciones en las que se requieran realizar movimientos que se semejen los movimientos humanos, además se pueden utilizar sistemas basados en el uso de joysticks, botones e interfaces gráficas para controlar al robot, permitiéndoles así la capacidad de interactuar con facilidad con humanos y operar en entornos reales.

Una de las principales aplicaciones de la teleoperación se desarrolla con el uso de robots humanoides, ya que estos al poseer una arquitectura humanoide y similar a la de los humanos, son capaces de realizar movimientos y tareas similares a las de los humanos, para la cual se utilizan sistemas capaces de mapear las posturas humanas,

algoritmos de aprendizaje y control de estabilidad con la finalidad de que el robot sea capaz de imitar los movimientos que realice un humano usando un aprendizaje por imitación.

2.2. Sistemas de capturas de movimiento del cuerpo humano

La captura y análisis del movimiento del cuerpo humano consiste en censar las posiciones y orientaciones de sus articulaciones al moverse, esto permite obtener las trayectorias de sus efectores, con la finalidad de obtener los patrones de movimiento que realiza al caminar, manipular objetos, posturas, etc. Esta información puede ser aplicada en diferentes áreas tales como la medicina con terapias de rehabilitación de personas y diagnóstico de patologías motoras, en el deporte con el análisis de los movimientos que realiza un deportista para poder mejorar su rendimiento y evitar posibles lesiones, en el cine con la realización de películas con personajes en 3d. (Vivas Mateos, 2016). En el ámbito de la robótica desempeña un papel muy importante ya que permite que un robot sea capaz de realizar movimientos similares a los que realiza un humano gracias al análisis y captura de sus movimientos, sin embargo, la cinemática y dinámica entre un robot humanoide y un humano son diferentes, lo que implica que una fiel reproducción de los movimientos del humanoide sean inestables, por lo tanto previo a la reproducción de movimientos se debe de realizar un análisis, corrección y control de las trayectorias con la finalidad de que el robot pueda realizar movimientos estables sin que pierda su equilibrio. (Vuga, y otros, 2013).

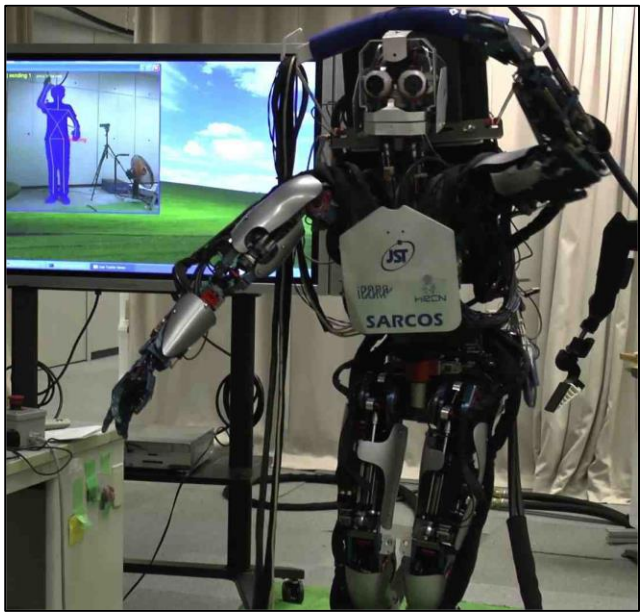


Figura 2. Imitacion de un robot.
Fuente: **(Vuga, y otros, 2013)**

Los sistemas de captación de movimientos tuvieron sus orígenes con el uso de herramientas cinematográficas, las cuales se basaban en el análisis de varias fotos tomadas en intervalos de tiempo a un objeto determinado en movimiento, actualmente se desarrollaron nuevos métodos y sensores, de entre los cuales los que más se destacan son: Sistemas basados en el uso de sensores inerciales, sistemas ópticos y sistemas con goniómetros. (Aventin, 2015).

Los sistemas con sensores inerciales son unidades de medición inercial (IMU), están conformados por un acelerómetro que mide la aceleración lineal, un giroscopio que mide la velocidad angular y magnetómetro que mide los campos magnéticos con referencia al norte magnético, para obtener estimaciones precisas de su orientación sin deriva se requiere procesar las señales descritas anteriormente, para lo cual se usan métodos y algoritmos que las combinan. Son sistemas micro electromecánicos (MEMS)

dándole un tamaño pequeño el cual le da una fácil manipulación y flexibilidad, además presentan un bajo consumo. Gracias a su practicidad han sido aplicados en su mayoría para capturar y analizar movimientos de objetos. (Roetenberg, Luinge, & Slycke, 2009).

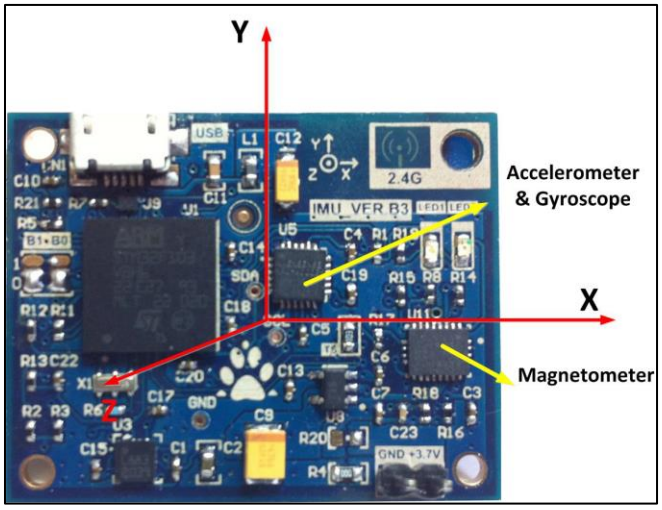


Figura 3. Vista de un IMU y sus coordenadas.
Fuente: **(Chen, 2013)**

Los sistemas ópticos se basan en el uso de cámaras y el análisis de sus fotografías, con la finalidad de obtener un modelo biomecánico de un humano y detectar puntos específicos, con el uso de varias cámaras se logra obtener imágenes 3D lo cual facilita la determinación de dichos puntos con su respectiva posición en un espacio 3D. Actualmente el sistema más usado de tipo óptico es el sensor Kinect desarrollado por Microsoft en 2009, ya que cuenta con sensores para obtener imágenes 3D obteniendo así la presencia, gestos corporales y movimientos de un humano, principalmente usados en el control videojuegos, programas educativos y teleoperación de robots. (Manasrah, 2012).

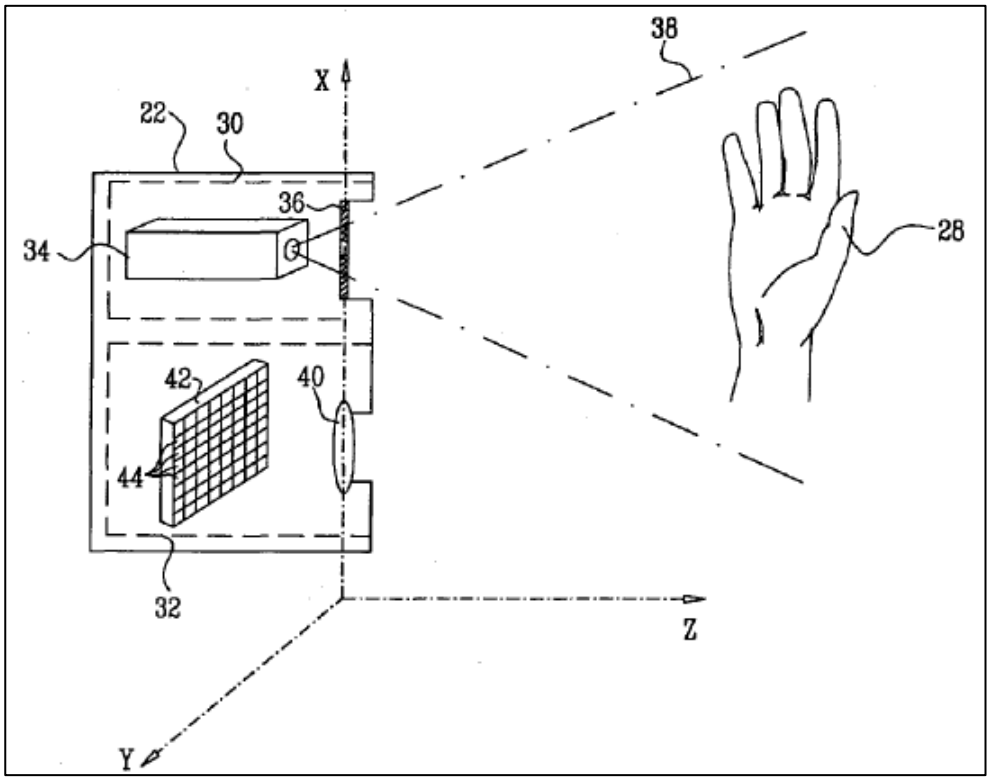


Figura 4. Representación simple del funcionamiento de Kinect.
Fuente: **(Manasrah, 2012)**

Un goniómetro está conformado por un potenciómetro conectado a una articulación y a dos barras en donde se medía la variación de resistencia con la finalidad de obtener el ángulo comprendido entre ambas barras, fueron utilizados en la medición angular de tres planos en las articulaciones de la cadera, rodilla y tobillo. (Isacson, Gransberg, & Knutsson, 1986).

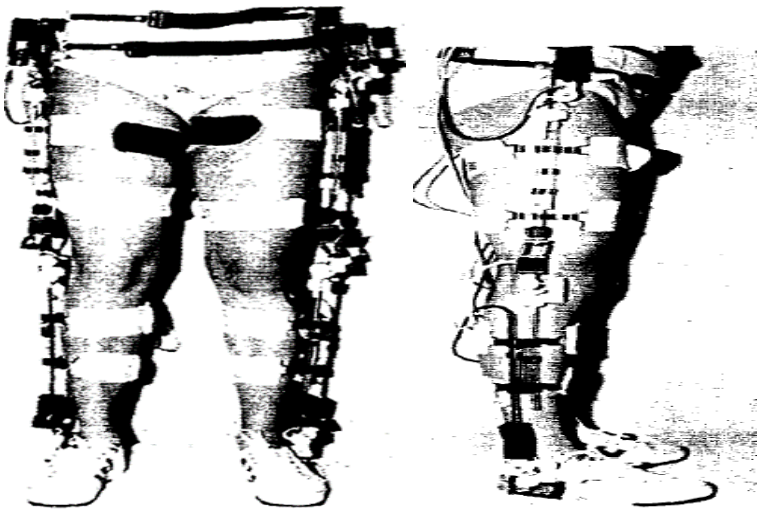


***Figura 5*.** Vista Frontal y lateral de goniómetros.
Fuente: **(Isacson, Gransberg, & Knutsson, 1986)**

El mejor sistema de captación de movimientos de un humano en base a una comparación que se realizado usando el método de Dominic, se destaca por su movilidad, precio e integración con plataformas robóticas, los sistemas realizados con sensores inerciales. (Vivas Mateos, 2016). Por lo tanto, para la realización del sistema de captación de movimientos realizado en la presente tesis se realizó con el uso de un arreglo de IMU's usando el sensor inercial MPU9250.

2.2.1. Sistemas inerciales

Un sistema inercial está formado por un conjunto de sensores como el acelerómetro, giroscopio y magnetómetro, los cuales se detallan a continuación:

- Acelerómetro: El acelerómetro es un sensor electromecánico sensible a las fuerzas de la aceleración estática que corresponde a la fuerza de gravedad que actúa sobre el objeto, y la aceleración dinámica que corresponde al movimiento del objeto. Está formada por placas capacitivas internas, tanto fijas como móviles, las cuales se

sujetan a resortes, su movimiento depende de las fuerzas de aceleración que actúan sobre el objeto, lo cual provoca un cambio de capacitancia entre las placas. Puede medir la aceleración en tres ejes: X, Y y Z. (Guallichico & Utreras, 2013).

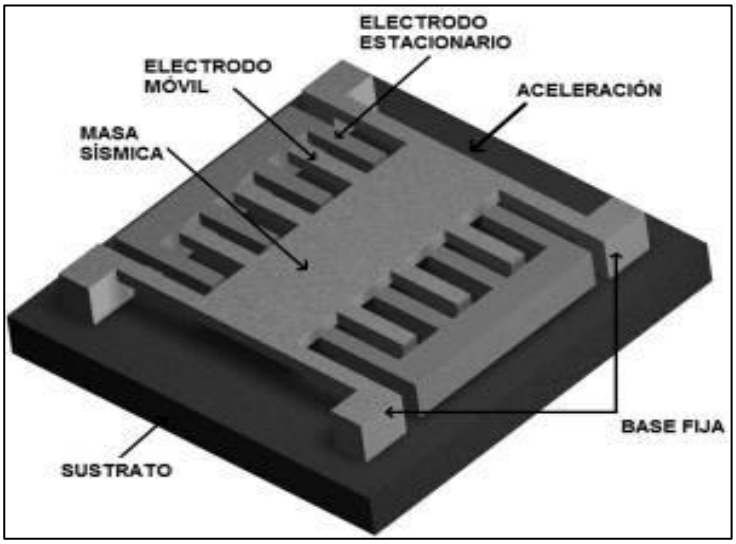


Figura 6. Sistema microelectromecánico.
Fuente: **(Aventin, 2015)**

- Giroscopio: El principio de funcionamiento de un giroscopio se basa en la conservación del momento angular, permite medir la rotación con respecto a los tres ejes: X, Y y Z. Solo pueden medir la velocidad angular, como se aprecia en la Figura 7, al realizarse un giro en el sensor, se desplaza una masa interna la cual provoca señales eléctricas proporcionales a la velocidad angular. (Bernal, 2014).

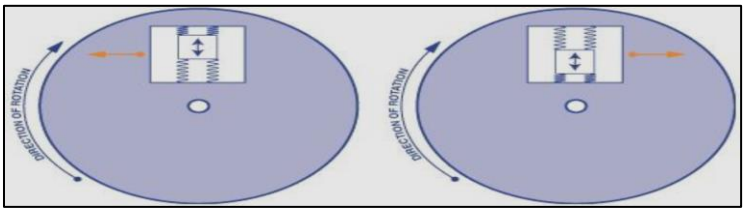


Figura 7. Funcionamiento giroscopio MEMS
Fuente: **(Aventin, 2015)**

- Magnetómetro: Un magnetómetro puede medir la intensidad del campo magnético de un objeto determinado, y otras pequeñas perturbaciones tales como la radiación electromagnética que se propaga en el ambiente, actualmente los magnetómetros MENS pueden medir la intensidad del campo magnético terrestre en los 3 ejes, que varía entre 25,000 y 65,000 nT. La medición se logra gracias al uso de magneto-resistencias que varían su valor en función del campo magnético terrestre que las atraviesa en su dirección. (Bernal, 2014).

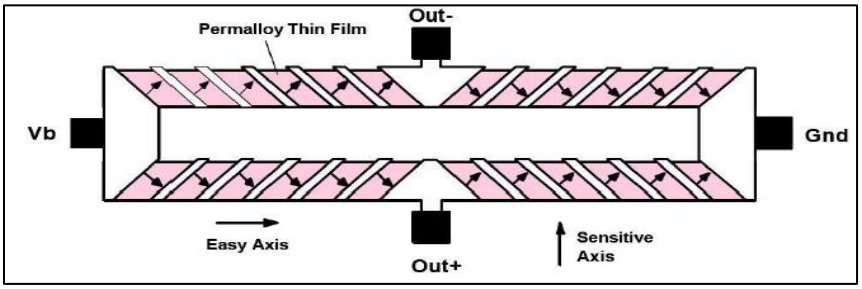


Figura 8. Estructura interna de un magnetómetro MEMS
Fuente: **(Bernal, 2014)**

2.3. Sensor MPU9250

El módulo MPU9250 es un conjunto de sensores de medición inercial de 9 ejes, formado por un acelerómetro de 3 ejes, un giroscopio de 3 ejes y un magnetómetro de 3 ejes, integra un chip procesador digital de movimientos (DMP), el cual puede realizar

complejos algoritmos de captura de movimientos, su comunicación y configuración la realiza mediante las interfaces: I2C y SPI, además puede interactuar con múltiples sensores digitales a través de un puerto auxiliar I2C. Cuenta también con 3 convertidores se señal analógica a digital de 16 bits para las señales medidas por los 3 sensores, con filtros digitales programables e interrupciones.

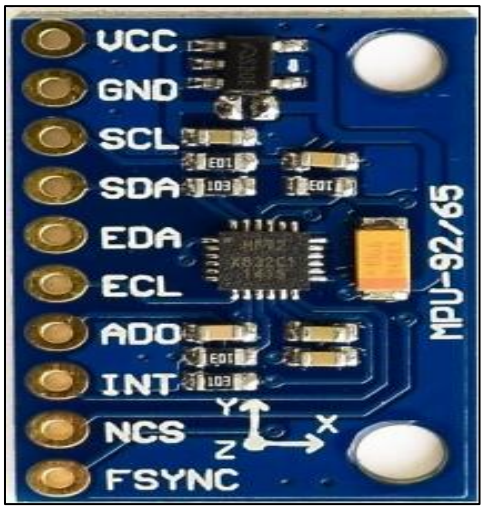


Figura 9. Sensor MPU9250

El diagrama de bloques de la estructura interna del módulo MPU9250 se presenta en la siguiente (Figura 10):

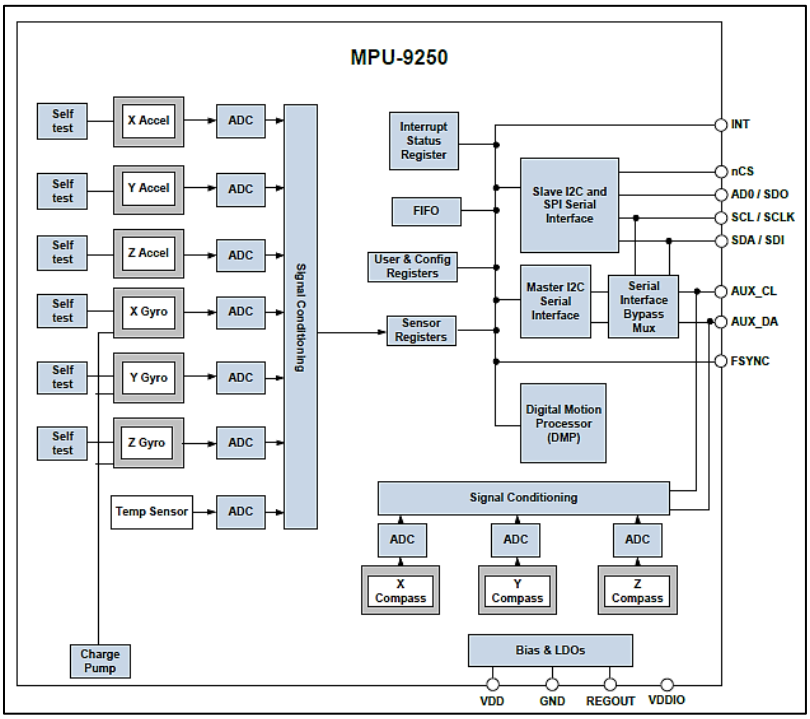


Figura 10. Diagrama de bloques del módulo MPU9250.
Fuente: **(InvenSense, 2016)**

La orientación del acelerómetro, giroscopio y magnetómetro en sus tres ejes: X, Y y Z, con respecto a la estructura del módulo en general, se presentan en las (Figuras 11 y 12), que se presentan a continuación.

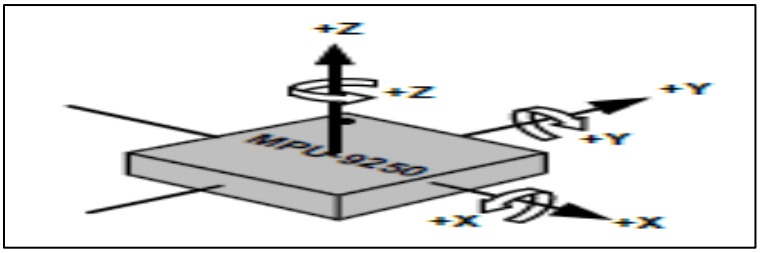


Figura 11. Orientación de acelerómetro y giroscopio.
Fuente: (InvenSense, 2016)

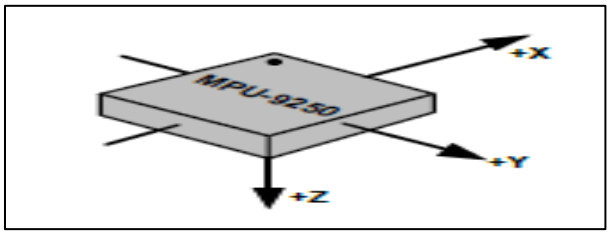


Figura 12. Orientación del magnetometro
Fuente: **(InvenSense, 2016)**

2.3.1. Características

Las características generales son:

- Interface auxiliar de tipo maestro I2C para la conexión con sensores externos.
- Consumo de corriente en funcionamiento de 3.5 mA.
- Rango de voltaje de alimentación: 2.4 – 3.6 [V].
- Cuenta con un buffer de tipo FIFO de 512 bytes para aplicaciones de lectura en ráfaga.
- Sensor de temperatura con salida digital.
- Filtros programables en diferentes rangos de frecuencia para el giroscopio, acelerómetro y el sensor de temperatura.
- Frecuencia de muestreo para comunicaciones rápidas con los registros usando el bus I2C de 400 KHz.
- Frecuencia de muestreo para comunicaciones con los registros usando el bus SPI de 1 MHz.

Las características de los sensores se presentan en la (Tabla 1).

Tabla 1

Características de los sensores que combinan al módulo MPU9250

Características	Acelerómetro	Giroscopio	Magnetómetro
Escala completa programable	± 2 g ± 4 g ± 8 g ± 16 g	$\pm 250\ °/s$ $\pm 500\ °/s$ $\pm 1000\ °/s$ $\pm 2000\ °/s$	$\pm 4800\ \mu T$
Factor de escala de sensibilidad	$16384\ LSB/s$ $8384\ LSB\ °/s$ $4096\ LSB\ °/s$ $2048\ LSB\ °/s$	$131\ LSB/(°/s)$ $65.5\ LSB/(°/s)$ $32.8\ LSB/(°/s)$ $16.4\ LSB/(°/s)$	1
Corriente en funcionamiento	$450\ \mu A$	$3.2\ mA$	$280\ \mu A$
Corriente en reposo	$8\ \mu A$	$8\ \mu A$	No especificado
Resolución de salida	16 bits	16 bits	$14\ bits$ $(0.6\ \mu T/LSB\ )$

Fuente: **(InvenSense, 2016)**

2.3.2. Configuración

El módulo MPU9250 cuenta con dos interfaces de comunicación, para el desarrollo de la presente tesis se usó la interfaz I2C, por lo cual no se detallará la interfaz SPI. Cuando el módulo se comunica con un controlador usando el bus I2C, trabaja como esclavo y el controlador como maestro. Su dirección como esclavo es de 7 bits, en donde el último bit menos significativo depende del nivel lógico al que se conecte al pin AD0, si el pin AD0 se encuentra conectado a nivel bajo su dirección en hexadecimal será 0x68, si se encuentra conectado a nivel alto su dirección en hexadecimal será 0x69.

Todas las configuraciones que se pueden realizar al módulo son usando el bus I2C, para lo cual es necesario primero realizar una configuración inicial con los parámetros generales del módulo, la frecuencia de muestreo, la sensibilidad del

acelerómetro y sus respectivos filtros, una vez realizada la configuración al módulo. Si se requiere trabajar con el magnetómetro se debe tener en cuenta que la frecuencia de salida de datos del módulo no debe ser mayor a los 100 Hz, ya que los datos del magnetómetro se sobrescribirían, por lo tanto, es necesario configurar un divisor de frecuencia en el módulo, la frecuencia de salida de datos del módulo se controla con la siguiente ecuación:

$$\boldsymbol{Frecuencia\ de\ salida} = \frac{\boldsymbol{Frecuencia\ interna}}{\mathbf{1} + \boldsymbol{Divisor\ de\ frecuencia}} \qquad (1)$$

Una vez realizada la configuración del módulo y sus sensores, se puede empezar a realizar la lectura de los datos de los sensores.

2.3.3. Procesamiento de datos en sistemas inerciales

El procesamiento de datos de los sensores inerciales se lo realiza con la finalidad de obtener la orientación y posición relativa del sensor, para lo cual se utilizan métodos y algoritmos que procesen los datos extraídos del acelerómetro, giroscopio y magnetómetro, los métodos más usados son el Filtro de Kalman y el filtro complementario.

Rudolph Kalman en 1960 realizó un filtro que usaba el método de mínimos cuadrados para la estimación de cada dato en base a un modelo matemático con ecuaciones diferenciales lineales, con el cual se diferenciaba las mediciones actuales y las predichas. La ganancia de Kalman es el factor de escala el cual es un indicador de la precisión de la medición que se realizó con la generada por el modelo, en donde, si el factor de escala es bajo quiere decir que existe baja confianza en la medición, si el factor

de escala es alto significa que existe alta prioridad en la medición al estimar los datos, por lo tanto, el factor de escala es calculado todo el tiempo que se realice la medición. Todo el filtro se resume en un método de predicción y corrección el cual se realiza para todos los datos medidos asumiendo que el sistema sea lineal. Se usa este método para calcular la orientación de los datos obtenidos de los sensores inerciales en tiempo real y filtrar el ruido de las mediciones y del sistema. (Villalobos, 2013).

El filtro complementario se usa para dar una estimación precisa de la orientación del sensor, para lo cual combina los datos obtenidos por el giroscopio con la del acelerómetro y magnetómetro, los valores del giroscopio tienden a ir a la deriva con el tiempo ya que solo perciben la velocidad angular generada por los movimientos, por lo cual solo son confiables a corto plazo, mientras que el acelerómetro y magnetómetro son confiables a largo plazo ya que son precisos en posiciones estáticas y no a corto plazo por el ruido que se genera por las altas frecuencias iniciales al realizarse movimientos. Por lo tanto, el filtro complementario se encarga de compensar la deriva del giroscopio a largo plazo y compensar el ruido del acelerómetro y magnetómetro a corto plazo. (Treffers & Wietmarschen, 2016).

El método del filtro de Kalman que se aplica para obtener la orientación requiere de complejos cálculos y al ser un filtro recursivo, implica una alta complejidad computacional y más aún si requiere implementar en un microcontrolador, además que lo haría lento si se requieren realizar más procesos o si se trabaja con más sensores inerciales. Por otro lado, el método del filtro complementario es computacionalmente más sencillo de implementar y tiene una respuesta mucho más rápida, ya que no se realizan

cálculos muy complejos, los dos filtros tienden a dar respuestas similares, aunque con distintas ventajas, hacen que el filtro complementario se destaque gracias a su fácil implementación computacional y velocidad de respuesta. (McCarron, 2013).

Se usará el filtro complementario de acuerdo a lo detallado anteriormente, por lo tanto, se procederá a realizar el cálculo del filtro pasa altos para las medidas del giroscopio y un filtro pasa bajo para las medidas del acelerómetro. Primero se debe calcular la orientación (aRoll y aPitch) con respecto a la gravedad en base a los valores medidos por el acelerómetro en los 3 planos, los cuales son: a_x, a_y, a_z, tal como se indica en las ecuaciones (2) y (3).

$$\boldsymbol{aRoll = arctan\left(\frac{a_y}{\sqrt{a_x^2 + a_z^2}}\right)} \quad (2)$$

$$\boldsymbol{aPitch = arctan\left(\frac{-a_x}{\sqrt{a_y^2 + a_z^2}}\right)} \quad (3)$$

Para calcular la orientación en base a Roll y Pitch, se requiere que los valores de orientación con respecto a la gravedad $(aRoll, aPitch)$, que se calcularon con las ecuaciones (2) y (3) se encuentren en radianes para proceder a formar los filtros, junto con los valores del giroscopio y determinar el tiempo que transcurre entre cada iteración (dT) . Para dar mayor estabilidad al filtro se tiene el parámetro del filtro (α), este parámetro se debe encontrar entre valores de 0 y 1, las ecuaciones que combinan al filtro complementario se presentan en las ecuaciones (4) y (5).

$$\boldsymbol{Roll = \alpha * (velocidad\_angular_x * dT + Roll) + (1 - \alpha) * aRoll} \quad (4)$$

$$\boldsymbol{Pitch = \alpha * \left(velocidad\_angular_y * \ dT + Pitch\right) + (1 - \alpha) * aPitch} \quad (5)$$

En donde los valores medidos por el giroscopio son $velocidad\_angular_x$ y $velocidad\_angular_y$, las medidas del magnetómetro (m_x, m_y, m_z) se usan exclusivamente para calcular el valor de orientación Yaw, ya que los valores del acelerómetro no varían si se realiza una rotación alrededor del eje Z, al cual se la asigna la misma dirección que tiene la gravedad. La dirección que tiene el campo magnético terrestre al ser perpendicular a la dirección de la gravedad, permite calcular la rotación que se realiza en el plano horizontal, si el sensor realiza alguna inclinación se debe realizar una corrección a las medidas del magnetómetro para calcular la rotación en Yaw. Las ecuaciones para calcular Yaw son las ecuaciones (6), (7) y (8).

$$\boldsymbol{mag_x = m_z * \sin(Roll) - m_y * \cos(Roll)} \quad (6)$$

$$\boldsymbol{mag_y = m_x * \cos(Pitch) + m_y * \sin(Pitch) + \sin(Roll) + m_z} \\ \boldsymbol{* \sin(Pitch) * \cos(Roll)} \quad (7)$$

$$\boldsymbol{Yaw = arctan\left(\frac{mag_x}{mag_y}\right)} \quad (8)$$

2.4. Robot Humanoide NAO H25

El robot humanoide NAO en su actual versión V6, fue desarrollado por la compañía francesa desde el año 2014 por Aldebaran Robotics subsidiaria del grupo SoftBank, cuenta con un total de 25 grados de libertad (DOF), de los cuales 11 DOF corresponden

a las 2 piernas y su pelvis, los otros 14 DOF corresponden a los 2 brazos, cabeza y tronco. Al tener una estructura similar a la de los humanos y ser autónomo y programable, se lo usa en diferentes áreas tales como: robótica social, teleoperación e investigación sobre robots bípedos.

***Figura 13*.** Robot NAO H25.

2.4.1. Especificaciones técnicas del robot humanoide NAO H25

El robot humanoide NAO pesa aproximadamente 5.4 Kg con una altura de 57.4 cm, cuenta con un CPU ATOM Z530 que trabaja a una frecuencia de 1.6 GHz, una memoria RAM de 1 GB, 2 GB de memoria Flash y 8 GB de memoria Micro SDHC, su sistema operativo es NAOqi basado en Linux, los lenguajes de programación compatibles son: C++, Python, Java, MATLAB, entre otros, de los cuales el más soportado es Python. Dispone de una batería de ion de litio de 21.6 Voltios y 2.25 Amperios, el cual le da una autonomía de entre 60 y 90 minutos de uso continuo. Conectividad Ethernet con un

puerto RJ45 ubicado detrás de la cabeza y conectividad inalámbrica Wifi IEEE 802.11 a/b/g/n.

El robot humanoide NAO está formado por un conjunto de actuadores y sensores los cuales le permiten desenvolverse e interactuar con el medio, los cuales se detallan en la Tabla 2 y su respectiva ubicación se presentan en la Figura 14.

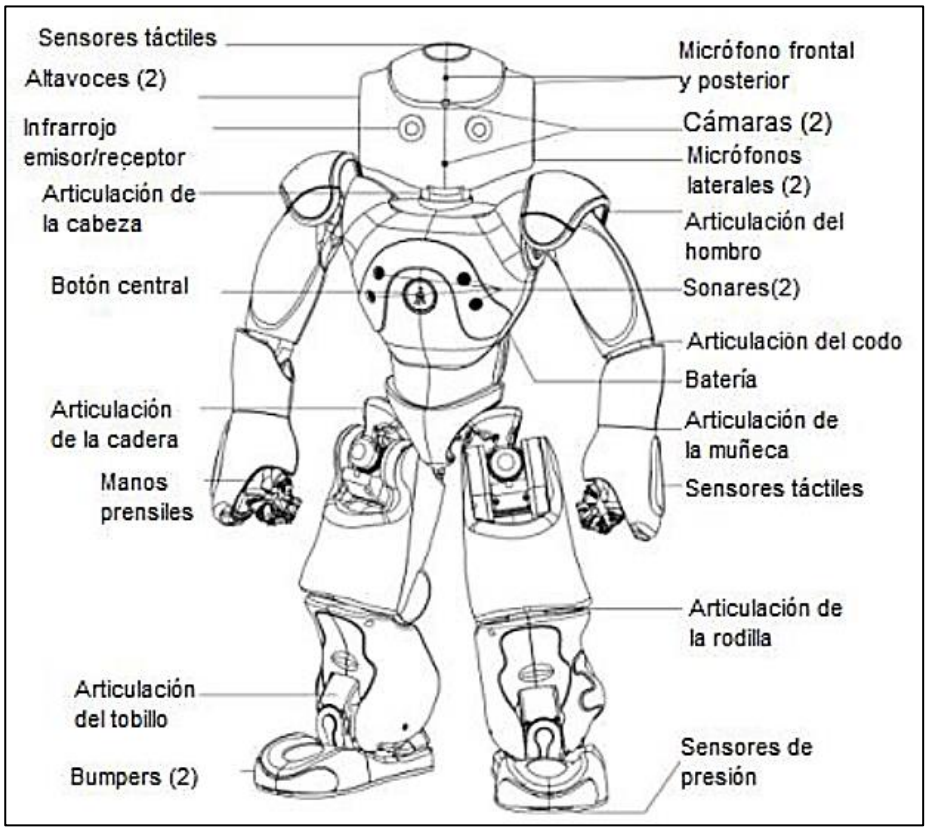


Figura 14. Partes del robot humanoide NAO.
Fuente: **(Enríquez Rodríguez, 2019)**

Tabla 2

Cantidad y elementos del robot humanoide NAO.

Cantidad	Elemento
2	Cámaras KBGA
4	Micrófonos
2	Altavoces
1	Sintetizador de voz
8	Sensores de presión
9	Sensores táctiles
2	Sensores ultrasónicos
1	Acelerómetro
1	Giroscopio
53	Leds RGB
36	Encoders magnéticos Rotatorios
25	Motores

Fuente: **(Aldebaran Robotics, 2018)**

El robot NAO cuenta con 25 DOF, los cuales son distribuidos en cadenas cinemáticas, las cadenas cinemáticas tienen como efectores, las manos, los pies y una de las dos cámaras del robot, dando un total de 5 cadenas cinemáticas.

Cada grado de libertad y su respectiva orientación se encuentra detallado en la (Figura 15), en donde se puede ver su orientación en Roll, Pitch y Yaw, la postura de referencia o cero del robot se realiza con las piernas rectas, los brazos y cabeza apuntando hacia adelante, tal como se indica en la (Figura 15). En la (Tabla 3) se presenta el rango de movilidad de cada articulación.

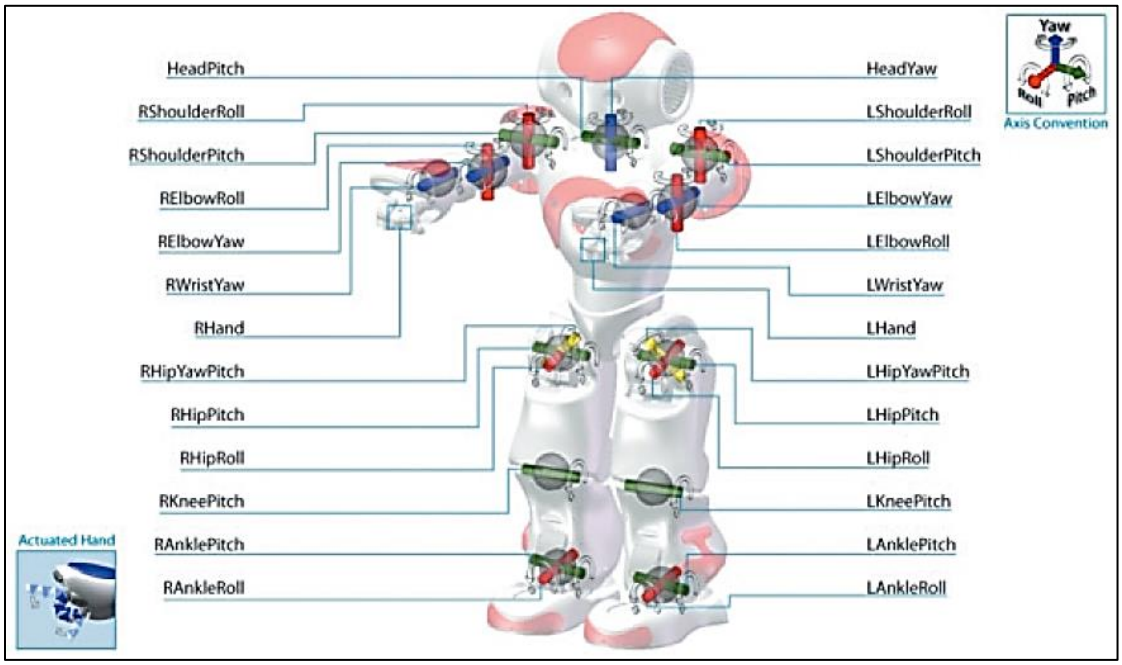


Figura 15. Vista de las articulaciones y su orientación en el robot NAO
Fuente: **(Aldebaran Robotics, 2018)**

Tabla 3
Rango de movilidad de cada articulación.

Ítem	Cadena Cinemática	Articulación	Rango (Deg)	Rango (Rad)
1	Cabeza	HeadYaw	-119.5 a 119.5	-2.0857 a 2.0857
		HeadPitch	-38.5 a 29.5	-0.6720 a 0.5149
2	Brazo Izquierdo	LShoulderPitch	-119.5 a 119.5	-2.0857 a 2.0857
		LShoulderRoll	-18 a 76	-0.3142 a 1.3265
		LElbowYaw	-119.5 a 119.5	-2.0857 a 2.0857
		LElbowRoll	-88.5 a -2	-1.5446 a -0.0349
		LWristYaw	-104.5 a 104.5	-1.8238 a 1.8238
3	Brazo Derecho	RShoulderPitch	-119.5 a 119.5	-2.0857 a 2.0857
		RShoulderRoll	-76 a 18	-1.3265 a 0.3142
		RElbowYaw	-119.5 a 119.5	-2.0857 a 2.0857
		RElbowRoll	2 a 88.5	0.0349 a 1.5446
		RWristYaw	-104.5 a 104.5	-1.8238 a 1.8238
4	Pierna Izquierda	LHipYawPitch	-65.62 a 42.44	-1.145303 a 0.740810
		LHipRoll	-21.74 a 45.29	-0.379472 a 0.790477
		LHipPitch	-88.00 a 27.73	-1.535889 a 0.484090
		LKneePitch	-5.29 a 121.04	-0.092346 a 2.112528

CONTINÚA

		LAnklePitch	-68.15 a 52.86	-1.189516 a 0.922747
		LAnkleRoll	-22.79 a 44.06	-0.397880 a 0.769001
5	Pierna Derecha	RHipYawPitch*	-65.62 a 42.44	-1.145303 a 0.740810
		RHipRoll	-45.29 a 21.74	-0.790477 a 0.379472
		RHipPitch	-88.00 a 27.73	-1.535889 a 0.484090
		RKneePitch	-5.90 a 121.47	-0.103083 a 2.120198
		RAnklePitch	-67.97 a 53.40	-1.186448 a 0.932056
		RAnkleRoll	-44.06 a 22.80	-0.768992 a 0.397935

Fuente: **(Aldebaran Robotics, 2018)**

Los enlaces entre las articulaciones y los eslabones del robot humanoide NAO se presentan en las siguientes figuras, en la (Tabla 4) se detalla las medidas de los eslabones que hay entre articulaciones.

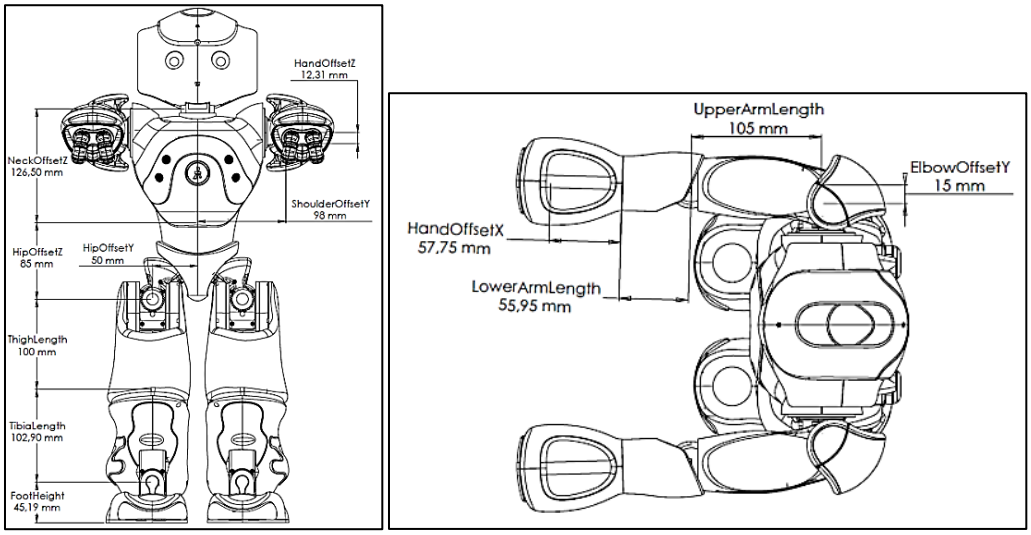


Figura 16. Vista Frontal (Izq.) y vista superior (Der.) del robot Nao V5.
Fuente: **(Aldebaran Robotics, 2018)**

Tabla 4
Distancias entre eslabones y articulaciones

Cadena Cinemática	Desde	hasta	X (mm)	Y (mm)	Z (mm)
Cabeza	Torso	HeadYaw	0,00	0,00	126,50
	HeadYaw	HeadPitch	0,00	0,00	0,00
Brazos	Torso	LShoulderPitch	0.00	98.00	100.00

CONTINÚA

	LShoulderPitch	LShoulderRoll	0.00	0.00	0.00
	LShoulderRoll	LElbowYaw	105.00	15.00	0.00
	LElbowYaw	LElbowRoll	0.00	0.00	0.00
	LElbowRoll	LWristYaw	55.95	0.00	0.00
Piernas	Torso	LHipYawPitch	0.00	50.00	-85.00
	LHipYawPitch	LHipRoll	0.00	0.00	0.00
	LHipRoll	LHipPitch	0.00	0.00	0.00
	LHipPitch	LKneePitch	0.00	0.00	-100.00
	LKneePitch	LAnklePitch	0.00	0.00	-102.90
	LAnklePitch	LAnkleRoll	0.00	0.00	0.00

Fuente: **(Aldebaran Robotics, 2018)**

Tabla 5

Distancias de las longitudes principales

Elemento	Medida (mm)
NeckOffsetZ	126.50
ShoulderOffsetY	98.00
ElbowOffsetY	15.00
UpperArmLength	105.00
LowerArmLength	55.95
ShoulderOffsetZ	100.00
HandOffsetX	57.75
HandOffsetZ	12.31
HipOffsetZ	85.00
HipOffsetY	50.00
ThighLength	100.00
TibiaLength	102.90
FootHeight	45.19
TopCameraX	53.9
TopCameraZ	67.9
BottonCameraX	48.8
BottonCameraZ	23.8

Fuente: **(Aldebaran Robotics, 2018)**

La posición del centro de masa y masa de cada objeto solido (S) del robot, se describen con respecto a su propio sistema de coordenadas (o, R), en la (Tabla 6) se

presentan dichos valores tomando en cuenta que fueron medidos con la postura cero del robot.

Tabla 6

Valores de posición y masa de cada objeto sólido del robot NAO

Parte	Objeto	Ubicación	Masa	X	Y	Z
Torso	Torso	Torso	1,0496	-0,00413	0	0,04342
Cabeza	Neck	HeadYaw	0,07842	-0,00001	0	-0,02742
	head	HeadPitch	0,60533	-0,00112	0	0,0528
Brazos	Right Shoulder	RShoulderPitch	0,09304	-0,00165	0,02663	0,00014
	Left Shoulder	LShoulderPitch	0,09304	-0,00165	-0,02663	0,00014
	Right Biceps	RShoulderRoll	0,15777	0,02455	-0,00563	0,0033
	Left Biceps	LShoulderRoll	0,15777	0,02455	0,00563	0,0033
	Right Elbow	RElbowYaw	0,06483	-0,02744	0	-0,00014
	Left Elbow	LElbowYaw	0,06483	-0,02744	0	-0,00014
	Right ForeArm	RElbowRoll	0,07761	0,02556	-0,00281	0,00076
	Left ForeArm	LEbowRoll	0,07761	0,02556	0,00281	0,00076
	Right Hand	RWristYaw	0,18533	0,03434	0,00088	0,00308
	Left Hand	LWristYaw	0,18533	0,03434	-0,00088	0,00308
Piernas	Right Pelvis	RHipYawPitch	0,06981	-0,00781	0,01114	0,02661
	Left Pelvis	LhipYawPitch	0,06981	-0,00781	-0,01114	0,02661
	Right Hip	RHipRoll	0,14053	-0,01549	-0,00029	-0,00515
	Left Hip	LHipRoll	0,14053	-0,01549	0,00029	-0,00515
	Right Thigh	RHipPitch	0,38968	0,00138	-0,00221	-0,05373
	Left Thigh	LHipPitch	0,38968	0,00138	0,00221	-0,05373
	Right Tibia	RKneePitch	0,30142	0,00453	-0,00225	-0,04936
	LeftTibia	LKneePitch	0,30142	0,00453	0,00225	-0,04936
	Right Ankle	RAnklePitch	0,13416	0,00045	-0,00029	0,00685
	Left Ankle	LAnklePitch	0,13416	0,00045	0,00029	0,00685
	Right Foot	RAnkleRoll	0,17184	0,02542	-0,0033	-0,03239
	Left Foot	LAnkleRoll	0,17184	0,02542	0,0033	-0,03239

Fuente: **(Aldebaran Robotics, 2018)**

2.4.2. Software del Robot Humanoide NAO H25

- **NAOqi:** Es un Framework el cual es el sistema operativo que ejecuta y controla al robot, al cual se lo puede programar para controlar e interactuar con el mismo,

proporciona las funciones de paralelismo, gestión de recursos, sincronización, control de eventos, permite establecer una comunicación homogénea con todos los módulos de robot (audio, video, movimiento, memoria). NAOqi es un sistema multiplataforma y un software programable con los lenguajes de programación C++ y Python.

El Framework NAOqi se compone de: Un Broker el cual es el archivo ejecutable de NAOqi, librerías las cuales contienen uno o más módulos y que a su vez están conformados por métodos. Lkos módulos forman un árbol de métodos adjuntos los cuales son gestionados por el Broker, con la finalidad de que los módulos puedan encontrar métodos específicos. Su estructura se presenta la (Figura 17).

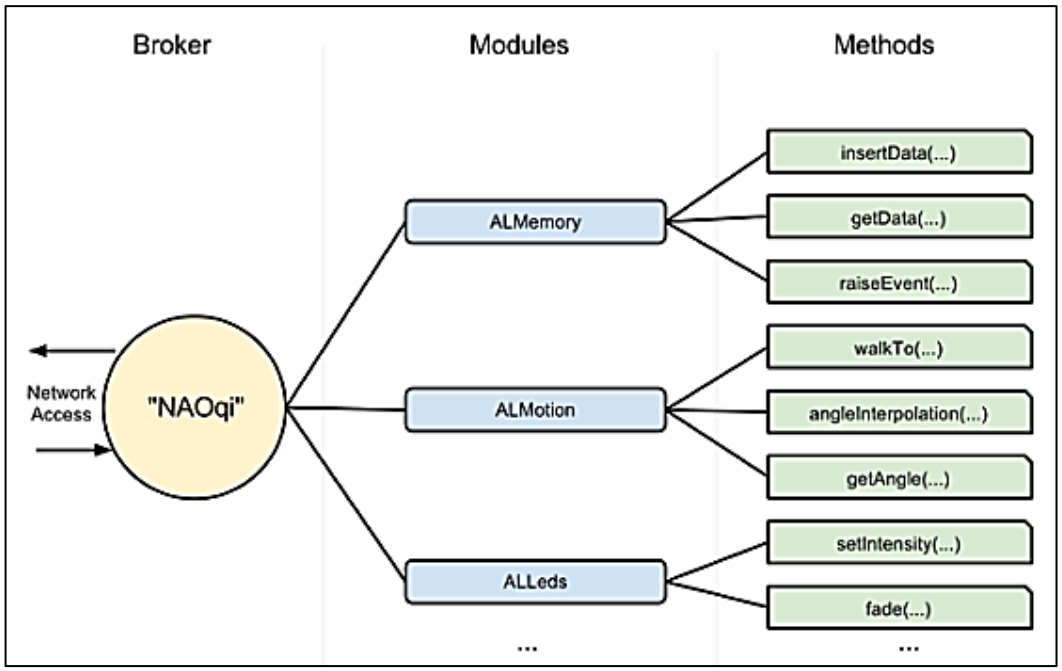


Figura 17. Estructura interna del Framework NAOqi
Fuente: **(Aldebaran Robotics, 2018)**

- **Python SDK:** El robot NAO cuenta con un software integrado que le permite trabajar de forma autónoma y de un software de escritorio el cual permite que el robot sea

controlado remotamente. Para controlarlo remotamente se puede usar el software desarrollado por el fabricante Choreographe o alguno de los SDK's disponibles los cuales fueron desarrollados para lenguajes de programación específicos, la forma del control del robot NAO de forma remota se muestra en la (Figura 18).

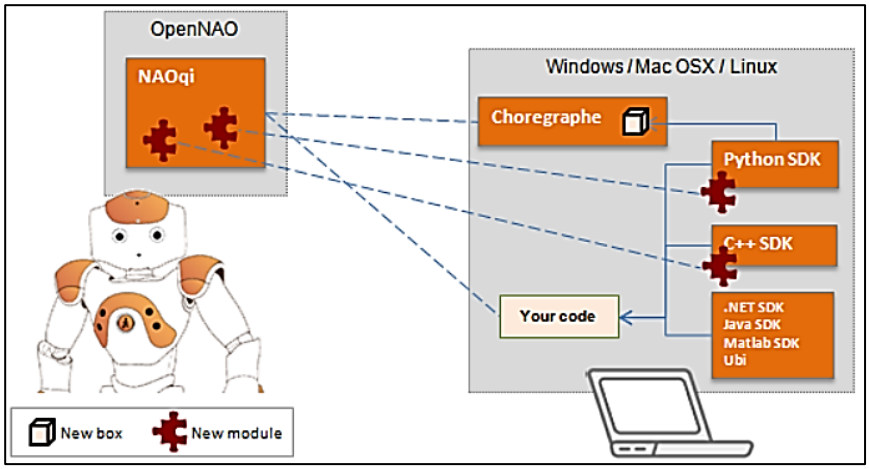


Figura 18. Estructura del control remoto del robot NAO
Fuente: **(Aldebaran Robotics, 2018)**

Se puede programar a la API NAOqi en 8 lenguajes de programación, de los cuales solo se la puede acceder de forma completa con los lenguajes C++ y Python, que permiten acceder a todas las funciones del robot de forma remota.

Python SDK es una API multiplataforma en su versión compatible con Python 2.7 – 32 bits, La API de Python al ser las más desarrollada para el robot NAO, le permite acceder a todo el control del robot desde una máquina remota o desde el mismo robot, para lo cual se usan los siguientes pasos:

- Importar ALProxy de NAOqi
- Crear un objeto ALProxy para los módulos que se usen

- Llamar a los métodos de los módulos llamados.

2.4.3. Reconocimiento de movimientos Humanos

La cinemática del humano es un sistema que puede definirse por la posición y orientación de sus articulaciones y sus efectores finales, tales como: las manos, pies y cabeza. Para la realización del modelado se considera que el cuerpo humano es un cuerpo rígido ya que los enlaces entre articulaciones no se deforman, se simplifica el cuerpo humano con un modelo formado por 10 enlaces o extremidades, los cuales son: la cabeza y el cuello, tronco, brazos, antebrazos, muslos y piernas. Para este modelo simplificado no se consideran los enlaces más pequeños, ya que no influyen en el movimiento en conjunto del humano, los cuales son los dedos de las manos y pies, el modelo se presenta en la (Figura 19). (Davidrajuh, 2015).

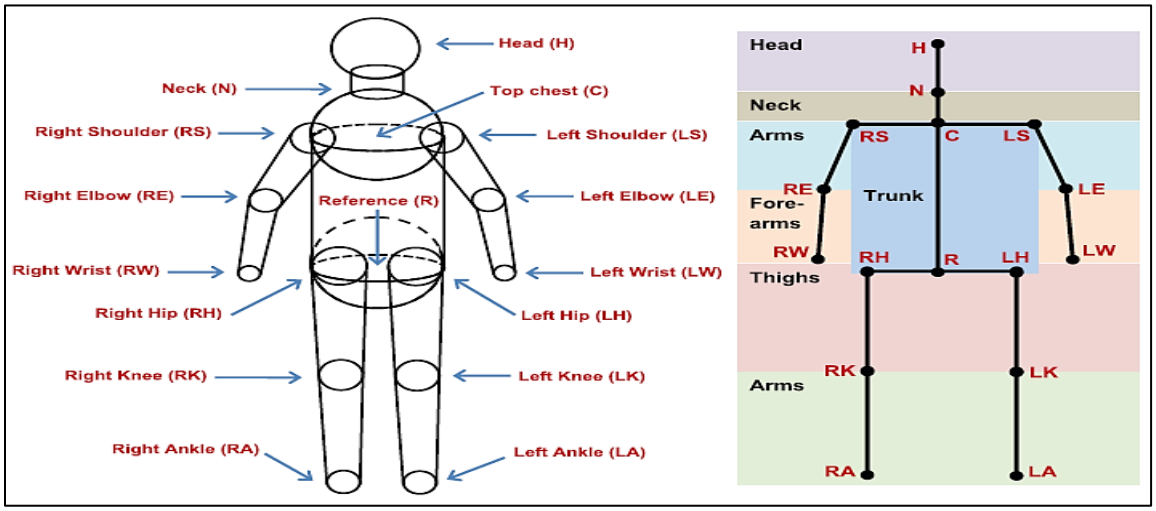


Figura 19. Modelado de cuerpo (Izq) esqueleto con enlaces y articulaciones (Der)
Fuente: **(Ronningen, Panggabean, & Overby, 2013)**

Como se puede apreciar en la (Figura 19) (Derecha), el modelo cuenta con 16 articulaciones, los cuales tienen restricciones angulares de movimiento, los cuales

dependen del rango de movilidad del enlace humano móvil al que se encuentre conectado. Los movimientos que puede realizar un humano se presentan en la (Figura 20).

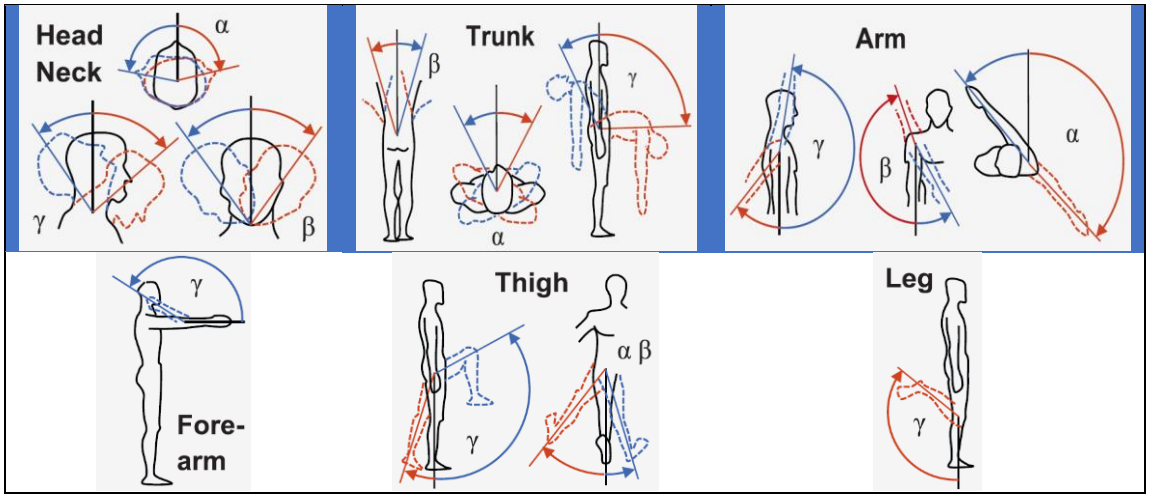


***Figura 20*.** Rango de movilidad angular humano de cada articulación
Fuente: **(Ronningen, Panggabean, & Overby, 2013)**

Como se puede observar en la (Figura *20)*, los ángulos de medida de cada movimiento se describen con los ángulos (α, β, γ), los ángulos son medidos contra la vertical con valores positivos, mientras que los valores negativos son para los ángulos con la dirección contraria, en la (Tabla 7) se presentan los rangos en los que varian los ángulos de cada articulación tomando en cuenta los movimientos descritos en la (Figura 20).

Tabla 7
Rango de movilidad angular de cada enlace del cuerpo humano

		Rango de movilidad angular		
Enlaces	Articulaciones	α (°)	β (°)	γ (°)
Cabeza y cuello	H	[-70,70]	[-63,63]	[-60,30]
Torso	N, C, LS, RS, LH, RH y R	[-30, 30]	[-40, 40]	[-90, 30]
Brazo izquierdo	LE	[-30, 130]	[-40, 170]	[-40, 170]
Brazo derecho	RE	[-130, 30]	[-170, 40]	[-40, 170]

CONTINÚA

Antebrazos	LR/RW	[0,0]	[0,0]	[0,150]
Muslo izquierda	LK	[-50, 45]	[-30, 45]	[-15, 90]
Muslo derecha	RK	[-45, 50]	[-45, 30]	[-15, 90]
Piernas derecha e izquierda	LA, RA	[0, 0]	[0, 0]	[-145, 0]

Fuente: (Ronningen, Panggabean, & Overby, 2013)

2.4.4. Cinemática del robot humanoide NAO H25

La cinemática de un robot se centra en estudiar los movimientos de su efector final con respecto a la orientación y posición de cada cadena cinemática, en robots con varios grados de libertad. La realización de la cinemática de un robot describe analíticamente sus movimientos espaciales con respecto a un sistema de referencia y en función del tiempo. La cinemática se divide en dos métodos los cuales son: Cinemática Directa que permite determina la posición y orientación del efector final de un robot con respecto a un sistema de referencia conociendo los valores angulares de cada articulación y dimensiones geométricas de los eslabones del robot. La Cinemática Inversa determina los valores angulares de cada articulación sabiendo la orientación y posición del efector final.

- **Cinemática Directa:** La Cinemática Directa realiza una traslación desde un sistema articular a un sistema cartesiano tridimensional de una cadena cinemática, con m articulaciones y sus respectivos valores articulares (θ_1, θ_2 ... θ_m), permite calcular la posición y orientación de su efector final en coordenadas cartesianas (P_x, P_y, P_z), siempre y cuando se produzca una solución analítica exacta.

El análisis cinemático se lo puede realizar con diferentes métodos, en la presente tesis se lo realizará aplicando la metodología de los parámetros de Denavit-Hartemberg (DH) y el uso de matrices de transformación, ya que permiten que cada matriz solo dependa

de los 4 parámetros de DH para definir el comportamiento del robot. (Zavala, Guzmán, & Galván, 2013)

La matriz de transformación describe la posición y orientación relativa entre los sistemas correspondientes a dos eslabones consecutivos del robot, con la finalidad de obtener la posición y orientación de un sistema de coordenadas del primer eslabón con respecto al sistema de coordenadas del segundo eslabón o en su defecto de la base.

La metodología de DH se basa en una matriz de transformación que establece un sistema de coordenadas sistemático para cada eslabón i de una cadena cinemática, permitiendo determinar ecuaciones cinemáticas de la cadena completa. Los parámetros DH son $(a_i, \theta_i, d_i, \alpha_i)$ de cada eslabón i, que permiten describir las matrices de transformación T de cada eslabón del robot logrando así relacionar todos los sistemas de coordenadas. A su vez la matriz de transformación consiste en 4 transformaciones básicas de rotaciones y traslaciones que relacionan el sistema de referencia del eslabón i con el sistema de referencia del eslabón $i-1$, las transformaciones son las siguientes:

- Rotación alrededor del eje x_i un ángulo α_i.

$$\boldsymbol{R_{x_i}(\alpha_i)} = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{cos}(\boldsymbol{\alpha_i}) & \boldsymbol{-sin}(\boldsymbol{\alpha_i}) & \mathbf{0} \\ \mathbf{0} & \boldsymbol{sin}(\boldsymbol{\alpha_i}) & \boldsymbol{cos}(\boldsymbol{\alpha_i}) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix} \quad (9)$$

- Traslación a lo largo de x_i una distancia a_i; dado el vector $a_i\,(0, 0, a_i)$.

$$\boldsymbol{A_{x_i}\,(a_i)} = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} & \boldsymbol{a_i} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix} \quad (10)$$

- Rotación alrededor del eje z_{i-1} con un ángulo θ_i.

$$\boldsymbol{R_{z_{i-1}}\,(\theta_i)} = \begin{bmatrix} \boldsymbol{\cos(\theta_i)} & \boldsymbol{-sin(\theta_i)} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{sin(\theta_i)} & \boldsymbol{\cos(\theta_i)} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{1} \end{bmatrix} \quad (11)$$

- Traslación a lo largo de z_{i-1} a una distancia d_i; dado el vector $d_i\,(0, 0, d_i)$.

$$\boldsymbol{A_{z_{i-1}}\,(d_i)} = \begin{bmatrix} \boldsymbol{1} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{1} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{1} & \boldsymbol{d_i} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{1} \end{bmatrix} \quad (12)$$

Tomando en cuenta las anteriores matrices y que su producto genera la matriz de transformación, se presenta la siguiente ecuación que describe la matriz de transformación T_i^{i-1}:

$$\boldsymbol{T_i^{i-1} = R_{x_i}(\alpha_i) * A_{x_i}\,(a_i) * R_{z_{i-1}}\,(\theta_i) * A_{z_{i-1}}\,(d_i)} \quad (13)$$

Los parámetros DH que describen cada eslabón se presentan a continuación:

- a: Longitud de la normal común.
- α: Ángulo sobre la normal común, desde el eje z_{i-1} hasta el eje z_i.
- d: Offset establecido a lo largo del eje z_{i-1} hasta la normal común.
- θ: Ángulo sobre el eje z_{i-1}, desde el eje x_{i-1} hasta el eje x_{i-1}.

Finalmente se realiza el traspaso de un marco de referencia base de algún eslabón al sistema de referencia de la matriz de transformación T_{DH} (Kofinas, 2012), tal como se indica en la siguiente ecuación:

$$\boldsymbol{T_{DH} = R_x(\alpha) * A_x(a) * R_z(\theta) * A_z(d)} \quad (14)$$

$$\boldsymbol{T_{DH} = \begin{bmatrix} \cos(\theta) & -sin(\theta) & 0 & 0 \\ sin(\theta) * \cos(\alpha) & \cos(\theta) * \cos(\alpha) & -\sin(\alpha) & -d * \sin(\alpha) \\ sin(\theta) * sin(\alpha) & \cos(\theta) * \sin(\alpha) & \cos(\alpha) & d * \cos(\alpha) \\ 0 & 0 & 0 & 1 \end{bmatrix}} \quad (15)$$

Las 25 articulaciones del robot NAO se dividen en 5 cadenas cinemáticas, con sus respectivos efectores finales, teniendo como referencia el torso del robot, tal como se presentan en la (Figura *21)*, en la Tabla 8 se presenta la distribución de articulaciones de acuerdo a las cadenas cinemáticas.

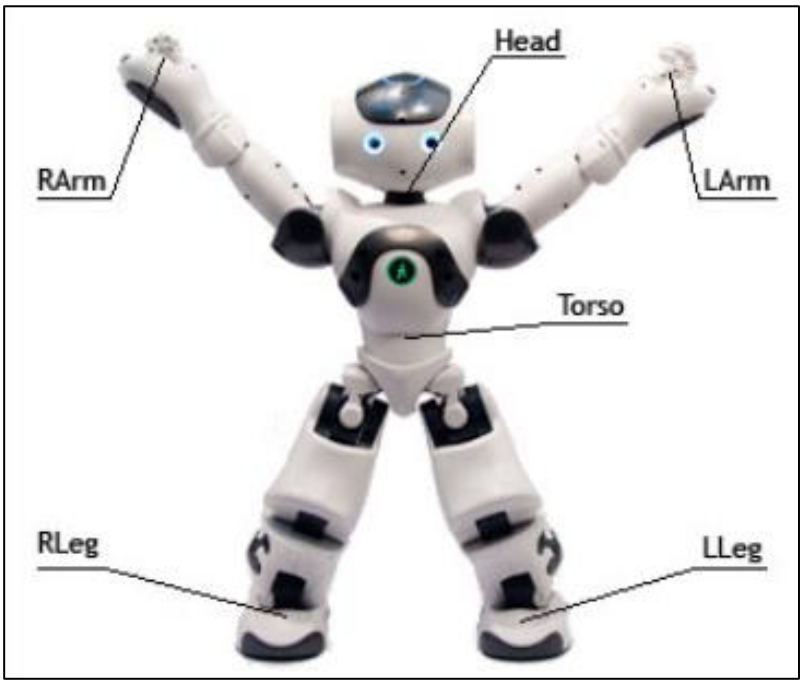


Figura 21. Efectores finales del robot NAO
Fuente: **(Aldebaran Robotics, 2018)**

Tabla 8
Cadenas cinemáticas del robot NAO

Cadenas Cinemáticas	Cabeza	Brazo izquierdo	Brazo derecho	Pierna izquierda	Pierna derecha
Articulaciones	HeadYaw HeadPitch	LShoulderPitch LShoulderRoll LElbowYaw LElbowRoll LWristYaw	RShoulderPitch RShoulderRoll RElbowYaw RElbowRoll RWristYaw	LHipYawPitch LHipRoll LHipPitch LKneePitch LAnklePitch LAnkleRoll	RHipYawPitch RHipRoll RHipPitch RKneePitch RAnklePitch RAnkleRoll
Efector Final	Cámara superior o Cámara inferior	LArm	RArm	LLeg	RLeg

Al realizarse la división en 5 cadenas cinemáticas, resulta más sencillo su análisis el determinar los parámetros DH y a su vez las matrices de transformación, sin embargo, el sistema de coordenadas de referencia para cada articulación se encontrará en las primeras articulaciones de cada cadena cinemática, por lo tanto luego de encontrar los parámetros DH de cada cadena cinemática se realizará una traslación de cada sistema de coordenadas al sistema de coordenadas base que se encuentra en el torso, para la obtención de los parámetros DH se requirieron los datos de las (Tablas 4, 5 y 6.)

- Cinemática directa de la cadena cinemática de la cabeza.

Tabla 9
Parámetros DH de la cadena cinemática de la cabeza

	Parámetros DH			
Articulaciones	a	α	d	θ
HeadYaw	0	0	0	θ_1
HeadPitch	0	$-\frac{\pi}{2}$	0	$\theta_2 - \frac{\pi}{2}$

Fuente: (Kofinas, 2012)

Se requiere trasladar el sistema de referencia de la cadena cinemática de la cabeza hasta el sistema de coordenadas del torso, para lo cual se usa la siguiente matriz de traslación:

$$\boldsymbol{A^{0}_{Base} = A(0, 0, NeckOffsetZ)} \quad (16)$$

La cadena cinemática de la cabeza solo se contempla hasta la articulación HeadPitch, y como el efector final de la cabeza es la cámara superior, se requerirá trasladar su posición al sistema de coordenadas de la cadena, esto se logra con las siguientes matrices de transformación:

$$\boldsymbol{A^{2}_{End} = A(TopCameraX, 0, TopCameraZ)} \quad (17)$$

$$\boldsymbol{R_x\left(\frac{\pi}{2}\right) * R_y\left(\frac{\pi}{2}\right) * A^{2}_{End}} \quad \textbf{(18)}$$

Finalmente, la matriz de transformación total es:

$$\boldsymbol{T^{Base}_{End} = A^{0}_{Base} * T^{1}_{0} * T^{2}_{1} * R_x\left(\frac{\pi}{2}\right) * R_y\left(\frac{\pi}{2}\right) * A^{2}_{End}} \quad (19)$$

- Cinemática directa de la cadena cinemática del brazo izquierdo

Tabla 10

Parámetros DH de la cadena cinemática del brazo izquierdo

	Parámetros DH			
Articulaciones	a	α	d	θ
LShoulderPitch	0	$-\frac{\pi}{2}$	0	θ_1
LShoulderRoll	0	$\frac{\pi}{2}$	0	$\theta_2 - \frac{\pi}{2}$
LElbowYaw	0	$-\frac{\pi}{2}$	$UpperArmLength$	θ_3

CONTINÚA

LElbowRoll	0	$\frac{\pi}{2}$	0	θ_4
LWristYaw	0	$-\frac{\pi}{2}$	$LowerArmLength$	θ_5

Fuente: **(Adi, Setijadi, & Syaichu, 2014)**

Se requiere trasladar el sistema de referencia de la cadena cinemática del brazo izquierdo hasta el sistema de coordenadas del torso, para lo cual se usa la siguiente matriz de traslación:

$$\boldsymbol{A^0_{Base} = A(0, ShoulderOffsetY + ElbowOffsetY, ShoulderOffsetZ)} \quad (20)$$

La cadena cinemática del brazo izquierdo solo se contempla hasta la articulación LWristYaw, y como su efector final es la mano izquierda, se requerirá trasladar su posición al sistema de coordenadas de la cadena, esto se logra con las siguientes matrices de transformación:

$$\boldsymbol{A^2_{End} = A(HandOffsetX, 0, -HandOffsetZ)} \quad (21)$$

$$\boldsymbol{R_x\left(\frac{\pi}{2}\right) * R_z\left(\frac{\pi}{2}\right) * A^2_{End}} \quad (22)$$

Finalmente, la matriz de transformación total es:

$$\boldsymbol{T^{Base}_{End} = A^0_{Base} * T^1_0 * T^2_1 * T^3_2 * T^4_3 * T^5_4 * R_x\left(\frac{\pi}{2}\right) * R_Z\left(\frac{\pi}{2}\right) * A^2_{End}} \quad (23)$$

- Cinemática directa de la cadena cinemática del brazo derecho

Tabla 11

Parámetros DH de la cadena cinemática del brazo derecho

	Parámetros DH			
Articulaciones	a	α	d	θ
LShoulderPitch	0	$\frac{\pi}{2}$	0	θ_1
LShoulderRoll	0	$-\frac{\pi}{2}$	0	$\theta_2 + \frac{\pi}{2}$
LElbowYaw	0	$\frac{\pi}{2}$	$UpperArmLength$	θ_3
LElbowRoll	0	$-\frac{\pi}{2}$	0	θ_4
LWristYaw	0	$\frac{\pi}{2}$	$LowerArmLength$	θ_5

Fuente: **(Adi, Setijadi, & Syaichu, 2014)**

Se requiere trasladar el sistema de referencia de la cadena cinemática del brazo derecho hasta el sistema de coordenadas del torso, para lo cual se usa la siguiente matriz de traslación:

$$\boldsymbol{A^0_{Base} = A(0, -ShoulderOffsetY - ElbowOffsetY, ShoulderOffsetZ)} \quad (24)$$

La cadena cinemática del brazo derecho solo se contempla hasta la articulación RWristYaw, y como su efector final es la mano derecha, se requerirá trasladar su posición al sistema de coordenadas de la cadena, esto se logra con las siguientes matrices de transformación:

$$\boldsymbol{A^2_{End} = A(HandOffsetX, 0, -HandOffsetZ)} \quad (25)$$

$$\boldsymbol{R_x\left(-\frac{\pi}{2}\right) * R_z\left(-\frac{\pi}{2}\right) * A^2_{End}} \quad (26)$$

Finalmente, la matriz de transformación total es:

$$T_{End}^{Base} = A_{Base}^{0} * T_{0}^{1} * T_{1}^{2} * T_{2}^{3} * T_{3}^{4} * T_{4}^{5} * R_x\left(-\frac{\pi}{2}\right) * R_Z\left(-\frac{\pi}{2}\right) * A_{End}^{2} \quad (27)$$

- Cinemática directa de la cadena cinemática de la pierna izquierda

Tabla 12

Parámetros DH de la cadena cinemática de la pierna izquierda

	Parámetros DH			
Articulaciones	a	α	d	θ
LHipYawPitch	0	$-\frac{3\pi}{4}$	0	$\theta_1 - \frac{\pi}{2}$
LHipRoll	0	$-\frac{\pi}{2}$	0	$\theta_2 + \frac{\pi}{4}$
LHipPitch	0	$\frac{\pi}{2}$	0	θ_3
LKneePitch	$-ThighLength$	0	0	θ_4
LAnklePitch	$-TibiaLength$	0	0	θ_5
LAnkleRoll	0	$-\frac{\pi}{2}$	0	θ_6

Fuente: **(Kofinas, 2012)**

Se requiere trasladar el sistema de referencia de la cadena cinemática de la pierna izquierda hasta el sistema de coordenadas del torso, para lo cual se usa la siguiente matriz de traslación:

$$A_{Base}^{0} = A(0, HipOffsetY, -HipOffsetZ) \quad (28)$$

La cadena cinemática de la pierna izquierda solo se contempla hasta la articulación LAnkleRoll, y como su efector final es el pie izquierdo, se requerirá trasladar su posición al sistema de coordenadas de la cadena, esto se logra con las siguientes matrices de transformación:

$$A_{End}^{2} = A(0, 0, -FootHeight) \quad (29)$$

$$\boldsymbol{R_z(\pi) * R_y\left(-\frac{\pi}{2}\right) * A_{End}^{2}} \quad (30)$$

Finalmente, la matriz de transformación total es:

$$\boldsymbol{T_{End}^{Base} = A_{Base}^{0} * T_0^1 * T_1^2 * T_2^3 * T_3^4 * T_4^5 * T_5^6 * R_z(\pi) * R_y\left(-\frac{\pi}{2}\right) * A_{End}^{2}} \quad (31)$$

- Cinemática directa de la cadena cinemática de la pierna derecha

Tabla 13

Parámetros DH de la cadena cinemática de la pierna derecha

	Parámetros DH			
Articulaciones	a	α	d	θ
LHipYawPitch	0	$-\frac{\pi}{4}$	0	$\theta_1 - \frac{\pi}{2}$
LHipRoll	0	$-\frac{\pi}{2}$	0	$\theta_2 - \frac{\pi}{4}$
LHipPitch	0	$\frac{\pi}{2}$	0	θ_3
LKneePitch	$-ThighLength$	0	0	θ_4
LAnklePitch	$-TibiaLength$	0	0	θ_5
LAnkleRoll	0	$-\frac{\pi}{2}$	0	θ_6

Fuente: **(Kofinas, 2012)**

Se requiere trasladar el sistema de referencia de la cadena cinemática de la pierna izquierda hasta el sistema de coordenadas del torso, para lo cual se usa la siguiente matriz de traslación:

$$\boldsymbol{A_{Base}^{0} = A(0, -HipOffsetY, -HipOffsetZ)} \quad (32)$$

La cadena cinemática de la pierna izquierda solo se contempla hasta la articulación LAnkleRoll, y como su efector final es el pie izquierdo, se requerirá trasladar su posición

al sistema de coordenadas de la cadena, esto se logra con las siguientes matrices de transformación:

$$\boldsymbol{A_{End}^{2} = A(0, 0, -FootHeight)} \quad (33)$$

$$\boldsymbol{R_z(\pi) * R_y\left(-\frac{\pi}{2}\right) * A_{End}^{2}} \quad (34)$$

Finalmente, la matriz de transformación total es:

$$\boldsymbol{T_{End}^{Base} = A_{Base}^{0} * T_0^1 * T_1^2 * T_2^3 * T_3^4 * T_4^5 * T_5^6 * R_z(\pi) * R_y\left(-\frac{\pi}{2}\right) * A_{End}^{2}} \quad (35)$$

- Cinemática Inversa: La cinemática inversa determina los valores angulares que deben tener las articulaciones para aproximarse hasta un punto conociendo la posición y orientación del efector final del robot, la resolución de la cinemática inversa no es sistemática y cerrada, ya que pueden llegar a darse múltiples soluciones para una misma posición y orientación, los métodos más comunes que se usan son: método iterativo a través de los parámetros DH y matrices de transformación inversa, método geométrico que se basa en encontrar relaciones trigonométricas para los ángulos de las articulaciones, método de desacoplamiento cinemático aplicable para robot con más de 6 grados de libertad, método de algebra de tornillos, método de cuaternios, etc. (Seo, 2011). Las soluciones que se obtengan de los diferentes métodos de solución para obtener la cinemática inversa, son únicas y específicas para el robot, se pueden obtener soluciones numéricas iterativas o soluciones analíticas.

La solución que se presenta para el cálculo de la cinemática inversa de las 3 cadenas cinemáticas, se la realizó en base al estudio realizado por Kofinas, N., Orfanoudakis, E., & Lagoudakis, M. G. en la investigación: Complete analytical forward and inverse kinematics for the NAO humanoid robot, realizada en el año 2015.

- Cinemática inversa de la cadena cinemática de la cabeza: La cadena cinemática de la cabeza solo cuenta con dos articulaciones $HeadYaw\ (x_1)$ y $HeadPitch\ (x_2)$, los cuales son los valores a calcular, y recibiendo como parámetros de entrada la posición del efector de la cámara superior (P_x, P_y, P_z), y asignado los valores de $L1 = (TopCameraX)$, $L2 = (TopCameraZ)$ Y $L3 = (NeckOffsetZ)$. Los valores para $X1$ y $X2$ son:

$$\boldsymbol{x2 = asin((L2 * (L1\text{^}2 + L2\text{^}2 - L3\text{^}2 + 2 * L3 * Pz - Pz\text{^}2)\text{^}(1/2) - L1 * Pz + L1 * L3)/(L1\text{^}2 + L2\text{^}2))} \quad (36)$$

$$\boldsymbol{x1 = asin(Py/(L1 * cos(x2) + L2 * sin(x2)))} \quad (37)$$

- Cinemática inversa de la cadena cinemática del brazo izquierdo: La cadena cinemática del brazo izquierdo con 5 articulaciones $LShoulderPitch\ (x_1), LShoulderRoll(x_2), LElbowYaw(x_3), LElbowRoll\ (x_4)$ y $LWristYaw(x_5)$, los cuales son los valores a calcular, y recibiendo como parámetros de entrada la posición del efector de la mano izquierda (P_x, P_y, P_z), y asignado los valores de $L1 = ShoulderOffsetY + ElbowOffsetY$, $L2 = ShoulderOffsetZ$ Y $L3 = UpperArmLength$ y $L4 = HandOffsetX +$

$LowerArmLength$. Se puede calcular una solución cerrada usando geometría, en la articulación $LElbowRoll\ (x_4)$ que se encuentra entre dos eslabones, siendo equivalentes a un triángulo formado por los eslabones, tal como se indica en la siguiente figura:

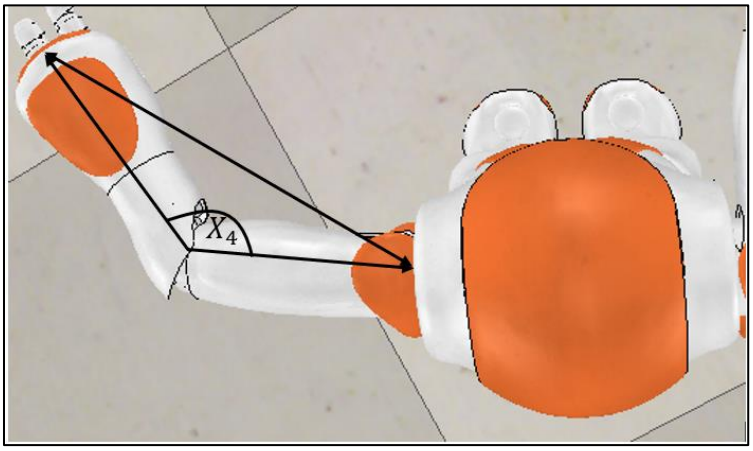


Figura 22. Triángulo formado por el ángulo $\mathbf{LElbowRoll\ (x_4)}$

Primero se calcula el valor de distancia entre dos puntos, entre la posición de la articulación $RShoulderPitch\ (P1)$ y el efector final $(P2)$, para por último aplicar una formula trigonométrica, tal como se indica a continuación:

$$\boldsymbol{d = sqrt\left(\left(P1(1) - P2(1)\right)^2 + \left(P1(2) - P2(2)\right)^2 + \left(P1(3) - P2(3)\right)^2\right)} \quad (38)$$

$$\boldsymbol{x4 = -\left(pi - acos\left(\frac{L3^2 + L4^2 - d^2}{2 * L3 * L4}\right)\right)} \quad (39)$$

El cálculo de los restantes valores articulares se lo realiza de forma analítica, tal como se presenta a continuación:

$$\boldsymbol{x1} = \boldsymbol{atan}\left(\frac{\left(\boldsymbol{sin}\left(\widehat{\boldsymbol{x1}}\right)\right) * \left(\boldsymbol{L2} * \boldsymbol{cos}\left(\widehat{\boldsymbol{x2}}\right) + \boldsymbol{L1} * \boldsymbol{sin}\left(\widehat{\boldsymbol{x2}}\right)\right)}{\left(\boldsymbol{cos}\left(\widehat{\boldsymbol{x2}}\right) * \left(\boldsymbol{L2} * \boldsymbol{cos}\left(\widehat{\boldsymbol{x2}}\right) + \boldsymbol{L1} * \boldsymbol{sin}\left(\widehat{\boldsymbol{x2}}\right)\right)\right)}\right) \quad (40)$$

$$\boldsymbol{x2} = \pm\boldsymbol{acos}\left(\frac{\boldsymbol{L2}\left(\boldsymbol{L2cos}\left(\widehat{\boldsymbol{x2}}\right) + \boldsymbol{L1sin}\left(\widehat{\boldsymbol{x2}}\right)\right) - \boldsymbol{L1}\left(-\boldsymbol{L1cos}\left(\widehat{\boldsymbol{x2}}\right) + \boldsymbol{L2sin}\left(\widehat{\boldsymbol{x2}}\right)\right)}{\boldsymbol{L2^2} + \boldsymbol{L1^2}}\right) \quad (41)$$

$$\boldsymbol{x3} = \boldsymbol{atan}\left(\frac{\boldsymbol{sin}\left(\widehat{\boldsymbol{x3}}\right)}{\boldsymbol{cos}\left(\widehat{\boldsymbol{x3}}\right)}\right) \quad (42)$$

$$\boldsymbol{xx5} = \boldsymbol{atan}\left(\frac{\boldsymbol{sin}\left(\widehat{\boldsymbol{x5}}\right)}{\boldsymbol{cos}\left(\widehat{\boldsymbol{x5}}\right)}\right) \quad (43)$$

- Cinemática inversa para la pierna izquierda: La cadena cinemática de la pierna izquierda con 6 articulaciones $LHipYawPitch\ (x_1)$, $LHipRoll\ (x_2), LHipPitch\ (x_3), LKneePitch\ (x_4), LAnklePitch\ (x_5)$ y $LAnkleRoll\ (x_6)$, los cuales son los valores a calcular, y recibiendo como parámetros de entrada la posición del efector del pie izquierdo (P_x, P_y, P_z), y asignado los valores de $L1 = HipOffsetY$, $L2 = HipOffsetZ, L3 = FootHeight$, $L4 = ThighLength$ y $L5 = TibiaLength$. Se puede calcular una solución cerrada usando geometría, en la articulación $LKneePitch\ (x_4)$ que se encuentra entre dos eslabones, siendo equivalentes a un triángulo formado por los eslabones, tal como se indica en la siguiente figura:

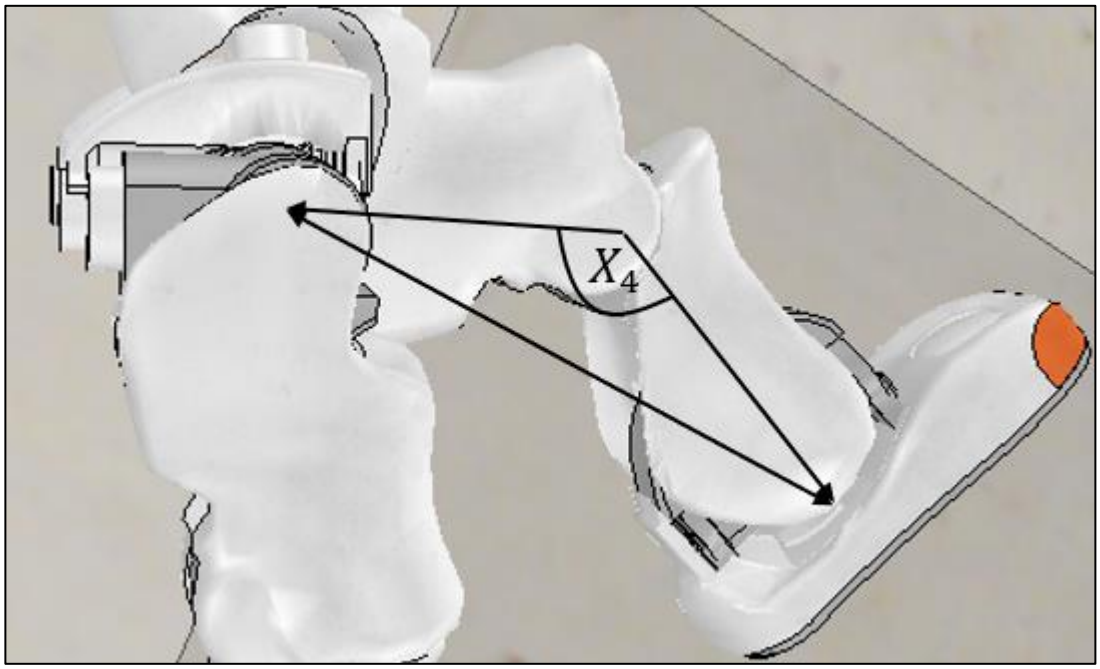


Figura 23. Triángulo formado por el ángulo $\mathbf{LKneePitch}\ (\mathbf{x_4})$

Primero se calcula el valor de distancia entre dos puntos, entre la posición de la articulación $LHipPitch\ (P1)$ y la articulación $\mathrm{LAnkleRoll}\ (P2)$, para por último aplicar una fórmula trigonométrica, tal como se indica a continuación:

$$\boldsymbol{d = sqrt\left(\left(P1(1) - P2(1)\right)^2 + \left(P1(2) - P2(2)\right)^2 + \left(P1(3) - P2(3)\right)^2\right)} \quad (44)$$

$$\boldsymbol{x4 = \left(pi - acos\left(\frac{L1^2 + L2^2 - d^2}{2 * L1 * L2}\right)\right)} \quad (45)$$

El cálculo de los restantes valores articulares se lo realiza de forma analítica, tal como se presenta a continuación:

$$x6$$

$$= atan\left(\frac{\left(sin(\widehat{x6})\right) * \left(L2 * cos(\widehat{x5}) + L1 * sin(\widehat{x4} + \widehat{x5})\right)}{\left(cos(\widehat{x2}) * \left(L2 * cos(\widehat{x5}) + L1 * sin(\widehat{x4} + \widehat{x6})\right)\right)}\right) \quad (46)$$

$$x5 = LAnklePitch \quad (47)$$

$$= asin(-((-L2 * (sin(\widehat{x5})) - L1 * ((sin(\widehat{x5}))$$

$$* (cos(\widehat{x4})) + (cos(\widehat{x5})) * (sin(\widehat{x4})))) * (L2 + L1$$

$$* cos(x4)) + L1 * (sin(x4)) * (L2 * (cos(\widehat{x5}))$$

$$+ L1 * ((cos(\widehat{x5})) * (cos(\widehat{x4})) - (sin(\widehat{x5}))$$

$$* (sin(\widehat{x4})))))/((L1\text{^}2) * ((sin(x4))\text{^}2) + (L2$$

$$+ L1 * (cos(x4)))\text{^}2))$$

$$x2 = -pi/4 + acos\left(cos(\widehat{x2})\right) \quad (48)$$

$$x3 = asin\left(\frac{\left(sin(\widehat{x2})\right) * \left(sin(\widehat{x3})\right)}{sin\left(x2 + \frac{pi}{4}\right)}\right) \quad (49)$$

$$x3 = asin\left(\frac{\left(sin(\widehat{x2})\right) * \left(sin(\widehat{x3})\right)}{sin\left(x2 + \frac{pi}{4}\right)}\right) \quad (50)$$

$$x1 = acos\left(\frac{\left(cos(\widehat{x1})\right) * \left(sin(\widehat{x2})\right)}{sin\left(x2 + \frac{pi}{4}\right)}\right) \quad (51)$$

2.5. Estabilidad de un robot humanoide bípedo

Para que un robot humanoide sea capaz de realizar movimientos como caminar, manipular objetos u otro, sin que pierda su estabilidad, se analizan las fuerzas internas y externas que influyen en el robot, aplicándose diferentes criterios de estabilidad dinámicos y estáticos que aseguren el que robot mantenga su estabilidad previa a que realice movimientos.

Principalmente previo aplicar los criterios de estabilidad se analiza el centro de masa, polígono de soporte y ZMP.

2.5.1. Centro de masa (CoM):

Es la ubicación del promedio de todas las masas que conforman el robot, puede ser calculada con la siguientes formulas:

$$\boldsymbol{P_{Com}} = \frac{\sum_{i=1}^{n} \boldsymbol{m_i} * \boldsymbol{p_i}}{\sum_{i=1}^{n} \boldsymbol{m_i}} \quad (52)$$

Donde m_i es la masa de cada eslabón del robot y p_i es la posición absoluta del CoM de cada eslabón del robot. Se requiere que cada p_i se mantenga con el mismo sistema de coordenadas de referencia que los demás.

2.5.2. Polígono de soporte

Es el área que conecta todos los puntos de contacto del apoyo del robot con el suelo o una base fija, mientras que el robot se encuentre en movimiento o una posición fija. El polígono de soporte en un robot humanoide es el área de sus pies que se encuentre en contacto con el suelo, si se encuentra apoyado con los dos pies polígono de soporte será

el área total que se genere entre los dos pies, si se encuentra apoyado en un solo pie, el polígono de soporte solo será el área de apoyo de ese pie, tal como se indica en la siguiente figura.

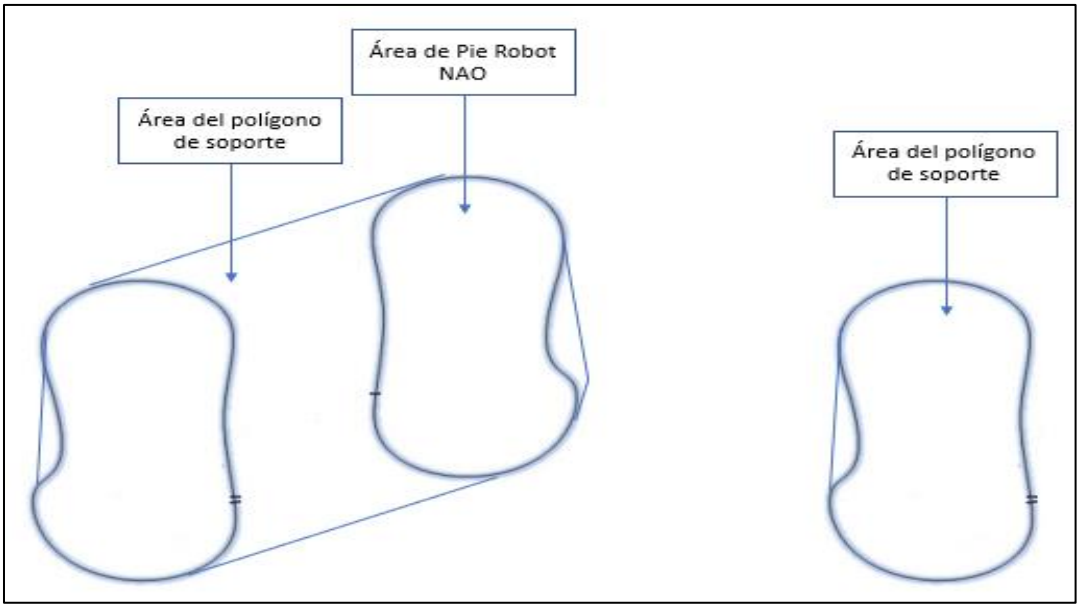


Figura 24. Polígono de soporte en: Apoyo doble (Izq.) y simple (Der.)

2.5.3. Punto de momento Cero (Zero Moment Point - ZMP)

Es un punto en la superficie en el cual momento de las fuerzas de inercia y las fuerzas de la gravedad que actúan entre la superficie del suelo y el área de soporte del pie dan una resultante de cero. Durante el momento en el que la aceleración del robot es cero o los movimientos del robot son relativamente lentos, la proyección del CoM en la superficie tiende aproximarse a la posición del ZMP. (Fadli, Hidayat, & Machbub, 2016).

2.5.4. Estrategias para mantener la estabilidad

El criterio de estabilidad estático y dinámico comúnmente usado para robots humanoides se basa en la posición del ZMP, en el cual se establece que la posición del ZMP se debe de encontrar dentro del área del polígono de soporte del robot, y si se encuentra apoyado en los dos pies, estos deben de estar en contacto con el mismo plano del suelo. Como criterio de estabilidad Dinámico se encuentra el modelo de péndulo invertido. (Wang, Tang, Ou, & Xu, 2012).

Cuando el robot se encuentra en movimiento se puede ver afectado por perturbaciones, efectos dinámicos mientras se mueve o movimientos que causen que el robot pierda el equilibro, por lo tanto, el análisis de la posición del ZMP se lo realiza constantemente, con la finalidad de determinar si los movimientos son estables o no, en el caso de que el ZMP salga del área del polígono de soporte, se requieren tomar medidas de compensación y corrección de las variables angulares de las articulaciones usando la cinemática inversa, principalmente se realiza esto para las articulaciones de las piernas, ya que tienen mayor influencia en la estabilidad del robot.

Las estrategias de balance y estabilidad de robots bípedos se basan en los estudios biomecánicos de estabilidad y marcha de un humano, en la cual se demostró que los humanos aplican 2 estrategias de estabilidad las cuales son: estrategia de tobillo, estrategia de cadera-tobillo. (Hofmann, 2006)

- Estrategia de tobillo: Si una perturbación mueve ligeramente al CoM provocando una inestabilidad, la estrategia de tobillo se encarga en aplicar un torque en las articulaciones del tobillo para reposicionar al CoM dentro del polígono de soporte,

teniendo en cuenta que el torque que se aplique no sea lo suficientemente fuerte ya que podría hacer girar el pie. Por lo tanto, esta estrategia se usa cuando las perturbaciones son pequeñas evitando así la necesidad de aplicar un torque grande en el tobillo, tal como se indica en la (Figura 25). (Yi, Zhang, Hong, & Lee, 2013).

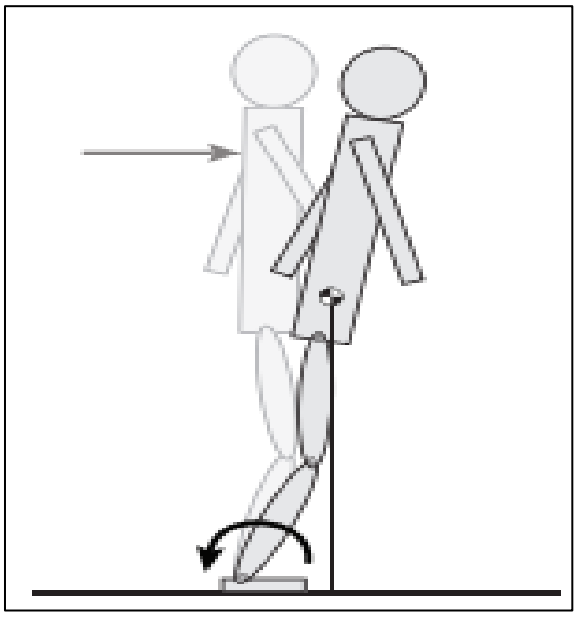


Figura 25. Estrategia del tobillo
(Kiemel, 2012)

- Estrategia de Cadera – Tobillo: Si una perturbación es muy fuerte y mueve abruptamente el CoM del polígono de soporte, la estrategia Cadera-Tobillo logra estabilizar al robot con dos pasos, primero aplica un torque en la articulación de la cadera en la misma dirección de la perturbación, provocando una inflexión del torso y después se aplica un torque en la articulación del tobillo en dirección contraria de la perturbación, con esto se logra mover al CoM hacia atrás de los pies y dejándolo dentro del área del polígono de soporte. Esta estrategia se usa para compensar la

estrategia de tobillo, ya que permite estabilizar al robot con perturbaciones leves y fuertes, tal como se indica en la (Figura 26). (Kiemel, 2012).

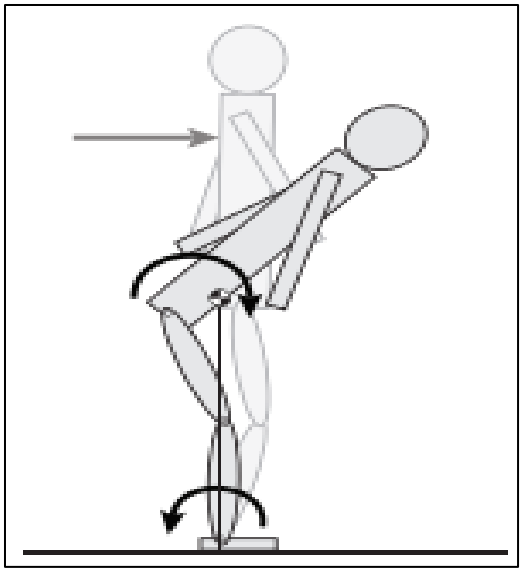


Figura 26. Estrategia cadera-tobillo
Fuente: **(Kiemel, 2012)**

CAPÍTULO III

DISEÑO Y CONSTRUCCIÓN DE LA RED DE SENSORES INERCIALES

En el presente capítulo se describe el desarrollo del diseño e implementación de la red de sensores inerciales, multiplexación de sensores, procesamiento de datos para la obtención de orientación de cada sensor y a su vez la transmisión de los mismos vía inalámbrica, además de la distribución y ubicación de cada sensor tomando en cuenta las referencias anteriormente descritas.

Para el procesamiento de datos obtenidos de los sensores inerciales se usó el filtro complementario descrito en el capítulo 2, para la comunicación inalámbrica entre el controlador de la red de sensores y la computadora, se usó el protocolo TCP/IP en el cual el controlador de la red de sensores trabaja como esclavo y el computador como maestro.

3.1. Diseño y distribución de la red de sensores inerciales

La ubicación y distribución de los sensores inerciales se realizó en base a la referencia: Conexión recomendada de una red de 16 IMU’s para el cuerpo humano de la empresa TECHNAID (Technaid, 2016). De la cual se destaca la distribución de los sensores inerciales de acuerdo a las cadenas cinemáticas del robot NAO, tal como se indica en la siguiente figura:

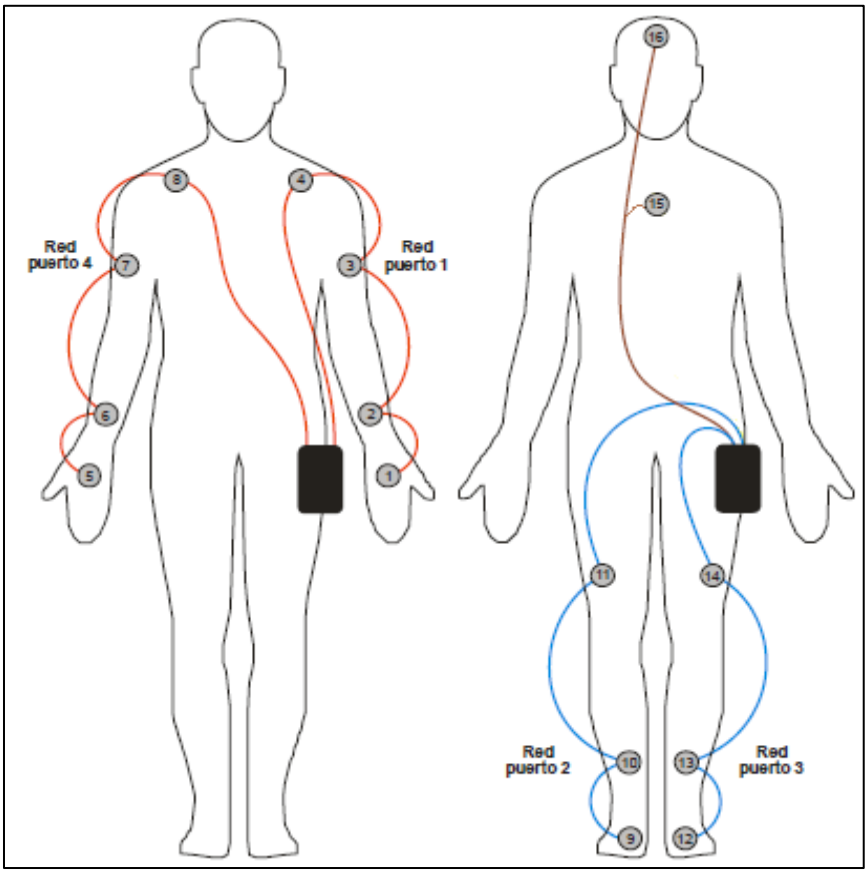


Figura 27. Distribución y ubicación de los IMU's. (Modificado)
Fuente: **(Technaid, 2016)**

De acuerdo al análisis de la anterior figura se presenta en la (Tabla 14), la distribución de los sensores de acuerdo a las siguientes características: el canal I2C al que se encuentre conectado el sensor, la red cinemática a la que pertenece, el código que se le asignó al sensor en donde el primer digito indica la cadena cinemática a la que pertenece y el segundo y tercer digito indica el número de sensor del total de la red, y por último su posición (Horizontal o Vertical) para que todos los sensores mantengan el mismo eje de coordenadas.

Tabla 14

Distribución y ubicación de cada sensor inercial

N° Sensor	Ubicación	Canal I2C	Red Cinemática	Sufijo	Orientación
1	Mano derecha	1	1	101	Horizontal
2	Antebrazo derecho			102	Horizontal
3	Brazo derecho			103	Horizontal
4	Escapula derecha			104	Vertical
5	Mano izquierda		2	205	Horizontal
6	Antebrazo izquierdo			206	Horizontal
7	Brazo izquierdo			207	Horizontal
8	Escapula izquierda			208	Vertical
9	Pecho o Lumbar	2	3	309	Vertical
10	Cabeza			310	Vertical
11	Pie derecho		4	411	Horizontal
12	Pierna derecha -			412	Vertical
13	Muslo derecho			413	Vertical
14	Pie Izquierdo		5	514	Horizontal
15	Pierna izquierda			515	Vertical
16	Muslo izquierdo			516	Vertical

La red de sensores inerciales se distribuyó de acuerdo a la (Tabla 14), para lo cual se usaron dos canales I2C, en donde los sensores de la red cinemática del brazo izquierdo y brazo derecho se conectan al primer canal I2C y las redes cinemáticas de la cabeza, pierna derecha e izquierda se conectan al segundo canal I2C.

Como controlador principal de la red de sensores se usó la tarjeta Wemos LoLin32 desarrollado por la empresa ESPRESSIF, la cual usa lenguaje de programación C++, cuenta con dos procesadores que trabajan a 240 MHz, 4 MB de memoria flash, Conectividad WiFi 802.11 b/g/n, Bluetooth 4.0 LE, 26 puertos GPIO,12 entradas analógicas, 2 canales I2C, 2 canales SPI, 3 canales UART, y un cargador para baterías de tipo LiPo. Sus ventajas principales para la selección de dicha tarjeta recaen en el

acceso a su antena de conexión a WiFi de forma directa, la capacidad de poder programarla desde el IDE de Arduino, frecuencia de trabajo de 80 MHz si se la programa con C++, dos canales I2C y su cargador de baterías LiPo, la tarjeta se la presenta en la siguiente figura:

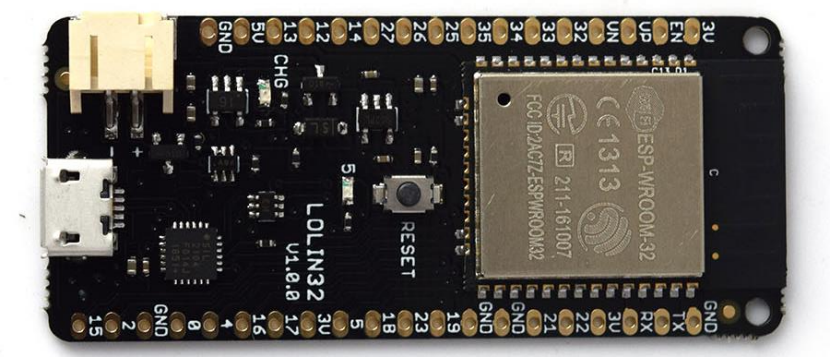


Figura 28. Vista superior de la tarjeta Wemos LoLin32
Fuente: **(Wemos, 2018)**

Como se detalló en el capítulo 2, los sensores inerciales MPU9250 solo cuentan con dos posibles direcciones, las cuales dependen del estado lógico del pin AD0, y sabiendo que el sensor trabaja como esclavo y el controlador como maestro. Se debe realizar una selección previa del sensor a través de sus pines de salida digital que se conectan a los pines AD0 de cada sensor, en donde se activan todas sus salidas y solo se desactiva el pin conectado al AD0 del sensor al que se requiere seleccionar, tal como se muestra en la siguiente tabla:

Tabla 15
Selección de sensor en base al estado del pin AD0 de cada sensor

	Pines de salida del Controlador															
Sensor	18	5	17	16	4	0	2	15	12	13	25	33	32	14	27	26
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
101	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

CONTINÚA

102	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
103	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
104	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
205	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1
206	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1
207	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1
208	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
309	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
310	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
411	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
412	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
413	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
514	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
515	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
516	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

Con la activación y desactivación de las salidas del controlador de acuerdo a la anterior tabla, se logra poder seleccionar a un sensor especifico de acuerdo a su dirección I2C, ya que el sensor MPU9250 solo cuenta con dos direcciones que son 0X68H y 0X69H. En el caso de que solo un pin AD0 de un sensor en específico tenga el estado bajo, el sensor tendrá la dirección 0X68H, con esto con el controlador se podrá comunicar con el sensor de dirección 0x68H con solo desactivar la salida del pin que se conecte a dicho sensor y manteniendo en alto sus demás salidas, ubicando a los demás sensores con la dirección 0X69H.

3.2. Configuraciones iniciales

La configuración inicial que se debe de realizar en el controlar tiene que ver con la declaración de parámetros tales como variables, canales I2C, comunicación inalámbrica y configuración de sensores, tal como se muestra en el siguiente diagrama de flujo correspondiente a la (Figura 29).

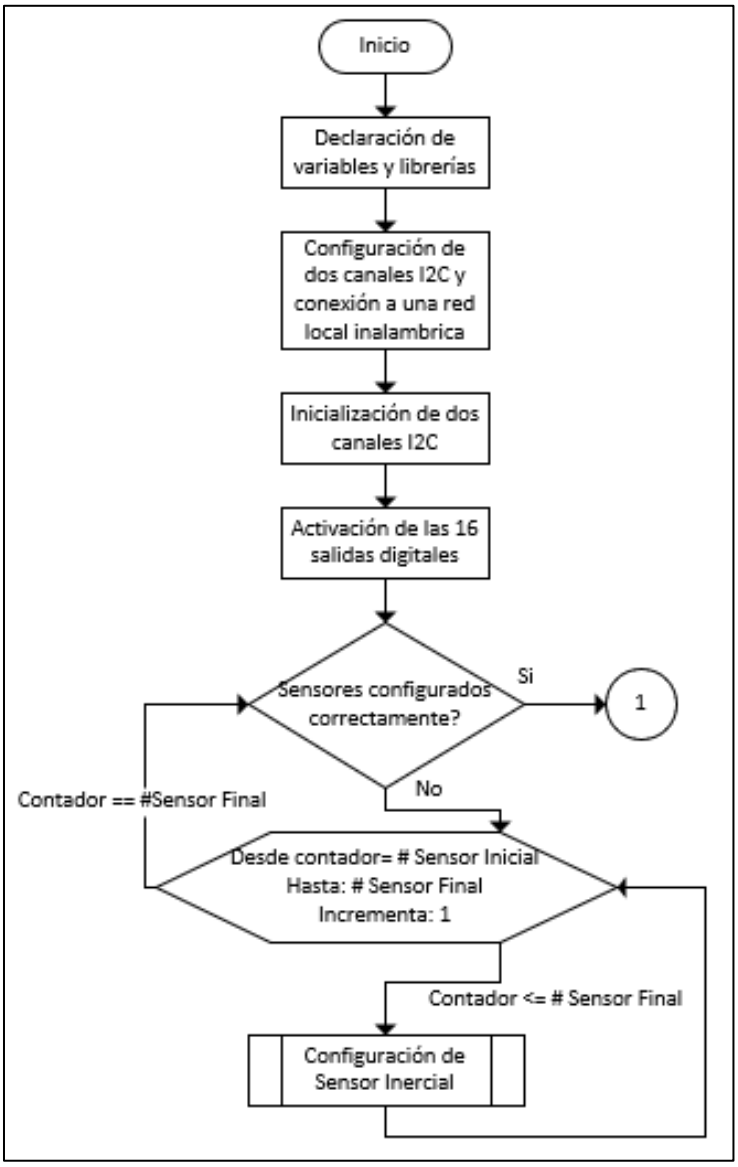


Figura 29. Configuraciones iniciales

Como configuración inicial se realiza la declaración de las variables, la configuración de dos canales I2C para la conexión con los sensores para lo cual se especifica los pines de conexión SDA y SCL, y velocidad de transmisión, al igual que se detalla que el controlador el cual trabaja como maestro de los dos canales mientras que los sensores trabajaran como esclavos, después se procede a habilitar los dos canales, seguidamente

se les asigna la dirección 0X69H a todos los sensores. Para comunicarse con los sensores MPU9250 se usó la librería MPU9250 desarrollada por la empresa Bolder Flight Systems. (GitHub, 2018). La configuración de los 16 sensores se lo realiza en orden de acuerdo a la (Tabla 15), para lo cual se usa la subrutina: Configuración de sensor inercial, en el caso de que no se configure algún sensor inercial se seguirá ejecutando el bucle hasta que todos los sensores inerciales hayan sido configurados, si se configuran todos los sensores inerciales se procede a ejecutarse la subrutina: Conexión a una red Inalámbrica.

El diagrama de flujo de la subrutina: Configuración de sensor inercial, se presenta en la siguiente Figura:

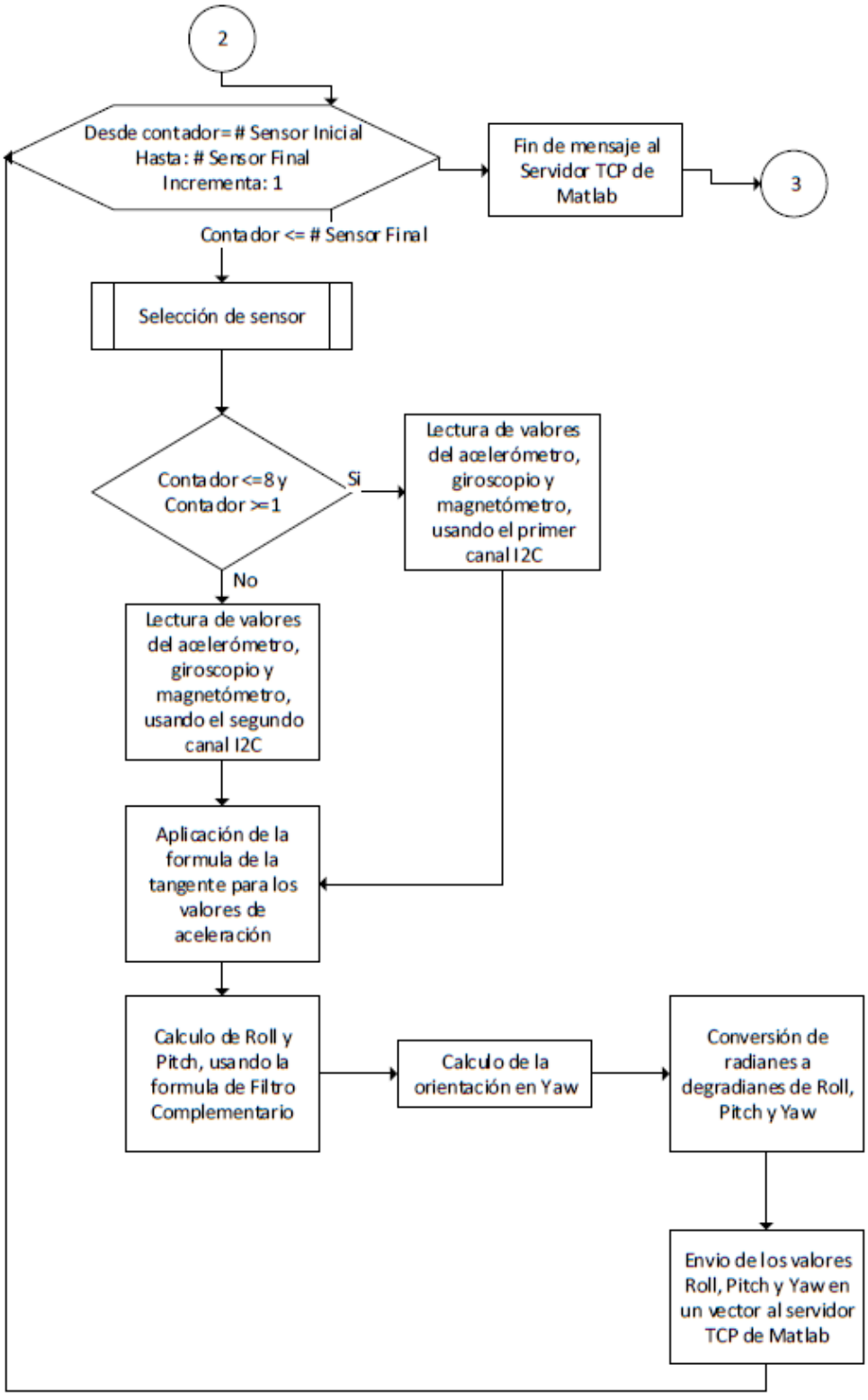


Figura 30. Subrutina: Configuración de sensor inercial

Para la configuración de los sensores inerciales se requiere primero seleccionar al sensor con el que se requiere trabajar, para lo cual se usa la subrutina: Selección de sensor, una vez seleccionado el sensor se procede a seleccionar el canal I2C con el que se va a trabajar de acuerdo al sensor que se va a configurar conforme a la (Tabla 14), con esto se logra que los sensores del 1 al 8 se configuren con el primer canal I2C, y los demás se configuren con el segundo canal, la configuración para los sensores es la misma independientemente del canal en el que se encuentre.

Primero se procede a configurar al acelerómetro con una sensibilidad de $\pm 8G$, si no se configura el acelerómetro se termina el proceso de configuración para dicho sensor y se incrementa un contador de error, si se configura satisfactoriamente se procede a configurar el giroscopio con una sensibilidad de $\pm 2000 \frac{rad}{s}$, después se configura un filtro de 20 Hz para filtrar el ruido, la última configuración a realizarse es la frecuencia de salida de datos.

Para determinar la frecuencia de salida se requiere primero determinar la frecuencia de muestreo de datos que debe de realizar el controlador. Por lo tanto, primero se midió el tiempo que toma el controlador en realizar la lectura de los datos del acelerómetro, giroscopio y magnetómetro, el cálculo de la orientación y envió de datos al maestro vía inalámbrica. Para determinar el tiempo se realizó una muestra de la toma de 100 medidas para el total de los 16 sensores, dejando una media de $0.06s$ y una frecuencia de $16.67Hz$. Por lo tanto, se asume como frecuencia de muestro $16\,Hz$.

Para determinar la frecuencia de salida de datos se usa la (Ecuación 36), para luego obtener el divisor de frecuencia, para calcular la frecuencia de salida de datos se tomó en cuenta el criterio de Nyquist que se detalla en la ecuación 36, tomando en cuenta que la salida de datos del magnetómetro no pueden ser mayores a $100\ Hz$ y que la frecuencia interna es de 1000 Hz si se trabaja con comunicación I2C, se procede a realizar para determinar el divisor de frecuencia, tal como se detalla a continuación:

$$\boldsymbol{Frecuencia\ de\ muestreo \geq 2\ frecuencias\ de\ salida} \quad (53)$$

$$\boldsymbol{Frecuencia\ de\ salida\ \leq 8\ Hz} \quad (54)$$

$$\boldsymbol{Divisor\ de\ frecuencia = \frac{Frecuencia\ interna}{Frecuencia\ de\ salida} - 1} \quad (55)$$

$$\boldsymbol{Divisor\ de\ frecuencia = \frac{1000\ Hz}{8\ Hz} - 1} \quad (56)$$

$$\boldsymbol{Divisor\ de\ frecuencia = 124} \quad (57)$$

De la (Ecuación 39) se tiene que el divisor de frecuencia es de 124, el cual es el último parámetro a configurar y reestableciendo el filtro pasa bajos del sensor en 20Hz.

El diagrama de flujo de la subrutina de: Selección de sensor, se presenta en la siguiente Figura:

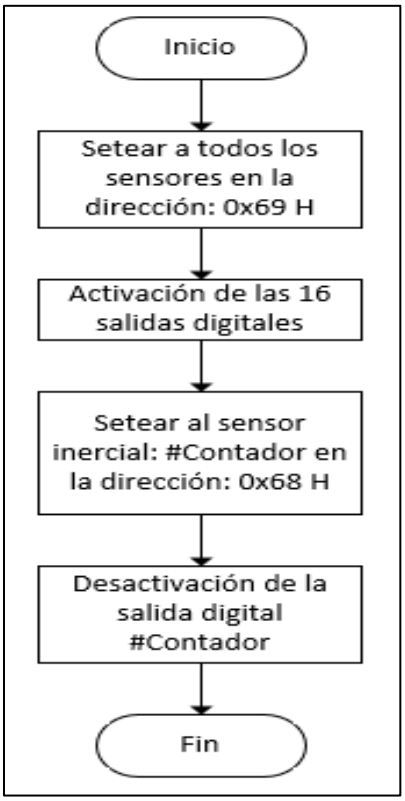


Figura 31. Subrutina: Selección

En esta subrutina se realiza el direccionamiento de los sensores, cambiando sus direcciones I2C, entre 0X68H y 0X69H, con la activación de las salidas que se conectan a los pines AD0 de cada sensor, de acuerdo a la (Tabla 15), esta selección se la realiza conforme al sensor que se encuentre configurándose o leyéndose.

3.3. Configuración de la comunicación inalámbrica

El diagrama de flujo de la etapa de la comunicación inalámbrica se presenta en la siguiente Figura:

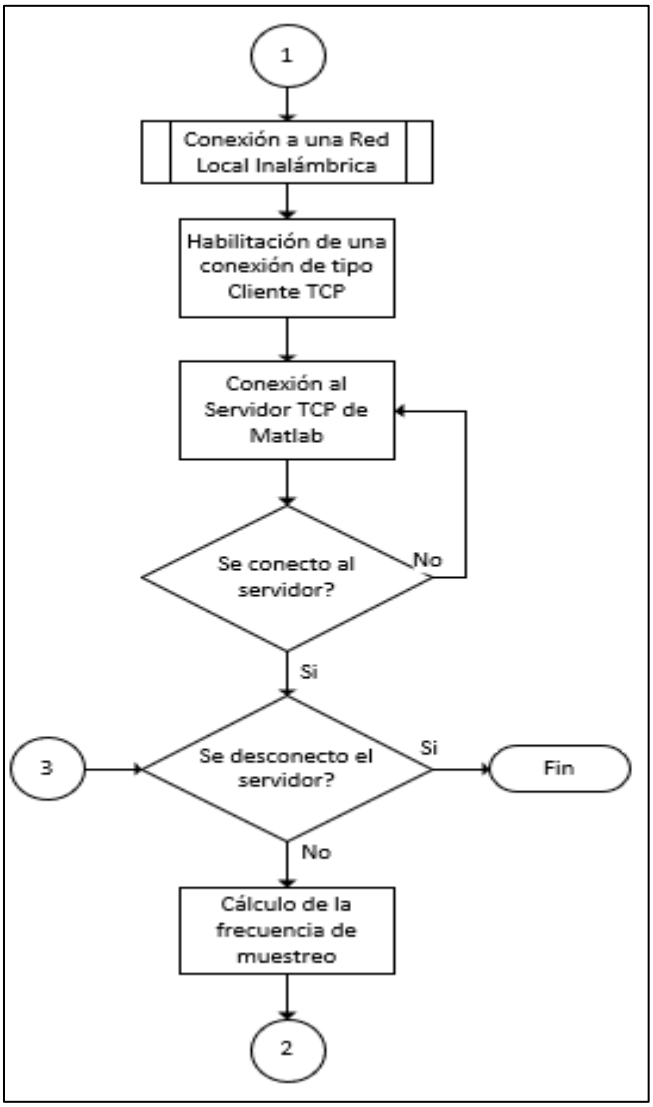


Figura 32. Etapa de comunicación inalámbrica

Esta etapa consiste en conectar el controlador a una red inalámbrica local, para luego poder conectarse al servidor de Matlab, se usó el protocolo de comunicación TCP/IP entre el servidor de Matlab y el cliente que es el controlador. Como primer paso se realiza la conexión a la red inalámbrica, después se establecen parámetros como la dirección IP del servidor y el puerto al que se debe de conectar que en este caso es el puerto 80, se

intentará realizar esta conexión con el servidor hasta que sea satisfactoria, en el caso de que se desconecte del servidor del cliente, se terminará el programa.

El diagrama de flujo de la subrutina: Conexión a una red inalámbrica se presenta en la siguiente Figura:

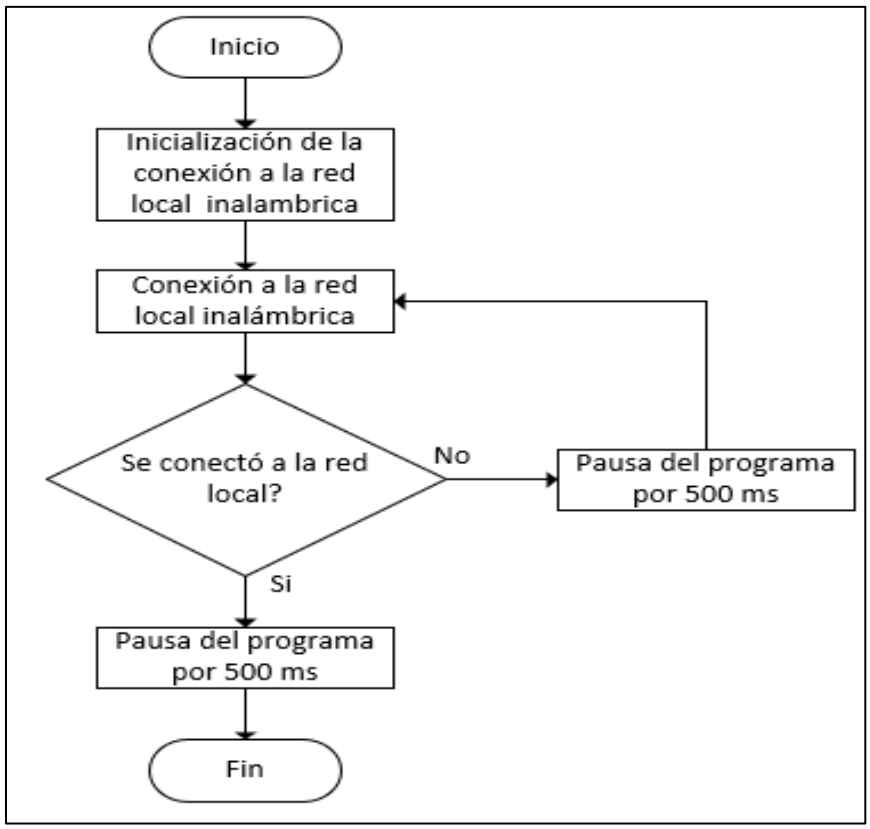


Figura 33. Subrutina de conexión a una Red Local Inalámbrica

Se usó una red inalámbrica local y se configuró una comunicación usando el protocolo TCP/IP entre un servidor desde Matlab y un cliente desde el controlador, esta configuración se la realiza en el controlador, para lo cual requiere previamente de la conexión a la red local, dando los parámetros del nombre de la red y la contraseña, el

intento de conexión a la red inalámbrica se mantendrá en un bucle hasta que se realice la conexión y se le asigne una dirección IP al controlador.

3.4. Procesamiento de datos de los IMU's

El proceso inicia con la selección del sensor con el que se va a trabajar usando la subrutina Selección sensor detallada anteriormente, este proceso se realiza de forma repetitiva para los 16 sensores. Se realiza la selección de canal I2C dependiendo del sensor en proceso de acuerdo a la tabla 15, del sensor se leen los datos del acelerómetro en m/s^2de los 3 ejes, los datos del giroscopio en $\frac{rad}{s}$ de los 3 ejes y finalmente los datos del magnetómetro en μT en los 3 ejes.

Con los datos obtenidos de los 3 sensores, se procede a realizarse el cálculo de la tangente de los valores de la aceleración usando las ecuaciones (2) y (3), con los valores de cada IMU. Después se procede a estimar la frecuencia de muestreo en segundos, para esto se utiliza la función millis (), el cual retorna el tiempo de ejecución desde el inicio del programa que ejecuta el controlador hasta el punto en que se lo llame, con esto se realiza la diferencia entre el tiempo de anterior iteración con el tiempo actual, dejando así la frecuencia de muestreo para dicho sensor. Con la frecuencia de muestreo y los datos de la tangente de los acelerómetros se aplica las ecuaciones (4) y (5) del filtro complementario, con la finalidad de obtener la orientación de cada sensor en Roll y Pitch.

Finalmente se procede a calcular la orientación en Yaw usando las ecuaciones (6), (7) y (8), y los valores de Roll y Pitch, después se realiza la conversión de las unidades de orientación de radianes a grados sexagesimales, finalmente para el envío de los datos de orientación de cada IMU, se conforma un mensaje en el cual se asigna cada dato

separado por una “x” y su frecuencia de muestreo, este mensaje contendrá los valores de orientación de los 16 IMU’s para luego poder ser enviado al servidor, una vez enviado el mensaje se procederá a realizar de nuevo la iteración desde el inicio.

El diagrama de flujo correspondiente al procesamiento de los datos de los sensores inerciales se presenta en la siguiente Figura:

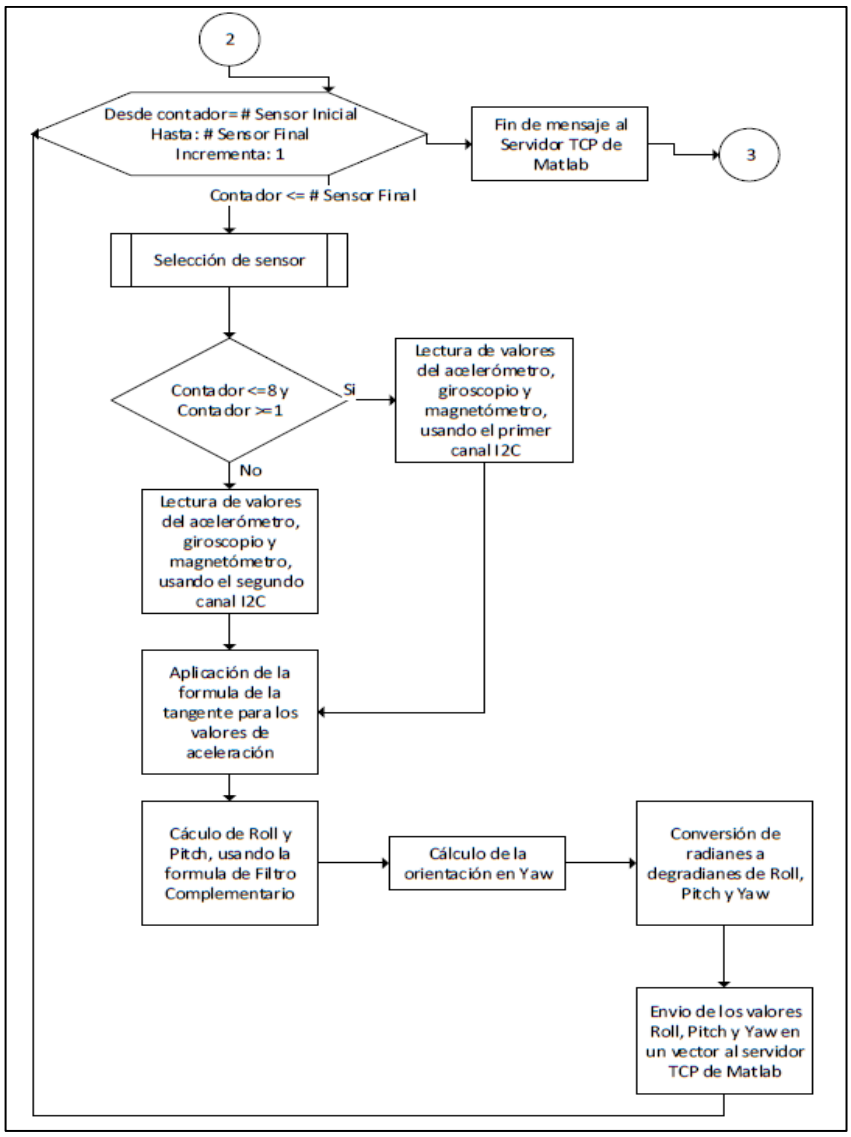


Figura 34. Etapa de procesamiento de datos de los 16 IMU’s

3.5. Elaboración e implementación de la red de sensores

La conexión entre IMU's y controlador de cada red cinemática se la realizó usando buses de datos, cada bus de datos cuenta con 10 conductores. El sistema completo está formado por una tarjeta principal en la que se encuentra el controlador Wemos LoLin32, conectores de bus datos para la conexión a los buses de datos de cada red cinemática, un pulsador de Reset, un switch de encendido y apagado, y una batería LiPo de 3.3V de 3A, el diagrama esquemático del controlador principal se presenta en la siguiente Figura:

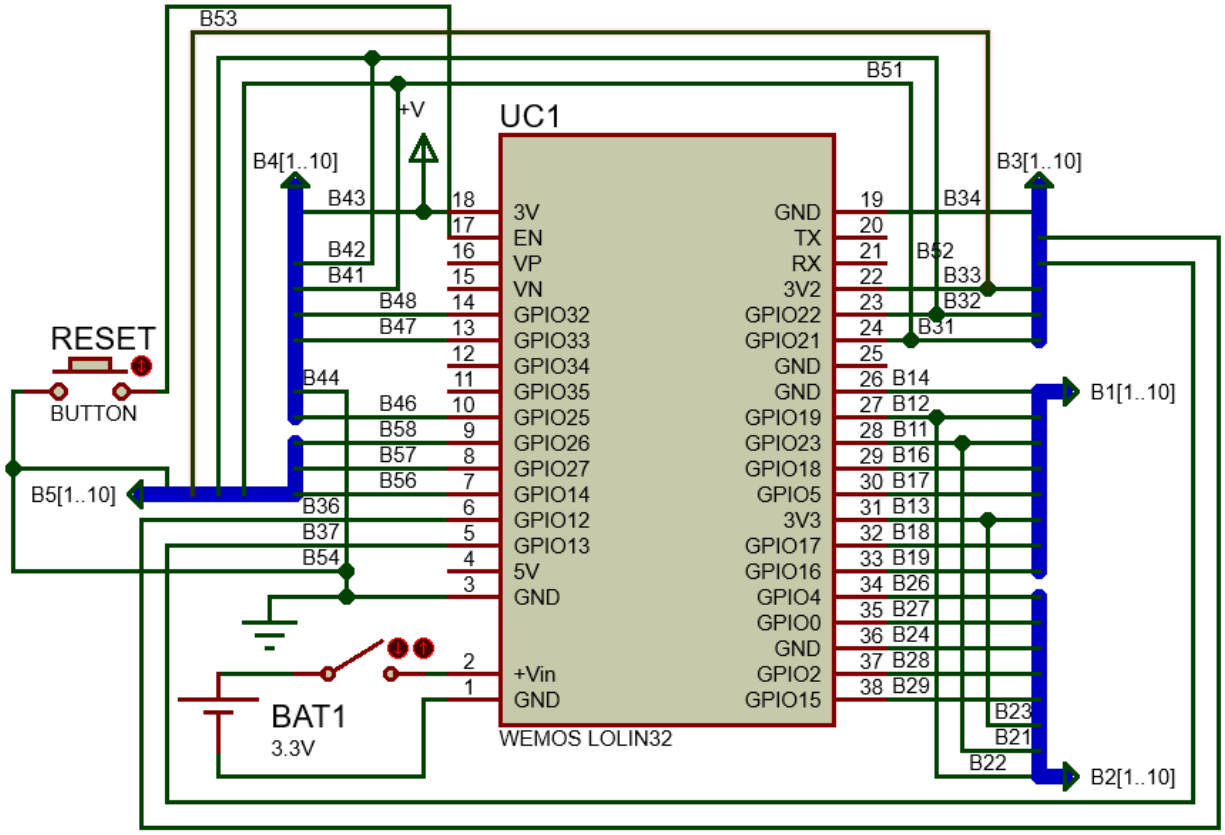


Figura 35. Diagrama esquemático del controlador principal

Los buses de datos que se usaron se representan con color azul tal como se indica en la anterior figura, el controlador cuenta con un cargador de baterías LiPo, por lo cual

su conexión es directa a la batería y esta se carga cuando el controlador se conecta a un computador por su puerto micro USB tipo B. se realizaron tarjetas para cada IMU, su conexión con el bus de datos se lo realiza a través de conectores de bus tipo macho, el diagrama esquemático realizado de la conexión de IMU con el bus de datos y este a su vez con el controlador, se presenta en la siguiente Figura:

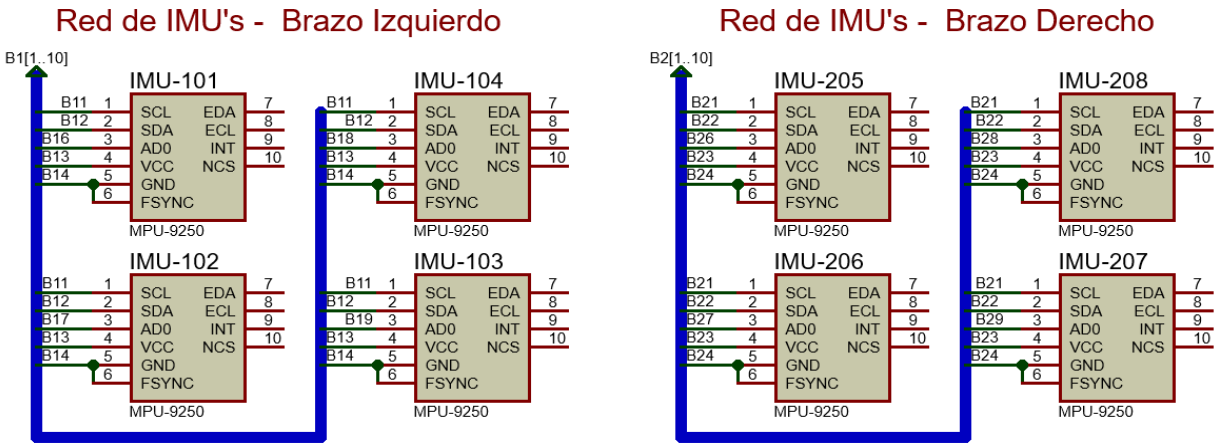


Figura 36. Diagrama esquemático de Red de IMU's para el brazo izquierdo y derecho

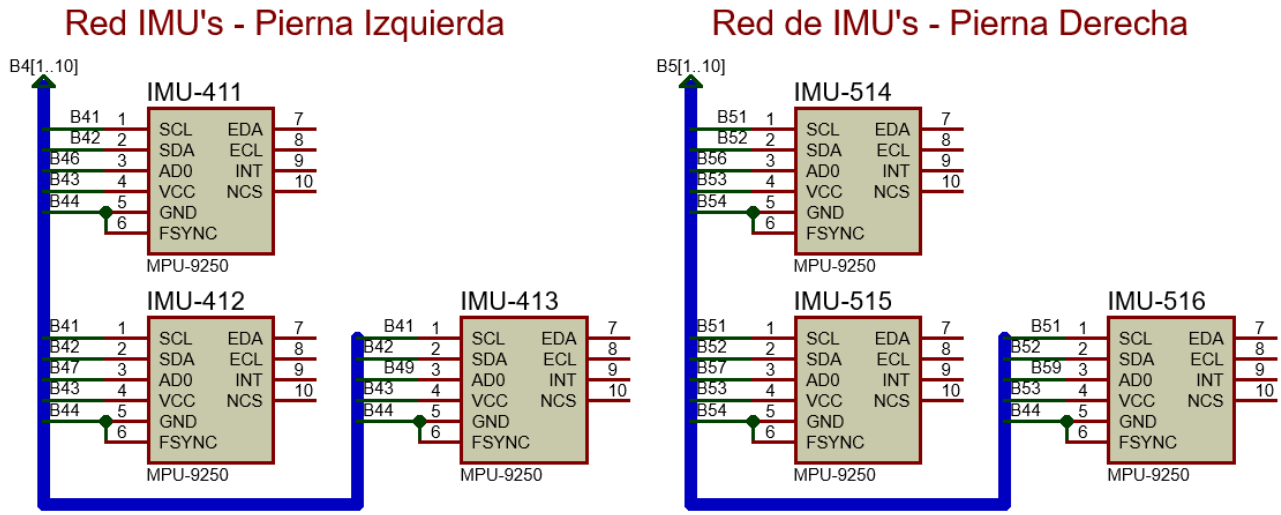


Figura 37. Diagrama esquemático de Red de IMU's de la pierna izquierda y Derecha

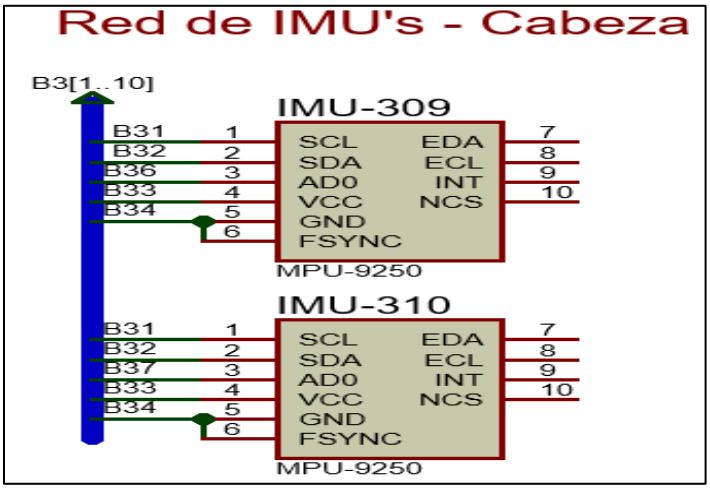


Figura 38. Diagrama esquemático: red de IMU's cabeza

El circuito impreso del controlador principal de la red de sensores se desarrolló con conectores de bus para cada red, tal como se especificó anteriormente, en las siguientes figuras se presenta la vista frontal del PCB y la ubicación de cada conector para la conexión del bus de datos de cada red, al igual que la vista normal del PCB diseñado:

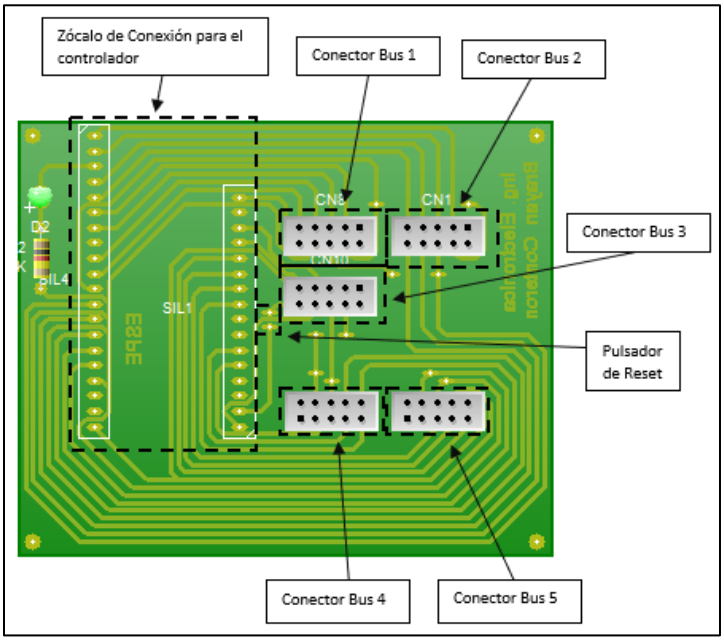


Figura 39. Vista normal del diseño PCB de la placa principal

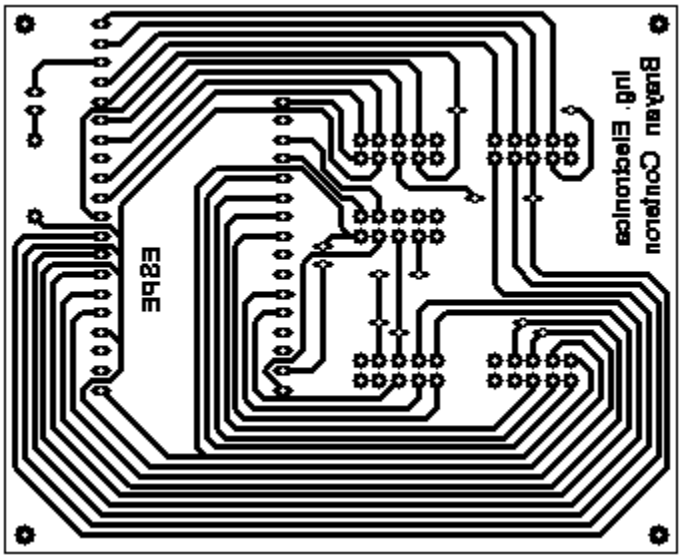


Figura 40. Diseño PCB de la tarjeta principal

Se diseñó un circuito con un formato general para todos los sensores, el cual permite conectar cada sensor dentro de una red de sensores, y que a su vez su pin AD0 pueda ser conectado a una salida específica del controlador, esto se logra usando un jumper en el conector de selección de posición, también cuenta con un zócalo de conexión de IMU para facilitar la conexión del IMU a la placa y a la red, tal como se indica en las siguientes Figuras:

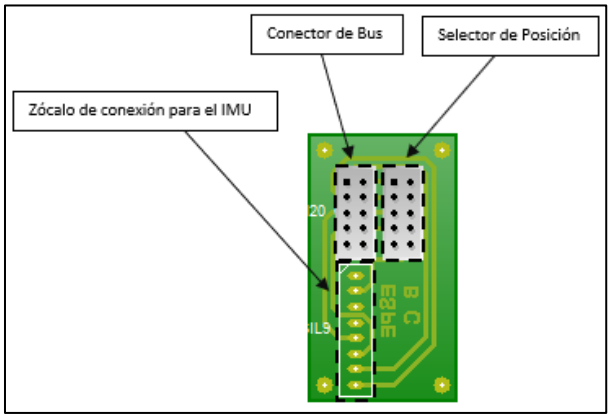


Figura 41. Vista del diseño PCB de la placa de cada IMU

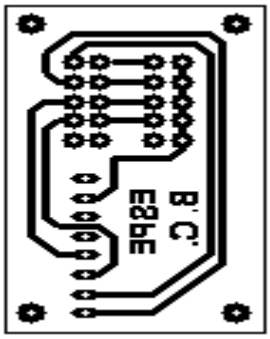


Figura 42. Diseño PCB de la placa de cada IMU

Se realizaron diseños de cajas para la placa principal y para las placas de sensores. Para la placa principal sirve como protección y ubicación del controlador, batería, pulsador y switch, además de facilitar su conexión y desconexión de los buses de datos de cada red de sensores, para las placas de sensores sirve para ubicar la placa, mantener fija la posición del sensor y la conexión del bus de datos. Los diseños 3D de las cajas se presentan en las siguientes Figuras:

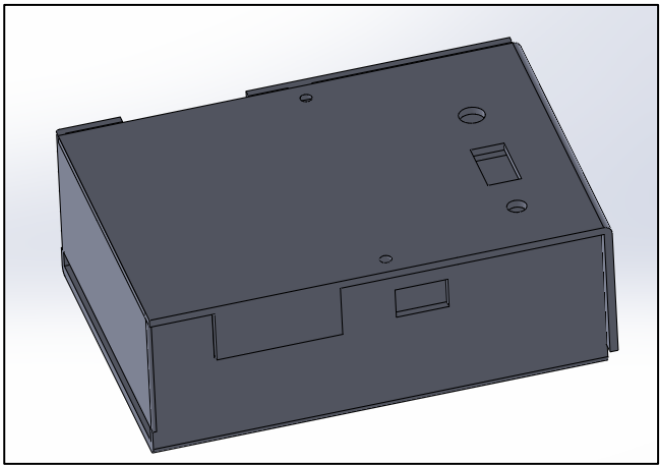


Figura 43. Diseño 3D de la caja principal armada

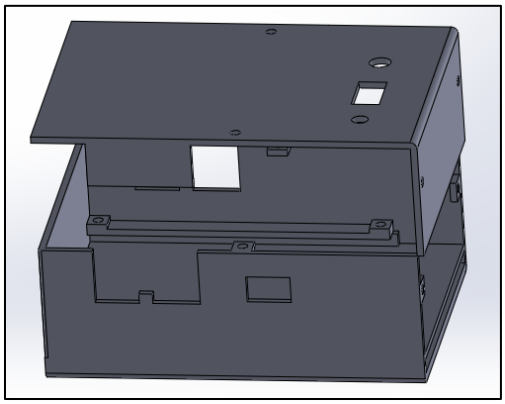


Figura 44. Diseño 3D de la caja principal desarmada

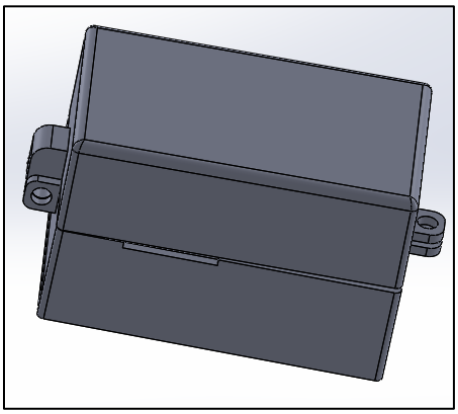


Figura 45. Diseño 3D de la caja de cada IMU armada

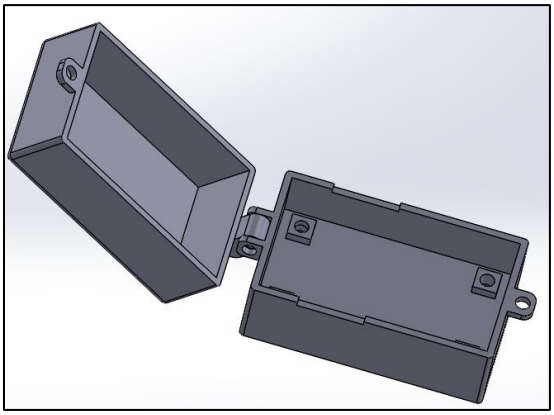


Figura 46. Diseño 3D de la caja de cada IMU abierta

Al igual que los diseños 3D para las cajas, también se realizaron diseños para corte láser en MDF de 3 mm, por su bajo costo, para la implementación del diseño de la red de sensores, los diseños se presentan en las siguientes figuras:

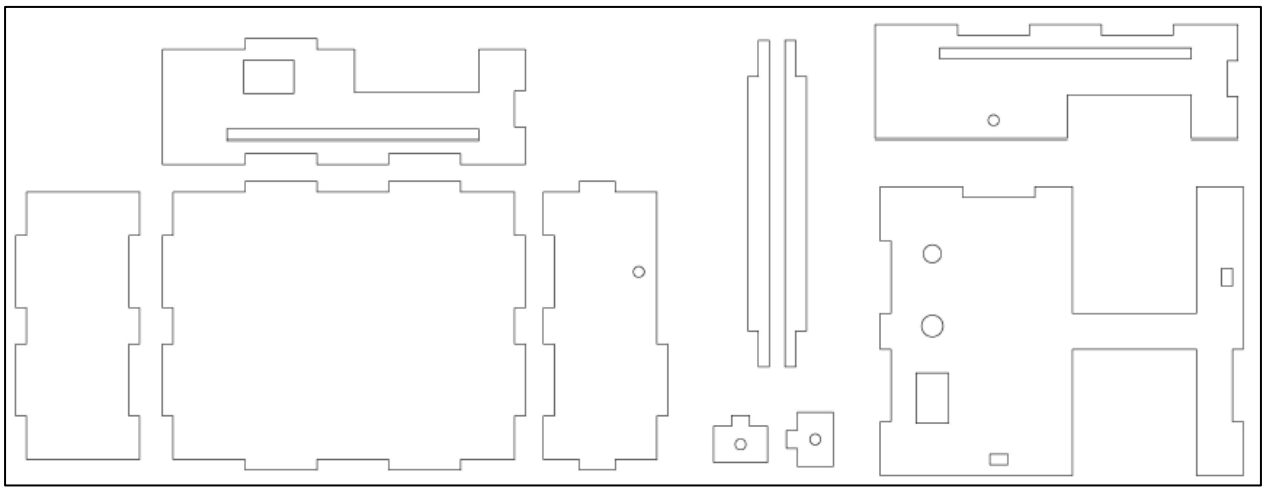


Figura 47. Diseño para corte láser de la caja principal

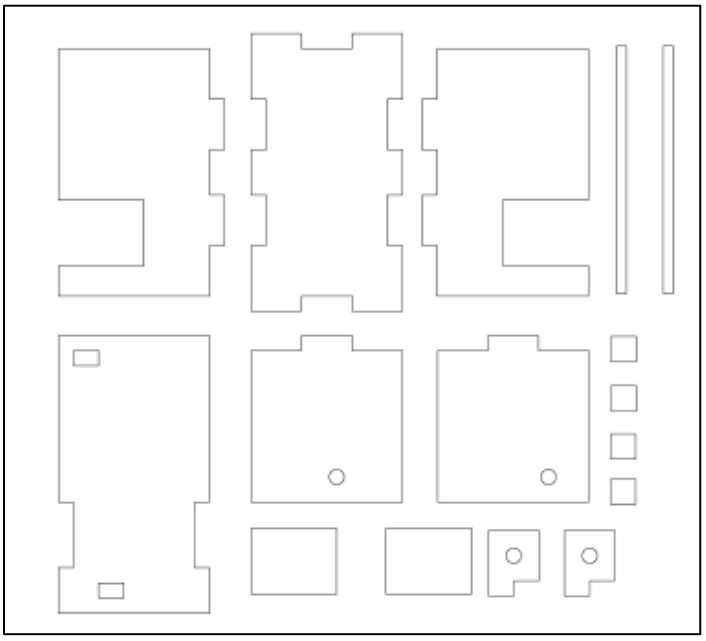


Figura 48. Diseño para corte láser de la caja para un IMU

La implementación de la red de sensores y la ubicación de cada IMU se realizó de acuerdo a la Figura 27, en donde se explica la ubicación de cada IMU. Se usó una faja lumbar como base de la red y en la cual se posición la caja del controlador, la implementación y ubicación de la red de sensores en un humano se presenta en la siguiente figura:

Figura 49. Montaje del prototipo de la red de IMU's en un humano

CAPÍTULO IV

CARACTERIZACIÓN DEL ROBOT HUMANOIDE NAO

La resolución de la cinemática directa e inversa, cálculo del CoM, ZMP y polígono de soporte, se la realizó en base a la información sobre el robot NAO detallada en el capítulo 2 sección 2 y 3. Además se usó el software matemático Matlab para dicha resolución y la simulación del robot NAO se la realizó con el simulador de robots V-Rep.

Se desarrollaron funciones para la solución de la cinemática directa en inversa con código de Matlab, desarrollados y probados en el software Matlab, las funciones se desarrollaron para trabajar en conjunto con la red de sensores inerciales, el simulador de Robots V-Rep y el Robot humanoide NAO, desarrollados en la presente tesis.

4.1. Obtención de la cinemática del robot NAO

4.1.1. Cinemática Directa

El desarrollo de la cinemática directa del robot Nao se la realizó en base a los parámetros DH de cada una de las 5 cadenas cinemáticas y las ecuaciones detalladas en el capítulo 2, sección 3, apartado 4 literal 1. Para su desarrollo se diseñó previamente funciones en Matlab tales como: Matrices de rotación para cada eje uno de los 3 ejes, matriz de traslación y matriz de transformación, el desarrollo para cada una de las cadenas cinemáticas se presenta a continuación:

- Cinemática directa de la cabeza

```
theta_h(1) = 0;          % HeadYaw
theta_h(2) = 0;         % HeadPitch

A1 = Traslacion(0,0,NeckOffsetZ);
A2 = Traslacion(53.9,0,67.9);
R1 = Rx(pi/2);
R2 = Ry(pi/2);
T1 = T_DH(0,0,0,theta_h(1));
T2 = T_DH(0,-pi/2,0,theta_h(2)-pi/2);
T_H=A1*T1*T2*R1*R2*A2
```

Figura 50. Código de Matlab para el cálculo de la cinemática directa de la cabeza

- Cinemática directa del brazo izquierdo

```
theta_ls(1) = 0;         % LShoulderPitch
theta_ls(2) = 0;         % LShoulderRoll
theta_ls(3) = 0;         % LElbowYaw
theta_ls(4) = 0;         % LElbowRoll
theta_ls(5) = 0;         % LWristYaw

A1 = Traslacion(0,ShoulderOffsetY + ElbowOffsetY, ShoulderOffsetZ);
A2 = Traslacion(HandOffsetX,0,-HandOffsetZ);
R1 = Rx(pi/2);
R2 = Rz(pi/2);
%a,alpha,d,theta
T1 = T_DH(0, -pi/2, 0,  theta_ls(1));
T2 = T_DH(0,  pi/2, 0,  theta_ls(2)-pi/2);
T3 = T_DH(0,  -pi/2, UpperArmLength,     theta_ls(3));
T4 = T_DH(0, pi/2, 0,  theta_ls(4));
T5 = T_DH(0,  -pi/2, LowerArmLength,     theta_ls(5));
T_LS=A1*T1*T2*T3*T4*T5*R1*R2*A2
```

Figura 51. Código de Matlab para el cálculo de la cinemática directa del brazo izquierdo

- Cinemática directa del brazo derecho

```
theta_rs(1) = 0;          % RShoulderPitch
theta_rs(2) = 0;          % RShoulderRoll
theta_rs(3) = 0;          % LElbowYaw
theta_rs(4) = 0;          % RElbowRoll
theta_rs(5) = 0;          % LWristYaw

A1 = Traslacion(0,-ShoulderOffsetY-ElbowOffsetY, ShoulderOffsetZ);
A2 = Traslacion(HandOffsetX,0,-HandOffsetZ);
R1 = Rx(-pi/2);
R2 = Rz(-pi/2);
%a,alpha,d,theta
T1 = T_DH(0,  pi/2, 0,  theta_rs(1));
T2 = T_DH(0,  -pi/2, 0,  theta_rs(2)+pi/2);
T3 = T_DH(0,  pi/2, UpperArmLength,     theta_rs(3));
T4 = T_DH(0,  -pi/2, 0,  theta_rs(4));
T5 = T_DH(0,  pi/2, LowerArmLength,     theta_rs(5));
T_RS=A1*T1*T2*T3*T4*T5*R1*R2*A2
```

Figura 52. Código de Matlab para el cálculo de la cinemática directa del brazo izquierdo

- Cinemática directa de la pierna derecha

```
theta_rl(1) = 0;              % RHipYawPitch
theta_rl(2) = 0;              % RHipRoll
theta_rl(3) = 0;              % RHipPitch
theta_rl(4) = 0;              % RKneePitch
theta_rl(5) = 0;              % RAnklePitch
theta_rl(6) = 0;              % RAnkleRoll

A1 = Traslacion(0,-HipOffsetY, -HipOffsetZ);
A2 = Traslacion(0,0,-FootHeight);
R1 = Rz(pi);
R2 = Ry(-pi/2);
%a,alpha,d,theta
T1 = T_DH(0,  -pi/4,  0,  theta_rl(1)-pi/2);
T2 = T_DH(0,  -pi/2,    0,  theta_rl(2)-pi/4);
T3 = T_DH(0,  pi/2,     0,  theta_rl(3));
T4 = T_DH(-ThighLength, 0,  0, theta_rl(4));
T5 = T_DH(-TibiaLength, 0,  0, theta_rl(5));
T6 = T_DH(0,  -pi/2,    0,  theta_rl(6));
T_RL = A1*T1*T2*T3*T4*T5*T6*R1*R2*A2
```

Figura 53. Código de Matlab para el cálculo de la cinemática directa de la pierna derecha

- Cinemática directa de la pierna izquierda

```
theta_ll(1) = 0;                % LHipYawPitch
theta_ll(2) = 0;                % LHipRoll
theta_ll(3) = 0;                % LHipPitch
theta_ll(4) = 0;                % LKneePitch
theta_ll(5) = 0;                % LAnklePitch
theta_ll(6) = 0;                % LAnkleRoll

A1 = Traslacion(0,HipOffsetY, -HipOffsetZ);
A2 = Traslacion(0,0,-FootHeight);
R1 = Rz(pi);
R2 = Ry(-pi/2);
%a,alpha,d,theta
T1 = T_DH(0,   -3*pi/4,   0,   theta_ll(1)-pi/2);
T2 = T_DH(0,   -pi/2,     0,   theta_ll(2)+pi/4);
T3 = T_DH(0,   pi/2,      0,   theta_ll(3));
T4 = T_DH(-ThighLength, 0,   0, theta_ll(4));
T5 = T_DH(-TibiaLength, 0,   0, theta_ll(5));
T6 = T_DH(0,   -pi/2,     0,   theta_ll(6));
T_LL=A1*T1*T2*T3*T4*T5*T6*R1*R2*A2
```

Figura 54. Código para el cálculo de la cinemática directa de la pierna izquierda

Para obtener la posición de los efectores finales de cada cadena cinemática, se toman los valores de las 3 primeras filas de la cuarta columna de la matriz total de transformación de cada cadena cinemática. Se desarrolló la función “Cinemática” en la cual, para simplificar los códigos anteriormente descritos, se tomaron las ecuaciones que describen la posición de cada efector final de sus respectivas cadenas cinemáticas, tal como se presenta en la siguiente figura:

```
L1 = 53.9;
L2 = 67.9;
L3 = NeckOffsetZ;
% posicion del efector camara superior
Posiciones(3,1) = L1*cos(HeadYaw)*cos(HeadPitch) + L2*cos(HeadYaw)*sin(HeadPitch);
Posiciones(3,2) = L1*cos(HeadPitch)*sin(HeadYaw) + L2*sin(HeadYaw)*sin(HeadPitch);
Posiciones(3,3) = L3 + L2*cos(HeadPitch) - L1*sin(HeadPitch);
```

Figura 55. Código simplificado para calcular la posición del efector final de la cabeza

4.1.2. Cinemática Inversa

Se desarrolló la función “CinemáticaI”, en la cual se programaron y probaron las fórmulas detalladas en el capítulo 2 sección 4. Por lo tanto, solo se realizó el cálculo y programación de la cinemática inversa para las cadenas cinemáticas de la cabeza, brazo derecho y pierna derecha. El desarrollo para cada una de las cadenas cinemáticas anteriormente mencionadas, se detalla a continuación:

- Cinemática inversa de la cabeza:

```
L1 = 53.9;
L2 = 67.9;
L3 = NeckOffsetZ;
x1 = Angulos(11);           % HeadYaw
x2 = Angulos(12);           % HeadPitch
Px=Pos(3,1);
Py=Pos(3,2);
Pz=Pos(3,3);
HeadPitch = asin((L2*(L1^2+ L2^2- L3^2+ 2*L3*Pz - Pz^2 )^(1/2)- L1*Pz + L1*L3)/(L1^2+ L2^2 ));
HeadYaw = asin(Py/(L1*cos(x2)+ L2*sin(x2)));
```

Figura 56. Código de Matlab para calcular la cinemática inversa de la cabeza

- Cinemática inversa del brazo izquierdo

```
x1 = Angulos(1);    % LShoulderPitch
x2 = Angulos(2);    % LShoulderRoll
x3 = Angulos(3);    % LElbowYaw
x4 = Angulos(4);    % LElbowRoll
x5 = Angulos(5);    % LWristYaw

L1 = ShoulderOffsetY + ElbowOffsetY;
L2 = ShoulderOffsetZ;
L3 = HandOffsetX;
L4 = HandOffsetZ;
L5 = UpperArmLength;
L6 = LowerArmLength;
```

CONTINÚA

```
LElbowRoll = real(-(pi-acos((L3^2+L4^2-d^2)/(2*L3*L4))));

L1 = UpperArmLength;
L2 = ElbowOffsetY;
LShoulderPitch = atan(((sin(x1))*(L2*cos(x2)+L1*sin(x2)))/( (cos(x1)*(L2*cos(x2)+L1*sin(x2)))));
LShoulderRoll = acos((L2*(L2*cos(x2)+L1*sin(x2))-L1*(-L1*cos(x2)+L2*sin(x2)))/(L2^2+L1^2));
LElbowYaw = atan(sin(x3)/cos(x3));
LWristYaw = atan(sin(x5)/cos(x5));
```

Figura 57. Código de Matlab calcular la cinemática inversa del brazo izquierdo.

- Cinemática inversa de la pierna derecha

```
x1 = Angulos(13);       % LHipYawPitch
x2 = Angulos(14);       % LHipRoll
x3 = Angulos(15);       % LHipPitch
x4 = Angulos(16);       % LKneePitch
x5 = Angulos(17);       % LAnklePitch
x6 = Angulos(18);       % LAnkleRoll

L1 = HipOffsetY;
L2 = HipOffsetZ;
L3 = FootHeight;
L4 = ThighLength;
L5 = TibiaLength;

L1 = ThighLength;
L2 = TibiaLength;
LKneePitch = real((pi-acos((L1^2+L2^2-d^2)/(2*L1*L2))))
LAnkleRoll = real(atan(((L2*cos(x5)+L1*cos(x4+x5))*sin(x6))/((L2*cos(x5)+L1*cos(x4+x6))*cos(x6))))
LAnklePitch = asin(-((-L2*(sin(x5))-L1*((sin(x5))*(cos(x4))+ (cos(x5))*(sin(x4))))*(L2+L1*cos(x4))
+L1*(sin(x4))*(L2*(cos(x5)) + L1*((cos(x5))*(cos(x4))- (sin(x5))*(sin(x4))))/((L1^2)*
((sin(x4))^2)+(L2+L1*(cos(x4)))^2))
LHipRoll = -pi/4 + acos(cos(x2))
xx2 = LHipRoll;
LHipPitch = asin(((sin(x2))*(sin(x3)))/(sin(xx2+pi/4)))
LHipYawPitch = acos(((cos(x1))*(sin(x2)))/sin(xx2+pi/4))
```

Figura 58. Código de Matlab para calcular la cinemática inversa de la pierna izquierda

4.2. Centro de Masa del robot humanoide NAO

La posición de ubicación del CoM se calcula a cada instante que realice movimientos el robot NAO, si cambian los valores angulares de las cadenas cinemáticas, también cambia el CoM. En la (Tabla 6), se presentan los valores de posición del centro de masa y masa de cada eslabón, de acuerdo a cada articulación y con posiciones absolutas ya

que tienen su propio sistema de referencia. Por lo tanto, para poder determinar las posiciones de cada CoM, es necesario trasladarlas al sistema de referencia del torso que es el sistema base de las cadenas cinemáticas.

Previo a la resolución del cálculo del centro de masa, primero se trasladarán las posiciones del centro de masa de cada eslabón al sistema de referencia del torso, para lo cual se realizarán múltiples cadenas cinemáticas correspondientes a cada articulación, y se tomará la posición de cada centro de masa como efector final de su correspondiente cadena cinemática. Se usarán los parámetros DH descritos en el capítulo 2 como parámetros para las 25 cadenas cinemáticas que corresponderán a cada articulación incluyendo la del torso, aparte de las matrices de transformación se usaron matrices de rotación y traslación.

Se usa el cálculo de cinemática directa para cada articulación para determinar la posición de cada centro de masa como efector final con referencia al sistema de coordenadas del torso, para lo cual se desarrollaron las siguientes ecuaciones que describen dicho proceso para obtener la matriz de transformación de cada centro de masa del robot NAO.

- Cadena cinemática de la cabeza
    - Matriz de transformación para la articulación HeadYaw.

$$\boldsymbol{T_{CoM} = A(0, 0, NeckOffsetZ) * T_0^1 * A(P_{comX}, P_{comY}, P_{comZ})} \quad (58)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (-6.7379,\ 0, -27.42)$.

- Matriz de transformación para la articulación HeadPitch.

$$\boldsymbol{T_{CoM} = A(0, 0, NeckOffsetZ) * T_0^1 * A(P_{comX}, P_{comY}, P_{comZ})} \quad (59)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (-1.12, 0, 52.8)$

- Cadena Cinemática del brazo derecho
  - Matriz de transformación para la articulación RShoulderPitch.

$$\boldsymbol{T_{CoM} = A(0, -\mathbf{ShoulderOffsetY} - \mathbf{ElbowOffsetY}, \mathbf{ShoulderOffsetZ}) * T_0^1 * Rx(-\frac{\pi}{2}) * A(P_{comX}, P_{comY}, P_{comZ})} \quad (60)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (-1.65, 26.63, 0.14)$

  - Matriz de transformación para la articulación RShoulderRoll.

$$\boldsymbol{T_{CoM} = A(0, -\mathbf{ShoulderOffsetY} - \mathbf{ElbowOffsetY}, \mathbf{ShoulderOffsetZ}) * T_0^1 * T_1^2 * Rz(-\frac{\pi}{2}) * A(P_{comX}, P_{comY}, P_{comZ})} \quad (61)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (\ 24.55, -5.63, 3.3)$

  - Matriz de transformación para la articulación RElbowYaw.

$$\boldsymbol{T_{CoM} = A(0, -\mathbf{ShoulderOffsetY} - \mathbf{ElbowOffsetY}, \mathbf{ShoulderOffsetZ}) * T_0^1 * T_1^2 * T_2^3 * Rx(-\frac{\pi}{2}) * Rz(-\frac{\pi}{2}) * A(P_{comX}, P_{comY}, P_{comZ})} \quad (62)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (\ -27.44, 0, -0.14)$

- Matriz de transformación para la articulación RElbowRoll.

$$\boldsymbol{T_{CoM}} = \boldsymbol{A}(\mathbf{0}, -\mathbf{ShoulderOffsetY} - \mathbf{ElbowOffsetY}, \mathbf{ShoulderOffsetZ}) * \boldsymbol{T_0^1} * \boldsymbol{T_1^2} * \boldsymbol{T_2^3} * \boldsymbol{T_3^4} * \boldsymbol{Rz}(-\frac{\boldsymbol{\pi}}{\mathbf{2}}) * \boldsymbol{A}(\boldsymbol{P_{comX}}, \boldsymbol{P_{comY}}, \boldsymbol{P_{comZ}}) \quad (63)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (25.56, -2.81, 0.76)$

- Matriz de transformación para la articulación RWristYaw.

$$\boldsymbol{T_{CoM}} = \boldsymbol{A}(\mathbf{0}, -\mathbf{ShoulderOffsetY} - \mathbf{ElbowOffsetY}, \mathbf{ShoulderOffsetZ}) * \boldsymbol{T_0^1} * \boldsymbol{T_1^2} * \boldsymbol{T_2^3} * \boldsymbol{T_3^4} * \boldsymbol{T_4^5} * \boldsymbol{Rx}(-\frac{\boldsymbol{\pi}}{\mathbf{2}}) * \boldsymbol{Rz}(-\frac{\boldsymbol{\pi}}{\mathbf{2}}) * \boldsymbol{A}(\boldsymbol{P_{comX}}, \boldsymbol{P_{comY}}, \boldsymbol{P_{comZ}}) \quad (64)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (34.34, 0.88, 3.08)$

- Cadena Cinemática del brazo izquierdo

- Matriz de transformación para la articulación LShoulderPitch.

$$\boldsymbol{T_{CoM}} = \boldsymbol{A}(\mathbf{0}, \mathbf{ShoulderOffsetY} + \mathbf{ElbowOffsetY}, \mathbf{ShoulderOffsetZ}) * \boldsymbol{T_0^1} * \boldsymbol{Rx}(-\frac{\boldsymbol{\pi}}{\mathbf{2}}) * \boldsymbol{A}(\boldsymbol{P_{comX}}, \boldsymbol{P_{comY}}, \boldsymbol{P_{comZ}}) \quad (65)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (-1.65, -26.63, 0.14)$

- Matriz de transformación para la articulación LShoulderRoll.

$$\boldsymbol{T_{CoM}} = \boldsymbol{A}(\mathbf{0}, \mathbf{ShoulderOffsetY} + \mathbf{ElbowOffsetY}, \mathbf{ShoulderOffsetZ}) * \boldsymbol{T_0^1} * \boldsymbol{T_1^2} * \boldsymbol{Rz}(\frac{\boldsymbol{\pi}}{\mathbf{2}}) * \boldsymbol{A}(\boldsymbol{P_{comX}}, \boldsymbol{P_{comY}}, \boldsymbol{P_{comZ}}) \quad (66)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (\ 24.55, 5.63, 3.3)$

- Matriz de transformación para la articulación LElbowYaw.

$$\boldsymbol{T_{CoM} = A(0, ShoulderOffsetY + ElbowOffsetY,\ ShoulderOffsetZ) * T_0^1 * T_1^2 * T_2^3 * Rx(\frac{\pi}{2}) * Rz(\frac{\pi}{2}) * Rz(-\frac{\pi}{2}) * A(P_{comX}, P_{comY}, P_{comZ})} \quad (67)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (\ -27.44, 0, -0.14)$

- Matriz de transformación para la articulación LElbowRoll.

$$\boldsymbol{T_{CoM} = A(0, ShoulderOffsetY + ElbowOffsetY,\ ShoulderOffsetZ) * T_0^1 * T_1^2 * T_2^3 * T_3^4 * Rz(\frac{\pi}{2}) * A(P_{comX}, P_{comY}, P_{comZ})} \quad (68)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (\ 25.56, 2.81, 0.76)$

- Matriz de transformación para la articulación LWristYaw.

$$\boldsymbol{T_{CoM} = A(0, -ShoulderOffsetY - ElbowOffsetY,\ ShoulderOffsetZ) * T_0^1 * T_1^2 * T_2^3 * T_3^4 * T_4^5 * Rx(\frac{\pi}{2}) * Rz(\frac{\pi}{2}) * A(P_{comX}, P_{comY}, P_{comZ})} \quad (69)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (\ 34.34, -0.88, 3.08)$

- Cadena Cinemática de la pierna izquierda

- Matriz de transformación para la articulación LhipYawPitch.

$$\boldsymbol{T_{CoM} = A(0,\quad HipOffsetY, -HipOffsetZ) * T_0^1 * Ry\left(\frac{\pi}{4}\right) * Rz\left(\frac{\pi}{2}\right) * Rx(\frac{\pi}{2}) * A(P_{comX}, P_{comY}, P_{comZ})} \quad (70)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (-7.81, 11.14, 26.61)$

- Matriz de transformación para la articulación LHipRoll.

$$\boldsymbol{T_{CoM} = A(0,\quad HipOffsetY, -HipOffsetZ) * T_0^1 * T_1^2 * Rz(\pi) * Ry(-\frac{\pi}{2}) * A(P_{comX}, P_{comY}, P_{comZ})} \quad (71)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (\ -15.49, -0.29, -5.15)$

- Matriz de transformación para la articulación LHipPitch.

$$\boldsymbol{T_{CoM} = A(0,\quad HipOffsetY, -HipOffsetZ) * T_0^1 * T_1^2 * T_2^3 * Ry(\frac{\pi}{2}) * Rz(\frac{\pi}{2}) * A(P_{comX}, P_{comY}, P_{comZ})} \quad (72)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (\ 1.38, -2.21, -53.73)$

- Matriz de transformación para la articulación LKneePitch.

$$\boldsymbol{T_{CoM} = A(0,\quad HipOffsetY, -HipOffsetZ) * T_0^1 * T_1^2 * T_2^3 * T_3^4 * Rz\left(\frac{\pi}{2}\right) * Rx\left(\frac{\pi}{2}\right) * A(P_{comX}, P_{comY}, P_{comZ})} \quad (73)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (\ 4.53, -2.25, -49.36)$

- Matriz de transformación para la articulación LAnklePitch.

$$\boldsymbol{T_{CoM} = A(0,\quad HipOffsetY, -HipOffsetZ) * T_0^1 * \quad {}_1^2 * T_2^3}$$
$$\boldsymbol{* T_3^4 * T_4^5 * Rz\left(\frac{\pi}{2}\right) * Rx\left(\frac{\pi}{2}\right)} \quad (74)$$
$$\boldsymbol{* A(P_{comX}, P_{comY}, P_{comZ})}$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (0.45, -0.29, 6.85)$

- Matriz de transformación para la articulación LAnkleRoll.

$$\boldsymbol{T_{CoM} = A(0,\quad HipOffsetY, -HipOffsetZ) * T_0^1 * T_1^2 * T_2^3}$$
$$\boldsymbol{* T_3^4 * T_4^5 * T_5^6 * Rz(\pi) * Ry\left(-\frac{\pi}{2}\right)} \quad (75)$$
$$\boldsymbol{* A(P_{comX}, P_{comY}, P_{comZ})}$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (25.42, -3.3, -32.39)$

- Cadena Cinemática de la pierna derecha
  - Matriz de transformación para la articulación RHipYawPitch.

$$\boldsymbol{T_{CoM} = A(0, -HipOffsetY, -HipOffsetZ) * T_0^1 * Ry\left(\frac{\pi}{4}\right)}$$
$$\boldsymbol{* Rz\left(\frac{\pi}{2}\right) * A(P_{comX}, P_{comY}, P_{comZ})} \quad (76)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (-7.81, -11.14, 26.61)$

  - Matriz de transformación para la articulación RHipRoll.

$$\boldsymbol{T_{CoM} = A(0, -HipOffsetY, -HipOffsetZ) * T_0^1 * T_1^2 * Rz(\pi)}$$
$$\boldsymbol{* Ry(-\frac{\pi}{2}) * A(P_{comX}, P_{comY}, P_{comZ})} \quad (77)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (-15.49, 0.29, -5.15)$

  - Matriz de transformación para la articulación RHipPitch.

$$\boldsymbol{T_{CoM} = A(0, -HipOffsetY, -HipOffsetZ) * T_0^1 * \ {}^2_1 * T_2^3 * Ry(\frac{\pi}{2}) * Rz\left(\frac{\pi}{2}\right) * A(P_{comX}, P_{comY}, P_{comZ})} \quad (78)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (\ 1.38, 2.21, -53.73)$

- Matriz de transformación para la articulación RKneePitch.

$$\boldsymbol{T_{CoM} = A(0, -HipOffsetY, -HipOffsetZ) * T_0^1 * T_1^2 * T_2^3 * T_3^4 * Rz\left(\frac{\pi}{2}\right) * Rx\left(\frac{\pi}{2}\right) * A(P_{comX}, P_{comY}, P_{comZ})} \quad (79)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (\ 4.53, 2.25, -49.36)$

- Matriz de transformación para la articulación RAnklePitch.

$$\boldsymbol{T_{CoM} = A(0, -HipOffsetY, -HipOffsetZ) * T_0^1 * T_1^2 * T_2^3 * T_3^4 * T_4^5 * Rz\left(\frac{\pi}{2}\right) * Rx\left(\frac{\pi}{2}\right) * A(P_{comX}, P_{comY}, P_{comZ})} \quad (80)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (0.45, 0.29, 6.85)$

- Matriz de transformación para la articulación RAnkleRoll.

$$\boldsymbol{T_{CoM} = A(0, -HipOffsetY, -HipOffsetZ) * T_0^1 * T_1^2 * T_2^3 * T_3^4 * T_4^5 * T_5^6 * Rz(\pi) * Ry\left(-\frac{\pi}{2}\right) * A(P_{comX}, P_{comY}, P_{comZ})} \quad (81)$$

En donde $(P_{comX}, P_{comY}, P_{comZ}) = (\ 25.42, 3.3, -32.39)$

Para el desarrollo del cálculo del CoM se desarrolló la función “PosicionCoM”, la cual tiene como parámetros de entrada los valores angulares de cada articulación y retorna la posición del CoM del todo el robot, descrito con respecto al sistema de referencia ubicado en el torso del robot. Para calcular la posición del ZMP se reemplaza el valor de la

posición del eje z por la posición del eje z del efector final de la cadena cinemática de cualquiera pierna que se encuentre como apoyo.

4.3. Simulación del Robot NAO

4.3.1. Configuración del entorno de simulación

La simulación del robot NAO se la realizó usando el software de simulación de robots V-Rep, con el cual mediante del uso de una API remota se realiza el control del robot desde un script desarrollado en el software de Matlab, para hacer el uso de las funciones y de la API Remota que provee V-Rep, es necesario instalar los plugin y copiar las librerías en la carpeta que se encuentre el script de control, las librerías que se usan son:

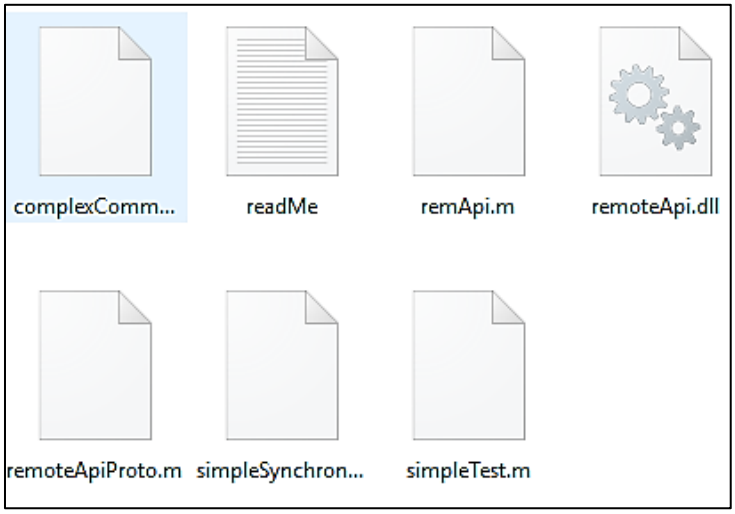


Figura 59. Librerías para conexión con la API Remota de V-Rep

Para realizar la conexión entre Matlab y V-Rep se requiere habilitar un puerto de conexión en un script de V-Rep, en el cual se establece el puerto de conexión por el cual se realizará la conexión por medio de la API Remota, tal como se indica en la siguiente figura:

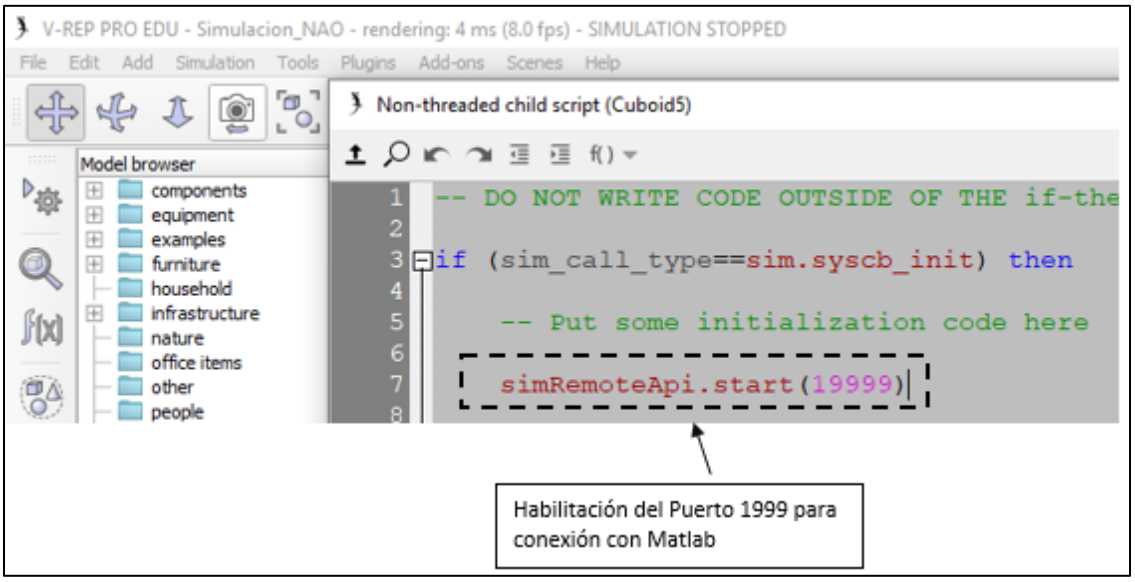


Figura 60. Configuración del puerto de conexión en V-Rep

El puerto para conectarse a V-Rep usando la API Remota es el 19999, este puerto se especifica en el script de control, para que se realice la conexión también se deben de configurar una serie de parámetros de la conexión, establecer una conexión y solicitar las direcciones articulares, tal como se muestra en el siguiente diagrama de flujo:

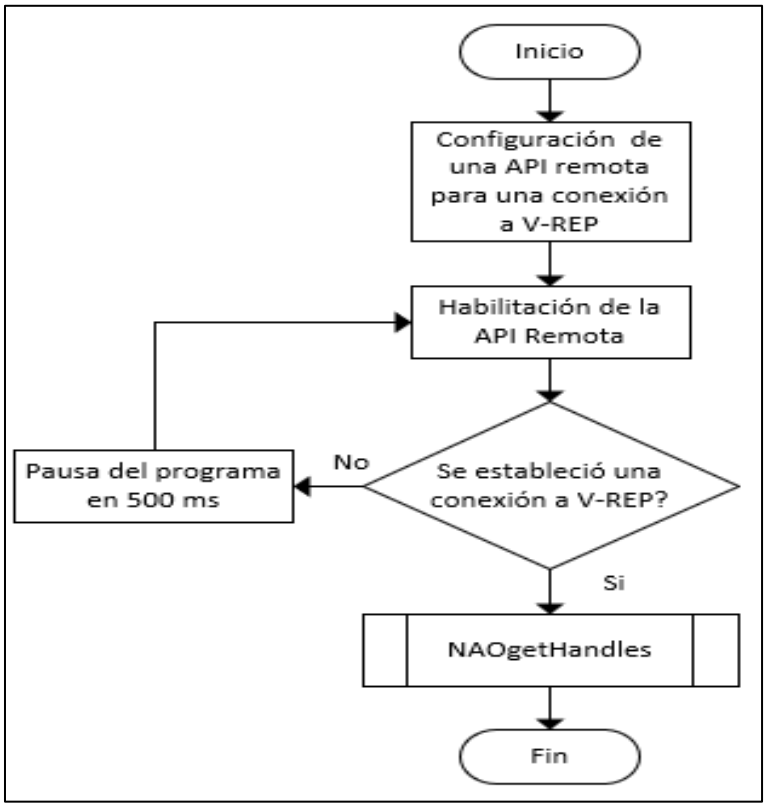


Figura 61. Diagrama de flujo de conexión con V-Rep

El primer paso a configurar en la conexión es la creación de un objeto de tipo cliente usando la función “remApi”, para que esta función se ejecute se requiere que los archivos de la (Figura 59) se encuentren en la carpeta en la que se encuentre el script de control, después se procede a configurar la conexión entre Matlab y V-Rep usando la función “simxStart”, que tiene los siguientes parámetros:

- IPaddres: Dirección IP en la que se encuentra el servidor, V-Rep trabajará como un servidor, en la presente tesis la conexión entre Matlab y V-Rep se la realiza en la misma computadora, por lo cual se usó la dirección del LocalHost: 127.0.0.1.
- Port: Dirección del puerto especificado en el script de V-Rep, usando la función: “simRemoteApi.start”, para el desarrollo de la simulación se usó el puerto 19999.

- waitUntilConnected: Si es igual a “true”, mantiene en espera a la función “simxStart” hasta que se pueda conectar al servidor.
- doNotReconnectOnceDisconnected: si es igual a “true”, en el caso de que se pierda la conexión el hilo no intentará reconectarse con el servidor.
- timeOutInMs: Establece un tiempo de conexión en milisegundos para el primer intento de conexión con el servidor, en este caso se estableció un tiempo de 500 milisegundos.
- commThreadCycleInMs: Establece la frecuencia de transferencia de datos entre el servidor y el cliente, se establece como valor predeterminado 5.
- clientID: La función “simxStart” retorna el valor de “-1” si no ha sido posible la conexión con el servidor.

NAOgetHandles (Cliente, vrep): Se desarrolló esta función para solicitar al simulador las direcciones de cada una de las articulaciones del robot.

El código en C desarrollado para realizar la conexión entre Matlab y V-Rep se presenta a continuación:

```
%% Definicion de una API remota para comunicarse con V-Rep, por el puerto 19999
vrep = remApi('remoteApi');                         % Creacion de un objeto cliente Vrep
clientID = vrep.simxStart('127.0.0.1',19999,true,true,5000,5) %Iniciar conexion con el simulador
while(clientID~=0)                                  % Espera a una conexion con V-Rep
    clientID = vrep.simxStart('127.0.0.1',19999,true,true,5000,5) % Configuracion de la conexion
    disp("Conectando con V-Rep");
    pause(0.5);
end
disp("Conectado a V-Rep");
JointVector = NAOgetHandles(clientID,vrep);         % Obtencion de las direcciones de las articulaciones
```

Figura 62. Configuración de la conexión de script Matlab con API remota a V-Rep

Por último, en la simulación se usa la librería NAO y se configura el encendido de una cámara, en la cual el software V-Rep accede a la cámara de la computadora en la que se encuentre, el entorno de simulación realizado se presenta en la siguiente figura:

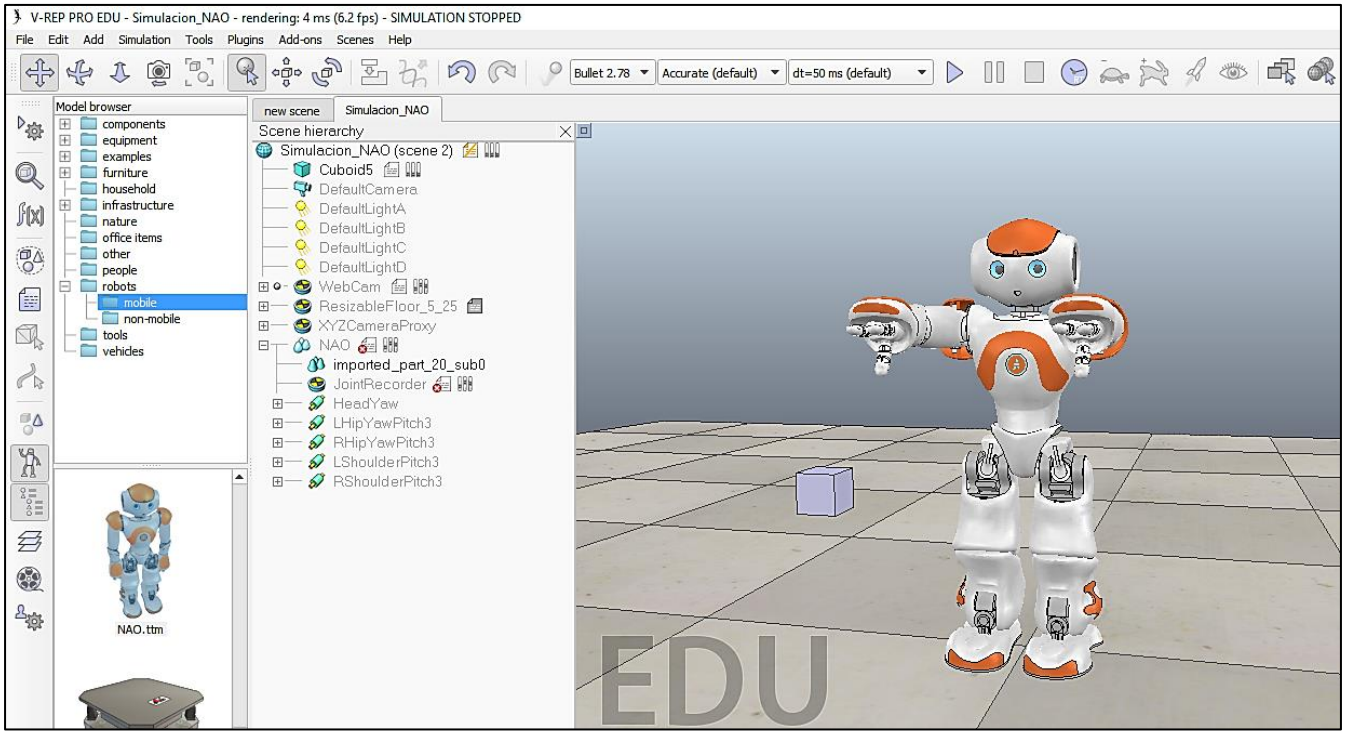


Figura 63. Simulación del robot Nao realizada en V-Rep

4.3.2. Control del robot NAO simulado

Se desarrolló la función “SimulacionNAO” que recibe como parámetros un vector con los 25 valores angulares para cada articulación, y retorna como valor un contador en el cual indica el número de fallos en el caso que haya habido problemas con la transferencia de datos, para transferir los datos de control de cada articulación al simulador se usó la función: simxSetJointTargetPosition

Los parámetros que usa la función son:

- Number ClientID: Identificación del cliente, se obtiene a partir de la creación de la conexión usando la función “simxStart”, en el código desarrollado es “ClientID”.
- Number jointHandle: Dirección de la articulación en la cual se va a realizar el control, esta dirección se obtiene de la función getHandles.
- Number targetPosition: Valor angular en radianes para el control de la articulación seleccionada en el anterior literal.
- Number operationMode: Selecciona el modo de operación, se usó el modo “simx_opmode_oneshot”, la cual envía el comando al servidor y no espera una respuesta real.
- NumberreturnCode: La función simxSetJointTargetPosition retornar el valor de 1 si se suscitó un error en la transferencia de datos.

El código en C realizado para transferir los datos de los valores angulares de las articulaciones de la cabeza se presenta en la siguiente figura:

```
%% Codigo para transferir los valores angulares de la cabeza
[contador] = vrep.simxSetJointTargetPosition(clientID,HeadYaw,Angulos(1,11),vrep.simx_opmode_oneshot); % HeadYaw
[contador] = vrep.simxSetJointTargetPosition(clientID,HeadPitch,Angulos(1,12),vrep.simx_opmode_oneshot); % HeadPitch
```

Figura 64. Código de control del movimiento en articulaciones de cabeza en robot

El proceso se realiza de la misma forma para las demás cadenas cinemáticas, en las cuales solo se cambiarán los valores de dirección de la articulación “jointHandle” y el valor angular “targetPosition”.

CAPÍTULO V

IMPLEMENTACIÓN DEL SISTEMA DE IMITACIÓN DE MOVIMIENTOS

Para la conexión entre el software Matlab y el controlador de la red de sensores, se levantó un servidor TCP usando el puerto 80 y el cual se mantiene en espera de que el cliente que es la red de IMU's se conecte al servidor. A su vez, se implementó la configuración y conexión de la API Remota para conectarse a V-Rep el cual se desarrolló en capítulo 4 sección 2.

Se implementó el cálculo de la cinemática directa para calcular las posiciones de los efectores de cada cadena cinemática, el cual se describió en el capítulo 4 sección 1. Después del cálculo cinemático se calculó la posición del CoM de cada articulación del robot para luego calcular el CoM general del sistema el cual se desarrolló y describió en el capítulo 4 sección 2.

La programación del robot Humanoide NAO se la realizó desde un script usando código en Python versión 2.7, el cual se programa usando el compilador Ninja IDE v2.7 e importando la librería "NAOqi", previo a su importación se requiere que se instale NAOqi SDK para Python. Se levantó y configuró un servidor TCP en Matlab para la conexión con el script de Python en la cual se realizará la transferencia de datos de los valores articulares desde Matlab para que el robot NAO ejecute los movimientos articulares requeridos.

5.1. Integración de la red de sensores con el robot NAO simulado

5.1.1. Configuración y conexión

Como primer paso se realiza la declaración de variables globales, ya que al usarse en la mayoría de scripts realizados se logra evitar su constante declaración.

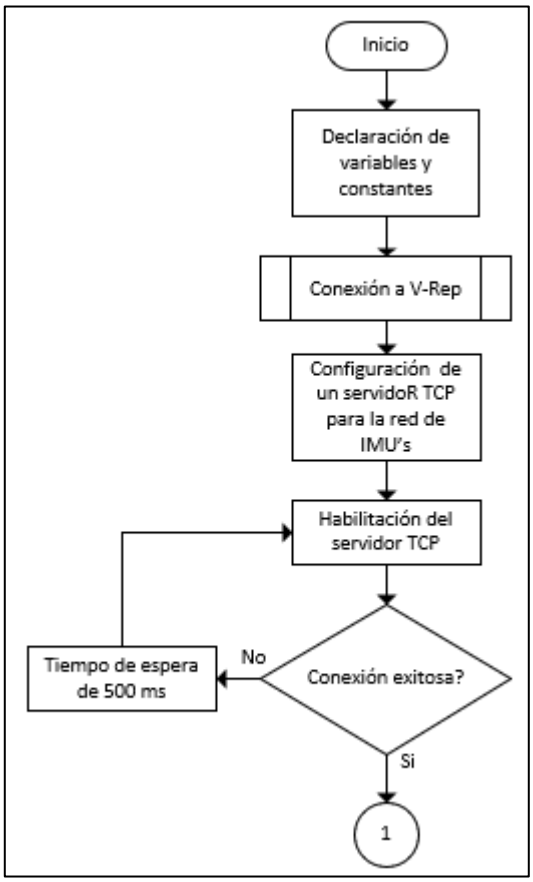


Figura 65. Diagrama de flujo de conexión con V-Rep y red de IMU's

La declaración de variables incluye valores como: medidas de cada eslabón, tabla de posiciones de CoM absolutos y sus respectivas masas, cantidad de sensores y valores de offset. La función "Conexión a V-Rep" se la realizó en base a lo descrito en el capítulo 4 sección 3, una vez establecida la conexión con V-Rep se procede a realizar la

configuración del servidor TCP para la conexión con la red de sensores, para lo cual se usa la función de Matlab “tcpip”.

```
wifi = tcpip('0.0.0.0', 80, 'NetworkRole', 'server');
```

Como primer valor a configurar en la función “tcpip” se configura la dirección IP del host o cliente que se conectará, en el caso de que se realice una conexión interna se usa como valor “localhost”, en el caso de que sea una conexión externa se especifica la dirección IP del cliente o se escribe la dirección “0.0.0.0”, la cual solo admite al primer cliente que intente una conexión. Después se especifica el puerto que se va a usar para la conexión entre la red de IMU’s y el servidor TCP de Matlab. Seguidamente se ingresa el valor “NetworkRole” para habilitar el uso de Server Sockets de Matlab que permite poder trabajar como cliente o servidor a Matlab y por último se especifica el tipo de trabajo que realizara Matlab, que puede ser “client” o “server”. Al ejecutarse la función se levanta el servidor y se mantiene en espera de conexión de un cliente cada 500 ms.

5.1.2. Recepción y procesamiento de mensajes

Establecida la conexión con V-Rep y la red de IMU’s, el programa se mantiene a la espera de un mensaje que sea diferente de nulo, del cliente de la red de IMU’s que envía los valores angulares de los 16 sensores en un solo vector, mientras que no exista alguna desconexión con el cliente o con el servidor, tal como se indica en el siguiente diagrama de flujo:

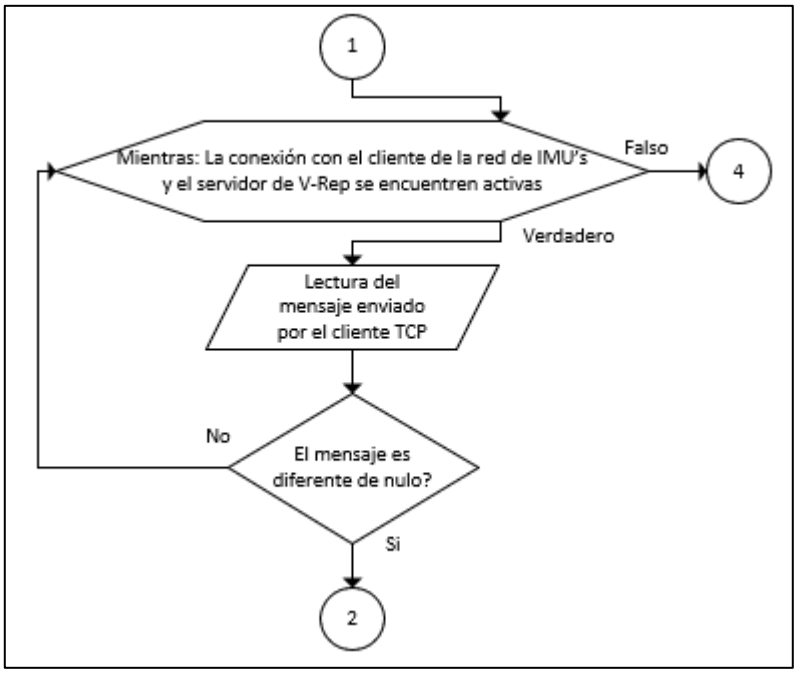


Figura 66. Diagrama de flujo de lectura de datos

Al recibirse un mensaje completo y no nulo, se termina el bucle de espera de recepción de mensajes, el mensaje recibido es un dato de tipo String, por lo cual primero se procedió a separar al mensaje e ir transformándolo a datos de tipo Double y finalmente distribuirlo en una matriz de datos angulares de todos los IMU's, para terminar de procesar el mensaje recibido, a la matriz con los valores angulares se la distribuye de acuerdo a los valores de orientación de Roll, Pitch y Yaw, dejando finalmente 3 matrices para cada valor de orientación, como último paso se calcula la media de los 10 primeros datos medidos de los 16 IMU's para calcula el offset de estos datos, para lo cual es necesario que la persona que se encuentre usando el prototipo de la red de sensores se mantenga en la posición de referencia. El diagrama de flujo realizado se presenta a continuación:

Figura 67. Diagrama de flujo del procesamiento del mensaje recibido

5.1.3. Cálculo cinemático, CoM y control del robot

Se desarrolló la función “Condiciones de ángulos”, la cual se basa en asignar cada valor de orientación medida con los valores articulares del robot NAO y se los limita de acuerdo a los rangos de movilidad de cada articulación del robot NAO, descritos en el capítulo 2, finalmente se asignan todos los valores articulares en un vector para poder retórnalos al programa principal. Con los valores articulares ya limitados y procesados, se los envía usando la función desarrollada “Simulación NAO”, la cual se basa en enviar todos los valores articulares usando la API Remota a V-Rep para que el robot NAO realice sus movimientos de acuerdo a los datos recibidos. El diagrama de flujo realizado se presenta en la siguiente figura:

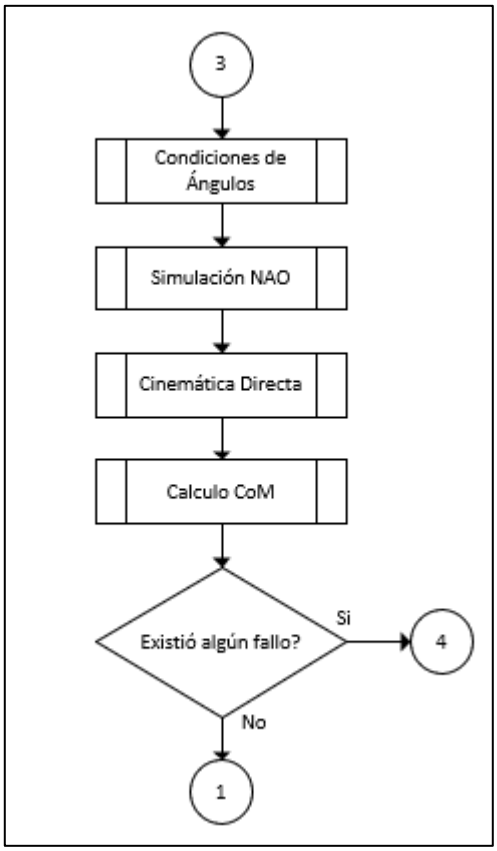


Figura 68. Diagrama de flujo de etapa de procesamiento de datos y control del robot

La función “Cinemática Directa” recibe los valores angulares de cada articulación y realiza los cálculos de posición para cada efector de su correspondiente cadena cinemática, al finalizar sus cálculos retorna una matriz con los valores de posición de cada efector, por último, se realiza el cálculo del CoM general del sistema para analizar la estabilidad del robot.

Las funciones desarrolladas para el programa principal se detallan a continuación:

- La función “Condición ángulos”: Se basa en primero realizar un escalamiento de ángulos, entre los cuales determinar los valores de Yaw para las articulaciones que lo necesitan, escalados los valores se realiza la asignación de cada valor angular

medido a los valores 24 valores articulares del robot NAO, detallaros en la (Tabla 3). cada articulación del robot nao tiene sus limitaciones de movilidad, las cuales se detallaron en el capítulo 2 sección 3, por lo cual, cada valor articular se comprueba que cumpla con los grados de movilidad de cada articulación y se los limita en caso de que exceda dichos limites, finalmente a los datos se los procede a asignar en un vector que contenga los 25 valores articulares para poder ser retornado al programa principal, el diagrama de flujo se presenta en la siguiente figura:

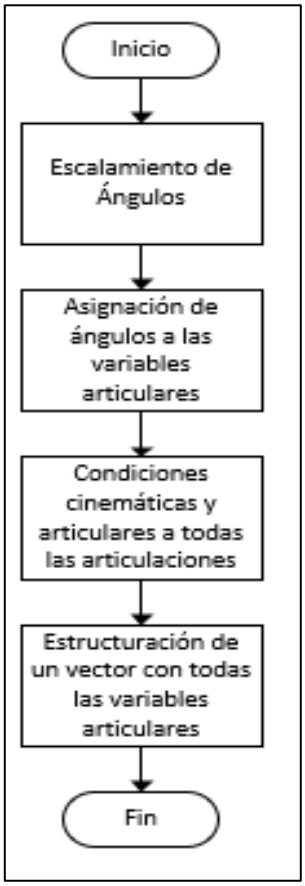


Figura 69. Condición de ángulos

La etapa de asignación de ángulos a las variables articulares, se la realizó en base a la siguiente tabla, en donde se presenta las articulaciones del robot NAO y el valor angular que se le asigna dependiendo de los valores Roll, Pitch o Yaw y del sensor al que corresponde su lectura:

Tabla 16
Valores angulares para cada articulación

Ítem	Articulación	Valor
1	RShoulderPitch	-Pitch (103)
2	RShoulderRoll	Yaw (103) - Yaw (309)
3	RElbowYaw	Roll (102)
4	RElbowRoll	Yaw (102) - Yaw (103)
5	RWristYaw	Roll (1,101)
6	LShoulderPitch	-Pitch (207)
7	LShoulderRoll	-(-Yaw (207) + Yaw (309))
8	LElbowYaw	Roll (206)
9	LElbowRoll	Yaw (206) - Yaw (207)
10	LWristYaw	Roll (1,205)
11	HeadYaw	Yaw (310) - Yaw (309)
12	HeadPitch	Pitch (310)
13	RHipYawPitch	Yaw (411)-Yaw (309)
14	RHipRoll	-Pitch (413)
15	RHipPitch	-Roll (413)
16	RKneePitch	-Roll (412) +Roll (413)
17	RAnklePitch	-Pitch (411)
18	RAnkleRoll	Roll (411)
19	LHipYawPitch	Yaw (514)-Yaw (309)
20	LHipRoll	Pitch (516)
21	LHipPitch	Roll (516)
22	LKneePitch	Roll (512) +Roll (513)
23	LAnklePitch	-Pitch (511)
24	LAnkleRoll	Roll (511)

- La función “Cinemática Directa”: Inicialmente declara las variables globales y las asigna según corresponda, primero calcula la posición del efector de la cámara

superior de la cabeza, después calcula la posición de los efectores de las cadenas cinemáticas del brazo izquierdo, brazo derecho, pierna izquierda y pierna derecha, finalmente asigna todas las posiciones de los efectores en una matriz para poder retornarla al programa principal, el cálculo de posiciones de los efectores se lo realizó en base a lo detallado en el capítulo 4 sección 1. El diagrama de flujo desarrollado se presenta en la siguiente figura:

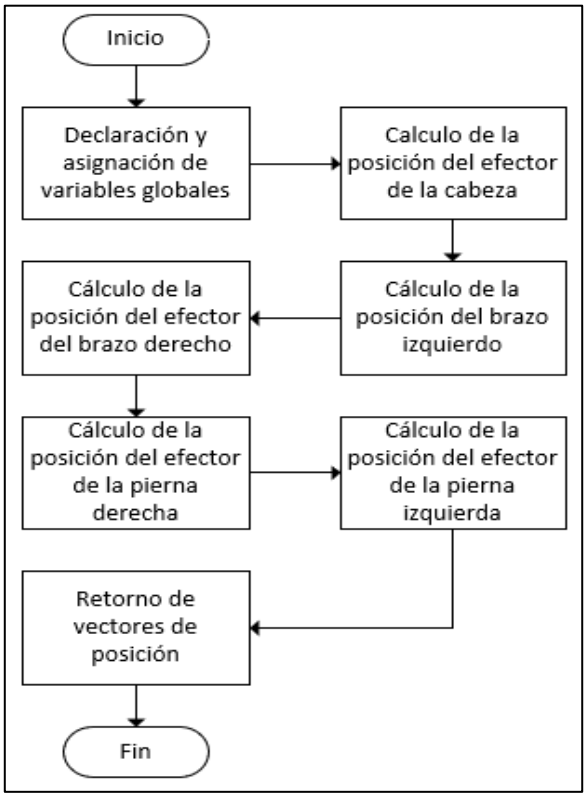


Figura 70. Función Cinemática directa

- Función “PosicionCoM”: inicialmente declara las variables globales y las asigna según corresponda, después se procede a calcular la cinemática directa de cada articulación y tomando como efector la posición del CoM absoluta de su eslabón, se

realiza este proceso para todas las articulaciones del robot, tal como se detalló en el capítulo 4 sección 3, finalmente se calcula la posición del CoM total del robot usando la ecuación (x), finalmente se asignan estos valores a un vector para luego poder ser retornados al programa final. El diagrama de flujo desarrollado se presenta en la siguiente figura:

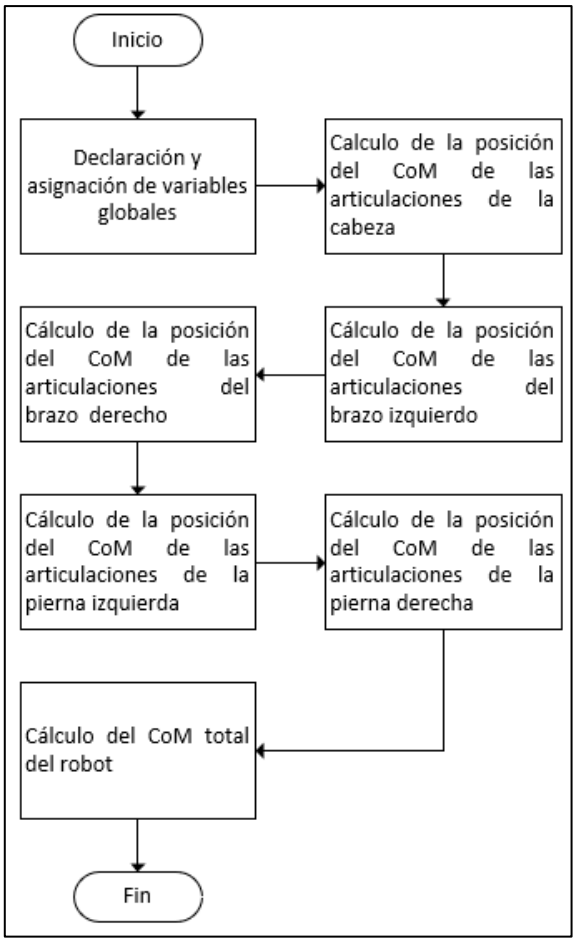


Figura 71. Función Posición CoM

- Función “Simulación de movimientos”: Inicialmente se declara las variables globales y las asigna según corresponda, en esta función recibe como parámetros los valores articulares y las direcciones de cada articulación para poder enviarlas usando la API

Remota a V-Rep, el proceso de transferencia de datos se lo realiza primero enviando los valores articulares de la cadena cinemática de la cabeza, después las de las cadenas cinemáticas del brazo izquierdo, derecho, pierna izquierda y derecha, el proceso de transferencia de datos se lo realiza de acuerdo a lo descrito en el capítulo 4 sección 3, de la misma forma en caso de que exista algún error en la transferencia de datos se contará el error y se retornará este valor al programa principal. El diagrama de flujo desarrollado se presenta en la siguiente figura:

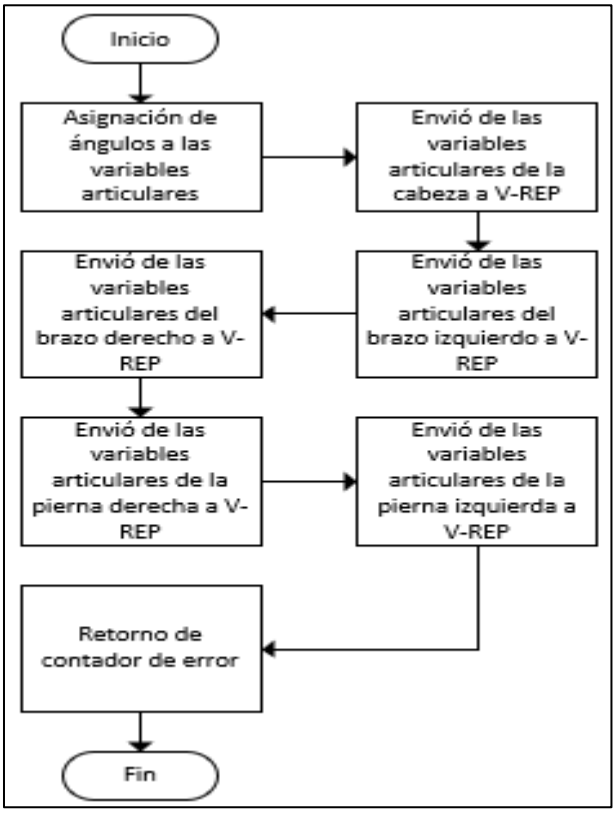


Figura 72. Función Simulación de movimientos

5.1.4. Configuración de fallos y finalización del programa

En el caso de que se presente algún fallo en el programa, sea por fallo en la transferencia de datos con V-Rep, con la red de sensores, fallo en algún procesamiento

de datos, esta etapa se encargará de finalizar el programa y terminar la conexión con el servidor de V-Rep y la conexión con el cliente de la red de IMU's, tal como se indica en el siguiente diagrama de flujo:

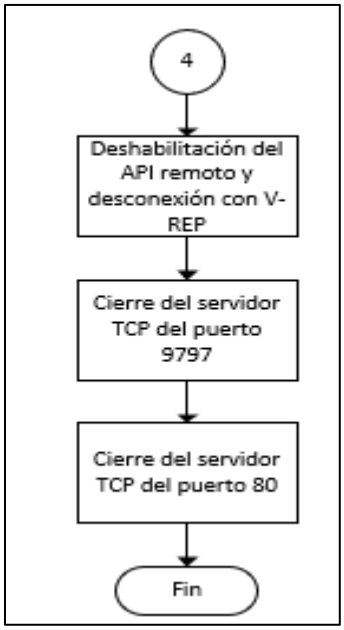


Figura 73. Configuración de fallos y fin del programa

5.2. Integración de la red de sensores con el robot NAO simulado y Físico

Previo al control del robot Nao desde el programa principal, se realizó el programa de control del robot Nao y su comunicación con el programa principal, para lo cual se detallan las siguientes secciones, para la realización del programa de control, se usó como lenguaje de programación Python, el cual se detalló en el capítulo 2 sección 4.

5.2.1. Configuración y conexión del programa de control del robot NAO

El control del robot NAO se realizó importando ALProxy de la librería NAOqi, de la cual se usaron sus métodos para poder usar las funciones del robot, seguidamente se

declaran las variables y constantes que se usaran durante la ejecución del programa, también se declara la dirección IP del robot NAO y el puerto al que se realizará la conexión del robot, en donde la dirección IP depende de la dirección que se le asigne en la red local a la que se conecte y el puerto de conexión es 9559. También se declara la dirección IP del servidor de Matlab para su conexión, en la cual, para el desarrollo de la presente tesis se usó la dirección del “Local Host”, y como puerto de conexión se usó el puerto 9797, tal como se indica en la siguiente figura:

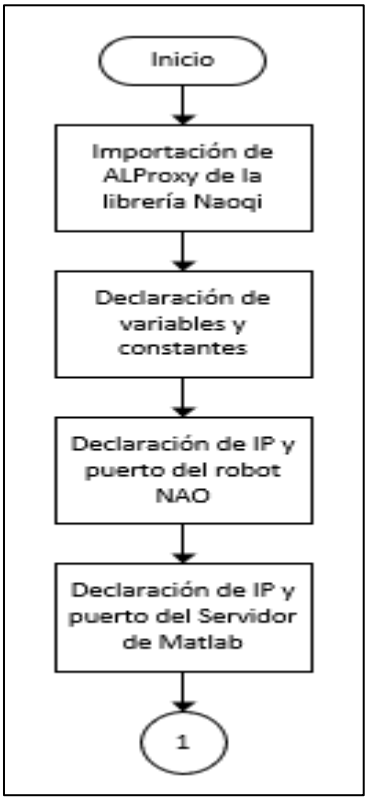


Figura 74. Etapa de configuración

Realizada la configuración detallada anteriormente, se procede a declarar los módulos y métodos de la librería declarada, para lo cual se realiza la conexión con el

robot NAO, para lo cual se usa el método “ALProxy”, y se declaran los módulos necesarios para el control del robot NAO, tales como: “ALTextToSpeech”, “ALRobotPosture” y “ALMotion”, tal como se indica en la siguiente figura:

```
tts = ALProxy('ALTextToSpeech', ipnao, puertonao)
postureProxy = ALProxy("ALRobotPosture", ipnao, puertonao)
motion = ALProxy('ALMotion', ipnao, puertonao)
```

Figura 75. Declaración de módulos del robot NAO

En donde “ipnao” es la dirección IP del robot nao, “puertonao” es la dirección del puerto del robot NAO, “ALProxy” es la librería, los métodos recibidos por lo módulos declarados se guardan en los objetos “tts”, “postureProxy” y “motion”. Finalmente se procede a posicionar al robot en su postura de referencia o HOME. La etapa anteriormente descrita se presenta en la siguiente figura:

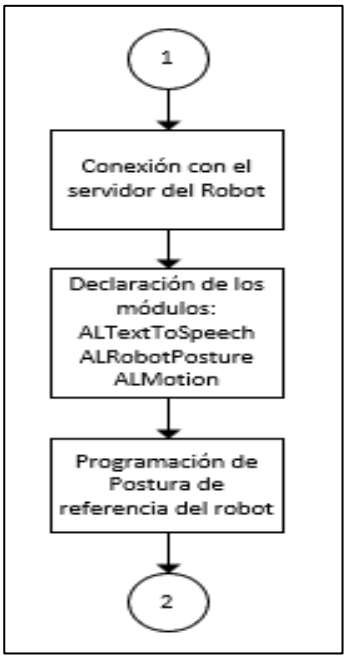


Figura 76. Etapa de configuración del robot

Finalmente se procede a realizar la conexión con el servidor de Matlab usando la dirección IP y el puerto declarado anteriormente, el programa se mantiene en un bucle hasta que se establezca la conexión con el servidor de Matlab, tal como se indica en la siguiente figura:

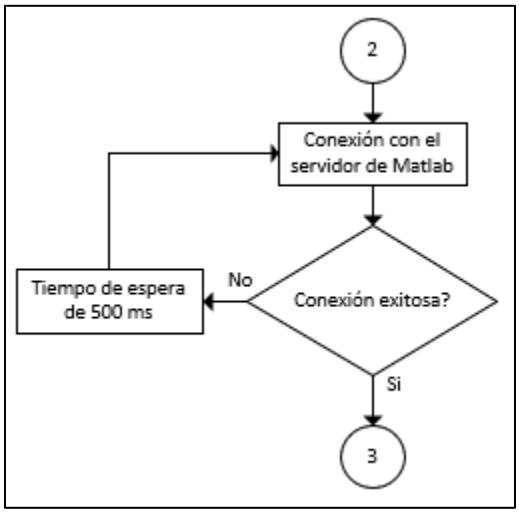


Figura 77. Etapa de conexión

5.2.2. Recepción y procesamiento de mensajes del programa de control del robot NAO

Una vez establecida la conexión con el servidor de Matlab y el servidor del robot NAO, el programa se mantendrá en espera de un mensaje que sea diferente de nulo, del servidor de Matlab, el cual envía los valores angulares de cada articulación en un solo vector, mientras que no exista alguna desconexión con los dos servidores o se reciba como mensaje la palabra “Fin”, tal como se indica en el siguiente diagrama de flujo:

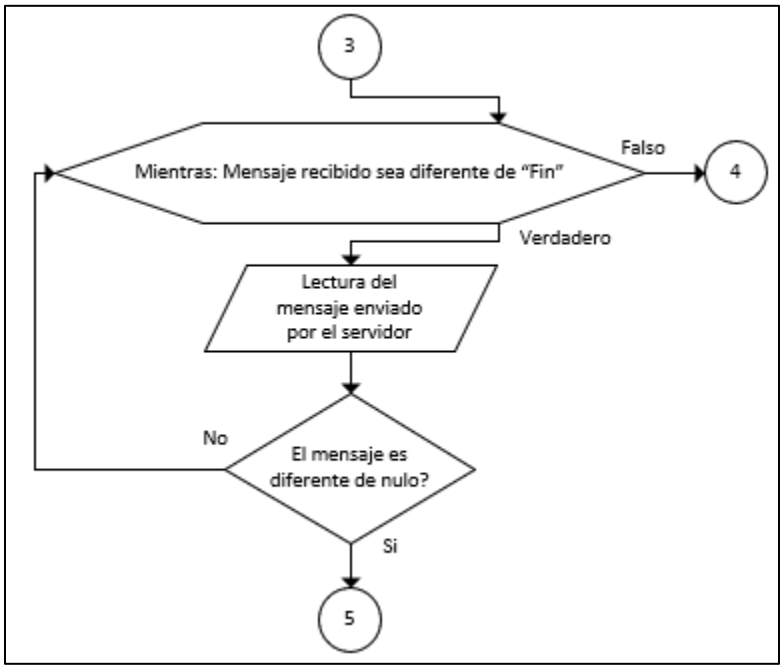


Figura 78. Etapa de lectura de datos

Al recibirse un mensaje completo y no nulo, se termina el bucle de espera de recepción de mensajes.

5.2.3. Control de movimientos del programa de control del robot NAO

El mensaje recibido del servidor de Matlab, recibe como parámetros un vector con los 24 valores angulares para cada articulación, los cuales son asignados a variables específicas de cada articulación, para luego proceder a enviarlas al robot y que este a su vez, pueda realizar los movimientos requeridos, dicho proceso se presenta en la siguiente figura:

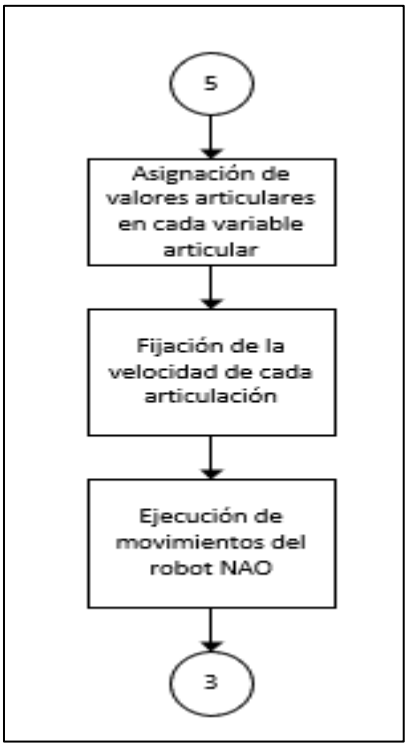


Figura 79. Etapa de control del robot

El método usado para que el robot NAO ejecute los movimientos definidos por los valores articulares es “setAngles”, el cual recibe como parámetros de entrada: las articulaciones que se moverán, en este caso se encuentra en el vector “nombres” el cual contiene los nombres de las 24 articulaciones del robot nao; el valor angular a ejecutar en radianes, en este caso es el vector “ángulos” el cual contienen los valores angulares de cada articulación, y por último, la velocidad en que se realizará el movimiento el cual varía entre 0 y 1, siendo 1 la máxima velocidad de movimiento y 0 la mínima velocidad, tal como se indica en la siguiente figura:

```
motion.setAngles(nombres, angulos, 1.0)
```

Figura 80. Módulo usado para la ejecución de movimientos

5.2.4. Configuración de fallos y finalización del programa de control del robot NAO

El programa finaliza su ejecución, solo si existe un fallo en la conexión con ambos servidores, o si se recibe el mensaje “Fin” del servidor de Matlab. En ambos casos el proceso a realizarse se detalla en la siguiente figura:

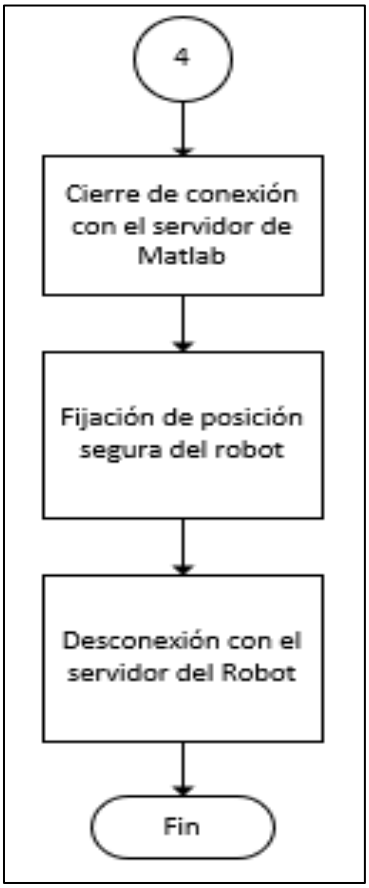


Figura 81. Etapa de fin del programa

El primer paso para terminar el programa es cerrar la conexión con el servidor de Matlab, ya que este a su vez se desactivará, para finalizar la conexión con el robot NAO, primero se lo posiciona en su posición de descanso para así evitar cualquier movimiento inesperado mientras se finaliza el programa, después de haberse posicionado el robot se procede a cerra la conexión con el robot NAO.

5.2.5. Control del robot NAO desde el programa principal

La conexión del programa principal con el programa en Python desarrollada para el control del robot NAO, se la realizó mediante el uso del protocolo de comunicación TCP/IP, de la misma forma en que se realizó la conexión con la red de IMU's. Para lo cual primero se configuró los parámetros del servidor, así como la dirección IP del servidor que es la del "LocalHost", y la dirección del puerto es 9797. Se usaron los mismos pasos de configuración que se realizaron en la sección 1 del presente capítulo. Al habilitarse el servidor para conexión con el cliente del programa en Python, se mantendrá en espera de que el cliente se conecte, tal como se indica en la siguiente figura:

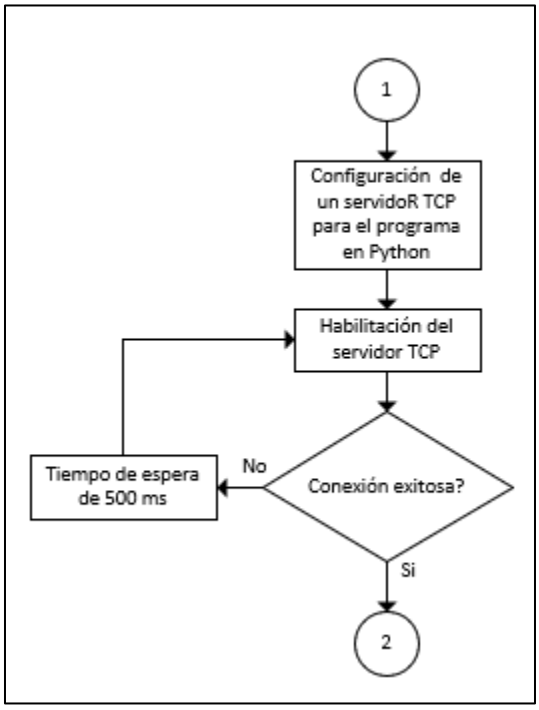


Figura 82. Etapa de conexión

La finalidad de establecer el servidor en Matlab en el programa principal, es permitir el envío de datos al programa que controla el robot NAO, ya que este se lo realiza

con el lenguaje de programación Python, mientras que el programa principal se encuentra realizado en lenguaje de programación C.

Se desarrolló la función “Movimiento NAO”, la cual se encarga de enviar los valores angulares de cada articulación al programa que controla al robot NAO, esta función se une al programa principal como parte final del programa, el cual se detalló en la sección 1 del presente capítulo, tal como se indica en la siguiente figura:

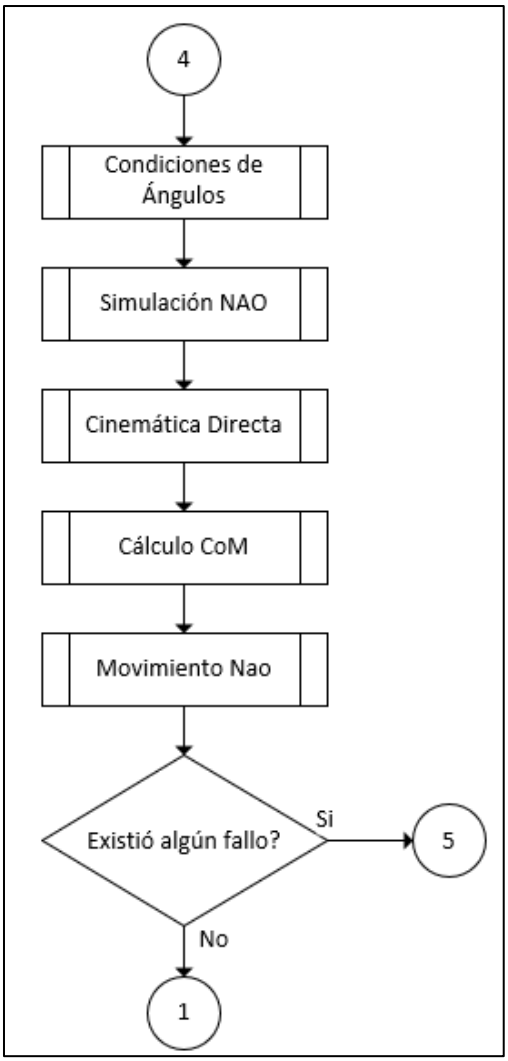


Figura 83. Etapa del control del robot

La función “Movimiento Nao” recibe como parámetros de entrada, un vector que contenga todos los valores angulares de las 24 articulaciones del robot, para luego poder ser transformadas en datos de tipo String y así poder juntarlas en un solo mensaje, con el mensaje listo se realiza él envió del mensaje al cliente que es el programa en Python. Esta función retorna al programa principal un contador de tipo entero, el cual indicará el número de fallos en el caso de que se hayan suscitado, en el caso de que se suscite algún fallo en la transmisión de datos, se envía un mensaje de “Fin” al programa en Python y se procede a finalizar el programa principal.

CAPÍTULO VI

PRUEBAS Y RESULTADOS

6.1. Descripción de las pruebas

La realización de pruebas y análisis del sistema desarrollado en la presente tesis, se las realiza de forma individual y en conjunto, de tal forma que se realizaron pruebas de cada sensor inercial de forma individual, de la simulación del movimiento del robot NAO y sus movimientos con el Robot NAO en físico. Posterior a las pruebas individuales, se realizaron pruebas en tiempo real del sistema en conjunto, con la integración de la red de IMU's, control de movimientos tanto del robot NAO en simulación y en físico, con la finalidad de analizar el comportamiento total del sistema durante su ejecución en tiempo real, al realizarse diferentes poses por la persona que use la red de sensores.

Se establecieron un conjunto de reglas e instrucciones para la medición y registro adecuada, de los sensores inerciales junto con el funcionamiento en conjunto del sistema. Las reglas e instrucciones especifican: la posición de los sensores inerciales sobre músculos específicos los cuales se detallaron en el capítulo 3 sección 1, así como también, la limitación del rango angular de cada articulación especificado en el capítulo 2 sección 4. El método aplicado para la realización de las pruebas de funcionamiento, se basa en la realización de ejercicios físicos o poses específicas por parte de la persona que use la red de sensores, durante un tiempo determinado. Las pruebas de funcionamiento fueron realizadas en conjunto con un estudiante y el autor de la presente

tesis, en el laboratorio de Control Industrial de la Universidad de las Fuerzas Armadas – ESPE.

Las mediciones realizadas por los 16 IMU's, registran y procesan la velocidad angular, la aceleración y los campos magnéticos, en los ejes X, Y y Z. Con la finalidad de obtener la orientación de cada IMU y a su vez, obtener la variación angular de cada articulación que deba de realizar el robot NAO, tanto en simulación como el físico. Con los valores angulares de cada articulación se calcula la posición de cada efector del robot por medio de la cinemática directa y el cálculo del centro de masa del robot.

6.1.1. Ejercicios y poses de pruebas

Se establecieron un conjunto de ejercicios de pruebas, con el fin de evaluar la respuesta total del funcionamiento del sistema y la capacidad del robot simulado y físico, de reproducir e imitar los movimientos del usuario dentro de un ambiente controlado. Los ejercicios que se realizaran se describen a continuación:

- Ejercicios del Cuello: Los movimientos a realizarse son Flexión, extensión y se reproducen con la articulación HeadPitch, y rotación que se reproduce con la articulación HeadYaw.
- Ejercicios de Hombros: Los movimientos a realizarse son flexión y extensión vertical del hombro, que se reproducen con las articulaciones RShoulderPitch y LShoulderPitch para el hombro derecho e izquierdo respectivamente. También se realiza los movimientos de flexión y extensión horizontal del hombro, los cuales se reproducen con las articulaciones RShoulderRoll y LShoulderRoll para el hombro derecho e izquierdo respectivamente.

- Ejercicios en codos: Los movimientos a realizarse son flexión y extensión del codo, que se reproducen con las articulaciones RElbowRoll y LElbowRoll para el codo derecho e izquierdo respectivamente.
- Ejercicios en cadera: Los movimientos a realizarse son flexión y extensión de las articulaciones de la cadera, que se reproducen con las articulaciones RHipPitch y LHipPitch. También se producen los movimientos que son de abducción y aducción que se reproduce con las articulaciones RHipRoll y LHipRoll para cada pierna.
- Ejercicios de rodilla: Los movimientos a realizarse son flexión y extensión de la rodilla, que se reproducen con las articulaciones RKneePitch y LKneePitch para la rodilla de la pierna derecha e izquierda respectivamente.

6.2. Análisis de Resultados

Las pruebas de funcionamiento que se realizaron se basaron en los ejercicios que se detallaron anteriormente, por lo cual se presentaran los valores obtenidos de los sensores inerciales ya procesados, los cuales controlan el movimiento angular de cada articulación, de acuerdo a la distribución de sensores detallado en la (Tabla 16), además se presentan la ejecución de dichos movimientos tanto con el robot simulado como en el físico.

6.2.1. Generación de movimientos

- Ejercicios del cuello: Para la realización de dicha prueba se realizaron los ejercicios de flexión y extensión del cuello.

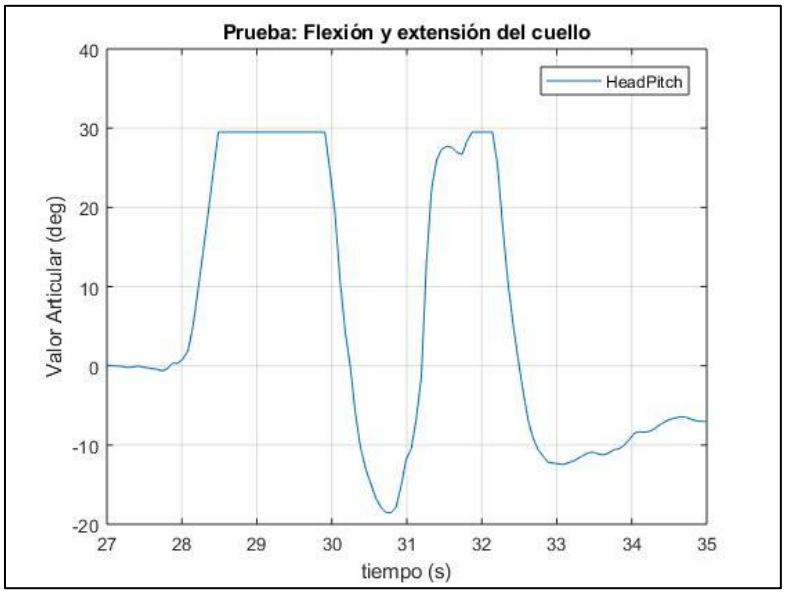


Figura 84. Señales medidas durante la prueba del cuello

En la (Figura 84) se muestra los valores articulares que se obtuvieron durante la flexión y extensión del cuello, en donde primero se realizó la flexión del cuello obteniéndose así valores positivos en el valor articular, seguidamente se realizó la extensión del cuello obteniéndose así valores negativos en el valor articular. El sistema realizado consta con la función “Condiciones”, la cual en el caso de que se produzcan valores angulares que sobrepasen los límites de movimiento de cada articulación, los procede a recortar dejando como valor máximo el permitido para la articulación en cuestión, como es el caso que se suscitó durante la flexión.

En las siguientes figuras se muestra las pruebas de funcionamiento con el robot simulado y físico, durante el ejercicio de movilidad del cuello.

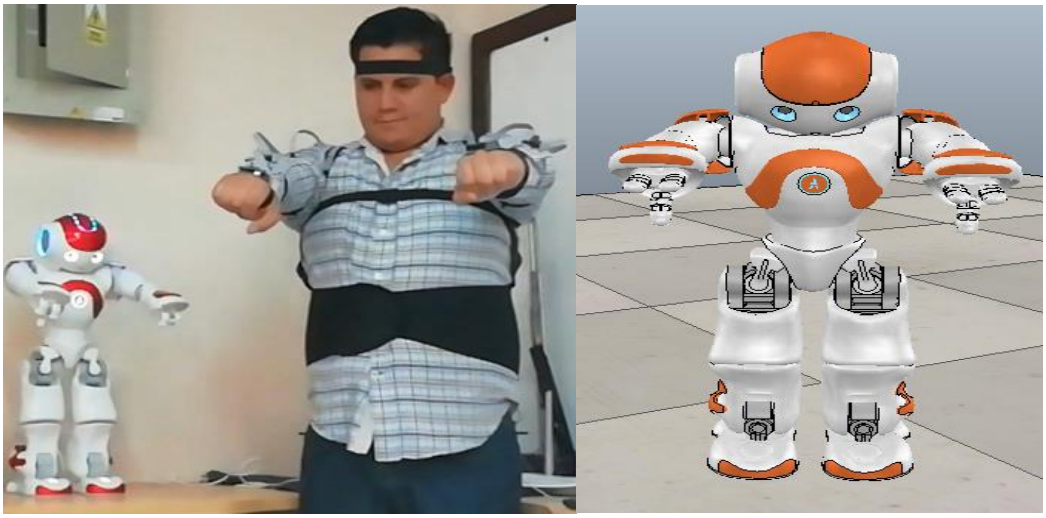


Figura 85. Prueba de flexión del cuello con robot físico (Izq.) y simulado (Der.)

En la siguiente tabla se presenta la posición del efector final de la cadena cinemática de la cabeza y del centro de masa total del robot, con respecto a los ejercicios realizados.

Tabla 17

Cálculos para las pruebas de ejercicios del cuello

Ejercicio	Valor Articular	Posición Efector (mm)	Posición CoM (mm)
Posición de referencia	$HeadYaw = 0°$ $HeadPitch = 0°$	$X = 53.9$ $Y = 0$ $Z = 194.4$	$X = 21.075$ $Y = -0.4117$ $Z = -35.9583$
Flexión Cuello	$HeadYaw = 0°$ $HeadPitch = 29°$	$X = 80.06$ $Y = 0$ $Z = 159.75$	$X = 21.075$ $Y = -0.3557$ $Z = -35.9597$
Extensión Cuello	$HeadYaw = 0°$ $HeadPitch = -38°$	$X = 0.6704$ $Y = 0$ $Z = 213.1901$	$X = 21.0755$ $Y = -0.3557$ $Z = -35.9597$

- Ejercicios de Hombros Derechos: Para la realización de dicha prueba se realizaron los ejercicios de flexión y extensión del hombro de forma vertical.

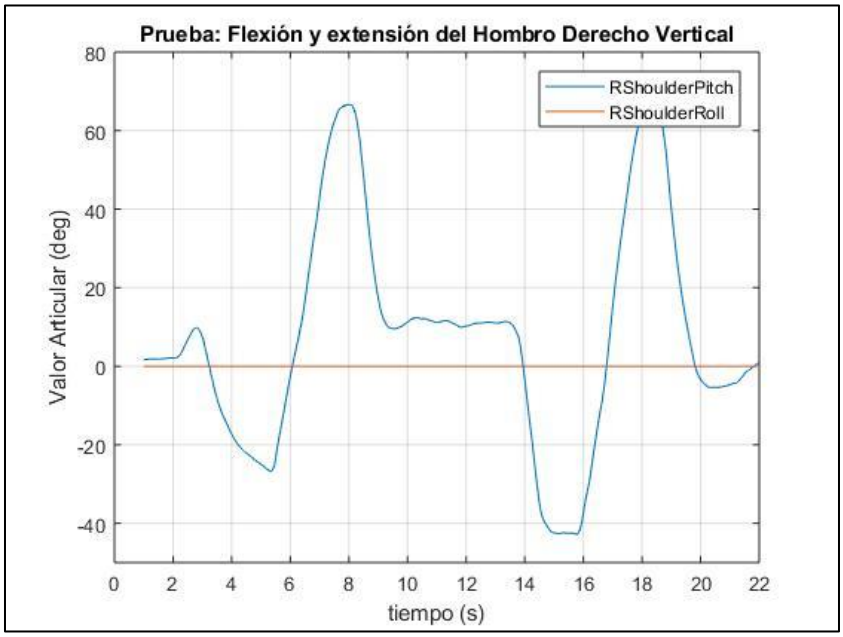


Figura 86 Señales medidas durante la prueba del hombro

En la (Figura 86) se muestra los valores articulares que se obtuvieron durante la extensión y flexión del hombro, en donde primero se realizó la extensión del hombro obteniéndose así valores negativos en el valor articular, seguidamente se realizó la flexión del hombro obteniéndose así valores positivos en el valor articular.

En las siguientes figuras se muestra las pruebas de funcionamiento con el robot simulado y físico, durante el ejercicio de movilidad del hombro.

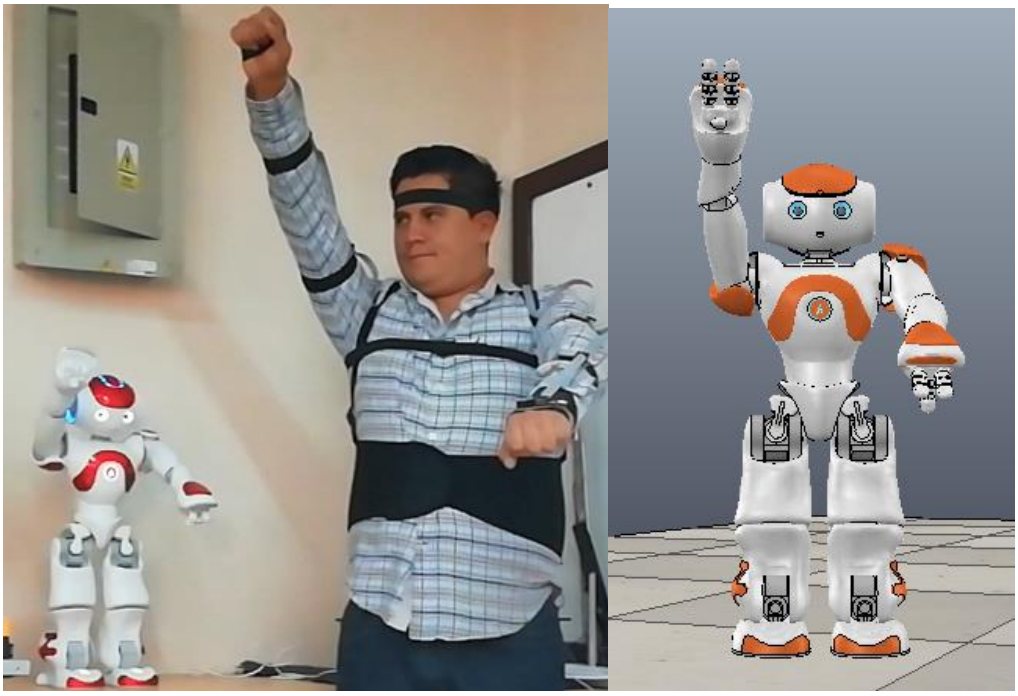


Figura 87. Prueba de extensión: hombro con robot (Izq.) y simulado (Der.)

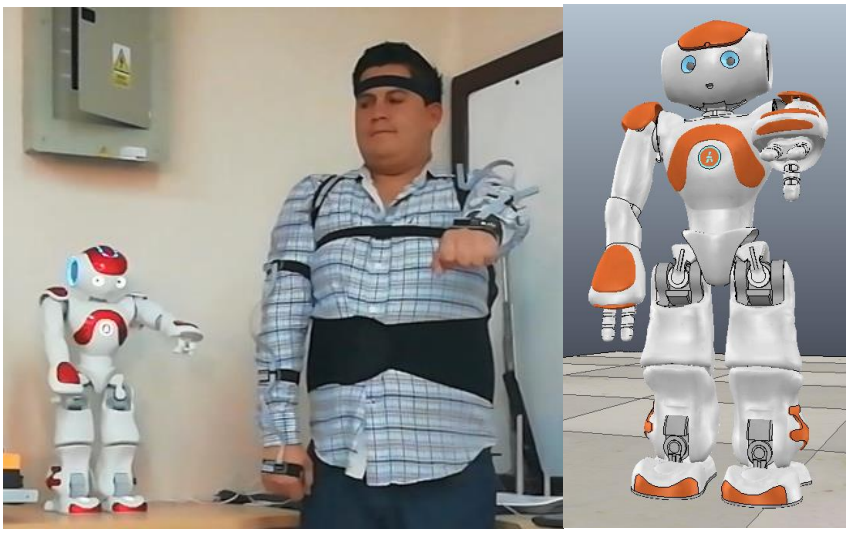


Figura 88. Prueba de flexión del hombro con robot (Izq.) y simulado (Der.)

En la (Tabla 18) se presenta el resultado del cálculo de la posición del efector final de la cadena cinemática del brazo derecho y del centro de masa total del robot, con respecto a los ejercicios realizados.

Tabla 18

Cálculos para las pruebas de ejercicios del hombro

Ejercicio	Valor Articular	Posición Efector (mm)	Posición CoM (mm)
Posición de referencia	$RShoulderPitch = 0°$ $RShoulderRoll = 0°$ $RElbowYaw = 0°$ $RElbowRoll = 0°$ $RWristYaw = 0°$	$X = 218.7$ $Y = -113$ $Z = 87.69$	$X = 21.075$ $Y = -0.3557$ $Z = -35.9597$
Flexión Hombro	$RShoulderPitch = 70°$ $RShoulderRoll = 0°$ $RElbowYaw = 0°$ $RElbowRoll = 0°$ $RWristYaw = 0°$	$X = 53.3674$ $Y = 0$ $Z = 194$	$X = 14.45$ $Y = -0.3557$ $Z = -45.85$
Extensión Hombro	$RShoulderPitch = -30°$ $RShoulderRoll = 0°$ $RElbowYaw = 0°$ $RElbowRoll = 0°$ $RWristYaw = 0°$	$X = 183.2448$ $Y = -113$ $Z = -20.0108$	$X = 19.5761$ $Y = -0.3557$ $Z = -30.7991$

- Ejercicios de Hombro Izquierdo: Para la realización de dicha prueba se realizaron los ejercicios de flexión y extensión del hombro de forma vertical.

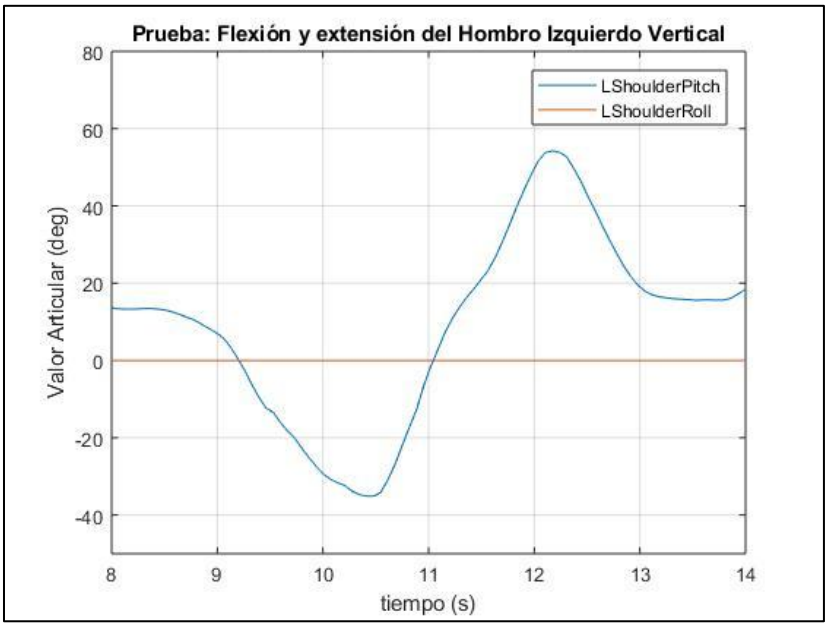


Figura 89. Señales medidas durante las prueba del Hombro

En la (Figura 89) se muestra los valores articulares que se obtuvieron durante la extensión y flexión del hombro, en donde primero se realizó la extensión del hombro obteniéndose así valores negativos en el valor articular, seguidamente se realizó la flexión del hombro obteniéndose así valores positivos en el valor articular.

En las siguientes figuras se muestra las pruebas de funcionamiento con el robot simulado y físico, durante el ejercicio de movilidad del hombro.

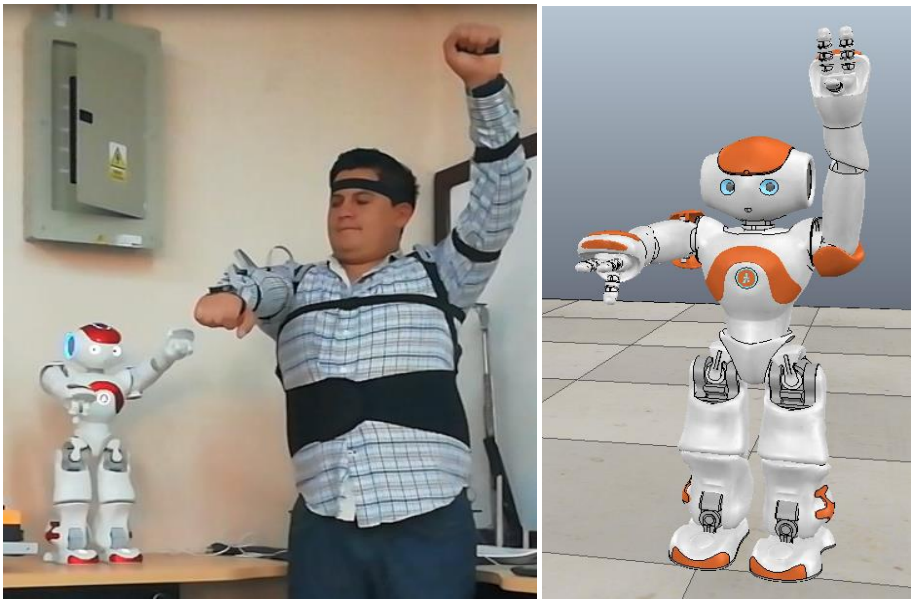


Figura 90. Prueba de extensión hombro con robot (Izq.) y simulado (Der.)

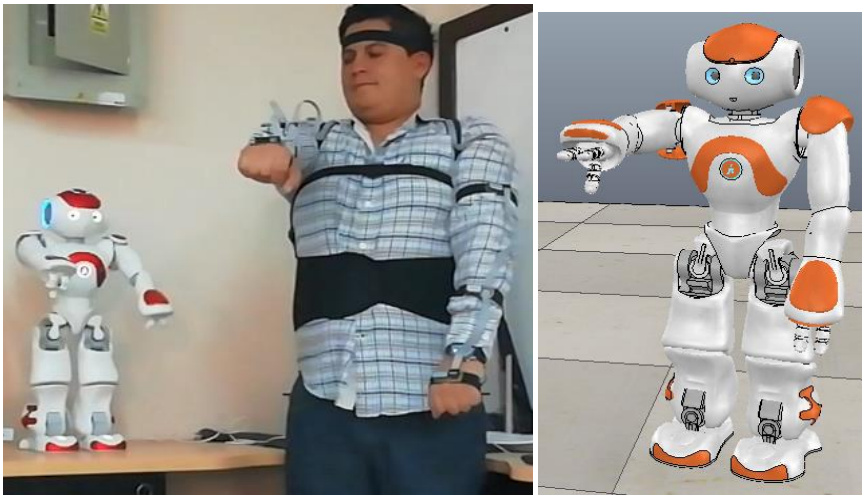


Figura 91. Prueba de flexión hombro con robot (Izq.) y simulado (Der.)

En la siguiente tabla se presenta el resultado del cálculo de la posición del efector final de la cadena cinemática del brazo izquierdo y del centro de masa total del robot, con respecto a los ejercicios realizados.

Tabla 19

Cálculos para las pruebas de ejercicios del hombro izquierdo

Ejercicio	Valor Articular	Posición Efector (mm)	Posición CoM (mm)
Posición de referencia	$LShoulderPitch = 0°$ $LShoulderRoll = 0°$ $LElbowYaw = 0°$ $LElbowRoll = 0°$ $LWristYaw = 0°$	$X = 218.7$ $Y = 113$ $Z = 87.69$	$X = 21.075$ $Y = -0.3557$ $Z = -35.9597$
Flexión Hombro	$LShoulderPitch = 70°$ $LShoulderRoll = 0°$ $LElbowYaw = 0°$ $LElbowRoll = 0°$ $LWristYaw = 0°$	$X = 63.2322$ $Y = 113$ $Z = -109.7210$	$X = 14.45$ $Y = -0.3557$ $Z = -26.0661$
Extensión Hombro	$LShoulderPitch = -30°$ $LShoulderRoll = 0°$ $LElbowYaw = 0°$ $LElbowRoll = 0°$ $LWristYaw = 0°$	$X = 195.5548$ $Y = 113$ $Z = 198.6892$	$X = 19.5761$ $Y = -0.3557$ $Z = -40.121$

- Ejercicios en cadera derecha: Para la realización de dicha prueba se realizaron movimientos de flexión y extensión de las articulaciones de la cadera derecha.

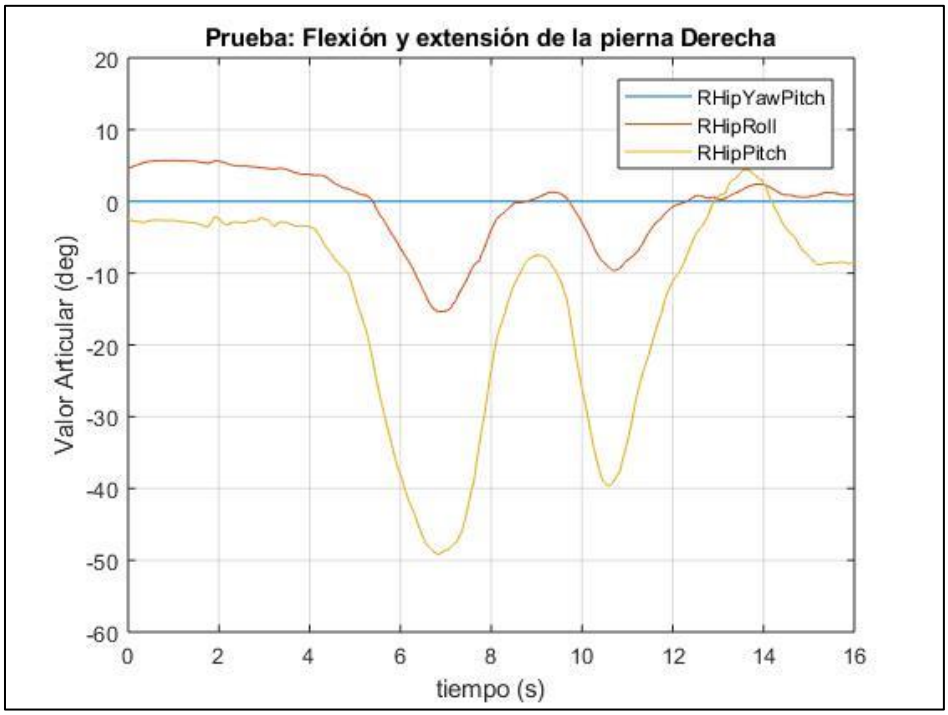


Figura 92. Señales medidas durante las prueba de pierna derecha

En la (Figura 92) se muestra los valores articulares que se obtuvieron durante la extensión y flexión de la pierna derecha, y a su vez se observa que se genera un leve valor angular en RHipRoll, proporcional al valor angular de la articulación RHipPitch, lo cual se debe a que la posición del sensor R413, el cual tiende a tener ligeras variaciones de posición cuando se flexiona o extiende la pierna derecha.

En las siguientes figuras se muestra las pruebas de funcionamiento con el robot simulado y físico, durante el ejercicio de movilidad de la pierna derecha

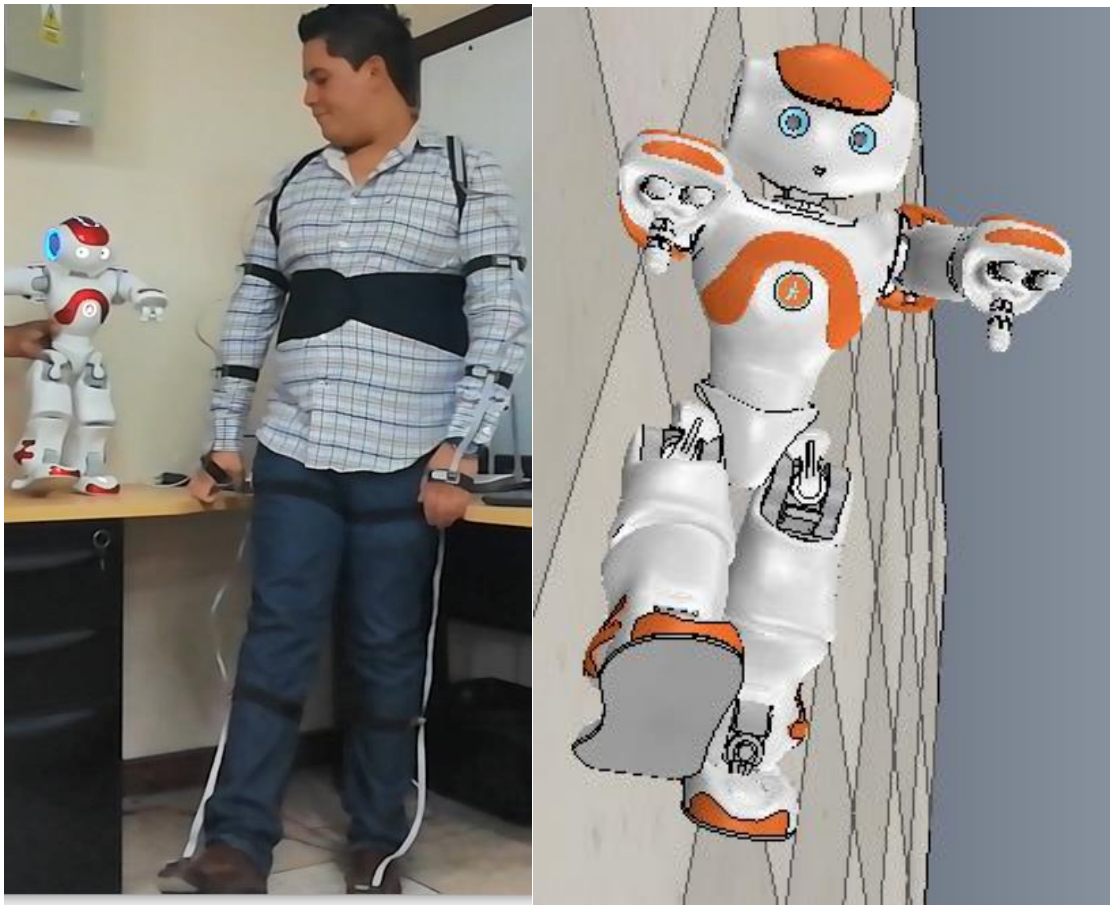


Figura 93. Prueba de flexión pierna derecha con robot (Izq.) y simulado (Der.)

En la (Figura 93) se observa la prueba realizada y que, al realizarse la flexión de la articulación de la pierna derecha, el robot pierde su estabilidad y corre el riesgo de caerse, por lo cual, para evitar las caídas con el robot físico y para que la prueba sea satisfactoria, se tuvo que sostener al robot, con la finalidad de evitar que pierda su estabilidad. Esto se debe a que el robot NAO al no contar con la misma morfología de un humano, no puede imitar los movimientos de un humano de forma exacta, para analizar la estabilidad del robot, se calculó el CoM, el cual se presenta en la (Tabla 20). En el caso del robot

simulado, al no contar con un apoyo para mantener su estabilidad, este tiende a caerse ya que su CoM se encuentra fuera del polígono de soporte.

En la siguiente tabla se presenta el resultado del cálculo de la posición del efector final de la cadena cinemática de la pierna derecha y del centro de masa total del robot, con respecto a los ejercicios realizados.

Tabla 20

Cálculos para prueba de flexión y extensión de pierna derecha

Ejercicio	Valor Articular	Posición Efector (mm)	Posición CoM (mm)
Posición de referencia	$RHipYawPitch = 0°$ $RHipRoll = 0°$ $RHipPitch = 0°$ $RKneePitch = 0°$ $RAnklePitch = 0°$ $RAnkleRoll = 0°$	$X = 0$ $Y = -50$ $Z = -333.09$	$X = 21.0755$ $Y = -0.3557$ $Z = -35.9597$
Flexión pierna	$RHipYawPitch = 0°$ $RHipRoll = -15°$ $RHipPitch = -50°$ $RKneePitch = 0°$ $RAnklePitch = 0°$ $RAnkleRoll = 0°$	$X = 190.0480$ $Y = -91.2737$ $Z = -239.0354$	$X = 39.8091$ $Y = -4.3014$ $Z = -25.4832$
Extensión pierna	$RHipYawPitch = 0°$ $RHipRoll = 5°$ $RHipPitch = 10°$ $RKneePitch = 0°$ $RAnklePitch = 0°$ $RAnkleRoll = 0°$	$X = -43.0804$ $Y = -28.7060$ $Z = -328.3912$	$X = 16.7142$ $Y = 1.8226$ $Z = -35.7278$

Al realizarse solo el movimiento de la pierna derecha, sin haber realizado ningún movimiento con la pierna izquierda o brazos, que permitan reubicar la proyección del CoM dentro del área del polígono de soporte, que en este caso al encontrarse solo un pie de apoyo ubicado en el pie de la pierna izquierda, el polígono de soporte se encontrara en dicho apoyo, en donde su posición es $P_{soporte} = (0{,}50, -333.9)$ y la de la proyección al

centro de masa para el primer ejercicio es $P_{CoM} = (39.8091, -4.3014, -25.4832)$, en donde las distancia entre los dos puntos es $67.33\,mm$ y teniendo en cuenta que la distancia máxima requerida debe de ser menor a $50\,mm$, se determina que el robot se encuentra desequilibrado y, por lo tanto, corre el riesgo de caerse.

También se realizaron pruebas con los ejercicios de abducción y aducción con la pierna derecha, tal como se indica a continuación:

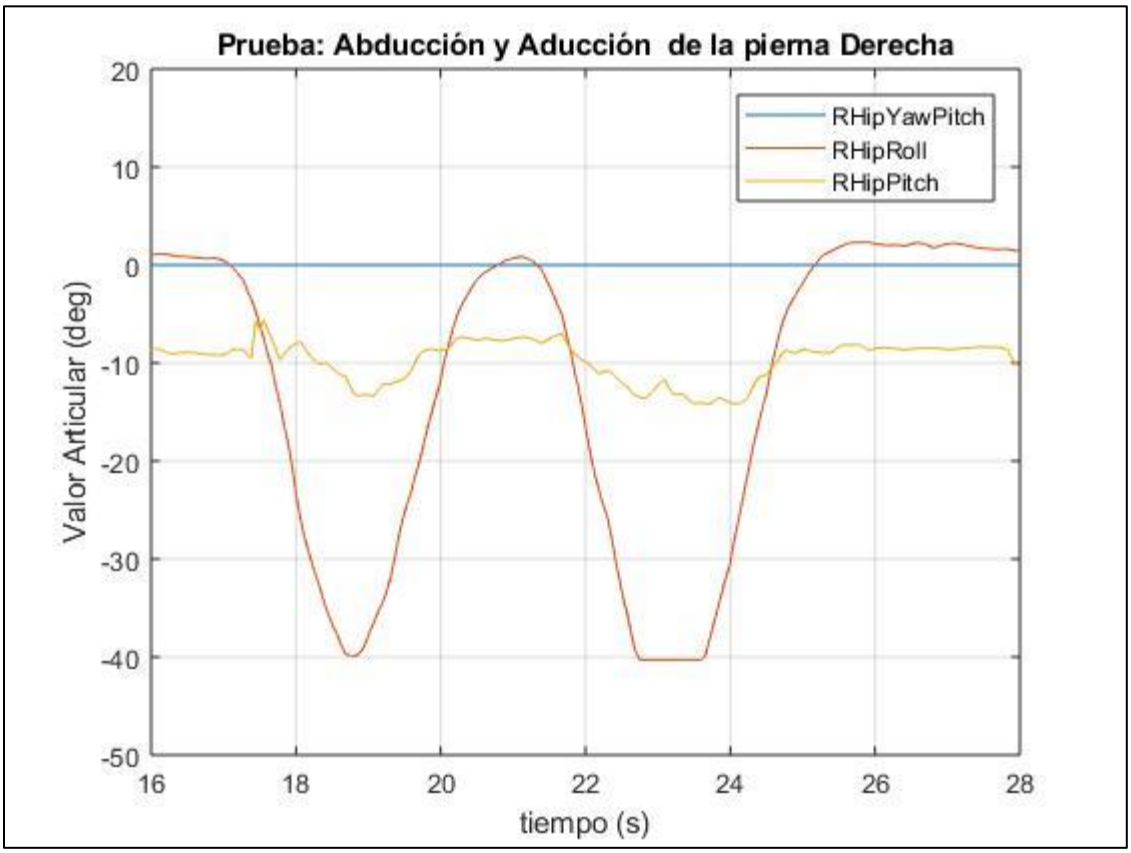


Figura 94. Señales medidas durante el segundo ejercicio de pierna derecha

En la (Figura 94) se observa que la señal medida correspondiente a RHipRoll es recortada en el segundo ejercicio de abducción, debido a la acción de la función

Condición, y a su vez se observa una ligera variación con respecto a la señal correspondiente a RHipPitch, lo cual se debe a los ligeros movimientos que se producen al realizarse dicho movimiento.

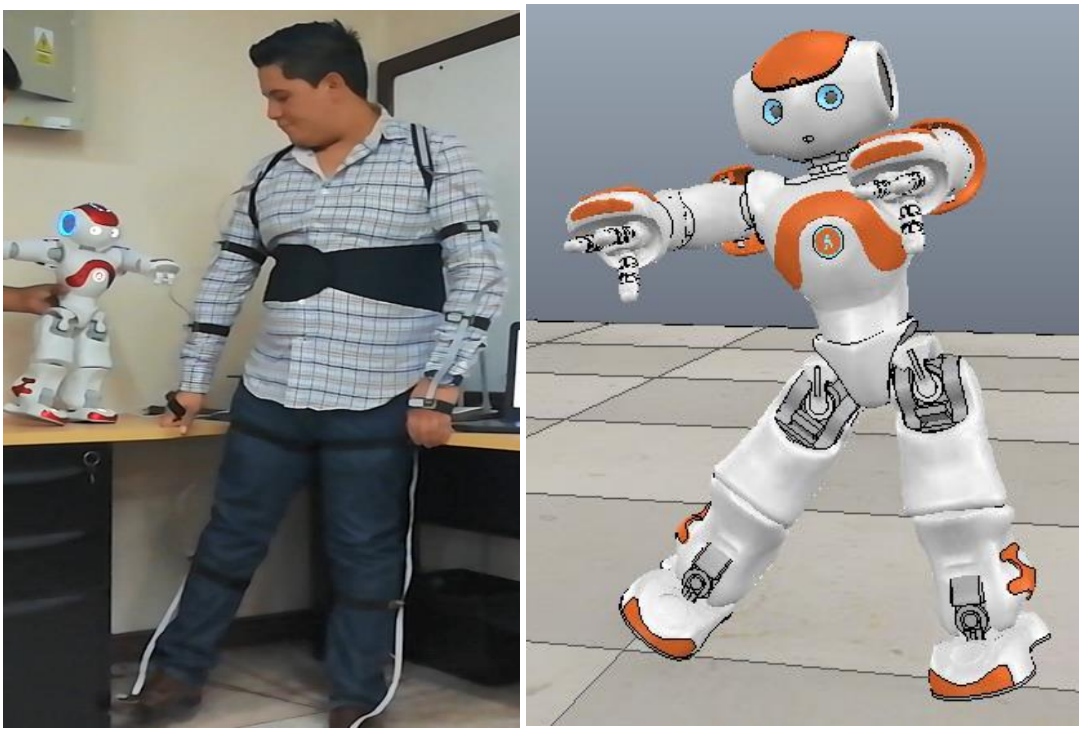


Figura 95. Prueba de abducción pierna derecha en robot (Izq.) y simulado (Der.)

En la (Figura 95), se observa la prueba de abducción realizada, en la cual el pie derecho no abandona su apoyo con el piso, lo cual permite que la proyección del CoM se encuentre dentro del polígono de soporte formado por los dos pies, dando como resultado un ejercicio equilibrado para el robot NAO.

En la siguiente tabla se presenta el resultado del cálculo de la posición del efector final de la cadena cinemática de la pierna derecha y del centro de masa total del robot, con respecto a los ejercicios realizados.

Tabla 21

Cálculos para prueba de abducción y aducción de pierna derecha

Ejercicio	Valor Articular	Posición Efector (mm)	Posición CoM (mm)
Posición de referencia	$RHipYawPitch = 0°$ $RHipRoll = 0°$ $RHipPitch = 0°$ $RKneePitch = 0°$ $RAnklePitch = 0°$ $RAnkleRoll = 0°$	$X = 0$ $Y = -50$ $Z = -333.09$	$X = 21.0755$ $Y = -0.3557$ $Z = -35.9597$
Abducción Pierna	$RHipYawPitch = 0°$ $RHipRoll = -40°$ $RHipPitch = -10°$ $RKneePitch = 0°$ $RAnklePitch = 0°$ $RAnkleRoll = 0°$	$X = 43.0804$ $Y = -207.0465$ $Z = -272.1607$	$X = 25.4006$ $Y = -16.0466$ $Z = -29.3616$
Aducción Pierna	$RHipYawPitch = 0°$ $RHipRoll = 5°$ $RHipPitch = -10°$ $RKneePitch = 0°$ $RAnklePitch = 0°$ $RAnkleRoll = 0°$	$X = 43.0804$ $Y = -28.7060$ $Z = -328.3912$	$X = 25.4006$ $Y = 1.7865$ $Z = -35.3150$

Al realizarse solo el movimiento de la pierna derecha, sin haber realizado ningún movimiento con la pierna izquierda o brazos, y teniendo en cuenta que el pie derecho no dejo de apoyarse con el suelo, permiten que la proyección del CoM, se mantenga dentro del área del polígono de soporte. Por lo tanto, el robot mantiene su estabilidad.

- Ejercicios en cadera izquierda: Para la realización de dicha prueba se realizaron movimientos de flexión y extensión de las articulaciones de la cadera izquierda.

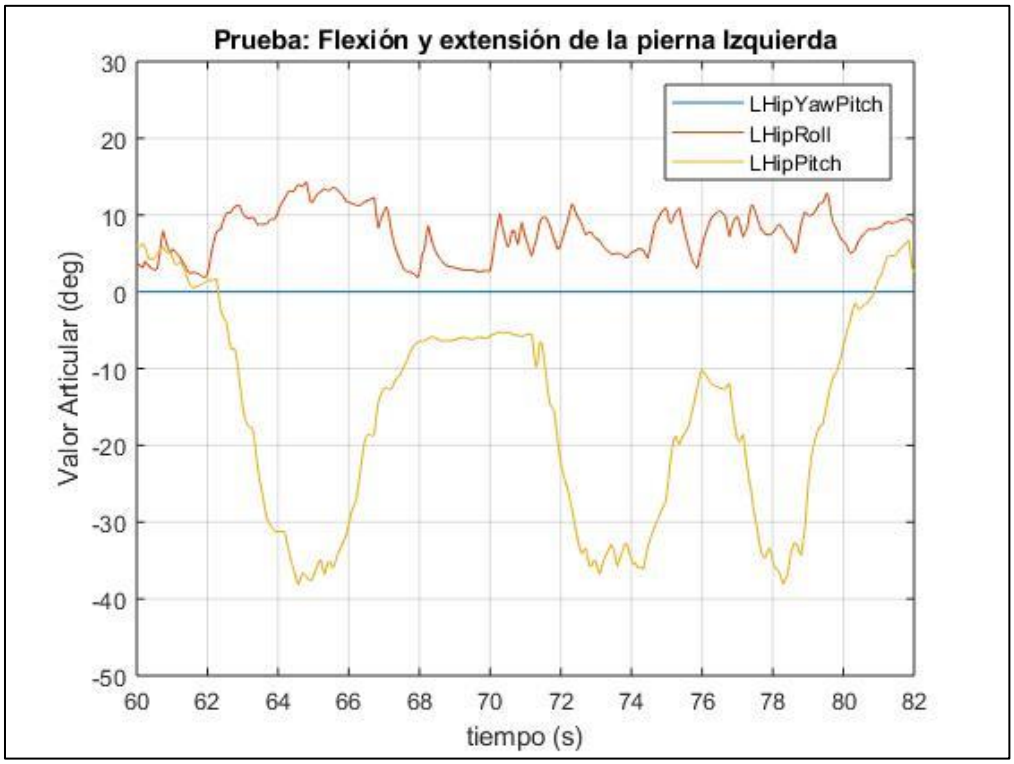


Figura 96. Señales medidas durante las prueba de pierna izquierda

En la (Figura 96) se muestra los valores articulares que se obtuvieron durante la extensión y flexión de la pierna izquierda, y a su vez se observa que se genera un leve valor angular en LHipRoll, se debe a que la posición del sensor L516, el cual tiende a tener ligeras variaciones de posición cuando se flexiona o extiende la pierna izquierda.

En las siguientes figuras se muestra las pruebas de funcionamiento con el robot simulado y físico, durante el ejercicio de movilidad de la pierna izquierda.

Figura 97. Prueba de flexión pierna izquierda en robot físico (Izq.) y simulado (Der.)

En la (Figura 97) se observa la prueba realizada y que, al realizarse la flexión de la articulación de la pierna izquierda, el robot pierde su estabilidad y corre el riesgo de caerse, por lo cual, para evitar las caídas con el robot físico y para que la prueba sea satisfactoria, se tuvo que sostener al robot de la misma forma que se lo realizó para el ejercicio de pierna derecha, con la finalidad de evitar que pierda su estabilidad.

En la siguiente tabla se presenta el resultado del cálculo de la posición del efector final de la cadena cinemática de la pierna izquierda y del centro de masa total del robot, con respecto a los ejercicios realizados.

Tabla 22

Cálculos para prueba de flexión y extensión de pierna izquierda.

Ejercicio	Valor Articular	Posición Efector (mm)	Posición CoM (mm)
Posición de referencia	$LHipYawPitch = 0°$ $LHipRoll = 0°$ $LHipPitch = 0°$ $LKneePitch = 0°$ $LAnklePitch = 0°$ $LAnkleRoll = 0°$	$X = 0$ $Y = 50$ $Z = -333.09$	$X = 21.0755$ $Y = -0.3557$ $Z = -35.9597$
Flexión Pierna	$LHipYawPitch = 0°$ $LHipRoll = 10°$ $LHipPitch = -35°$ $LKneePitch = 0°$ $LAnklePitch = 0°$ $LAnkleRoll = 0°$	$X = 0$ $Y = 85.2894$ $Z = -285.1360$	$X = 35.2057$ $Y = 3.1006$ $Z = -30.3775$
Extensión Pierna	$LHipYawPitch = 0°$ $LHipRoll = 10°$ $LHipPitch = 10°$ $LKneePitch = 0°$ $LAnklePitch = 0°$ $LAnkleRoll = 0°$	$X = 0$ $Y = 92.4259$ $Z = -325.6092$	$X = 16.7142$ $Y = 3.9750$ $Z = -35.3360$

Al realizarse solo el movimiento de la pierna izquierda, sin haber realizado ningún movimiento con la pierna derecha o brazos, que permitan reubicar la proyección del CoM dentro del área del polígono de soporte ubicado en el pie de la pierna derecho, en donde su posición es $P_{soporte} = (0, -50, -333.9)$ y la de la proyección al centro de masa para el primer ejercicio es $P_{CoM} = (35.2057, 3.1006, -333.9)$, en donde la distancia entre los dos puntos es $63.7163\ mm$ y teniendo en cuenta que la distancia máxima requerida debe de ser menor a $50\ mm$, se determina que el robot se encuentra desequilibrado y, por lo tanto, corre el riesgo de caerse.

También se realizaron pruebas con los ejercicios de abducción y aducción con la pierna izquierda, tal como se indica a continuación:

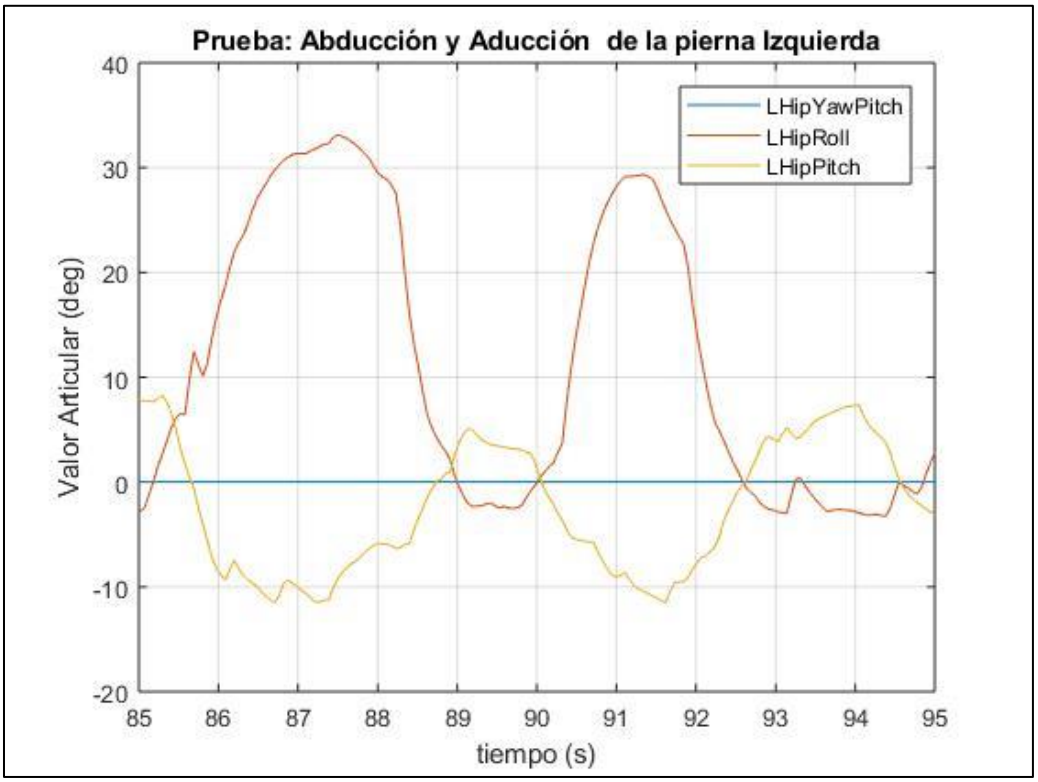


Figura 98. Señales medidas durante el segundo ejercicio de pierna izquierda

En la (Figura 98) se observa que la señal medida correspondiente a LHipRoll es recortada en el segundo ejercicio de abducción, debido a la acción de la función Condición, y a su vez se observa una ligera variación con respecto a la señal correspondiente a LHipPitch, lo cual se debe a los ligeros movimientos que se producen al realizarse dicho movimiento.

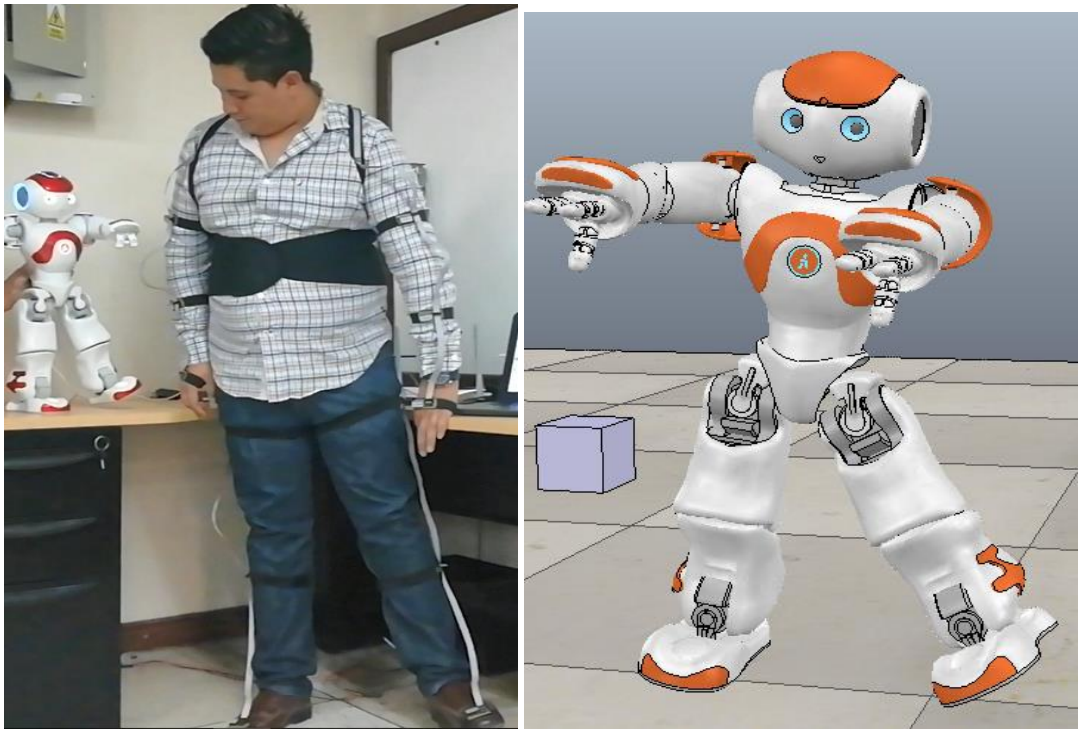


Figura 99. Prueba de abducción pierna izquierda en robot (Izq.) y simulado (Der.)

En la (Figura 99), se observa la prueba de abducción realizada, en la cual el pie derecho no abandona su apoyo con el piso, lo cual permite que la proyección del CoM se encuentre dentro del polígono de soporte formado por los dos pies, dando como resultado un ejercicio equilibrado para el robot NAO.

En la siguiente tabla se presenta el resultado del cálculo de la posición del efector final de la cadena cinemática de la pierna izquierda y del centro de masa total del robot, con respecto a los ejercicios realizados.

Tabla 23

Cálculos para prueba de abducción y aducción de pierna izquierda

Ejercicio	Valor Articular	Posición Efector (mm)	Posición CoM (mm)
Posición de referencia	$LHipYawPitch = 0°$ $LHipRoll = 0°$ $LHipPitch = 0°$ $LKneePitch = 0°$ $LAnklePitch = 0°$ $LAnkleRoll = 0°$	$X = 0$ $Y = 50$ $Z = -333.09$	$X = 21.0755$ $Y = -0.3557$ $Z = -35.9597$
Abducción pierna	$LHipYawPitch = 0°$ $LHipRoll = 35°$ $LHipPitch = -10°$ $LKneePitch = 0°$ $LAnklePitch = 0°$ $LAnkleRoll = 0°$	$X = 0$ $Y = 190.1367$ $Z = -285.1360$	$X = 25.4006$ $Y = 13.6572$ $Z = -30.6944$
Aducción Pierna	$LHipYawPitch = 0°$ $LHipRoll = -10°$ $LHipPitch = 10°$ $LKneePitch = 0°$ $LAnklePitch = 0°$ $LAnkleRoll = 0°$	$X = 0$ $Y = 7.5741$ $Z = -325.6093$	$X = 16.7142$ $Y = -4.6988$ $Z = -35.4791$

Al realizarse solo el movimiento de la pierna izquierda, sin haber realizado ningún movimiento con la pierna derecha o brazos, y teniendo en cuenta que el pie izquierdo no dejo de apoyarse con el suelo, permiten que la proyección del CoM, se mantenga dentro del área del polígono de soporte. Por lo tanto, el robot mantiene su estabilidad.

- Ejercicios de rodilla derecha: Para la realización de la prueba, se realizaron los movimientos de flexión y extensión de la rodilla.

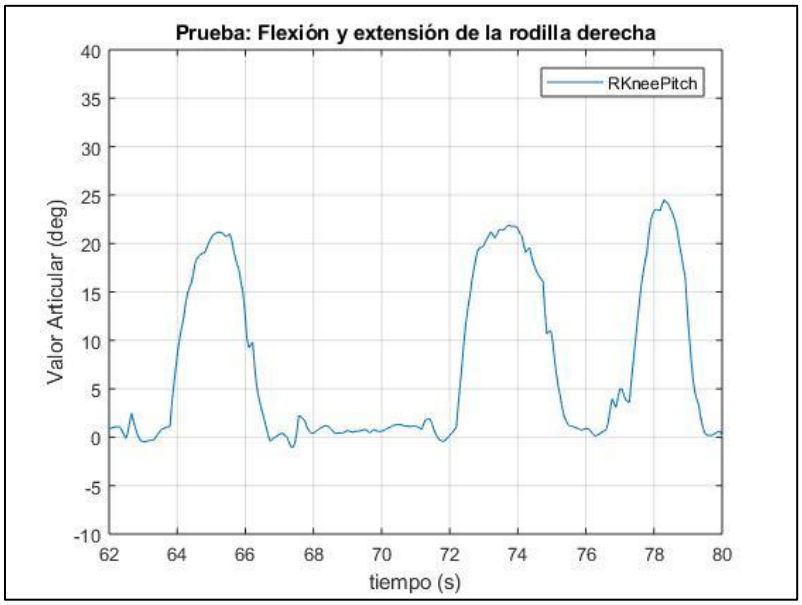


Figura 100. Señales medidas durante el segundo ejercicio de la rodilla Derecha

En la (Figura 100) se muestra los valores articulares que se obtuvieron durante la extensión y flexión de la rodilla derecha.

En las siguientes figuras se muestra las pruebas de funcionamiento con el robot simulado y físico, durante el ejercicio de movilidad de la rodilla derecha.

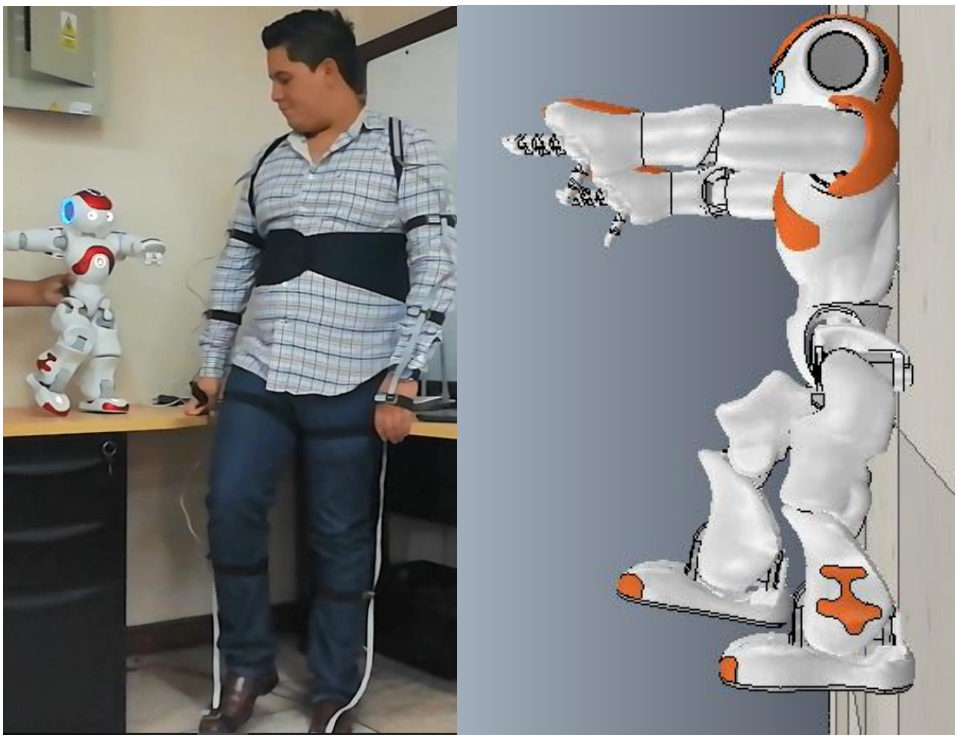


Figura 101. Prueba de flexión de rodilla derecha en robot (Izq.) y simulado (Der.)

En la (Figura 101) se observa la prueba realizada y que, al realizarse la flexión de la articulación de la rodilla derecha, el robot pierde su estabilidad y corre el riesgo de caerse, por lo cual, para evitar las caídas con el robot físico y para que la prueba sea satisfactoria, se tuvo que sostener al robot, con la finalidad de evitar que pierda su estabilidad.

En la siguiente tabla se presenta el cálculo de la posición del efector final de la cadena cinemática de la pierna derecha y del centro de masa total del robot, con respecto a los ejercicios realizados.

Tabla 24

Cálculos para prueba de flexión y extensión de rodilla derecha

Ejercicio	Valor Articular	Posición Efector (mm)	Posición CoM (mm)
Posición de referencia	$RHipYawPitch = 0°$ $RHipRoll = 0°$ $RHipPitch = 0°$ $RKneePitch = 0°$ $RAnklePitch = 0°$ $RAnkleRoll = 0°$	$X = 0$ $Y = -50$ $Z = -333.09$	$X = 21.0755$ $Y = -0.3557$ $Z = -35.9597$
Flexión rodilla derecha	$RHipYawPitch = 0°$ $RHipRoll = 0°$ $RHipPitch = -50°$ $RKneePitch = 25°$ $RAnklePitch = 0°$ $RAnkleRoll = 0°$	$X = 139.19$ $Y = -50$ $Z = -283.4939$	$X = 36.7948$ $Y = -0.3557$ $Z = -29.02$

Al realizarse solo movimientos de la pierna y rodilla derecha, sin haber realizado ningún movimiento con la pierna izquierda o brazos, que permitan reubicar la proyección del CoM dentro del área del polígono de soporte ubicado en el pie de la pierna izquierda, en donde su posición es $P_{soporte} = (0{,}50, -333.9)$ y la de la proyección al centro de masa para el ejercicio es $P_{CoM} = (36.7948, -0.3557, -333.9)$, en donde las distancia entre los dos puntos es $61.33\ mm$ y teniendo en cuenta que la distancia máxima requerida debe de ser menor a $50\ mm$, se determina que el robot se encuentra desequilibrado y, por lo tanto, corre el riesgo de caerse.

- Ejercicios de rodilla izquierda: Para la realización de la prueba, se realizaron los movimientos de flexión y extensión de la rodilla.

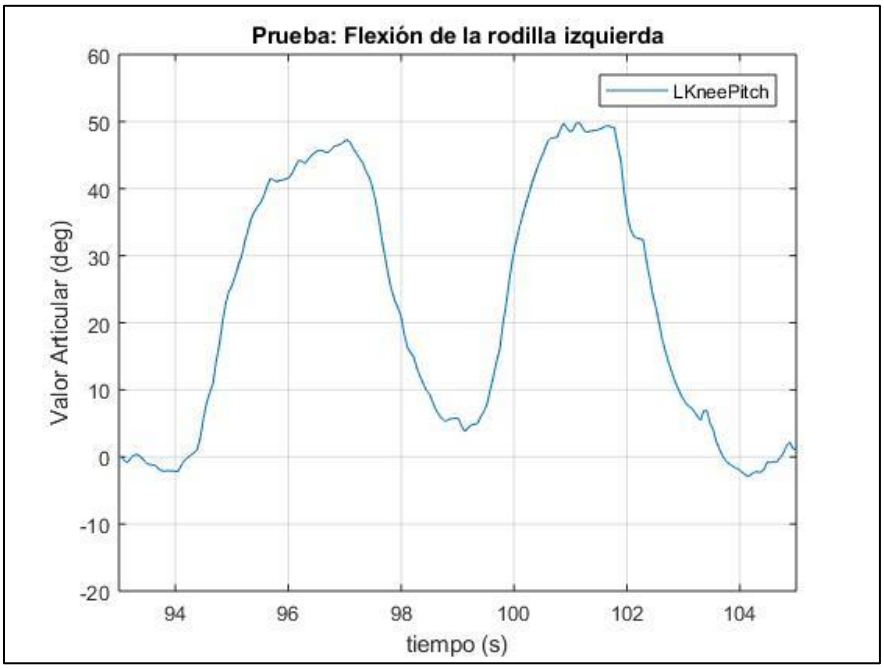


Figura 102. Señal medida del ejercicio de la rodilla izquierda

En la (Figura 102) se muestra los valores articulares que se obtuvieron durante la flexión de la rodilla izquierda.

En las siguientes figuras se muestra las pruebas de funcionamiento con el robot simulado y físico, durante el ejercicio de movilidad de la rodilla izquierda.

Figura 103. Prueba de flexión de rodilla izquierda en robot (Izq.) y simulado (Der.)

En la (Figura 103) se observa la prueba realizada y que, al realizarse la flexión de la articulación de la rodilla izquierda, el robot pierde su estabilidad y corre el riesgo de caerse, por lo cual, para evitar las caídas con el robot físico y para que la prueba sea satisfactoria, se tuvo que sostener al robot, con la finalidad de evitar que pierda su estabilidad.

En la siguiente tabla se presenta el cálculo de la posición del efector final de la cadena cinemática de la pierna izquierda y del centro de masa total del robot, con respecto a los ejercicios realizados.

Tabla 25

Cálculos para prueba de flexión y extensión de rodilla izquierda

Ejercicio	Valor Articular	Posición Efector (mm)	Posición CoM (mm)
Posición de referencia	$LHipYawPitch = 0°$ $LHipRoll = 0°$ $LHipPitch = 0°$ $LKneePitch = 0°$ $LAnklePitch = 0°$ $LAnkleRoll = 0°$	$X = 0$ $Y = 50$ $Z = -333.09$	$X = 21.0755$ $Y = -0.3557$ $Z = -35.9597$
Flexión rodilla izquierda	$LHipYawPitch = 0°$ $LHipRoll = 0°$ $LHipPitch = -50°$ $LKneePitch = 50°$ $LAnklePitch = 0°$ $LAnkleRoll = 0°$	$X = 139.19$ $Y = 50$ $Z = -297.3688$	$X = 32.8335$ $Y = -0.3557$ $Z = -30.3823$

Al realizarse solo movimientos de la pierna y rodilla izquierda, sin haber realizado ningún movimiento con la pierna derecha o brazos, que permitan reubicar la proyección del CoM dentro del área del polígono de soporte ubicado en el pie de la pierna derecha, en donde su posición es $P_{soporte} = (0, -50, -333.9)$ y la de la proyección al centro de masa para el ejercicio es $P_{CoM} = (32.8335, -0.3557, -333.9)$, en donde las distancia entre los dos puntos es $59\ mm$ y teniendo en cuenta que la distancia máxima requerida debe de ser menor a $50\ mm$, se determina que el robot se encuentra desequilibrado y, por lo tanto, corre el riesgo de caerse.

CAPÍTULO VII

CONCLUSIONES Y RECOMENDACIONES

7.1. Conclusiones

Los sensores inerciales MPU9250 cuentan solo con dos direcciones I2C (0x68H y 0x69H), siendo esta una desventaja si se requiere conectar más de 2 sensores en un mismo canal I2C. Por lo tanto, al trabajar con 8 sensores en un mismo canal, se optó por mantener a todos los sensores con la primera dirección y solo cambiar a la segunda dirección al sensor con el que se requiera realizar una comunicación, brindando así la capacidad de conectar más de dos sensores MPU9250 en un mismo canal.

Se utilizo el protocolo de comunicación TCP/IP, por el cual se envía los valores angulares de todas las articulaciones del robot NAO, desde el programa principal realizado en Matlab a otro programa en Python, desde el cual se puede programar y controlar al robot NAO, por medio del uso de las librerías, módulos y métodos de NAOqi SDK desarrollado en Python.

El cálculo de la orientación de Roll, Pitch y Yaw de cada sensor inercial, se realizó por medio del uso del filtro complementario, debido a su bajo coste computacional, facilidad de implementación y excelente respuesta para la aplicación realizada.

La configuración, lectura y procesamiento de datos de los sensores inerciales, se realizó satisfactoriamente con el controlador de la red de sensores ESP32, el cual, a su vez, se encarga de direccionar a los sensores y transmitir los valores angulares calculados por medio de wifi al servidor desarrollado en Matlab.

Los valores de posición de centros de masa de cada eslabón provistos por el fabricante son absolutos, por lo cual para calcular el centro de masa total del sistema y este al ser dinámico, se procedió a realizar cadenas cinemáticas hasta la articulación previa a la conexión con el eslabón en cuestión y tomando como efector final de la cadena cinemática la posición del centro de masa, logrando así obtener las posiciones relativas de los centros de masa de cada eslabón, para finalmente calcular la posición del CoM total del robot.

Se tomo como criterio de estabilidad del robot suficiente para la presente tesis, que el robot se mantendrá estable si y solo si, la proyección de la posición del CoM en el plano de apoyo, se encuentre dentro del área del polígono de soporte formado por los pies de apoyo. O en el caso de que se apoye sobre un pie, el polígono de soporte será solo el área del pie de apoyo.

En base a las pruebas realizadas, los movimientos de los brazos o cabeza casi no influyen en la estabilidad del robot, mientras que los movimientos realizados por las articulaciones de las piernas influyen directamente en la posición del CoM y, por lo tanto, en la estabilidad del robot.

Los valores articulares generados para el control de los movimientos, son condicionados de acuerdo a las limitaciones mecánicas que tiene el robot, estos valores generados son válidos para el robot simulado y para el robot físico.

Los movimientos que realiza el robot NAO son la captura de movimientos que realiza el humano que use la red de sensores, por lo cual, se debe de tener en cuenta que la morfología de un humano es diferente a la de un robot. Por lo tanto la estabilidad que

tiene un humano es diferente a la del robot NAO, por lo cual muchos movimientos que mantengan estable a una persona, pueden desestabilizar al robot NAO.

7.2. Recomendaciones

Si se conectan más de dos sensores en un mismo canal I2C, se recomienda conectar las resistencias Pull-up a las líneas SDA y SCL del canal I2C, cuyo valor resistivo depende de la velocidad de comunicación I2C, ya que las resistencias internas del controlador ESP32, no suministran suficiente corriente al canal si se conectan más de 2 sensores, con la finalidad de así evitar problemas de comunicación en el canal.

Se recomienda usar Python como lenguaje de programación del robot NAO, ya que permite obtener el mejor rendimiento de las librerías del robot NAO, y a su vez, permite entablar una comunicación en tiempo real con otros programas independientemente del lenguaje de programación con los que hayan sido realizados, restricción con la que cuenta el software Choreographe desarrollado por el fabricante del robot.

Se recomienda usar el filtro de Kalman para aplicaciones en las que se requiera un alto nivel de exactitud en los valores de orientación, para el presente caso, al no requerir valores tan exactos de orientación, se recomienda usar el filtro complementario, que, a su vez, tiende a ser una versión simplificada del filtro de Kalman.

Se recomienda realizar una red de sensores inerciales con controladores independientes para cada sensor, que permita censar, procesar los datos y enviarlos de forma inalámbrica al servidor, con la finalidad de así evitar problemas de comunicación sea por el cableado o canal de comunicación, que se pueden presentar al usar un bus datos y un solo canal de comunicación para dichos sensores.

El cálculo del CoM del robot, se realizó basándose en el análisis cinemático directo del robot NAO. Por lo cual se recomienda, que para calcular el CoM, se lo realice de la misma que se realizó el cálculo cinemático directo, y solo tomando la consideración que el CoM de dicho eslabón es el efector final de dicha cadena.

Se recomienda que, para futuros trabajos, previo a que se realice algún movimiento el robot NAO, se analice la estabilidad del robot para dicho movimiento, ya que en el caso de que el movimiento pueda desestabilizar al robot, se aplique alguna medida correctiva de pose para evitar que el robot realice movimientos futuros que puedan desestabilizarlo.

Se recomienda que para realizar una corrección en la estabilidad del robot NAO ante cualquier futuro movimiento se aplique la estrategia de estabilización “Cadera - Tobillo”, ya que esta estrategia reposiciona de forma más rápida al CoM dentro del polígono de soporte.

Previo al procesamiento cinemático y cálculo del CoM de los valores angulares de cada articulación, se recomienda condicionarlos de acuerdo a los rangos de movilidad angular de cada articulación, con respecto a los rangos de movilidad de cada articulación de la misma cadena cinemática, con la finalidad de evitar posibles choques entre eslabones.

Para mantener el equilibro del robot NAO, se debe de considerar el análisis previo de los movimientos del humano que use la red de sensores y determinar si son estales o no para el robot. Por lo cual, se recomienda implementar un controlador que modifique los valores angulares sea de la cadera o tobillo, para que estabilice al robot NAO ante cualquier posible movimiento.

REFERENCIAS BIBLIOGRÁFICAS

Adi, S., Setijadi, A., & Syaichu, A. (24 de Noviembre de 2014). Design and implementation of kinematics model and trajectory planning for NAO humanoid robot in a tic-tac-toe board game. *2014 IEEE 4th International conference on system engineering and technology (ICSET)*. Recuperado el 5 de Febrero de 2019, de https://ieeexplore.ieee.org/document/7111783

Aldebaran Robotics. (2018). *Technical specifications.* Recuperado el 2019, de Aldebaran documentation: http://doc.aldebaran.com/2-1/family/nao_h25/index_h25.html

Aventin, J. (Junio de 2015). *Red inalámbrica de sensores inerciales para stream de registros goniométricos en biomecánica.* Universitat Oberta de Catalunya. Recuperado el 2018, de https://pdfs.semanticscholar.org/91f8/0f89a6da27fbb8013b77ceb8cf316aebbcff.pdf

Barrientos, A., Penin, L. F., Balaguer, C., & Santoja, R. (2007). *Fundamentals for robotics* (2 ed.). McGraw-Hill/Interamericana.

Bernal, J. (2014). *Diseño, construcción e implementación de un sistema de captura de movimiento para análisis ergonómico de riesgo laboral de extremidades superiores.* Universidad Politécnica Salesiana Sede Cuenca. Recuperado el 2019, de https://dspace.ups.edu.ec/bitstream/123456789/7508/1/UPS-CT004422.pdf

Chalodhorn, R., & Rao, R. (Octubre de 2009). Using eigenposes for lossless periodic puman motion imitation. *2009 IEEE/RSJ International conference on intelligent robots and systems*. Recuperado el 2 de Septiembre de 2018, de https://ieeexplore.ieee.org/document/5354391

Chen, X. (Agosto de 2013). *Human motion analysis with wearable inertial.* University of Tennessee, Knoxville. Recuperado el 20 de Agosto de 2018, de University of Tennessee, Knoxville: https://trace.tennessee.edu/utk_graddiss/2407/

Davidrajuh, R. (Noviembre de 2015). Modeling humanoid robot as a discrete event system: A modular approach on petri nets. *Conference: 2015 3rd International conference on artificial intelligence, modelling & simulation (AIMS)*. Recuperado el 5 de Marzo de 2019, de https://www.researchgate.net/publication/309433294_Modeling_Humanoid_Robot_as_a_Discrete_Event_System_A_Modular_Approach_Based_on_Petri_Nets/citations

Enríquez Rodríguez, K. A. (2019). *Diseño e implementación de un sistema de clasificación de objetos basado en el robot humanoide NAO.* Universidad de las

Fuerzas Armadas ESPE, Sangolqui. Recuperado el 12 de Octubre de 2019, de http://repositorio.espe.edu.ec/xmlui/bitstream/handle/21000/20584/T-ESPE-039662.pdf?sequence=1&isAllowed=y

Fadli, H., Hidayat, E., & Machbub, C. (2016). Design and implementation of walking pattern and trajectory compensator of NAO humanoid robot. *2016 IEEE 6th International conference on system engineering and technology (ICSET)*. Recuperado el 7 de Noviembre de 2019, de https://ieeexplore.ieee.org/document/7849647

GitHub. (2018). *MPU92050 Library*. Recuperado el 15 de Enero de 2018, de Bolder flight systems: https://github.com/bolderflight/MPU9250

Guallichico, J. D., & Utreras, C. A. (2013). *Diseño e implementación de un sistema de navegación inercial tipo STRAPDOWN para estimar la posición de un robot móvil, aplicable a un prototipo de autopiloto de un UAV.* Escuela Politécnica Nacional. Recuperado el 18 de Septiembre de 2019, de https://bibdigital.epn.edu.ec/bitstream/15000/5912/1/CD-4723.pdf

Hofmann, A. G. (2006). *Robust execution of bipedal walking tasks from biomechanical principles.* Massachusetts Institute of Technology. Recuperado el 21 de Agosto de 2019, de http://hdl.handle.net/1721.1/38444

InvenSense. (6 de Junio de 2016). *MPU-9250 Product specification.* Recuperado el 15 de Noviembre de 2018, de http://www.invensense.com/wp-content/uploads/2015/02/PS-MPU-9250A-01-v1.1.pdf

Isacson, J., Gransberg, L., & Knutsson, E. (1986). Three-dimensional electrogoniometric gait recording. *Journal of Biomechanics, 19*(8), 627-629, 631-635. Recuperado el 25 de Julio de 2019, de https://www.sciencedirect.com/science/article/abs/pii/0021929086901685?via%3Dihub

Jin, S., Dai, C., Liu, Y., & Wang, C. (Julio de 2016). Motion imitation based on sparsely sampled correspondence. *Asme journal of computing and information science in engineering*. Recuperado el 14 de Agosto de 2018, de https://www.researchgate.net/publication/305401821_Motion_Imitation_Based_on_Sparsely_Sampled_Correspondence

Kah, T., & Nasiruddin, M. (Octubre de 2017). A study of walking gait stability and gait efficiency of a cost-effective small humanoid bipedal robot: Analysis, simulation and implementation. *2017 IEEE International symposium on robotics and intelligent sensors (IRIS)*. Recuperado el 3 de Septiembre de 2018, de https://ieeexplore.ieee.org/document/8250109

Kiemel, S. (2012). *Balance maintenance of a humanoid robot using the hip-ankle strategy.* Universidad Tecnológica de Delft, BioMechanical Engineering. Recuperado el 23 de Abril de 2019, de http://resolver.tudelft.nl/uuid:9e390296-9ba9-46d5-9b3f-3b7653b2d1ba

Koenemann, J., Burget, F., & Bennewitz, M. (Junio de 2014). Real-time imitation of human Whole-Body motions by humanoids. *2014 IEEE International conference on robotics and automation (ICRA)*. Recuperado el 21 de Agosto de 2018, de https://ieeexplore.ieee.org/document/6907261

Kofinas, N. (2012). *Forward and inverse kinematics for the NAO humanoid robot.* Technical University of Crete. Recuperado el 5 de Octubre de 2019, de https://link.springer.com/article/10.1007/s10846-013-0015-4

Manasrah, A. (2012). Human motion tracking for assisting balance training and control of a humanoid robot. *University of South Florida*. Obtenido de http://scholarcommons.usf.edu/etd/4141

McCarron, B. (2013). *Low-Cost IMU implementation via sensor fusion algorithms in the arduino environment.* California Polytechnic State University. Recuperado el 2019, de https://digitalcommons.calpoly.edu/aerosp/125/

Montalvo López, M. (Abril de 2017). *Programación por demostración del robot Aldebaran NAO H25 de la Universidad Politécnica Salesiana para la enseñanza y preservación de tradiciones, leyendas y expresiones orales del Ecuador.* Recuperado el 24 de Agosto de 2018, de Universidad Politécnica Salesiana: https://dspace.ups.edu.ec/bitstream/123456789/14159/1/UPS-CT006971.pdf

Poubel, L., Sakka, S., Ćehajić, D., & Creusot, D. (Junio de 2014). Support changes during online human motion imitation by a humanoid robot using task specification. *2014 IEEE International conference on robotics and automation (ICRA)*, Junio. Recuperado el 25 de Agosto de 2018, de https://ieeexplore.ieee.org/document/6907092

Prayudi, I., & Kim, D. (Agosto de 2012). Design and implementation of IMU-based Human arm motion capture system. *2012 IEEE International conference on mechatronics and automation*, 670-675. Recuperado el 13 de Agosto de 2018, de https://ieeexplore.ieee.org/document/6283221

Roetenberg, D., Luinge, H., & Slycke, P. (Enero de 2009). *Xsens MVN: Full 6DOF Human motion tracking using miniature inertial sensors.* Recuperado el 15 de Agosto de 2018, de ResearchGate: https://www.researchgate.net/publication/239920367

Ronningen, L. A., Panggabean, M., & Overby, H. (2013). Modeling and simulating motions of human bodies in a futuristic distributed tele-immersive collaboration system for synthesizing transient input traffic. *Simulation modelling practice and theory*(31),

132-148. Recuperado el 2019, de https://www.sciencedirect.com/science/article/abs/pii/S1569190X12001529

Seo, K. (2011). *Using NAO: Introduction to interactive humanoid robots.* Aldebaran robotics.

Suleiman, W., Yoshida, E., Kanehiro, F., Laumond, J.-P., & Monin, A. (Mayo de 2008). On human motion imitation by humanoid robot. *2008 IEEE International conference on robotics and automation*. Recuperado el 25 de Agosto de 2018, de https://www.researchgate.net/publication/224318636_On_Human_Motion_Imitation_by_Humanoid_Robot

Teachasrisaksakul, K., Zhang, Z., Lo, B., & Yang, G.-Z. (Mayo de 2015). Imitation of dynamic walking with BSN for humanoid robot. *IEEE Journal of biomedical and health informatics*, 794 - 802. Recuperado el 20 de Agosto de 2018, de https://ieeexplore.ieee.org/document/7096914

Technaid. (Mayo de 2016). *Conexión recomendada de una red de 16 IMUs para el cuerpo humano.* Recuperado el 13 de Agosto de 2018, de Technaid: http://www.technaid.com/wp-content/uploads/2016/05/Conexio%CC%81n-recomendada-de-una-red-de-16-IMUs-para-el-cuerpo-Humano-ES.pdf

Treffers, C., & Wietmarschen, L. (2016). *Position and orientation determination of a probe with use of the IMU MPU9250 and a ATmega328 microcontroller.* Universidad Técnica de Delft. Recuperado el 2018, de http://repository.tudelft.nl/.

Turner, M., Findley, R., Griffin, W., Cutkosky, M., & Gomez, D. (2000). Development and testing of a telemanipulation system with arm and hand motion. *ASME IMECE DCS-Symposium on haptic interfaces*, 1-8. Recuperado el 7 de Octubre de 2019, de http://www-cdr.stanford.edu/DML/publications/turner_asme00.pdf

Villalobos, E. (2013). *Intrumentación de un robot bípedo de 12 gdl: Sensores de posición, presión e ingercial.* Universidad Autonoma de Mexico. Recuperado el 2019, de http://www.ptolomeo.unam.mx:8080/xmlui/bitstream/handle/132.248.52.100/6835/Tesis.pdf?sequence=1

Vivas Mateos, G. (Septiembre de 2016). *Desarrollo de un método de captación de movimiento humano para el control remoto de terapias de rehabilitación robóticas.* Recuperado el 26 de Agosto de 2018, de Universidad Politécnica de Madrid: http://oa.upm.es/43792/1/TFG_GUILLERMO_VIVAS_MATEOS.pdf

Vuga, R., Ogrinc, M., Gams, A., Petric, T., Sugimoto, N., Ude, A., & Morimoto, J. (6 de Mayo de 2013). Motion capture and reinforcement learning of dynamically stable humanoid movement primitives. *2013 IEEE International conference on robotics and automation (ICRA)*. Recuperado el 5 de Marzo de 2018, de https://ieeexplore.ieee.org/document/6631333

Wang, F., Tang, C., Ou, Y., & Xu, Y. (2012). A real-time human imitation system. *World congress on intelligent control and automation*. Recuperado el 2019, de https://ieeexplore.ieee.org/document/6359088

Wemos. (2018). *LoLin32 A Wifi&Bluetooth board based ESP-32*. Obtenido de https://wiki.wemos.cc/products:lolin32:lolin32

Wenk, F. (27 de Septiembre de 2016). *Inertial motion capturing rigid body pose and posture estimation with inertial sensors.* Recuperado el 10 de Septiembre de 2018, de Universität Bremen: https://d-nb.info/1126093440/34

Yi, S., Zhang, B., Hong, D., & Lee, D. (2013). Online learning of low dimensional strategies for high-level push recovery in bipedal humanoid robots. *2013 IEEE International conference on robotics and automation (ICRA)*. Recuperado el 2019, de https://ieeexplore.ieee.org/document/6630791

Zavala, J., Guzmán, S., & Galván, J. (2013). Cinemática inversa, Fanuc LR Mate 200ic. *Pistas educativas*(103).

ANEXOS