# Algoritmo de control para marcha adaptativa en robots hexápodos sobre el framework RoboComp


John Euler Chamorro Fuertes
Jairo José Marín Arciniegas

Director: PhD. Oscar Andrés Vivas Albán

*Universidad del Cauca*
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Electrónica, Instrumentación y Control
Popayán, Enero 2019

# Algoritmo de control para marcha adaptativa en robots hexápodos sobre el framework RoboComp

John Euler Chamorro Fuertes
Jairo José Marín Arciniegas

Trabajo de grado presentado a la Facultad de Ingeniería Electrónica y Telecomunicaciones de la Universidad del Cauca para la obtención del titulo de

Ingenieros en:
Automática Industrial

*Universidad del Cauca*
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Electrónica, Instrumentación y Control
Popayán
Enero 2019

## Agradecimientos

... A nuestras familias, que nos apoyaron incondicionalmente en todo momento, quienes nos animaron a seguir adelante y fueron la mayor motivación para confrontar todos los retos que se nos presentaron.

... A nuestro Director PhD. Andrés Vivas, quien en todo momento estuvo atento de las etapas del proyecto y con su experiencia nos guió en todo el proceso de investigación.

... A nuestros amigos y compañeros de pregrado que siempre se mostraron con una gran sonrisa disponibles a brindar su ayuda, con quienes compartimos grandes momentos e hicieron más entretenido éste periodo.

... A la Universidad del Cauca, nuestra alma máter, por acogernos durante toda nuestra etapa de aprendizaje.

... A Pablo Bustos, que muy pacientemente acompañó, supervisó, dió grandes ideas y dispuso de su tiempo y conocimiento para la realización de éste proyecto.

... A Nicolás, Esteban, Luis, Agustín y Pedro, quienes hicieron que nuestra paso por la Universidad de Extremadura sea mucho más ameno.

... A todo el equipo de RoboLab, que nos permitieron hacer uso se sus instalaciones, recursos y conocimientos para el desarrollo de éste proyecto.

## Resumen

El objetivo de este trabajo es determinar las características de la marcha adaptativa sobre robots hexápodos, en este caso el robot PhantomX. Para ello se diseñó un algoritmo de control sobre el *framework* RoboComp que ofreciera las herramientas necesarias para que la marcha adaptativa se pueda realizar en ambientes con variaciones de nivel controladas y en ambientes irregulares reales. En primera instancia se obtuvo el modelo cinemático inverso de las extremidades del robot para hacer que éstas llegaran a una posición deseada en el espacio y así, posteriormente poder implementar dos modos de marcha: marcha regular y adaptativa. Además, se desarrolló un sistema de estabilización para conseguir que el robot se mantenga en equilibrio al momento de desplazarse por terrenos con alteraciones de nivel. Se hizo necesario la implementación de un procedimiento para generar cambios de dirección. También se realizó la planeación de un método que evite llevar al robot a ambientes en los que salga de su espacio de trabajo seguro. Distintas pruebas sobre varios terrenos, junto con los datos obtenidos de ellas, hicieron evidente los sistemas necesarios para un buen funcionamiento de la marcha adaptativa y las limitaciones a las que el hardware puede llevar ante la implantación de algoritmos complejos.

**Palabras clave: Hexápodo, marcha adaptativa, establización, Robocomp.**

## Abstract

The objective of this work is to determine the characteristics of the adaptive gait on hexapod robots, in this case the PhantomX robot. A control algorithm was designed on the RoboComp framework which offers the necessary tools so that adaptive gait can be carried out in environments with controlled level variations and real irregular environments. In the first instance the inverse kinematic model of the limbs of the robot was obtained to make them reach a desired position in space and subsequently implement two running modes: regular and adaptive gait. Also, a stabilization system was developed so that the robot stays in balance when moving through terrain with level alterations. It became necessary to implement a procedure to generate changes of direction. It was also planning a method that avoids taking the robot to environments where it leaves its safe work space. Different tests on several terrains ans data obtained from them, made evident the systems required for proper operation of adaptive gait and the limitations to implement complex algorithms due to hardware.

**Keywords: Hexapod, adaptive gait, stabilization, Robocomp.**

# Tabla de contenido

# Lista de figuras

# Lista de tablas

# Capítulo 1

# Introducción

La exploración de lugares nocivos, inhóspitos o de difícil acceso, así como terrenos hostiles para el ser humano, motiva el desarrollo de herramientas que garanticen seguridad, confiabilidad y eficiencia [1]. En los últimos años la robótica móvil se ha convertido en foco de distintos estudios como el transporte [2], la exploración planetaria [3], la asistencia en minas [4], el rescate de personas [5] y en el sector industrial [6].

Una parte importante del planeta es inaccesible para muchos tipos de mecanismos con ruedas. Obstáculos tales como rocas grandes, suelo aceitoso, agujeros grandes o suelos empinados son ejemplos en los cuales los vehículos con ruedas a menudo son ineficaces. Dentro de la clasificación de robots móviles existen plataformas con patas los cuales requieren solo una serie de puntos de apoyo discretos a lo largo del camino para la locomoción fuera de ambientes totalmente planos. Esta propiedad permite a los robots con patas atravesar estos terrenos exigentes [7]. Durante muchas décadas, varios estudios sobre robots con patas se han dedicado al desarrollo de estrategias para aplicar la adaptabilidad del terreno a la locomoción eficiente y estable [8].

Los robots móviles caminantes mayormente estudiados a través del tiempo han sido los hexápodos debido al éxito que han tenido para adaptarse a terrenos irregulares, además de ser veloces y con gran estabilidad al momento de moverse por distintos ambientes. Su característica principal es que poseen 6 patas a lo que se denomina como morfología de hexápodo [9], el número de patas implica más puntos de contacto con la superficie durante la locomoción. Esta cualidad les permite mantenerse estables [10].

Sin embargo, el cálculo de diversos grados de libertad, los requisitos de estabilidad al caminar y la viabilidad cinemática hacen que la metodología de control de los robots con patas sea más compleja que la de los robots con ruedas. Los robots con patas requieren coordinar el movimiento del cuerpo y patas para lograr caminar de manera estable y veloz [11]. Además, se debe tener

en cuenta factores importantes como lo son la implementación de dicha estabilización, las limitaciones físicas que el sistema muestra, qué tipo de marcha se puede ejecutar dependiendo del ambiente en el que se encuentre, y se hace importante el uso de un software de robótica que permita programación, control y comunicación estable.

## Objetivos

Para el presente trabajo se proponen los siguientes objetivos.

### Objetivo General

Proponer un algoritmo de control para ejecución de la marcha adaptativa en distintos terrenos para el robot hexápodo PhantomX.

### Objetivos Específicos

1. Diseñar un algoritmo de control en la marcha adaptativa para el robot hexápodo PhantomX, sobre la plataforma RoboComp.
2. Determinar las características de la marcha adaptativa del robot hexápodo PhantomX en cambios de dirección y posición sobre diferentes ambientes (terreno regular con variaciones de nivel y terreno irregular).

Con el fin de contribuir a la línea de investigación de robots móviles enfocados a búsqueda se hace importante la realización de este proyecto, dando solución a las dificultades antes mencionadas. Además, se hace uso del software para robótica RoboComp, el cual es un framework de código abierto que proporciona las herramientas para crear y modificar componentes de software que se comunican a través de interfaces públicas.

El proyecto contó con el apoyo académico y material del laboratorio RoboLab, perteneciente a la Escuela Politécnica de la Universidad de Extremadura, España, en donde se realizó una pasantía de 5 meses, haciendo uso de las instalaciones de la universidad para realizar el diseño del algoritmo y las pruebas con el robot real.

## Estructura de la monografía

La presente monografía se divide en 6 capítulos. El capítulo1 (ya abordado) presenta una introducción conceptual al tema de investigación, la justificación y los objetivos del proyecto. El

capítulo 2 contiene una conceptualización sobre el área de investigación y los avances en esta. El capítulo 3 abarca la información relacionada con la estructura del robot real con el que se trabajó. El capítulo 4 presenta el proceso de desarrollo del algoritmo de control de marcha adaptativa en el robot hexápodo, haciendo uso del software RoboComp. El capítulo 5 presenta el proceso de experimentación llevado a cabo y los resultados obtenidos. Finalmente en el capítulo 6 se presentan las conclusiones del trabajo y se plantean trabajos futuros.

# Capítulo 2

# Conceptualización

Este capítulo busca dar una introducción conceptual a la temática de este proyecto, haciendo una revisión bibliográfica de conceptos y avances relacionados con robots hexápodos y el software de robótica usado.

## 2.1. Robótica móvil

En la década de los 60 se desarrolló en los Estados Unidos el primer robot móvil llamado Shakey [12], el cual fue una plataforma móvil independiente controlada mediante visión artificial. A partir de ese momento, creció exponencialmente la investigación y diseño de robots móviles. Actualmente la robótica móvil es considerada un área de la tecnología avanzada, esto se debe a la versatilidad de los robots para adaptarse a diferentes aplicaciones, debido a que estos tipos de autómatas durante los últimos años son equipados con un alto grado de inteligencia artificial [13].

Un robot móvil es una máquina que cuenta con la capacidad de trasladarse en un ambiente particular de una forma telecontrolada o autónoma con el propósito de realizar una actividad concreta [14]. Otros aspectos importantes en un robot móvil son la cinemática, la percepción y localización en un entorno determinado. Además, evitar situaciones de colisión con objetos estáticos o en movimiento [15].

Una de las áreas de aplicación más interesantes de la robótica móvil consiste en el diseño de robots capaces de operar en condiciones exteriores sobre terrenos no preparados, un claro ejemplo de esto son los robots espaciales, robots militares, robots en operaciones de búsqueda y rescate, etc. Sin embargo, obtener que dicho robot realice un movimiento eficiente y de gran precisión en este tipo de entornos no es sencillo. En la figura 2.1 se observa el robot móvil *Rover Curiosity* en una misión espacial de exploración marciana.

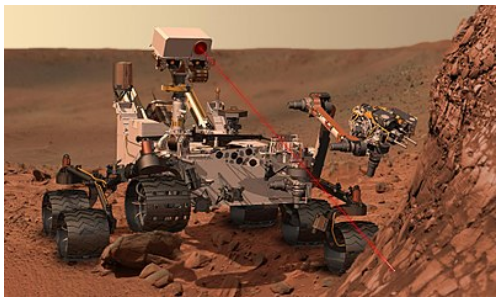


Figura 2.1: Robot móvil *Rover Curiosity* [16].

Dentro de los robots móviles encontramos diferentes configuraciones en función de características como la operación a la cual están destinados, las limitaciones del terreno donde van a operar, clase de actuadores de los que se dispongan o el tipo de alimentación o fuentes de energía utilizadas [17]. En la figura 2.2 se clasifican los robots móviles por el tipo de locomoción utilizado:

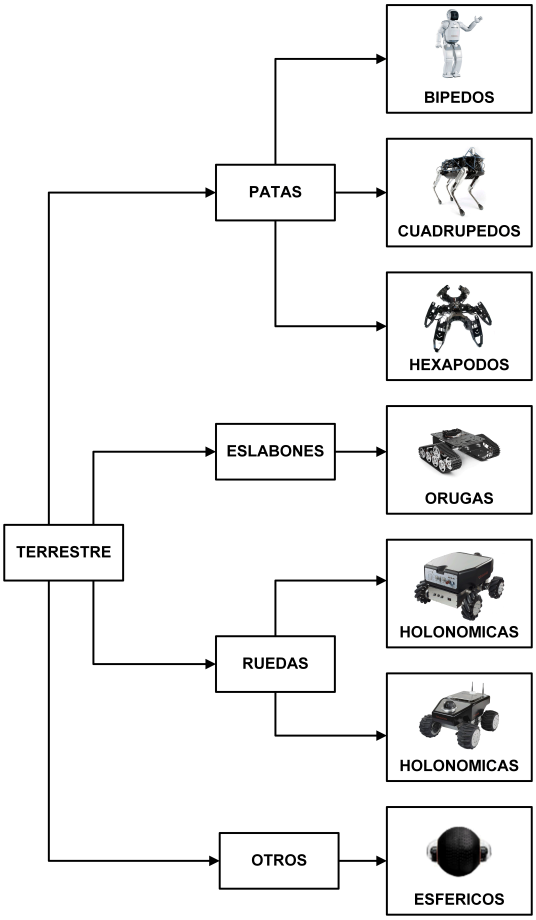



Figura 2.2: Clasificación de los robots terrestres según su sistema de locomoción [18].

### 2.1.1. Robots con ruedas

Son el tipo de robot más común, en este caso las ruedas proporcionan a este tipo de robot una movilidad eficiente en relación con la distancia recorrida y energía consumida, pero queda restringido a superficies planas con pocas irregularidades [19]. El mayor desarrollo de robots móviles esta en este tipo de locomoción debido a las ventajas que presentan las ruedas respecto a las patas y orugas. Dentro de los atributos que más destacan está el que no causa desgaste en la superficie donde se mueven y requieren un número menor de partes, por lo general menos complejas en comparación a los robots de diferente locomoción.

### 2.1.2. Robots oruga o con cadena

Este tipo de robot generalmente lo integran bandas laterales para lograr su desplazamiento, el cual ofrece una mejor tracción respecto a los robots con ruedas, especialmente en terrenos no estructurados como es la arena y la grava. Estos sistemas están diseñados para facilitar algunas tareas como son la inspección y reconocimiento de lugares remotos, donde el uso de estos robots libera al operador de exponerse a situaciones de riesgo [20].

### 2.1.3. Robots con patas o zoomórficos

Se los conoce por imitar la locomoción de diversos seres vivos [21]. Una de sus aplicaciones es la exploración de terrenos debido a que se adaptan al terreno, a pesar de tener una baja eficiencia en su movilidad [22]. Debido a las diferentes morfologías que pueden llegar a tener sus sistemas de locomoción se agrupan en dos categorías: caminadores y no caminadores.

Los robots zoomórficos caminantes son muy numerosos, son objeto de experimentos dado que son capaces de desenvolverse en superficies muy accidentadas, por lo que son principalmente utilizados en aplicaciones de exploración espacial y estudio de volcanes. El grupo de los robots zoomórficos no caminadores son muy poco evolucionados, no se conoce aplicaciones útiles por ello solo se han desarrollado en proyectos de investigación. Se ha encontrado experimentos en Japón basados en un movimiento relativo de rotación mediante segmentos cilíndricos acoplados axialmente entre sí [23].

Los robots con patas han llamado la atención de la robótica dada su capacidad para trabajar en entornos inaccesibles para robots con ruedas y operar por más tiempo que los robots aéreos no tripulados [24]. Un claro ejemplo de ellos son los hexápodos [25], puesto que estos robots pueden ser utilizados en terrenos con poca o sin previa información, y tienen la capacidad de adaptar sus patrones de marcha a diferentes tipos de campos con el fin de mantener una marcha eficiente.

#### 2.1.3.1 Robots hexápodos

Los robots hexápodos pertenecen a una de las ramas de la robótica denominada bioinspiracion, también conocida biomimética. En ella se desarrollan ideas aplicando el conocimiento de la naturaleza, la cual ha estado en continuo progreso durante millones de años, mejorando constantemente sus especies. Esto ha servido de inspiración tanto a los diseños físicos como a los algoritmos de nuevos sistemas robóticos [53].

Desarrollar un robot que esté capacitado para desplazamientos en terrenos abruptos ha llevado a estudiar y analizar movimientos de diferentes animales como son las cucarachas, escarabajos y arañas, obteniendo adaptaciones de forma mecánica y similitudes en sus medios de locomoción[53].

Debido a que un robot puede estar estable al incorporar tres o más patas, un robot hexápodo al desplazarse mediante 6 patas obtiene una gran flexibilidad en la cantidad de movimientos que puede realizar [26].

De forma predeterminada, un robot hexápodo tiene dos modos de distribución estructural de las patas alrededor de su cuerpo, las cuales son:

- **Configuración bilateral** En la configuración bilateral visualizamos una simetría a lo largo del eje longitudinal del robot [27].

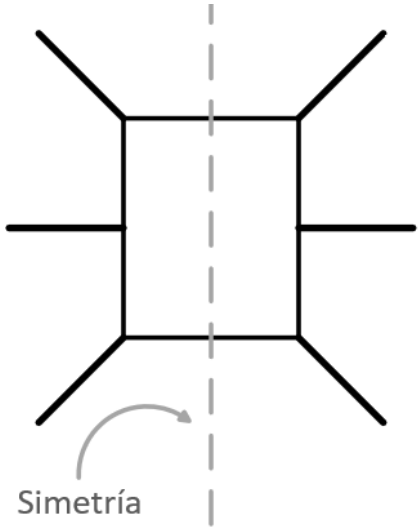



Figura 2.3: Gráfico de configuración bilateral en hexápodos [27].

  Esta configuración presenta una ventaja en la programación de movimientos paralelos a su eje de simetría debido a la configuración física del robot, presentando un inconveniente en la limitación en movimientos en otras direcciones como son los giros.

- **Configuración radial** La distribución radial no presenta limitaciones en los movimientos, dado que en cualquier dirección es exactamente igual a su opuesta, es decir que son robots holonómicos [27].

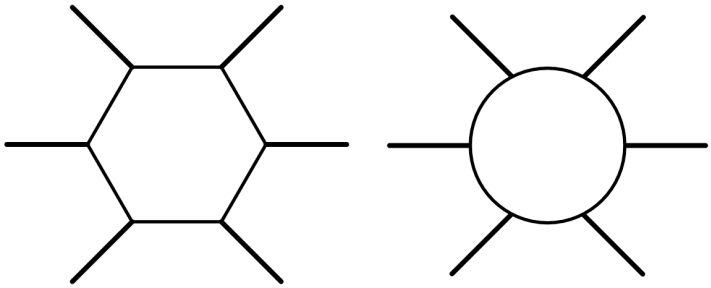


Figura 2.4: Gráfico de configuración radial en hexápodos [27].

Otro aspecto importante en un robot hexápodo es la estructura de las patas debido a los grados de libertad:

1. **Un grado de libertad:** Un hexápodo con extremidades de un grado de libertad tiene una configuración constantemente utilizada, consiste en que las patas centrales son capaces de hacer subir o bajar el robot y sus patas extremas son las que cumplen la labor de hacer avanzar o retroceder [28].

   El diseño es especifico para el terreno donde el robot se desplazará, en la figura 2.5 se presentan diferentes diseños utilizadas en el robot RHex.

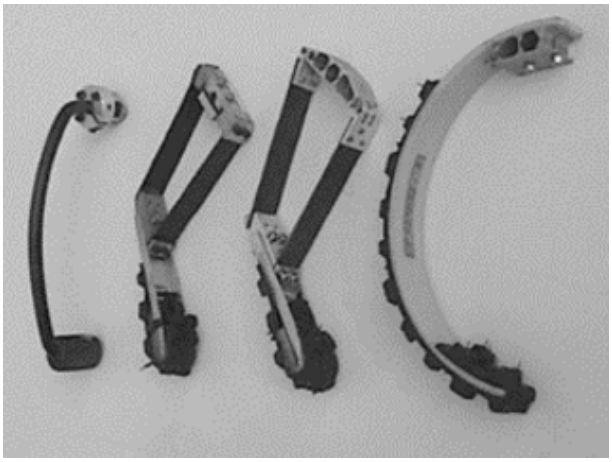


Figura 2.5: Diseños de patas usadas en los modelos RHex [28].

2. **Dos grados de libertad:** Los hexápodos que cuentan con extremidades con dos grados de libertad son más complejos, ya que conllevan a una programación con mayor dificultad, obteniéndose también más movimientos [27].

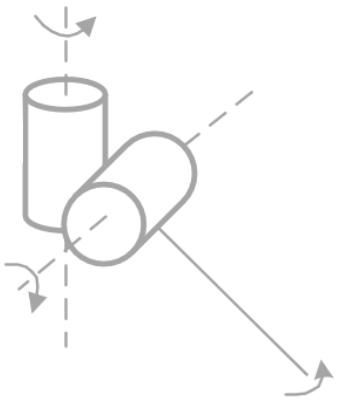


Figura 2.6: Configuración de 2 grados de libertad [27].

3. **Tres grados de libertad:** Los hexápodos con una configuración de 3 grados de libertad se presentan debido a la inspiración morfológica de animales [27].

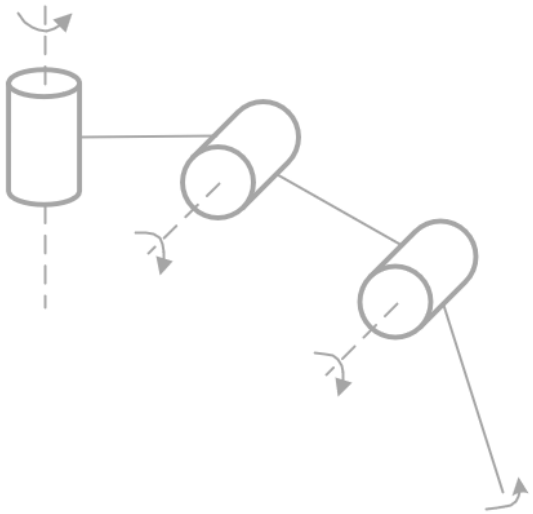


Figura 2.7: Configuración de 3 grados de libertad [27].

Los robots hexápodos cuentan con una gran estabilidad y adaptabilidad, sus seis patas les permiten movilizarse y adaptar una postura estable en terrenos donde se presentan obstáculos y desniveles. Éstas cualidades son de gran interés para los robots USAR (Robots de búsqueda y rescate), donde un proyecto basado en hexápodos es el robot RHex, un robot que se caracteriza por sus patas en forma de C, las cuales son flexibles con un solo grado de libertad. Este robot se desarrolló en diversas universidades de Estados Unidos financiado por DARPA (Agencia de proyectos de investigación avanzados de defensa) para tareas de búsqueda y rescate [29]. Este robot hexápodo es uno de los más importantes debido a su portabilidad y versatilidad a la hora de enfrentarse a diferentes terrenos con una gran velocidad, también tiene la capacidad de realizar complejas acciones como son dar saltos, subir escaleras e incluso andar de forma bípeda [30].

En la tabla 2.1 se presenta una comparación de diferentes robots hexápodos usados en tareas de búsqueda y rescate respecto a la velocidad alcanzada dependiendo de la longitud de dicho robot.

| Robot | Longitud (m) | Peso (Kg) | Velocidad Max (m/s) | Velocidad/Longitud |
|---|---|---|---|---|
| Case Western Robot II | 0.5 | 1 | 0.083 | 0.16 |
| Dante II | 3 | 770 | 0.017 | 0.006 |
| Atilla | 0.36 | 2.5 | 0.03 | 0.083 |
| Genghis | 0.39 | 1.8 | 0.038 | 0.097 |
| Adaptive Suspension Vehicle | 5 | 3200 | 1.1 | 0.22 |
| Boadicea | 0.5 | 4.9 | 0.11 | 0.22 |
| Sprawlita | 0.17 | 0.27 | 0.42 | 2.5 |
| RHex | 0.53 | 7 | 0.55 | 1.04 |

Tabla 2.1: Tabla comparativa de rendimiento entre diferentes robots hexápodos [31].

A partir de 2001 diversos grupos de investigación han desarrollado ideas basados en el modelo original RHex, a continuación, están los principales modelos de RHex desarrollados hasta la fecha:

- **EduBot:** Robot con fines educativos y de investigación académica de la Universidad de Pensilvania, Estados Unidos, de manera que sea fácil mejorar alguna de sus funciones o repara algo en él.
- **X-RHex:** Diseñado con mayor carga y autonomía que el RHex, además incluye una arquitectura modular de carga lo cual apoya las diversas necesidades de investigación. Desarrollado en la Universidad de Pensilvania, Estados Unidos.
- **XRL:** Es una versión más ligera del X-RHex, se podría denominar como la versión actualizada, desarollada por la Universidad de Pensilvania, Estados Unidos, la cual está dotada de mayor agilidad y reparte la carga que puede llevar en sus compartimientos.
- **Desert RHex:** Desarrollado en el año 2009 para evaluar el desempeño en terrenos desérticos montañosos. También desarrollado por la Universidad de Pensilvania, Estados Unidos.
- **SandBot:** Robot desarrollado por Georgia Tech, Estados Unidos, con el fin de estudiar el comportamiento de la locomoción en arena.
- **Aqua:** Un robot diseñado para moverse por debajo del agua, aunque pueden nadar por la superficie y caminar en el fondo del mar. Proyecto desarrollado por las Universidades de McGill, York y Dalhousie, Canadá.
- **Rugged RHex:** Es la versión comercial del RHex, ha sido desarrollado por Boston Dynamics, Estados Unidos. Se trata de un robot orientado a las operaciones militares, capas de caminar por pendientes de hasta 60 %, subir escaleras y moverse a una velocidad de 0.9 m/s.

La figura 2.8 muestra los modelos de cada uno de los robots mencionados, siendo estos adaptaciones y/o actualizaciones del RHex.

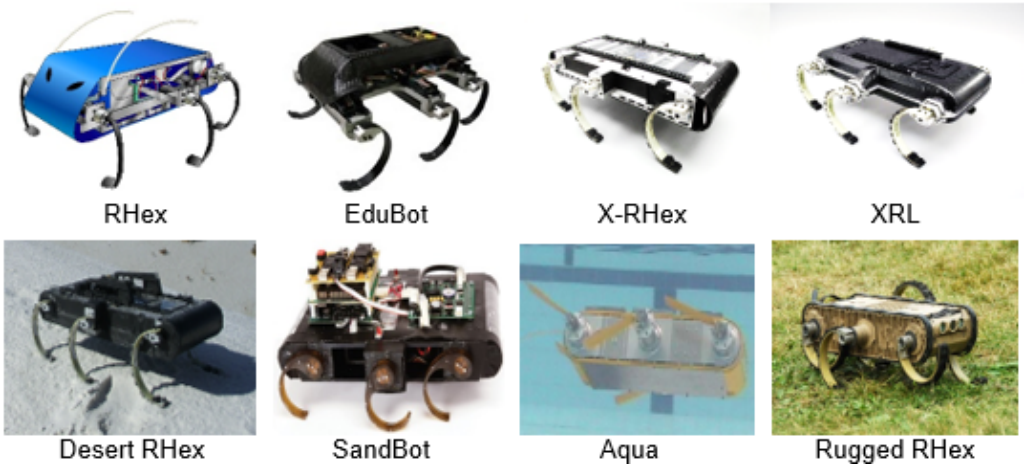



Figura 2.8: Versiones robot RHex [32].

## 2.2. Avances en el área de investigación

La robótica móvil incluye cualquier sistema robótico destinado a realizar desplazamientos en su entorno [10]. Durante los últimos años el diseño del sistema de locomoción de los robots con patas se ha derivado del estudio de sistemas biológicos, especialmente de zoomorfos [21]. Estos han llamado la atención de la robótica dada su capacidad para trabajar en entornos inaccesibles para robots con ruedas y operar por más tiempo que los robots aéreos no tripulados [24]. Un claro ejemplo de ellos son los hexápodos [25]. Puesto que estos robots pueden ser utilizados en terrenos con poca o sin previa información, y tienen la capacidad de adaptar sus patrones de marcha a diferentes tipos de campos con el fin de mantener una marcha eficiente.

Para la elaboración de control sobre robots móviles es necesario un software que lo haga posible. El software para robótica Robocomp fue creado en 2005 por el Laboratorio de Robótica y Visión Artificial de la Universidad de Extremadura, España [33]. Robocomp es un *framework* de fuente abierta y basado en componentes, lo cual proporciona facilidad de uso y un desarrollo rápido en la programación de robots [34, 35]. Robocomp se basa en el middleware Ice (*Internet Comunication Engine*) orientado a objetos [36], el cual asiste a una aplicación para interactuar o comunicarse con otras aplicaciones, o paquetes de programas, redes, hardware o sistemas operativos. Tiene un amplio soporte de idioma y plataforma, permitiendo a los desarrolladores diseñar aplicaciones distribuidas con un alto grado de tolerancia a fallos y seguridad, utilizando excepciones remotas y conexiones seguras. Robocomp admite *"Java Remote Method Invocation"* [37], el cual es un mecanismo para invocar un método de manera remota en modo síncrono y asíncrono, y comunicaciones de publicación y suscripción, para lo cual hace uso de un intermediario centralizado llamado IceStorm [38].

En el Centro de Investigación y Estudios Avanzados del Instituto Politécnico Nacional de México,

se presenta un artículo en el año 2017 donde se describe un algoritmo de un planificador cinemático para generar un patrón de marcha de un robot hexápodo llamado K3P [39], el cual tiene como principal novedad que el patrón de marcha se adapta en tiempo real para que el centro de masa siga un recorrido arbitrario, limitando así el deslizamiento entre el punto de contacto de cada extremidad con la superficie que se apoya. Entonces, se puede notar que existen diferencias con respecto a estrategias heurísticas las cuales permiten realizar "marcha a ciegas", puesto que los reflejos y la secuencia lógica le permiten desplazarse en un ambiente desconocido gracias a que cuenta con sensores táctiles [40] o por medio de la medición de la corriente consumida por los actuadores de sus articulaciones [41] que le permiten detectar obstáculos o un cambio en el nivel de terreno.

La estrategia del K3P a diferencia de la heurística, modifica la línea de secuencia de movimientos para la marcha e incorpora comportamientos de tipo reactivo, operando en un lazo abierto, obteniendo como resultado de simulación una marcha del robot hexápodo con una disposición radial estáticamente estable, dejando así como un trabajo futuro la ejecución de este algoritmo al robot real.

En Suecia, realizan pruebas de clasificación de terreno mediante acústica basados en los estudios de Best y Ozkul [42, 43]. La clasificación de terreno es importante para los humanos dado que al escuchar el sonido de los pasos de una forma intuitiva se obtiene información sobre el terreno por donde se camina incluso en la oscuridad, éste método es utilizado para que los robots puedan adaptar patrones de marcha a diferentes tipos de terreno. Las pruebas se realizaron con un robot PhantomX con un tipo de marcha de trípode alterno y un micrófono omnidireccional. Los datos recolectados fueron recolectados en 7 diferentes tipos de terrenos, se presentó un método de eliminación de ruido que mejoro la sensibilidad del reconocimiento de terreno practicamente completo, obteniendo un sistema efectivo para que el robot pueda realizar una navegación robusta a través de un terreno desconocido [44].

Acuña, en México [45], presenta un proyecto con un diseño e implementación de un algoritmo para la locomoción de un robot hexápodo sobre una superficie vertical. Como primer objetivo se diseñó un hexápodo morfológicamente similar a una cucaracha [46], pero se optó por un sistema de sujeción tipo ventosa compuesto por electroválvulas y generadores de vacío ideal para superficies planas y totalmente lisas, como módulo de control de los servomotores se utilizó una tarjeta desarrollada por Pololu, la cual se comunica vía serial desde el computador con implementación de una secuencia de locomoción llamada trípode alterno.

Los robots con patas con misiones donde se presenten terrenos accidentados deberían tener suficiente autonomía para aprovechar sus capacidades de locomoción, para ello es necesario que el robot construya una representación espacial del entorno y se localice dentro de él, este problema en la robótica es conocido como SLAM (*"Simultaneous Localization and Mapping"*) [47].

Las técnicas para los robots con patas, por lo que en la Universidad Técnica de Republica Checa evalúan un método *"Stereo Parallel Tracking and Mapping"* [48] al cual se le realiza la integración

de un “Filtro de Kalman extendido” utilizando mediciones de velocidad angular y aceleración lineal proporcionados por una IMU (Unidad de medición inercial). El sistema se validó en un ambiente interior diseñado con obstáculos específicos como rampas, escaleras e irregularidades en el terreno. Además, se comparó el rendimiento de dicho sistema con el método RGB-D (*“Red Green Blue Depth”*) SLAM más actual [49]. Los resultados obtenidos demostraron que el sistema es adecuado para la localización del robot hexápodo y supera al sistema RGB-D utilizado en términos de precisión y velocidad, con la ventaja adicional de que posee la capacidad de trabajar en entornos al aire libre [24].

La exploración espacial requiere de sistemas robóticos complejos, capaces de una operación autónoma, así como la capacidad de enfrentarse a entornos hostiles y obstáculos. Para probar dichas tecnologías se realizan competencias como la copa SpaceBot [50] en la cual el equipo LAUROPE (Robot para la Exploración Planetaria) del Centro de Investigación de Tecnología de la Información FZI, Alemania, participó con un robot hexápodo LAURON V, basado en conceptos de movilidad desarrollados en el robot *“SpaceClimber”* de seis patas [51]. El robot está equipado con un escáner láser 3D para localización y mapeo, el cual ayuda al robot con sus características de marcha y comportamiento a los reflejos rápidos asegurando que el robot mantenga el contacto con el suelo, incluso cuando el suelo es cambiante [52].

El robot hexápodo PhantomX-AX tiene la capacidad de desplazarse a una gran velocidad. Está equipado con una placa de microprocesador Arduino y un software llamado Phoenix, el cual incluye un algoritmo genérico para modo de marcha en lazo abierto. La gran fluidez de movimiento del robot se debe a que cuenta con un total de 3 servomotores inteligentes Dynamixel AX-12 en cada pata, proporcionando 3 grados de libertad que nos permiten obtener un desplazamiento totalmente fluido [53, 42] y así lograr distintas aplicaciones.

Para el desarrollo de un algoritmo de ejecución de movimientos en un robot hexápodo, se han propuesto diversas formas de locomoción con su principal objetivo de mantener una estabilidad. En el Instituto Tecnológico de Querétaro, México, se presenta un algoritmo de locomoción libre mediante lógica difusa, en el cual el algoritmo determina a qué pata le es posible moverse sin alterar su centro de gravedad. Obtuvo un buen desempeño en las trayectorias y condiciones en las cuales se probaron, pero con una desventaja en el tiempo de desplazamiento, el cual es mayor comparado con el algoritmo de locomoción fija [54].

En la mayoría de los trabajos realizados con robots hexápodos se utiliza un modo de marcha llamado trípode alterno o triángulo de apoyo [55, 56]. En el Instituto Tecnológico de la Laguna Torreón, México, implementan un sistema de control electrónico multivariable para realizar la locomoción de un robot hexápodo utilizando el modo de marcha mencionado anteriormente, logrando que el robot avance de forma frontal y lateral. Dicha locomoción se controló usando el modelo cinemático inverso individual de cada pierna almacenado en el microcontrolador del robot, el cual permite en tiempo real calcular los valores que requiere cada actuador, facilitando el cambio de programación de las trayectorias [57].

# Capítulo 3

# Estructura robot PhantomX

El robot hexápodo sobre el que se llevó a cabo éste proyecto es el *PhantomX AX Hexapod Mark II* [58] indicado en la figura 3.1

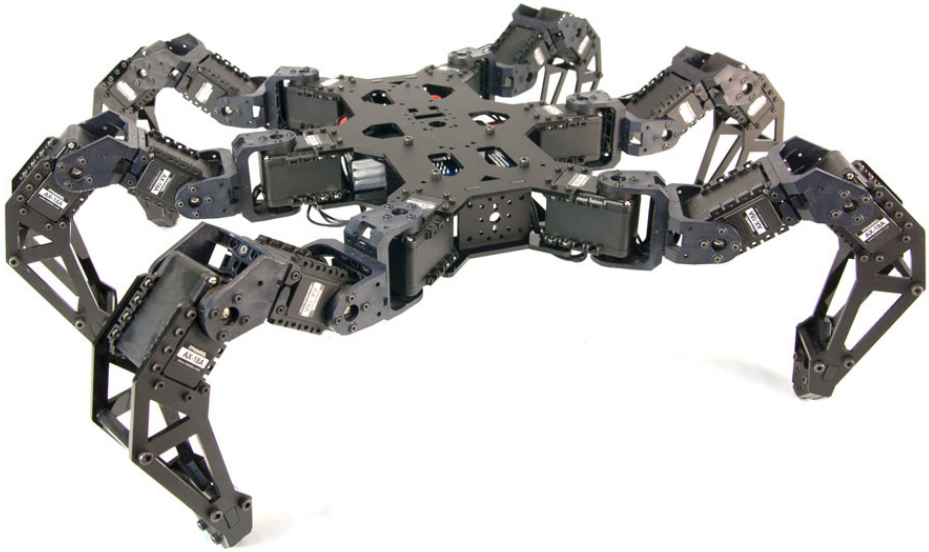


Figura 3.1: *PhantomX AX Hexapod Mark II* comercializado por Trossen Robotics [58].

Está desarrollado por Interbotix Labs y es la segunda versión de su robot hexápodo PhantomX. Utiliza tres servos por tramo para un total de 18 servos con los cuales el PhantomX proporciona 18 grados de libertad. La unidad de procesamiento es una placa basada en Arduino llamada ArbotiX-M que utiliza un procesador ATmega644p de 16MHz. El ArbotiX-M se comunica con los 18 Servos Dynamixel AX-12A a través de un canal de comunicación serial UART dúplex medio. Esta comunicación se realiza a 1 Mbit por segundo.

Los 18 servomotores Dynamixel AX-12A, mostrados en la figura 3.2, tienen la capacidad de ras-

trear su velocidad, temperatura, posición del eje, voltaje y carga. Además, el algoritmo de control utilizado para mantener la posición del eje en el actuador AX-12 se puede ajustar individualmente para cada servo, lo que le permite controlar la velocidad y la fuerza de la respuesta del motor. Toda la gestión del sensor y el control de posición son manejados por el microcontrolador incorporado del servo. Este enfoque distribuido deja su controlador principal libre para realizar otras funciones. En la figura 3.2 se muestra el modelo de los servomotores instalados en el robot.


Figura 3.2: Servo Dynamixel AX-12A [59].

Además, las especificaciones de los actuadores se muestran en la tabla 3.1.

| **MODELO** | **AX-12A** |
|---|---|
| Par de torsión | 1.5 N.m-1 |
| Velocidad (RPM) | 59 |
| Valor nominal de operación | 12 V |
| Corriente | 1.5 A |
| Dimensiones | 32x50x40 mm |
| Peso | 54.6 g |
| Resolución | 0.29 grados |
| Ángulo de funcionamiento | 300 grados |
| Reducción de engranajes | 254:1 |
| Sensor de posición | Potenciómetro |
| Protocolo de comunicación | TTL |
| Velocidad de comunicación | 1 Mbps |

Tabla 3.1: Especificaciones Servo Dynamixel AX-12A [59].

## 3.1. Sensores

Junto con otros investigadores de la Universidad de Extremadura, se llevaron a cabo modificaciones del robot original para que el sistema tenga realimentación mediante sensores y así poder realizar un control de variables. Se instalaron dos tipos de sensores: Sensores de presión, uno de ellos al final de cada pata; y una unidad de medición inercial (IMU) en el centro del robot.

### 3.1.1. Sensores de presión

El sensor de presión, a partir de un medidor de fuerza que se convierte en cambios de valor de resistencia, obtiene información de la presión. Cuanto mayor es la presión, menor es la resistencia. Permite medir la presión de 0-10 kg. Se puede utilizar para la detección de presión al final de una extremidad de un robot hexápodo, experimentos biológicos de medición de la fuerza de mordedura de un mamífero y una amplia gama de aplicaciones. Sin embargo, debido a que la detección de la presión no es muy precisa, no se recomienda utilizar cuando se requiere medir con precisión.

En la figura 3.3 se muestra el sensor elegido y la instalación sobre una de los finales de las extremidades del robot.

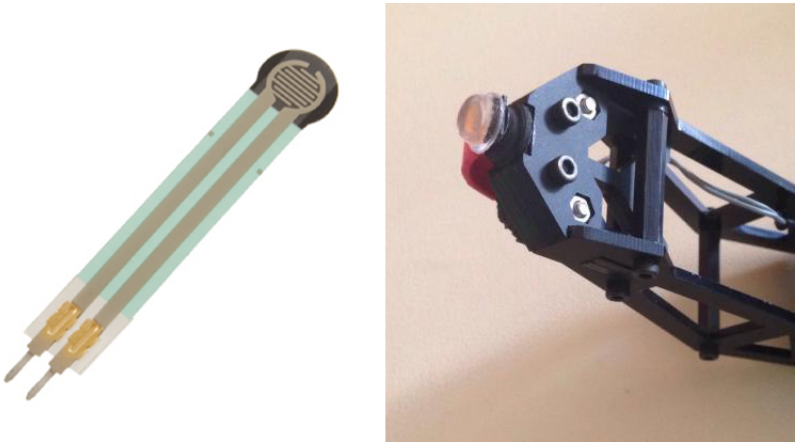


Figura 3.3: *Sensor de presión y su instalación en la extremidad. Fuente propia.*

### 3.1.2. Unidad de medición inercial (IMU)

Este tipo de sensores normalmente tienen una complicación en el tratamiento de sus datos. El *Adafruit BNO055 Absolute Orientation Sensor* convierte los datos del sensor de un acelerómetro, giroscopio y magnetómetro en orientación espacial 3D real. Además, lo hace con un procesador basado en ARM Cortex-M0 de alta velocidad para obtener todos los datos del sensor, abstraer la

fusión del sensor y los requisitos de tiempo real de distancia para así entregar datos que se pueden usar en cuaterniones, ángulos de Euler o vectores. Las salidas que entrega son: Orientación absoluta (en cuaternión o vectores de Euler), velocidad angular, aceleración, campo magnético, aceleración lineal, gravedad y temperatura. En la figura 3.4 se indica la unidad de medición inercial escogida para éste proyecto.

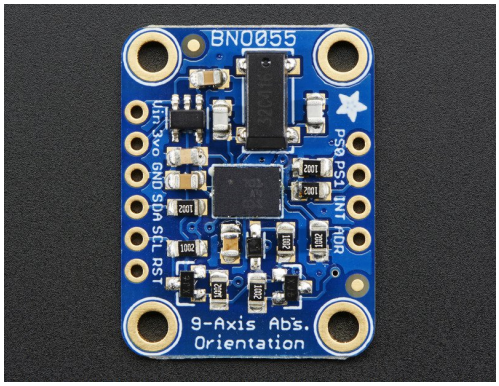



Figura 3.4: *BNO055 Absolute Orientation Sensor [60].*

## 3.2. Unidad central de control

Para el proyecto se hace importante tener una gran velocidad de procesamiento, por lo cual es necesario elegir una unidad de control de mayor procesamiento que la placa Arduino. En esta sección se describen dos computadoras de una sola placa con una capacidad superior a la presente en el robot PhantomX. La primera es la tercera generación de Raspberry Pi. La segunda computadora es la Odroid-XU4, la cual es la placa de mayor rendimiento de Odroid. En la tabla 3.2 se presentan especificaciones técnicas de ambas computadoras.

| | **COMPUTADORA** | |
|---|---|---|
| | **Raspberry Pi 3** | **Odroid-XU4** |
| **Arquitectura** | ARM | ARM |
| **CPU** | 4x 1.2Ghz | (4+4)x 2Ghz |
| **RAM (Gb)** | 1 | 2 |
| **USB** | 4 - 2.0 | 1 - 2.0, 2 - 3.0 |
| **Dimensiones(mm)** | 85x56 | 83x58 |

Tabla 3.2: Comparación entre computadoras [61, 62].

Del hardware presentado, el Odroid-XU4 tiene una frecuencia de CPU mayor, la cual nos brinda la capacidad de ejecutar múltiples procesos en paralelo; para este trabajo en específico se eligió una unidad con ventilador de enfriamiento activo, acompañado de 32 GB de memoria flash sobre la cual se instaló el sistema operativo Ubuntu. En la figura 3.5 se indica la unidad central de control elegida.

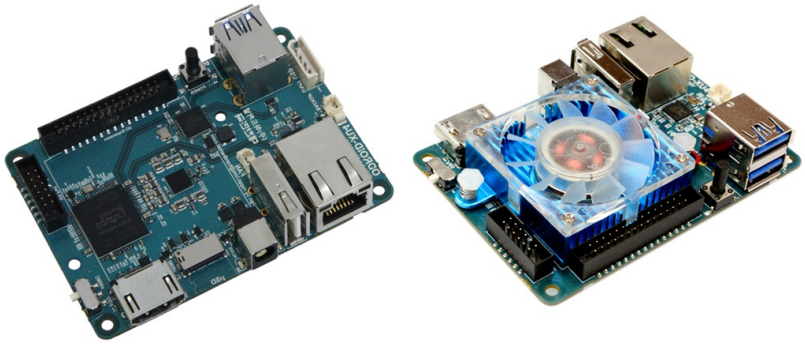


Figura 3.5: Unidad central de procesamiento Odroid XU4 con unidad de enfriamiento [63].

## 3.3. Conexiones hardware

La transmisión entre la computadora externa y la computadora se puede realizar mediante una red cableada (*Ethernet*) o una comunicación inalámbrica. Dado que la comunicación inalámbrica no es tan confiable ni tan rápida como la cableada, se escoge una conexión cableada.

Los sensores resistivos de presión están conectados a un Arduino UNO, el cual junto al sensor de unidad inercial IMU se comunican al Odroid a través de USB.

En la figura 3.6 se muestra cómo se debe llevar a cabo la conexión entre el sensor resistivo de presión y la placa Arduino.

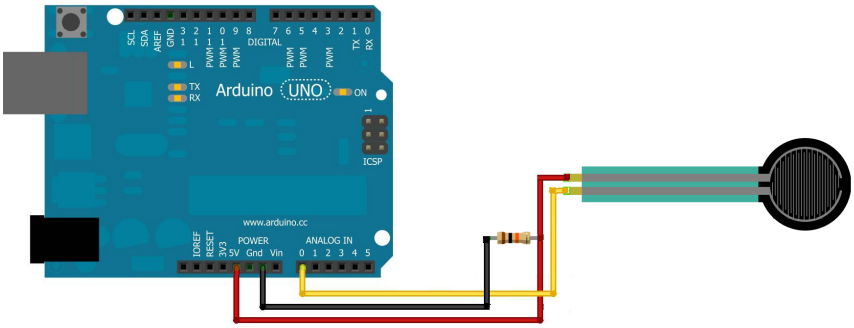



Figura 3.6: Representación de la conexión sensor-Arduino [64].

Los servomotores Dynamixel AX-12A son actuadores inteligentes controlados mediante el protocolo TTL que utiliza la comunicación en serie del receptor asíncrono y transmisor asíncrono (UART) semidúplex universal. Los actuadores están conectados a un bus serie y cada actuador puede direccionarse por su ID única, la cual se logra evidenciar en la figura 3.7.

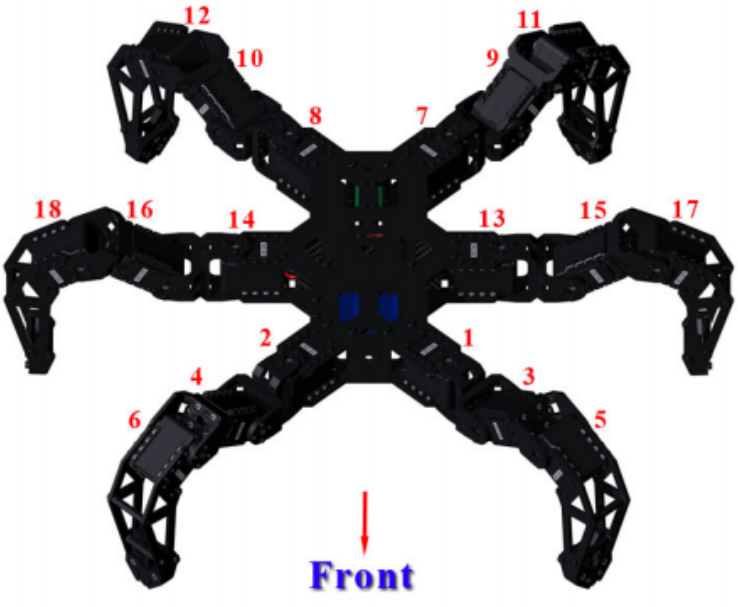



Figura 3.7: Ubicación e identificación de los motores en el robot [65].

La comunicación entre los actuadores y la unidad de control se puede realizar de diversas maneras dependiendo de la unidad de control escogida. En este caso se realiza una comunicación mediante USB, en la cual la unidad de control envía un comando a través de un cable USB, que se transfiere al TTL mediante el adaptador USB2 DYNAMIXEL, como se indica en la figura 3.8.

Figura 3.8: Adaptador USB2 DYNAMIXEL [66].

La fuente de alimentación se conecta mediante un par de interruptores a ambos concentradores de potencia AX/MX (dentro del convertidor de UART a USB) y unidad de control central Odroid.

Finalmente, la conexión física de todo el sistema se muestra en la figura 3.9.

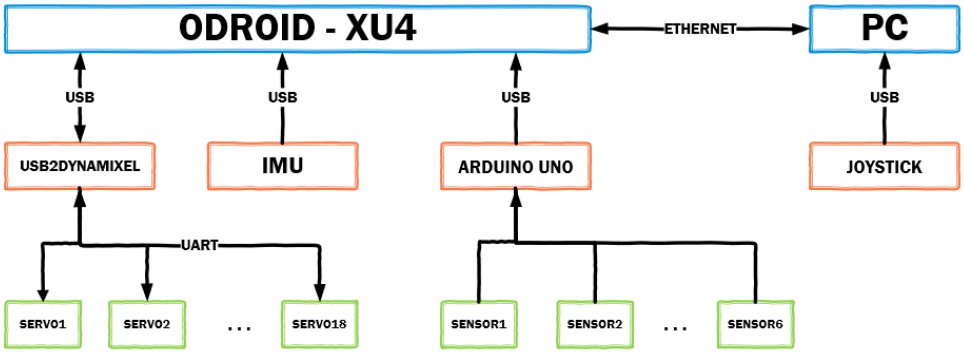



Figura 3.9: Conexiones del sistema completo. *Fuente propia*.

## 3.4. Configuración final

Contemplando el sistema genérico y las mejoras que se le aplicaron, el robot hexápodo queda finalmente como se muestra en la figura 3.10.

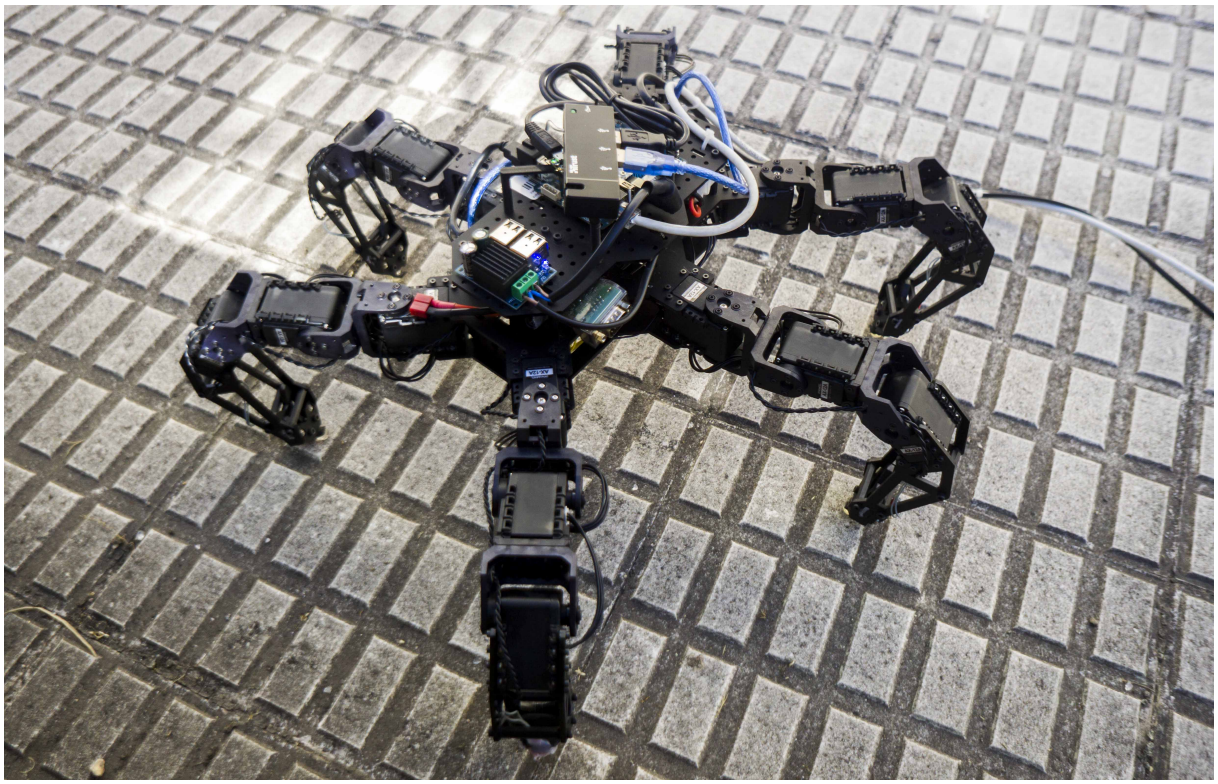


Figura 3.10: Estructura final del robot hexápodo PhantomX. *Fuente propia*.

# Capítulo 4

# Diseño del algoritmo de control para marcha adaptativa

En el presente capítulo se desglosan todos los aspectos relevantes en el proceso de desarrollo del algoritmo de control para ejecución de la marcha adaptativa en el robot. Se muestran los lineamientos para ejecución de los dos tipos de marcha. Se explica cada uno de los componentes realizados para lograr el funcionamiento del algoritmo principal y se explica el algoritmo principal.

## 4.1. Configuraciones de las extremidades

Los robots con múltiples patas en su mayoría son más complejos que los robots con ruedas u orugas. La complejidad que presentan las múltiples patas mejora las capacidades de movilidad del robot al atravesar por un terreno abrupto, pero esto requiere un control más avanzado [57].

La manera de controlar un hexápodo es utilizando un patrón de movimiento definido como es la marcha. En la marcha se distinguen dos fases de locomoción para cada pata, mientras una pata está en la fase de oscilación “giro” donde se mueve a un nuevo punto de apoyo, otra pata apoya el cuerpo del robot en la fase de apoyo “postura”. Durante el desplazamiento del robot las patas se alternan entre las dos fases [57].

Los patrones de movimiento se proporcionan de acuerdo al número de patas que soportan y mantienen estable el robot (esto implica la cantidad de patas en movimiento). De acuerdo al número de patas que se mueven simultáneamente, se definen los modos de marcha n-pod. Los diferentes modos n-pod hacen que varíe la velocidad y estabilidad del hexápodo. Para mantener estable el cuerpo de un robot hexápodo se requieren al menos 3 patas, con esta restricción se obtienen 3 modos de marcha [17]: trípode, tetrápoda y pentápodo.

- **Trípode**

  En el modo de marcha por trípode, las patas se dividen en dos grupos de tres. En la marcha, un trío alterna al otro. Mientras tres patas se mueven y tres patas sostienen el cuerpo del robot. El grupo más estable consiste en un trípode alterno conformado por la pata delantera y trasera de un lado del hexápodo y la pata del medio del lado opuesto [67]. El ciclo de marcha se realiza en dos pasos como se muestra en la figura 4.1.

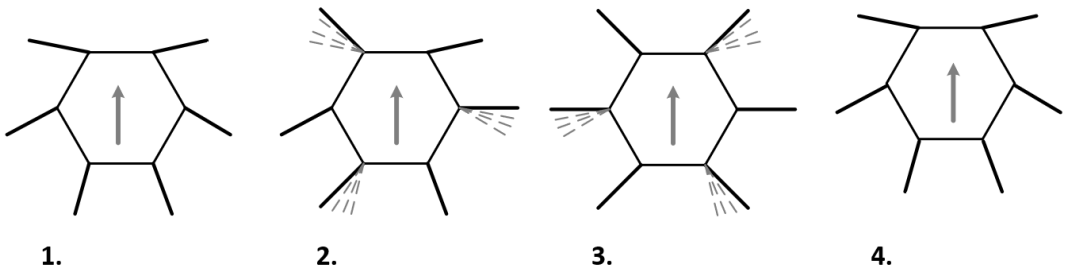



Figura 4.1: Tipo de marcha en trípode [67].

- **Tetrápoda**

  En la marcha tetrápoda se definen tres pares de patas. Mientras dos pares soportan el robot un par se mueve. Se pueden obtener diversas combinaciones de patas, pero las combinaciones más estables requieren que cada par este conformado por patas de diferentes lados del robot con la variación en la posición delantera-trasera [67]. Dicha configuración se muestra en la figura 4.2.

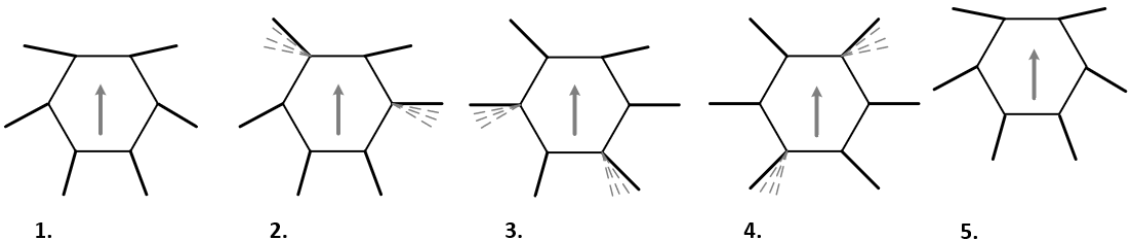



Figura 4.2: Tipo de marcha en configuración tetrápoda [67].

- **Pentápoda**

  El modo de marcha pentápodo se realiza con cada pata del hexápodo, esto indica que se moverá solo una pata mientras que las cinco patas restantes sostienen el robot [67]. El orden de las patas en la marcha no afecta la estabilidad del robot, pero el método más común es utilizar el orden en sentido de las agujas del reloj o en sentido contrario en el ciclo de marcha como se muestra en la figura 4.3.

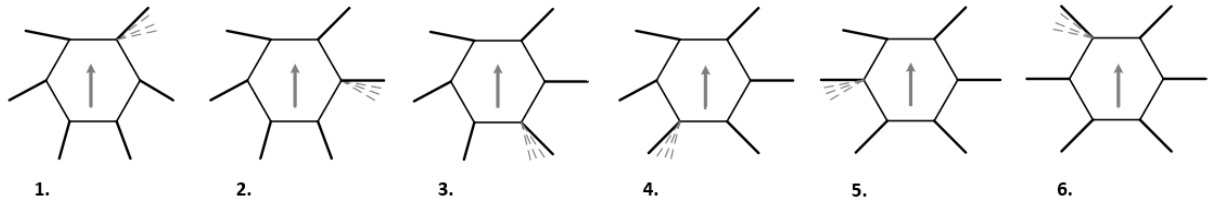



Figura 4.3: Tipo de marcha en configuración pentápoda [67].

Los modos de marcha indicados en las figuras 4.1, 4.2 y 4.3 son movimientos periódicos. También existen marchas no periódicas como lo son la marcha libre o marcha en onda. En este tipo de marcha el robot hace uso de otros factores como la percepción externa mediante sensores para elegir el orden de las piernas para movilizarse en terrenos con dificultades.

El modo elegido para este proyecto fue la configuración de marcha en trípode. Este modo de marcha es el que menos esfuerzo genera a los motores ya que el esfuerzo se reparte en tres unidades. Además, es también el que menos resistencia exige a las patas ya que soporta el peso del robot en 3 patas. También permite una mayor estabilidad, ya que el centro de gravedad cae dentro del triángulo formado por las tres patas que están apoyadas. No genera una fuerza el deslizamiento cuando tres patas están moviendo al robot, las otras tres están en el aire. Sin embargo, este tipo de marcha limita la complejidad de los ambientes en los cuales puede moverse el robot y reduce su velocidad con respecto a otro tipo de configuraciones.

## 4.2. Modelo cinemático inverso

Para la marcha del hexápodo a través de trayectorias y la posterior implementación de la estabilización se hace necesario un sistema mediante el cual los finales de las extremidades del hexápodo lleguen a un punto deseado y así tener el efecto requerido sobre el cuerpo del robot. Para dicho objetivo lo ideal es obtener el modelo cinemático inverso. El cuerpo se mueve simultáneamente con las extremidades, por lo que una marcha sincronizada se aplica en cada uno de los actuadores [68].

Básicamente, el análisis cinemático de un robot comprende el estudio de su movimiento con respecto a un sistema de referencia, de aquí que esto implica:

- **Cinemática directa:** A partir de valores conocidos de las articulaciones y parámetros geométricos de los eslabones del robot, se calcula la posición y orientación del efector final con respecto a un sistema de referencia. Se controla al robot de manera indirecta, a partir del movimiento de sus articulaciones, siendo muy difícil intuir el movimiento del efector final para realizar un determinado propósito [69]. Siendo:

$$x = f(q) \tag{4.1}$$

en donde, *x* es el vector de posición y orientación del efector final, y *q* el vector de las articulaciones (grados de libertad) de la cadena cinemática.

- **Cinemática inversa:** Consiste en determinar la configuración que debe adoptar un robot para obtener una posición y orientación determinadas del efector final [70]. Mediante su resolución se puede controlar al efector final de una manera explícita, es decir, directamente a través de la posición y orientación. Entonces:

$$q = f^{-}1(x) \tag{4.2}$$

$$q_k = f_k(x, y, z, \alpha, \beta, \gamma) \tag{4.3}$$

en donde, *k* representa el número de grados de libertad de la cadena cinemática. Para definir la posición se hace uso de las coordenadas cartesianas *(x,y,z)* y los ángulos $(\alpha, \beta, \gamma)$ para definir la rotación.

Se debe establecer un sistema de referencia del robot como se muestra en la figura 4.4. Los finales de extremidades están denotados como Foot*i*, con $i \in (1, \ldots, 6)$

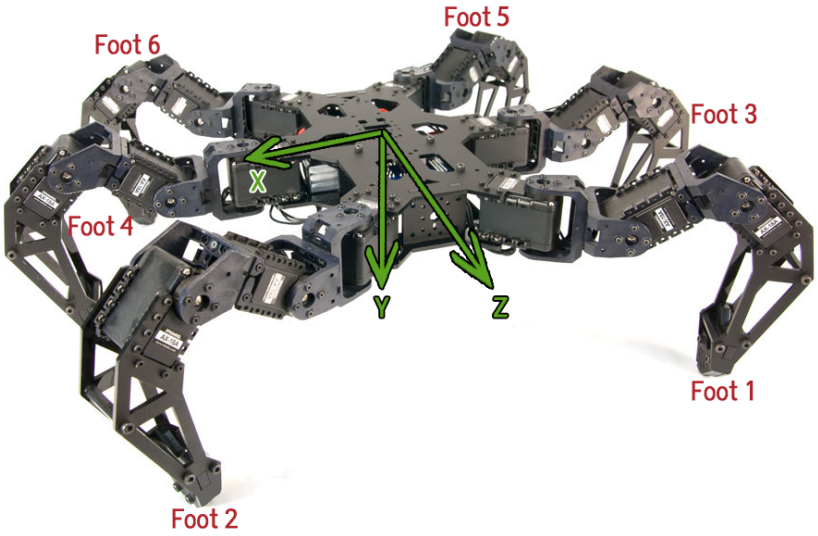



Figura 4.4: Sistema de coordenadas del robot PhantomX. *Fuente propia.*

Para que el robot logre una marcha se necesita que las patas sigan ciertas trayectorias para que el robot ejecute el movimiento, trayectorias en las cuales se debe asignar una posición a los servos para cada punto de la trayectoria. Para lograrlo se toma la posición del final de la tibia y mediante cinemática inversa se obtienen los ángulos de cada articulación que llevarán a que el final de la pata llegue a la posición requerida. Todo ello basados en el eje de referencia indicado en la figura anterior

Se logra evidenciar los ejes cartesianos sobre los cuales el robot está guiado y cómo es nombrada cada una de las extremidades.

Además, por cada extremidad se tiene un sistema de coordenadas (x, y, z) basado en el sistema de coordenadas del robot, como se muestra en la figura 4.5.

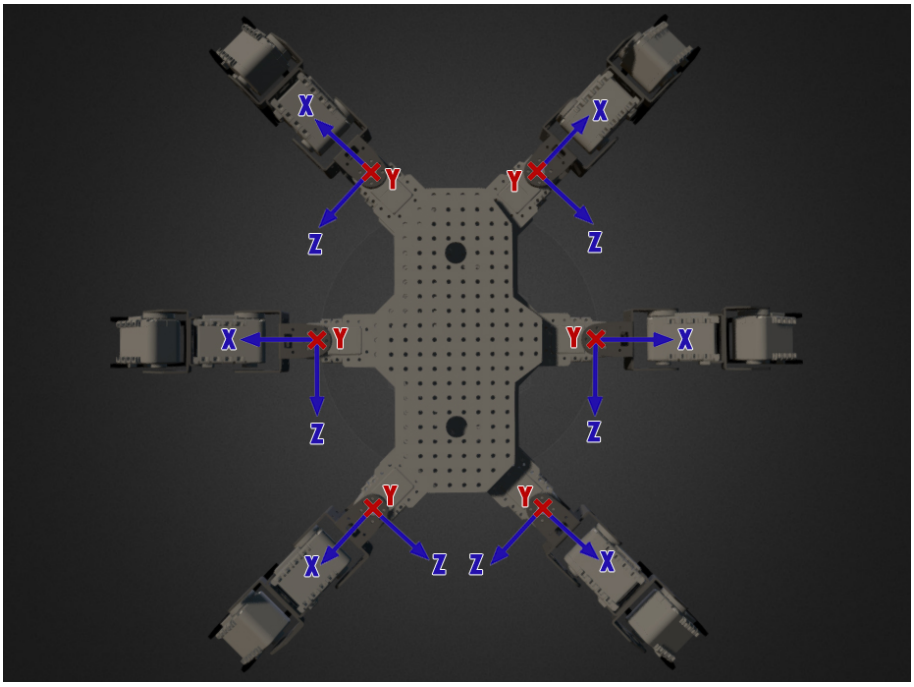



Figura 4.5: Sistema de coordenadas en cada una de las extremidades del robot PhantomX. *Fuente propia*

Para lograr que los finales de las patas lleguen a la posición deseada, se deberá calcular antes el ángulo para cada servomotor mediante las ecuaciones del modelo geométrico inverso del robot.

En la figura 4.6 se observa la notación que se usa para los ángulos objetivo de cada articulación (*V1, V2 y V3*) ubicados en el espacio 3D para entender cómo están ubicados y observar que todas las articulaciones son rotoides.

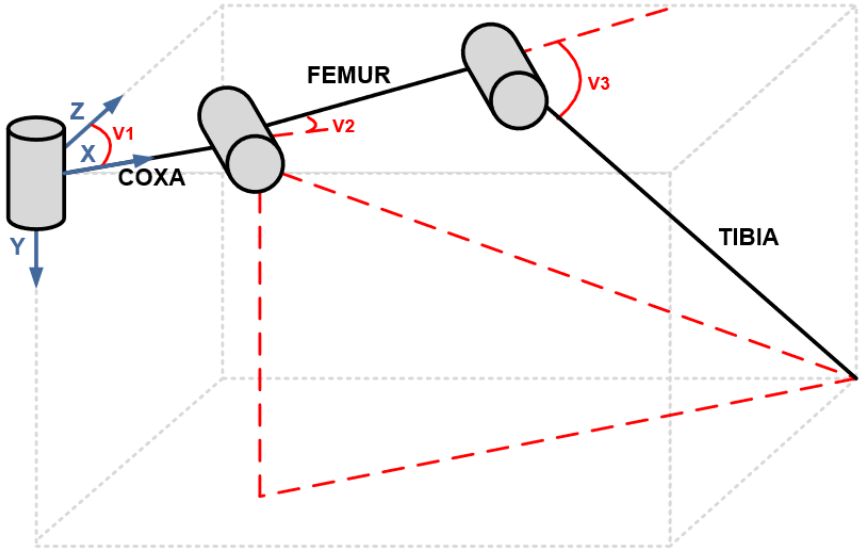



Figura 4.6: Representación geométrica 3D de la extremidad y sus articulaciones. *Fuente propia*

Para realizar el cálculo de las ecuaciones se efectúa un análisis en 2D y este se debe ejecutar por partes ya que dos articulaciones (fémur y tibia) están alineadas en un mismo eje, mientras que la restante (coxa) se mueve en otro.

Para la primera articulación se plantea que la extremidad está totalmente estirada y recta, así tenemos que *r_leg* será la medida de toda la pata, como se ve en la figura 4.7, y el ángulo *V1* que es el que define la orientación del servo de la coxa, se calcula fácilmente a partir de los ejes *x* y *z* con la ecuación 4.4.

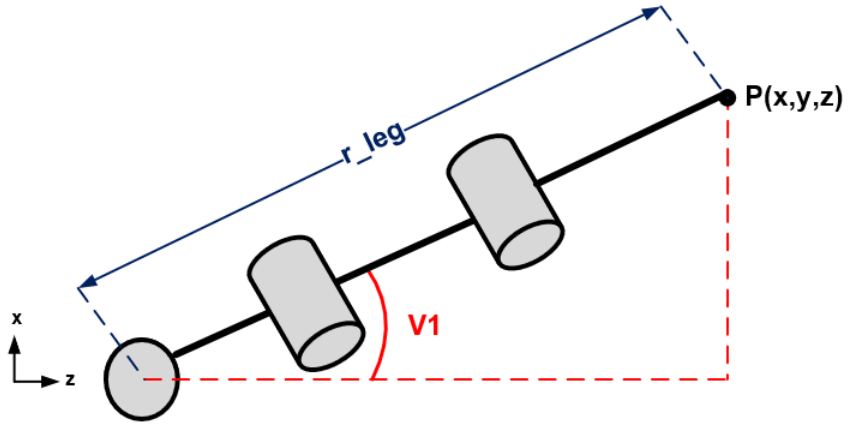



Figura 4.7: Representación 2D de la extremidad en el plano XZ. *Fuente propia*.

$$V1 = tan^{-}1(\frac{x}{z}) \tag{4.4}$$

$$r\_leg = \sqrt{(x^2 + z^2)} \tag{4.5}$$

Para encontrar las ecuaciones que nos llevan a obtener los valores de orientación de los servos del fémur y la tibia, respectivamente, se hace necesario el uso de más parámetros, los cuales se ven en la tabla 4.1.

| Descripción | Parámetro(s) |
|---|---|
| Longitud de los eslabones | L1, L2 y L3 |
| Ángulos complementarios | A1, A2 y B1 |
| Desplazamiento sobre el eje *y* | y |
| Medida en línea recta entre el servo del fémur y el final de la tibia | Hf |
| Desplazamiento en el eje *z* desde el servo del fémur hasta el final de la tibia | ro_leg |

Tabla 4.1: Tabla de parámetros. *Fuente propia*.

Los parámetros necesarios para los cálculos se pueden observar en la figura 4.8, la cual tiene una representación visual de cada uno sobre una vista de plano YZ.

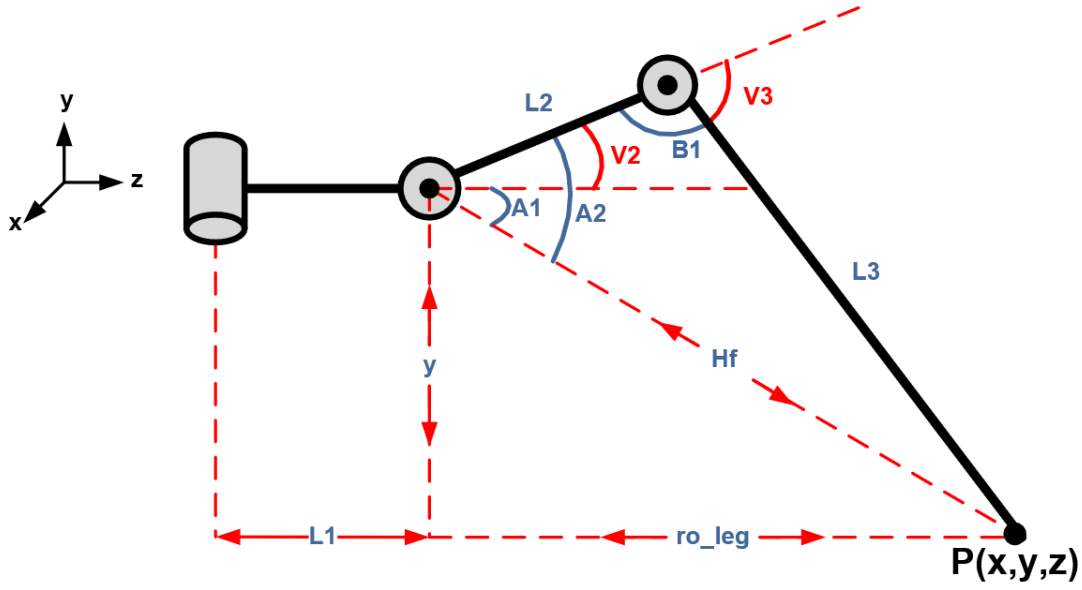



Figura 4.8: Representación 2D de la extremidad en el plano XY. *Fuente propia*.

Las siguientes ecuaciones hacen uso de los parámetros necesarios para que posteriormente se pueda realizar el cálculo de los valores de los ángulos V1 y V2.

$$ro\_leg = r\_leg - L1 \tag{4.6}$$

$$po\_S = ro\_leg^2 + y^2 \tag{4.7}$$

$$A1 = tan^{-}1\left(\frac{-y}{ro\_leg}\right) \tag{4.8}$$

$$A2 = cos^{-}1\left(\frac{L3^2 - L3^2 - po\_S}{-2 * L2 * \sqrt{po\_S}}\right) \tag{4.9}$$

$$B1 = cos^{-}1\left(\frac{po\_S - L3^2 - L2^2}{-2 * L3 * L2}\right) \tag{4.10}$$

$$V2 = A1 - A2 \tag{4.11}$$

$$V3 = 180 - B1 \tag{4.12}$$

Entonces, se tiene que para llegar a un punto de la trayectoria, cada servo debe llegar a un ángulo, dicho valor del ángulo se calcula a partir de las ecuaciones 4.4, 4.11 y 4.12, respectivamente para cada articulación.

Hasta este punto se ha trabajado sobre una estructura ideal de la pata como la indicada en la figura 4.8. Pero posee una diferencia con las patas equipadas por el robot real, ya que las articulaciones no están completamente alineadas entre ellas, lo cual genera una descompensación. Para solucionar tal problema se llevaron a cabo mediciones de la estructura real y se realizó el diseño geométrico real para calcular el desfase y llegar a encontrar el ángulo que lo compense.

En la figura 4.9 se evidencia el desfase existente entre la ubicación de los motores debido a la estructura de la pata.

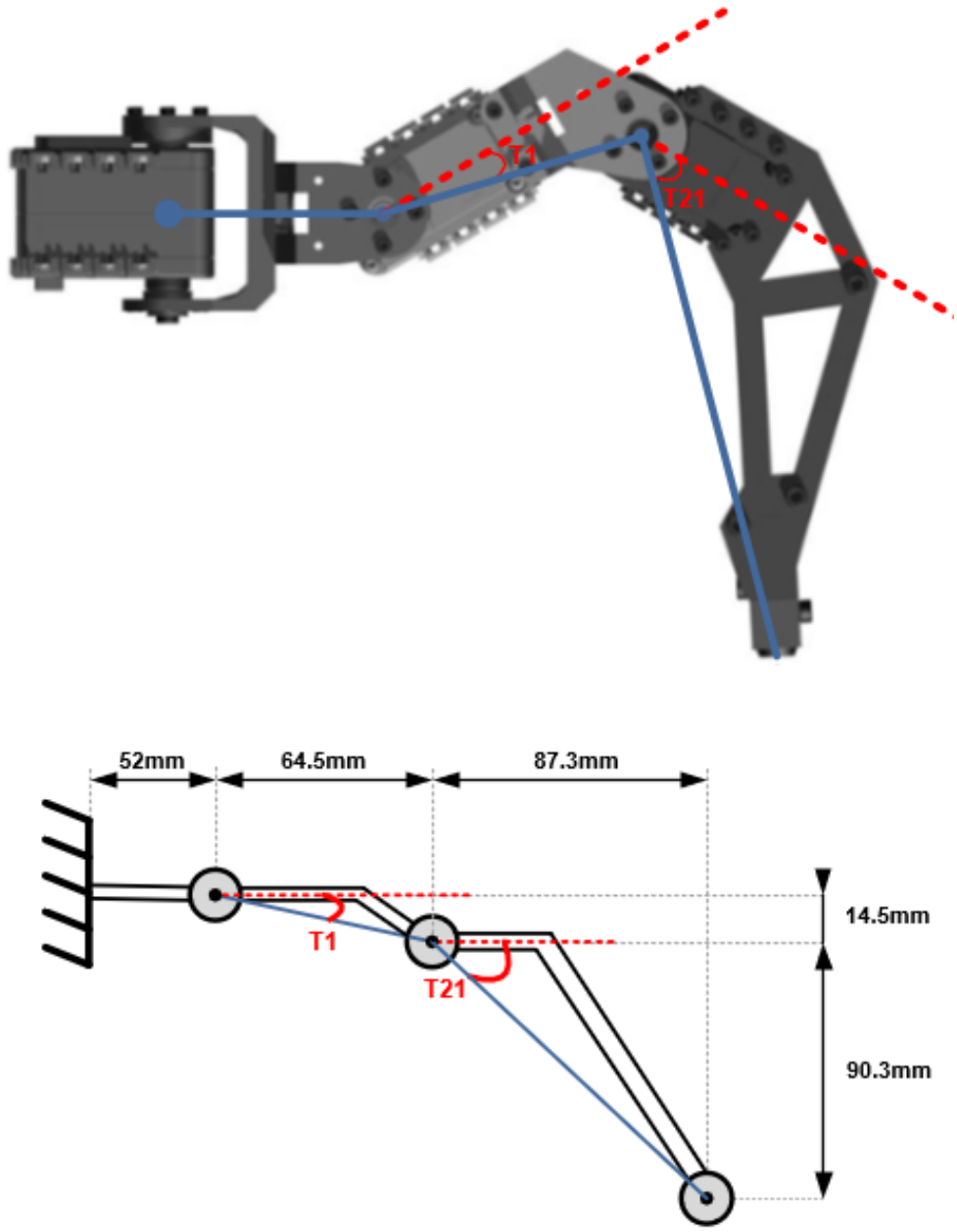



Figura 4.9: Representación geométrica de la estructura real de las extremidades. *Fuente propia*.

Entonces, se procede a realizar el cálculo de los ángulos de descompensación.

$$T1 = tan^{-}1\left(\frac{14.5}{64.5}\right) \tag{4.13}$$

$$T21 = tan^{-}1\left(\frac{90.3}{87.8}\right) \tag{4.14}$$

$$T2 = T21 - T1 \tag{4.15}$$

El ángulo T1 será el ángulo de compensación para el desfase entre el servo de la coxa y el servo de la tibia, el ángulo T2 para el desfase entre el servo de la tibia y el final de la pata. Cada uno se resta a V2 y V3 respectivamente, así se logra compensar el desfase.

Para la realización del proyecto se tomaron las medidas reales de la estructura del robot, de donde se saca la longitud de los eslabones con los cuales se trabaja.

| | **Eslabón** | **Longitud** |
|---|---|---|
| Coxa | L1 | 52 mm |
| Fémur | L2 | 66.109 mm |
| Tibia | L3 | 125.948 mm |

Tabla 4.2: Medida de eslabones. *Fuente propia*.

## 4.3. Implementación de trayectorias

Los robots móviles utilizados en ambientes naturales, tienen que sobrepasar una gran cantidad de obstáculos y algunos de estos aparecen de forma inesperada. La locomoción de dichos robots presenta diversos problemas dado que las condiciones del terreno pueden ser irregulares, encontrando áreas donde el robot no consigue hallar un soporte o ambientes con perturbaciones [71].

Los robots zoomórficos poseen la habilidad de superar estos terrenos irregulares y cumplir con su tarea de desplazarse sobre su entorno en la dirección indicada.

Para llevar a cabo la marcha es necesario tener en cuenta varios aspectos en el robot hexápodo:

- **Mantener el equilibrio del robot:** Se logra conservando la proyección del centro de gravedad dentro del polígono de soporte.
- **Distribución uniforme de la carga soportada:** Durante la marcha del robot cada una de las patas que están en contacto con la superficie soportan el mismo peso.
- **Control de la movilidad de las patas:** El movimiento de las patas debe estar dentro de su espacio de trabajo, evitando colisiones entre las patas o entre una pata y el cuerpo del robot.
- **Búsqueda de soporte de las patas:** Se debe precisar el lugar donde cada una de las patas va a soportar el robot, en el caso de una marcha en una superficie plana esta labor no es complicada, pero en el caso contrario si el robot marcha sobre una superficie irregular esta tarea se vuelve complicada.

En cumplimiento de los aspectos anteriormente mencionados se define:

- El espacio de trabajo de cada una de las patas está definido por el volumen descrito por la cinemática del mecanismo que conforma la pata, el conocimiento del espacio de trabajo permite saber cuál será la movilidad de cada una de las patas. Los espacios de trabajos varían en función de la altura del cuerpo del robot con relación al suelo, restricciones mecánicas y configuración del robot [72].

  En la figura 4.10, el arco rojo representa el espacio de trabajo teórico en el plano de proyección del suelo (aproximadamente 100mm por debajo del robot en forma predeterminada). El arco discontinuo azul representa el posible espacio de trabajo debajo del robot, asumiendo que la pata pueda girar hacia el robot (radio negativo). En la práctica esto no es factible ya que el punto final de la pata sigue la línea de paso a lo largo de la dirección de la marcha (línea gris). Además, un ángulo típico de operación de la pata es pequeño para evitar colisiones entre las demás patas, por lo cual, la longitud del paso de las patas también es limitado.

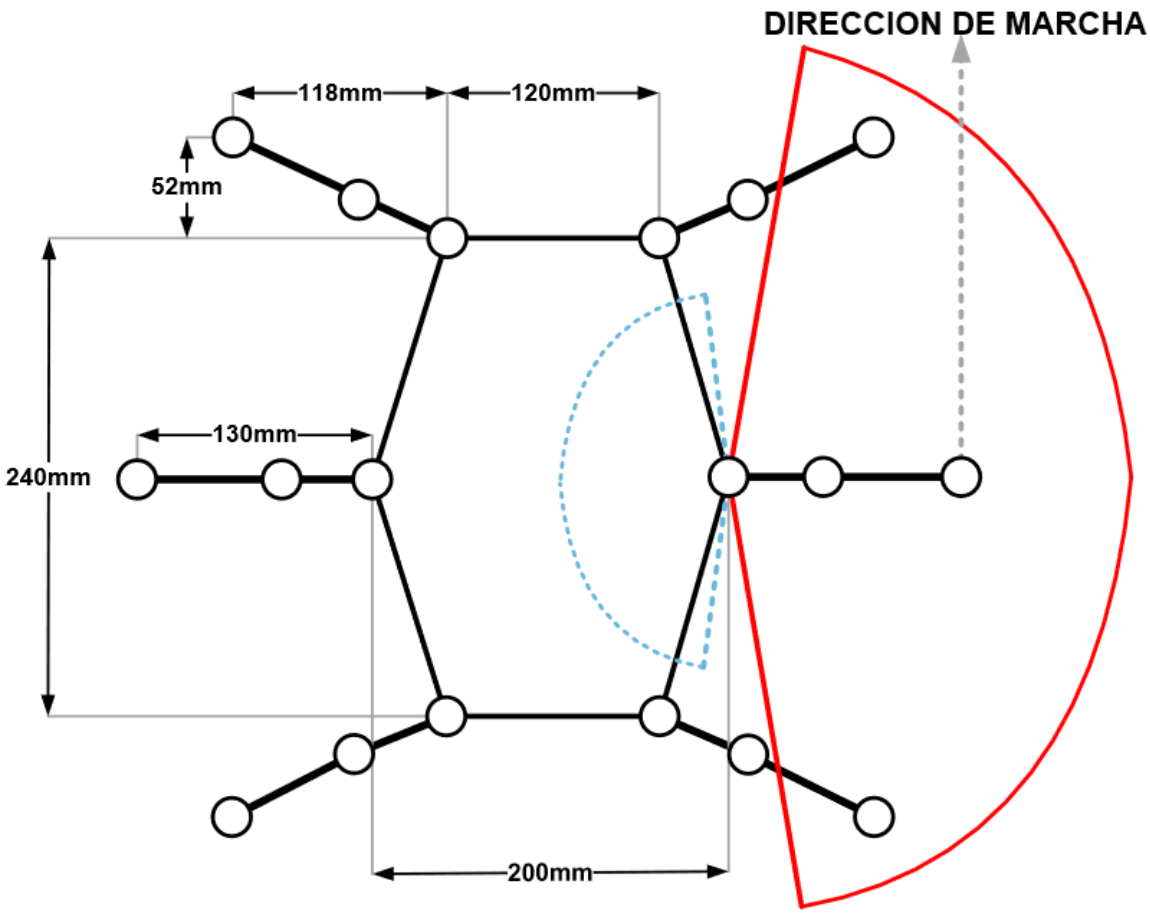



Figura 4.10: Esquema de posición inicial por defecto de un robot hexápodo. *Fuente propia*

- La estabilidad del robot es una de las características más importantes dentro de la marcha, evitando que éste se caiga al momento de realizar un desplazamiento o movimiento de las patas. Para asegurar una buena estabilidad en el cuerpo del hexápodo se considera que el mínimo numero de patas es de tres las cuales al ser apoyadas en la superficie generan un triángulo donde el peso del robot será soportado. El modo de marcha que más se adapta a estas configuraciones y se implementó en el presente trabajo es el de trípode alterno, el cual se indica en la figura 4.11.

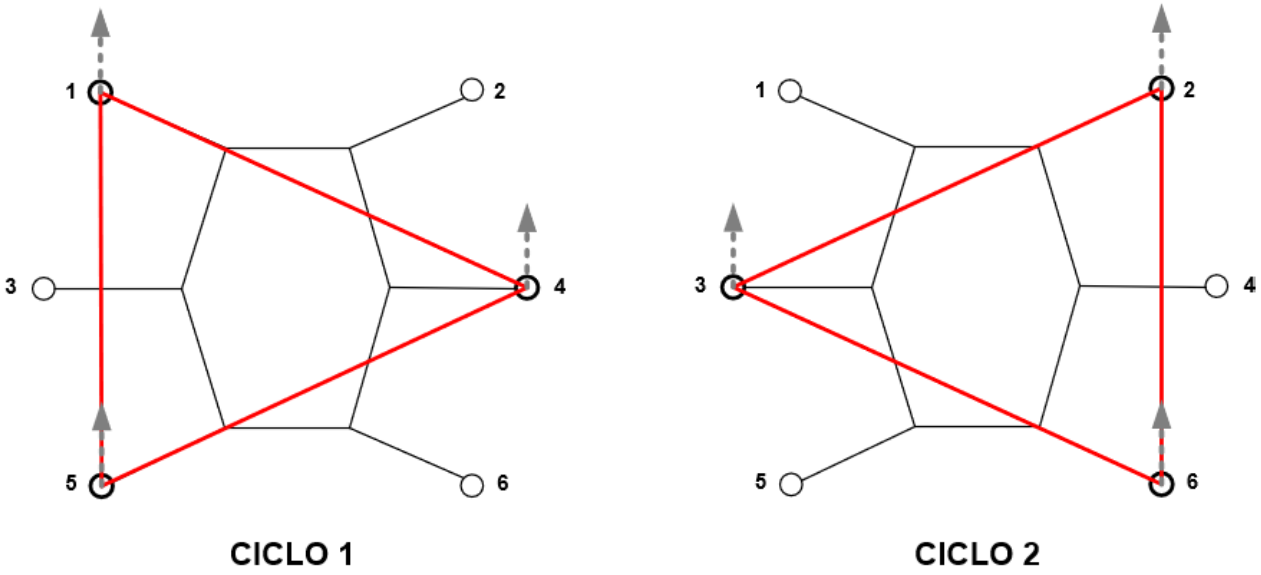



Figura 4.11: Trípodes para locomoción del robot. *Fuente propia*

Un ciclo de marcha trípode consta de ocho fases en total, dos de ellas totalmente compatibles (todas las piernas en el suelo), dedicadas a nivelar el cuerpo. Las piernas se dividen en dos grupos según el patrón. La primera mitad de un ciclo (sin considerar la fase de soporte), las piernas 1, 4 y 5 se mueven en una fase de oscilación, mientras que las piernas 2, 3 y 6 están descansando y apoyando. La segunda mitad del ciclo, sus roles cambian. Una vez que el ciclo termina, el siguiente continúa. El patrón ofrece un buen equilibrio entre la velocidad de movimiento y la estabilidad, el aspecto crucial para atravesar terrenos difíciles.

- Para que un robot con patas pueda moverse hacia cualquier dirección en el espacio, es necesario que cada una de las patas siga una trayectoria compuesta por una curva y una recta sobre la superficie (línea de apoyo). Durante la fase de oscilación la pata recorrerá cada uno de los puntos suspendidos formando una curva en el aire, como se evidencia en la figura 4.12.

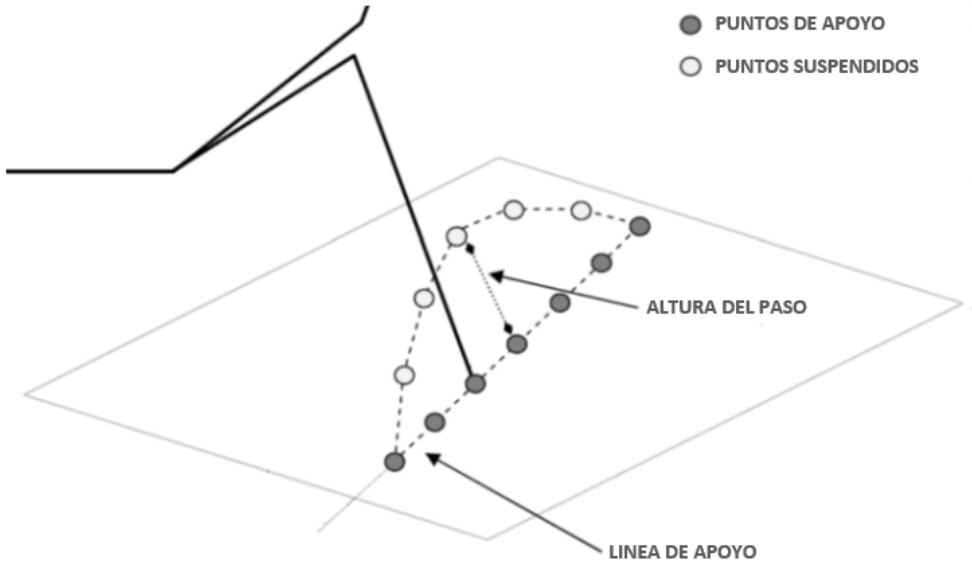



Figura 4.12: Trayectoria a seguir por los finales de las extremidades. *Fuente propia*

La capacidad de locomoción de los robots con patas para atravesar terrenos irregulares puede basarse en dos tipos de control: la primera es la marcha regular para terrenos planos y la segunda es la marcha adaptativa para terrenos irregulares.

### 4.3.1. Modo marcha regular

En la marcha regular se tiene una trayectoria fija predeterminada, donde el movimiento de la pata en la fase de oscilación se divide en dos líneas, pero es posible definir trayectorias de diferentes formas. La marcha regular solo se adapta en superficies horizontales planas, por lo tanto, todos los puntos están estrictamente definidos, este tipo de marcha puede usarse sin la necesidad de sensores, lo que permite que el robot se mueva más rápido. La marcha más rápida del robot hexápodo se traduce en que el robot consume menos energía por distancia recorrida [73]. La trayectoria ideal para este tipo de marcha es la mostrada en la figura 4.13.

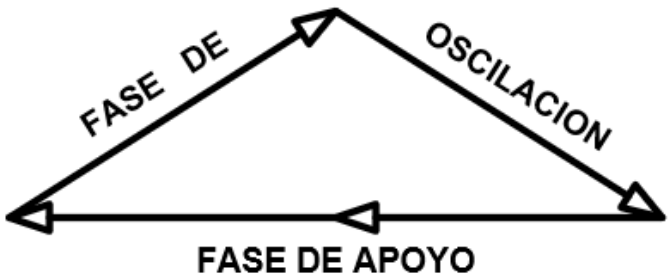



Figura 4.13: Trayectoria de marcha regular [67]

La secuencia que deben seguir los finales de cada pata debe ser guiada con lo mostrado en la figura 4.12 y 4.13. Algunas variaciones de esas trayectorias son basadas en aumentar la velocidad de marcha, incrementar el tamaño del paso, disminuir el movimiento en el cuerpo, entre otras.

Para el trabajo se hizo uso de la marcha en trípode como se indicó en la figura 4.11 y cada trípode debe ejecutar el mismo ciclo de movimiento, pero al tenerse dos trípodes deben realizar acciones contrarias. Mientras un trípode arrastra (fase de apoyo) el otro debe retornar al punto inicial (fase de oscilación) tal como se muestra en la figura 4.14 y se debe llevar a cabo de manera sincronizada en el tiempo.

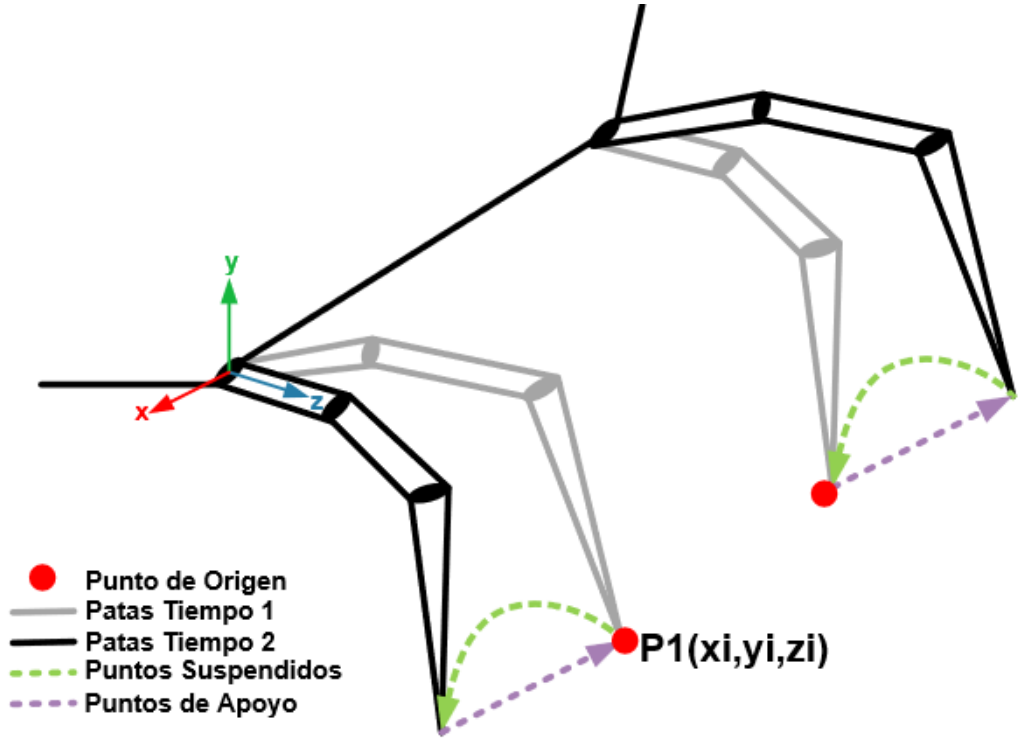



Figura 4.14: Ciclo de ejecución de la trayectoria. *Fuente propia*

Como en una marcha regular sin retroalimentación el robot es incapaz de atravesar un terreno irregular, es posible usar una marcha adaptativa, que es más lenta que la marcha regular, pero permite al robot atravesar un terreno más difícil. Si un terreno es extremadamente difícil, hay solo unos pocos lugares donde el robot puede colocar sus patas, entonces se hace importante desarrollar un sistema de búsqueda de puntos de apoyo [74].

### 4.3.2. Modo marcha adaptativa

La marcha adaptativa permite que el robot se mueva relativamente rápido con un costo computacional bajo con respecto al enfoque de planificación de colocación de punto de apoyo [75]. La

trayectoria guía a seguir para este modo de marcha es la indicada en la figura 4.15

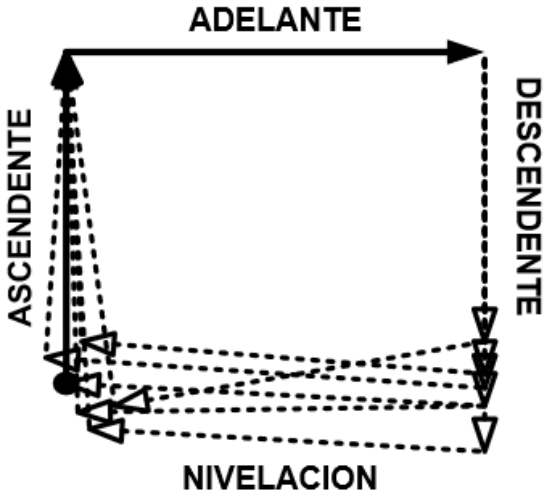



Figura 4.15: Trayectoria de marcha adaptativa [67]

En la marcha adaptativa se definen ciertos puntos de la trayectoria, esto proporciona adaptabilidad a la forma de la trayectoria. Sin embargo, son posibles diferentes formas de trayectoria. En este trabajo se implementó la marcha adaptativa dividiendo la fase de oscilación en tres pasos, el primero paso ascendente, paso hacia adelante y paso descendente que da como resultado una trayectoria en forma de rectángulo [41]. Para este caso se definieron los puntos superiores (el punto final del paso ascendente y el punto inicial del paso descendente), los puntos de contacto con el suelo son variables y se ven limitados por el punto mínimo alcanzable de la pata del robot hexápodo. En los terrenos difíciles, la posición de los puntos de contacto con el suelo cambian durante la marcha debido a las diferencias de altura del terreno, por lo que se requiere una retroalimentación que permite detectar cuando una pata toca el suelo, dato obtenido con un sensor de presión cuando excede un valor de referencia.

- **Búsqueda de puntos**

  En los modos de marcha se necesita encontrar un lugar de apoyo para que las patas puedan soportar el peso del robot, pero en la marcha por terrenos irregulares esta tarea se complica cuando la pata del robot no encuentra apoyo en el lugar programado y se procede a la búsqueda de un nuevo punto de apoyo hasta que este se encuentre y pueda seguir la locomoción. En la figura 4.16 se evidencia el movimiento que debe realizar una pata cuando no encuentra punto de apoyo.

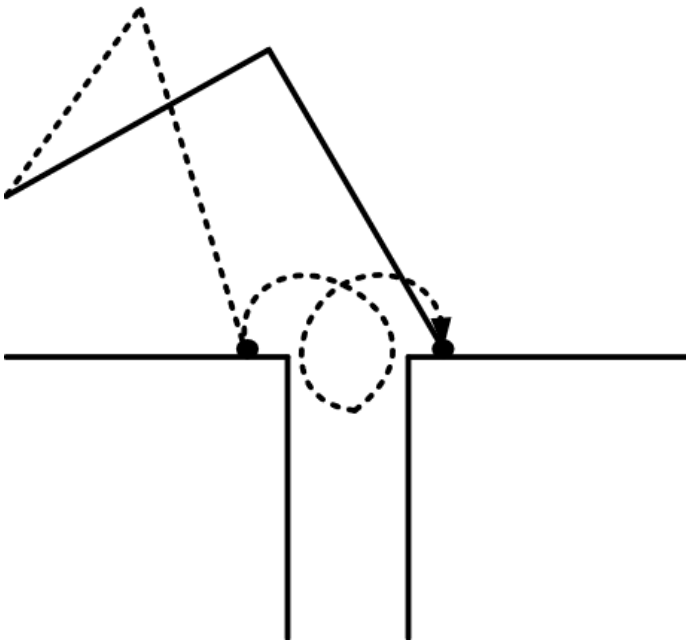


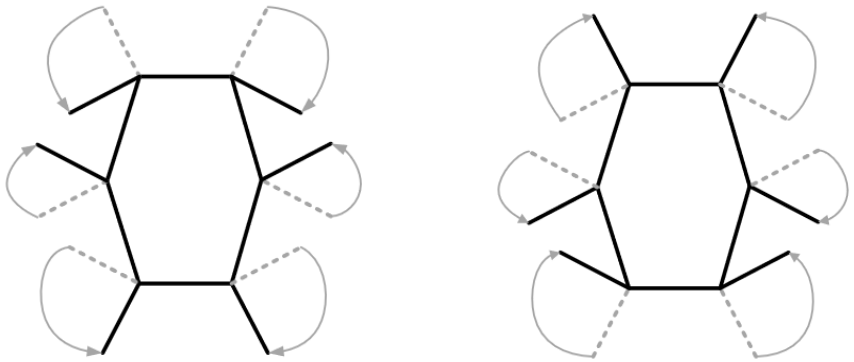
Figura 4.16: Búsqueda puntos de apoyo. *Fuente propia*

Para nuestro proyecto la búsqueda de puntos se realizó de forma aleatoria, el proceso se repetía en 4 oportunidades teniendo en cuenta el espacio de trabajo (limitaciones) de las extremidades. Si se encuentra un punto de apoyo la locomoción sigue hacia adelante, pero si todos los intentos son fallidos y no encuentra punto de apoyo quiere decir que no podrá seguir hacia adelante, ya sea por características extremas del terreno o ubicación del robot, por lo cual al darse dicha situación el robot sigue su locomoción pero esta vez hacia atrás hasta alejarse de esa gran irregularidad.

## 4.4. Sistema de giro

Todo robot tiene limitaciones en su espacio de trabajo, debido a restricciones físicas o del ambiente, los robots hexápodos no son la excepción. En ocasiones el terreno no permite que el robot continúe su locomoción hacia adelante, a veces debido a que existe un abismo o una pared que lo impiden y en ocasiones simplemente se quiere que llegue a un lugar determinado y para ello el robot requiere realizar un giro de cierta cantidad de grados. Se hace importante entonces la implementación de un giro sobre su propio eje, el eje *y* para nuestro caso.

Para realizar el giro partimos de la misma configuración de marcha en trípode e igual del mismo ciclo sincronizado, lo que cambia será el sentido de ejecución de la trayectoria en algunas extremidades. En la figura 4.17 se muestra cual es el sentido de movimiento de cada pata para realizar un giro a la izquierda y a la derecha respectivamente.



(a) Giro izquierda (b) Giro derecha

Figura 4.17: Lógica de giro. *Fuente propia*

Además, es necesario que se planee el ángulo al cual se quiere que el robot llegue con respecto a su anterior posición, para así llegar a un punto deseado.

En la figura 4.18 se puede observar la secuencia de pasos de las patas 3 y 4 para realizar un giro de 90°, la pata 3 comenzando en fase de apoyo.

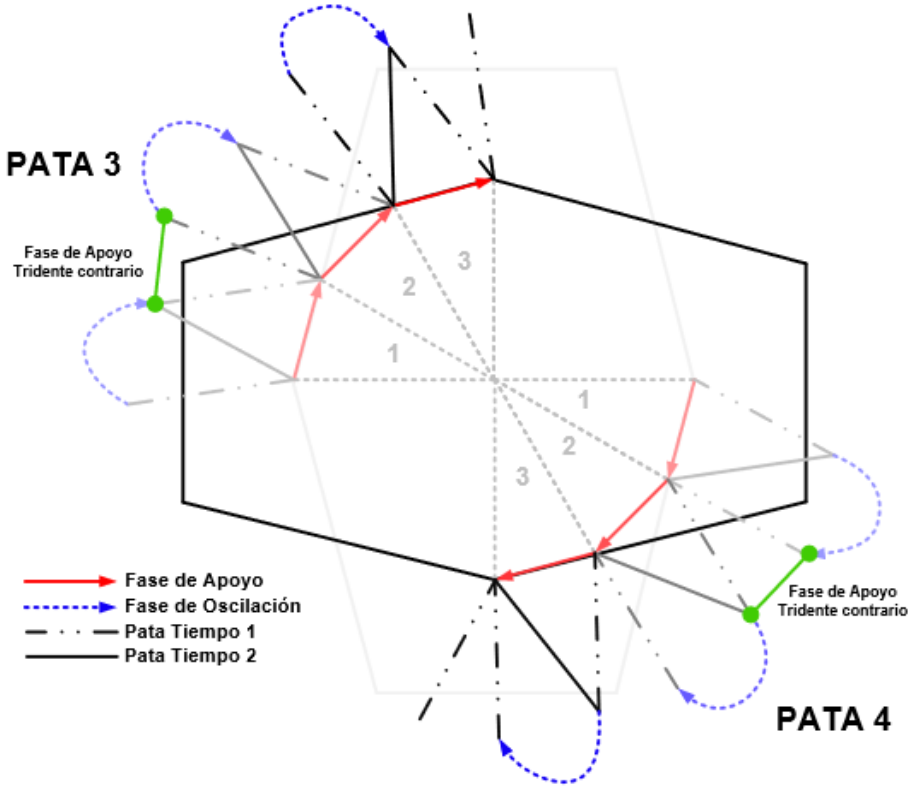



Figura 4.18: Ángulos necesarios para cálculo del giro. *Fuente propia*

La línea roja representa la fase de apoyo propia de cada pata y la línea verde el desfase que genera la fase de apoyo de la pata contraria. El mismo proceso de cálculo sirve para cualquier ángulo deseado e igual con el número de pasos, siempre y cuando estos no lleven a que las patas salgan de su espacio de trabajo seguro. Para los cálculos de giro sobre el eje *y* del robot, se debe conocer el ángulo de giro y la cantidad de pasos a realizar, en este trabajo se predeterminaron los valores anteriores partiendo de 90° y 3 la cantidad de pasos. El programa calcula las posiciones x, y, z de las patas para la fase de oscilación y fase de apoyo, esta secuencia se repetirá según el número de pasos establecidos y siendo alternada en las fases por su trípode alterno.

La realización del giro se lleva a cabo en dos situaciones: la primera cuando existe un abismo, el robot hace una locomoción hacia atrás unos pasos hasta que el terreno sea estable y después gira de forma automática eligiendo un ángulo al azar entre 45°, 90°, 135°, 180° y sus correspondientes ángulos negativos. La segunda situación se da a partir de una orden recibida por medio del joystick, esa orden activa el modo de giro pero la elección sobre el ángulo objetivo se lleva a cabo manualmente por un usuario.

## 4.5. Sistema de estabilización

La estabilidad del robot es una de las características más importantes en la locomoción. Uno de los propósitos del estudio de la estabilidad es primeramente evitar que el robot se caiga cuando este lleve a cabo un desplazamiento.

Hay dos tipos de estabilización: la estática y la dinámica. La estabilización dinámica se realiza al tiempo que se realiza la locomoción, mientras la estática se lleva a cabo cuando el robot está completamente quieto [76]. Para este proyecto se hizo una unión de los dos tipos, es decir, la estabilización se lleva a cabo para la marcha pero mientras el robot está estático y todas las patas poseen un punto de apoyo.

Para lograr la estabilidad se parte de un centro gravitacional y en ese punto se ubica el sensor que nos entrega valores de ángulos de desfase con respecto al punto estable. La IMU entrega los valores de los ángulos de Euler de los que vamos a hacer uso de *Roll* y *Pitch*, los cuales nos entregarán ángulos correspondientes al desajuste en cada eje. La ubicación de los ejes de *Roll* y *Pitch* se indica en la figura 4.19, denotando los 4 cuadrantes de combinaciones entre los dos ángulos positivos y negativos.

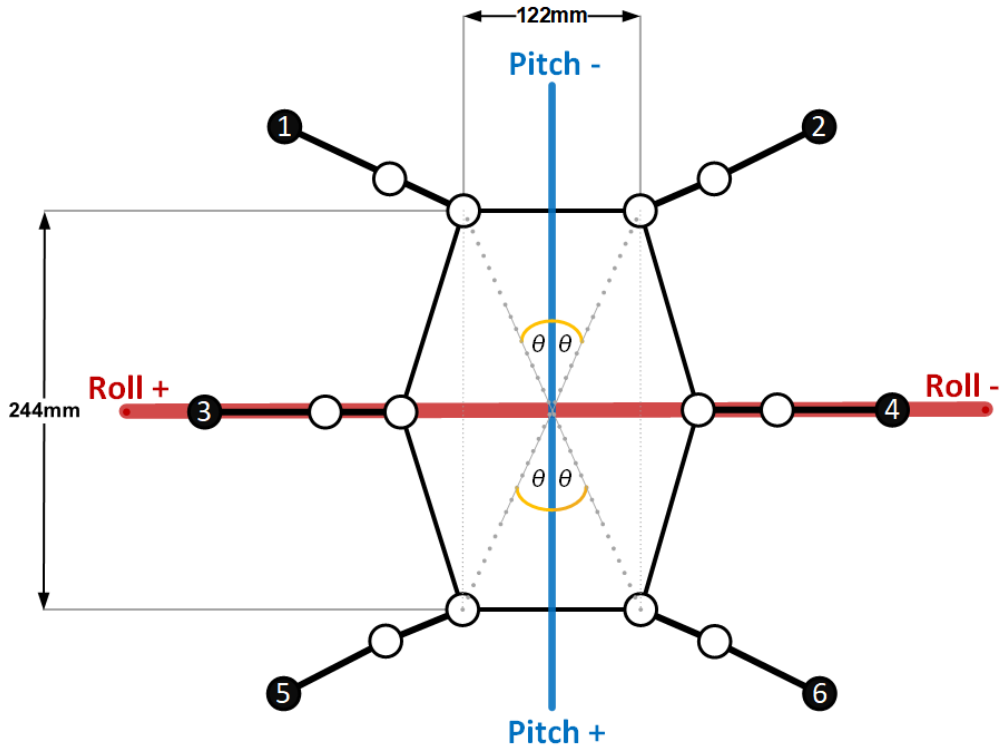



Figura 4.19: Ángulos de desfase para cada pata. *Fuente propia*

Entonces, con la siguiente ecuación se obtiene el ángulo que refiere al grado de incidencia de *Roll* y *Pitch* en las patas ubicadas en las esquinas del robot. Esto debido a que las patas no se encuentran alineadas al mismo eje de los ángulos que calcula la IMU.

$$\Theta = tan^{-}1\left(\frac{61}{122}\right) \tag{4.16}$$

Siendo $\Theta$ el ángulo que se forma entre el centro del cuerpo y el inicio de las patas ubicadas en las esquinas. Así, con los valores de *Roll* y *Pitch* se logra conocer el ángulo de inclinación de cada pata con respecto al cuerpo. La incidencia de cada ángulo con respecto a las patas se indica en la tabla 4.3.

| Extremidad | Roll | Pitch |
|---|---|---|
| 1 | *Ángulo positivo* | *Ángulo negativo* |
| 2 | *Ángulo negativo* | *Ángulo negativo* |
| 3 | *Ángulo positivo* | *Sin incidencia* |
| 4 | *Ángulo negativo* | *Sin incidencia* |
| 5 | *Ángulo positivo* | *Ángulo positivo* |
| 6 | *Ángulo negativo* | *Ángulo positivo* |

Tabla 4.3: Relación de los ángulos *Pitch* y *Roll* con cada pata. *Fuente propia*

$$\textit{Angulo de Desfase}1 = \textit{Roll} * Sin(\Theta) - \textit{Pitch} * Cos(\Theta) \tag{4.17}$$

Entonces, con la ecuación 4.17 logramos conocer el ángulo de desfase de cada pata, a partir del cual se deberá sumar o restar una cantidad a la altura en el eje *y* segun corresponda.

## 4.6. RoboComp

Cuando se desarrolla software para robots se tienen que abordar problemas muy específicos. Además, ha de ser a la vez eficiente, fácil de utilizar y de extender. Para poder conseguir buenos resultados se deben seguir técnicas concretas de ingeniería del software, que aborden las siguientes cuestiones: complejidad conceptual del software, reusabilidad y escalabilidad del código, distribución de la computación, soporte multiplataforma y multilenguaje e independencia del hardware [77].

RoboComp [78] es un *framework* robótico de código abierto. Está compuesto por diferentes componentes de software robótico y las herramientas necesarias para utilizarlos. Los componentes en ejecución forman un gráfico de procesos que pueden distribuirse en varios núcleos y CPU utilizando la tecnología de componentes de software. Fue desarrollado por el equipo del laboratorio RoboLab de la Universidad de Extremadura, España, el cual desde el año 2010 posee una versión estable.

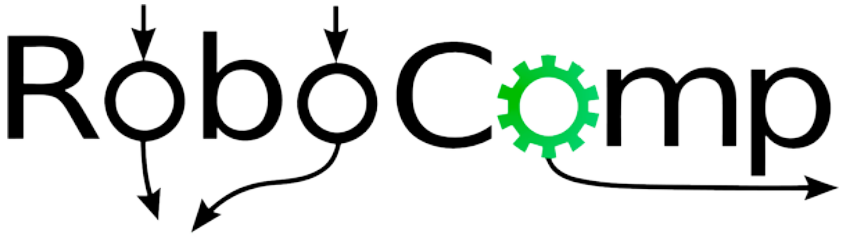



Figura 4.20: Logo de RoboComp [79]

El framework de desarrollo para robótica RoboComp está basado en la programación orientada a componentes (COP), una técnica utilizada para la implementación de sistemas software. Este tipo de programación aumenta el nivel de abstracción respecto a técnicas como la orientación a objetos, lo que mejora notablemente dos de los grandes problemas de la creación de software: la reutilización y la escalabilidad. Un componente [80] es un programa software que forma parte de un sistema software mayor y ofrece servicios mediante una interfaz predefinida, a través de la cual, es capaz de comunicarse con otros componentes. Dicho componente será totalmente reemplazable por otro que cumpla con las interfaces declaradas.

Las características principales de un componente son:

- Ser reutilizable.
- Ser intercambiable.
- Poseer interfaces definidas.
- Ser cohesivos.

Para hacer posible la inclusión y comunicación de componentes se hace uso del middleware Ice (*Internet Comunication Engine*) orientado a objetos [36], el cual asiste a una aplicación para interactuar o comunicarse con otras aplicaciones, programas, redes, hardware o sistemas operativos; al igual que el enfoque adoptado por el sistema de robótica Orca2 [81]. Los principales objetivos en el diseño del sistema son la eficiencia, la simplicidad y la reutilización.

Para realizar la conexión entre dos componentes en RoboComp, sólo se necesita: el nombre de la interfaz, la dirección IP de la máquina donde se aloja el componente y el número de puerto asociado, así:

<Nombre de la Interfaz >: <TCP / UDP>-p puerto -h host

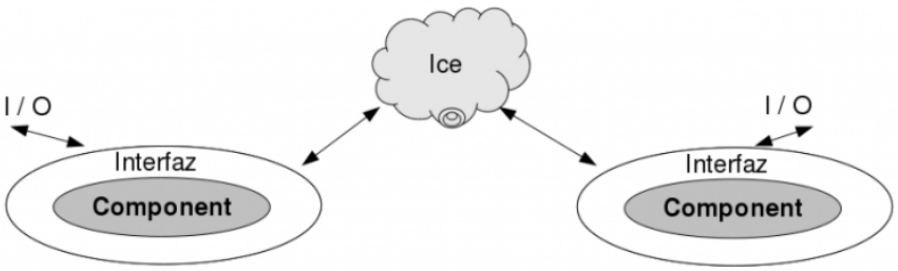



Figura 4.21: Representación genérica de componentes [82]

Entonces, RoboComp es un *framework* de desarrollo para robótica de código abierto que ofrece la posibilidad de crear componentes de una manera fácil y sencilla, comunicándose estos a través de interfaces públicas. Para generar un componente de RoboComp se utiliza un lenguaje de dominio específico, llamado CDSL, un ejemplo de dicho lenguaje sería:

```
import "import1.idsl";
import "import2.idsl";

Component myComponent
{
        Communications
        {
                implements interfaceName;
                requires otherName;
                subscribesTo topicToSubscribeTo;
                publishes topicToPublish;
        };
        language Cpp;
        gui Qt(QWidget);
};
```

Figura 4.22: Generación de un archivo CDSL [83]

De esta manera, un componente generado con RoboComp puede suscribirse o publicar un tópico e implementar o hacer uso de una interfaz, mediante los lenguajes de programación Python o C++. Los componentes generados con RoboComp tienen un archivo de configuración en el que se puede configurar el puerto y la dirección IP de una interfaz implementada en otro componente, el tamaño de los mensajes de comunicación, añadir variables de configuración, etc. Además, RoboComp cuenta con otro lenguaje específico de dominio para la definición de interfaces, IDSL, que genera los archivos ".idsl". Gracias al *framework* de Ice, los componentes escritos con RoboComp pueden comunicarse con componentes escritos en otros lenguajes, como Java, PHP, etc.

RoboComp también utiliza en algunos de sus componentes otras herramientas como CMake [84], Qt4 [85], IPP [86], OpenSceneGraph [87] y OpenGL [88]. Los lenguajes de programación utilizados en el proyecto son principalmente C ++ y Python, pero el marco de trabajo de Ice hace posible el uso fácil de componentes escritos en muchos otros idiomas. RoboComp, como ROS(*Robot Operating System*), también se puede llamar un sistema operativo de robot siempre que proporcione una capa de abstracción al hardware del robot. En comparación con otros proyectos similares (como Orca [89] o ROS [90]), la principal ventaja de Robocomp es su eficiencia y su facilidad de uso.

### 4.6.1. RCIS

Para los proyectos de robótica es imperante, además del *framework*, tener un simulador que ofrezca las condiciones necesarias para generar una simulación confiable del robot para realizar

pruebas antes de aplicar tareas al robot real.

Una de las principales tareas realizadas en RoboComp ha sido el diseño y la construcción de un simulador de robótica. Este esfuerzo ha sido parcialmente mitigado por la reutilización de los elementos *InnerModelDSL* y de la tecnología de visualización *OpenSceneGraph*. La combinación de estos componentes, junto con un diseño cuidadoso, ha llevado a la creación del simulador *RoboComp Innermodel Simulator* (RCIS), un simulador 3D. La característica más importante de este simulador es que también es un componente nativo de RoboComp. Siendo así, puede implementar todas las interfaces de los componentes existentes en la capa de abstracción de hardware, es decir, cámaras, láseres, Kinect, motores, parachoques, ultrasonidos y entradas táctiles. El resto de los componentes en un determinado gráfico de implementación pueden comunicarse con estas interfaces como si fueran los componentes originales, lo que facilita enormemente el ciclo de desarrollo de algoritmos complejos [91]. La figura 4.23 muestra una captura de pantalla de RCIS.

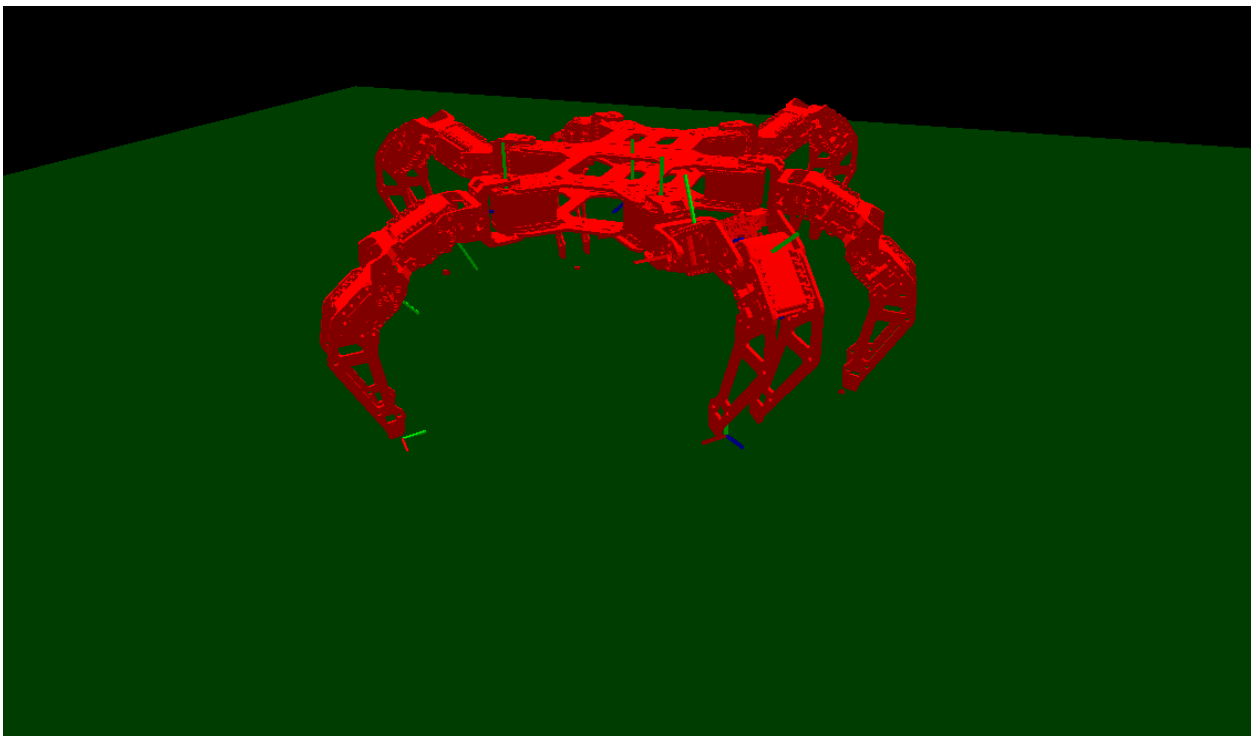


Figura 4.23: Simulación en RCIS. *Fuente propia*

### 4.6.2. Comunicación de componentes

Para llevar a cabo todo el algoritmo se dividieron los procesos en distintos componentes de RoboComp, de los cuales algunos eran ejecutados en el Odroid instalado sobre el hexápodo y otros componentes en el ordenador. Dicha distribución se muestra en la figura 4.24.

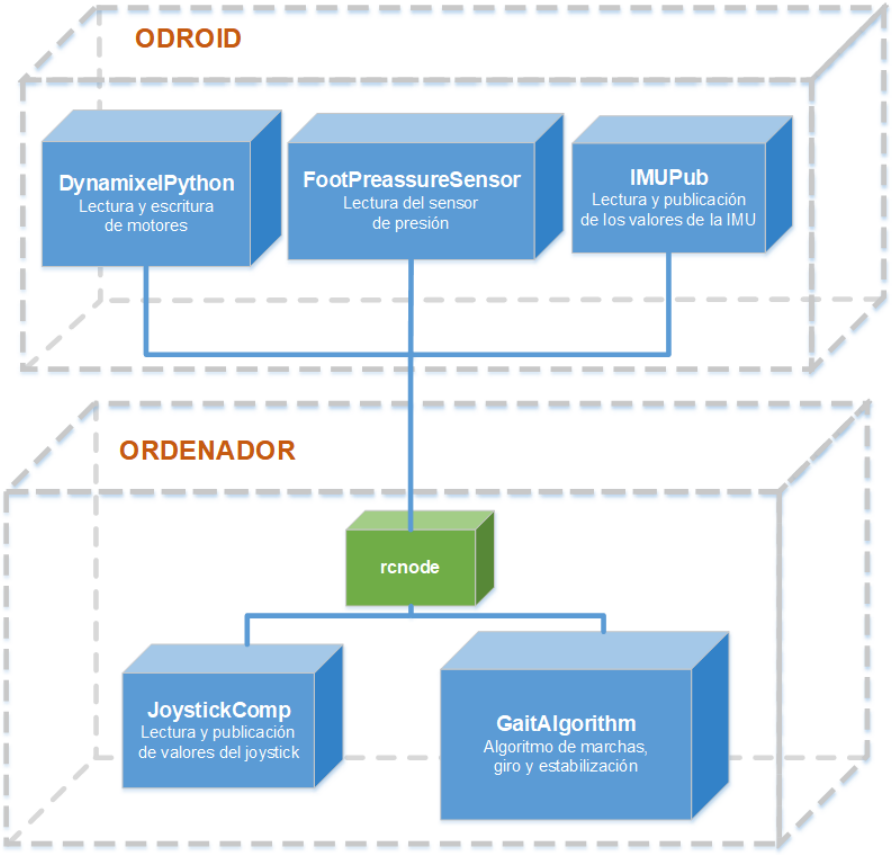



Figura 4.24: Distribución de componentes de RoboComp. *Fuente propia*

Para el desarrollo del algoritmo se requirió la existencia de 5 componentes desarrollados en RoboComp y el uso directo de una de sus herramientas (*rcnode*). Tres componentes fueron ejecutados en el Odroid y dos en el ordenador, esta distribución era necesaria ya que cada componente cumple con un objetivo específico.

- ***DynamixelPython***

  El componente ya hacía parte de RoboComp. Este componente implementa el predeterminado de RoboComp llamado *JointMotor* para comunicación con los motores Dynamixel, a través del cual se realiza lectura y escritura de sus datos. El desarrollo del componente era necesario ya que el software de uso original está en lenguaje de programación C++ y este proyecto fue desarrollado en lenguaje de programación Python. Se realizó una adecuación al código ya existente en lo que respecta al manejo de datos de los motores y la velocidad de comunicación para mejorar la respuesta.

- ***FootPreassureSensor***

  Se hace necesario también la comunicación con los sensores de presión ubicados en los finales de las extremidades del robot. La secuencia de comunicación es de los sensores a

la placa Arduino y de esta hacia el Odroid, por lo cual este componente estará encargado de la comunicación Arduino-Odroid. Para el componente ya existente no fueron necesarias adecuaciones.

- ***IMUPub***

  Este componente se desarrolló en el marco de la implementación de la estabilización sobre el robot, para lograrlo se necesitan datos de desfase del dispositivo y eso lo conseguimos con la instalación de una Unidad de Medida Inercial (IMU). Este componente fue encargado de leer todos los datos en tiempo real de la IMU y publicarlos.

- ***JoystickComp***

  En el proyecto se hizo uso de un *joystick* para cambiar el modo de marcha (entre regular y adaptativa), dar órdenes de ejecución de giros y en la fase de desarrollo para realizar distintas pruebas. El componente es uno de los ya desarrollados en RoboComp, por lo que simplemente se debe ejecutar y se encarga de publicar los datos del *joystick* para un componente distinto que se suscriba.

- ***GaitAlgorithm***

  El *Gait Algorithm* es el algoritmo de control contenedor de la lógica que logra hacer que el robot tenga una marcha regular y una marcha adaptativa, los cambios de modos, giro, cálculo de trayectorias, etc. Para todo ello se hace uso de los datos publicados por otros componentes como el *IMUPub* y el *JoystickComp*. Para hacer uso de ellos se debe realizar una suscripción a los mismos. Además, para comunicarse con los componentes *FootPreassureSensor* y el *JointMotor* hay que requerirlos para hacer uso de ellos.

- ***rcnode***

  *rcnode* es el componente que permite la comunicación entre todas las interfaces de los componentes y, después de todo, los componentes mismos. Se encarga de inicializar el middleware Ice como protocolo de comunicación. Para la comunicación de distintos componentes solo debe existir un *rcnode*, de haber varios en la misma comunicación ésta no será posible. Puntualmente en este proyecto se hizo la ejecución del *rcnode* en el ordenador y los componentes que requieran uso de Ice tienen que cambiar su proxy de comunicación al del ordenador.

Uno de los puntos más importantes y determinantes para que el algoritmo fuese efectivo es el uso de programación por hilos, cada pata es un hilo y así funcionan en paralelo. La programación por hilos conlleva un gran costo computacional, es por ello que el programa principal se ejecuta en el ordenador y solo envía y recibe una menor cantidad de datos desde el Odroid.

El código posee las ecuaciones de la cinemática inversa, por lo que puede calcular las posiciones de los servos para cada punto de la trayectoria. También posee las ecuaciones del giro y la estabilización para hacer los cálculos necesarios y que se lleven a cabo de manera adecuada.

En la figura 4.25, podemos ver el diagrama de flujo del algoritmo de control completo con la ejecución de los subsistemas.

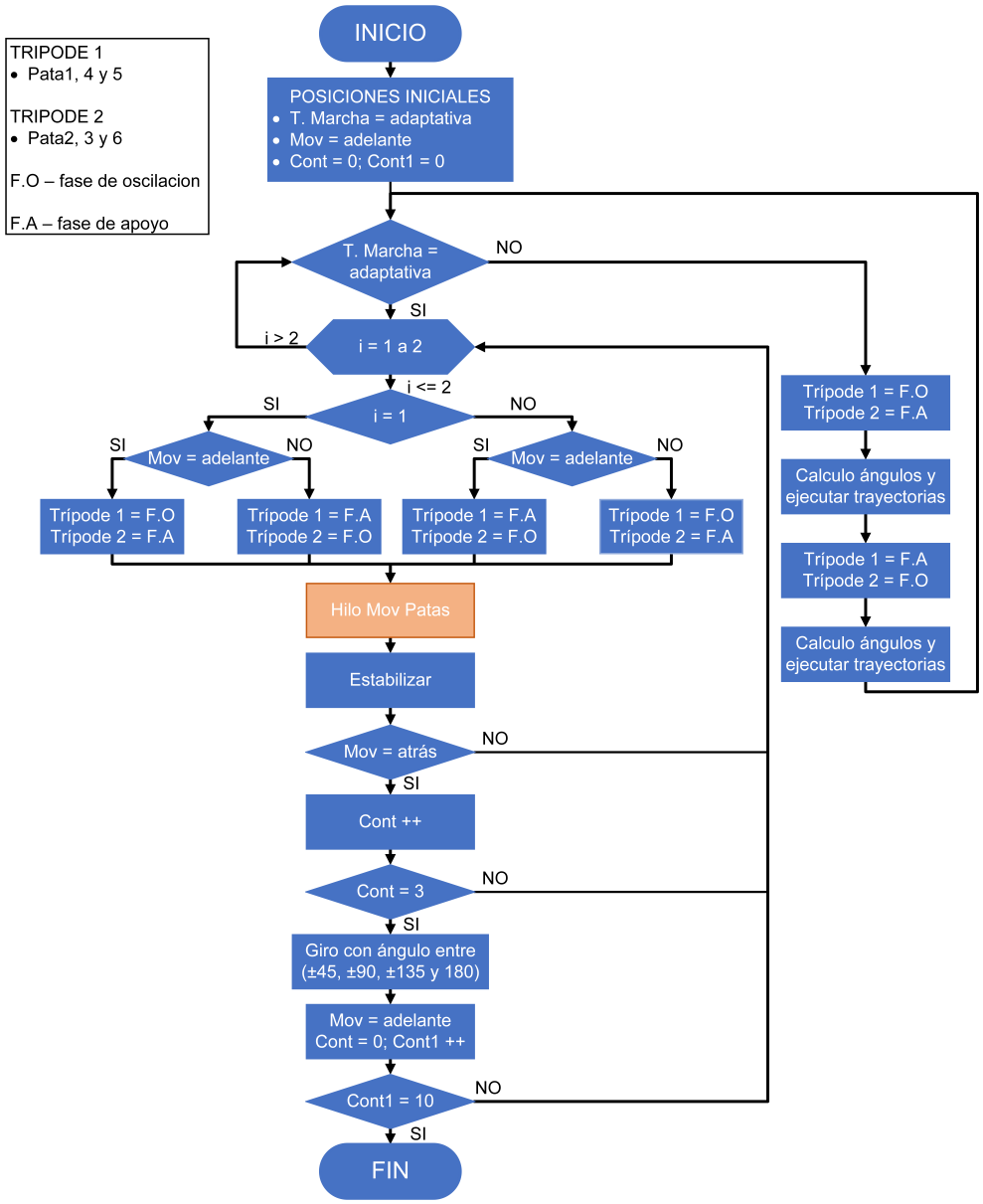



Figura 4.25: Diagrama de flujo de un hilo del algoritmo. *Fuente propia*.

En la figura 4.26, podemos ver la representación de un hilo, el funcionamiento básico que se debe ejecutar por cada pata mientras está la marcha adaptativa accionada.

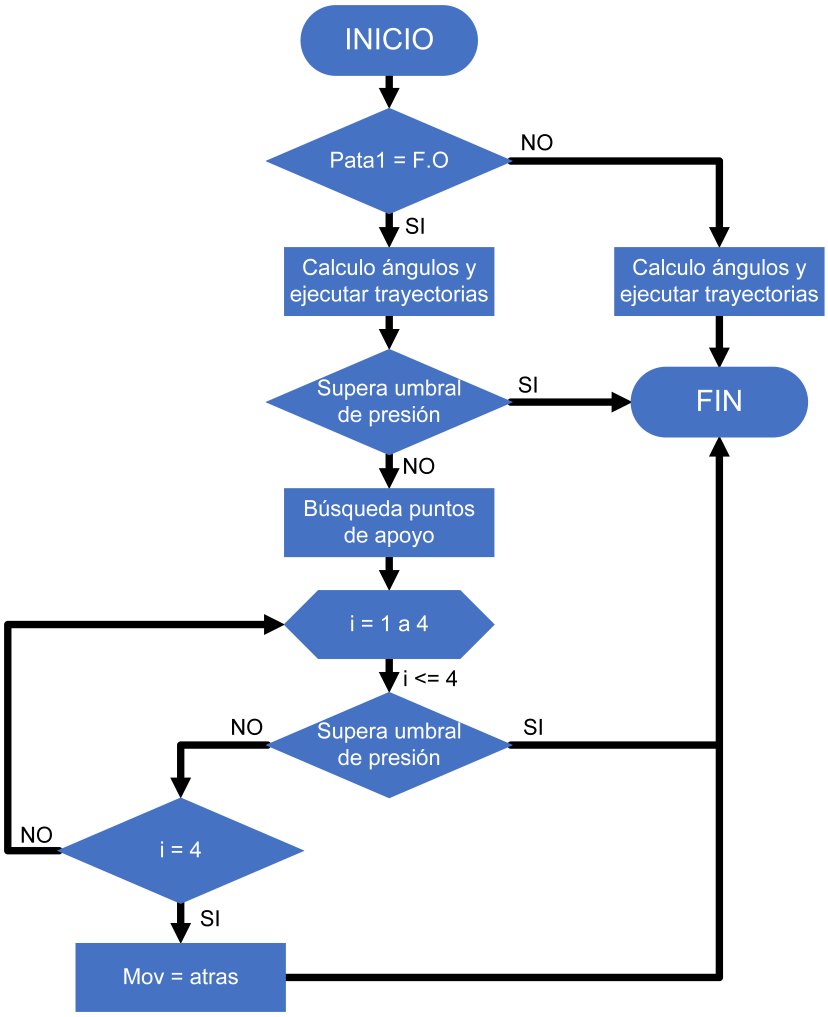



Figura 4.26: Diagrama de flujo de un hilo del algoritmo. *Fuente propia*.

Además, para realizar adecuadamente las marchas, los trípodes deben sincronizarse. Dicha sincronización se llevó a cabo con un condicional, el cual evaluaba que para que un trípode entre al ciclo de oscilación, todas las extremidades (6) deben tener un apoyo previo. Por lo tanto, si un trípode está listo para pasar al ciclo de oscilación, tendrá que esperar a que el trípode contrario tenga puntos de apoyo.

# Capítulo 5

# Experimentación y resultados

En el presente capítulo se presenta el proceso de experimentación y los resultados obtenidos en las pruebas. Se muestran también las características de los terrenos y las pruebas realizadas.

## 5.1. Ambientes de trabajo

Para lograr la locomoción del robot hexápodo, cada una de las patas debe seguir una trayectoria determinada, ya sea para una marcha regular o para una marcha adaptativa. El resultado de las trayectorias reales que ejecutó cada pata cambia dependiendo de algunas variables como son: el tipo de terreno, la ubicación de la araña, presencia de obstáculos, existencia de huecos, encuentro con abismos, etc.

Para determinar cuáles son las características de la marcha adaptativa del robot PhantomX en cambios de dirección y posición sobre diferentes ambientes, se debe partir de la respuesta de la marcha regular, su trayectoria en funcionamiento y definir en qué ambientes puede actuar cada tipo de locomoción.

Para este proyecto se realizaron pruebas en 3 ambientes distintos, de los cuales 2 permitían locomoción regular y adaptativa, mientras que para realizar locomoción en el último esta debía ser obligatoriamente adaptativa por lo complicado del terreno.

Los primeros dos ambientes son los indicados en la figura 5.1.

(a) Ambiente *Mesa* (b) Ambiente *Piso en cerámica*

Figura 5.1: Ambientes para marcha dual. *Fuente propia*.

Los primeros dos terrenos son planos, sin ninguna inclinación, son ambientes de laboratorio controlados. Además, para las pruebas de marcha adaptativa en ellos se adicionaron obstáculos con superficie plana, entonces la locomoción regular se desarrolla en el terreno plano y en el momento de arribar a la posición del obstáculo se realiza el cambio a marcha adaptativa.

El ambiente restante no es controlado, es un terreno real en exteriores al cual se llevó el robot. Dicho terreno tiene inclinaciones con las que se puede poner en acción la estabilización y comprobar su eficacia. El ambiente se muestra en la figura 5.2.

Figura 5.2: Ambiente *Campo*. Terreno irregular real. *Fuente propia*.

Se realizaron varias pruebas de marcha, estabilización y giro sobre dichos ambientes, de los cuales se obtuvieron datos que se analizan a continuación.

## 5.2. Estabilización

La primera prueba de estabilización en la marcha adaptativa se realiza en el ambiente “Mesa” con un trayecto de aproximadamente 50 centímetros por un tiempo de 50 segundos. Los resultados mostrados en la figura 5.3.a indican que durante la fase de apoyo y oscilación se presentan variaciones de ángulos debido a la inclinación de cuerpo del robot que se produce durante la marcha. La obtención de los datos se hace en tiempo real, por lo que se ve la transición de la inestabilidad a cuando llega a ser estable.

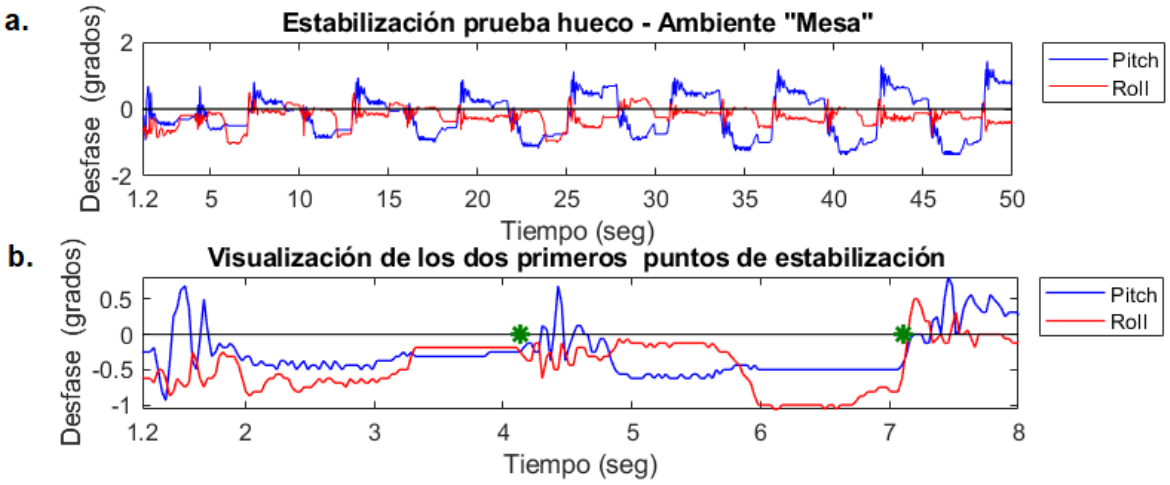



Figura 5.3: Prueba de estabilización ambiente ”Mesa”. *Fuente propia*.

Se logra evidenciar la evolución de los ángulos de rotación durante todo el trayecto del robot, los cuales no son considerablemente grandes, pero la acumulación de estos puede afectar a la estabilidad del robot, aumentar el desfase en cada paso y llevar al colapso del sistema. En figura 5.3.b se observa los datos de un tramo de tiempo de la primera gráfica. Además, en la tabla 5.1 se puede ver los datos de desfase tomados al ejecutar la estabilización (los puntos verdes de la gráfica 5.3.b)

| Tiempo (sg) | Roll (grados) | Pitch (grados) |
|---|---|---|
| 4.1320 | -0.1875 | -0.2500 |
| 7.1050 | -0.6250 | -0.6250 |

Tabla 5.1: Datos de desfase después de ejecutar la estabilización. Ambiente ”Mesa”. *Fuente propia*.

Así podemos observar de manera más clara la representación de los ángulos obtenidos durante la primera fase de oscilación y apoyo comenzando desde el segundo 1.2, debido a que es el tiempo que se tarda el robot en asignarle a las patas las posiciones iniciales antes de comenzar su desplazamiento. La estabilización se ejecuta previo a la finalización de cada fase de marcha indicados por el punto verde donde se obtiene una mínima o nula variación de dichos ángulos representados por una línea recta.

A continuación, se realiza una prueba similar a la anterior pero en un ambiente distinto y con una mayor trayectoria. Dicha prueba se realiza en el ambiente “Campo” que posee terreno irregular sobre una base plana. Los resultados son mostrados en la figura 5.4.a, donde la estabilización mantiene una pequeña variación de los ángulos respecto a una inclinación nula.

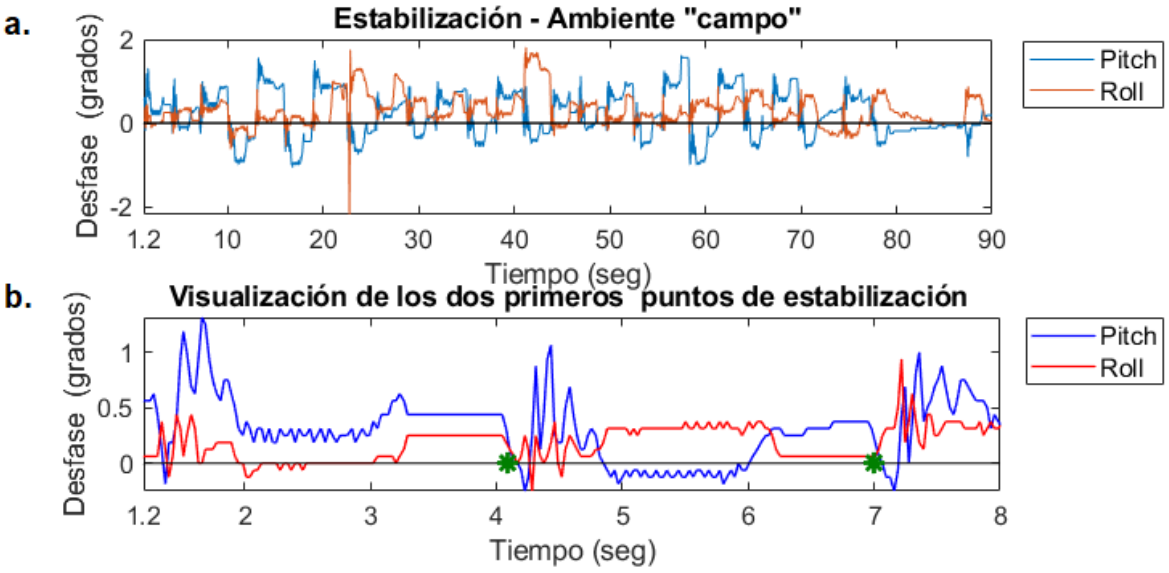



Figura 5.4: Prueba de estabilización en mesa. *Fuente propia*.

Se observa que la estabilización se lleva a cabo de manera satisfactoria con un error pequeño. Además, en la tabla 5.2 se puede ver los datos de desfase tomados al ejecutar la estabilización (los puntos verdes de la gráfica 5.4.b).

| Tiempo (sg) | Roll (grados) | Pitch (grados) |
|---|---|---|
| 4.0877 | 0.2500 | 0.4375 |
| 7.0039 | 0.3750 | 0.0625 |

Tabla 5.2: Datos de desfase después de ejecutar la estabilización. Ambiente ”Campo”. *Fuente propia*.

En condiciones más complejas, como lo son los terrenos irregulares, actúa también de buena manera y logra estabilizar el sistema completo de manera que se pueda llevar a cabo una marcha adaptativa con un buen equilibrio del robot.

## 5.3. Giro

Es importante para un robot móvil poder realizar cambios de dirección y que estos sean lo más exactos posibles respecto de la orden a realizar. En la figura 5.5 se muestran los resultados obtenidos en una secuencia de giros realizados por el robot. La prueba se realizó en el ambiente ”Mesa”, no hubo desplazamientos con marchas, solo se realizó una secuencia de giros.

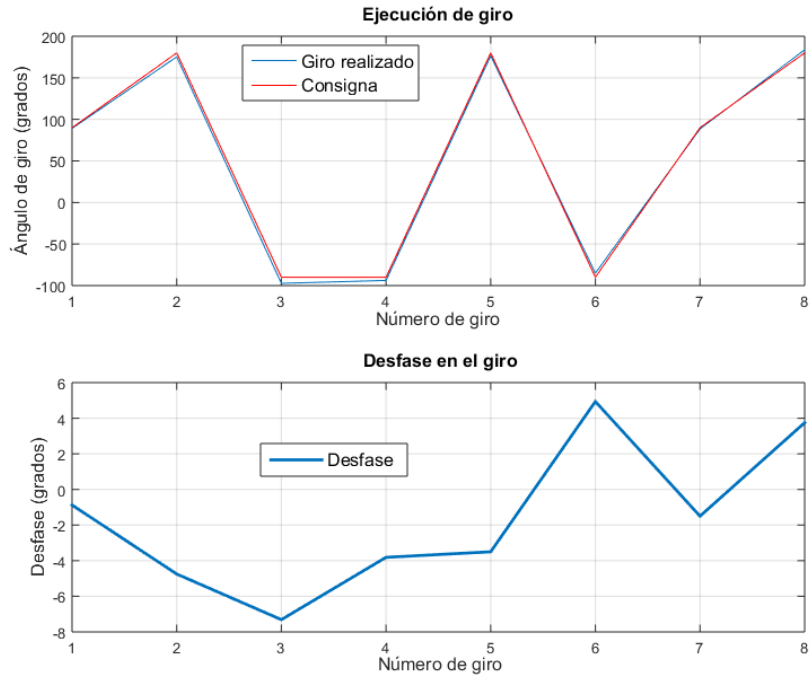



Figura 5.5: Desfase en el giro. *Fuente propia*.

Se evidencia que el desfase entre la orden enviada y la acción realizada por el robot es pequeño. A pesar de ello, la diferencia entre el ángulo realizado y la orden varía, por lo cual no se recomienda

trayectos mayores a 100cm después de un giro ya que se desviaría del punto final. En la tabla 5.3 se muestran los valores de desfase y el tiempo usado para cada giro.

| Orden de giro | Giro realizado | Tiempo (seg) |
|---|---|---|
| 90 | 89.125 | 12.3731 |
| 180 | 175.25 | 12.3447 |
| -90 | -97.3125 | 12.3493 |
| -90 | -93.81 | 12.4812 |
| 180 | 176.5 | 12.4946 |
| -90 | -85.06 | 12.6352 |
| 90 | 88.5 | 12.5643 |
| 180 | 183.75 | 12.5056 |

Tabla 5.3: Datos de giro. Prueba 1. *Fuente propia*.

También se realizaron pruebas de giro sobre el ambiente ”Piso en cerámica”, en el cual existieron desplazamientos después de los giros para determinar qué tanto puede afectar el desfase entre la orden de giro y el giro real al momento de moverse en el entorno. Los resultados son los mostrados en la figura 5.6.

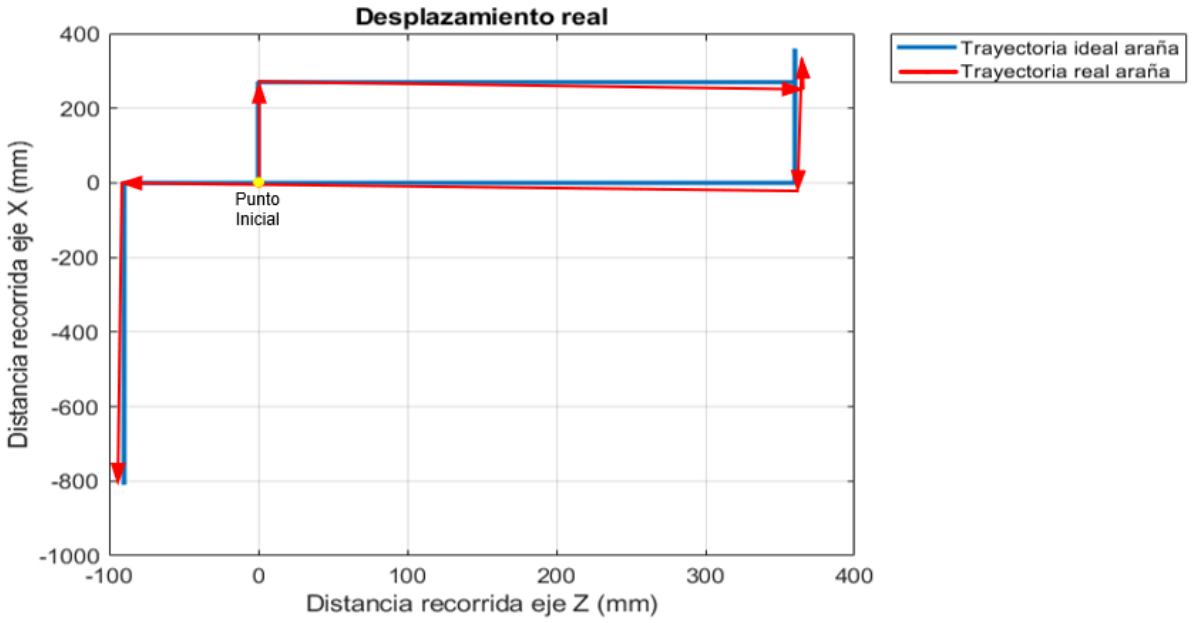



Figura 5.6: Desplazamiento sobre el espacio con presencia de giros. *Fuente propia*.

Para la prueba, los desplazamientos no fueron mayores a 500 mm y se logra observar que el desfase con respecto a la trayectoria que se debía seguir no es considerable. En la tabla 5.4 se muestran los datos obtenidos de la prueba.

| Orden de giro | Giro realizado | Tiempo (seg) |
|---|---|---|
| 90 | 90.8125 | 12.3387 |
| -90 | -93.5630 | 12.3537 |
| 180 | 175.4370 | 12.3788 |
| 90 | 88.50 | 12.3802 |
| -90 | -95.9370 | 12.5257 |

Tabla 5.4: Datos de giro. Prueba 2. *Fuente propia*.

Entonces logramos evidenciar que la implementación del giro se realizó de manera satisfactoria, se lograr llegar a un punto entre tanto no se encuentre a una distancia muy extensa. Por los resultados a partir de las pruebas realizadas se recomienda trayectos no mayores a 100 cm para que así el robot llegue a los puntos deseados.

## 5.4. Trayectorias

Las trayectorias ejecutadas por los finales de las patas fueron diseñadas con el fin de que el cuerpo del robot se desplace hacia adelante en línea recta y de forma veloz para la marcha regular y con el fin de que el robot se mantenga estable en la ejecución de la marcha adaptativa sobre terrenos irregulares.

### 5.4.1. Marcha regular

La marcha regular se caracteriza por ser una marcha rápida para desplazamiento sobre superficies totalmente planas y sin irregularidades. Las pruebas se realizaron principalmente sobre el ambiente ”Piso en cerámica”, sin irregularidades ni cambios de nivel.

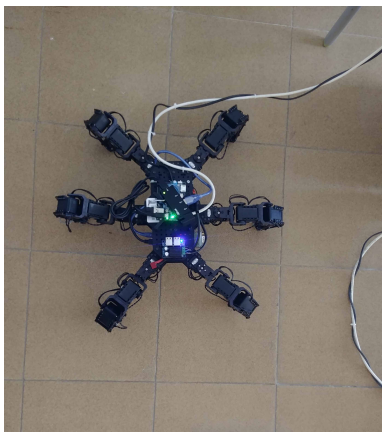


Figura 5.7: Ambiente ”Piso en cerámica”. *Fuente propia*.

Los resultados obtenidos en trayectorias fueron los indicados en la figura 5.8. Las variaciones entre real e ideal son muy pequeñas y en todas las patas se obtuvieron aproximadamente los mismos resultados.

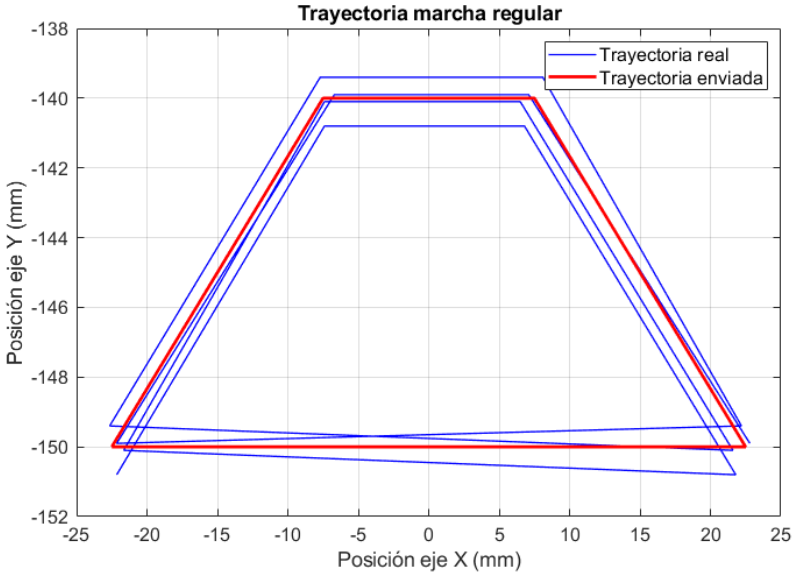



Figura 5.8: Trayectoria ideal vs. Trayectoria ejecutada. *Fuente propia*.

Podemos ver que la marcha regular seguirá de manera adecuada la trayectoria ideal para así obtener un mejor rendimiento en lo que respecta a tiempo y desplazamiento.

Además, en la tabla 5.5 se muestran los datos de 2 trayectoria consecutivas y los errores que obtuvieron con respecto a la consigna.

| | **Puntos Realizados** | |
|---|---|---|
| **Consigna [X (mm) Y (mm)]** | **Ciclo 1 [X (mm) Y (mm)]** | **Ciclo 2 [X (mm) Y (mm)]** |
| [-22.5 -150] | [-23.35 -149.8] | [-23.25 -150.7] |
| [-7.50 -140] | [-8.50 -140.5] | [-6.60 -139.8] |
| [7.50 -140] | [8.00 -140.5] | [8.30 -139.8] |
| [22.5 -150] | [23.2 -149.8] | [22.1 -150.5] |

Tabla 5.5: Datos obtenidos en ejecución de la marcha regular. *Fuente propia*.

### 5.4.2. Marcha adaptativa

La marcha adaptativa fue desarrollada para lograr una caminata estable sobre terrenos con perturbaciones de nivel. Se realizaron varias pruebas sobre terrenos planos con alteraciones de nivel controladas y otras sobre terreno irregular real. La trayectoria base que deben seguir todas las

patas en cualquier terreno es la indicada en la figura 5.9, la cual fue obtenida en pruebas sencillas sobre un terreno totalmente plano.

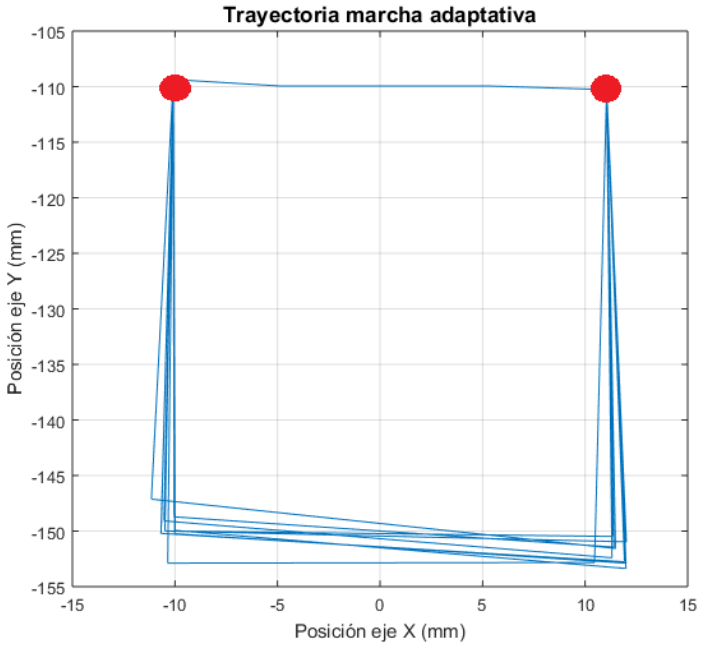



Figura 5.9: Trayectoria guía obtenida en pruebas. *Fuente propia*.

La consigna de la trayectoria es la mostrada en la figura 4.15, doóde se muestra la trayectoria guía de la marcha adaptativa. Los puntos superiores marcados en rojo se los puede denominar como puntos de consigna, ya que son los puntos fijos en la trayectoria, los puntos inferiores de la trayectoria son distintos para cada ejecución de paso debido a muchos factores (terreno, fricción, influencia del movimiento de las otras patas).

La prueba fue realizada sobre el ambiente ”Mesa”, el cual es totalmente plano y las únicas perturbaciones existentes son las generadas por la acción de las patas o su rozamiento con la superficie. En condiciones ideales de terreno (sin grandes perturbaciones) se obtiene una trayectoria afín a la que sería una trayectoria ideal de marcha adaptativa.

Se realizaron distintas pruebas sobre los 3 ambientes de trabajo. La marcha adaptativa se ejecutó sobre el ambiente ”Mesa” en condiciones de terreno regular con variaciones de nivel controladas, las cuales fueron obstáculos de superficie plana montados sobre el terreno como se puede evidenciar en la figura 5.10. Además, se hace la comparación de la misma prueba con y sin implementación de la estabilización.

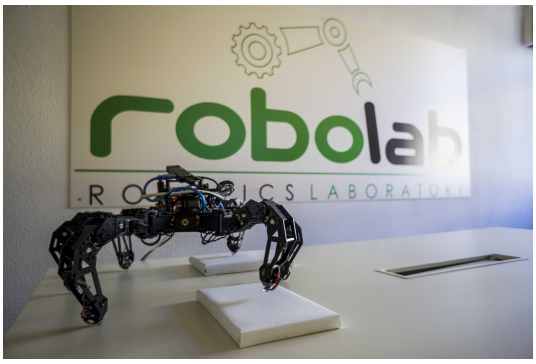



Figura 5.10: Ambiente pruebas en Mesa. *Fuente propia*.

Los valores ideales de la trayectoria están dados para que en la fase de arrastre la pata llegue a los -150 mm en el eje Y y a los -110 mm en la fase de oscilación sobre el mismo eje. Si el arrastre está por debajo de -150 mm significa que hubo un hueco al cual la pata pudo llegar y realizar un arrastre. Si el arrastre se realiza por encima de -150 mm significa que existe un obstáculo que sobresale del suelo.

En la figura 5.11 se logra observar las trayectorias que las 6 patas desarrollaron en la prueba sin presencia de estabilización.

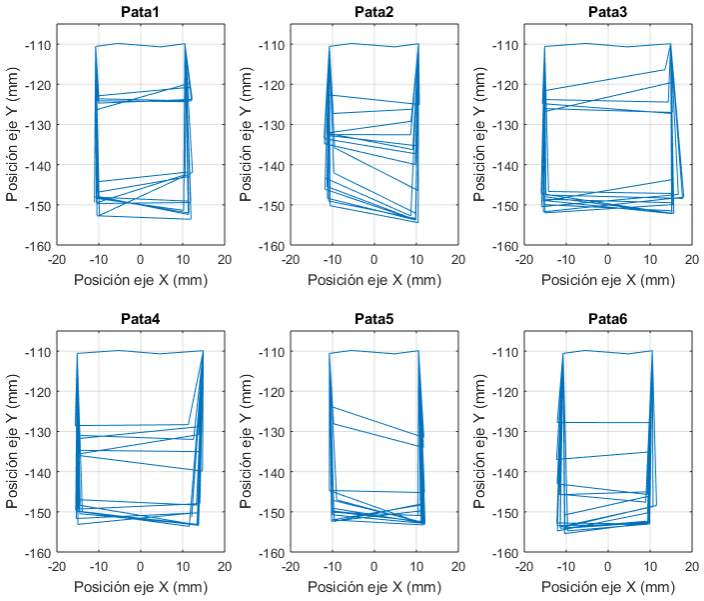



Figura 5.11: Trayectorias realizadas por las patas sin estabilización. Ambiente Mesa. *Fuente propia*.

Posteriormente se llevó a cabo la misma prueba con presencia de estabilización, las trayectorias de las extremidades se pueden ver en la figura 5.12.

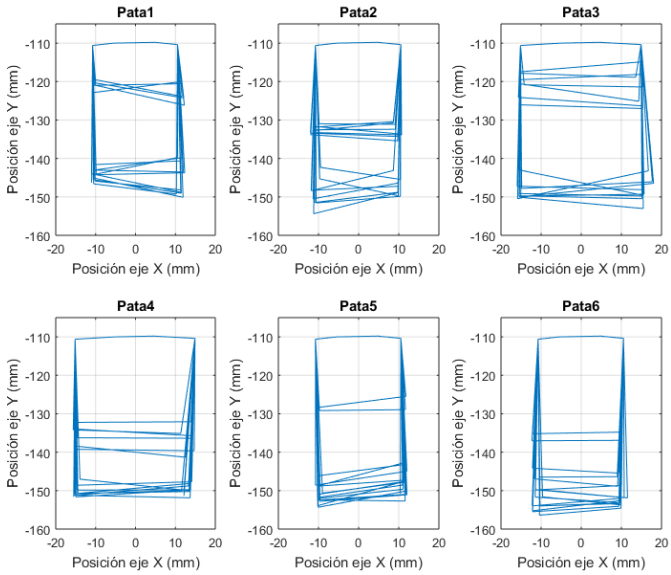



Figura 5.12: Trayectorias realizadas por las patas con estabilización. Ambiente Mesa. *Fuente propia*.

A partir de los gráficos se puede evidenciar que la ejecución del sistema de estabilización hace que el ciclo de arrastre sea más suave y así evita que la acción de una pata afecte de gran manera a las demás.

Para verlo de manera más clara, en la figura 5.13, se muestran las trayectorias ejecutadas por una de las patas laterales de la araña. La línea en color verde es la realizada por la pata sin proceso de estabilización y la roja con la presencia de estabilización.

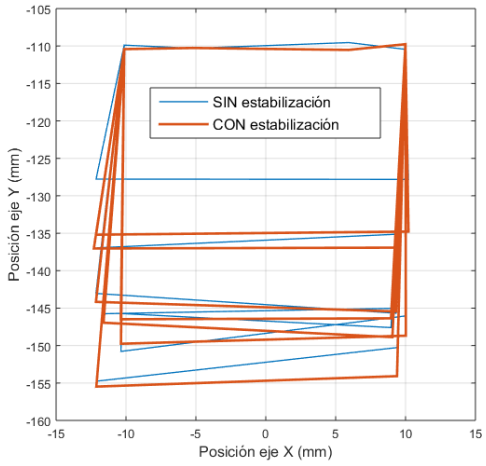



Figura 5.13: Comparación de trayectorias de una pata en una esquina del cuerpo. Ambiente Mesa. *Fuente propia*.

En la comparación de la figura anterior se exhibe el requerimiento de la estabilización, ya que, de

no ser implementado, los pasos de las extremidades tendrían repercusión en las demás, el sistema permanecería en carencia de equilibrio y así no se podría llevar a ambientes más complejos.

Sobre el mismo ambiente ”Mesa” se realizaron otro tipo de pruebas enfocadas en que la araña no solo pueda marchar por obstáculos que sobrepasen el suelo, sino además poder desplazarse en situaciones donde existan depresiones de suelo.

En la figura 5.14 se puede ver el ambiente en el cual se desarrolló la prueba, mostrando que existe una depresión de suelo en el centro de la mesa.

Figura 5.14: Ambiente de pruebas de concavidad. *Fuente propia*.

La prueba se realizó en presencia de una concavidad insinuada por la cual una de las patas pasa y reacciona con la superficie de ese desnivel controlado.

En la figura 5.15 se muestra la trayectoria realizada por cada una de las patas en la realización de la prueba con ausencia de estabilización.

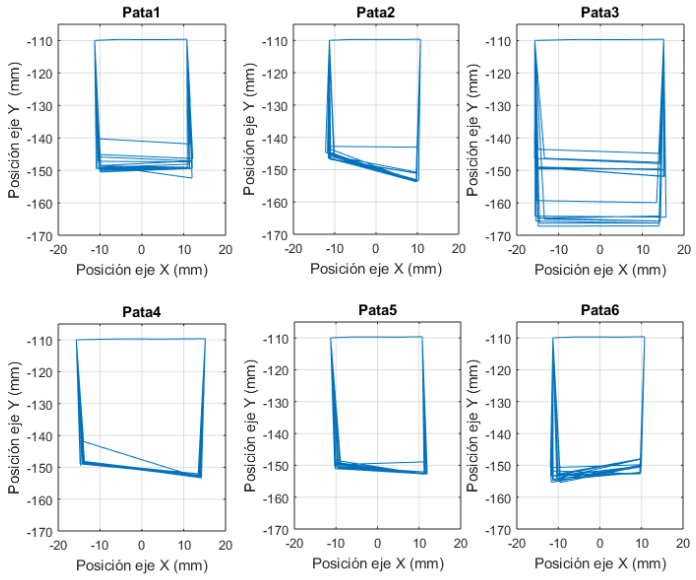



Figura 5.15: Trayectorias realizadas por las patas sin estabilización. Ambiente Mesa. Prueba hueco. *Fuente propia*.

La prueba se llevó a cabo de manera satisfactoria pero al final se notó una inclinación insinuada de la araña.

Se llevó a cabo el mismo experimento de la araña con presencia de estabilización, las trayectorias obtenidas se muestran en la figura 5.16.

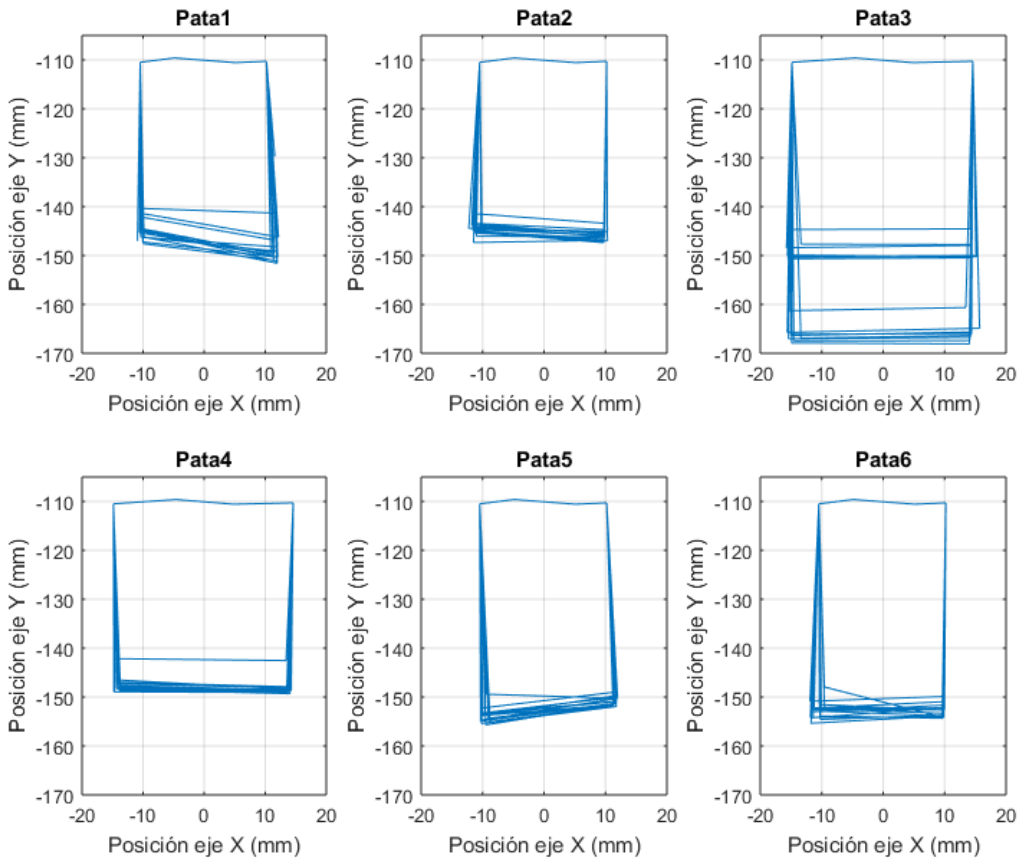



Figura 5.16: Trayectorias realizadas por las patas con estabilización. Ambiente Mesa. Prueba hueco. *Fuente propia*.

De igual manera que en la prueba de obstáculos, se puede ver que la estabilización es requerida y hace que el punto inicial de la fase de arrastre sea muy similar al punto final de la misma fase, logrando así evitar grandes repercusiones sobre los movimientos de las otras patas y sobre el sistema completo.

Al observar la figura 5.17 se puede evidenciar de manera más clara el cambio que genera la estabilización. La diferencia de altura entre los puntos de arrastre se debe a que el sistema viene de un desequilibrio que repercute en todas las patas, por lo que el movimiento de las mismas cambiará el punto final de arrastre de las demás.

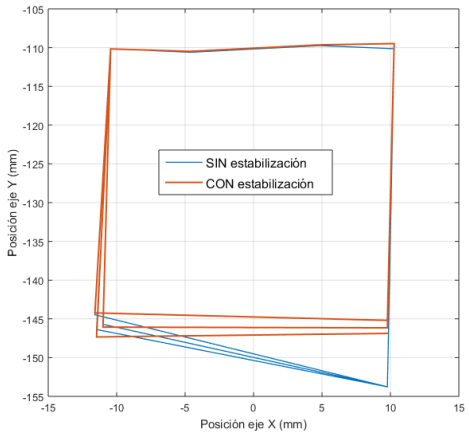



Figura 5.17: Comparación de trayectorias de una pata lateral. Ambiente Mesa. Prueba hueco. *Fuente propia*.

Se graficó también la trayectoria realizada por la pata implicada en el paso por la concavidad. Aquí podemos notar que cuando la variación entre los puntos de inicio y fin del arrastre es baja, la presencia de estabilización los vuelve casi idénticos. Vemos la comparación en la figura 5.18.

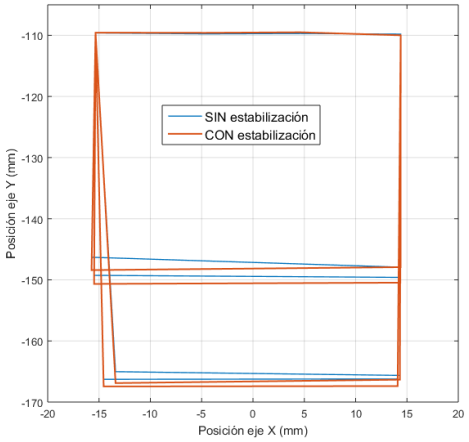



Figura 5.18: Comparación de trayectorias de una pata en esquina. Ambiente Mesa. Prueba hueco. *Fuente propia*.

Todas las pruebas anteriores fueron llevadas a cabo con el fin de verificar el funcionamiento de la marcha adaptativa sobre condiciones de ambientes planos con desniveles controlados, y basados en ello poder llevar la araña a un ambiente irregular real y que su locomoción se lleve de manera satisfactoria y sin problemas.

De los anteriores experimentos se obtuvieron datos que nos llevaron a decidir que el sistema estaba listo para interactuar con un terreno real con irregularidades no previstas, por lo cual se llevaron a cabo pruebas sobre el ambiente ”Campo”, el cual se logra ver en la figura 5.19.

Figura 5.19: Ambiente de pruebas en terreno irregular real. *Fuente propia*.

En primera instancia se llevó a cabo la prueba sin estabilización, las trayectorias realizadas por las patas son las indicadas en la figura 5.20.

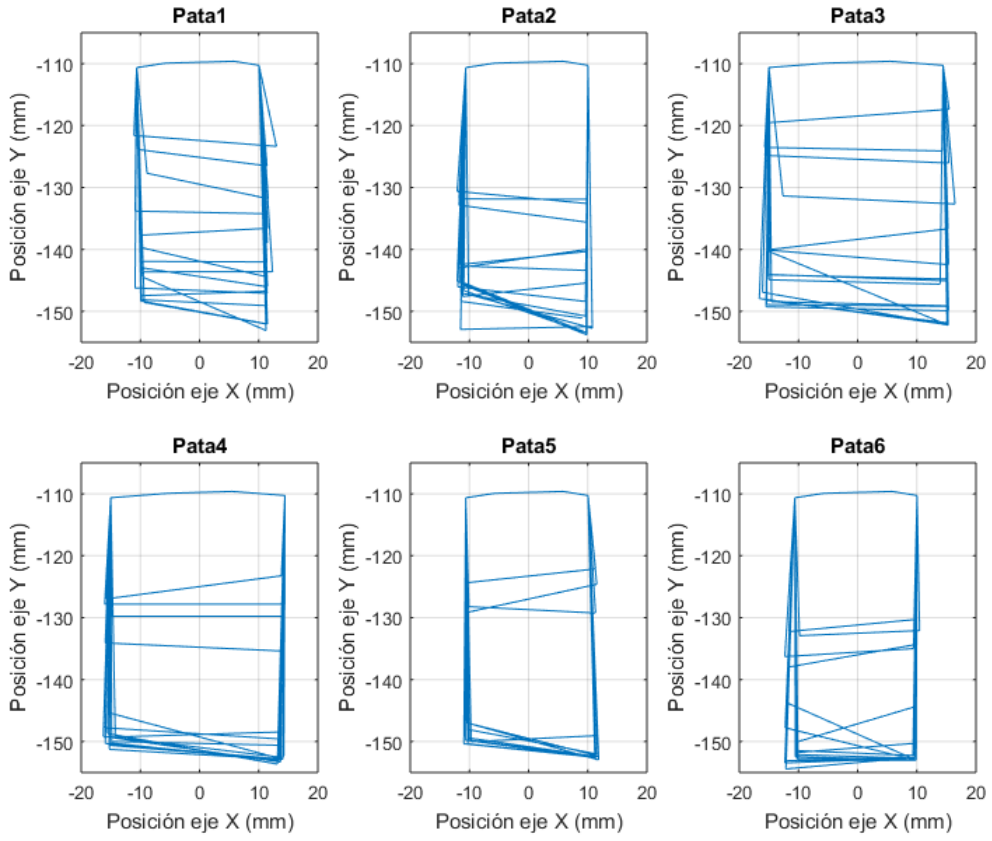



Figura 5.20: Trayectorias realizadas por las patas sin estabilización. Ambiente Campo. *Fuente propia*.

Se puede ver que las transiciones por la fase de arrastre son difíciles y en ocasiones la diferen-

cia entre el punto inicial y el final de la fase de arrastre es muy grande, generando así mayor desequilibrio del sistema hasta el punto de llevarlo al colapso y no poder continuar con la marcha.

Se llevó a cabo la misma prueba con ejecución de estabilización. Las trayectorias de tal prueba se muestran en la figura 5.21.

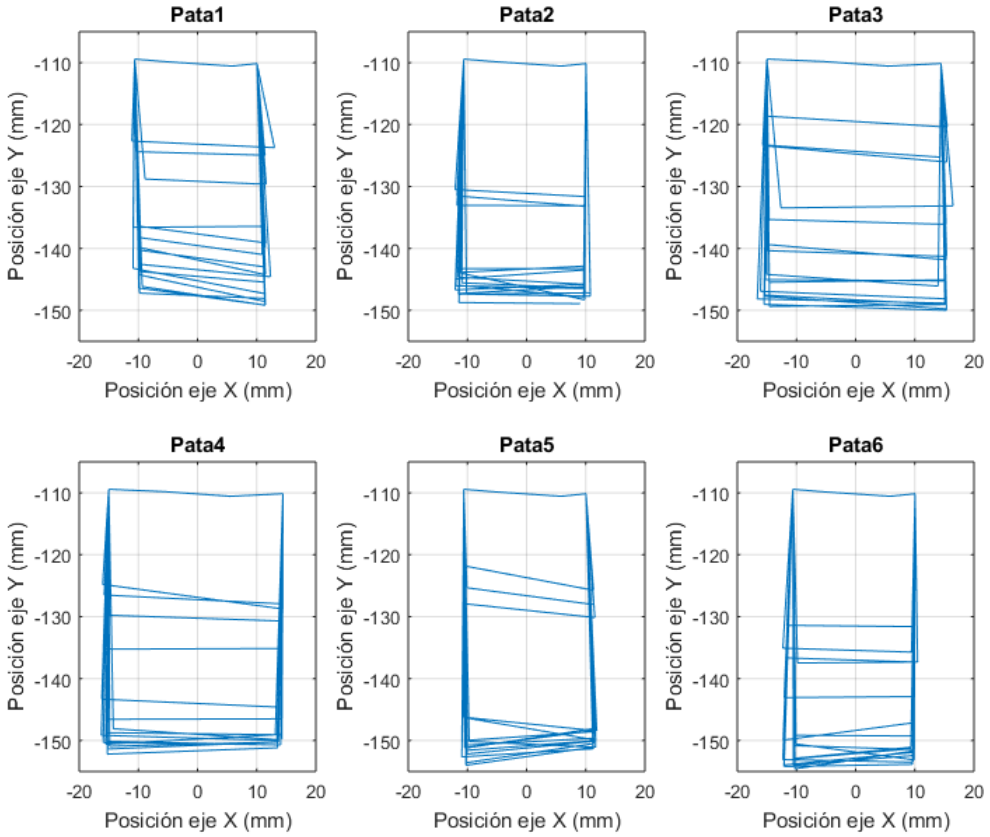



Figura 5.21: Trayectorias realizadas por las patas con estabilización. Ambiente Campo. *Fuente propia*.

Evidentemente se puede observar que las transiciones en la fase de arrastre son mucho más suaves que si no hubiera estabilización, evitan el efecto en las demás patas y mantiene el sistema estable. Se realizó la prueba por completo y sin problema alguno.

En las figuras 5.23 y 5.22 se realiza una comparación entre los resultados de la marcha en ausencia y presencia de estabilización. En la primera son las trayectorias ejecutadas por una pata lateral del cuerpo y la segunda por una pata ubicada en esquina del cuerpo.

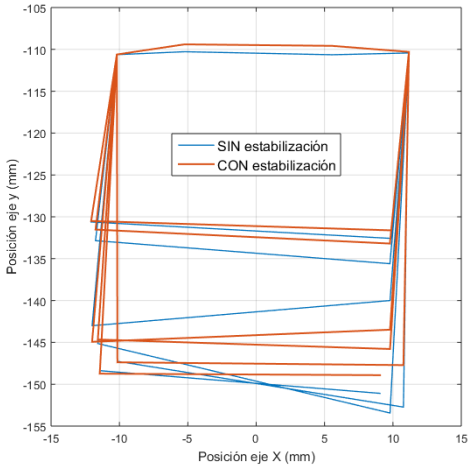



Figura 5.22: Comparación de trayectorias de una pata en esquina del cuerpo. Ambiente Campo. *Fuente propia*.

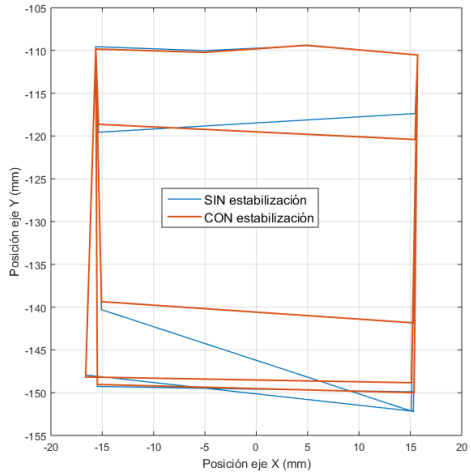



Figura 5.23: Comparación de trayectorias de una pata en lateral del cuerpo. Ambiente Campo. *Fuente propia*.

En ambos casos la aplicación de estabilidad mejoró su respuesta ante irregularidades y se muestra claramente en las figuras que la trayectoria con estabilización es mucho más suave que cuando no la tiene.

## 5.5. Posición, velocidad y tiempo

Se realizaron dos pruebas para determinar la velocidad de la marcha regular frente a la marcha adaptativa. Cada prueba se realizó en igualdad de condiciones de terreno y mismo número de

ciclos de marcha para cada trípode. Para la selección del terreno donde se llevaría acabo las pruebas, se consideró que la marcha regular no puede sobrepasar obstáculos y su funcionalidad en un terreno irregular no sería adecuada debido a los desniveles, basado en ello se precisa que el terreno sea uniforme. En la selección del número de ciclos se tuvo en cuenta que en un ciclo de la marcha regular se avanza un aproximado de 90 milímetros y en la marcha adaptativa 60 milímetros. Con los datos presentados anteriormente, se define que 18 es el número de ciclos a realizar en cada prueba, obteniendo una trayectoria estimada cercana a un metro en cada modo de marcha.

- **Prueba 1:** Realizada por la araña en los dos modos de marcha (regular y adaptativa) sobre el ambiente ”Mesa” en ausencia de obstáculos.

- **Prueba 2:** Realizada por la araña en los dos modos de marcha (regular y adaptativa) sobre el ambiente ”Piso en cerámica” en ausencia de obstáculos.

A continuación se presentan los resultados de tiempo y distancia recorrida en cada una de las pruebas, donde se estima la velocidad en los dos modos de marcha, en los cuales se asigna como punto de partida (1.172(tiempo); 0(distancia)) siendo 1.172 el promedio que tarda el robot en situarse en la posición inicial.

Los datos obtenidos en las dos pruebas se muestran en las figuras 5.24 y 5.25.

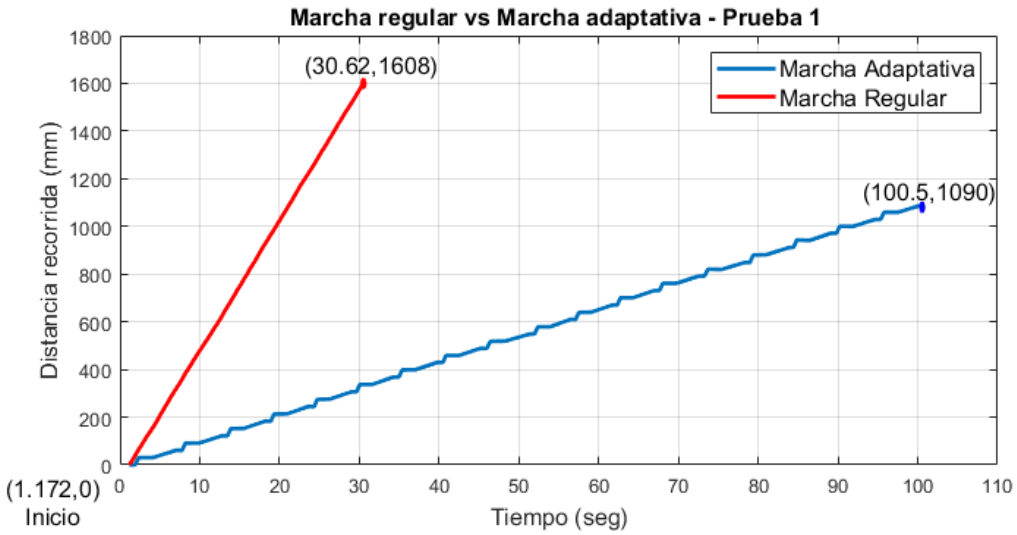



Figura 5.24: Comparación de desplazamiento en las marchas. Prueba 1. *Fuente propia*.

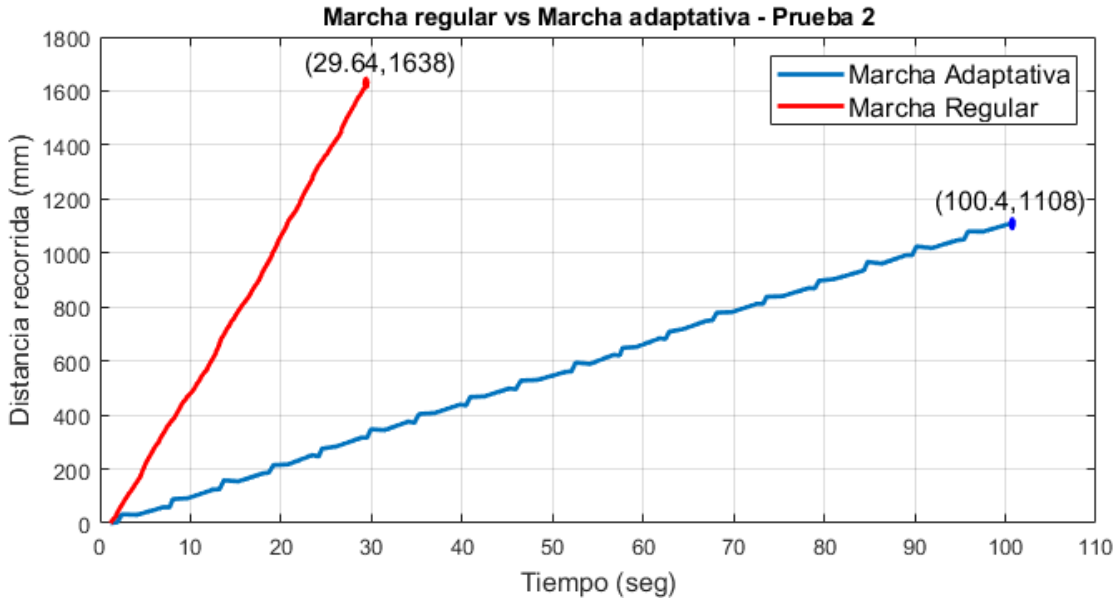



Figura 5.25: Comparación de desplazamiento en las marchas. Prueba 2. *Fuente propia*.

Los datos obtenidos de la velocidad de las marchas se muestran en la tabla 5.6.

| | **Velocidad (cm/sg)** | |
|---|---|---|
| | **Marcha Regular** | **Marcha Adaptativa** |
| **Prueba 1** | 5.4604 | 1.0973 |
| **Prueba 2** | 5.7538 | 1.1166 |

Tabla 5.6: Comparación de velocidad entre marchas. *Fuente propia*.

Se puede concluir que en la prueba 2 se obtuvo una mayor velocidad respecto a la prueba 1. Además, la marcha regular es mucho más rápida con respecto a la adaptativa, eso se debe a que la adaptativa toma un tiempo en buscar el punto de apoyo y su trayectoria en el eje Y es 30 mm mayor que la regular. Adicionalmente, la trayectoria de la marcha adaptativa en el eje X es 15 mm menor.

En la figura 5.26 se explica porqué en las pruebas de la marcha adaptativa en el recorrido se ve un rizo cíclico. Se debe a que la fase de oscilación se divide en tres partes, de las cuales dos no se transforman en distancia recorrida sobre el eje X dado que estas se ven reflejadas en el eje Y.

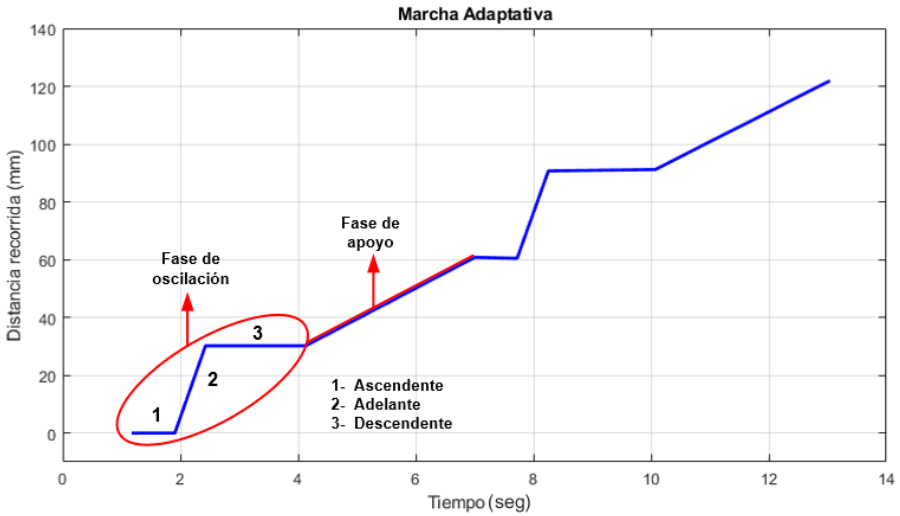



Figura 5.26: Etapas de la marcha adaptativa en el desplazamiento. *Fuente propia*.

En las diversas pruebas realizadas sobre el robot hexápodo, se presentaron diferencias entre los puntos de trayectoria enviados y los puntos de trayectoria obtenidos durante el desplazamiento del robot. A continuación, se indican los errores de posición que se presentaron en la marcha regular en la figura 5.27 y la marcha adaptativa en la figura 5.28.

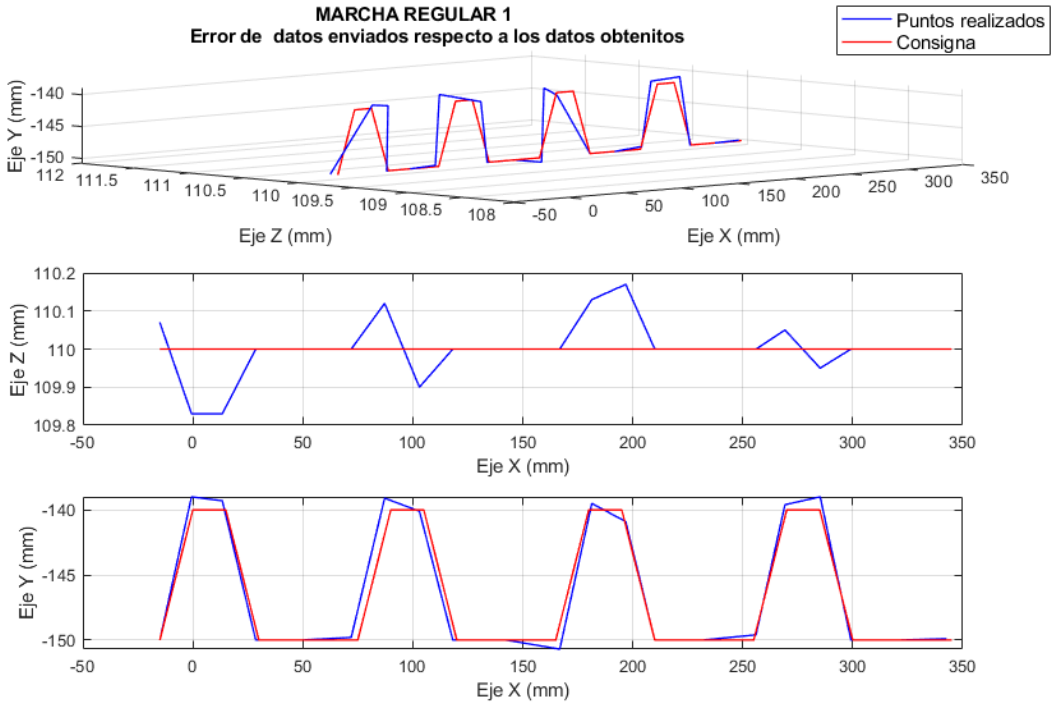



Figura 5.27: Error en las trayectorias de la marcha regular. *Fuente propia*.

Además, en la tabla 5.7 se muestra el primer ciclo ejecutado por la marcha regular.

| **Consigna [X(mm) Y(mm) Z(mm)]** | **Puntos Realizados [X(mm) Y(mm) Z(mm)]** |
|---|---|
| [-15 -150 110] | [-15.0 -150.0 110.1] |
| [0.0 -140 110] | [-0.6 -139.0 109.8] |
| [15 -140 110] | [13.4 -139.3 109.8] |
| [30 -150 110] | [28.6 -150.0 110.0] |

Tabla 5.7: Datos del primer ciclo de marcha regular. *Fuente propia*.

Se observa que el error en la marcha regular es mínimo teniendo en cuenta los puntos sobre los tres ejes, presentando un error pequeño en los puntos obtenidos sobre el eje Z, el cual afecta a que se lleve a cabo la trayectoria ideal en el espacio tridimensional.

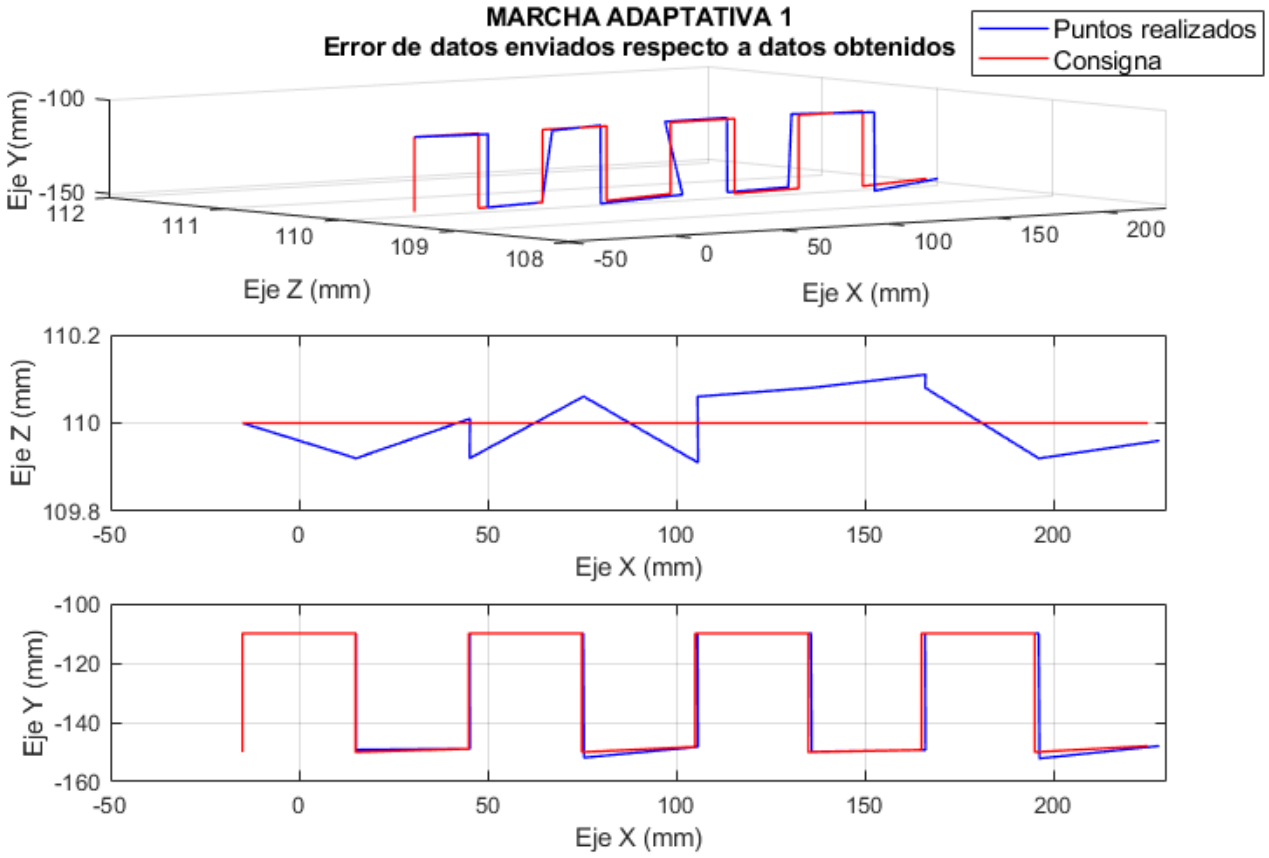



Figura 5.28: Error en las trayectorias de la marcha adaptativa. *Fuente propia*.

Además, en la tabla 5.8 se muestra el primer ciclo ejecutado por la marcha regular.

| Consigna [X(mm) Y(mm) Z(mm)] | Puntos Realizados [X(mm) Y(mm) Z(mm)] |
|---|---|
| [-15 -150 110] | [-15 -150 110] |
| [-15 -110 110] | [-15 -110 110] |
| [15 -110 110] | [15.12 -110 109.9] |
| [15 -150 110] | [15.12 -149.2 109.9] |

Tabla 5.8: Datos del primer ciclo de marcha adaptativa. *Fuente propia*.

En las diversas pruebas sobre marcha adaptativa aplicando la estabilización, se observa que el seguimiento de los puntos de la trayectoria son mayormente fieles a la trayectoria ideal con respecto a cuando no hay estabilización, presentando mejor seguimiento que la marcha regular.

En la figura 5.29 se observa la comparativa de distancia recorrida y forma de las fases de oscilación y apoyo con respecto al tiempo. Se puede evidenciar que el terreno es totalmente plano. Dado que estos datos se recolectaron sobre dicho terreno se obvió la ejecución de estabilización. Además, se observa que en la marcha adaptativa se presenta una pequeña curvatura durante todo el trayecto debido a los errores que se presentan al ejecutar los puntos reales sobre el eje X y eje Y.

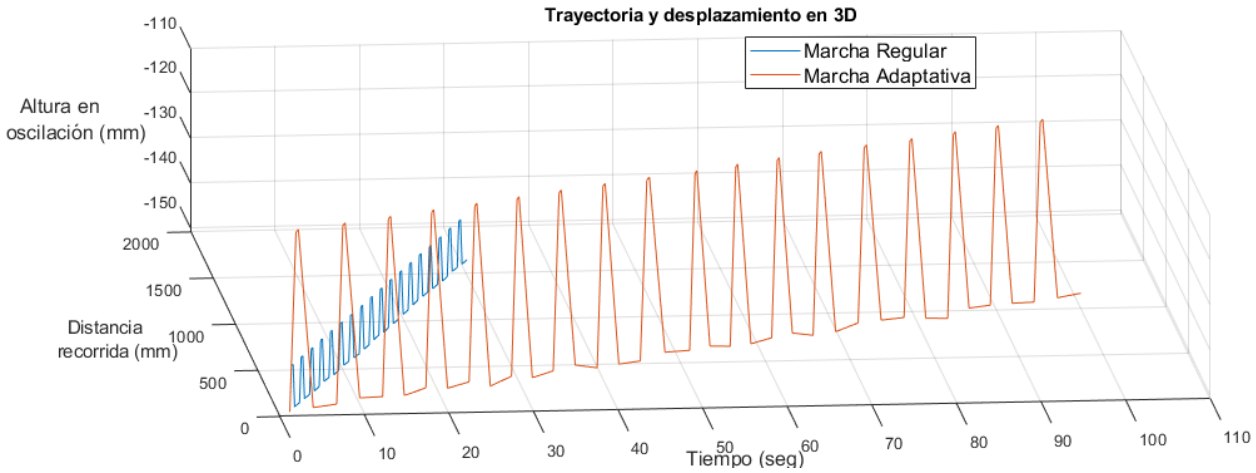



Figura 5.29: Ejecución de trayectorias y dezplazamiento representado en el espacio tridimensional. *Fuente propia*.

Se puede concluir que, para la misma cantidad de ciclos ejecutados en cada marcha, a la marcha regular le toma menos tiempo realizarlos, obteniendo así una mayor distancia recorrida con respecto a la marcha adaptativa.

Los resultados obtenidos en la gráfica 5.29 son los esperados, dado que en el momento de llevar a cabo las pruebas se evidenció que la marcha regular era considerablemente más veloz que la marcha adaptativa, teniendo en cuenta que la trayectoria de los finales de las patas son más pequeños en la marcha regular.

## 5.6. Búsqueda de puntos de apoyo

La búsqueda de puntos de apoyo se hace importante a la hora de llevar al robot a condiciones de terrenos extremos, en los cuales el robot no pueda continuar con su locomoción hacia el frente debido a no encontrar puntos de apoyo (que seguramente se relaciona con un desnivel muy grande) y así requerir una locomoción hacia atrás.

En la figura 5.30 se muestra el espacio en el cual pueden estar los posibles puntos de apoyo sin salir del espacio de trabajo de las patas y la ubicación a donde llegó a buscar el punto.

Los puntos verdes ubicados en el centro de las circunferencias, refieren a los lugares donde quedó la pata en el instante en que no encontró punto de apoyo mientras se ejecutaba la marcha. Los puntos azules son de búsqueda pero sin apoyo y el punto morado representa el punto en el que encontró apoyo.

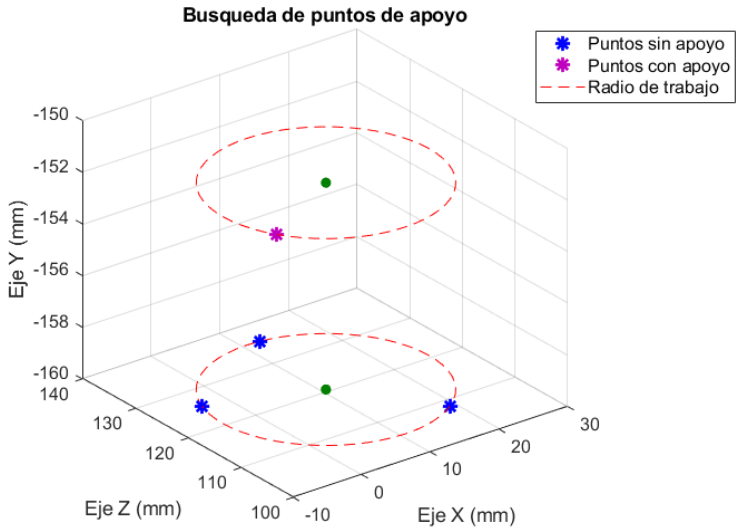



Figura 5.30: Búsqueda de puntos representados en espacio 3D. *Fuente propia.*

Cuando una pata no encuentra punto de apoyo en una locomoción normal, esta procede a buscar un nuevo punto de apoyo y realiza 4 intentos en puntos aleatorios dentro de ese espacio, al encontrar un punto continúa con la marcha hacia adelante y si no logra encontrar ninguno, realiza una marcha hacia atrás. En la figura 5.31 se muestra el recorrido de la pata para buscar el punto de apoyo.

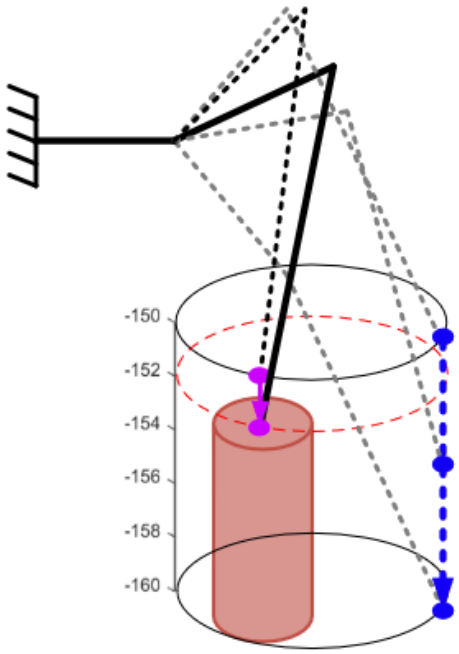



Figura 5.31: Búsqueda de puntos representados en espacio 3D. *Fuente propia*.

Se observa la realización de dos ciclos de búsqueda de un nuevo punto de apoyo. El primer ciclo representado en color gris, se muestra la búsqueda de un punto de apoyo en una coordenada aleatoria del espacio de trabajo (representado con la circunferencia superior). El final de la pata se movió desde la parte superior hasta un centímetro por debajo realizando pasos de un milímetro sin encontrar un apoyo. En el segundo ciclo representado de color negro, se repitió la misma secuencia en una nueva coordenada aleatoria obteniendo un punto de apoyo dos milímetros por debajo de la base, así poder continuar con la locomoción hacia adelante.

# Capítulo 6

# Conclusiones y trabajos futuros

## 6.1. Conclusiones

Mediante el proyecto se realizó la implementación de un algoritmo de control para ejecución de marcha adaptativa en el robot comercial PhantomX, haciendo uso del framework RoboComp. Se implementaron dos tipos de marcha: marcha regular y marcha adaptativa. La marcha regular mostró buen desempeño en terrenos planos y, debido a que tiene una trayectoria fija en los finales de las extremidades, no puede actuar ante irregularidades en el terreno.

La marcha adaptativa por su parte, tiene un buen desempeño en los ambientes de laboratorio en cuanto a adaptarse ante protuberancias o concavidades pronunciadas y controladas que el terreno presente. Además, con la marcha adaptativa en ejecución se llevó el robot PhantomX a un terreno irregular real de rocas pequeñas, donde realizó locomoción de buena manera. El sistema de planificación de trayectorias junto con los sistemas adicionales que se implementaron (giro, estabilización y búsqueda de puntos de apoyo) lograron que los finales de las patas del robot se adaptaran a las irregularidades del terreno, así cada paso logra obtener un nuevo punto de apoyo para realizar el arrastre, estabilizar el sistema en dos puntos de soporte del mismo cicle (punto inicial y punto final) y realizar giros para que el robot marche hacia un punto requerido.

El robot comercial se obtiene con una configuración estándar, a la cual se le realizaron modificaciones junto a otros equipos de trabajo para lograr que todos los sistemas planteados puedan funcionar. Se instaló una CPU más potente (Odroid XU4) que la tarjeta que viene en el robot, junto con un adaptador USB2 Dynamixel para comunicación de la nueva CPU con el bus de comunicación de los motores. También se implementaron dos tipos de sensores: un sensor de presión ubicado en el final de cada una de las patas para detectar el momento en que estas encuentran un punto de apoyo y un sensor de medida inercial (IMU) para obtener datos de los ángulos de rotación del robot, los cuales se traducen en desfase del sistema con respecto al punto estable.

El framework de robótica RoboComp ofrece las bases necesarias en cuanto a simulación, comunicación e implementación del algoritmo en el robot. Sobre este se realizó el algoritmo de programación y permitió la abstracción de este al robot real. También, se realizaron simulaciones sobre RCIS (simulador propio de RoboComp) previas a la implementación y ejecución de los tipos de marcha. Para las órdenes de giro y cambios entre los tipos de marcha se hizo uso de un joystick.

## 6.2. Trabajos futuros

- La instalación de una cámara RGB-D y desarrollo de un sistema de visión artificial con el fin de que la marcha adaptativa pronostique y evalúe el terreno y así definir los mejores puntos de apoyo.
- Implementación de nuevos modos de marcha dado que existe gran variedad de ellos que se pueden adaptar al PhantomX.
- Desarrollo de una interfaz gráfica que permita al usuario dirigir la dirección de locomoción, el tipo de marcha y su velocidad de forma remota.

# Bibliografía


[1] M. Rojas, N. Certad, J. Cappelletto, and J. C. Grieco, “Foothold planning and gait generation for a hexapod robot traversing terrains with forbidden zones,” in *12th Latin American Robotics Symposium and 3rd Brazilian Symposium on Robotics (LARS-SBR)*, Uberlândia, Brasil, 2015, pp. 49–54. 2015.

[2] J. Chen, M. Gauci, W. Li, A. Kolling, and R. Groß, “Occlusion-based cooperative transport with a swarm of miniature mobile robots,” *IEEE Transactions on Robotics*, vol. 31, no. 2, pp. 307–321, 2015.

[3] L. Ding, H. Gao, Z. Deng, K. Nagatani, and K. Yoshida, “Experimental study and analysis on driving wheels’ performance for planetary exploration rovers moving in deformable soil,” *Journal of Terramechanics*, vol. 48, no. 1, pp. 27 – 45, 2011.

[4] R. Murphy, J. Kravitz, S. Stover, and R. Shoureshi, “Mobile robots in mine rescue and recovery,” *IEEE Robotics Automation Magazine*, vol. 16, no. 2, pp. 91–103, 2009.

[5] K. Nagatani, S. Kiribayashi, Y. Okada, K. Otake, K. Yoshida, S. Tadokoro, T. Nishimura, T. Yoshida, E. Koyanagi, M. Fukushima, and S. Kawatsuma, “Emergency response to the nuclear accident at the fukushima daiichi nuclear power plants using mobile rescue robots,” *Journal of Field Robotics*, vol. 30, no. 1, pp. 44–63, 2013. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21439

[6] M. Shneier and R. Bostelman, “Literature review of mobile robots for manufacturing,” *NISTIR-8022: National Institute of Standards and Technology, US Department of Commerce*, Gaithersburg, USA. 2015.

[7] J. Yang and J. Kim, “A fault tolerant gait for a hexapod robot over uneven terrain,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 30, no. 1, pp. 172–180, 2000.

[8] L. Bruzzone and G. Quaglia, “Locomotion systems for ground mobile robots in unstructured environments,” *Mechanical Sciences*, vol. 3, no. 2, pp. 49–62, 2012.

[9] A. Sanchez, H. Reyes, and E. Vargas, “Diseño de un robot caminante hexápodo tipo hormiga,” in *4to Congreso Nacional de Mecatrónica*, Coahuila, México, 2005.

[10] M. Cárdenas, "Diseño, construcción y control de un robot hexápodo," Ph.D. dissertation, Tesis de grado, Universidad Nacional Autónoma de México, México DF, 2011.

[11] Y. Zhao, X. Chai, F. Gao, and C. Qi, "Obstacle avoidance and motion planning scheme for a hexapod robot octopus-iii," *Robotics and Autonomous Systems*, vol. 103, pp. 199 – 212, 2018.

[12] N. Nilsson, "Shakey the robot," SRI International, Menlo Park, CA, Tech. Rep., 1984.

[13] G. A. Bekey, *Autonomous robots: from biological inspiration to implementation and control*. MIT press, Cambridge, 2005.

[14] B. A. Chacón Hernández and I. N. Rodríguez Ovalle, "Desarrollo de un objeto virtual de aprendizaje para los robots bioloid con finalidad de aplicaciones en robótica," B.S. thesis, Universidad Piloto de Colombia, Bogotá, 2017.

[15] F. Fahimi, *Autonomous robots: modeling, path planning, and control*. Springer Science & Business Media, 2008, vol. 107.

[16] NASA-JetPropulsionLaboratory, "Mars science laboratory," 2017. [Online]. Available: https://mars.nasa.gov/msl/

[17] J. Borrella, "Desarrollo de un control versátil de trayectorias y modos de marcha para un robot hexápodo," B.S. thesis, Tesis de grado, Universidad Politecnica de Madrid, España, 2017.

[18] J. Tordesillas, "Diseño y simulación del sistema de locomoción de un robot hexápodo para tareas de búsqueda y rescate," Ph.D. dissertation, Universidad Politecnica de Madrid, España, 2016.

[19] W. Chung and K. Iagnemma, "Wheeled robots," in *Springer Handbook of Robotics*. Springer, Berlín, Alemania, 2016, pp. 575–594.

[20] S. Jatsun, L. Y. Vorochaeva, S. Savin, and A. Yatsun, "Study of caterpillar-like motion of a four-link robot," in *Proceedings of the 14th IFToMM World Congress*, 2015, pp. 367–372.

[21] H. Kimura, K. Tsuchiya, A. Ishiguro, and H. Witt, *Adaptive motion of animals and machines*. Springer Science & Business Media, 2006.

[22] M. H. Raibert, "Legged robots," *Communications of the ACM*, vol. 29, no. 6, pp. 499–514, 1986.

[23] D. Zugasti, "Vehículo móvil no tripulado provisto de sensores y actuadores para reconocer entornos," Master's thesis, Universidad de Alcalá, Madrid, España, 2015.

[24] T. Fischer, T. Pire, P. Čížek, P. D. Cristóforis, and J. Faigl, "Stereo vision-based localization for hexapod walking robots operating in rough terrains," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2492–2497. Daejeon, Corea. 2016.

[25] M. Hörger, N. Kottege, T. Bandyopadhyay, A. Elfes, and P. Moghadam, “Real-time stabilisation for hexapod robots,” in *Experimental Robotics*. Springer, Beijing, 2016, pp. 729–744.

[26] P. de Santos, E. Garcia, R. Ponticelli, and M. Armada, “Minimizing energy consumption in hexapod robots,” *Advanced Robotics*, vol. 23, no. 6, pp. 681–704, 2009.

[27] J. Fernández, “Diseño y construcción de un robot hexápodo: Software y hardware de control,” Master’s thesis, E.T.S.I.Informática - Universidad de Málaga, España, 1994.

[28] X. Zhou and S. Bi, “A survey of bio-inspired compliant legged robot designs,” *Bioinspiration & biomimetics*, vol. 7, no. 4, pp. 41 – 61, 2012.

[29] B. Siciliano and O. Khatib, *Springer handbook of robotics*. Springer, Beijing, 2016.

[30] N. Neville and M. Buehler, “Towards bipedal running of a six-legged robot,” Centre for intelligent machines, Montreal, Tech. Rep., 2003.

[31] U. Saranli, M. Buehler, and D. E. Koditschek, “Rhex: A simple and highly mobile hexapod robot,” *The International Journal of Robotics Research*, vol. 20, no. 7, pp. 616–631, 2001.

[32] R. Cebolla, “Sistema de control para robot hexápodo de exploración,” B.S. thesis, Universidad Politecnica de Madrid, España, 2017.

[33] J. Mateos, A. Sánchez, L. Manso, P. Bachiller, and P. Bustos, “RobEx: an open-hardware robotics platform,” in *Journal of Physical Agents*, Valencia, España, 2010. [Online]. Available: http://www.jopha.net/waf/index.php/waf/waf10/paper/viewPDFInterstitial/69/55

[34] L. Manso, P. Bachiller, P. Bustos, P. Núñez, R. Cintas, and L. Calderita, “Robocomp: a tool-based robotics framework,” in *Lecture Notes in Computer Science. Simulation, Modeling and Programming in Autonomous RobotsRobots*. Darmstadt, Alemania: Springer, 2010, vol. 6472.

[35] R. Cintas, L. Calderita, L. Manso, P. Bustos, P. Bachiller, and P. Núñez, “Un framework de Desarrollo para Robótica,” in *I Jornadas Jóvenes Investigadores Universidad de Extremadura*, Caceres, España, 2010.

[36] M. Henning, “A new approach to object-oriented middleware,” *IEEE Internet Computing*, vol. 8, no. 1, pp. 66–75, 2004.

[37] K. Sharan, “Java remote method invocation,” in *Java APIs, Extensions and Libraries*. Apress, Montgomery, 2018, pp. 489–513.

[38] J. Martínez, A. Romero, L. Manso, and P. Bustos, “Improving a robotics framework with real-time and high-performance features,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6472 LNAI, pp. 263–274, 2010.

[39] D. Soto and J. Torres, “K3p: A walking gait generator algorithm,” in *14th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)*, 2017, pp. 1–6. Ciudad de México , México. 2017.

[40] S. Marais, A. Nel, and P. Robinson, “Reflex assisted walking for a hexapod robot,” in *Pattern Recognition Association of South Africa and Robotics and Mechatronics International Conference (PRASA-RobMech)*, pp. 1–6. Stellenbosch, Sudáfrica. 2016.

[41] J. Mrva and J. Faigl, “Tactile sensing with servo drives feedback only for blind hexapod walking robot,” in *10th International Workshop on Robot Motion and Control (RoMoCo)*, pp. 240–245. Poznań, Polonia. July 2015.

[42] G. Best, P. Moghadam, N. Kottege, and L. Kleeman, “Terrain classification using a hexapod robot,” in *Australasian Conference on Robotics and Automation*, Sydney, Australia, 2013.

[43] M. Ozkul, A. Saranli, and Y. Yazicioglu, “Acoustic surface perception from naturally occurring step sounds of a dexterous hexapod robot,” *Mechanical Systems and Signal Processing*, vol. 40, no. 1, pp. 178–193, 2013.

[44] J. Christie and N. Kottege, “Acoustics based terrain classification for legged robots,” in *IEEE International Conference on Robotics and Automation*, pp. 3596–3603. Estocolmo, Suecia. 2016.

[45] J. Acuña, “Diseño e implementación de un algoritmo para la locomoción en superficie vertical de un robot hexápodo,” Master’s thesis, Instituto Politécnico Nacional. México, 2015.

[46] L. Palmer, E. Diller, and R. Quinn, “Toward a rapid and robust attachment strategy for vertical climbing,” in *IEEE International Conference on Robotics and Automation*, pp. 2810–2815. Anchorage, Alaska, Estados Unidos. 2010.

[47] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT Press, Cambridge, Massachusetts. United States, 2005.

[48] T. Pire, T. Fischer, J. Civera, P. De Cristóforis, and J. J. Berlles, “Stereo parallel tracking and mapping for robot localization,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1373–1378. Hamburgo, Alemania. 2015.

[49] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard, “An evaluation of the rgb-d slam system,” in *IEEE International Conference on Robotics and Automation*, pp. 1691–1696. Minnesota, Estados Unidos. 2012.

[50] T. Kaupisch and D. Noelke, “Spacebot cup 2013: A space robotics competition,” *KI-Künstliche Intelligenz*, vol. 28, no. 2, pp. 111–116, 2014.

[51] S. Bartsch, T. Birnschein, F. Cordes, D. Kuehn, P. Kampmann, J. Hilljegerdes, S. Planthaber, M. Roemmermann, and F. Kirchner, “Spaceclimber: Development of a six-legged climbing robot for space exploration,” in *41st International Symposium on Robotics and 6th German Conference on Robotics*, pp. 1–8. Munich, Alemania. 2010.

[52] G. Heppner, A. Roennau, J. Oberländer, S. Klemm, and R. Dillmann, “Laurope-six legged walking robot for planetary exploration participating in the spacebot cup,” *WS on Advanced Space Technologies for Robotics and Automation*, vol. 2, no. 13, pp. 69 – 76, 2015.

[53] J. Caballero, “Diseño e impresión 3d de la estructura mecánica de un robot bioinspirado,” B.S. thesis, Universitat Politècnica de Catalunya, Barcelona, España, 2017.

[54] E. Gorrostieta and E. Vargas, “Algoritmo difuso de locomoción libre para un robot caminante de seis patas,” *Computación y Sistemas*, vol. 11, no. 3, pp. 260–287, 2008.

[55] X. Duan, W. Chen, S. Yu, and J. Liu, “Tripod gaits planning and kinematics analysis of a hexapod robot,” in *2009 IEEE International Conference on Control and Automation*, Christchurch, New Zealand, 2009, pp. 1850–1855.

[56] M. Lewis, A. Fagg, and G. Bekey, “Genetic algorithms for gait synthesis in a hexapod robot,” in *Recent trends in mobile robots*. World Scientific, Singapore, 1993, pp. 317–331.

[57] E. Vazquez, J. Saenz, V. Santibañez, and A. Dzul, “Sistema de control de locomoción de un robot hexápodo de 18 gdl,” in *Congreso Internacional de Robótica y Computación, Cabo San Lucas, Mexíco*, 2015.

[58] T. Robotics, “Phantomx ax hexapod kit,” 2008. [Online]. Available: https://www.trossenrobotics.com/hex-mk2

[59] TrossenRobotics, “Dynamixel ax-12a robot actuator from robotis,” 2008. [Online]. Available: https://www.trossenrobotics.com/dynamixel-ax-12-robot-actuator.aspx

[60] K. Townsend, “Adafruit bno055 absolute orientation sensor — adafruit learning system,” 2015. [Online]. Available: https://learn.adafruit.com/adafruit-bno055-absolute-orientation-sensor/overview

[61] Hardkernel, “Odroid-xu4,” 2016. [Online]. Available: https://www.hardkernel.com/main/products/prdt_info.php

[62] R. P. Foundation, “Raspberry pi 3 model b,” 2016. [Online]. Available: https://www.hardkernel.com/main/products/prdt_info.php

[63] Hardkernel, “Odroid-xu4,” 2018. [Online]. Available: https://www.hardkernel.com/shop/odroid-xu4-special-price/

[64] L. Ada, “Using an fsr — force sensitive resistor (fsr) — adafruit learning system,” 2012. [Online]. Available: https://learn.adafruit.com/force-sensitive-resistor-fsr/using-an-fsr

[65] T. Robotics, “Phantomx hexapod assembly guide,” 2016. [Online]. Available: https://learn.trossenrobotics.com/projects/133-phantomx-hexapod-assembly-guide.html

[66] Robotis, “Usb2dynamixel,” 2010. [Online]. Available: http://support.robotis.com/en/product/auxdevice/interface/usb2dxl_manual.htm

[67] J. Mrva, “Design of motion primitives for a hexapod walking robot operating in a rough environment,” Master’s thesis, Czehc Technical University in Prague, República Checa, 2014.

[68] D. Masri, “Terrain modeling and motion planning for hexapod walking robot control,” Master’s thesis, Czech Technical University in Prague, República Checa, 2016.

[69] O. Vele, “Cinemática inversa de un robot bípedo,” *Research Gate*, 2006. [Online]. Available: https://www.researchgate.net/publication/329028019_CINEMATICA_INVERSA_DE_UN_ROBOT_BIPEDO

[70] K. Grochow, S. L. Martin, A. Hertzmann, and Z. Popović, “Style-based inverse kinematics,” in *ACM transactions on Graphics*, vol. 23, no. 3, pp. 522–531. 2004.

[71] J. Estremera, J. A. Cobano, and P. G. De Santos, “Continuous free-crab gaits for hexapod robots on a natural terrain with forbidden zones: An application to humanitarian demining,” *Robotics and Autonomous Systems*, vol. 58, no. 5, pp. 700–711, 2010.

[72] J. Mrva, “Design of motion primitives for a hexapod walking robot operating in a rough environment,” Master’s thesis, Czehc Technical University in Prague, República Checa, 2014.

[73] L. Černỳ, P. Čížek, and J. Faigl, “On evaluation of motion gaits energy efficiency with a hexapod crawling robot,” *Acta Polytechnica CTU Proceedings*, vol. 6, pp. 6–10, 2016.

[74] D. Belter, P. Labkeki, and P. Skrzypczyński, “Adaptive motion planning for autonomous rough terrain traversal with a walking robot,” *Journal of Field Robotics*, vol. 33, no. 3, pp. 337–370, 2016.

[75] P. Čížek, D. Masri, and J. Faigl, “Foothold placement planning with a hexapod crawling robot,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, Canada*. IEEE, 2017, pp. 4096–4101.

[76] B. Sai, B. Kumar, B. Reddy, and A. Kumaar, “Dynamic stability algorithm for a hexapod robot,” in *Recent Developments in Control, Automation & Power Engineering (RDCAPE), Noida, India*. IEEE, 2017, pp. 7–12.

[77] P. Bustos, I. Garcıa, J. Martınez, J. Mateos, A. Sánchez, and L. Rodrıguez, “Loki, a mobile manipulator for social robotics,” in *Workshop of Physical Agents, Santiago de Compostela, España*, 2012.

[78] P. Bustos, “Robocomp,” 2014. [Online]. Available: https://github.com/robocomp/robocomp

[79] P. Bustos, L. Manso, and M. Gutierrez, “Robocomp: A simple robotics framework,” 2014. [Online]. Available: https://robocomp.github.io/web/

[80] G. Heineman and W. Councill, “Component-based software engineering,” *Putting the pieces together, Addison Wesley, Boston*, p. 5, 2001.

[81] A. Brooks, T. Kaupp, A. Makarenko, S. Williams, and A. Oreback, "Towards component-based robotics," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Sendai, Japón*. IEEE, 2005, pp. 163–168.

[82] P. Bustos, "What is robocomp?. robolab - universidad de extremadura." [Online]. Available: https://robolab.unex.es/index.php/robocomp/

[83] P. Bustos and L. Manso, "Using robocompdsl: the command line component generator," 2014. [Online]. Available: https://github.com/robocomp/robocomp/blob/stable/doc/robocompdsl.md#using-robocompdsl-the-command-line-component-generator

[84] K. Martin and B. Hoffman, *Mastering CMake: a cross-platform build system: version 3.1*. Kitware, New York, 2015.

[85] D. Gutiérrez, "Tutorial de qt4 designer y qdevelop," B.S. thesis, Universitat Politècnica de Catalunya, Barcelona, España, 2008.

[86] R. Tappeta, J. Renaud, A. Messac, and G. Sundararaj, "Interactive physical programming: tradeoff analysis and decision making in multicriteria optimization," *Journals : The American Institute of Aeronautics and Astronautics*, vol. 38, no. 5, pp. 917–926, 2000.

[87] R. Wang and X. Qian, *OpenSceneGraph 3.0: Beginner's guide*. Packt Publishing Ltd, Birmingham, 2010.

[88] D. Shreiner, *OpenGL programming guide: the official guide to learning OpenGL, versions 3.0 and 3.1*. Pearson Education, Boston, 2009.

[89] A. Makarenko, A. Brooks, and T. Kaupp, "Orca: Components for robotics," in *International Conference on Intelligent Robots and Systems (IROS), Beijing, China*, 2006, pp. 163–168.

[90] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3. Kobe, Japan, 2009.

[91] M. Gutierrez, A. Romero, P. Bustos, and J. Martínez, "Progress in robocomp," *Journal of Physical Agents*, vol. 7, no. 1, pp. 39–48, 2013.

# Anexos

# Anexo A

# ANEXO: Instalación de software

En este anexo se muestra cómo se debe llevar a cabo la correcta instalación del software necesario para la consecución del proyecto realizado.

### A.0.1. Instalación de Ubuntu - Metacity

Se debe realizar la instalación de Ubuntu 16.04.03 (Distribución de Linux) ya que es la última versión en la cual RoboComp tiene un uso verificado. Además, se lleva a cabo la instalación de la interfaz de Ubuntu-Metacity, que provee al usuario una menor carga gráfica para uso del sistema operativo y así lograr que sea un entorno veloz y eficaz en las tareas a realizar. Los pasos que se deben seguir para llevar a cabo la instalación de Ubuntu con Metacity en una partición del disco duro del ordenador (para instalar junto a Windows) es la siguiente:

1. Realizar una partición del disco duro con el tamaño que se le asignará a Ubuntu La partición quedará con nombre “Espacio libre” o “Espacio no asignado” (Se recomienda sea de al menos 200Gb).

2. Hacer una memoria booteable con Ubuntu 16.04.03.

3. Entrar a la “BIOS” y cambiar el orden de ejecución para que la memoria booteable quede al principio y reiniciar la computadora.

4. Marcar opción *“Install Ubuntu”*. Posterior a ella se deben marcar las casillas: “Descargar actualizaciones al instalar Ubuntu” e “Instalar software de terceros. . . ” (Para las cuales se debe tener una conexión a internet, ya sea por conexion LAN o inalámbrica).

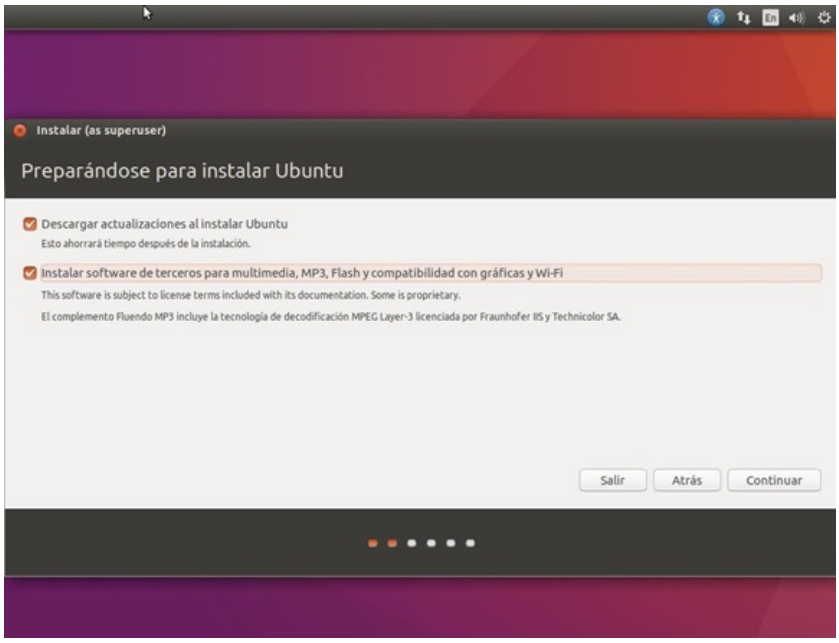



Figura A.1: Preparación para la instalación de Ubuntu. *Fuente propia*.

5. En el tipo de instalación se deberá marcar “Más opciones” para instalar Ubuntu de una forma adecuada junto a otro sistema operativo.

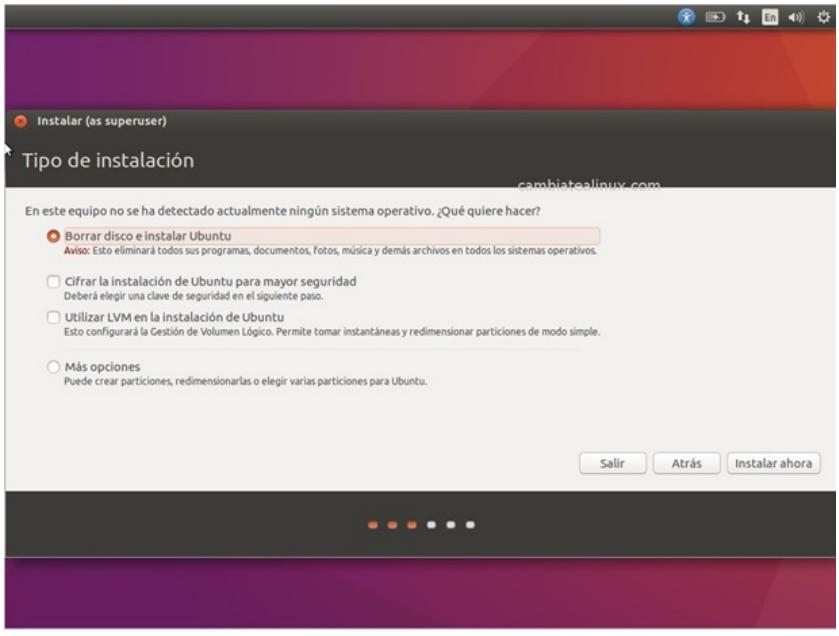



Figura A.2: Elección del tipo de instalación. *Fuente propia*.

6. Se deben realizar 3 particiones a partir del “espacio libre” que se le asignará a Ubuntu

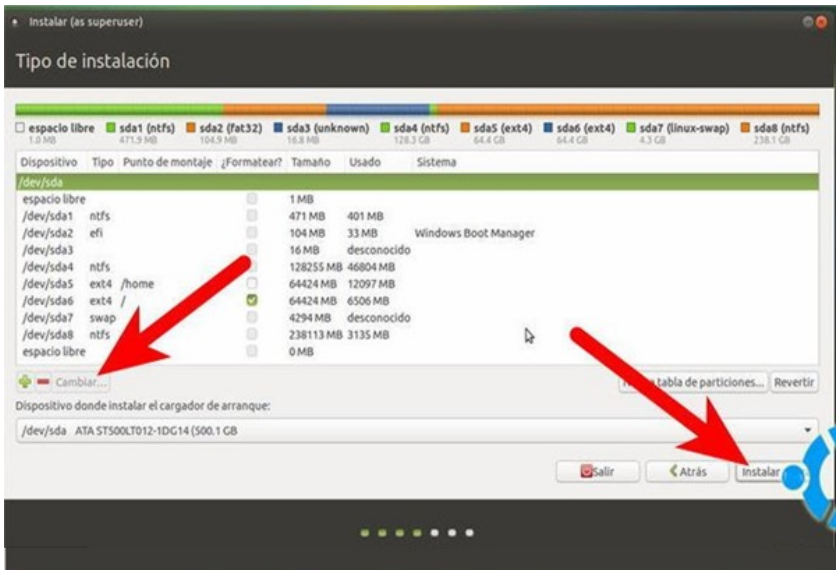



Figura A.3: Creación de nuevas particiones. *Fuente propia*.

Las características de las nuevas particiones se muestran en la siguiente tabla:

| Tipo de Partición | Tamaño | Usar como | Punto de montaje |
|---|---|---|---|
| Lógica | 80 Gb | Sistema de ficheros ext4 transaccional | / |
| Lógica | 124 Gb | Sistema de ficheros ext4 transaccional | /home |
| Lógica | El doble de la memoria RAM que tenga la computadora | | Área de intercambio |

Tabla A.1: Características de las particiones. *Fuente propia*.

7. Se ejecuta “Instalar” y se prosigue la instalación.

8. Al terminar la instalación se retira la memoria y se reinicia el dispositivo. Al momento de ingresar por primera vez a Ubuntu se deben esperar algunos segundos (dependiendo del hardware) para que este se ejecute y cargue todas sus aplicaciones y configuraciones. Como se dijo anteriormente, se usará Metacity como interfaz de Ubuntu, para lo cual se procede de la siguiente manera.

9. Se abre un terminal (Ctrl + Alt + T) y se ejecutan los siguientes comandos (uno a la vez):

   - *sudo apt-get install aptitude* (Instala el paquete aptitude que reemplaza las instalaciones del apt-get, eficaz para una interfaz no gráfica)

- *sudo aptitude update* (Actualiza la lista de paquetes disponibles)
- *sudo aptitude upgrade* (Actualiza el sistema sin borrar nada)
- *sudo aptitude install gnome-flashback* (Instalar una interfaz con menor carga grafica)
- *sudo aptitude install gnome-flashback yakuake* (Instalar un emulador de terminal de carga rápida) (Posteriormente se accede a él con la tecla F12)
- *sudo aptitude install gnome-session-flashback* (Se instala inicio de sesión para entrar a la interfaz “Metacity”)
- Cerrar sesión y entrar por “GNOME Flashback (Metacity)”.

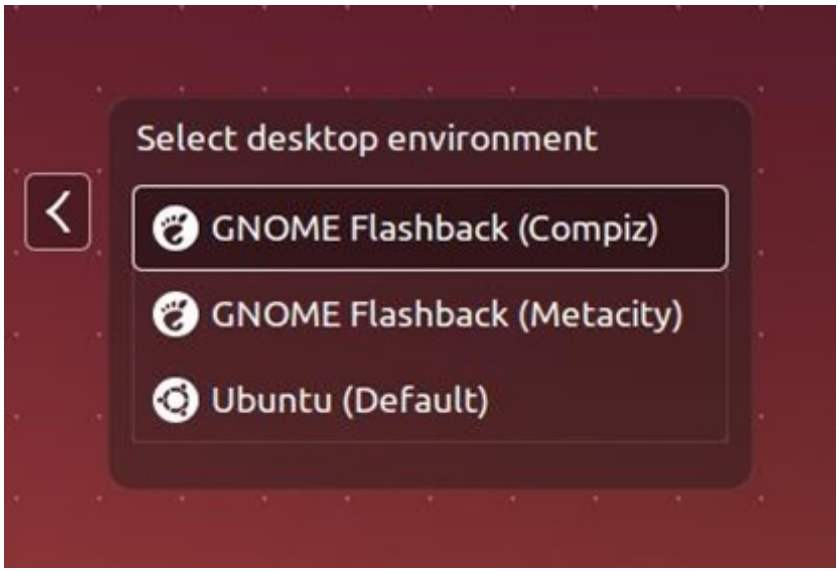



Figura A.4: Interfaces de Ubuntu instaladas. *Fuente propia*.

10. Para tener “Yakuake” en lanzamiento desde el inicio se debe ir a: Aplicaciones – Herramientas del sistema – Preferencias – Aplicaciones al inicio – Añadir:

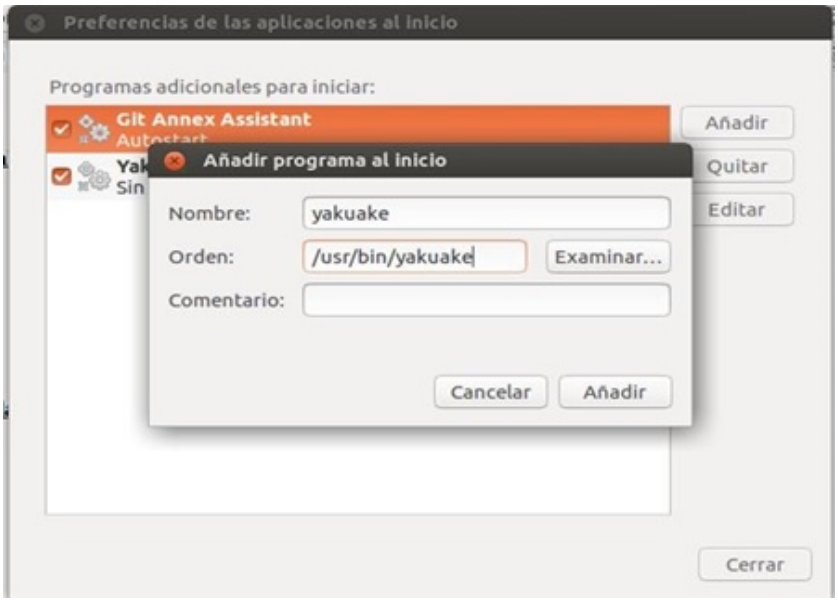



Figura A.5: Ejecución de ”Yakuake” al inicio. *Fuente propia*.

Al final se tendrá una interfaz como la siguiente:

Figura A.6: Interfaz de Ubuntu al finalizar la instalación. *Fuente propia*.

### A.0.2. Instalación de RoboComp

RoboComp es el software de robótica a usar en este proyecto. Se requiere llevar a cabo una instalación adecuada ya que de no llevarse a cabo de la manera correcta se tendrán errores en la ejecución del programa y/o realización de proyectos. Además, RoboComp mantiene en constante actualización algunos de sus componentes debido a nuevos proyectos que lo requieran, por lo

que si se instala correctamente no habrá problemas con la instalación de nuevas actualizaciones.

Para la instalación de RoboComp, se debe realizar:

1. Instalar los siguientes paquetes desde el terminal de Ubuntu:
   *sudo apt-get install git git-annex cmake g++ libgsl0-dev libopenscenegraph-dev cmake-qtgui zeroc-ice35 freeglut3-dev libboost-system-dev libboost-thread-dev qt4-dev-tools yakuake python-pip python-pyparsing python-numpy python-pyside pyside-tools libxt-dev pyqt4-dev-tools qt4-designer libboost-test-dev libboost-filesystem-dev libqt4-dev libqt4opengl-dev*
2. Clonar los repositorios fuente de RoboComp:
   *git clone https://github.com/robocomp/robocomp.git*
3. Ahora se crea un enlace simbólico para que RoboComp pueda encontrarlo todo:
   *sudo ln -s /home/robocomp*
4. Editar el archivo " */.bashrc*" con el siguiente comando:
   *gedit /.bashrc*
5. Agregar las siguientes líneas al final del archivo export ROBOCOMP=~robocomp:
   *export PATH=$PATH:/opt/robocomp/bin*
6. Hacer que bash procese el archivo modificado escribiendo:
   *source ~/.bashrc*
7. Se ejecutan los siguientes comandos línea por línea para a construcción e instalación de RoboComp:
   *sudo [ -d /opt/robocomp ] && rm -r /opt/robocomp*
   *cd robocomp mkdir build cd build cmake .. make*
   *sudo make install*
8. Ahora se le indica a Linux dónde encontrar las bibliotecas de RoboComp
   *sudo nano /etc/ld.so.conf*
9. Agregar la siguiente línea al final
   */opt/robocomp/lib/*
10. Guarde el archivo (Ctrl+O - Enter) y escriba:
    *sudo ldconfig*
11. Instalación de algunas mallas y texturas utilizadas por el simulador:
    *cd ~/robocomp git annex get .*

12. Ahora se ejecuta el simulador para verificar que la instalación se haya realizado exitosamente:

    *cd ~/robocomp/files/innermodel rcis simpleworld.xml*

    RCIS debería estar funcionando con un robot simple dotado de un láser y una cámara RGBD, moviéndose sobre un piso de madera. No olvide dar la vuelta al piso para ver el robot desde arriba.

13. Instalar algunos componentes de RoboLab de GitHub

    *cd /robocomp/components git clone https://github.com/robocomp/robocomp-robolab.git*

    El software de los robots que utilizan RoboComp está compuesto por diferentes componentes de software que trabajan en conjunto y se comunican entre ellos. Lo que acabamos de instalar es solo el núcleo de RoboComp (el simulador, un generador de componentes y algunas bibliotecas). Para tener otras funciones como el control mediante joystick, debemos ejecutar componentes de software adicionales disponibles desde otros repositorios.

    **Conectar un joyStick**

    Si se tiene un joystick, conéctelo al puerto USB y ejecute la siguiente linea

    *cd ~/robocomp/components/robocomp-robolab/components/joystickComp cmake . make*

    *cd bin sudo addgroup your-user dialout*

    *./startJoyStick.sh*

    Tu joystick debería estar ejecutándose. Hará que el robot avance y gire a voluntad. Si el componente no se inicia o el robot no se mueve, detenga joystickcomp con:

    *./forceStopJoyStickComp.sh*

    y compruebe dónde se ha creado el archivo del dispositivo joystick (por ej: /dev/input/js0). Si no es /dev/input/js0, edite /robocomp/components/robocomprobolab/components/joystickComp/etc/config cámbielo y reinícielo. Tenga en cuenta que es posible que desee guardar el archivo de configuración en el directorio de inicio del componente para que no interfiera con futuras actualizaciones de github.

    **Usar el teclado como JoyStick**

    Si no tiene un JoyStick, instale este componente:

    *cd ~/robocomp/components/robocomp-robolab/components/keyboardrobotcontroller cmake .*

    *make*

    *src/keyboardrobotcontroller.py –Ice.Config=etc/config*

    Usar las teclas de flecha para navegar por el robot, la barra espaciadora para detenerlo y una ’q’ para salir.