UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Control para estabilidad estática de un robot bípedo de 10 GDL

TESIS

Que para obtener el título de

Ingeniero Mecatrónico

P R E S E N T A

Luis Ulises García Aguilera

DIRECTOR DE TESIS

M. en I. Serafín Castañeda Cedeño

Ciudad Universitaria, Cd. Mx., 2021

La combinación correcta
de tecnologías y humanos
impulsará la prosperidad
- Ulrich Spiesshofer

Agradecimientos

A mi madre y a mi padre, Yael y Luis, por todo el apoyo brindado a lo largo de mi vida, siempre he estado profundamente agradecido por su amor y sus consejos. En cada uno de mis logros veo reflejados su esfuerzo y dedicación.

A mi hermano, Gael, por estar presente como un gran amigo y compañero, por todas las experiencias que hemos compartido y por tu cariño. Siempre puedes contar conmigo.

A mis amigas y amigos por cada uno de los aprendizajes que me brindaron y que me han permitido crecer en los ámbitos académico, profesional y personal.

Al maestro Serafín Castañeda por la confianza para desarrollar este trabajo y por la sabia elección de palabras para orientarme en esta etapa.

A cada persona involucrada en mi trayectoria: profesoras, profesores, compañeras, compañeros, familiares, transportistas, personal de limpieza, vendedores de comida, investigadores, entre muchas otras. Valoro cada uno de sus aportes para realizar mis estudios. Sus contribuciones son de suma importancia para la sociedad.

Índice

Resumen

Este trabajo presenta el diseño y la implementación de un control para lograr la estabilidad de postura de un robot bípedo de 10 GDL. Se abordan los antecedentes históricos de la robótica humanoide y se describen los conceptos relevantes de la locomoción bípeda. Se resuelven los problemas de cinemática directa e inversa para el robot; para la cinemática inversa se consideran 4 GDL por pierna. Se propone un controlador LQR (regulador cuadrático lineal) para cumplir con el criterio de estabilidad estática a partir del modelo dinámico simplificado de un péndulo invertido. Además, se describen las modificaciones hechas al robot y el algoritmo para implementar el controlador. Por último, se muestran y analizan los resultados obtenidos en las pruebas.

Abstract

This work presents the design and implementation of a control for reaching the posture stability of a 10 DOF biped robot. Historical background on humanoid robotics is approached and relevant concepts on biped locomotion are described. The direct and inverse kinematics problems are solved; for the inverse kinematics, 4 DOF per leg are considered. To accomplish the static stability criterion, an LQR (linear quadratic regulator) controller from the simplified dynamic model of an inverted pendulum is proposed. Also, the modifications made to the robot and the algorithm to implement the controller are described. Finally, the results obtained in the tests are shown and analyzed.

Objetivo

Diseñar e implementar un controlador LQR para lograr la estabilidad de postura de un robot bípedo de 10 GDL utilizando un modelo dinámico simplificado.

Alcance

El presente trabajo se enfoca en conseguir la estabilidad de postura de un robot bípedo de 10 GDL. El problema se abordó en las etapas enunciadas a continuación:

- Modelado
  En esta etapa se describen las características del robot bípedo Bioloid GP, se obtienen la cinemática directa e inversa del robot como el modelo dinámico simplificado. La cinemática inversa considera únicamente cuatro grados de libertad por pierna y hace una simplificación a la geometría de cada pierna del robot en el plano sagital. El modelo dinámico corresponde al de un péndulo invertido considerando que la altura de la masa concentrada es constante.

- Diseño del controlador
  Luego de obtener el modelo dinámico del sistema, se diseña un controlador LQR considerando que el objetivo de control a lograr es el de estabilización. Asimismo, se simula el modelo simplificado para verificar la respuesta esperada del sistema retroalimentado.

- Implementación
  En la etapa de implementación se proporcionan las características principales de los componentes electrónicos empleados para lograr el objetivo de control y se describe el algoritmo codificado en el microcontrolador para realizar las pruebas.

- Pruebas
  La etapa final corresponde a las pruebas sobre el robot bípedo. Éstas se realizan sobre una superficie plana y en una plataforma móvil con posibilidad de inclinarse en cualquier dirección mientras el robot se soporta sobre ambos pies. Se analiza la respuesta del controlador propuesto ante diferentes perturbaciones.

1. Robótica bípeda

1.1. Historia de la robótica

Los límites de la creatividad y de la imaginación de la humanidad parecen aún inalcanzables. Desde la antigüedad, los humanos han dejado vestigios de su imaginación en numerosas obras de arte, ya sea en pinturas, esculturas o textos. La capacidad de construir y materializar las nuevas ideas propició que la creatividad no dejara de crecer y que cada vez más desarrollos fueran realizados. Hoy, en pleno siglo XXI, muchos de los avances tecnológicos que se han logrado parecen irreales.

A lo largo de la historia, los artesanos se han inspirado en los movimientos y en el comportamiento humano para muchas de sus creaciones. Los juguetes son un elemento de la cultura actual en que se hace visible la existencia de estas inspiraciones, pero la ciencia ficción se ha convertido en el más importante y creativo promotor de conjugar características humanas en nuevas creaciones. Entre las creaciones surgidas como parte de una necesidad de la sociedad reflejada en las expresiones artísticas se encuentran los robots. Desde 1927, en la película Metrópolis se muestra la creación de una mujer-máquina con una apariencia bastante similar a la humana y que, más adelante en la filmación, es representada por una actriz. Cada vez es mayor el número de películas en que se han mostrado los desarrollos tecnológicos a los que se pretende llegar por medio de la investigación; Star Wars es una de las películas más destacadas en este ámbito y de ella surgen dos robots icónicos: R2D2 y C3P0 [1] [2] [3].

Figura 1.1 C3P0 y R2D2, icónicos robots de la película Star Wars [4].

Algunos autores consideran a la robótica como un arte, y es justamente en una obra dramática donde surge la palabra robot. En 1920, Karel Capek escribió la obra teatral de ciencia ficción R.U.R. (Rossum's Universal Robots), que sería estrenada en el Teatro Nacional de Praga en 1921 y en Nueva York en 1922. En esta obra, se muestra una empresa que construye humanos artificiales para que realicen el trabajo de las personas; sin embargo, estos entes llamados robots más tarde hacen una revolución dando fin a la existencia de la humanidad. El significado de la palabra *robota* es trabajo pesado, haciendo

referencia al trabajo realizado por los obreros en las fábricas, mientras que Capek utiliza este término checo para referirse a los seres construidos para realizar las tareas humanas, apareciendo así la primera connotación de robot en la historia [5].

Justamente, el auge de los robots estuvo y, aún en la actualidad, está en las fábricas. La evolución de la producción consistió en pasar de lo manual a lo automático, destacando la producción en masa desarrollada por Ford en 1905, donde se contaba con máquinas de propósito especial. Más adelante, la Segunda Guerra Mundial trajo consigo nuevas necesidades en diferentes industrias, destacando las máquinas automáticas en el incremento de la producción y las máquinas de control numérico en la calidad de los productos. Las industrias espacial y nuclear propiciaron el desarrollo de manipuladores de varios grados de libertad para trabajar con materiales peligrosos y en entornos hostiles. En primera instancia, implementar el control numérico en los manipuladores propició el surgimiento de los robots [2] [6].

Los desarrollos tempranos de la robótica se dieron en programas de investigación. En la década de 1940, se avanzó en los manipuladores mecánicos controlados remotamente, anteriormente mencionados como precursores de los robots; estos mecanismos eran del tipo maestro-esclavo. En los años 50, se remplazó el acoplamiento mecánico de estos sistemas por potencia eléctrica e hidráulica en manipuladores como el Handyman de General Electric y el Minotaur I de General Mills. Tras la mejora de estos dispositivos para lograr operaciones autónomas y repetitivas, la primera muestra tangible de estos avances la dio George C. Devol en 1954, creando el primer robot programable, capaz de seguir una secuencia de movimientos determinada por las instrucciones introducidas en un programa. Trabajando en este concepto, Unimation Inc. introdujo el primer robot industrial en 1959. En los años 60, la retroalimentación mediante sensores permitió un gran progreso en las aplicaciones de la robótica [1].

1.2. Historia de los robots humanoides

Muchos investigadores concentraron sus esfuerzos en el notable potencial de la robótica industrial; sin embargo, algunos grupos realizaron grandes desarrollos en la robótica humanoide. La mayoría de los grupos dedicados a la investigación de robots humanoides se han concentrado en Japón, donde surgió el grupo pionero en la Universidad Waseda. Un gran número de las empresas con mayor presencia en el mundo en cuestiones referentes a la robótica residen en Japón, como Honda. Además, con el paso de los años, los avances se han dado en cada vez más grupos y de manera más constante para su implementación en diferentes áreas [7].

1.2.1 Desarrollos en la Universidad Waseda

En la Universidad Waseda, desde hace varias décadas, se ha tenido un amplio progreso en robótica humanoide, realizando investigación principalmente en la marcha de robots bípedos, en brazos y piernas artificiales, en el modelo de control de postura, en sistemas de diseño por computadora para robótica, entre otros temas.

En 1970, el proyecto WABOT (WAseda roBOT) fue iniciado por el profesor Ichiro Kato y sus colegas. Posteriormente, en 1973, presentaron el primer robot antropomórfico a escala completa, el WABOT-1, que consistía en un sistema de control de extremidades, un sistema de visión y un sistema de conversación. Era capaz de comunicarse con una persona en japonés y de medir tanto distancias como direcciones de los objetos por medio de receptores externos como oídos y ojos artificiales. El WABOT-1 estaba formado por el WAM-4 y el WL-5, brazos y piernas artificiales, respectivamente. El robot lograba caminar con sus extremidades inferiores y podía sostener y transportar objetos con sus manos, estimando que tenía la facultad mental de un niño de un año y medio. En 1980, comenzaron el proyecto del WABOT-2, siendo un robot "especialista" que podía conversar con una persona, leer una partitura y tocar melodías de dificultad media en un órgano eléctrico [8] [9].

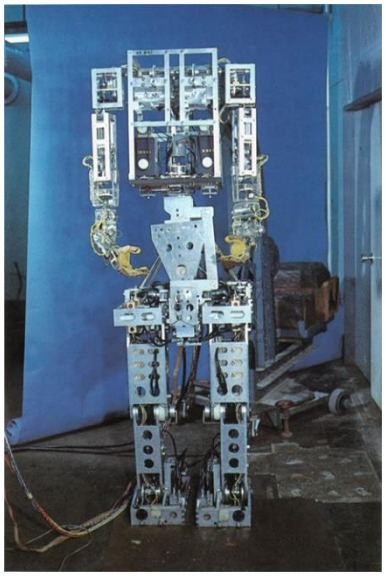


Figura 1.2 WABOT-1, desarrollado en la Universidad Waseda, fue el primer robot antropomórfico a escala completa [8]

Más adelante, trabajaron en un nuevo diseño con la serie WABIAN (WAseda BIpedal humANoid). El WABIAN se desarrolló siguiendo un plan de diseño que permitiera investigar la marcha dinámica cooperativa y el trabajo colaborativo con humanos. Las características principales con que debía contar eran: tamaño de una mujer japonesa adulta promedio, capacidad de caminar a una velocidad aproximada a la humana, tres grados de libertad en el torso y seis en los brazos, uso de servomotores para las articulaciones e instalación de una computadora de control y los drivers de los motores. Se consiguieron diversas caminatas como la dinámica hacia adelante y hacia atrás, la estática, bailando, llevando una carga, etc. Una versión posterior, el WABIAN-RV realizaba movimientos de marcha por medio de la generación en línea de patrones de movimiento para cada paso, mejorando la adaptabilidad al ambiente de trabajo [9].

La última versión de esta serie es el WABIAN-2R, que fue diseñado con 1500 [mm] de altura y 64 [kg] de peso. El objetivo de este robot es imitar los movimientos humanos,

por lo que cuenta con 41 GDL y sus articulaciones poseen un rango de movimiento diseñado tomando como referencia el rango de los humanos. Entre las novedades implementadas a lo largo del desarrollo del WABIAN-2R se encuentran un mecanismo de cintura de dos grados de libertad, un pie con una junta de tobillo pasiva, un mecanismo de pie que imita la estructura en arco del pie humano, un mecanismo de piernas imitando la marcha en un plano frontal y horizontal, entre otras [9].

1.2.2 Desarrollos en Honda

La empresa japonesa Honda comenzó sus trabajos de investigación en robótica bípeda humanoide en 1986 observando todas las formas de marcha, realizando numerosos experimentos y recolectando una gran cantidad de datos. En ese año, el E0 (*E* de electrónica) caminaba colocando una pierna anteponiéndose a la otra, con pasos que duraban cerca de cinco segundos y sólo podía caminar lentamente y en línea recta [7].

Entre los años 1987 y 1991, Honda trabajó en un diseño más parecido al humano y en tecnología para una caminata más rápida. La empresa se enfocó en la investigación y el análisis tanto de la caminata humana como de otros tipos de caminata. Se estudió el movimiento y la localización de las juntas necesarias para la caminata. Los desarrollos en este período incluyeron los robots E1, E2 y E3. Fue hasta el modelo E2 cuando se logró una caminata rápida, llegando a una velocidad de 1.2 [km/h]. Además, el E3 llegó a una velocidad de 3 [km/h] con sus piernas que asemejaban los muslos humanos. En los años posteriores, hasta 1993, el desarrollo de los modelos E4, E5 y E6 se centró en la tecnología para lograr una marcha estable y se generaron tres controles de postura de forma exitosa [10].

De 1993 a 1997, con los modelos P1, P2 y P3, la investigación consistió en evolucionar hacia un robot humanoide que contara con parte superior del cuerpo. Los estudios pretendían determinar la apariencia de un robot humanoide para funcionar apropiadamente en sociedad y en un entorno humano. El P1 fue el primer prototipo de un modelo semejando a un humano con extremidades superiores y un cuerpo. El modelo que es considerado como el más revolucionario de la industria es el modelo P2, presentado en 1996. Este modelo, además de tener un tamaño humano, era capaz de realizar movimientos humanos de manera muy realista. Los diferentes grupos de investigación en robótica humanoide vieron en este modelo la transición de esta área de un simple objetivo futuro a una realidad, lo que impulsó a que la industria tuviera un mayor auge [7] [10].

En el año 2000, nació ASIMO, robot con un tamaño adecuado tanto para operar en un entorno humano como para ser amigable con los humanos. Su nombre deriva del inglés de la expresión “paso avanzado en movilidad innovadora” (*Advanced Step in Innovative MObility*). Realizaba una marcha suave y natural más aproximada a la humana gracias a su tecnología i-WALK, sistema de caminata flexible inteligente en tiempo real. Los movimientos de los brazos de ASIMO fueron expandidos y su operabilidad se mejoró con un controlador portátil [11].

En 2002, ASIMO obtuvo la capacidad de interpretar posturas y gestos humanos y de responder de forma autónoma a estos estímulos, además se integró a Internet y otras redes. En 2004, se mejoró con tecnología que le permitía reconocer situaciones de la vida real y actuar de manera rápida como respuesta, los movimientos fueron más veloces y trabajaban en sincronía con los humanos. 2005 fue el año en que ASIMO mejoró sus funciones para operar con humanos como sostener la mano de una persona mientras camina y cargar objetos en una bandeja o un carro [11].

Figura 1.3 ASIMO, último robot humanoide desarrollado por Honda [11].

La última presentación de ASIMO fue en 2011, cuando el robot ganó mayor autonomía y adaptabilidad a diferentes situaciones. El robot medía 130 [cm] y pesaba 48 [kg], la velocidad máxima que lograba era de 9 [km/h]. Honda desarrolló tecnología para inteligencia avanzada de ASIMO, que evalúa las entradas de múltiples sensores equivalentes a los sentidos humanos de la vista, el oído y el tacto, para luego estimar la situación del entorno y determinar el comportamiento adecuado del robot. Se implementaron una combinación de piernas fortificadas, un rango de movimiento de piernas expandido y una nueva tecnología de control para que ASIMO pudiera caminar, correr, correr de espaldas, saltar en una o dos piernas continuamente y caminar en superficies irregulares. También se desarrolló una mano con dedos, que cuenta con sensores táctiles y un sensor de fuerza embebido en la palma y en cada dedo que sirve para controlar cada dedo de manera independiente [11].

1.2.3 Otros desarrollos en robótica humanoide

Desde que los precursores de la robótica bípeda dieron pasos enormes en esta área, han sido múltiples los desarrollos que se han dado alrededor del mundo. Otro desarrollo japonés con importantes logros fue el HRP (Humanoid Robot Project), iniciado en 1998. Este proyecto estaba a cargo de Kawada Industries en colaboración con el AIST (Japan's National Institute of Advanced Industrial Science and Technology). El modelo HRP-2 tenía un diseño futurista, podía cooperar con humanos en el levantamiento de objetos pesados, caminar en superficies irregulares y levantarse por sí mismo. El modelo HRP-4 es el último modelo realizado en alianza, los ingenieros de Kawada se enfocaron en el *hardware*,

mientras los investigadores del AIST desarrollaron el *software* de control de movimiento [7] [12].

En 1997, surgió el proyecto Robonaut con el objetivo de desarrollar un robot humanoide capaz de asistir astronautas con tareas de manipulación. La NASA (National Aeronautics and Space Administration) y General Motors se aliaron para desarrollar la segunda versión del Robonaut. El R2 fue anunciado en 2010, impresionando tanto a los directores de misión que decidieron hacer espacio en un lanzamiento para enviar el robot a la Estación Espacial Internacional. Su objetivo principal es demostrar cómo se comportan los robots en el espacio, pero se espera que pueda suplir a los astronautas en labores peligrosas o repetitivas [12].

Atlas es un robot humanoide anunciado como el más ágil en existencia. Es un robot de la empresa norteamericana Boston Dynamics, presentado en 2016. Usa habilidades de cuerpo completo para moverse rápidamente y balancearse dinámicamente. Puede levantar y cargar objetos, además de correr, saltar y hacer piruetas [12].

Agility Robotics es una empresa norteamericana dedicada a los robots bípedos. Aunque su primer lanzamiento fue en 2016 con Cassie, en 2019, trabajando en alianza con Ford, presentaron su segundo robot, Digit, equipado con un torso lleno de sensores y un par de brazos, usados para lograr balance, movilidad y manipulación. La idea de Ford es implementar un servicio de entrega de paquetería autónomo en el que Digit será el encargado de hacer llegar los paquetes a la puerta de las casas [12] [13].

Figura 1.4 Digit, robot bípedo de Agility Robotics en colaboración con Ford, fue ideado para la entrega de paquetería [13].

Además de los desarrollos en investigación e industriales, se tienen desarrollos más interactivos como Bruno, un robot jugador de futbol desarrollado en Alemania, haciendo de su equipo uno de los mejores en el torneo organizado en la RoboCup. Recientemente ha sido de gran interés la educación, donde se manejan diferentes modelos que permiten tanto el aprendizaje de niños y jóvenes como los desarrollos de investigación con menores presupuestos. En esta área, se tienen robots como Nao de la empresa francesa SoftBank

Robotics y Darwin-OP de la surcoreana Robotis [12]. Dentro de esta categoría se encuentra el Bioloid GP, también de Robotis, que se utilizó para la realización de este trabajo.

1.3. Preliminares de locomoción bípeda

Los robots bípedos están diseñados para lograr imitar la caminata humana [14]. Con la finalidad de estudiar su movimiento, se requiere conocer algunos términos específicos. A continuación, se definen los términos útiles para el desarrollo del presente trabajo.

1.3.1 Sistema de referencia

Para describir los movimientos de un robot bípedo se emplean tres planos perpendiculares a través del cuerpo [14]. Estos planos se muestran en la Fig. 1.5 y se definen como:

- Plano frontal: plano paralelo al plano *yz*, divide el cuerpo en anterior y posterior, es decir, parte trasera y frontal, respectivamente.
- Plano sagital: plano paralelo al plano *xz*, divide el cuerpo en lados derecho e izquierdo. Existe un plano paralelo al plano sagital que contiene al centro de masa llamado plano medial.
- Plano transversal: plano paralelo al plano *xy*, divide al cuerpo en mitad superior e inferior. [14] [15]

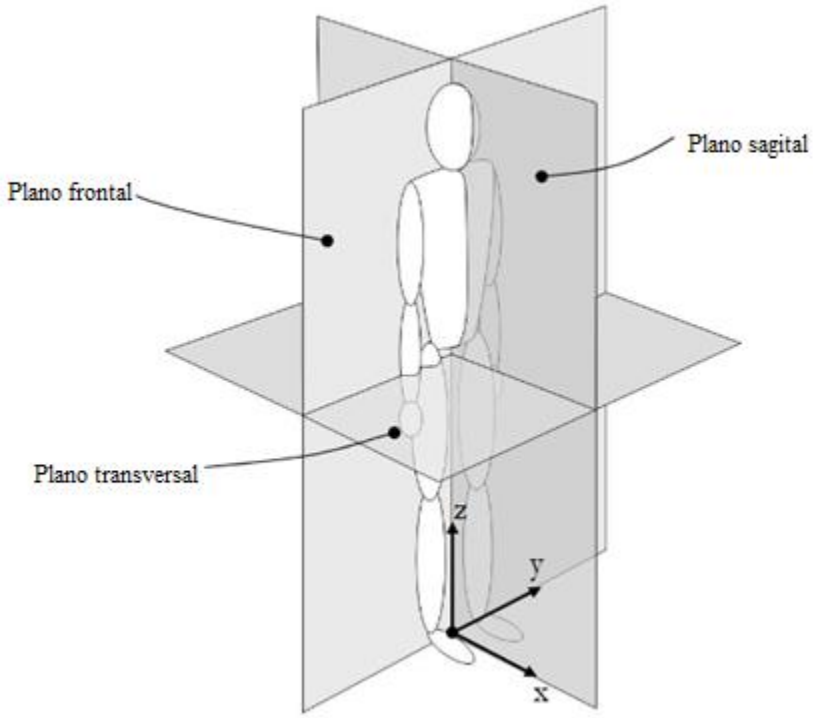


Figura 1.5 Planos perpendiculares al cuerpo de un robot bípedo y sistema de referencia [14].

En cada uno de estos planos actúa un ángulo diferente que describe la orientación del robot. Los ángulos de *roll*, *pitch* y *yaw* ocurren en los planos frontal, sagital y transversal, respectivamente.

1.3.2 Centro de masa

Un robot bípedo está sometido a la fuerza gravitacional todo el tiempo, al punto donde actúa una fuerza equivalente a las fuerzas gravitacionales que experimenta cada una de las partes del robot se le conoce como centro de masa (CoM) [14].

1.3.3 Polígono de soporte

El polígono de soporte se forma por la superficie alrededor de los puntos de apoyo del robot en el suelo [14]. El polígono de soporte cambia según el apoyo, ya que pueden estar apoyados ambos pies o solo uno; en la Fig. 1.6 se representa el polígono de soporte para tres casos:

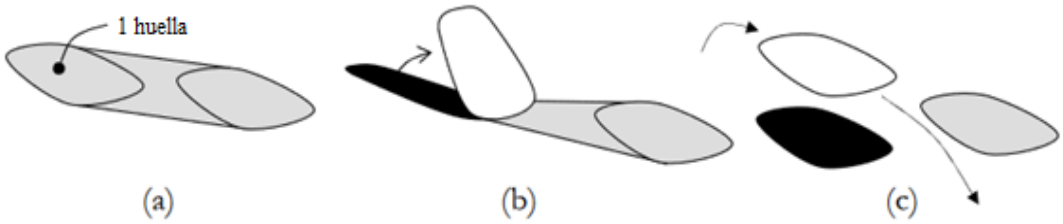


Figura 1.6 Figuras típicas del polígono de soporte marcadas en gris: (a) soporte doble, (b) soporte doble en fase previa al balanceo, (c) soporte simple [14].

1.3.4 Criterio de estabilidad estática

Si la proyección sobre el suelo del centro de masa de un robot bípedo se encuentra dentro de la superficie que describe el polígono de soporte, se cumple la estabilidad estática del robot [16]. Empleando este criterio se asegura que el robot no caerá al permanecer en una posición determinada.

2. Modelo

2.1. Bioloid

El Bioloid GP es un robot humanoide bípedo desarrollado por la empresa surcoreana Robotis, forma parte de la serie Bioloid, cuyo nombre hace referencia a la unión de las palabras en inglés *Bio, All* y *Droid.* El Bioloid es un kit educacional que permite el aprendizaje de los fundamentos de estructuras y los principios de las juntas robóticas. Se puede expandir su aplicación a la ingeniería, por medio de la cinemática inversa y la cinética. También está pensado en aficionados que construyen modelos personalizados y es un kit de robótica ampliamente utilizado en diversas competencias [17].

En el manual proporcionado por Robotis, se describe que el modelo cuenta con resistentes y ligeros marcos de aluminio, tiene una excelente movilidad que le permite girar mientras camina y realizar caminata de alta velocidad, además, es capaz de realizar movimientos básicos de humanos, ejemplificados por sus modos de operación para combate y para jugar fútbol. Realiza autocorrección de posición por medio de un giroscopio, funcionalidad proveniente de fábrica y no utilizada para el desarrollo de este trabajo. Su controlador principal es un CM-530, cuenta con un giroscopio y un sensor de distancia (DMS), utiliza servomotores DYNAMIXEL AX-12A para las pinzas y la parte superior del cuerpo, y AX-18A para la parte inferior. Además, cuenta con una batería LIPO 11.1V y un control remoto RC-100B que opera con un módulo de comunicación inalámbrica ZIG-110A [17] [18].

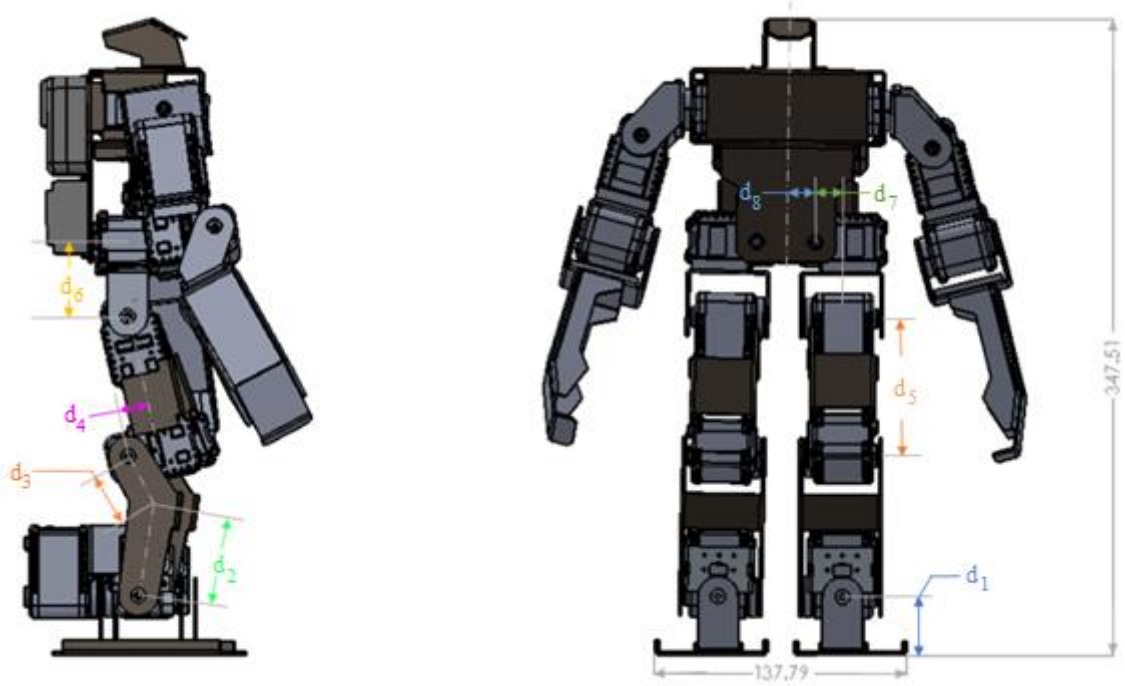


Figura 2.1 Dimensiones del robot bípedo Bioloid GP de Robotis.

En la Fig. 2.1 se muestran las dimensiones generales del modelo y las distancias de los eslabones para la cinemática directa del robot indicadas como d_n. La Figura muestra únicamente las medidas del lado derecho d_n y, por simetría, se conocen las del lado izquierdo i_n, donde n toma valores de 1 hasta 8. Los valores de los parámetros son los siguientes:

$$d_1 = i_1 = 32.7\ [mm]$$

$$d_2 = i_2 = 53.03\ [mm]$$

$$d_3 = i_3 = 26.06\ [mm]$$

$$d_4 = i_4 = 16.12\ [mm]$$

$$d_5 = i_5 = 74.28\ [mm]$$

$$d_6 = i_6 = 41.09\ [mm]$$

$$d_7 = i_7 = 19.36\ [mm]$$

$$d_8 = i_8 = 16\ [mm]$$

2.2. Cinemática

El análisis de posición del efector final de un robot manipulador tiene gran importancia para lograr que el robot realice una tarea. Existen dos tipos de problema en el análisis de posición: la cinemática directa y la cinemática inversa. Por un lado, en la cinemática directa, los valores de las variables correspondientes a cada articulación son proporcionados y el problema está en encontrar la posición del efector final. Por otro lado, en la cinemática inversa se tiene la posición del efector final y se requiere encontrar los valores para cada articulación que permitan alcanzar dicha posición [6].

Para el análisis en este trabajo, se considera que al Bioloid GP únicamente como un robot bípedo; de esta manera, se sólo se analizan las piernas, despreciando el papel que juegan el torso y los brazos del robot.

2.2.1 Cinemática directa

El método Denavit-Hartenberg es uno de los más utilizados para el análisis de posición debido a que es sistemático. En este método, se definen sistemas de referencia unidos a cada eslabón del manipulador. Exceptuando los eslabones de la base y del efector final, el sistema de referencia *i* está unido al eslabón *i* y sigue tres reglas: (1) el eje z_i está alineado al eje de la junta *i+1;* (2) el eje *x* está definido por la normal común entre los ejes de las juntas *i* e *i+1*, si estos ejes son paralelos, el eje *x* puede elegirse en cualquier posición manteniendo la perpendicularidad con ambos ejes; y (3) el eje *y* se determina mediante la regla de la mano derecha [6].

En [6] se presenta un procedimiento de seis pasos para establecer los sistemas de referencia siguiendo la convención de Denavit-Hartenberg. Usualmente, para un robot de *n* grados de libertad (GDL) se definen *n+1* sistemas de referencia; sin embargo, pueden

definirse sistemas adicionales relacionándolos con los correspondientes a los grados de libertad mediante una matriz de transformación.

Se puede establecer una relación entre dos sistemas de referencia sucesivos mediante una matriz de transformación. Tras una serie de rotaciones y traslaciones puede llegarse del sistema *i-1* al sistema *i*. Las transformaciones sucesivas son las siguientes:

- Traslación a lo largo del eje z_{i-1} una distancia d_i.

$$T(z,d) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

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

$$T(z,\theta) = \begin{bmatrix} c\theta_i & -s\theta_i & 0 & 0 \\ s\theta_i & c\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Traslación a lo largo del eje x_i una distancia a_i.

$$T(x,a) = \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Rotación alrededor del eje x_i un ángulo α_i.

$$T(x,\alpha) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c\alpha_i & -s\alpha_i & 0 \\ 0 & s\alpha_i & c\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Donde $c\theta_i = \cos(\theta_i)$, $s\theta_i = \sin(\theta_i)$, $c\alpha_i = \cos(\alpha_i)$ y $s\alpha_i = sin(\alpha_i)$. Entonces, la matriz de transformación resultante queda como:

$$A_i^{i-1} = T(z,d)\,T(z,\theta)\,T(x,a)\,T(x,\alpha) \qquad (2.1)$$

Al expandir la ecuación (2.1) se obtiene la matriz de transformación de Denavit-Hartenberg (D-H):

$$A_i^{i-1} = \begin{bmatrix} c\theta_i & -c\alpha_i s\theta_i & s\alpha_i s\theta_i & a_i c\theta_i \\ s\theta_i & c\alpha_i c\theta_i & -s\alpha_i c\theta_i & a_i s\theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2.2)$$

Una vez mostradas las transformaciones que corresponden a la matriz de transformación D-H, es posible plantear las matrices desde la base del manipulador hasta el efector final. Para los fines de este trabajo se considera que los pies son la base y que el efector final corresponde a la cadera del robot bípedo, ubicando el origen del sistema de

referencia en medio de ambos pies. En las tablas 2.1 y 2.2 se presentan los parámetros para las matrices de D-H que expresan la cinemática directa del Bioloid GP para la pierna derecha y la pierna izquierda, respectivamente.

Tabla 2.1 Parámetros de Denavit-Hartenberg para la pierna derecha.

Junta i	d_i	θ_i	a_i	α_i
1	0	90°	$-pd$	0°
2	d_1	0°	0	90°
3	0	$\theta_{d1} - 90°$	0	90°
4	0	θ_{d2}	$-d_2$	0°
5	0	33.8°	$-d_3$	0°
6	0	$\theta_{d3} + 56.2°$	d_4	0°
7	0	$-90°$	$-d_5$	0°
8	0	θ_{d4}	$-d_6$	0°
9	$-d_7$	0°	0	$-90°$
10	0	$\theta_{d5} + 90°$	d_8	0°

Tabla 2.2 Parámetros de Denavit-Hartenberg para la pierna izquierda.

Junta i	d_i	θ_i	a_i	α_i
1	0	90°	pi	0°
2	i_1	0°	0	90°
3	0	$\theta_{i1} - 90°$	0	90°
4	0	θ_{i2}	$-i_2$	0°
5	0	33.8°	$-i_3$	0°
6	0	$\theta_{i3} + 56.2°$	i_4	0°
7	0	$-90°$	$-i_5$	0°
8	0	θ_{i4}	$-i_6$	0°
9	i_7	0°	0	$-90°$
10	0	$\theta_{i5} + 90°$	$-i_8$	0°

El robot bípedo cuenta con 5 GDL por pierna, sin embargo, se utilizan cinco sistemas de referencia adicionales para generar la cinemática completa del robot a partir del método de Denavit-Hartenberg. Al multiplicar las matrices de transformación de manera consecutiva desde la primera hasta la última articulación, se obtiene la matriz de transformación desde el sistema de referencia fijo en el suelo hasta el sistema móvil en el efector final. Esto queda representado para cada una de las piernas como:

$$D_{10}^{0} = D_{1}^{0}D_{2}^{1}D_{3}^{2}D_{4}^{3}D_{5}^{4}D_{6}^{5}D_{7}^{6}D_{8}^{7}D_{9}^{8}D_{10}^{9} \qquad (2.3)$$

$$I_{10}^{0} = I_{1}^{0}I_{2}^{1}I_{3}^{2}I_{4}^{3}I_{5}^{4}I_{6}^{5}I_{7}^{6}I_{8}^{7}I_{9}^{8}I_{10}^{9} \qquad (2.4)$$

donde D_i^j corresponde a las matrices de transformación para la pierna derecha e I_i^j a las de la pierna izquierda.

2.2.2 Cinemática inversa

Con la finalidad de simplificar el análisis para la cinemática inversa se consideró que de los 5 GDL del robot únicamente actuarán cuatro, descartando el grado de libertad correspondiente al motor más cercano a la cadera y dejando su valor fijo en 0°. Este grado de libertad se representó como θ_{d5} y θ_{i5} para cada una de las piernas y corresponde a una rotación sobre el plano frontal. La cinemática inversa resolverá los ángulos de las articulaciones para obtener la posición deseada $P = (Px, Py, Pz)$ con una inclinación en plano sagital ϕ.

Primero, considerando la pierna derecha, se resuelve el primer grado de libertad θ_{d1} a partir de modificar la ecuación (2.3) como sigue:

$$(D_{1}^{0}D_{2}^{1}D_{3}^{2})^{-1}D_{10}^{0} = D_{4}^{3}D_{5}^{4}D_{6}^{5}D_{7}^{6}D_{8}^{7}D_{9}^{8}D_{10}^{9} \qquad (2.5)$$

Los componentes (3,4) de las matrices resultantes de la ecuación (2.5) permiten obtener la siguiente igualdad:

$$-(pd + Py)\cos(\theta_{d1}) + (d_1 - Pz)\sin(\theta_{d1}) = -(d_7 + d_8) \qquad (2.6)$$

Definiendo:

$$a_1 = d_1 - P_z$$

$$b_1 = -(p_d + P_y)$$

$$c_1 = -(d_7 + d_8)$$

La expresión (2.6) se escribe como:

$$a_1 \sin(\theta_{d1}) + b_1 \cos(\theta_{d1}) = c_1$$

Se divide todo entre $\sqrt{{a_1}^2 + {b_1}^2}$:

$$\frac{a_1}{\sqrt{{a_1}^2 + {b_1}^2}}\sin(\theta_{d1}) + \frac{b_1}{\sqrt{{a_1}^2 + {b_1}^2}}\cos(\theta_{d1}) = \frac{c_1}{\sqrt{{a_1}^2 + {b_1}^2}}$$

Considerando la identidad trigonométrica: $\sin^2\alpha_1 + \cos^2\alpha_1 = 1$; ésta se satisface si: $\sin\alpha_1 = \frac{a_1}{\sqrt{{a_1}^2+{b_1}^2}}$ y $\cos\alpha_1 = \frac{b_1}{\sqrt{{a_1}^2+{b_1}^2}}$. Además, se puede obtener α_1 mediante la relación

trigonométrica $\alpha_1 = atan2(a_1, b_1)$ tras obtener la tangente como el cociente del seno y el coseno de α_1. Así, la ecuación (2.8) queda como sigue:

$$\sin(\theta_{d1}) \sin \alpha_1 + \cos(\theta_{d1}) \cos \alpha_1 = \frac{c_1}{\sqrt{{a_1}^2 + {b_1}^2}}$$

que mediante la identidad trigonométrica del coseno de la diferencia de dos ángulos es:

$$\cos(\theta_{d1} - \alpha_1) = \frac{c_1}{\sqrt{{a_1}^2 + {b_1}^2}}$$

Por lo tanto:

$$\theta_{d1} - \alpha_1 = atan2(\sqrt{1 - \frac{{c_1}^2}{{a_1}^2 + {b_1}^2}}, \frac{c_1}{\sqrt{{a_1}^2 + {b_1}^2}})$$

Finalmente, el ángulo correspondiente al primer grado de libertad se resuelve como:

$$\theta_{d1} = atan2(a_1, b_1) + atan2(\sqrt{1 - \frac{{c_1}^2}{{a_1}^2 + {b_1}^2}}, \frac{c_1}{\sqrt{{a_1}^2 + {b_1}^2}}) \qquad (2.7)$$

donde $a_1 = d_1 - Pz$, $b_1 = -(pd + Py)$ y $c_1 = -(d_7 + d_8)$.

Para los tres grados de libertad restantes, θ_{d2}, θ_{d3} y θ_{d4}, se considera que forman un manipulador plano de 3 GDL. Los grados de libertad corresponden a la ubicación en el plano y a la inclinación del efector final. La Fig. 2.2 muestra geometría del robot bípedo para los grados de libertad mencionados.

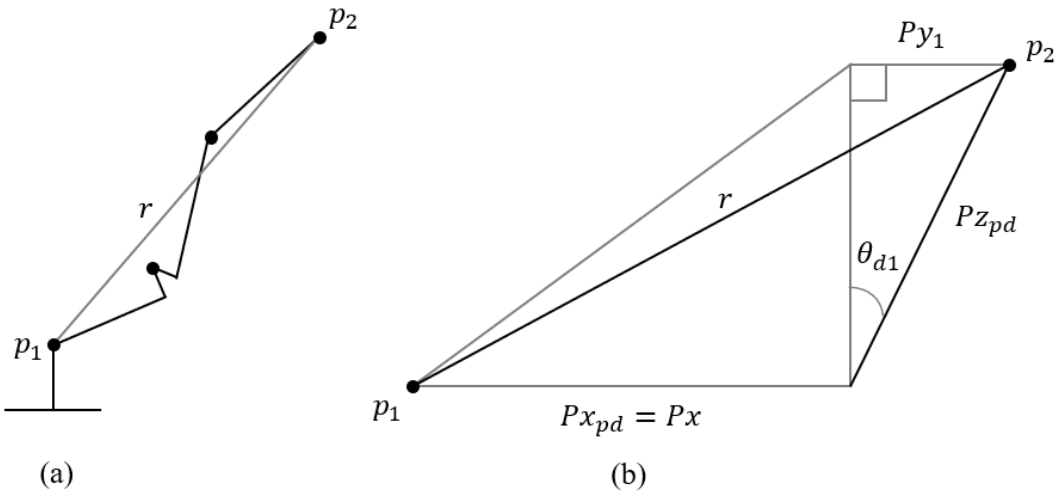


Figura 2.2 Geometría del robot bípedo: (a) vector *r* de p_1 a p_2 pasando por las articulaciones correspondientes a los ángulos θ_{d2}, θ_{d3} y θ_{d4}, (b) triángulo formado por el vector *r* inclinado un ángulo θ_{d1} del plano XZ.

A partir de la Fig. 2.2 se conocen los parámetros para resolver la cinemática inversa del manipulador plano:

$$Px_{pd} = Px \quad (2.8)$$

$$Pz_{pd} = \begin{cases} \left|\frac{Py_1}{\sin\theta_{d1}}\right| & si\ \theta_{d1} \neq 0 \\ Pz & si\ \theta_{d1} = 0 \end{cases}, \text{ con } Py_1 = Py - (d_7 + d_8)\cos\theta_{d1} \quad (2.9)$$

$$\phi_{pd} = \phi + 90° \quad (2.10)$$

Por lo tanto, el manipulador a resolver es el mostrado en la Fig. 2.3, donde $d_{2p} = d_2 + d_3\cos(33.8°)$ y $\theta_{d2p} = \theta_{d2} + 90°$.

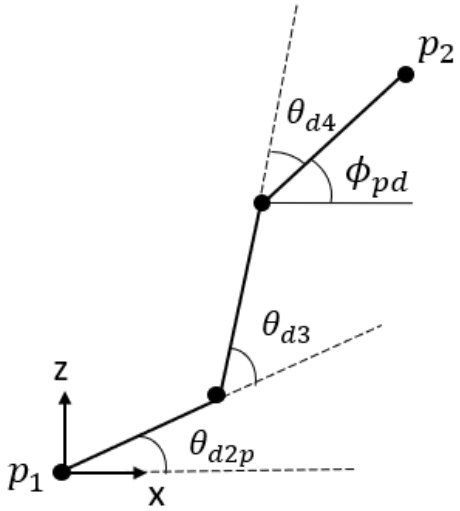


Figura 2.3 Manipulador plano de 3 GDL de los puntos p_1 a p_2.

Las ecuaciones que describen la posición y orientación del efector final son:

$$Px_{pd} = d_{2p}\cos(\theta_{d2p}) + d_5\cos(\theta_{d2p} + \theta_{d3}) + d_6\cos(\theta_{d2p} + \theta_{d3} + \theta_{d4}) \quad (2.11)$$

$$Pz_{pd} = d_{2p}\sin(\theta_{d2p}) + d_5\sin(\theta_{d2p} + \theta_{d3}) + d_6\sin(\theta_{d2p} + \theta_{d3} + \theta_{d4}) \quad (2.12)$$

$$\phi_{pd} = \theta_{d2p} + \theta_{d3} + \theta_{d4} \quad (2.13)$$

Se sustituye (2.13) en (2.11) y (2.12), y se reordenan:

$$Px_{pd} - d_6\cos\phi_{pd} = d_{2p}\cos(\theta_{d2p}) + d_5\cos(\theta_{d2p} + \theta_{d3}) \quad (2.14)$$

$$Pz_{pd} - d_6\sin\phi_{pd} = d_{2p}\sin(\theta_{d2p}) + d_5\sin(\theta_{d2p} + \theta_{d3}) \quad (2.15)$$

Se elevan al cuadrado y se suman ambas expresiones:

$$(Px_{pd} - d_6\cos\phi_{pd})^2 + (Pz_{pd} - d_6\sin\phi_{pd})^2 = {d_{2p}}^2 + {d_5}^2 + 2d_{2p}d_5(\cos(\theta_{d2p})\cos(\theta_{d2p} + \theta_{d3}) + \sin(\theta_{d2p})\sin(\theta_{d2p} + \theta_{d3}))$$

Por medio de la identidad trigonométrica de la diferencia de dos ángulos se obtiene:

$$(Px_{pd} - d_6 \cos\phi_{pd})^2 + (Pz_{pd} - d_6 \sin\phi_{pd})^2 = {d_{2p}}^2 + {d_5}^2 + 2d_{2p}d_5 \cos(\theta_{d3})$$

Despejando el coseno de θ_{d3}:

$$\cos(\theta_{d3}) = \frac{(Px_{pd} - d_6 \cos\phi_{pd})^2 + (Pz_{pd} - d_6 \sin\phi_{pd})^2 - {d_{2p}}^2 - {d_5}^2}{2d_{2p}d_5}$$

Entonces:

$$\theta_{d3} = atan2(\sqrt{1 - \cos^2(\theta_{d3})}, \cos(\theta_{d3})) \quad (2.16)$$

Nuevamente se toman las ecuaciones (2.14) y (2.15), y se expanden las funciones trigonométricas mediante identidades:

$$Px_{pd} - d_6 \cos\phi_{pd} = d_{2p}\cos(\theta_{d2p}) + d_5(\cos(\theta_{d2p})\cos(\theta_{d3}) - \sin(\theta_{d2p})\sin(\theta_{d3}))$$

$$Pz_{pd} - d_6 \sin\phi_{pd} = d_{2p}\sin(\theta_{d2p}) + d_5(\sin(\theta_{d2p})\cos(\theta_{d3}) + \cos(\theta_{d2p})\sin(\theta_{d3}))$$

Se reordenan las ecuaciones como sigue:

$$Px_{pd} - d_6 \cos\phi_{pd} = (-d_5 \sin(\theta_{d3}))\sin(\theta_{d2p}) + (d_{2p} + d_5 \cos(\theta_{d3}))\cos(\theta_{d2p}) \quad (2.17)$$

$$Pz_{pd} - d_6 \sin\phi_{pd} = (d_{2p} + d_5 \cos(\theta_{d3}))\sin(\theta_{d2p}) + (d_5 \sin(\theta_{d3}))\cos(\theta_{d2p}) \quad (2.18)$$

Al multiplicar (2.18) por $\frac{d_5 \sin(\theta_{d3})}{d_{2p}+d_5 \cos(\theta_{d3})}$ y sumar el resultado con (2.17) se obtiene:

$$Px_{pd} - d_6 \cos\phi_{pd} + \left(\frac{d_5 \sin(\theta_{d3})}{d_{2p} + d_5 \cos(\theta_{d3})}\right)(Pz_{pd} - d_6 \sin\phi_{pd}) = (d_{2p} + d_5 \cos(\theta_{d3}) + \frac{{d_5}^2 \sin^2(\theta_{d3})}{d_{2p} + d_5 \cos(\theta_{d3})})\cos(\theta_{d2p})$$

Se despeja el coseno de θ_{d2p}:

$$\cos(\theta_{d2p}) = \frac{Px_{pd} - d_6 \cos\phi_{pd} + \left(\frac{d_5 \sin(\theta_{d3})}{d_{2p} + d_5 \cos(\theta_{d3})}\right)(Pz_{pd} - d_6 \sin\phi_{pd})}{d_{2p} + d_5 \cos(\theta_{d3}) + \frac{{d_5}^2 \sin^2(\theta_{d3})}{d_{2p} + d_5 \cos(\theta_{d3})}}$$

Por lo tanto:

$$\theta_{d2p} = atan2\left(\sqrt{1 - \cos^2(\theta_{d2p})}, \cos(\theta_{d2p})\right)$$

$$\theta_{d2} = atan2\left(\sqrt{1 - \cos^2(\theta_{d2p})}, \cos(\theta_{d2p})\right) - 90° \quad (2.19)$$

El último ángulo está dado por la siguiente expresión:

$$\theta_{d4} = \phi_{pd} - \theta_{d2p} - \theta_{d3} \qquad (2.20)$$

Para la pierna izquierda se obtienen los ángulos de las articulaciones luego de seguir el mismo proceso.

$$\theta_{i1} = atan2(a_2, b_2) + atan2(\sqrt{1 - \frac{{c_2}^2}{{a_2}^2 + {b_2}^2}}, \frac{c_2}{\sqrt{{a_2}^2 + {b_2}^2}}) \qquad (2.21)$$

con $a_2 = i_1 - Pz$, $b_2 = pi - Py$ y $c_2 = i_7 + i_8$. Luego, el ángulo θ_{i3}:

$$\cos(\theta_{i3}) = \frac{(Px_{pi} - i_6 \cos\phi_{pi})^2 + (Pz_{pi} - i_6 \sin\phi_{pi})^2 - {i_{2p}}^2 - {i_5}^2}{2 i_{2p} i_5}$$

$$\theta_{i3} = atan2(\sqrt{1 - \cos^2(\theta_{i3})}, \cos(\theta_{i3})) \qquad (2.22)$$

con $i_{2p} = i_2 + i_3 \cos(33.8°)$ y:

$$Px_{pi} = Px \qquad (2.23)$$

$$Pz_{pi} = \begin{cases} \left| \frac{Py - (i_7 + i_8)\cos\theta_{i1}}{\sin\theta_{i1}} \right| & si\ \theta_{i1} \neq 0 \\ Pz & si\ \theta_{i1} = 0 \end{cases} \qquad (2.24)$$

$$\phi_{pi} = \phi + 90° \qquad (2.25)$$

El ángulo θ_{i2} recordando que $\theta_{i2p} = \theta_{i2} + 90°$:

$$\cos(\theta_{i2p}) = \frac{Px_{pi} - i_6 \cos\phi_{pi} + \left(\frac{i_5 \sin(\theta_{i3})}{i_{2p} + i_5 \cos(\theta_{i3})} \right) (Pz_{pi} - i_6 \sin\phi_{pi})}{i_{2p} + i_5 \cos(\theta_{i3}) + \frac{{i_5}^2 \sin^2(\theta_{i3})}{i_{2p} + i_5 \cos(\theta_{i3})}}$$

$$\theta_{i2} = atan2\left(\sqrt{1 - \cos^2(\theta_{i2p})}, \cos(\theta_{i2p})\right) - 90° \qquad (2.26)$$

Por último, el ángulo θ_{i4}:

$$\theta_{i4} = \phi_{pi} - \theta_{i2p} - \theta_{i3} \qquad (2.27)$$

2.3. Dinámica simplificada

Los trabajos relacionados con la robótica bípeda emplean dos metodologías para su estudio. La primera hace uso de los parámetros dinámicos de un robot, teniendo conocimiento de su masa, la localización de su centro de masa y la inercia de cada uno de sus eslabones. Por su parte, la segunda se sirve de un conocimiento limitado de la dinámica, utilizando información como el centro de masa total y el momento angular total [19]. Esta última

categoría ofrece estudios mediante modelos simplificados que reducen el costo computacional necesario para implementar sistemas de control en robots bípedos.

Para cumplir con el objetivo de este trabajo se empleó el modelo simplificado de un péndulo invertido tomado de [20]. Este modelo consiste en un péndulo invertido simple que conecta el suelo con el centro de masa de todo el robot por medio de un segmento virtual interpretado como una pierna telescópica de masa despreciable.

En la Fig. 2.4 se muestra dicho péndulo. La posición de la masa concentrada $p = (x, y, z)$ se especifica mediante las variables de estado $q = (\theta_r, \theta_p, r)$, donde θ_r corresponde al ángulo entre el péndulo y el plano XZ y tiene un valor negativo, θ_p se refiere al ángulo entre el péndulo y el plano YZ y tiene un ángulo positivo, y r indica la distancia que hay entre el origen y la masa concentrada.

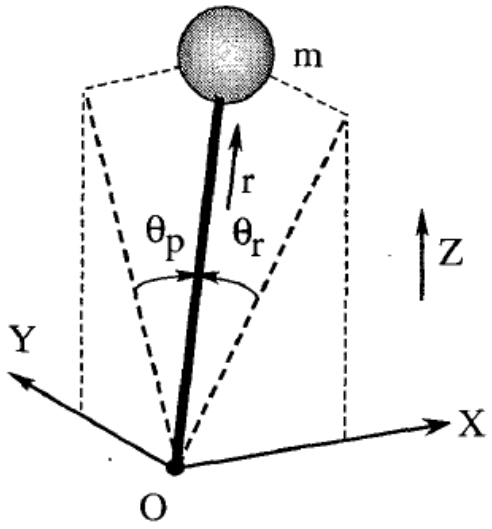


Figura 2.4 Péndulo invertido simple [20].

Así, las coordenadas cartesianas de la masa puntual están dadas por:

$$x = rS_p \qquad (2.28)$$

$$y = -rS_r \qquad (2.29)$$

$$z = rD \qquad (2.30)$$

donde $S_r = \sin\ \theta_r$, $S_p = \sin\ \theta_p$ y $D = \sqrt{1 - {S_r}^2 - {S_p}^2}$.

Considerando (τ_r, τ_p, f) como los pares y la fuerza asociadas a las variables de estado, se plantea la ecuación de movimiento del péndulo invertido:

$$m\begin{pmatrix}\ddot{x}\\ \ddot{y}\\ \ddot{z}\end{pmatrix} = (J^T)^{-1}\begin{pmatrix}\tau_r\\ \tau_p\\ f\end{pmatrix} + \begin{pmatrix}0\\ 0\\ -mg\end{pmatrix} \qquad (2.31)$$

donde m es la masa del péndulo y g la aceleración gravitatoria. El jacobiano J se construye como:

$$J = \frac{\partial p}{\partial q} = \begin{pmatrix} 0 & rC_p & S_p \\ -rC_r & 0 & -S_r \\ -rC_rS_r/D & -rC_pS_p/D & D \end{pmatrix} \qquad (2.32)$$

con $C_r = \cos\theta_r$ y $C_p = \cos\theta_p$.

Se premultiplica J^T para no tener la matriz inversa del jacobiano, obteniendo:

$$m\begin{pmatrix} 0 & -rC_r & -rC_rS_r/D \\ rC_p & 0 & -rC_pS_p/D \\ S_p & -S_r & D \end{pmatrix}\begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{pmatrix} = \begin{pmatrix} \tau_r \\ \tau_p \\ f \end{pmatrix} - mg\begin{pmatrix} -rC_rS_r/D \\ -rC_pS_p/D \\ D \end{pmatrix} \qquad (2.33)$$

Tomando el primer renglón y el segundo renglón de la expresión y sustituyendo las relaciones establecidas en (2.28), (2.29) y (2.30), se obtienen las ecuaciones que describen la dinámica a lo largo del eje y y del eje x, respectivamente:

$$m(-z\ddot{y} + y\ddot{z}) = \frac{D}{C_r}\tau_r - mgy \qquad (2.34)$$

$$m(z\ddot{x} - x\ddot{z}) = \frac{D}{C_p}\tau_p + mgx \qquad (2.35)$$

Se aplica una restricción al movimiento del péndulo, limitándolo a un plano cuyo vector normal es $(k_x, k_y, -1)$ y una intersección con el eje z en z_c:

$$z = k_x x + k_y y + z_c \qquad (2.36)$$

Si el plano de la restricción es horizontal ($k_x = k_y = 0$), la dinámica que describe el comportamiento del robot está dada por:

$$\ddot{y} = \frac{g}{z_c}y - \frac{1}{mz_c}u_r \qquad (2.37)$$

$$\ddot{x} = \frac{g}{z_c}x + \frac{1}{mz_c}u_p \qquad (2.38)$$

donde $u_r = \frac{D}{C_r}\tau_r$, $u_p = \frac{D}{C_p}\tau_p$.

Los valores de los parámetros del modelo son m =1.6 [kg], $g = 9.78\ [\frac{m}{s^2}]$ y $z_c = 0.16\ [m]$. Finalmente, se expresa el sistema en variables de estado, definiendo los vectores de estado y de entrada como sigue:

$$w = \begin{pmatrix} x \\ \dot{x} \\ y \\ \dot{y} \end{pmatrix} \qquad v = \begin{pmatrix} u_p \\ u_r \end{pmatrix}$$

$$\dot{w} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ \frac{g}{z_c} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{g}{z_c} & 0 \end{pmatrix} w + \begin{pmatrix} 0 & 0 \\ \frac{1}{mz_c} & 0 \\ 0 & 0 \\ 0 & -\frac{1}{mz_c} \end{pmatrix} v \qquad (2.39)$$

$$y_{sist} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} w + \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} v \qquad (2.40)$$

El sistema descrito por el modelo simplificado tiene como estados las posiciones y las velocidades del centro de masa del robot bípedo. Asumiendo que el centro de masa se encuentra en la cadera del Bioloid GP, la cinemática inversa permite conocer los valores articulares para que el efector final, en este caso la cadera, alcance una posición deseada.

3. Diseño del controlador

La ingeniería de control está cada vez más inmersa en la sociedad dadas las numerosas aplicaciones propuestas e implementadas como los lanzamientos de cohetes espaciales, los vehículos autónomos y los robots industriales. Un sistema de control se conforma por un conjunto de subsistemas unidos de tal manera que produzcan una salida deseada con un desempeño deseado dada una entrada específica [21]. En otras palabras, controlar un sistema implica obtener un resultado deseado como salida aplicando un determinado estímulo como entrada.

Para este trabajo se define al robot bípedo Bioloid GP descrito anteriormente como el sistema a controlar. En el Capítulo 2 del presente trabajo, se obtuvo el modelo matemático del robot, definiendo restricciones para simplificar el modelo. Una vez obtenido el modelo del sistema, se procede a establecer los objetivos de control.

Se requiere que el robot bípedo se mantenga de pie sobre una plataforma móvil a pesar de las perturbaciones y los cambios de inclinación que ocurran. Para lograr la estabilidad de postura del robot bípedo se retoman los conceptos del Capítulo 1, donde se indica que la proyección sobre el suelo del centro de masa del robot debe estar dentro del polígono de soporte. Dado que ambas plantas del pie se mantienen fijas al suelo, el polígono de soporte se describe como la superficie de las plantas del pie y la que existe entre éstas. El origen del sistema de referencia se posiciona en el punto medio entre los pies, por lo tanto, el objetivo de control a cumplir es el de estabilización, es decir, llevar la posición del centro de masa en los ejes *x* y *y* al valor constante cero. Como desempeño, se plantea que el modelo simplificado se estabilice en un segundo.

3.1. xAnálisis del modelo del sistema

La estabilidad, la controlabilidad y la observabilidad son tres conceptos muy importantes en el análisis de sistemas de control. Para realizar este análisis, se emplea el modelo simplificado del sistema. Éste está definido por las siguientes matrices:

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ \frac{g}{z_c} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{g}{z_c} & 0 \end{pmatrix} \qquad B = \begin{pmatrix} 0 & 0 \\ \frac{1}{mz_c} & 0 \\ 0 & 0 \\ 0 & -\frac{1}{mz_c} \end{pmatrix} \qquad C = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{3.1}$$

Para los sistemas lineales como el de la Ec. (3.1), se logra una estabilidad del tipo BIBO (bounded-input, bounded-output), en la que un sistema es estable si toda entrada acotada produce una salida acotada, o inestable si cualquier entrada acotada produce una salida no acotada [21].

Para analizar la estabilidad de un sistema su utiliza su ecuación característica ($|\lambda I - A| = 0$) y la ubicación de sus polos. La ecuación característica del modelo simplificado del Bioloid GP está dada por:

$$\left|\begin{pmatrix}\lambda & 0 & 0 & 0\\ 0 & \lambda & 0 & 0\\ 0 & 0 & \lambda & 0\\ 0 & 0 & 0 & \lambda\end{pmatrix}-\begin{pmatrix}0 & 1 & 0 & 0\\ \frac{g}{z_c} & 0 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & \frac{g}{z_c} & 0\end{pmatrix}\right|=0$$

$$\lambda\left(\lambda\left(\lambda^2-\frac{g}{z_c}\right)\right)+\frac{g}{z_c}(-1)\left(\lambda^2-\frac{g}{z_c}\right)=0$$

$$\lambda^4-2\frac{g}{z_c}\lambda^2+\left(\frac{g}{z_c}\right)^2=0$$

$$(\lambda^2-\frac{g}{z_c})^2=0$$

$$\lambda_{1,2,3,4}=\pm\sqrt{\frac{g}{z_c}} \qquad (3.2)$$

Los polos del sistema muestran la repetición de polos, los cuatro son reales, dos negativos y dos positivos. Dado que existen polos en el semiplano complejo derecho, el sistema descrito por el modelo simplificado es inestable.

Un sistema es controlable si existe al menos una entrada para el sistema que lleve a cada variable de estado de cualquier condición inicial a un estado final deseado en un tiempo finito [21]. Para analizar la controlabilidad de un sistema, se emplea la matriz de controlabilidad de Kalman definida como [21]:

$$MC=(B \quad AB \quad A^2B \quad \ldots \quad A^{n-1}B) \qquad (3.3)$$

Así, para el robot, la matriz de controlabilidad (MC) es:

$$MC=\begin{pmatrix}0 & 0 & \frac{1}{mz_c} & 0 & 0 & 0 & \frac{g}{mz_c{}^2} & 0\\ \frac{1}{mz_c} & 0 & 0 & 0 & \frac{g}{mz_c{}^2} & 0 & 0 & 0\\ 0 & 0 & 0 & -\frac{1}{mz_c} & 0 & 0 & 0 & -\frac{g}{mz_c{}^2}\\ 0 & -\frac{1}{mz_c} & 0 & 0 & 0 & -\frac{g}{mz_c{}^2} & 0 & 0\end{pmatrix} \qquad (3.4)$$

Por inspección, se sabe que el rango de la matriz es 4 y que es de rango completo, por lo tanto, el sistema es controlable.

Un observador es un sistema dinámico que reconstruye las señales de las variables de estado que no se miden por medio de las señales que sí se conocen. En los sistemas de control, un observador es usado para calcular las variables de estado que no se pueden conocer a través de un instrumento de medición ya sea por el costo, la disponibilidad o la exactitud. Los observadores también son llamados estimadores por la función que cumplen

[21]. Para conocer si un sistema es observable se utiliza la matriz de observabilidad con la forma:

$$MO = \begin{pmatrix} C \\ CA \\ \dots \\ CA^{n-1} \end{pmatrix} \qquad (3.5)$$

Entonces, la matriz de observabilidad (MO) para el sistema abordado en este trabajo está dado por:

$$MO = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{g}{z_c} & 0 & 0 & 0 \\ 0 & 0 & \frac{g}{z_c} & 0 \\ 0 & \frac{g}{z_c} & 0 & 0 \\ 0 & 0 & 0 & \frac{g}{z_c} \end{pmatrix} \qquad (3.6)$$

Tras inspeccionar la matriz MO, se sabe que su rango es 4 y que es de rango completo, esto indica que el sistema es observable.

3.2. Control LQR

Posterior al análisis del modelo del sistema, se procede a la selección del controlador a diseñar. Se sabe que la planta es inestable, por lo que se requiere que el controlador vuelva al sistema estable; además, las matrices de controlabilidad y de observabilidad indican que el sistema es controlable y observable, es decir, todas las variables de estado pueden llevarse a un estado deseado y pueden ser estimadas mediante un observador. En esta sección se describe el controlador seleccionado y se muestra la obtención de sus parámetros.

Dadas las características del sistema y que la instrumentación permitirá conocer todos los estados, se selecciona un controlador LQR (*linear quadratic regulator*) para cumplir con el objetivo de control de estabilización y no hay necesidad de utilizar un observador para reconstruir los estados, ya que todos se miden. El control óptimo permite minimizar o maximizar un índice de comportamiento de tal manera que se cumpla con los objetivos de diseño.

En [22] se muestra el desarrollo para problemas de control óptimo cuadrático. Se considera el sistema descrito por la ecuación:

$$\dot{w} = Aw + Bv \qquad (3.7)$$

entonces el problema de regulador cuadrático lineal consiste en determinar la matriz de ganancia K_{lqr}:

$$v(t) = -K_{lqr}w(t) \qquad (3.8)$$

de manera tal que se minimice la función de costo *J* definida como:

$$J = \int_0^\infty (w^T Q w + v^T R v)dt \qquad (3.9)$$

donde *Q* y *R* son las matrices de pesos para los estados y las entradas, respectivamente, ambas son simétricas y positivas definidas. Las matrices *Q* y *R* establecen la importancia relativa del error y del coste de esta energía.

Sustituyendo (3.8) en (3.7), se obtiene:

$$\dot{w} = Aw - BK_{lqr}w$$

$$\dot{w} = (A - BK_{lqr})w \qquad (3.10)$$

donde la matriz $A - BK_{lqr}$ es estable.

Luego de sustituir (3.10) en (3.9), se obtiene:

$$J = \int_0^\infty (w^T Q w + w^T {K_{lqr}}^T R K_{lqr} w)dt$$

$$J = \int_0^\infty w^T \left(Q + {K_{lqr}}^T R K_{lqr}\right) w \, dt \quad (3.11)$$

La expresión en la integral se iguala como sigue:

$$w^T \left(Q + {K_{lqr}}^T R K_{lqr}\right) w = -\frac{d}{dt}(w^T P w) \qquad (3.12)$$

siendo *P* una matriz simétrica y positiva definida. Así, la ecuación se desarrolla:

$$w^T \left(Q + {K_{lqr}}^T R K_{lqr}\right) w = -w^T P \dot{w} - \dot{w}^T P w$$

$$w^T \left(Q + {K_{lqr}}^T R K_{lqr}\right) w = -w^T P (A - BK_{lqr})w - w^T (A - BK_{lqr})^T P w$$

$$w^T \left(Q + {K_{lqr}}^T R K_{lqr}\right) w = -w^T (P\left(A - BK_{lqr}\right) + \left(A - BK_{lqr}\right)^T P) w \qquad (3.13)$$

Para cualquier *w*, se debe cumplir que:

$$P\left(A - BK_{lqr}\right) + \left(A - BK_{lqr}\right)^T P = -(Q + {K_{lqr}}^T R K_{lqr}) \qquad (3.14)$$

Retomando la expresión de (3.12) y sustituyendo en *J*:

$$J = -\int_0^\infty \frac{d}{dt}(w^T P w)dt$$

$$J = -w^T P w \Big|_0^\infty$$

$$J = -w^T(\infty)Pw(\infty) + w^T(0)Pw(0) \qquad (3.15)$$

Como el sistema descrito por (3.7) es estable, entonces $w(\infty) \to 0$ y la función de costo sólo depende de *P* y de las condiciones iniciales del vector de estados:

$$J = w^T(0)Pw(0) \qquad (3.16)$$

Como *R* es una matriz simétrica positiva definida, se puede expresar como:

$$R = T^T T \qquad (3.17)$$

entonces:

$$PA - PBK_{lqr} + A^T P - {K_{lqr}}^T B^T P = -Q - {K_{lqr}}^T T^T T K_{lqr}$$

$$PA + A^T P - PBK_{lqr} - {K_{lqr}}^T B^T P + {K_{lqr}}^T T^T T K_{lqr} = -Q \qquad (3.18)$$

Reescribiendo:

$$PA + A^T P + [TK_{lqr} - T^T B^T P]^T \left[TK_{lqr} - T^T B^T P\right] - PB(T^T T)^{-1} B^T P = -Q$$

$$PA + A^T P + [TK_{lqr} - (T^T)^{-1} B^T P]^T \left[TK_{lqr} - (T^T)^{-1} B^T P\right] - PBR^{-1} B^T P = -Q \qquad (3.19)$$

Para la minimización de *J* se requiere que la expresión entre corchetes sea igual a cero:

$$TK_{lqr} - (T^T)^{-1} B^T P = 0$$

$$K_{lqr} = T^{-1}(T^T)^{-1} B^T P \qquad (3.20)$$

Por lo tanto:

$$K_{lqr} = R^{-1} B^T P \qquad (3.21)$$

De esta forma, el procedimiento general para el diseño de un controlador LQR está dado por tres pasos:

1. Proponer las matrices de pesos *Q* y *R*, de tal manera que $Q = Q^T > 0$ y $R = R^T > 0$.
2. Obtener la solución de la matriz *P* a partir de la ecuación algebraica de Riccati:

$$PA + A^T P - PBR^{-1} B^T P = -Q \qquad (3.22)$$

3. Calcular la matriz de ganancias K_{lqr} mediante la expresión:

$$K_{lqr} = R^{-1} B^T P \qquad (3.23)$$

El software de Matlab ofrece una función que calcula la matriz de ganancias para el control óptimo cuyas entradas son las matrices *A* y *B* de la representación de estados del sistema, las matrices *Q* y *R*:

$$K_{lqr} = lqr(A, B, Q, R)$$

Esta herramienta permitió simular el modelo simplificado con diferentes valores de pesos para los estados y las entradas. Las iteraciones se detuvieron una vez alcanzado un comportamiento del modelo simplificado que se estabilizara en 1 segundo, tomando este valor como referencia ya que un tiempo mayor podría ocasionar la volcadura del robot y un tiempo mucho menor sería más complicado de lograr para los actuadores. Se da mayor importancia relativa a los estados correspondientes a la posición para dar prioridad a su regulación. Así, la propuesta de las matrices *Q* y *R* queda como sigue:

$$Q = \begin{pmatrix} 10 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0.1 \end{pmatrix} \qquad R = \begin{pmatrix} 0.5 & 0 \\ 0 & 0.5 \end{pmatrix}$$

Definiendo los parámetros del sistema ($g = 9.78\ \left[\frac{m}{s^2}\right], m = 1.6[kg], z_c = 0.16\ [m]$), se obtiene:

$$K_{lqr} = \begin{pmatrix} 31.9225 & 4.0675 & 0 & 0 \\ 0 & 0 & -31.9225 & -4.0675 \end{pmatrix}$$

Por último, la Fig. 3.1 muestra el diagrama de bloque del sistema de control para el robot bípedo:

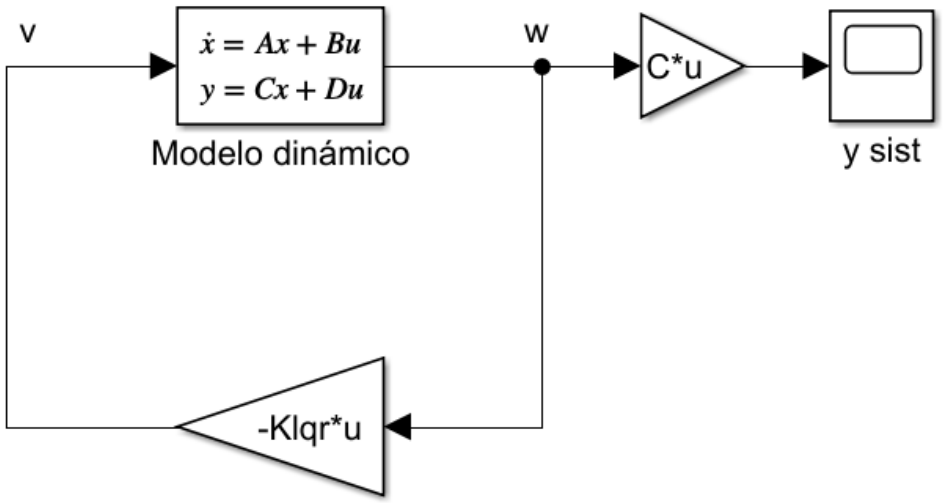


Figura 3.1 Control LQR para el modelo dinámico simplificado del robot bípedo Bioloid GP.

La matriz de ganancias muestra el mismo valor absoluto para las constantes correspondientes a los ejes X y Y debido a que los pesos para cada eje fueron iguales tanto para los estados como para las salidas. En las matrices de pesos, dar mayor importancia a la regulación de los estados que representan la posición se debe a que el objetivo de control busca mantener el centro de masa en el origen del plano XY.

3.3. Simulación del sistema de control

Primero, para validar que el modelo dinámico simplificado cumple con el comportamiento esperado, se simuló el sistema en lazo abierto, aplicando como entradas un par de torsión de $0.01\ [Nm]$ tanto en el eje x como en el eje y. En las figuras, las variables x y y representan las posiciones y las variables xp y yp, las velocidades del centro de masa. La Fig. 3.2 muestra el comportamiento de los estados en los cinco segundos de simulación, se observa que para un par positivo los valores de posición y velocidad en x crecen indefinidamente, mientras que en y decrecen indefinidamente. Esta simulación es evidencia de que el modelo simplificado representa correctamente el comportamiento de la posición del centro de masa del robot bípedo al ser sometido a pares de torsión en diferentes direcciones.

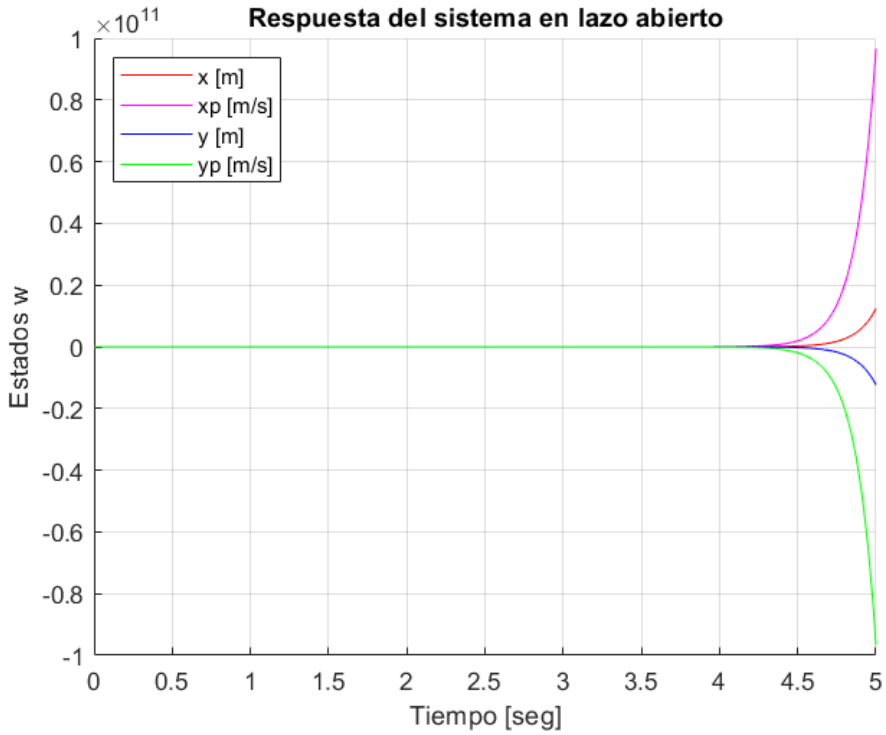


Figura 3.2 Respuesta del sistema descrito por el modelo dinámico simplificado en lazo abierto.

Retomando el sistema de lazo cerrado representado en la Fig. 3.1, se espera que, al ser un controlador diseñado para lograr la estabilización del sistema, todos los estados tiendan a cero después de cierto tiempo. Para la simulación se consideró que el centro de masa se encontraba en una posición inicial $x = 0.1\ [m], y = 0.05\ [m]$. La Fig. 3.3 contiene el gráfico con la respuesta de los estados con el sistema realimentado y la Fig. 3.4 muestra la entrada exigida para lograr esta respuesta.

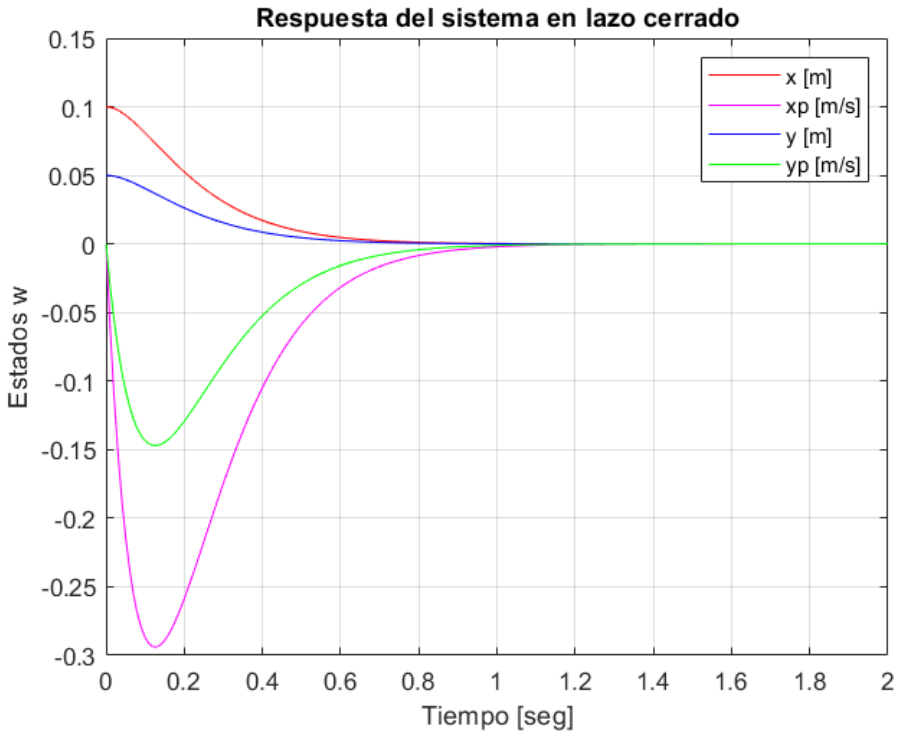


Figura 3.3 Respuesta del sistema en lazo cerrado mediante un control LQR.

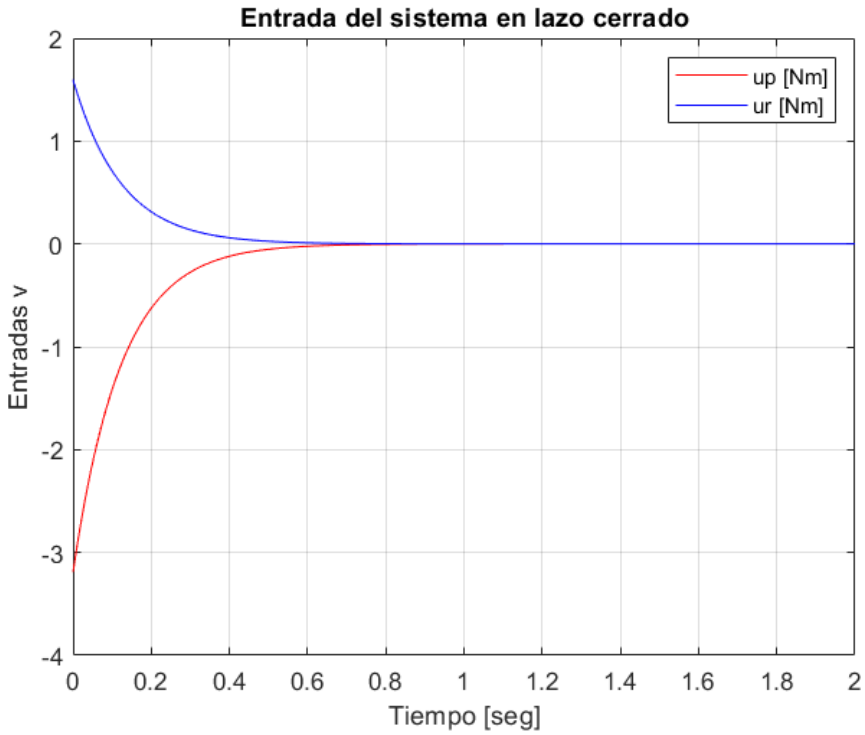


Figura 3.4 Entradas del sistema en lazo cerrado.

Los estados del sistema alcanzan el valor de cero en aproximadamente 1 segundo, es decir, se cumple la estabilización y la proyección del centro de masa se encuentra dentro del polígono de soporte. La respuesta esperada es suave, por lo que no se esperan cambios bruscos en la posición. La velocidad en ambas coordenadas crece en poco tiempo y después decrece gradualmente; esto indica que el crecimiento de velocidad ocurre cuando comienza a moverse y que se reduce cuando está cerca de alcanzar el valor deseado. En cuanto a las entradas, puede observarse que la entrada que tiene influencia en la posición en x es la que exige un par mayor, superando los 3 $[Nm]$ en sentido negativo. Por la dificultad para tener una medición exacta del valor de torsión a la entrada, la consecuencia esperada en caso de no alcanzar este valor de entrada es que el sistema alcance el valor deseado en un tiempo mayor. Además, se observa que las entradas comienzan con una magnitud muy grande que se reduce hasta alcanzar un valor nulo cuando el sistema ha alcanzado el valor deseado.

En la Fig. 3.5 se muestran los estados correspondientes al eje x cuando se aplica una perturbación en dicho eje. La perturbación corresponde a un escalón de magnitud 0.5 $[Nm]$ que inicia tras transcurrir un segundo y mantiene su magnitud durante medio segundo. En la gráfica de la simulación puede observarse la perturbación reflejada como un cambio tanto en la posición como en la velocidad. La posición x comienza a crecer hasta un valor cercano a 0.03 $[m]$, en cuanto deja de aplicarse la perturbación, la posición se acerca al valor de cero y se estabiliza en alrededor de un segundo. En el caso de la velocidad, ésta crece en el sentido positivo hasta que se detiene y decrece. El comportamiento hace que vuelva a crecer hasta estabilizarse a los 2.5 segundos de iniciada la simulación, mismo momento en que la velocidad alcanza el valor de cero.

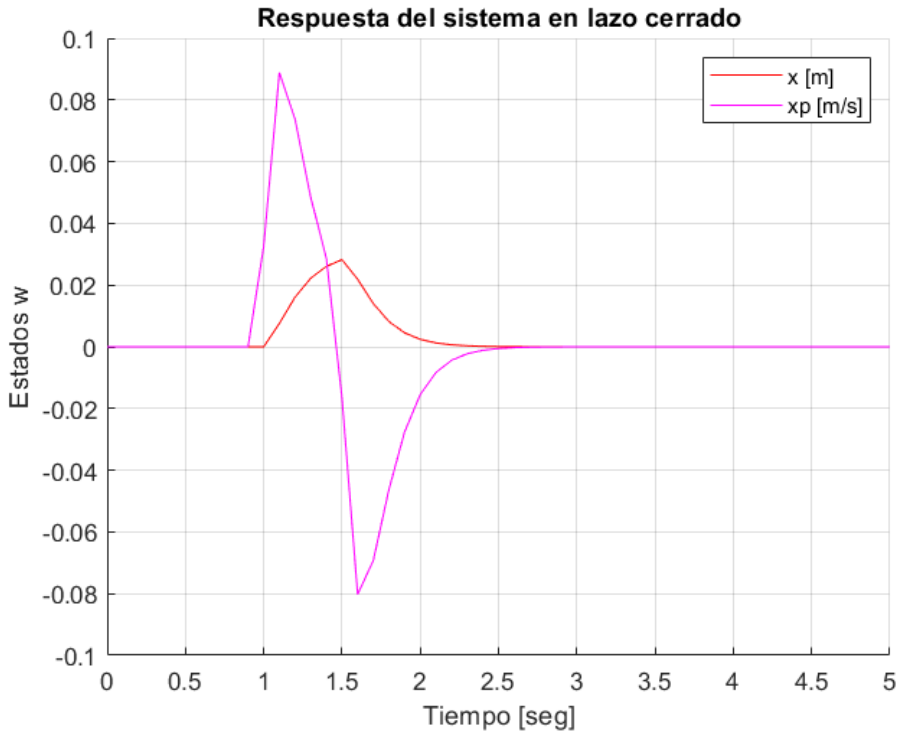


Figura 3.5 Respuesta del sistema en lazo cerrado ante una perturbación en el eje x.

La Fig. 3.6 muestra la entrada del sistema incluyendo la perturbación en el eje x. Debido a la respuesta del sistema en lazo cerrado, la entrada escalón aplicada como perturbación no puede apreciarse, sin embargo, alrededor del primer segundo se pone en evidencia la perturbación en forma de una rampa, primero positiva y luego con inclinación negativa. Luego, la entrada tiene un valor negativo hasta que cerca del segundo 1.5 comienza a regresar al valor de cero. Al inicio de la simulación, la perturbación es la que predomina como entrada y la compensación del sistema en lazo cerrado únicamente evita que los estados crezcan indefinidamente. En cuanto la perturbación deja de ser aplicada, el sistema aplica la entrada necesaria para estabilizar el sistema.

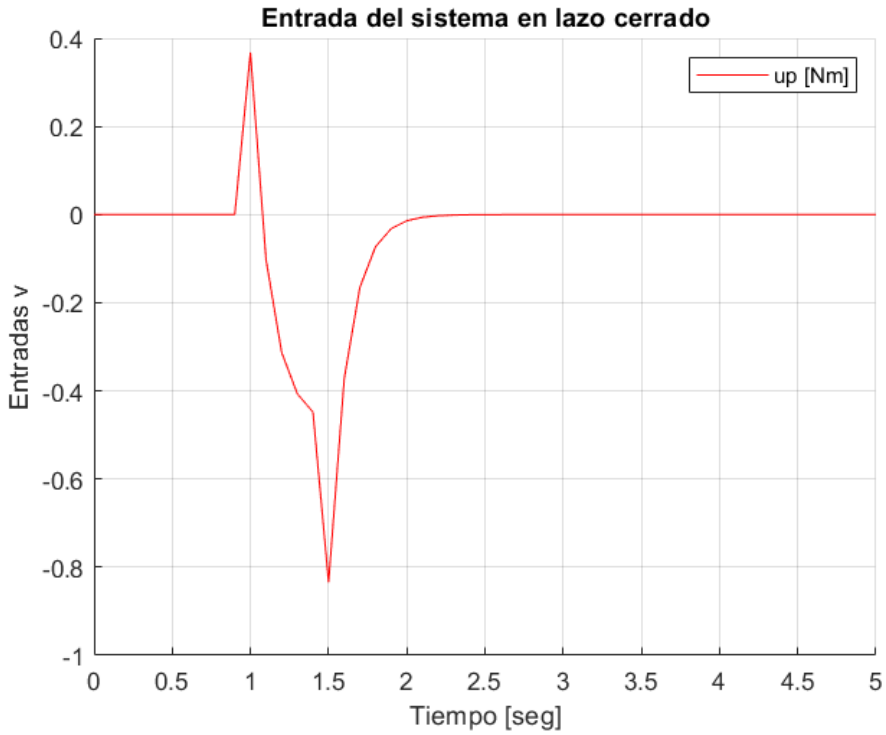


Figura 3.6 Entrada del sistema en lazo cerrado con una perturbación en el eje x.

4. Implementación

4.1. Instrumentación

En el capítulo 2 se mostraron las características técnicas del Bioloid GP proporcionadas por el fabricante del producto. Para implementar el controlador diseñado en el presente trabajo se requiere de una instrumentación específica. El sistema modificado del robot bípedo consta de un microcontrolador ESP32, servomotores DYNAMIXEL AX-12A y AX-18A y otros componentes auxiliares para el funcionamiento del sistema. En el Anexo 1 se muestra el diagrama de conexiones del circuito empleado para el funcionamiento del sistema Bioloid GP.

4.1.1 Microcontrolador

El microcontrolador es el elemento que controla las funciones del sistema. En él se cargan las instrucciones del programa principal y se realizan los cálculos tanto de la cinemática inversa como del control LQR. Se requiere un microcontrolador que ofrezca buena velocidad de cálculo y que permita la comunicación con todos los elementos del sistema.

El microcontrolador seleccionado fue el ESP32-WROOM-32D contenido en la tarjeta de desarrollo ESP32-DevKitC. Este dispositivo fabricado por Espressif cuenta con una unidad Tensilica Xtensa 32-bit LX6 de dos núcleos, memoria interna con 448 KB de ROM y 520 KB de SRAM y reloj con frecuencia ajustable de 80 a 240 MHz. El ESP32 ofrece diferentes interfaces como I2C, PWM, ADC, DAC y GPIO. Además, los microcontroladores de esta serie integran una conectividad Wifi y Bluetooth. Hay tres opciones para la alimentación del dispositivo: conector USB, pines 5V/GND y pines 3V3/GND [23] [24].

Figura 4.1 Tarjeta de desarrollo ESP32-DevKitC [24].

4.1.2 Medición de posición y velocidad

El controlador diseñado tiene como estados la posición y la velocidad del centro de masa del robot, por lo tanto, estos estados deben ser medibles para retroalimentar el sistema. Para lograr esto se seleccionó una cámara web, la cámara empleada fue la HP Webcam HD-2200. La cámara se utiliza con un programa desarrollado en Matlab que permite la identificación de objetos de un color, en este caso rojo, para detectar su ubicación en el plano mediante procesamiento de imagen [25]. La captura se toma desde la parte superior, el objeto detectado corresponde a una marca roja colocada en el robot y con esta

información se obtiene tanto la posición como la velocidad del centro de masa del robot en las coordenadas (x,y).

4.1.3 Actuadores

Las articulaciones del robot cuentan con servomotores DYNAMIXEL como actuadores. La parte inferior del cuerpo, correspondiente a los grados de libertad analizados en este trabajo, tiene motores AX-18A y la parte superior, que comprende los brazos del robot, tiene motores AX-12A.

Los DYNAMIXEL son actuadores inteligentes desarrollados para conectar juntas en robots o estructuras mecánicas. Estos servomotores integran un motor de corriente directa, un tren de engranes para la reducción, un controlador y una red para la comunicación [26]. El modelo AX-18A tiene una resolución de 0.29°, un par de 1.8 [Nm] y una velocidad sin carga de 97 [rpm]. El voltaje de entrada puede ir de los 9 [V] a los 12 [V], aunque se recomienda utilizar 11.1 [V]. El protocolo de comunicación es serial asíncrono *half duplex* y la conexión física consta de un bus multipunto con tecnología TTL [27].

Figura 4.2 Servomotor DYNAMIXEL AX-18A de Robotis [27].

4.1.4 Componentes auxiliares

El sistema cumple su función gracias a la acción de los componentes principales, sin embargo, es necesario utilizar componentes auxiliares para que exista la correcta alimentación de cada uno de los elementos, la correcta comunicación entre todos ellos y un mejor funcionamiento del sistema. A continuación, se describe cada uno estos:

- Regulador: El sistema en su conjunto requiere que se le suministre energía eléctrica, pero cada componente tiene sus propias características técnicas y el voltaje de entrada es diferente para cada uno. En este sentido, el regulador permite utilizar una fuente de alimentación con un voltaje de 11.1 [V] para los servomotores mientras se alimenta al sensor y al microcontrolador con 5 [V].

  El regulador de voltaje de tres terminales LM7805 admite hasta 35 [V] a la entrada y tiene una salida de 5 [V] [28]. El circuito propuesto en la hoja de datos de este componente se muestra en la Fig. 4.1.

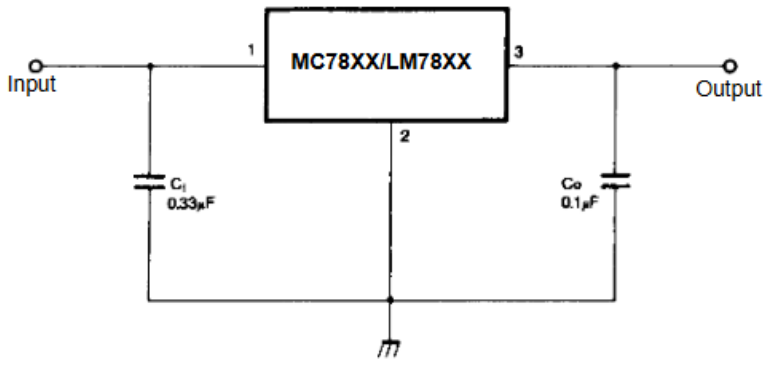


Figura 4.3 Circuito propuesto para regulación de voltaje con LM7805 [28].

- Búfer de 3 estados: Para controlar los motores DYNAMIXEL, el microcontrolador debe convertir sus señales de tipo UART al tipo *half duplex* [27]. Para lograr la comunicación con los servomotores, se utiliza un búfer de 3 estados. El componente seleccionado fue el circuito integrado 74LS241, descrito como un búfer octal en línea con salidas con 3 estados diseñado para mejorar el desempeño de controladores de dirección de memoria de tres estados, controladores de reloj y receptores y transmisores basado en comunicación por bus. Se alimenta con 5 [V] [29].

4.2. Algoritmo

Para implementar el control LQR que permita la estabilidad estática del robot bípedo es necesario realizar varias operaciones. Luego de cada operación se obtiene cierta información como salida y, como el sistema es retroalimentado, el algoritmo es un ciclo que se repite indefinidamente. Los pasos que conforman el algoritmo se muestran de forma secuencial en la Fig. 4.4, los bloques son las operaciones y las flechas indican el sentido del flujo del proceso. Adicionalmente, en el Anexo 2 se encuentra el programa del microcontrolador.

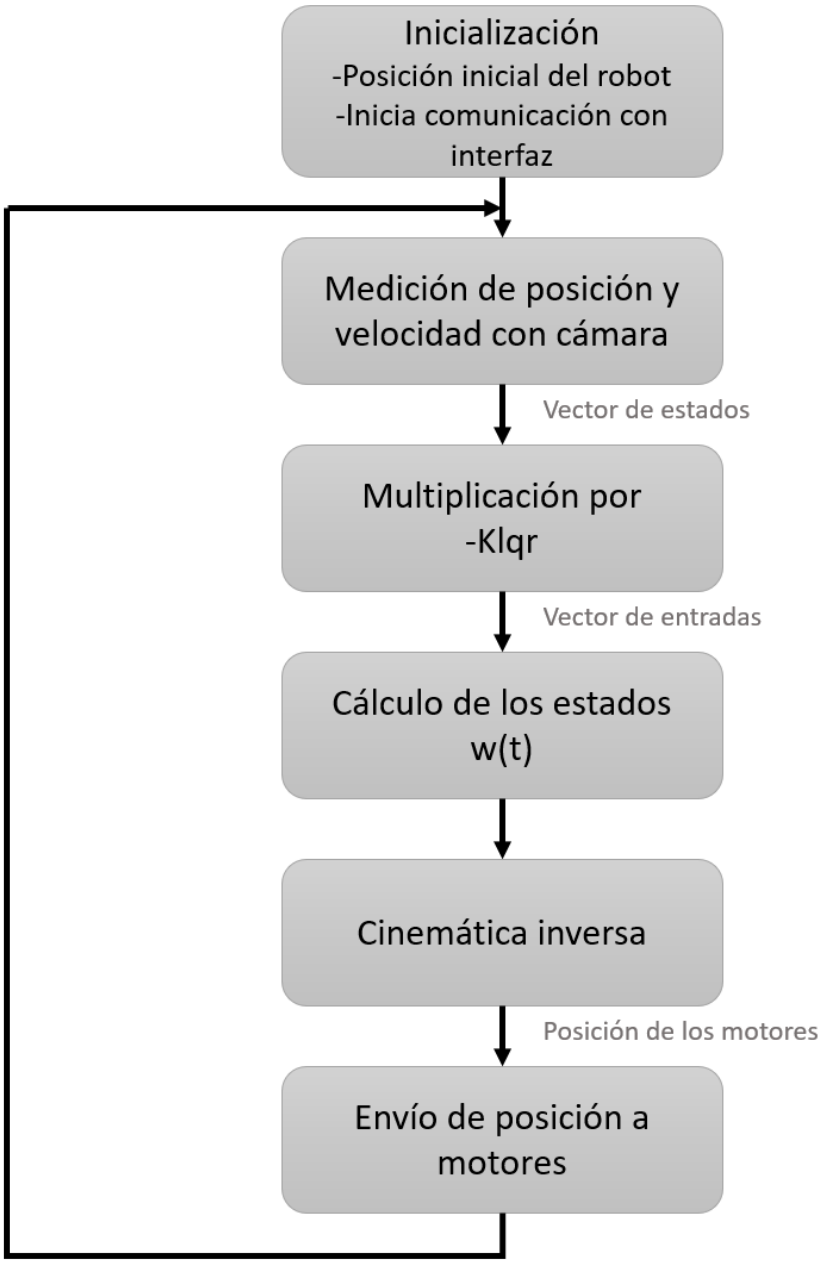


Figura 4.4 Algoritmo para la implementación del control LQR en el Bioloid GP.

A continuación, se describe cada una de las tareas y se incluyen detalles sobre la codificación realizada en Arduino:

i. Inicialización

En este paso se inicia la comunicación bluetooth de la computadora con el microcontrolador y la comunicación con los motores. También, se envía la información a los motores para que el robot se coloque en la posición inicial. Para la comunicación bluetooth se utiliza la librería "BluetoothSerial.h" y para la comunicación con los motores la librería "DynamixelSerial.h".

ii. Medición de posición y velocidad con cámara

El programa desarrollado en Matlab captura una imagen con la cámara web, sustrae la componente correspondiente al color rojo de la imagen en escala de grises para quitar los componentes rojos en la imagen, usa un filtro para quitar el ruido y convierte la imagen en escala de grises a una imagen binaria. Luego, identifica los componentes que están conectados y mediante un análisis de Blob obtiene el centroide de la región en rojo [25] que en este caso representa el centro de masa del robot. Una vez obtenida la posición (x,y) del centro de masa, se calcula la velocidad en ambos ejes como la diferencia de la posición actual y la anterior, dividida entre el tiempo transcurrido entre mediciones.

Los datos de posición y velocidad se envían como una cadena por bluetooth al microcontrolador. La cadena separa cada dato con una coma "," y utiliza un punto y coma ";" para indicar el final del mensaje. Así, el microcontrolador obtiene la información del vector de estados. El programa y la interfaz empleadas se muestran en el Anexo 3.

iii. Multiplicación por -Klqr

El vector obtenido en el paso anterior se multiplica por el negativo de la matriz Klqr como se muestra en la Fig. 3.1. Con esta operación matricial se obtiene el vector de entradas al sistema. En Arduino, se emplea la función *Multiply()* de la librería "BasicLinearAlgebra.h" para realizar este paso.

iv. Cálculo de los estados w(t)

Las entradas obtenidas mediante el modelo simplificado corresponden a pares de torsión. Dado que el Bioloid recibe los datos de la posición de cada articulación, es necesario obtener la posición que debe tener el centro de masa del robot luego de aplicar las entradas calculadas. La matriz exponencial permite resolver la ecuación diferencial del modelo simplificado a partir del vector de entradas con la finalidad de conocer los nuevos estados para un tiempo futuro. Para estos cálculos en el microcontrolador, también se utilizan diversas funciones de la librería "BasicLinearAlgebra.h".

v. Cinemática inversa

En esta operación, se realiza el cálculo de la cinemática inversa con las ecuaciones desarrolladas en el capítulo 2, teniendo como entrada la posición indicada por los estados calculados en el paso anterior. De esta manera, el microcontrolador obtiene los valores correspondientes a los ángulos de cada articulación para que el centro de masa se ubique en la posición deseada.

vi. Envío de posición a motores

Como último paso, los ángulos de cada articulación se envían a los motores DYNAMIXEL a través del búfer de tres estados. Para esto, se utiliza la función

move() de la librería “DynamixelSerial.h”. Tras este último paso, el robot realiza un movimiento que modifica la posición de su centro de masa y es necesario repetir el proceso desde el paso (ii) para que la estabilidad estática del robot se cumpla.

Con base en el algoritmo, los componentes seleccionados para implementar el sistema de control se relacionan como se muestra en la Fig. 4.5. En la figura se aprecia que la cámara web se conecta mediante un puerto USB a una computadora, ésta se comunica con el microcontrolador mediante bluetooth. El microcontrolador utiliza comunicación serial para trasmitir los ángulos al búfer, donde se convierten al tipo *half duplex* para hacerlos llegar a los motores Dynamixel.

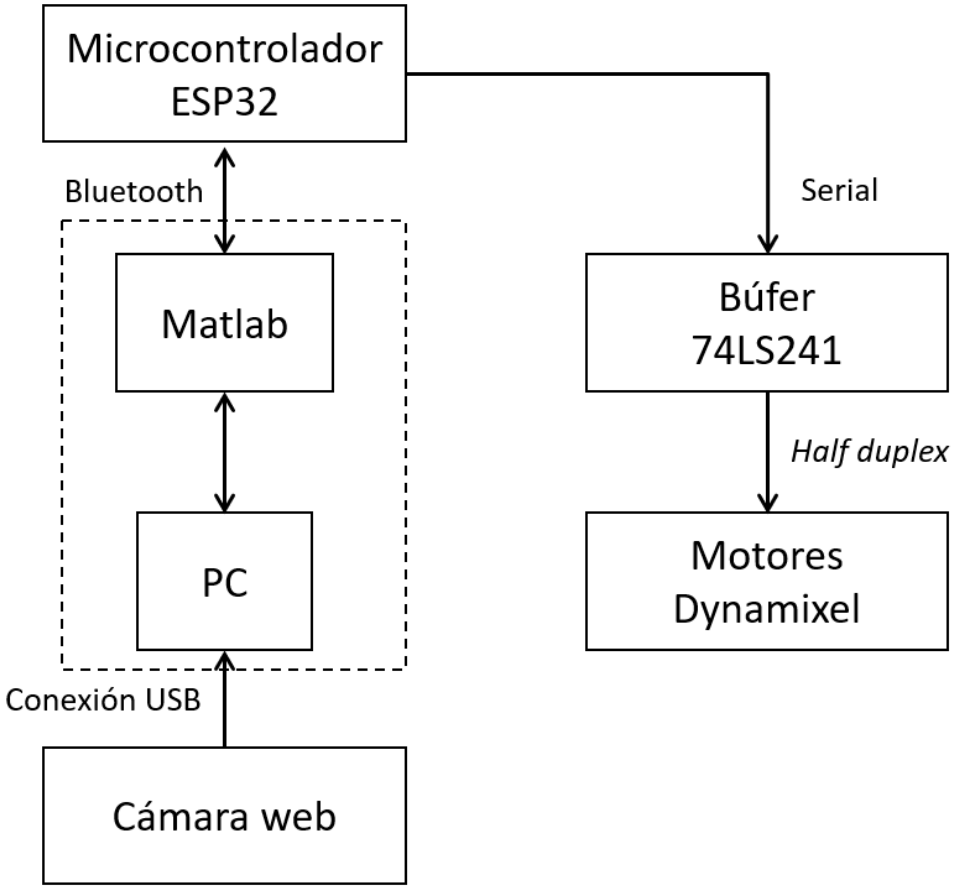


Figura 4.5 Esquema relacional de los componentes utilizados.

5. Pruebas y resultados

Con la finalidad de verificar el funcionamiento del controlador diseñado, se realizaron varias pruebas con el robot bípedo Bioloid GP haciendo uso de la instrumentación y del algoritmo descritos en el capítulo anterior. En este capítulo se describen las pruebas realizadas y se muestran los resultados obtenidos en cada una de ellas. Se probó el funcionamiento del controlador sin perturbaciones y ante perturbaciones en diferentes sentidos. Las primeras pruebas se realizaron sobre una superficie plana y las últimas inclinando la superficie sobre la que se soportaba el robot; en ambos casos, los brazos se posicionaron extendidos a los lados. El tiempo de muestreo fue próximo a los 140 [ms], con pequeñas variaciones entre cada iteración debido al tiempo de procesamiento de imagen.

5.1. Superficie plana

5.1.1 Sin movimiento

En esta prueba el robot se colocó de pie sobre una superficie plana y no se aplicó ninguna perturbación sobre él. La Fig. 5.1 muestra la posición del CoM del robot durante la prueba. La posición del centro de masa se mantuvo en el orden de las centésimas de milímetro, es decir, su valor en ambos ejes era prácticamente cero. No alcanzó el valor exacto de cero debido al ruido presente en las mediciones. Como el objetivo de control es la estabilización, al no existir ningún desplazamiento y estar el centro de masa muy cerca del origen, la respuesta esperada del controlador es mantener la ubicación del centro de masa en el mismo lugar; entonces, ninguna entrada debería ser aplicada. La prueba tuvo una duración de 16 segundos, tiempo en el cual la posición se mantuvo dentro de un rango pequeño de valores, todos muy próximos a cero.

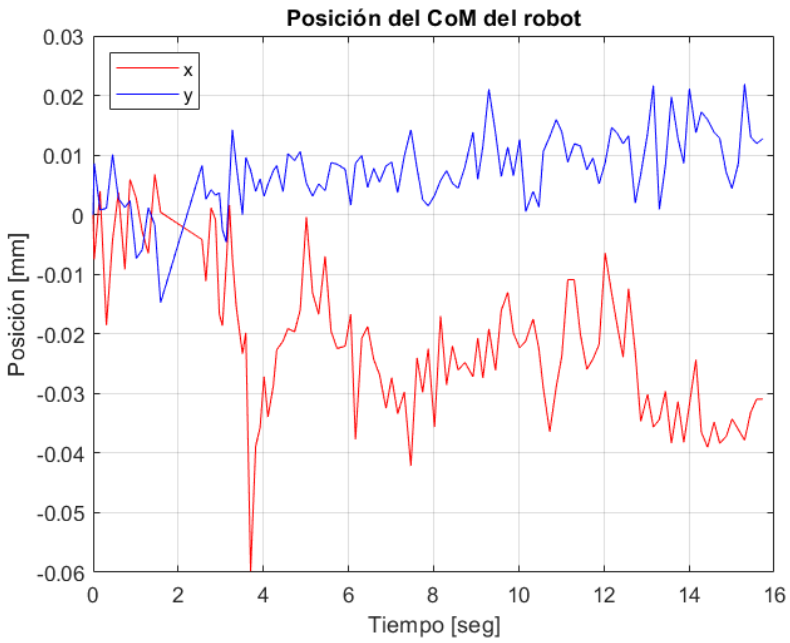


Figura 5.1 Posición del centro de masa del robot durante una prueba sobre una superficie plana sin aplicar perturbaciones.

La Fig. 5.2 muestra el comportamiento de la velocidad a lo largo de la misma prueba. Puede observarse que la velocidad tanto en el eje X como en el eje Y tienen una magnitud muy pequeña a lo largo de toda la prueba. Prácticamente todo el tiempo, la velocidad en cada eje estuvo en el rango de $\pm 0.1\ [\frac{mm}{s}]$. Esta prueba verifica que el controlador funciona correctamente cuando el sistema no es sometido a perturbaciones.

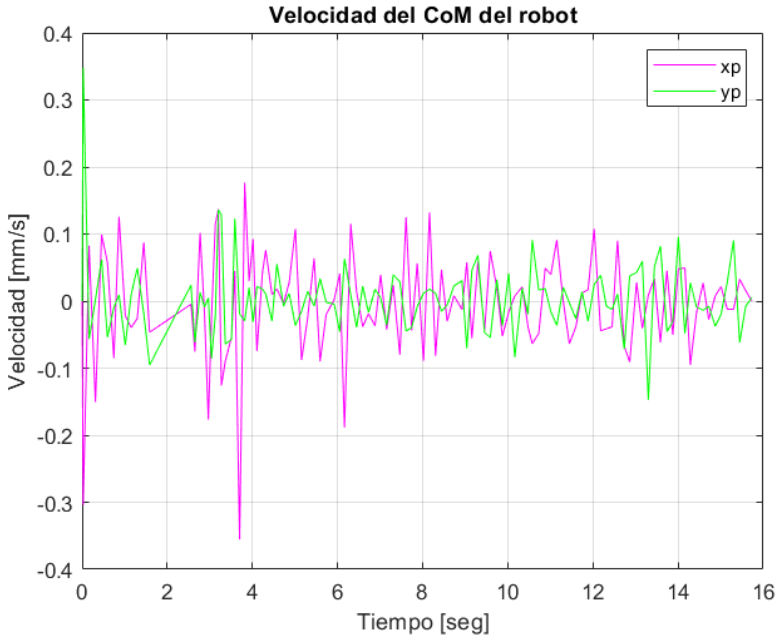


Figura 5.2 Velocidad del centro de masa del robot durante una prueba sobre una superficie plana sin aplicar perturbaciones.

5.1.2 Perturbación en X

Las siguientes pruebas se realizaron con el robot de pie sobre una superficie plana y se aplicaron perturbaciones al robot sobre el eje X. La Fig. 5.3 muestra la posición y la Fig. 5.4 la velocidad del centro de masa cuando se aplicó una perturbación en sentido positivo. En los resultados de la prueba mostrada, el centro de masa del robot se desplazó $11\ [mm]$ en sentido positivo sobre el eje X, tras esto la respuesta del controlador lleva a un sobrepaso de la posición y a que ocurran tres oscilaciones cada vez más pequeñas, hasta que cerca de 4 segundos después de la perturbación, la posición se estabiliza nuevamente. La velocidad del CoM ilustrada en la figura muestra las oscilaciones ya mencionadas de la posición y alcanza valores positivos y negativos de $90\ [mm]$, reduciéndose gradualmente hasta estabilizarse.

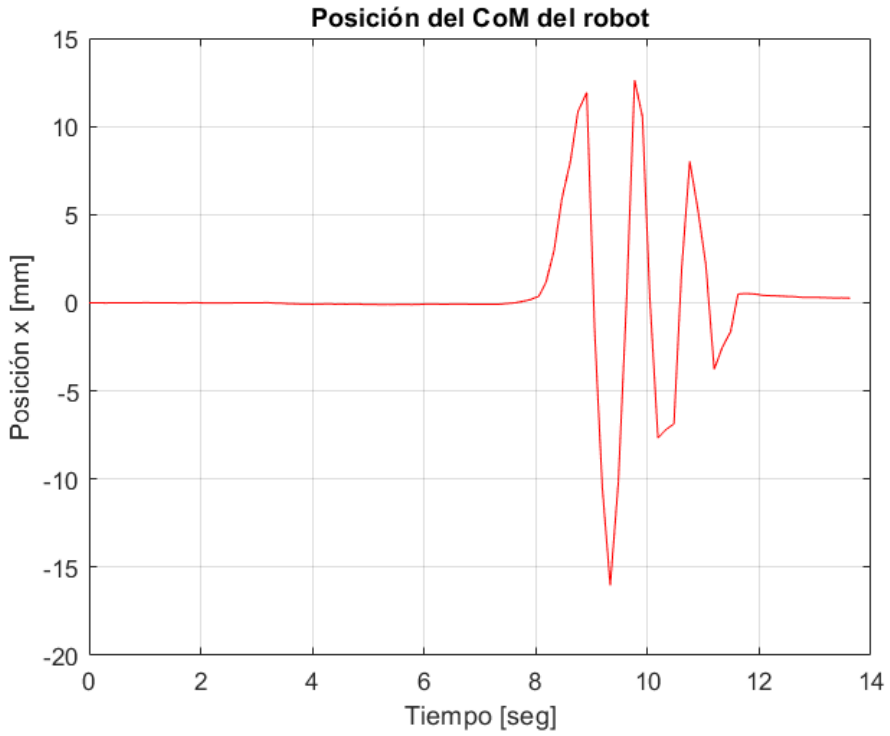


Figura 5.3 Posición del centro de masa del robot durante una prueba en una superficie plana aplicando una perturbación en el eje X en sentido positivo.

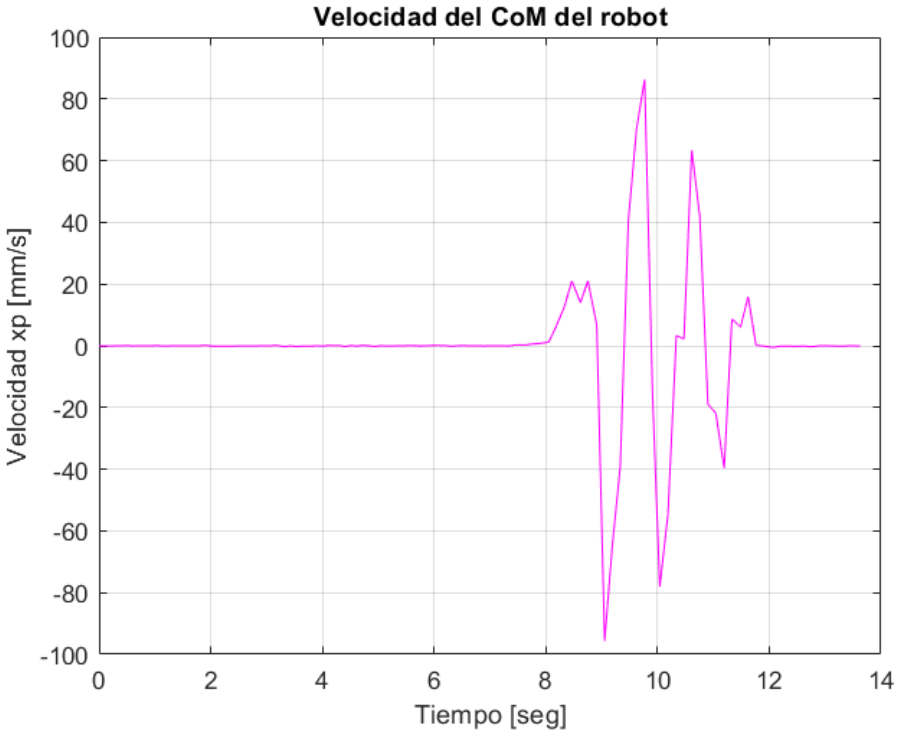


Figura 5.4 Velocidad del centro de masa del robot durante una prueba en una superficie plana aplicando una perturbación en el eje X en sentido positivo.

La Fig. 5.5 muestra la posición del CoM durante una prueba en la que se aplicó una perturbación en sentido negativo. La posición alcanzó $-5.1\ [mm]$ y, como respuesta ante la perturbación, la salida tuvo un sobrepaso de $10\ [mm]$. Luego, se presentaron un par de oscilaciones hasta que, transcurridos 3 segundos, el sistema se estabiliza.

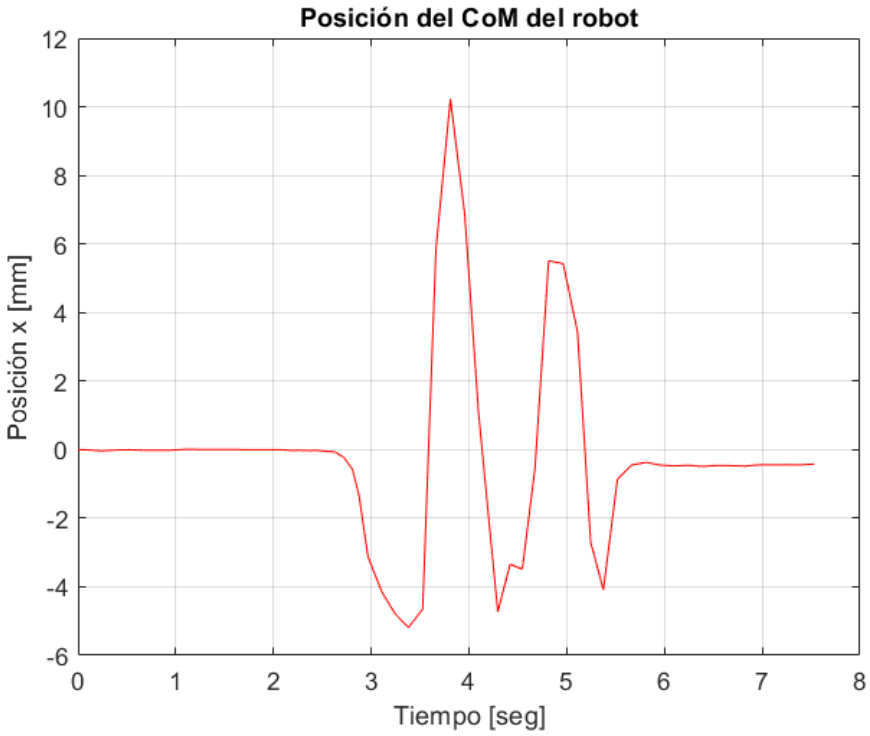


Figura 5.5 Posición del centro de masa del robot durante una prueba en una superficie plana aplicando una perturbación en el eje X en sentido negativo.

En una prueba con características similares, pero aplicando una perturbación ligeramente mayor, las oscilaciones estuvieron presentes durante poco más de 8 segundos. La Fig. 5.6 muestra la posición en X del CoM durante ésta. Puede observarse que el CoM alcanza un valor cercano a $-7\ [mm]$ y el controlador responde llevándolo a un sobrepaso de $12\ [mm]$. Debido a que la perturbación fue mayor, el sobrepaso también fue mayor, además, este efecto en la salida ocasionó que las oscilaciones se mantuvieran durante un período más largo.

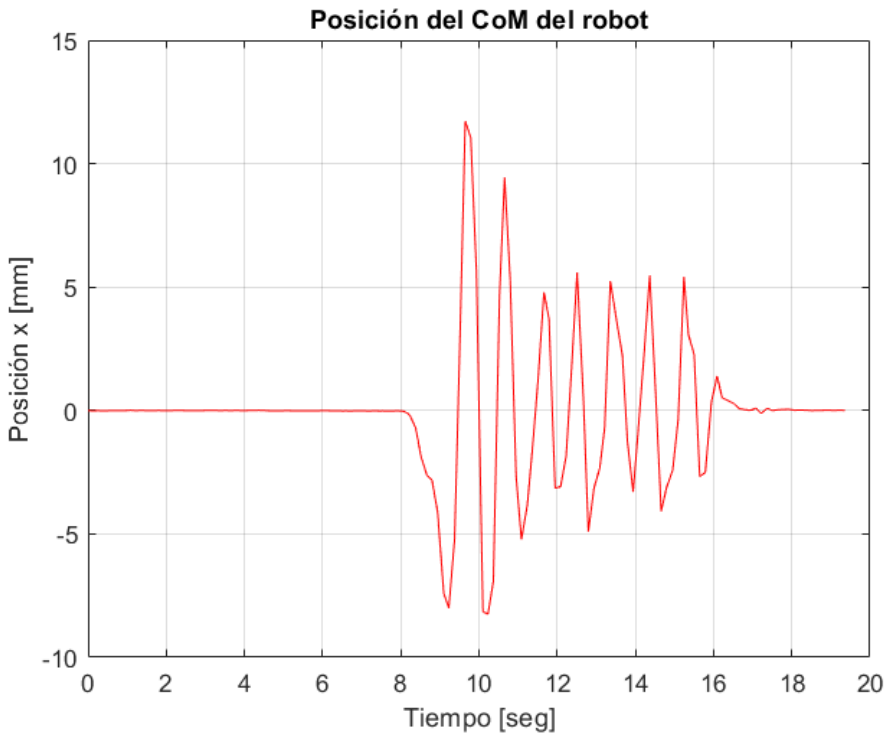


Figura 5.6 Posición del centro de masa del robot durante una prueba en una superficie plana aplicando una perturbación en el eje X en sentido negativo.

5.1.3 Perturbación en Y

Una vez verificado el funcionamiento del sistema en el eje X, se procedió a realizar pruebas ante perturbaciones en el eje de las ordenadas. En estas pruebas el robot también se soportó sobre una superficie plana. Los resultados mostrados en las Figs. 5.7 y 5.8 corresponden a una prueba aplicando una perturbación en el eje Y en sentido positivo al robot bípedo. La perturbación llevó el CoM del robot a cerca de $10\,[mm]$. La gráfica muestra que, en primera instancia, la respuesta del controlador es la correcta, acercando el valor de la posición a cero, sin embargo, ocurre un sobrepaso mucho más grande que el ocurrido en las pruebas en el eje X. Este sobrepaso llevó a que el robot cayera al intentar compensarlo. Además, la cinemática inversa para el Bioloid GP no ofreció una buena aproximación en la dirección del eje Y, lo que ocasionó más inconvenientes al momento que el robot intentaba responder como el controlador lo solicitaba.

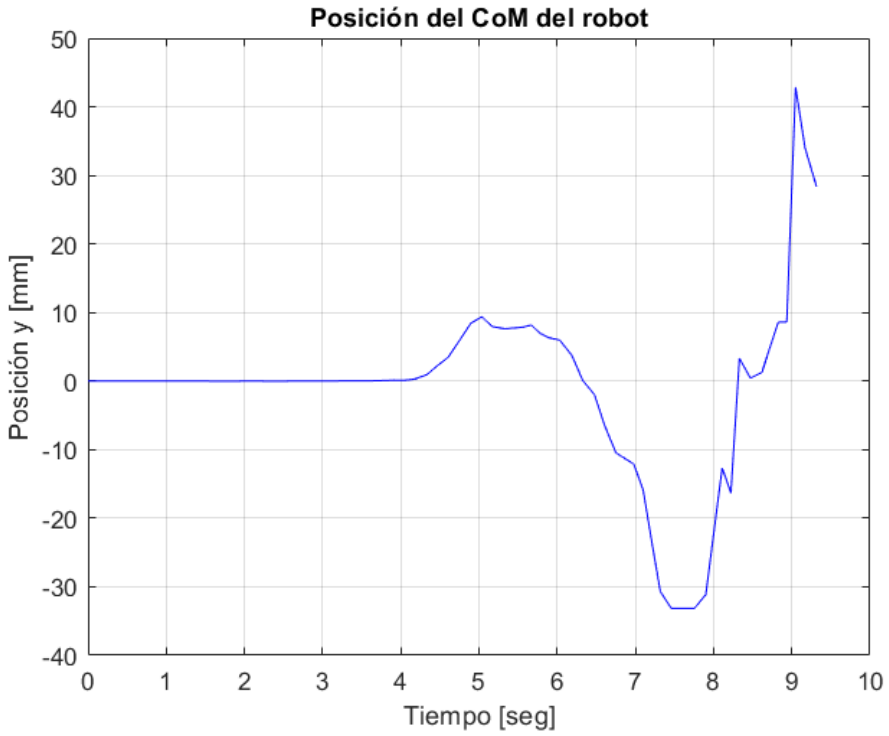


Figura 5.7 Posición del centro de masa del robot durante una prueba en una superficie plana aplicando una perturbación en el eje Y en sentido positivo.

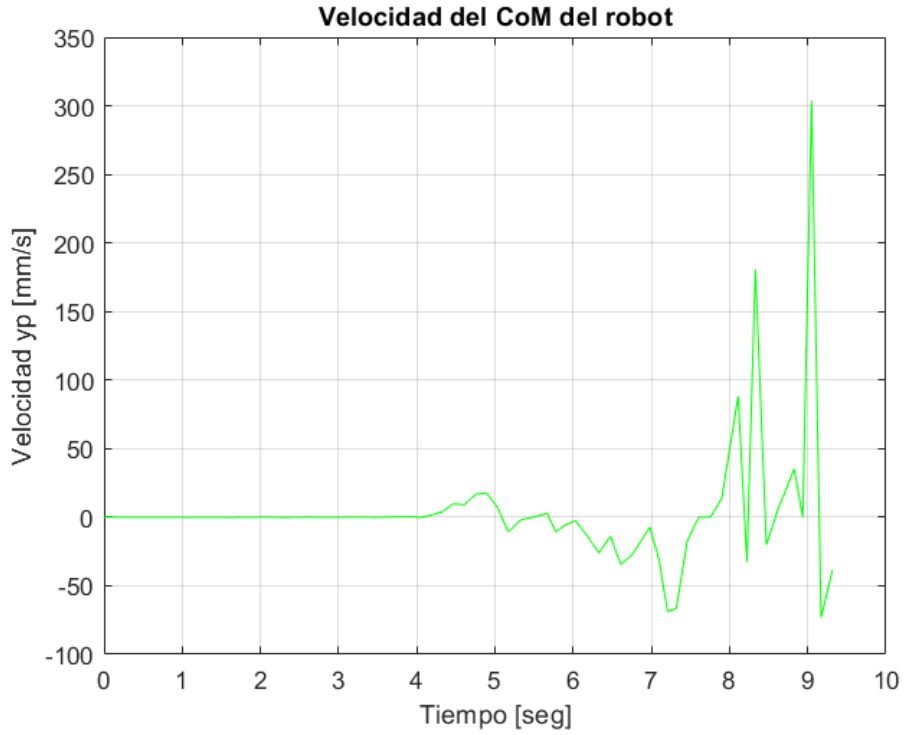


Figura 5.8 Velocidad del centro de masa del robot durante una prueba en una superficie plana aplicando una perturbación en el eje Y en sentido positivo.

En la Fig. 5.9 se muestra la respuesta obtenida ante una perturbación en sentido negativo sobre el eje Y. Como en el caso anterior, la posición del CoM tuvo un sobrepaso muy grande y el robot no logró estabilizarse, volcándose. En esta prueba, la perturbación fue muy pequeña y aún así, el robot no fue capaz de responder satisfactoriamente.

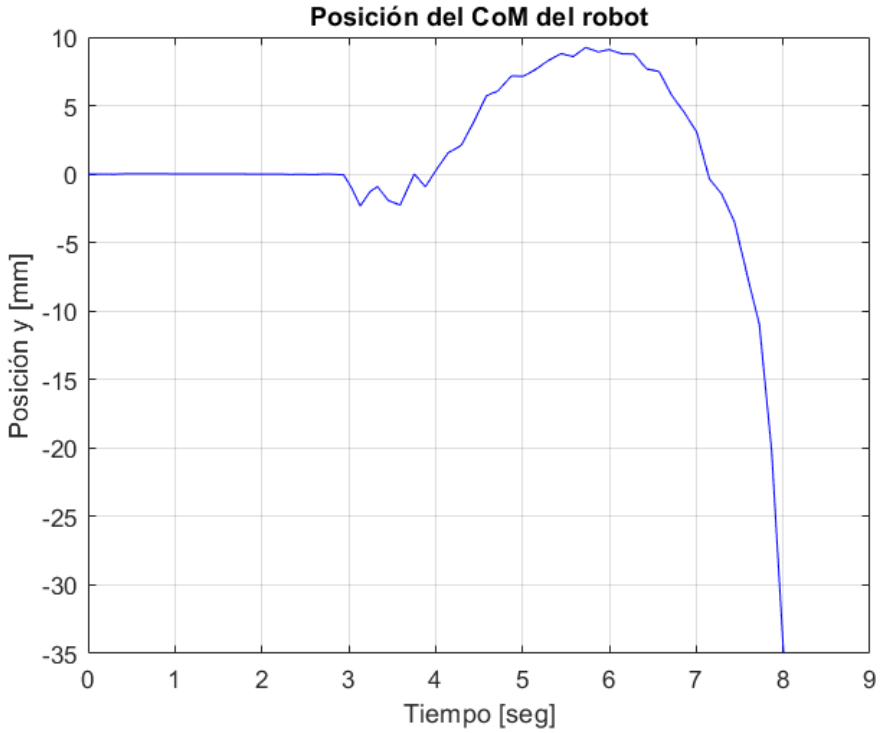


Figura 5.9 Posición del centro de masa del robot durante una prueba en una superficie plana aplicando una perturbación en el eje Y en sentido negativo.

5.2. Superficie inclinada

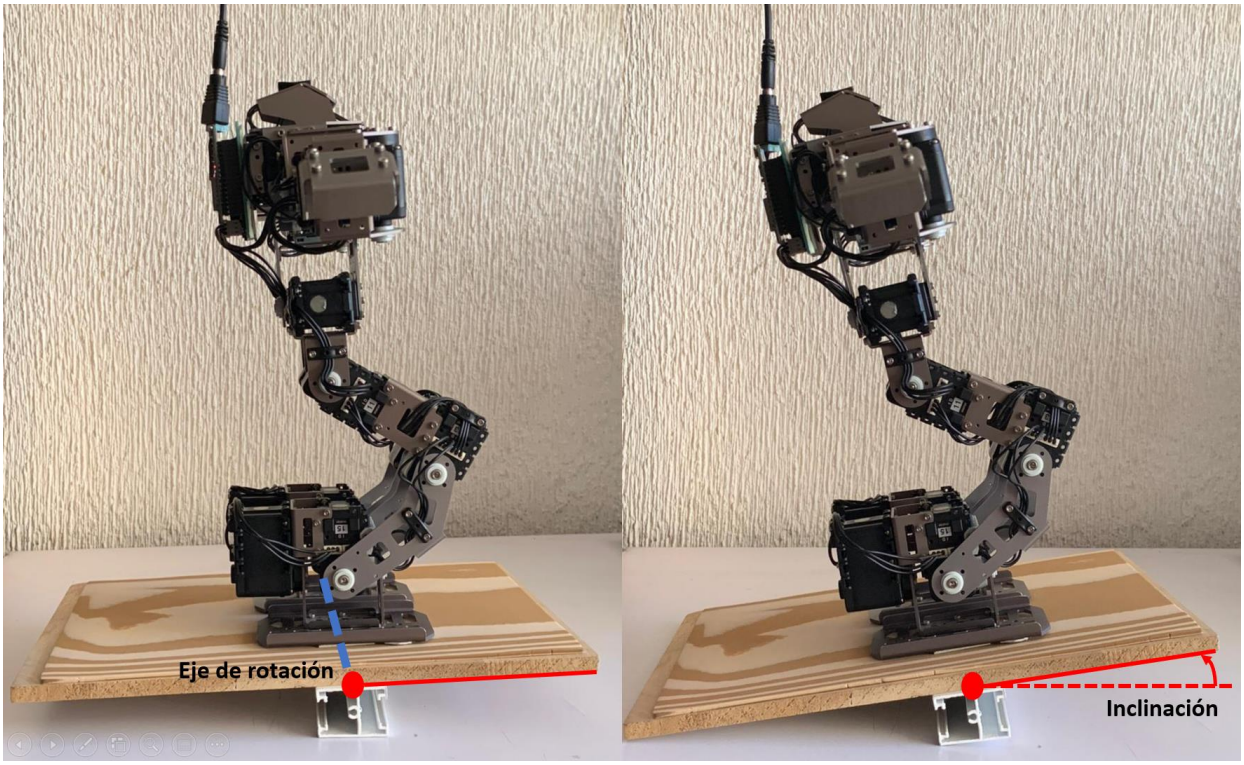


Figura 5.10 Inclinación de la superficie de soporte del robot

Por último, se probó el funcionamiento del controlador inclinando la superficie de soporte del robot. La posición inicial del robot fue de pie con una inclinación de 90° respecto al suelo, luego, la superficie se inclinó gradualmente como se ilustra en la Fig. 5.10. Estas pruebas se realizaron únicamente con perturbaciones en el eje X debido a los inconvenientes detectados en las pruebas anteriores.

En las Figs. 5.11 y 5.12 se muestra el comportamiento de la posición del CoM durante una de las pruebas. En esta prueba, la inclinación de la superficie llevó al centro de masa del robot a una posición de 3.3 [mm] en el eje X aproximadamente y el robot se estabilizó 3 segundos después de iniciada la perturbación con un error de 1.3 [mm] (Fig. 5.11). Luego, la inclinación alejó el CoM una distancia cercana a los 7 [mm] en sentido positivo; sin embargo, las oscilaciones de la respuesta provocaron que el robot cayera (Fig. 5.12).

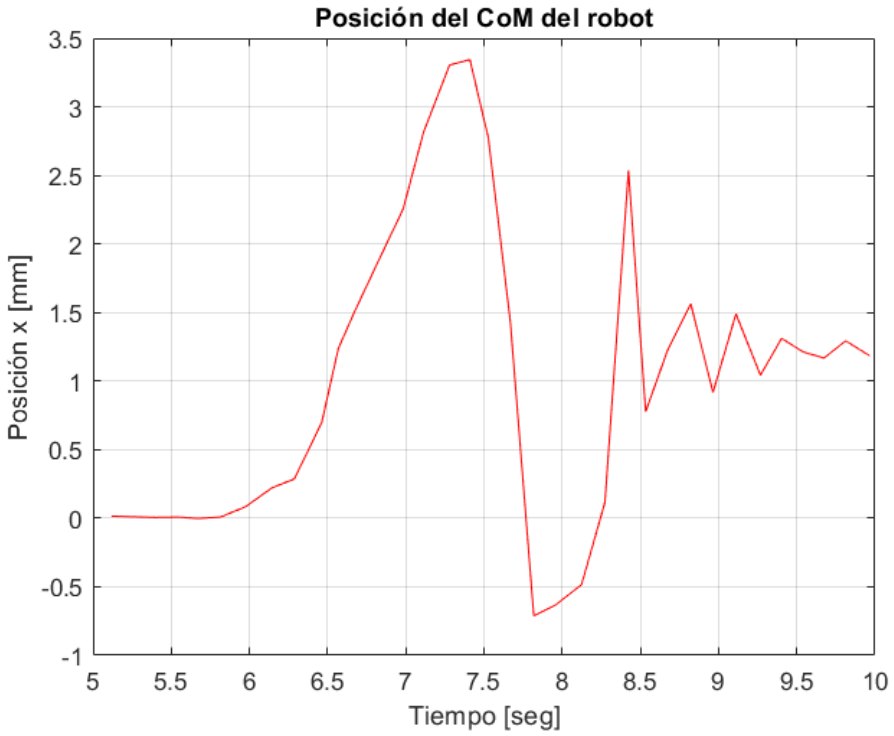


Figura 5.11 Posición del centro de masa del robot desde t=5 [seg] hasta t=10 [seg] durante una prueba inclinando la superficie de soporte.

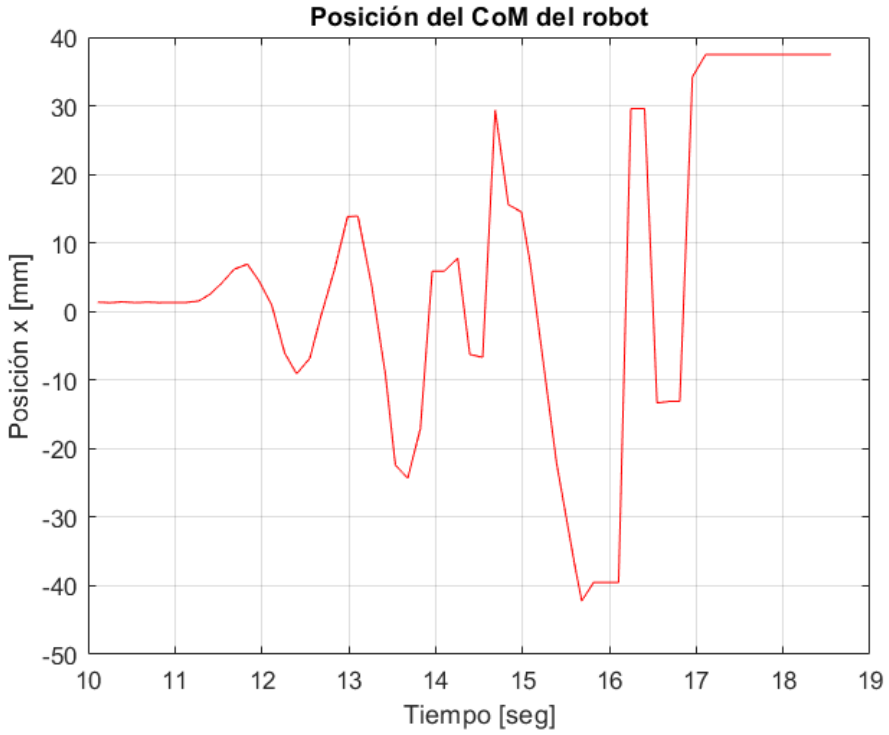


Figura 5.12 Posición del centro de masa del robot desde t = 10 [seg] hasta t= 19 [seg] durante una prueba inclinando la superficie de soporte.

En otra prueba similar, en que la inclinación de la superficie incrementaba el valor de la posición del CoM, la respuesta del sistema fue satisfactoria antes las primeras perturbaciones aplicadas. Las Fig. 5.13, 5.14 y 5.15 muestran los resultados de la prueba mencionada y pueden observarse tres perturbaciones a lo largo de la prueba. Una primera perturbación de $1.3\ [mm]$ fue compensada en cerca de 4 segundo, sin sobrepaso. La segunda perturbación llevó al CoM a $5.2\ [mm]$ y el robot se estabilizó en 5 segundos tras tres oscilaciones del sistema, este caso es similar al de los resultados obtenidos con el robot sobre una superficie plana. En la tercera perturbación, la posición alcanzó $9.5\ [mm]$, pero la respuesta provocó la caída del robot.

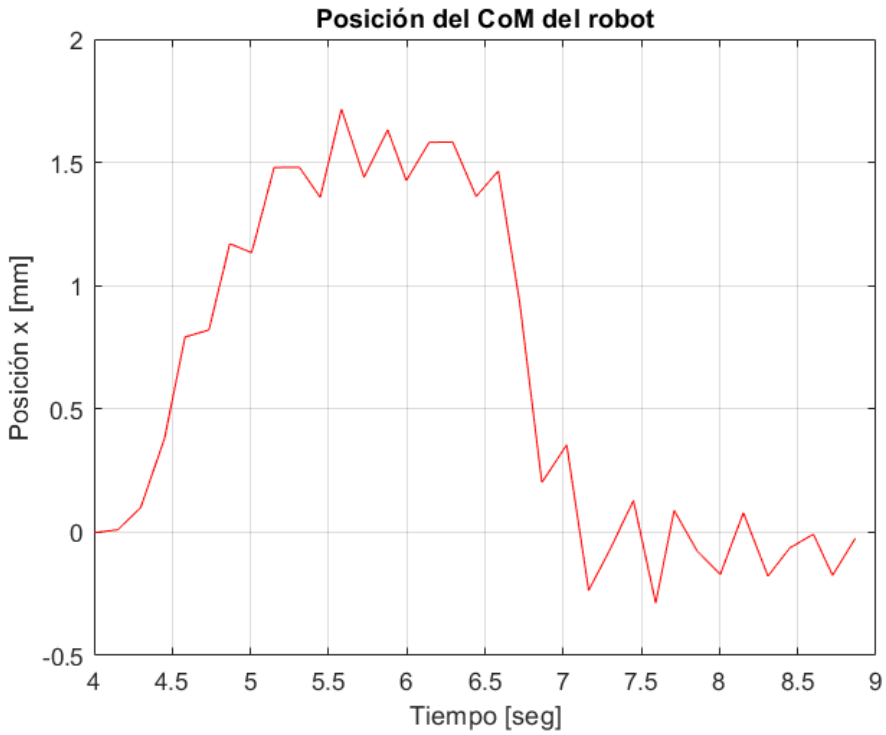


Figura 5.13 Posición del centro de masa del robot entre los segundos 4 y 9 durante una prueba inclinando la superficie de soporte.

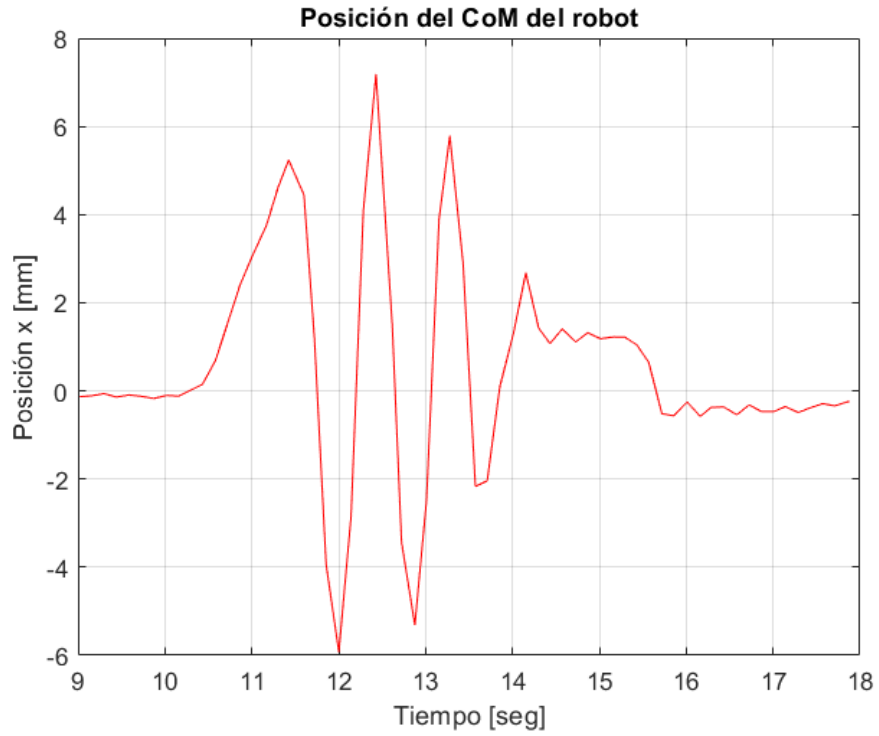


Figura 5.14 Posición del centro de masa del robot entre los 9 y 18 segundos durante una prueba inclinando la superficie de soporte.

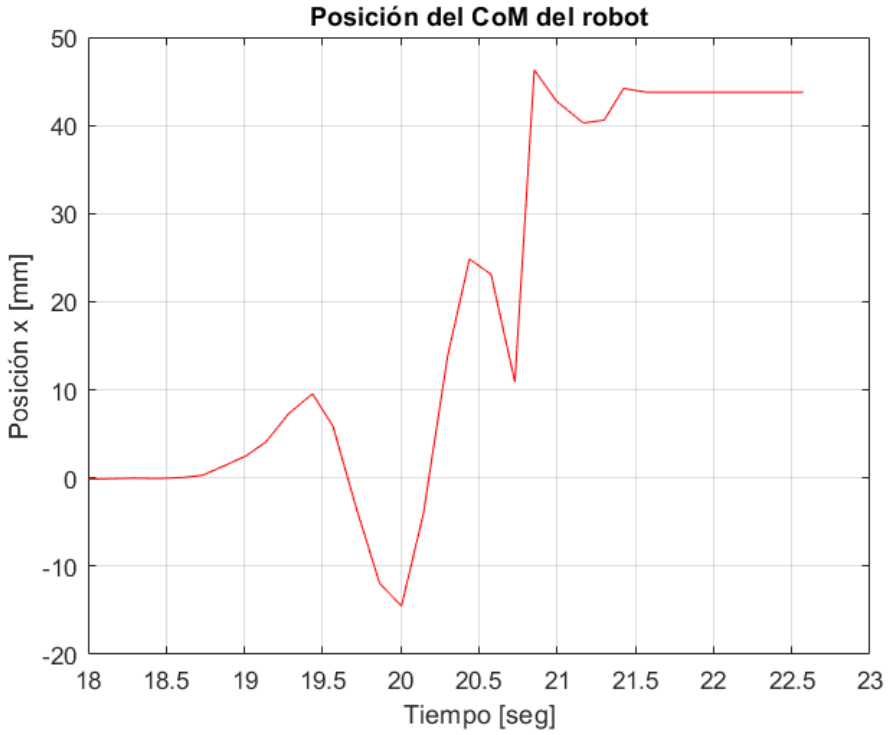


Figura 5.15 Posición del centro de masa del robot después de los 18 segundos durante una prueba inclinando la superficie de soporte.

La gráfica de la Fig. 5.16 muestra la posición del centro de masa del Bioloid GP durante una prueba en la que se inclinó la superficie de soporte del robot de manera que la posición decreciera. En la prueba, la perturbación llevó al CoM a $-4.5\ [mm]$. La respuesta del sistema ocasionó oscilaciones durante 10 segundos hasta estabilizarse. La oscilación, como en los casos anteriores, se debió al sobrepaso.

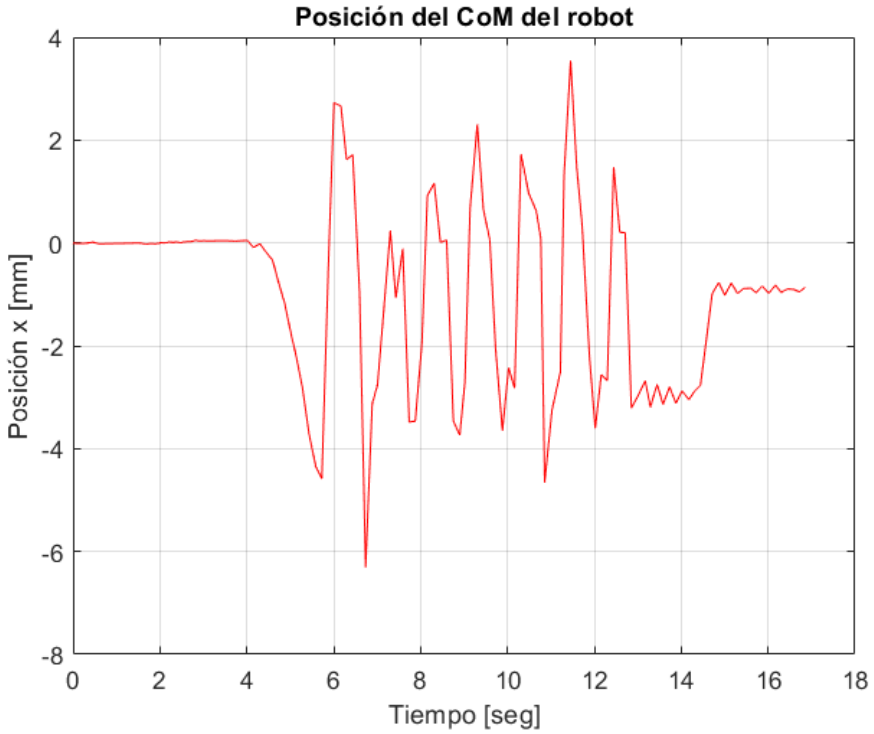


Figura 5.16 Posición del centro de masa del robot durante una prueba inclinando la superficie de soporte.

Los resultados ilustrados en la Fig. 5.17 corresponden a una prueba en que la inclinación de la superficie de soporte se incrementó gradualmente en cuatro ocasiones, luego, se redujo la inclinación de la superficie dos veces. El controlador ofreció una respuesta satisfactoria ante las perturbaciones descritas y puede verse en el gráfico que no tardó más de 3 segundos en estabilizarse, sin embargo, las perturbaciones eran de pocos milímetros. El robot no cayó nunca durante los 40 segundos de la prueba. Cuando la posición del CoM creció en sentido positivo a causa de la perturbación, la respuesta tuvo más oscilaciones. En la Fig. 5.18 se observa una secuencia de imágenes del video de la prueba descrita.

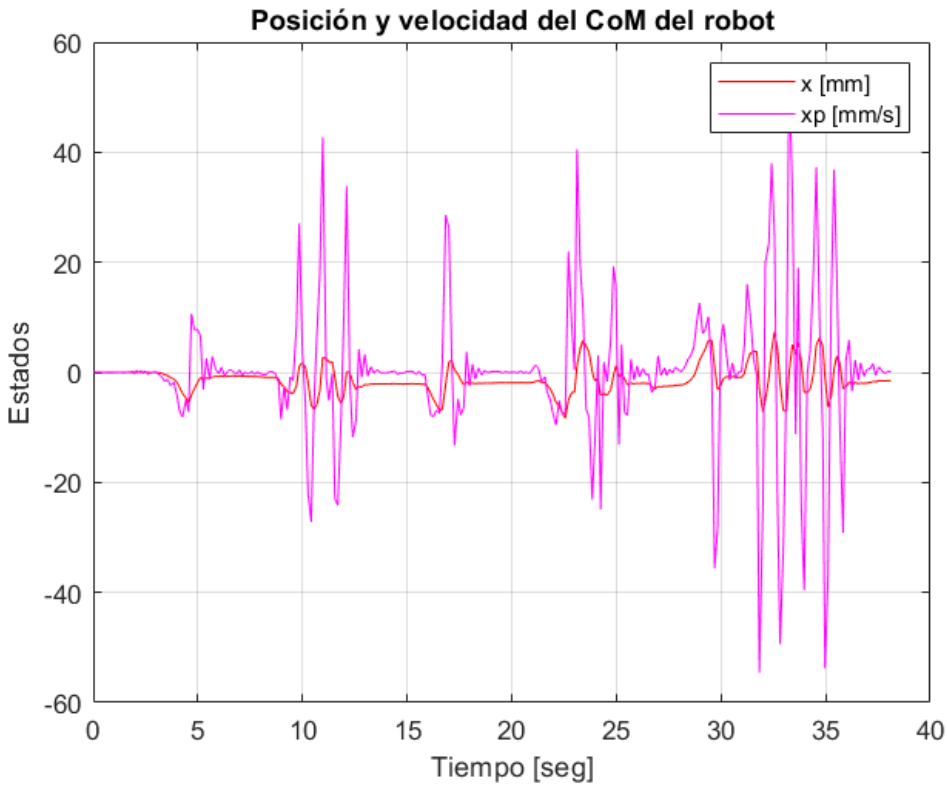


Figura 5.17 Posición y velocidad del centro de masa del robot durante una prueba inclinando la superficie de soporte.

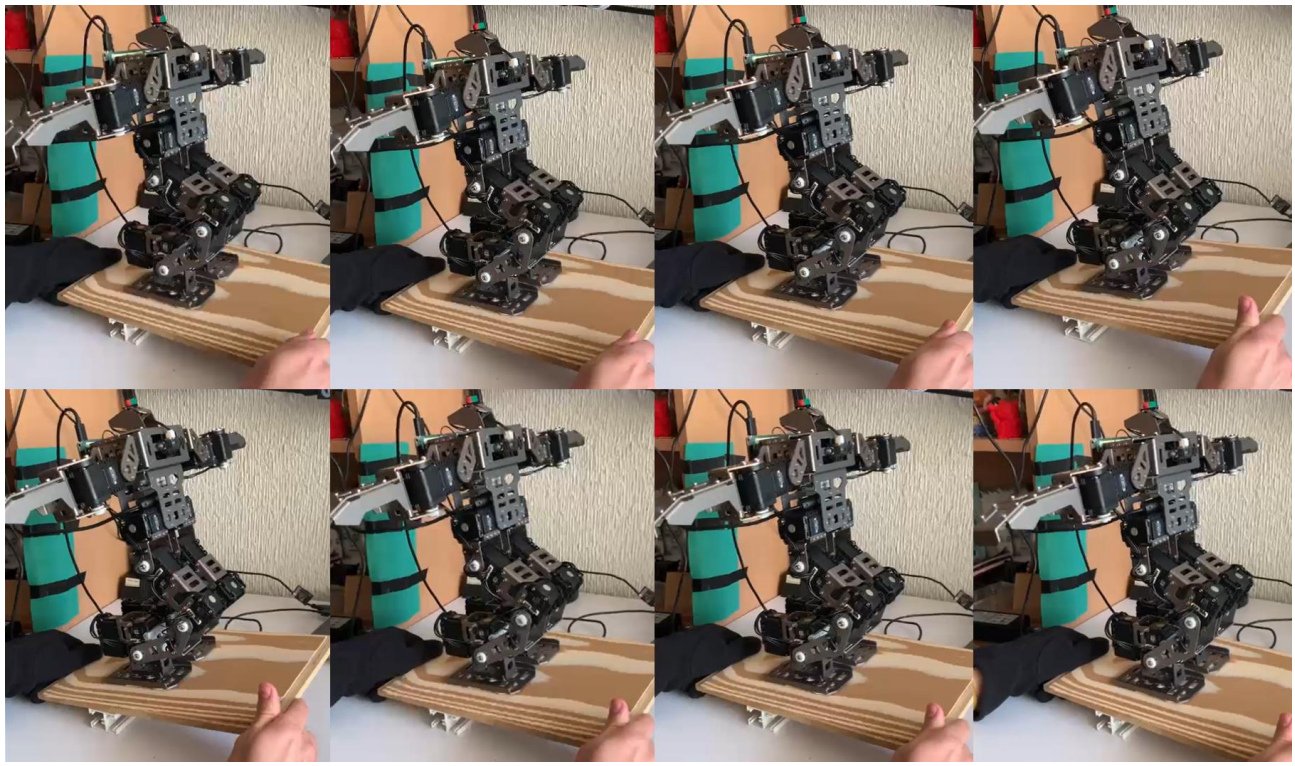


Figura 5.18 Secuencia de imágenes de una prueba inclinando la superficie de soporte.

5.3. Análisis de resultados

Como primera concepción del controlador, se verificó que su funcionamiento fuera el correcto al no enfrentarse a ninguna perturbación. En este caso, el robot bípedo permaneció estático, tanto la posición como la velocidad en ambos ejes mantuvieron valores muy pequeños, lo que explica que el controlador no ocasionara movimiento en el robot.

Las pruebas con perturbación en el eje X mostraron dos puntos relevantes: 1) el tiempo de estabilización fue mayor al obtenido mediante la simulación del modelo simplificado y 2) en todas las pruebas se presentó un sobrepaso en el estado correspondiente a la posición. En la simulación, el sistema se estabilizaba en un segundo, mientras que el sistema durante las pruebas tardaba cerca de 2 segundos luego de dejar de aplicar la perturbación. Cuando las perturbaciones eran más grandes, las oscilaciones que presentaba el sistema tenían una mayor duración. Esta demora en la estabilización se explica principalmente por el sobrepaso mencionado y este, a su vez, es causa del algoritmo empleado para la implementación, en que se toma la diferencia del estado medido y el estado calculado mediante la matriz exponencial, la imposibilidad de aplicar directamente la señal de control conlleva que se presente este error dado que se suma esta diferencia a la posición actual del robot. Además, la respuesta del controlador exige que se alcance el valor de cero en un segundo y esto hace que la acumulación del error también sea grande.

Las pruebas perturbando la posición del CoM del robot en el eje Y mostraron una respuesta inicial en la dirección adecuada, sin embargo, el sobrepaso que se presentaba en el eje X también se encontró en estas pruebas. Como consecuencia del sobrepaso, el controlador intentaba compensarlo, pero ocasionaba la caída del robot. El problema no era ocasionado únicamente por el algoritmo, sino que la cinemática inversa del Bioloid GP no ofrecía una buena aproximación en el eje Y.

En las pruebas inclinando la superficie sobre la que se soportaba el robot, también se obtuvieron tiempos de estabilización mayores a los de la simulación. La respuesta adecuada en estas pruebas se obtenía ante perturbaciones menores. Sobre una superficie plana, el sistema se estabilizaba cuando la posición alcanzaba cerca de $10\ [mm]$ debido a una perturbación, mientras que, inclinando la superficie, la posición no debía sobrepasar los $5\ [mm]$ en cualquier sentido porque provocaba la caída del robot. Al inclinar la superficie de soporte gradualmente, el controlador implementado respondió de manera correcta y mantuvo al robot de pie.

Conclusiones y trabajo a futuro

El objetivo planteado para este trabajo fue diseñar e implementar un controlador para lograr la estabilidad estática de un robot bípedo de 10 GDL. La propuesta de un control LQR basado en un modelo simplificado del robot y la instrumentación utilizada permitieron cumplirlo. Sin embargo, el análisis de los resultados muestra algunas problemáticas que enfrenta el sistema y en las que se podría poner más atención para mejorar el funcionamiento del controlador.

Respecto al diseño, se propuso un controlador LQR basado en el modelo simplificado de un péndulo invertido asumiendo que la altura a la que se encontraba el centro de masa era siempre constante. Fue posible simular la respuesta con el control para tener una idea del comportamiento que tendría el sistema real, teniendo en cuenta que la simplificación del modelo dinámico y las suposiciones hechas influyen en que las pruebas y las simulaciones tengan diferencias. La ventaja del control propuesto está en que se puede dar mayores pesos a las entradas para que se regulen más y que la respuesta del sistema sea más lenta, reduciendo así el sobrepaso que se presentó en los resultados. También es posible plantear diferentes pesos en cada eje coordenado considerando que la dinámica real del robot cambia en cada dirección.

El control de la estabilidad estática ha sido poco trabajado en general para la robótica bípeda y en particular en la Facultad de Ingeniería de la UNAM, por lo que resulta novedosa la propuesta de un control de estabilidad estática basado en modelo dentro de la facultad. Cabe mencionar que por primera vez se documenta un control de este tipo en el Bioloid GP.

Por otro lado, la implementación del control propuesto permitió verificar su funcionamiento pese a que los mayores retos y los inconvenientes detectados en los resultados se encuentran en esta parte. La imposibilidad de aplicar directamente la señal de control en el sistema exige que se utilice la matriz exponencial para calcular los estados futuros. Al codificar esto, se incrementa la diferencia entre los estados medidos y los estados calculados a la posición actual de la cadera y se presenta una acumulación excesiva de esta diferencia, afectando la respuesta del sistema al generar sobrepaso y varias oscilaciones.

En este trabajo también se aportó una primera noción de la cinemática inversa del robot bípedo Bioloid GP. A pesar de que el robot cuenta con 10 GDL, únicamente se consideraron 8 GDL; es decir, cuatro por pierna, esto con la finalidad de simplificar el análisis. Con el mismo fin, se simplificó la geometría de los eslabones en el plano sagital. En suma, estas suposiciones produjeron que la aproximación sea buena cerca del origen del plano XY, pero mala entre más se alejaba del origen. En el eje Y, la aproximación resultaba mala y no permitió probar el funcionamiento del sistema en esa dirección.

A pesar de ser de forma aproximada, la obtención de la cinemática inversa permitió tener un tiempo de procesamiento bajo a comparación de lo que podría hacerse mediante métodos numéricos. Así, la cinemática inversa del Bioloid GP es un trabajo a futuro; en un

principio, sin simplificaciones de la geometría de los eslabones, y posteriormente, considerando la totalidad de los grados de libertad del robot.

De igual manera, la medición de los estados del sistema fue un factor importante para cumplir el objetivo. Se utilizó una cámara web para medir la posición y, a partir de este dato, la velocidad; sin embargo, la posición medida correspondió a la de la cabeza del robot y no a la del centro de masa de éste. No obstante, la medición ofrecía información buena para probar el funcionamiento del controlador.

Otra vertiente de trabajo posterior sería la instrumentación del Bioloid GP para que las condiciones de las pruebas fueran más rigurosas y las medidas más confiables. Por ejemplo, se puede utilizar equipo de procesamiento de imágenes más especializado, diseñar un banco de pruebas específicamente para este tipo de pruebas o implementar una unidad de medición inercial (IMU) con un algoritmo para la obtención de la posición y la velocidad.

Por último, los resultados mostraron que sobre una superficie plana el robot bípedo se estabilizaba ante perturbaciones que lo llevaran a ± 10 $[mm]$ aproximadamente en el eje X, mientras que inclinando la superficie este valor se reducía a ± 5 $[mm]$. De ahí que, otro trabajo a futuro esté en robustecer el control de la estabilidad estática mediante acciones de reflejo del robot como agacharse o dar un paso para garantizar la estabilidad estática en un rango más amplio.

Los resultados obtenidos abren la puerta a más actividades relacionadas con la robótica bípeda dentro de la Facultad de Ingeniería de la UNAM dado que sugieren ampliar los puntos desarrollados en este trabajo tanto para complementar la investigación como para que se generen nuevos avances en el ámbito de la robótica humanoide.

Referencias

[1] K. S. Fu, R. C. Gonzalez y C. S. G. Lee, *Robotics: control, sensing, vision and intelligence*. Nueva York: McGraw-Hill, 1987, pp. 1-6.

[2] S. B. Niku, *Introduction to robotics: analysis, control, applications*, 2a edición. Nueva Jersey: John Wiley & Sons, 2011, pp. 1-5.

[3] Encadenados, (2013, oct. 29). "Metrópolis (Metropolis, 1927)." [En línea]. Disponible en: https://www.encadenados.org/rdc/rashomon/123-rashomon-n-81-fritz-lang/3472-metropolis-metropolis-1927.

[4] M. E. Foster, (2015, dic. 17). "The Conversation." [En línea]. Disponible en: https://theconversation.com/how-long-until-we-can-build-r2-d2-and-c-3po-52400.

[5] K. Capek, *R.U.R. (Rossum's Universal Robots)*, Styx Classics, 1921.

[6] L. W. Tsai, *The mechanics of serial and parallel manipulators*. Nueva York: John Wiley & Sons, 1999, pp. 1-6.

[7] M. R. Arbulú, "Stable locomotion of humanoid robots based on mass concentrated model," Tesis de doctorado, Depto. de Ing. de Sistemas y Automática, Universidad Carlos III de Madrid, España, 2009, pp. 36-65.

[8] Waseda University Humanoid Research Institute, (s.f.). "WABOT." [En línea]. Disponible en: http://www.humanoid.waseda.ac.jp/booklet/kato_2.html.

[9] Waseda University Takanishi Laboratory, (s.f.). "Research." [En línea]. Disponible en: http://www.takanishi.mech.waseda.ac.jp/top/research/index.htm.

[10] Honda, (s.f.). "Robot Development History." [En línea]. Disponible en: https://global.honda/innovation/robotics/robot-development-history.html.

[11] Honda, (s.f.). "ASIMO." [En línea]. Disponible en: https://global.honda/innovation/robotics/ASIMO.html#2011.

[12] IEEE, (s.f.). "Robots: Your guide to the world of robotics." [En línea]. Disponible en: https://robots.ieee.org/robots/?t=sort.

[13] E. Ackerman y E. Guizzo, "Ford self-driving vans will use legged robots to make deliveries," *IEEE Spectrum*. [En línea]. Disponible en: https://spectrum.ieee.org/automaton/robotics/humanoids/ford-self-driving-vans-will-use-legged-robots-to-make-deliveries?utm_source=robots.ieee.org.

[14] M. Dekker, "Zero-moment point for stable biped walking," Reporte de prácticas, Depto. De Ing. Mecánica, Eindhoven University of Technology, Países Bajos, 2009, pp. 3-8.

[15] S. González Mejía, J. M. Ramírez Scarpetta y E. J. Avella Rodríguez, "Técnicas de control para el balance de un robot bípedo: un estado del arte," *Tecnura*, vol. 19, no. 43, pp. 133-156, enero-marzo, 2015.

[16] F. Sáenz Navarro, "Generación de trayectorias para el tren inferior del robot humanoide TEO subiendo escaleras," Proyecto de fin de carrera, Depto. de Ing. de Sistemas y Automática, Universidad Carlos III de Madrid, España, 2012, pp. 10-19,

34-37.

[17] Robotis, (s.f.). “Bioloid Series.” [En línea]. Disponible en: https://www.robotis.us/bioloid-1/.

[18] Robotis, (s.f.). “Bioloid GP Manual.” [En línea]. Disponible en: https://emanual.robotis.com/docs/en/edu/bioloid/gp/.

[19] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Yokoi y H. Hirukawa, “A Realtime Pattern Generator for Biped Walking,” en *IEEE International Conference on Robotics & Automation*, 2002, pp. 31-37.

[20] S. Kajita, O. Matsumoto y M. Saigo, “Real-time 3D walking pattern generation for a biped robot with telescopic legs,” en *IEEE International Conference on Robotics & Automation*, 2001, pp. 2299-2306.

[21] N. S. Nise, *Control systems engineering*, 6a edición. Pomona: John Wiley & Sons, Inc., 2011, pp. 2-10, 301-338, 663-722.

[22] K. Ogata, *Ingeniería de control moderna*, 5a edición. Madrid: Pearson Educación, 2010, pp. 793-806.

[23] Espressif Systems, "ESP32-WROOM-32D & ESP32-WROOM-32U Datasheet V1.9,” 2019.

[24] Espressif, (s.f.). “Development Boards.” [En línea]. Disponible en: https://www.espressif.com/en/products/devkits.

[25] A. Bhargav Anand. (2021). *Tracking red color objects using matlab*. [En línea]. Disponible en: https://www.mathworks.com/matlabcentral/fileexchange/28757-tracking-red-color-objects-using-matlab, Consultado en: Dic. 2, 2020.

[26] Robotis, (s.f.). “Dynamixel.” [En línea]. Disponible en: http://www.robotis.us/dynamixel/.

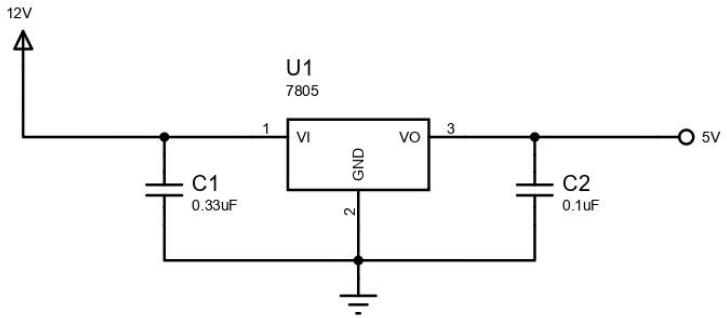
[27] Robotis, (s.f.). “AX-18A Manual.” [En línea]. Disponible en: https://emanual.robotis.com/docs/en/dxl/ax/ax-18a/.

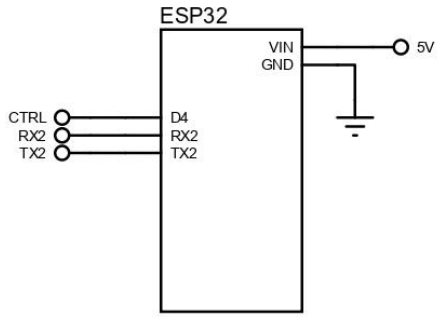
[28] Fairchild Semiconductor Corporation, “MC78XX/LM78XX/MC78XXA3-Terminal 1A Positive Voltage Regulator,” 2001.

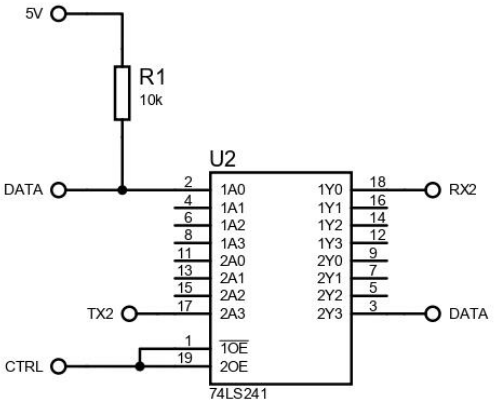
[29] Texas Instruments, “Octal buffers and line drivers with 3-state outputs”, Dallas, EEUU, 2002.

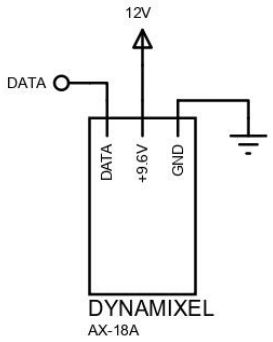
[30] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi y H. Hirukawa, “Biped Walking Pattern Generation by using Preview Control of Zero-Moment Point,” en *IEEE International Conference on Robotics & Automation*, 2003, pp. 1620-1626.

Anexos









Nombre del archivo: Conexiones.pdsprj		Fecha: 25/01/2021
Título: Conexiones Bioloid GP		Página: 1 de 1
Diseñó: Luis Ulises García Aguilera		Hora: 12:35:24 p. m.
UNAM_FI	Ing. Mecatrónica	

Anexo 2. Programa del microcontrolador

```
/*  Programa para el control de la estabilidad estática de un robot bípedo de 10 GDL.
 *
 *  Autor: Luis Ulises García Aguilera
 *  Robot: Bioloid GP
 */

//Librerias utilizadas
#include "Wire.h"
#include "math.h"
#include "BluetoothSerial.h"
#include "DynamixelSerial.h"
#include "BasicLinearAlgebra.h"
using namespace BLA;

#if !defined(CONFIG_BT_ENABLED) || !defined(CONFIG_BLUEDROID_ENABLED)
#error Bluetooth is not enabled! Please run `make menuconfig` to and enable it
#endif

//Definicion de pines
#define LED 2
#define SDA 21
#define SCL 22
#define RXD2 16
#define TXD2 17
#define CTRL 4

//Dimensiones del robot bipedo en milímetros
const float pL = 35.36;
const float L1 = 32.7;
const float L2 = 53.03+26.06*cos(33.8*PI/180);
const float L5 = 74.28;
const float L6 = 41.09;
const float L7 = 19.36;
const float L8 = 16;

//Parametros del modelo dinamico
const float m = 1.6;    //kg
const float g = 9.78;   //m/s^2
const float Zc = 0.16;  //m

//Otros valores constantes
const float scMotor = 512.0/150.0;  //Conversión de grados a valor numérico para el motor
const float deg = 180/PI;   //Conversión de radianes a grados

//Ganancia de control LQR
ArrayMatrix<2,4> Klqr ={31.9225, 4.0675, 0, 0,
                        0, 0, -31.9225, -4.0675};

//Coeficientes matriz exponencial
const float coefExp = sqrt(g/Zc);
const float coefExp1 = 1/(m*g);
const float coefExp2 = 1/(m*sqrt(g*Zc));
const float tExp = 100;
```

```
//Matriz exponencial
  //Respuesta libre
ArrayMatrix<4,4> MatExpL = {cosh(coefExp*tExp/1000.0),(1/coefExp)*sinh(coefExp*tExp/1000.0),0,0,
                            coefExp*sinh(coefExp*tExp/1000.0),cosh(coefExp*tExp/1000.0),0,0,
                            0,0,cosh(coefExp*tExp/1000.0),(1/coefExp)*sinh(coefExp*tExp/1000.0),
                            0,0,coefExp*sinh(coefExp*tExp/1000.0),cosh(coefExp*tExp/1000.0)};
  //Respuesta forzada
ArrayMatrix<4> MatExpF = {coefExp1*(-1+cosh(coefExp*tExp/1000.0)),
                    coefExp2*(sinh(coefExp*tExp/1000.0)),
                    coefExp1*(1-cosh(coefExp*tExp/1000.0)),
                    -coefExp2*sinh(coefExp*tExp/1000.0)};
  //Funcion
ArrayMatrix<4> MatExponencial(ArrayMatrix<4,4> lib, ArrayMatrix<4> forz, ArrayMatrix<4> cond,
     ArrayMatrix<2> ent);

//Declaracion de variables globales

ArrayMatrix<4> EstadosWm = {0,0,0,0};    //Vector de estados medidos
ArrayMatrix<4> EstadosWc = {0,0,0,0};    //Vector de estados calculados
ArrayMatrix<2> EntradasV = {0,0};       //Vector de entradas

int Px=0, Py=0, Pz=160;      //Parametros posicion y orientacion del efector final
float Phi=90*PI/180;
float rd1, ri1;            //Resultante para pierna derecha e izquierda
float thd1, thd2, thd3, thd4, thi1, thi2, thi3, thi4;   //Variables articulares
const int thd5 = 0, thi5 = 0;   //Variables articulares constantes

BluetoothSerial SerialBT;   //Puerto serial comunicación BT
String linea;   //Linea de información recibida
float sa[4];    //Vector de valores recibidos
int r=0, t=0;

void setup()
{
   Serial.begin(115200);   //Inicia comunicacion serial
   SerialBT.begin("ESP32test"); //Bluetooth: nombre del dispositivo

   //Comunicación serial con el búfer
   Serial2.begin(1000000,SERIAL_8N1,RXD2,TXD2);
   Dynamixel.setSerial(&Serial2);
   Dynamixel.begin(1000000,CTRL);
   delay(1000);

   //Establece par máximo para los motores de las piernas
   int maxT = 512;

   Dynamixel.setMaxTorque(11,maxT);
   Dynamixel.setMaxTorque(12,maxT);
   Dynamixel.setMaxTorque(13,maxT);
   Dynamixel.setMaxTorque(14,maxT);
   Dynamixel.setMaxTorque(15,maxT);
   Dynamixel.setMaxTorque(16,maxT);
   Dynamixel.setMaxTorque(17,maxT);
   Dynamixel.setMaxTorque(18,maxT);

   //Posicion inicial del Robot
```

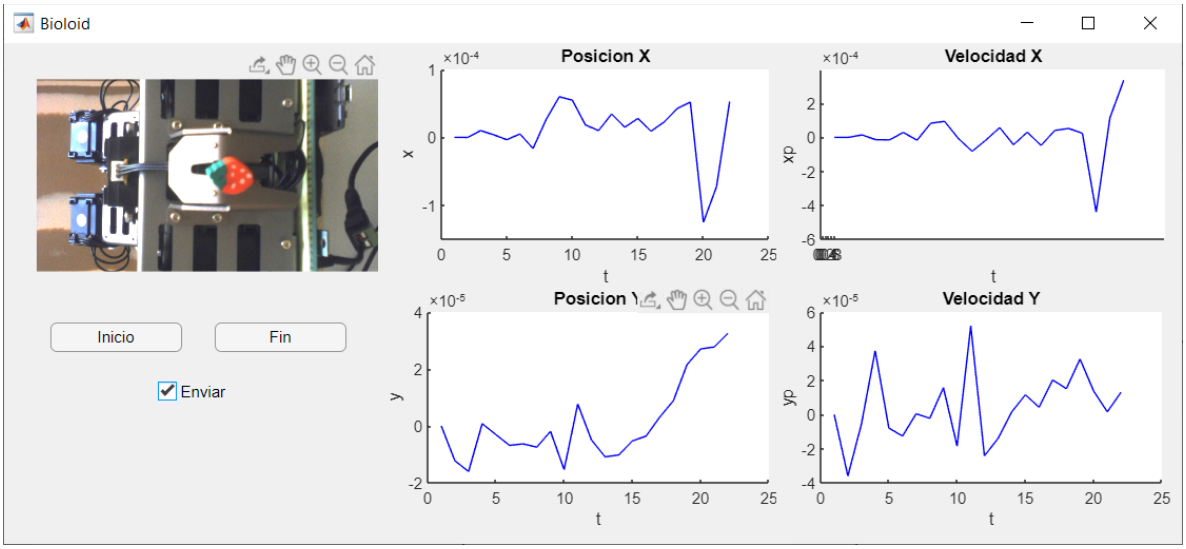
```
    CinematicaInversa();
    EnviarPosicion();
    Dynamixel.move(1,512);    //Posición de los brazos
    Dynamixel.move(2,512);
    Dynamixel.move(3,800);
    Dynamixel.move(4,200);
    Dynamixel.move(5,200);
    Dynamixel.move(6,800);
    delay(2000);

    pinMode(LED,OUTPUT);      //LED indica cuando finalizó la inicialización
    for(int i=0;i<3;i++){
      delay(250);
      digitalWrite(LED,HIGH);
      delay(250);
      digitalWrite(LED,LOW);
    }
}

void loop(){
  while(1){

    //Comunicación BT
    while(!SerialBT.available()){}
    linea = SerialBT.readStringUntil(';');    //Lee cadena hasta ";"

    //Medicion de posicion y velocidad obtenido mediante la cámara
    /*
     * Devuelve el vector de estados w medidos
     */
    for(int i=0; i < linea.length(); i++){
      if(linea.charAt(i) == ','){
        EstadosWm(t) = linea.substring(r,i).toFloat();     //Guarda los estados
        r = (i+1);
        t++;
      }
    }
    r=0;
    t=0;

    //Multiplica por -Klqr
    Multiply(-Klqr,EstadosWm,EntradasV);   //Devuelve el vector de entradas v

    //Calculo de los estados w
    EstadosWc = MatExponencial(MatExpL,MatExpF,EstadosWm,EntradasV);  //Devuelve el vector de
estados w

    //Parametros de la cinematica inversa
    Px += (EstadosWc(0)-EstadosWm(0))*1000;
    Py += (EstadosWc(2)-EstadosWm(2))*1000;

    //Cinematica inversa
    /*
     * A partir de las posiciones deseadas para el COM del robot
     * se aplica la cinematica inversa para obtener los angulos de
     * las articulaciones del robot
```

```
     * Devuelve valor de articulaciones
     */
    CinematicaInversa();

    //Enviar posicion a motores
    /*
     * Se comunica a los motores los angulos de las articulaciones
     */
    EnviarPosicion();

    delay(10);
    SerialBT.print("K");
  }
}


//Funcion de matriz exponencial para calculo de estados
ArrayMatrix<4> MatExponencial(ArrayMatrix<4,4> lib, ArrayMatrix<4> forz, ArrayMatrix<4> cond,
      ArrayMatrix<2> ent){
  ArrayMatrix<4> libre;
  ArrayMatrix<4> forz2;
  Multiply(lib,cond,libre);
  forz2(0) = forz(0)*ent(0);
  forz2(1) = forz(1)*ent(0);
  forz2(2) = forz(2)*ent(1);
  forz2(3) = forz(3)*ent(1);
  ArrayMatrix<4> total;
  Add(libre,forz2,total);

  return total;
}

//CINEMATICA INVERSA
//Funcion que actualiza las variables articulares
void CinematicaInversa(){
  thd1 = CalcTh1(Py,Pz,1);
  rd1 = ParamPlanar(thd1,Py,Pz,1);
  thd3 = CalcTh3(Px,Phi,rd1);
  thd2 = CalcTh2(Px,Phi,thd3,rd1);
  thd4 = Phi-thd2-thd3-(90*PI/180);
  thi1 = CalcTh1(Py,Pz,0);
  ri1 = ParamPlanar(thi1,Py,Pz,0);
  thi3 = CalcTh3(Px,Phi,ri1);
  thi2 = CalcTh2(Px,Phi,thi3,ri1);
  thi4 = Phi-thi2-thi3-(90*PI/180);
}

//Funcion que calcula Theta1
float CalcTh1(float y,float z,bool pie){
  float a1,b1,c1,alpha1,Cta1,ta1,th1;
  a1 = L1-z;
  if(pie){
    b1 = -pL-y;
    c1 = -L7-L8;
  }
  else{
```

```
    b1 = pL-y;
    c1 = L7+L8;
  }
  alpha1 = atan2(a1,b1);
  Cta1 = c1/sqrt(a1*a1+b1*b1);
  ta1 = atan2(sqrt(1-Cta1*Cta1),Cta1);
  th1= ta1+alpha1;

  return th1;
}

//Funcion que calcula la resultante (parametro del manipulador plano)
float ParamPlanar(float th1, float y, float z,bool pie){
  float Py1, r1;
  if(pie){
    Py1 = y+pL-(L7+L8)*cos(th1);
  }
  else{
    Py1 = y-pL+(L7+L8)*cos(th1);
  }

  if(th1 == 0){
    r1 = z-L1;
  }
  else{
    r1 = abs(Py1/sin(th1));
  }

  return r1;
}

//Funcion que calcula Theta3
float CalcTh3(float x, float ph, float r1){
  float Ct3 = (pow(x-L6*cos(ph),2)+pow(r1-L6*sin(ph),2)-L2*L2-L5*L5)/(2*L2*L5);
  float th3 = atan2(sqrt(1-Ct3*Ct3),Ct3);

  return th3;
}

//Funcion que calcula Theta2
float CalcTh2(float x, float ph, float th3, float r1){
  float var = (L5*sin(th3))/(L2+L5*cos(th3));
  float Ct2 = (x-L6*cos(ph)+var*(r1-L6*sin(ph)))/(var*L5*sin(th3)+L2+L5*cos(th3));
  float th2 = atan2(sqrt(1-Ct2*Ct2),Ct2)-90*PI/180;

  return th2;
}

//Funcion que envia la posición a los motores
void EnviarPosicion(){
  Dynamixel.move(18,-(thd1*deg*scMotor)+512);
  Dynamixel.move(17,-(thi1*deg*scMotor)+512);

  Dynamixel.move(16,(thd2*deg*scMotor)+512);
  Dynamixel.move(15,-(thi2*deg*scMotor)+512);
```

```
  Dynamixel.move(14,-(thd3*deg*scMotor)+512);
  Dynamixel.move(13,(thi3*deg*scMotor)+512);

  Dynamixel.move(12,(thd4*deg*scMotor)+512);
  Dynamixel.move(11,-(thi4*deg*scMotor)+512);

  Dynamixel.move(10,-(thd1*deg*scMotor)+512);
  Dynamixel.move(9,-(thi1*deg*scMotor)+512);
}
```

Anexo 3. Programa de Matlab




```
classdef appBioloidTesis < matlab.apps.AppBase
```

```
    % Properties that correspond to app components
    properties (Access = public)
        UIFigure    matlab.ui.Figure
        axCam       matlab.ui.control.UIAxes
        btnInicio   matlab.ui.control.Button
        btnFin      matlab.ui.control.Button
        axPx        matlab.ui.control.UIAxes
        axPy        matlab.ui.control.UIAxes
        axVx        matlab.ui.control.UIAxes
        axVy        matlab.ui.control.UIAxes
        chbEnviar   matlab.ui.control.CheckBox
    end
```

```
    properties (Access = public)
        cam;
        bt;
        posX;posY;velX;velY;temp;
    end
```

```
    % Callbacks that handle component events
    methods (Access = private)
```

```
        % Code that executes after component creation
        function startupFcn(app)
            app.cam = webcam(2,'Resolution','1280x720');        %Cámara web
            frame = snapshot(app.cam);                        %Variable de captura de imagen
            im = image(app.axCam,zeros(size(frame),'uint8'));   %Variable de imagen
            axis(app.axCam,'image');                           %Muestra captura en recuadro
            preview(app.cam,im);
```

```
    app.bt = Bluetooth('ESP32test', 1);                %Conexión bluetooth
    fopen(app.bt);
end


% Button pushed function: btnInicio
function btnInicioPushed(app, event)
    global parar;
    parar=false;
    escala = 0.06/720;
    contador = 1;
    app.posX=[];app.posY=[];app.velX=[];app.velY=[];app.temp=[];
    while(1)
        drawnow;
        if(parar)
            break;
        end
        %%Envio de datos a microcontrolador
        if(app.chbEnviar.Value)
            fprintf(app.bt,'%.4f,%.4f,%.4f,%.4f,;', ...
                    [app.posX(contador-1),app.posY(contador-1), ...
                    app.velX(contador-1),app.velY(contador-1)]);
            fread(app.bt,1);
        end
        data = snapshot(app.cam);       %Captura de imagen

        %Lineas tomadas de la referencia [25]
        diff_im = imsubtract(data(:,:,1), rgb2gray(data));
        diff_im = medfilt2(diff_im, [3 3]);
        diff_im = imbinarize(diff_im,0.18);
        diff_im = bwareaopen(diff_im,300);
        bw = bwlabel(diff_im, 8);
        stats = regionprops(bw,'Centroid');
        for object = 1:length(stats)
            bc = stats(object).Centroid;
        end

        %Posicion del CoM
        app.posX(contador) = bc(1);
        app.posY(contador) = bc(2);
        %Primer valor del vector de posicones y velocidades guardadas
        if(contador==1)
            centX = app.posX(contador);
            centY = app.posY(contador);
            app.posX(contador) = 0;
            app.posY(contador) = 0;
            app.velX(contador) = 0;
            app.velY(contador) = 0;
            app.temp(contador) = 0;
            tic;        %Corre tiempo de ejecución
        else
            app.temp(contador) = toc;                       %Tiempo transcurrido
            tExp = app.temp(contador)-app.temp(contador-1); %Diferencia de tiempo
            app.posX(contador) = -(app.posX(contador)-centX)*escala;
            app.posY(contador) = (app.posY(contador)-centY)*escala;
            app.velX(contador) = (app.posX(contador)-app.posX(contador-1))/tExp;
            app.velY(contador) = (app.posY(contador)-app.posY(contador-1))/tExp;
        end

        %Graficar los valores de posicion y velocidad
        plot(app.axPx,app.posX,'-b');
```

```
            plot(app.axPy,app.posY,'-b');
            plot(app.axVx,app.velX,'-b');
            plot(app.axVy,app.velY,'-b');
            contador = contador+1;
        end
    end


    % Button pushed function: btnFin
    function btnFinPushed(app, event)
        global parar;
        parar=true;

        %Guardar los datos en variables en el Workspace
        assignin('base','tiempo',app.temp);
        assignin('base','pX',app.posX);
        assignin('base','pY',app.posY);
        assignin('base','vX',app.velX);
        assignin('base','vY',app.velY);
    end


    % Close request function: UIFigure
    function UIFigureCloseRequest(app, event)
        fclose(app.bt);
        delete(app);
    end
end


% Component initialization
methods (Access = private)


    % Create UIFigure and components
    function createComponents(app)


        % Create UIFigure and hide until all components are created
        app.UIFigure = uifigure('Visible', 'off');
        app.UIFigure.Position = [0 450 894 380];
        app.UIFigure.Name = 'Bioloid';
        app.UIFigure.CloseRequestFcn = createCallbackFcn(app, @UIFigureCloseRequest,
          true);


        % Create axCam
        app.axCam = uiaxes(app.UIFigure);
        title(app.axCam, 'Cam')
        xlabel(app.axCam, '')
        ylabel(app.axCam, '')
        app.axCam.Position = [1 180 289 201];


        % Create btnInicio
        app.btnInicio = uibutton(app.UIFigure, 'push');
        app.btnInicio.ButtonPushedFcn = createCallbackFcn(app, @btnInicioPushed,
          true);
        app.btnInicio.Position = [36 147 100 22];
        app.btnInicio.Text = 'Inicio';
```

```
            % Create btnFin
            app.btnFin = uibutton(app.UIFigure, 'push');
            app.btnFin.ButtonPushedFcn = createCallbackFcn(app, @btnFinPushed, true);
            app.btnFin.Interruptible = 'off';
            app.btnFin.Position = [161 147 100 22];
            app.btnFin.Text = 'Fin';


            % Create axPx
            app.axPx = uiaxes(app.UIFigure);
            title(app.axPx, 'Posicion X')
            xlabel(app.axPx, 't')
            ylabel(app.axPx, 'x')
            app.axPx.Position = [289 196 300 185];


            % Create axPy
            app.axPy = uiaxes(app.UIFigure);
            title(app.axPy, 'Posicion Y')
            xlabel(app.axPy, 't')
            ylabel(app.axPy, 'y')
            app.axPy.Position = [289 12 300 185];


            % Create axVx
            app.axVx = uiaxes(app.UIFigure);
            title(app.axVx, 'Velocidad X')
            xlabel(app.axVx, 't')
            ylabel(app.axVx, 'xp')
            app.axVx.XTick = [0 0.2 0.4 0.6 0.8 1];
            app.axVx.Position = [587 196 300 185];


            % Create axVy
            app.axVy = uiaxes(app.UIFigure);
            title(app.axVy, 'Velocidad Y')
            xlabel(app.axVy, 't')
            ylabel(app.axVy, 'yp')
            app.axVy.Position = [587 12 300 185];


            % Create chbEnviar
            app.chbEnviar = uicheckbox(app.UIFigure);
            app.chbEnviar.BusyAction = 'cancel';
            app.chbEnviar.Text = 'Enviar';
            app.chbEnviar.Position = [118 106 56 22];


            % Show the figure after all components are created
            app.UIFigure.Visible = 'on';
        end
    end


    % App creation and deletion
    methods (Access = public)
```

```
        % Construct app
        function app = appBioloidTesis


            % Create UIFigure and components
            createComponents(app)


            % Register the app with App Designer
            registerApp(app, app.UIFigure)


            % Execute the startup function
            runStartupFcn(app, @startupFcn)


            if nargout == 0
                clear app
            end
        end


        % Code that executes before app deletion
        function delete(app)


            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```