



INSTITUTO POLITÉCNICO NACIONAL

**CENTRO DE INVESTIGACIÓN EN CIENCIA
APLICADA Y TECNOLOGÍA AVANZADA
UNIDAD QUERÉTARO**

Tesis

**MAESTRÍA EN TECNOLOGÍA
AVANZADA**

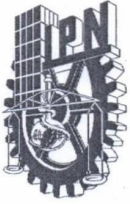
**DISEÑO E IMPLEMENTACIÓN DE UN
ALGORITMO PARA LA LOCOMOCIÓN EN
SUPERFICIE VERTICAL DE UN ROBOT
HEXÁPODO**

**ALUMNO:
JONNATHAN FRANCO ACUÑA**

**Director de Tesis:
DR. EDUARDO CASTILLO CASTAÑEDA**



QUERÉTARO, QRO. JUNIO DEL 2015



INSTITUTO POLITÉCNICO NACIONAL SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

ACTA DE REVISIÓN DE TESIS

En la Ciudad de QUERÉTARO siendo las 12:00 horas del día 28 del mes de MAYO del 2015 se reunieron los miembros de la Comisión Revisora de la Tesis, designada por el Colegio de Profesores de Estudios de Posgrado e Investigación de CICATA - QRO para examinar la tesis titulada:
DISEÑO E IMPLEMENTACIÓN DE UN ALGORITMO PARA LA LOCOMOCIÓN EN SUPERFICIE VERTICAL DE UN ROBOT HEXÁPODO

Presentada por el alumno:

FRANCO
Apellido paterno

ACUÑA
Apellido materno

JONNATHAN
Nombre(s)

Con registro:

B	1	3	1	2	0	1
---	---	---	---	---	---	---

aspirante de:

MAESTRÍA EN TECNOLOGÍA AVANZADA

Después de intercambiar opiniones, los miembros de la Comisión manifestaron **APROBAR LA TESIS**, en virtud de que satisface los requisitos señalados por las disposiciones reglamentarias vigentes.

LA COMISIÓN REVISORA

Director(a) de tesis

EDUARDO CASTILLO CASTAÑEDA

EDUARDO MORALES SÁNCHEZ

JOSÉ JOEL GONZÁLEZ BARBOSA

MAXIMILIANO FRANCISCO RUIZ TORRES

ANTONIO HERNÁNDEZ ZAVALA

PRESIDENTE DEL COLEGIO DE PROFESORES

DRA. EVA GONZÁLEZ JASSO
SECRETARÍA DE EDUCACIÓN PÚBLICA
SECRETARÍA DE INVESTIGACIÓN Y POSGRADO
INSTITUTO POLITÉCNICO NACIONAL
CENTRO DE INVESTIGACIÓN EN
CIENCIA APLICADA
Y TECNOLOGÍA AVANZADA
UNIDAD QUERÉTARO
DIRECCIÓN




INSTITUTO POLITÉCNICO NACIONAL
SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

CARTA CESIÓN DE DERECHOS

En la Ciudad de México, D.F. el día 01 del mes de JUNIO del año 2015, el que suscribe JONNATHAN FRANCO ACUÑA alumno del Programa de Maestría en Tecnología Avanzada, con número de registro B131201, adscrito al CICATA – QRO., manifiesto que es el autor intelectual del presente trabajo de Tesis bajo la dirección del Dr. EDUARDO CASTILLO CASTAÑEDA y cede los derechos del trabajo titulado Diseño e implementación de un algoritmo para la locomoción en superficie vertical de un robot hexápodo, al Instituto Politécnico Nacional para su difusión, con fines académicos y de investigación.

Los usuarios de la información no deben reproducir el contenido textual, gráficas o datos del trabajo sin el permiso expreso del autor y/o director del trabajo. Este puede ser obtenido escribiendo a las siguientes direcciones: jonnathanfranco.1100@gmail.com, jonnathan_franco@hotmail.com. Si el permiso se otorga, el usuario deberá dar el agradecimiento correspondiente y citar la fuente del mismo.

Jonnathan Franco Acuña 

JONNATHAN FRANCO ACUÑA

Índice general

1. Introducción	1
1.1. Antecedentes	1
1.2. Justificación	2
1.3. Objetivos	3
1.3.1. Objetivo general	3
1.3.2. Objetivos específicos	3
1.4. Contenido de la tesis	3
2. Marco teórico	5
2.1. Robots móviles	5
2.1.1. Tipos de locomoción en robots terrestres	6
2.1.2. Robots hexápodos	7
2.2. Planeación de trayectorias	8
2.2.1. Generalidades	8
2.2.2. Trayectorias de movimiento punto a punto	9
2.3. Localización por medio de análisis de imágenes	11
2.3.1. Cálculo de una distancia a partir de una referencia en un patrón	12
2.3.1.1. Parámetros de calibración	12
3. Estado del arte	15
3.1. Locomoción en superficie horizontal	15
3.2. Locomoción en superficie vertical	19
4. Diseño y construcción del prototipo	23
4.1. Configuración de las extremidades	23
4.2. Sistema de sujeción	24
4.3. Análisis cinemático de posición	26
4.4. Equilibrio de momentos	29
4.5. Modelado CAD	31

4.5.1. Modelado CAD SolidWorks	31
4.5.2. Modelado CAD ADAMS	33
4.6. Selección de componentes	35
4.7. Activación de electroválvulas	38
4.8. Construcción	39
4.9. Alimentación Servo-Motores	40
5. Descripción de la estrategia de locomoción	43
5.1. Concepto	43
5.2. Descripción del algoritmo por medio de diagrama de flujo	44
5.3. Programación	46
5.3.1. Cálculo de la cinemática inversa	46
5.3.2. Uso de la MCI en función del tiempo	47
5.4. Comunicación serie Matlab-MCC	48
5.4.1. Maestro Control Center “Frames secuence”	49
5.4.2. Protocolo de comandos MCC	50
5.4.2.1. Bits de destino	50
5.4.3. Programación MCC desde Matlab	51
5.5. Cálculos fuera de línea	53
5.6. Secuencia de inicialización	53
6. Validación	55
6.1. Simulación Matlab	55
6.2. Pruebas en SolidWorks	59
6.3. Pruebas en ADAMS	63
6.4. Cálculo de distancia por medio de análisis de imágenes	69
7. Conclusiones	73
Referencias	76
A. Anexos	81

Índice de figuras

2.1. Configuración robot tipo oruga. [Solaque Guzman et al., 2014]	6
2.2. Configuración tipo ruedas. [SuperRobotica, 2012]	7
2.3. Configuración tipo hexápodo. [Reder, 2012]	7
2.4. Gráfica de aceleración de un polinomio cúbico.	9
2.5. Perfiles de trayectoria con polinomio grado 5.	11
3.1. Posición deseada en el momento de apoyo. [Schmitz and Cruse, 2004]	16
3.2. Marcha de onda lenta. [Porta and Celaya, 2004]	17
3.3. Trípede alterno y polígono de estabilidad. [Estremera et al., 2010] . .	18
3.4. Postura óptima con adhesivos secos. [Boscariol et al., 2013]	20
3.5. a)Robot Hexpiderix, Polígono de estabilidad. b)Secuencia de locomoción «Tripode alterno». [Sandoval-Castro et al., 2013]	21
4.1. Configuración de las patas de la cucaracha.	24
4.2. Esquemático de configuración de las patas.	24
4.3. Ventosa paralela al plano de sujeción.	25
4.4. Recorrido angular de la ventosa.	26
4.5. Modelo geométrico inverso «Patas Grupo 1».	27
4.6. Vista superior «Patas grupo 1».	27
4.7. Modelo geométrico inverso «grupo 2».	28
4.8. Vista superior «patas grupo 2».	29
4.9. Esquemático con ubicación de los motores.	31
4.10. Modelado CAD.	32
4.11. Trayectoria del paso.	33
4.12. Grupos de patas.	33
4.13. Pruebas de traslación del tórax.	34
4.14. Rotación del tórax.	34
4.15. Simulación de un paso.	35
4.16. Gráfica de par en los motores para un paso en superficie vertical. . .	35

4.17. Diagrama de componentes.	36
4.18. Mini Maestro 24 canales [Pololu, 2014].	36
4.19. Motor Turnigy.	37
4.20. Electroválvula FESTO.	37
4.21. Generador de vacío FESTO.	38
4.22. Ventosa FESTO.	38
4.23. Esquemático optoacoplador.	39
4.24. Ensamblaje del Robot.	40
5.1. Diagrama de locomoción.	44
5.2. Inicio de paso con cada trío de ventosas.	44
5.3. Matriz «MCI».	47
5.4. Relación MCI y tiempo.	47
5.5. PinOut del Mini Maestro Pololu.	48
5.6. Interfaz del MCC.	49
5.7. Ajuste de bits para comunicación serial.	51
5.8. Relación MCC - Grados.	52
6.1. Simulación seguimiento de perfil posición.	55
6.2. Simulación de secuencia de locomoción con medidas reales.	56
6.3. Tabla de posiciones angulares.	56
6.4. Cuadro 1.	57
6.5. Cuadro 2.	57
6.6. Cuadro 3.	58
6.7. Cuadro 4.	58
6.8. Cuadro 5.	58
6.9. Etapa 1, «ubicación frente a la superficie vertical».	59
6.10. Etapa 2, «elevación tórax».	60
6.11. Etapa 3 «ubicación de las ventosas en la pared».	61
6.12. Etapa 4, «re-ubicación de las ventosas».	61
6.13. Etapa 5, «orientación tórax».	62
6.14. Etapa 6, «ubicación de ventosas».	63
6.15. Traslación en el eje Z.	64
6.16. Gráfica de par en traslación del eje Z.	64
6.17. Gráfica de par «Pata 1» en traslación en eje Z.	65
6.18. Gráfica de par «Pata 3» en traslación en eje Z.	65
6.19. Gráfica de par «Pata 5» en traslación en eje Z.	65

6.20. Traslación en el eje X.	66
6.21. Gráfica de par en traslación del eje X.	66
6.22. Traslación en el eje Y.	67
6.23. Gráfica de par en traslación del eje Y.	67
6.24. Rotación del tórax en tres ejes.	68
6.25. Horizontal - Vertical.	68
6.26. Comparación de par Horizontal - Vertical.	69
6.27. Patrón cuadriculado del tórax.	70
6.28. Distancia recorrida por el tórax.	71

Resumen

En este trabajo se presenta el diseño de un algoritmo para la locomoción de un robot hexápodo en superficie vertical. El Robot llamado Wall-Bot es un prototipo bio-inspirado y se compone de seis cadenas cinemáticas abiertas unidas a un tórax rígido y distribuidas uniformemente, donde las dos patas traseras y las dos patas delanteras, están compuestas por cuatro articulaciones rotacionales, y las dos patas laterales, están compuestas por tres articulaciones rotacionales. El algoritmo consiste en realizar una serie de cálculos antes de iniciar la secuencia de marcha, donde se inicia un contador en tiempo real y se pregunta por la distancia según el perfil de posición, esta distancia es almacenada y se compara con el primer vector columna de una matriz con información angular de cada articulación, se escoge el dato más cercano a los valores calculados y así podemos saber las posiciones angulares de cada motor en función del tiempo.

El sistema de sujeción implementado es tipo ventosa, el cual se compone de electro válvulas, generadores de vacío y ventosas. Como modulo de control de los servomotores, se utilizó una tarjeta desarrollada por Pololu, la cual se comunica vía serial desde el computador y la secuencia de locomoción implementado se llama trípode alterno.

Capítulo 1

Introducción

En este proyecto se presenta el diseño de un algoritmo de locomoción para un robot hexápodo escalador, en el que inicialmente se presenta introducción, objetivos y justificación. Posteriormente se presenta el marco teórico y el estado del arte donde se ofrece el preámbulo general de conocimientos previos para comprender el desarrollo del proyecto. Dentro del cuerpo del trabajo se encuentra el marco teórico, estado del arte, diseño del prototipo, selección de componentes y validación. Finalmente se explica la estrategia de locomoción y el programa principal desarrollado en MatLab, así como, las conclusiones y el trabajo a futuro.

1.1. Antecedentes

La robótica es un campo multidisciplinario que permite facilitar las labores de alto riesgo que representan ciertas tareas como supervisar actividades radioactivas en una planta nuclear o monitorear zonas peligrosas como el cráter de un volcán con actividad sísmica, que requieren la implementación de prototipos tele-operados con el fin de mantener una comunicación en tiempo real con el robot. Existen diferentes tipos de robots y variadas clasificaciones de estos robots, por ejemplo, robots industriales, robots de servicio y robots tele-operados. Estas clasificaciones nos permite visualizar los diferentes enfoques que ha tenido el desarrollo de la robótica donde uno de ellos es la robótica de inspección que se enfoca en la exploración de métodos poco convencionales que permitan descubrir formas de locomoción más eficientes. Uno de estos métodos es el sistema de locomoción híbrido que hace uso de los mejores atributos que ofrecen los métodos incorporados, por ejemplo el uso de patas y ruedas que utiliza el (Roller-Walker), mencionado en el artículo publicado por la Universidad Europea de Madrid [Asbeck and Kim, 2006].

Debido al riesgo que involucra el desarrollo de ciertas actividades se ha venido desarrollando herramientas que facilitan algunas labores reemplazando el hombre y permitiendo la ejecución de tareas a distancia, este tipo de herramientas son conocidos también como robots a pesar de no ser autónomos en su totalidad [Correa, 2005]. Con el fin de facilitar las tareas de supervisión o incluso reparación y mantenimiento, se han desarrollado robots provistos de herramientas para la visualización del entorno y manipulación de herramientas a distancia los cuales se han empleado en diferentes campos de los cuales se manejan ambientes o componentes inestables como químicos o explosivos, facilitando las labores de alto riesgo y protegiendo la integridad física del ser humano.

A medida que se desarrollan sistemas robóticos para el monitoreo y manipulación de herramientas en sitios inaccesibles, surgen necesidades más específicas donde los robots hexápodos están encontrando un gran número de aplicaciones, donde se aprovecha la facilidad para desplazarse en terrenos irregulares y la estabilidad estática que ofrece al ser comparado con un robot bípedo o cuadrúpedo, por lo tanto, los robots con patas tienen mejores propiedades para desplazarse en terrenos rugosos o llenos de obstáculos [Correa, 2005], como por ejemplo, robots diseñados para servicios de limpieza y mantenimiento en superficies verticales [Luk et al., 2005], o robots diseñados con capacidad de llevar altas cargas [Grieco et al., 1998].

Dentro de la robótica de inspección existen diferentes sistemas y algoritmos desarrollados para solucionar problemas específicos como optimización de energía o cambios en la secuencia de locomoción aunque dentro de la literatura revisada no se ha propuesto una solución general para el desplazamiento de cualquier robot caminante en el espacio. Con el desarrollo de este proyecto se propondrá un algoritmo de locomoción que permitirá el desplazamiento de un robot hexápodo siguiendo un perfil de posición de cualquier trayectoria. El algoritmo es validado en un prototipo hexápodo diseñado para caminar en superficies verticales, no obstante, se propone un método de implementación para seguimiento de trayectorias punto a punto donde únicamente se modifica del modelo geométrico inverso del robot dentro del programa para aplicarse en cualquier prototipo con diferentes cantidades de actuadores.

1.2. Justificación

Uno de los campos de investigación y desarrollo de gran interés que han encontrado las comunidades científicas ha sido la robótica de inspección, siendo los robots un

componente de gran uso en procesos industriales ya que hay sitios de poca accesibilidad donde los trabajadores no pueden ingresar por motivos de seguridad o porque simplemente no se tiene una vía de acceso. Existen diferentes circunstancias donde es necesario la implementación de robots que realicen inspección y mantenimiento en zonas de contaminación o alto riesgo como la inspección de los tubos del generador de vapor en un reactor nuclear, la búsqueda de grietas estructurales en puentes y túneles o la inspección del fondo marino para la explotación petrolera. En síntesis, la búsqueda de nuevas alternativas en el desarrollo tecnológico nos ha llevado a mejorar la calidad en la realización de tareas y al aumento en la seguridad del personal de trabajo, por este motivo se pretende impulsar el desarrollo y la investigación en el campo de la robótica, diseñando robots más robustos y confiables con la intención de incentivar el uso y la implementación de robots desarrollados en México.

Existen métodos desarrollados para la locomoción, como optimización de energía, evasión de obstáculos o incluso temas de estabilidad, sin embargo, dentro de la literatura revisada no se encontró un algoritmo que permita la locomoción de un robot caminante para realizar tareas de inspección en superficies verticales, por lo tanto, en este proyecto se desarrollará un algoritmo que permita la locomoción de un robot hexápodo en superficie vertical, proponiendo un algoritmo de fácil implementación en robots caminantes.

1.3. Objetivos

1.3.1. Objetivo general

Diseñar e implementar un algoritmo para la locomoción en superficie vertical de un robot hexápodo.

1.3.2. Objetivos específicos

- Diseñar y construir un robot hexápodo con sistema de sujeción tipo ventosa.
- Implementar un algoritmo de locomoción para superficie vertical.
- Aplicar un método para seguir trayectorias en superficie vertical.

1.4. Contenido de la tesis

En el presente documento se presenta el desarrollo de la temática iniciando con una introducción a la robótica de inspección donde se especifica la justificación y

los objetivos de la tesis; en el capítulo 2 se expone el marco teórico correspondiente a la temática a tratar, dando lugar al estado del arte presente en el capítulo 3 donde se presentan trabajos relacionados con la robótica móvil y un breve análisis de contenido para cada uno de los trabajos expuestos. En el capítulo 4, se presenta de manera detallada los criterios de diseño, así como la especificación del sistema de sujeción, modelado CAD y cálculos matemáticos necesarios en el diseño mecánico. En los siguientes capítulos se presentan los resultados del trabajo realizado que son; construcción y ensamble, descripción de la estrategia de locomoción, validación experimental y finalmente las conclusiones y referencias. En la parte final del documento se presentan los anexos donde se muestra a detalle características específicas de programación.

Capítulo 2

Marco teórico

2.1. Robots móviles

La robótica ha tenido grandes avances en los últimos años tanto en la variedad de los diseños de locomoción como en los sistemas de control dando lugar a tres grandes grupos en los que se pueden clasificar los robots móviles que son:

- Robots de funcionamiento repetitivo, son los más utilizados en la industria, trabajan realizando tareas invariantes y repetitivas y con una limitada percepción de las variables del entorno, este tipo de robots son relativamente rápidos y de alta repetitividad. Esta clase de robots pretende ahorrar costos en mano de obra, mejorar la productividad y realizar tareas peligrosas para trabajadores humanos [Beekman, 1999].
- Robots tele-operados, su principal función es la realimentación sensorial del entorno donde se encuentre, tales como imágenes, fuerzas o distancias. Son dispositivos diseñados para ser manipulados por un operador con cierto grado de automatización, es decir, el operador cierra un bucle de control de alto nivel. En robots tele-operados con sistema de manipulación se emplean brazos y manos antropomórficos con sistemas automáticos que facilitan el sistema de manipulación por parte del operador. Este tipo de robots es usado para realizar tareas a distancia donde el acceso es difícil o en ambientes contaminantes o peligrosos, tales como las que se realizan en obras de construcción o en mantenimientos de redes eléctricas.
- Robots autónomos o inteligentes, son los más complejos desde el punto de vista del procesamiento de información. Son máquinas capaces de percibir y analizar información del entorno para la toma de decisiones, pueden trabajar en entornos

poco estructurados y dinámicos. En los últimos años se han hecho importantes esfuerzos en la aplicación de técnicas de inteligencia artificial empleando diferentes técnicas que su única limitación es la capacidad de procesamiento requerida para el tratamiento de la información en tiempo real. Una técnica para la solución de problemas de incertidumbre por variación de la información del entorno es incrementar la información que se dispone del entorno para la realimentación sensorial. Desde el punto de vista de planificación, existen diferentes arquitecturas diseñadas, teniendo en cuenta el tiempo de respuesta disponible para que el sistema actúe [Baturone, 2001].

2.1.1. Tipos de locomoción en robots terrestres

Dentro de la clasificación de los robots terrestres existen diferentes tipos de locomoción que son, tipo oruga, con ruedas y con patas, cada uno tiene sus ventajas y desventajas dependiendo del terreno en que se desplace, a continuación se describirá genéricamente cada uno de estos sistemas de locomoción.

- Tipo oruga: Estos robots tienen unida la rueda delantera con la rueda trasera y su ventaja es que brinda una mayor adherencia al terreno y ofrece una mayor adaptabilidad a terrenos irregulares. La desventaja de este tipo de locomoción es que no puede realizar trayectorias curvas, la única forma de realizar giros es invertir el giro en uno de los lados, lo que ocasiona un giro en un solo lugar, pero el desplazamiento solo se realiza en línea recta. En la figura 2.1 se muestra un chasis con locomoción tipo oruga.



Figura 2.1: Configuración robot tipo oruga. [Solaque Guzman et al., 2014]

- Robots con ruedas: La principal ventaja que ofrece este tipo de robots es la facilidad de construcción y programación que tienen, además que puede desplazarse a velocidades mayores que con orugas o patas. Su principal desventaja es que no puede pasar por encima de obstáculos o de un tamaño superior al radio

de la rueda [Ramírez, 2004]. Existen diferentes estructuras en este tipo de locomoción como 2 ruedas motrices direccionables con dos ruedas no direccionables no motrices, ruedas motrices direccionables y ruedas motrices no direccionables combinadas con ruedas no motrices direccionables. En la figura 2.2 se muestra el chasis de un robot con sistema de locomoción con ruedas.



Figura 2.2: Configuración tipo ruedas. [SuperRobotica, 2012]

- Robots con patas: Los robots con patas son el resultado del enfoque bio-inspirado que ha tomado el desarrollo en la robótica, este tipo de locomoción tiene como ventaja la adaptabilidad al desplazarse en terrenos irregulares, según el número de patas el robot se puede denominar bípedo, cuadrúpedo, hexápodo u octópodo, también existen diferentes grados de libertad por pata de cada robot, lo que aumenta la complejidad a la hora de programar la rutina de desplazamiento. En la figura 2.3 se muestra un chasis de un robot tipo hexápodo.



Figura 2.3: Configuración tipo hexápodo. [Reder, 2012]

2.1.2. Robots hexápodos

La bio-inspiración ha abierto grandes posibilidades hacia la solución de diferentes problemáticas, de manera que la locomoción por medio de patas tiene como ventaja la adaptabilidad al desplazarse por terrenos irregulares y actualmente se le han encontrado muchas aplicaciones que incluyen exploraciones, aplicaciones militares, inspección y mantenimiento industrial entre otras. Debido a la comprobada destreza de los hexápodos para mantener la estabilidad, es uno de los diseños más usados ya que puede

navegar en terrenos rocosos, arenosos, sobrepasar ciertos obstáculos, caminar por superficies inclinadas etc [Salavert and Graciani, 1995]. El sistema de locomoción por medio de patas ha generado grandes expectativas en la comunidad científica que basándose en estudios realizados indican que ciertos animales obtienen la adaptabilidad y flexibilidad principalmente de una organización muy descentralizada del sistema de control que generan en el periodo de marcha [Wettergreen and Thorpe, 1996]. Los robots hexápodos presentan un concepto de mecanismo que proporciona movimientos rotacionales por uno o varios motores que manejados por un sistema de control genera movimientos de traslación, ofreciendo al sistema varios grados de libertad. Un robot con patas requiere de mayor complejidad en el sistema de control y número de sensores que independientemente si el robot es bípedo, cuadrúpedo o hexápodo, hay que considerar un análisis estático y otro dinámico para lograr la estabilidad mientras camina.

Uno de los aspectos a tener en cuenta en la programación de un robot tipo hexápodo es la coordinación de las seis patas en el movimiento de tal forma que se garantice su estabilidad en el terreno, esta estabilidad es lograda a partir de información retroalimentada que indique al robot las irregularidades presentes en el terreno y de esta forma evadir obstáculos o sobrepasarlos.

2.2. Planeación de trayectorias

2.2.1. Generalidades

La planificación de trayectorias es el cálculo de una función $q_d(t)$ que especifica completamente el movimiento del robot a medida que atraviesa alguna ruta [Spong et al., 2006].

Para poder determinar un comportamiento en el desplazamiento de un robot, es necesario saber la dirección de desplazamiento, la velocidad y la aceleración que tendrá el centro del tórax de nuestro robot, lo que conduce a la planificación de trayectorias. Al momento de planificar una trayectoria, hay que tener en cuenta ciertos requerimientos como por ejemplo la velocidad y aceleración de una articulación mientras atraviesa un camino, en nuestro caso, se propone un caso muy específico donde se supone una superficie totalmente plana y lisa, con el fin de demostrar el funcionamiento y los requisitos mínimos para seguir un perfil de velocidad. Esto conduce a la planificación de trayectorias que satisfagan los requerimientos del robot.

Para generar una trayectoria hay que considerar en primer lugar la ruta que deberá seguir el robot para desplazarse entre dos puntos, donde se considerará como primer

caso un desplazamiento en línea recta con un espacio libre de obstáculos. En algunos casos pueden haber restricciones sobre la trayectoria, por ejemplo, si el robot debe empezar y terminar con una velocidad cero, sin embargo, es fácil darse cuenta que hay un número infinito de trayectorias que satisfagan un número finito de restricciones, por lo tanto, es muy práctico elegir trayectorias que satisfagan un número finito de restricciones, por ejemplo el uso de polinomios de grado “n” en donde “n” depende del grado de limitaciones que debe cumplir [Spong et al., 2006].

2.2.2. Trayectorias de movimiento punto a punto

El problema principal es encontrar la conexión entre una configuración del robot y unas restricciones especificadas en los puntos iniciales y finales de la trayectoria, por ejemplo, velocidad y/o aceleración. Para generar una trayectoria entre dos puntos con ciertas especificaciones iniciales y finales, es posible generar una trayectoria con un polinomio de grado tres como se muestra en la ecuación 2.1.

$$q(t) = q_0 + 3(q_f - q_0)t^2 - 2(q_f - q_0)t^3 \quad (2.1)$$

La anterior ecuación conduce a posiciones y velocidades continuas, pero discontinuas en aceleración como se puede observar en la figura 2.1.

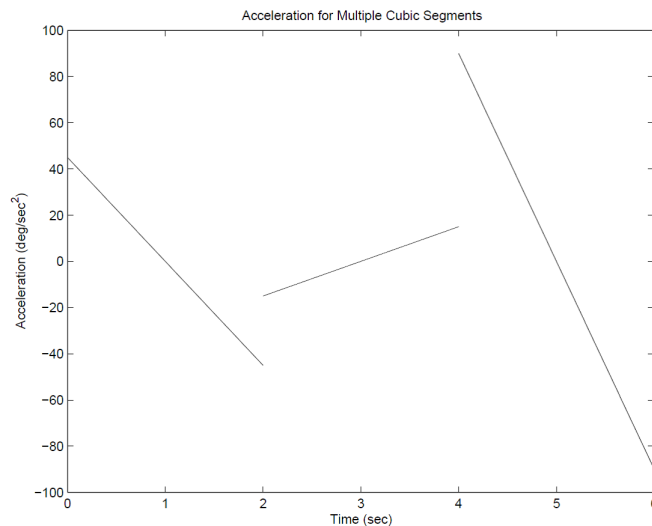


Figura 2.4: Gráfica de aceleración de un polinomio cúbico.

Como muestra el ejemplo anterior, la planificación de trayectorias utilizando un polinomio cúbico produce posiciones discontinuas en la aceleración llamados impulsos o «Jerks» que pueden excitar modos de vibración en el manipulador y reducir la

precisión de seguimiento de la trayectoria. Por esta razón, se especifica las limitaciones de la aceleración, así como en la posición y la velocidad [Spong et al., 2006].

El derivado de la aceleración es un Jerk y una discontinuidad en la aceleración conduce a un impulso Jerk que puede ocasionar modos de vibración en el robot y reducir la capacidad de seguimiento de la trayectoria [Spong et al., 2006]. Por esta razón, se puede especificar las condiciones de la aceleración, así como posición y velocidad. En este caso, tenemos seis limitaciones (condiciones iniciales y finales en posición, velocidad y aceleración.), por lo tanto se requiere un polinomio de quinto orden como se muestra en la ecuación 2.2.

$$q(t) = a_0 + a_1t + a_2t^2 + a_3t^3 + a_4t^4 + a_5t^5 \quad (2.2)$$

Al derivar obtenemos la velocidad,

$$v(t) = a_1 + 2a_2t_0 + 3a_3t_0^2 + 4a_4t_0^3 + 5a_5t_0^4 \quad (2.3)$$

Al derivar nuevamente tenemos la aceleración,

$$a(t) = 2a_2 + 6a_3t_0 + 12a_4t_0^2 + 20a_5t_0^3 \quad (2.4)$$

Construimos nuestra matriz de condiciones para la generación de trayectorias con valores iniciales y valores finales. En las siguientes ecuaciones se especifica cada condición para generar el arreglo matricial:

$$q_0 = a_0 + a_1t + a_2t^2 + a_3t^3 + a_4t^4 + a_5t^5 \quad (2.5)$$

$$v_0 = a_1 + 2a_2t_0 + 3a_3t_0^2 + 4a_4t_0^3 + 5a_5t_0^4 \quad (2.6)$$

$$a_0 = 2a_2 + 6a_3t_0 + 12a_4t_0^2 + 20a_5t_0^3 \quad (2.7)$$

$$q_f = a_0 + a_1t + a_2t^2 + a_3t^3 + a_4t^4 + a_5t^5 \quad (2.8)$$

$$v_f = a_1 + 2a_2t_0 + 3a_3t_0^2 + 4a_4t_0^3 + 5a_5t_0^4 \quad (2.9)$$

$$a_f = 2a_2 + 6a_3t_0 + 12a_4t_0^2 + 20a_5t_0^3 \quad (2.10)$$

De esta manera obtenemos seis ecuaciones con seis incógnitas, donde podemos encontrar los valores de los coeficientes y saber la posición, velocidad y aceleración para cada instante de tiempo. Para encontrar los coeficientes “a” multiplicamos la

inversa de los valores de la ecuación en su respectivo orden por el vector columna de restricciones, en la siguiente matriz ecuación se muestra el arreglo matricial de las ecuaciones quedando de la siguiente manera:

$$\begin{bmatrix} a_0 & a_1t_0 & a_2t_0^2 & a_3t_0^3 & a_4t_0^4 & a_5t_0^5 \\ 0 & a_2 & 2a_3t_0 & 3a_4t_0^2 & 4a_5t_0^3 & 5a_6t_0^4 \\ 0 & 0 & 2a_3 & 6a_4t_0 & 12a_5t_0^2 & 20a_6t_0^3 \\ a_0 & a_1t & a_2t^2 & a_3t^3 & a_4t^4 & a_5t^5 \\ 0 & a_2 & 2a_3t & 3a_4t^2 & 4a_5t^3 & 5a_6t^4 \\ 0 & 0 & 2a_3 & 6a_4t & 12a_5t^2 & 20a_6t^3 \end{bmatrix}^{-1} \begin{bmatrix} q_0 \\ v_0 \\ a_0 \\ q \\ v \\ a \end{bmatrix} = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix}$$

Cada uno de los perfiles de la trayectoria se pueden graficar en función del tiempo como se muestra en la figura 2.5

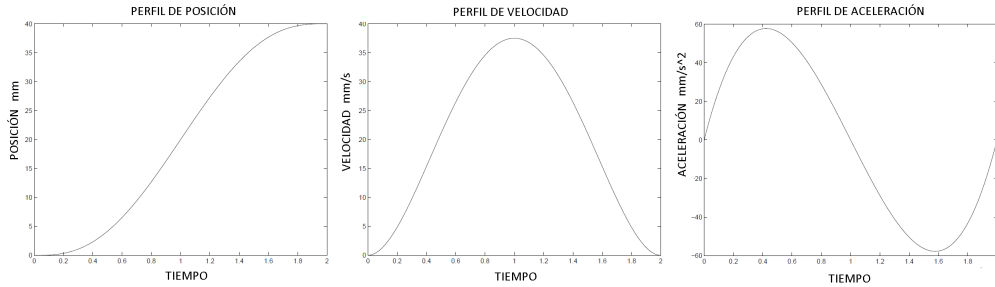


Figura 2.5: Perfiles de trayectoria con polinomio grado 5.

2.3. Localización por medio de análisis de imágenes

En el método mencionado en el presente documento se presentan diferentes parámetros de un proceso conocido como calibración geométrica de la cámara para poder hacer la estimación de parámetros que serán útiles en el resto del proceso. A continuación se describen brevemente los sistemas de coordenadas y las coordenadas homogéneas.

Las coordenadas cartesianas «x, y, z» de un punto «p» en este sistema de coordenadas se representa de la siguiente manera:

$$\begin{cases} x = \overrightarrow{OP} \cdot \mathbf{i} \\ y = \overrightarrow{OP} \cdot \mathbf{j} \\ z = \overrightarrow{OP} \cdot \mathbf{k} \end{cases} \iff \overrightarrow{OP} = x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$$

Se considera que el tórax del robot se desplazará en un plano, este plano es calculado al colocar el patrón de calibración en el plano de desplazamiento.

2.3.1. Cálculo de una distancia a partir de una referencia en un patrón

Para realizar el cálculo de una distancia deseada es necesario realizar la calibración de la cámara [Trucco and Verri, 1998], por lo tanto hay que encontrar la aproximación matemática de la conversión entre una imagen al plano de desplazamiento. Al tener la información que corresponde a la dirección de cada pixel, se puede deducir información en tres dimensiones del mundo real, sin embargo, en nuestra propuesta los pixeles de la imagen serán proyectados al plano de desplazamiento, por lo que el desplazamiento será calculado sobre este plano.

2.3.1.1. Parámetros de calibración

La calibración de la cámara es muy importante en el proceso de reconstrucción del entorno en tres dimensiones, debido a que calcula los parámetros intrínsecos y extrínsecos del dispositivo que se está usando. La idea clave de la calibración de cámaras es escribir las ecuaciones de enlace de coordenadas conocidas de un conjunto de puntos 3D con sus proyecciones y resolver estas ecuaciones para los parámetros de la cámara.

Sean $[X^C, Y^C, Z^C]^T$, las coordenadas del punto P en la referencia de la cámara, con $Z^c > 0$ para que P sea visible. La posición y orientación de la cámara son desconocidas, pues a diferencia de los marcos de referencia de la imagen y el mundo, los marcos de la cámara son inaccesibles directamente. Esto equivale a decir que nosotros no conocemos los parámetros extrínsecos de la cámara, esto es una matriz de rotación 3×3 y un vector de traslación T [Trucco and Verri, 1998], representado de la siguiente manera:

$$\begin{bmatrix} X^c \\ Y^c \\ Z^c \\ 1 \end{bmatrix} = \begin{bmatrix} R & T \\ \vec{0} & 1 \end{bmatrix} + \begin{bmatrix} X^w \\ Y^w \\ Z^w \\ 1 \end{bmatrix} \quad (2.11)$$

Si deseamos que un punto representado en la cámara, sea representado en el mundo, se invierte la ecuación 2.11 de la siguiente forma:

$$\begin{bmatrix} X^w \\ Y^w \\ Z^w \\ 1 \end{bmatrix} = \begin{bmatrix} R^T & -RT \\ \vec{0} & 1 \end{bmatrix} \begin{bmatrix} X^c \\ Y^c \\ Z^c \\ 1 \end{bmatrix} \quad (2.12)$$

La ecuación 2.11 representa los puntos respecto a la cámara, después existe una proyección perspectiva para formar la imagen representada por:

$$s \begin{bmatrix} U \\ V \\ W \end{bmatrix} = \begin{bmatrix} \alpha & 0 & U_0 & 0 \\ 0 & \beta & V_0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R & T \\ \vec{0} & 1 \end{bmatrix} \begin{bmatrix} X^w \\ Y^w \\ Z^w \\ 1 \end{bmatrix} \quad (2.13)$$

Donde la matriz $\begin{bmatrix} \alpha & 0 & U_0 & 0 \\ 0 & \beta & V_0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ son los parámetros intrínsecos y la matriz $\begin{bmatrix} R & T \\ \vec{0} & 1 \end{bmatrix}$ son los parámetros extrínsecos. Haciendo el producto se tiene:

$$S \begin{bmatrix} U_i \\ V_i \\ W_i \end{bmatrix} = M \begin{bmatrix} X_i^w \\ Y_i^w \\ Z_i^w \\ 1 \end{bmatrix} \quad (2.14)$$

Ahora se divide U y V entre S de la siguiente manera:

$$U = \frac{m_{11}X_i^w + m_{12}Y_i^w + m_{13}Z_i^w + m_{14}}{m_{31}X_i^w + m_{32}Y_i^w + m_{33}Z_i^w + m_{34}} \quad (2.15)$$

$$V = \frac{m_{21}X_i^w + m_{22}Y_i^w + m_{23}Z_i^w + m_{24}}{m_{31}X_i^w + m_{32}Y_i^w + m_{33}Z_i^w + m_{34}} \quad (2.16)$$

La matriz M está definida con un factor de escala arbitrario y tiene por tanto 11 entradas independientes que pueden ser determinados a través de sistemas lineales homogéneos.

Capítulo 3

Estado del arte

En la robótica de inspección se implementa el tipo de locomoción dependiendo la conveniencia por las condiciones del terreno en el que se vaya a desplazar, sin embargo, se han venido implementando diferentes técnicas de locomoción bio-inspiradas, debido a la gran eficiencia que presentan tipos de locomoción como los que se mencionan en el presente capítulo. Existen prototipos desarrollados para la locomoción en superficies verticales que dependiendo de las condiciones de diseño, se han elaborado diferentes estrategias y configuraciones de prototipos que dan solución a problemas específicos en el campo. Teniendo en cuenta los trabajos analizados, se diseña un prototipo que permita el cumplimiento de los objetivos planteados, para ello se realizó un estudio de las diferentes configuraciones propuestas por los autores mencionados.

3.1. Locomoción en superficie horizontal

En relación a los robots hexápodos caminantes existen trabajos como «Behaviour-based modelling of hexapod locomotion» [Schmitz and Cruse, 2004], que compara los insectos caminadores y los robots de seis patas, explicando por que requieren un control simultáneamente de 18 o más articulaciones y además hay que tener en cuenta las condiciones externas como fricción y las pendientes cuando son impredecibles. Estos robots caminadores requieren ser adaptativos y dependiendo de la situación y el control de varios grados de libertad, como consecuencia el modelado de la locomoción de las patas teniendo en cuenta varios aspectos como el comportamiento en general de cualquier motor. Con el fin de lograr la orientación y regulación de la oscilación movimientos, el esquema más simple que se ha propuesto para modelar los movimientos de las piernas no acoplados mecánicamente corresponde a tres entradas sensoriales que corresponden a tres ángulos de las articulaciones de la postura actual donde tres entradas más reciben ángulos de destino que definen la posición deseada

en el momento de afirmar el efector final en la superficie de apoyo. En la figura 3.1 se muestra la secuencia de locomoción implementada en este prototipo y consiste en realizar cambios de postura dependiendo de las condiciones del terreno [Schmitz and Cruse, 2004].

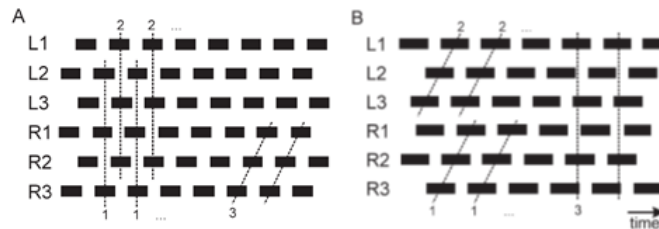


Figura 3.1: Posición deseada en el momento de apoyo. [Schmitz and Cruse, 2004]

También se han implementado algoritmos de control como el presentado en el trabajo «Complex-order dynamics in hexapod locomotion» [Silva et al., 2006] donde se estudia la dinámica de interacción que presentan las patas con respecto al suelo en los sistemas de locomoción hexápodo. El robot se caracteriza en términos de diferentes variables de locomoción y se presenta una propuesta para modelar el suelo a través de un sistema no lineal amortiguador-muelle. Por otra parte, se adoptó un algoritmo «foot-ground» de retroalimentación para controlar la locomoción del robot. Un conjunto de experimentos basados en el modelo revela la influencia de la locomoción en la velocidad de la función de transferencia «foot-ground», que presenta el orden dinámico complejo [Silva et al., 2006].

En otra publicación en el 2004, se presenta un proyecto titulado «Reactive free-gait generation to follow arbitrary trajectories with a hexapod robot», que desarrolla un controlador capaz de desplazar un robot con patas a lo largo de una trayectoria arbitraria con un alto grado de exactitud. Se han diseñado distintos módulos de nuestro controlador, para que puedan hacer frente a configuraciones arbitrarias que requieren las piernas del robot. De esta manera cualquier movimiento necesario para superar las irregularidades del terreno inesperados puede ser compensado correctamente por el controlador, mientras que se sigue la trayectoria establecida por el usuario. Los resultados muestran que al utilizar diferentes robots con patas y en diferentes entornos se puede comprobar el ajuste de nuestro enfoque. Una característica importante del estado de la marcha es (CCN “clockwise circularity number”), que se define como el número de las relaciones en el estado de avance [Porta and Celaya, 2004]. El CCN puede tomar cualquier valor entre un rango determinado con sus valores correspondientes para las secuencias establecidas y define una partición en el conjunto de estados de la

marcha. Está bien establecido que la llamada «wave gaits» o paso de onda, se observa a menudo en animales de patas que constituyen de la manera más eficiente y estable cuando se desplazan sobre una superficie plana. Wave Gaits se caracterizan por realizar el movimiento desde atrás hacia adelante por propagación de acciones paso a paso, que forman dos ondas, uno en cada lado del cuerpo con la misma frecuencia y en oposición de fase. Los pasos de patas adyacentes están alternadas y por lo tanto, asumiendo las prioridades de desempeño el paso solo cambia cuando el anterior es ejecutado, después se ejecuta el paso con la siguiente pata y así se lleva a cabo la secuencia de caminado.

La secuencia de locomoción en función del tiempo es descrita en la figura 3.2.

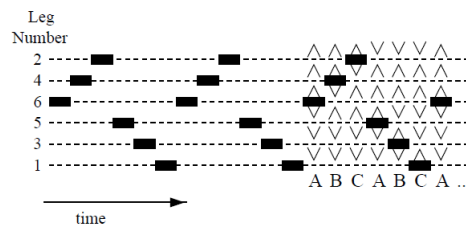


Figura 3.2: Marcha de onda lenta. [Porta and Celaya, 2004]

Los robots autónomos están dejando los laboratorios para dominar nuevas aplicaciones al aire libre y en particular los robots con patas los cuales ya han demostrado sus ventajas potenciales en estos ambientes, especialmente en un terreno de entorno natural como el mencionado en el artículo «Continuous free-crab gaits for hexapod robots on a natural terrain with forbidden zones: An application to humanitarian demining» [Estremera et al., 2010], quienes han desarrollado una secuencia de caminado que se basa en dos normas empíricas que se derivan tres módulos de control que han sido probados tanto en virtud de simulación y por el experimento. El modelo geométrico del robot «SILO – 6» que es un robot caminante, se ha utilizado para fines de simulación, mientras el verdadero «SILO - 6» se ha utilizado en los experimentos. Este robot fue construido como una plataforma móvil con un sistema sensorial para detectar y localizar minas terrestres antipersonas en misiones humanitarias [Estremera et al., 2010]. En misiones humanitarias, un robot con patas podría realizar una marcha periódica, que puede ser implementada fácilmente y de manera eficiente. Sin embargo, cuando hay zonas prohibidas involucradas, el periodo de marcha puede ser muy difícil de mantener. En tal caso, lo mejor es utilizar un modo de marcha en el que se pueda mover cualquier pierna en cualquier momento, es decir, que el robot no está forzado a mantener una secuencia periódica de movimiento de las patas. Estas

secuencias de caminado, como se mencionó anteriormente, son conocidas como secuencias libres. Una selección preliminar de las elevaciones del pie y equilibrios han sido basadas en la marcha trípole alterno. La marcha trípole alterno puede alcanzar la velocidad máxima teórica de un hexápodo, y también es la marcha óptima desde el punto de vista de la estabilidad. Estas son las principales ventajas de nuestra aplicación, teniendo en cuenta que estáticamente los robots caminantes son intrínsecamente vehículos lentos, y se trata de una aplicación real en terreno irregular en la que debe garantizar la estabilidad del vehículo. Además, este método limita drásticamente número de diferentes soluciones disponibles cuando hay planificación de los movimientos de las piernas de un robot de seis patas. El uso de una secuencia de caminado sobre la base de trípole alterno, se puede simplificar en gran medida los instantes que se levantan los pies y puntos de apoyo, mientras que la secuencia de la pierna se vuelve trivial. En la figura 3.3 se muestra un instante de tiempo donde la secuencia de trípole alterno se hace evidente, mostrando la proyección del centro de masa dentro del polígono que forman las extremidades de apoyo.

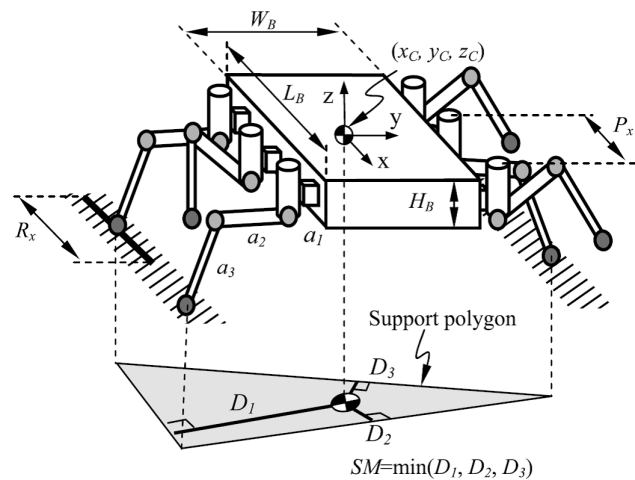


Figura 3.3: Trípole alterno y polígono de estabilidad. [Estremera et al., 2010]

Existe otra configuración como los radiales simétricos que han atraído la atención de la comunidad científica debido a su flexibilidad. No obstante, aún queda mucho por estudiar en su cinemática, dinámica y locomoción. En este trabajo titulado «Mobility analysis of the typical gait of a radial symmetrical six-legged robot» se estudia la cinemática de un robot de radio simétrico de seis patas con movimientos estáticamente estables. En los robots hexápodos se han introducido muchas maneras de caminar, sin embargo, la comparación de estabilidad entre cada una de ellas sigue abierto a la investigación. En este documento se analizan dos aspectos principales de la movilidad,

el primero se refiere a tres maneras estáticamente estables de caminar, la marcha en onda de insectos, la marcha de mamífero y la marcha mixta con el mismo factor de trabajo en la simetría radial del hexápodo. Se ha estudiado la estabilidad, eficiencia energética, flexibilidad al giro y la adaptabilidad al terreno ambiente [Wang et al., 2011]. En este documento se introdujo el modelo cinemático completo de un robot de seis patas. En este artículo se revisa el modelo cinemático del «NOROS» por completo y una forma compacta del modelo dinámico presentado en detalle. Tanto el análisis de la cinemática y la dinámica modelo se utilizará en el análisis de la movilidad y experimentos.

3.2. Locomoción en superficie vertical

Los robots escaladores bio-inspirados que dependen de sistemas de adhesión para su desplazamiento, se han convertido en una herramienta esencial para varias tareas en la industria creando aplicaciones para un futuro próximo. En los últimos años, la investigación se ha centrado principalmente en el modelado de fenómenos de adhesión de micro-escala y se han desarrollado un trabajos experimentales con el fin de definir los parámetros de un modelo específico. En el trabajo titulado «A Point-Wise Model of Adhesion Suitable for Real-Time Applications of Bio-Inspired Climbing Robots» se ha diseñado un macro-modelo de la adhesión que se ha implementado en un entorno de múltiples actividades basados en la dinámica del motor abierto (ODE) para simular un robot bio-inspirado. Simulaciones basadas en este modelo son adecuadas para representar el comportamiento de los robots escaladores y también para optimizar su diseño [Pretto et al., 2008]. Este tipo de robots con patas que dependen de adhesivos secos, requieren recargar sus patas contra la pared para aumentar el contacto de superficie y por lo tanto maximizar la fuerza de adhesión. Una inapropiada redistribución de estas fuerzas puede causar en el robot un desprendimiento de la superficie vertical. Este trabajo investiga la precarga óptima y una separación estratégica minimiza el consumo de energía, al tiempo que conserva la seguridad, durante la locomoción sobre superficies verticales. La marcha del robot de seis patas se diseñó usando un modelo cuasi-estático que toma en cuenta tanto la estructura del robot como las características del material adhesivo. Este último fue modelado a partir de datos experimentales obtenidos para este artículo. Se utiliza una optimización de rutina con restricciones, y su salida es una secuencia de la postura óptima y los valores establecidos del par en los motores [Boscariol et al., 2013]. La postura óptima del robot fue determinada por la aplicación del procedimiento de optimización en un entorno de MATLAB. La

optimización se obtuvo a través de un algoritmo de búsqueda de patrones. La figura 3.4 muestra la postura óptima del robot cuando se adhiere a una pared vertical con sólo 5 piernas. La primera pata se muestra en rojo, la que entra en fase de movimiento (la masa de las piernas se supone que son despreciables). La postura óptima se logra cuando el centro de masa estaba cerca de las patas delanteras y las piernas traseras extendidas. Con el fin de apoyar de manera óptima el peso del robot utilizando sólo 5 piernas [Boscariol et al. [2013].

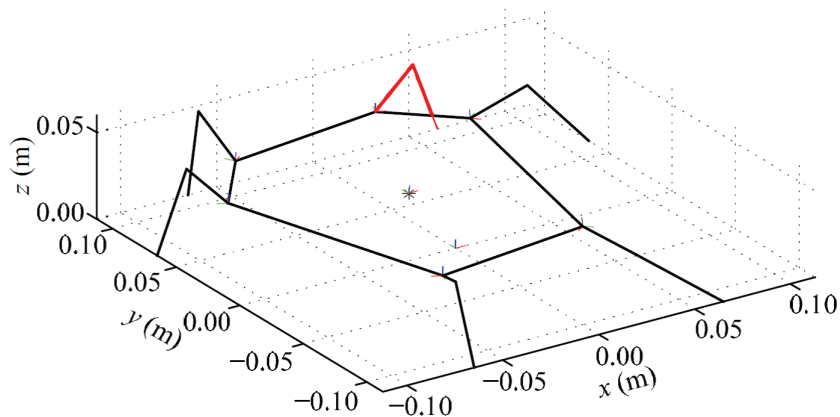


Figura 3.4: Postura óptima con adhesivos secos. [Boscariol et al., 2013]

Depende del tipo de sujeción de cada robot, existen ciertas ventajas y desventajas de cada sistema y uno de ellos es este robot que escala superficies verticales tipo Ventosas, desarrollado y probado [Yano et al., 1998]. El robot no está conectado al equipo de cómputo en el suelo físicamente. Dispone de dos bombas de vacío, una fuente de alimentación, un sistema de control y un sistema de comunicación inalámbrica, este robot resiste 30 minutos en movimiento. El espacio de trabajo del robot cubre 10 metros de alcance de control remoto. La velocidad máxima de marcha del robot es 30c m/min. Se espera que el robot desarrollado sea utilizado en inspección de edificios. El principal problema de un robot escalador con bomba de vacío externa es la limitación del espacio de trabajo. El espacio de trabajo está limitado por la longitud de los tubos y los cables. También es difícil para el robot, ir a lo largo caminos complicados. Incluso si el robot puede ir a lo largo de un camino complicado, el robot debe volver por la misma ruta para recuperar los cables. Además del problema anterior, existen los siguientes problemas por el hecho que el robot está conectado al equipo desde el tierra. A medida que los tubos y los cables se hacen más largos, su peso aumenta y la carga útil del robot se vuelve más pequeña y a medida que los tubos se hacen

más largos, el nivel de vacío en las ventosas disminuye por la longitud del tubo y las características que sobresalen del robot disminuyen [Yano et al., 1998].

Un experimento de simulación como el del robot cuadrúpedo caminador que asciende verticalmente superficies, deduce que con una programación lineal de caminado, la actuación del radio del robot es maximizada cuando las configuraciones de marcha del robot son dadas. Para la comparación de resultados, en la investigación es propuesta dos tipos de secuencias de caminado, cuando el ángulo de ubicación es de 0° y 45° , con respecto al ángulo formado con la dirección frontal y la dirección de propulsión. La matriz jacobiana del grupo rotacional de patas depende de que posición del rango de trabajo sea indicada. Por lo tanto, el ciclo de actuación es calculado para el punto medio óptimo [Hirose and Sato, 1989].

Otro trabajo realizado para locomoción en superficie vertical es el «A Six-Legged Walking Climbing Robot to Perform Inspection Tasks on Vertical Surfaces» en el que se presenta el análisis de la cinemática de un robot escalador llamado "Hex-piderix", el cual tiene 6 patas con 3DOF cada una, además cada pata tiene en el extremo una ventosa, con el fin de escalar superficies verticales. El autor propone un sistema de visión por computador utilizando análisis de imágenes para la inspección, este robot tiene una cámara montada en su tórax para capturar imágenes y propone un método invariante a la iluminación para detectar grietas en la superficie. En la figura 3.5 a), se puede observar el prototipo construido y en la figura 3.5 b), la secuencia de locomoción usada para el desplazamiento del robot [Sandoval-Castro et al., 2013].

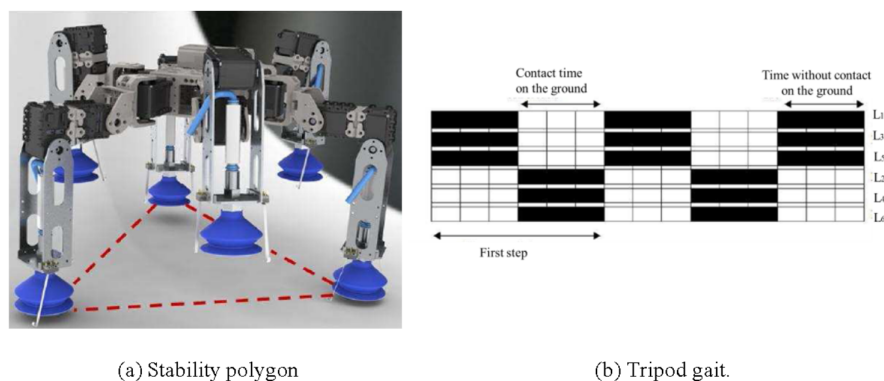


Figura 3.5: a) Robot Hexpiderix, Polígono de estabilidad. b) Secuencia de locomoción «Tripode alterno». [Sandoval-Castro et al., 2013]

De acuerdo a la bibliografía consultada, se realizó una extracción de las ideas principales, optando como mejor opción el sistema de sujeción tipo ventosa y secuencia

de locomoción trípode alterno, ofreciendo al robot una fuerte sujeción a la superficie. Dentro de los algoritmos de locomoción se encontraron diferentes formas de control para posturas eficientes de bajo consumo de energía y estabilidad, así como algoritmos para la sujeción en mallas para el caso de sujeción por clavos. En síntesis, no se encontró un algoritmo básico para el desplazamiento en superficies verticales de un robot hexápodo semi-autónomo que pueda seguir cualquier trayectoria con un perfil de posición en función del tiempo.

Capítulo 4

Diseño y construcción del prototipo

4.1. Configuración de las extremidades

Existen prototipos desarrollados tipo hexápodo para la locomoción en superficies verticales, que dependiendo de las necesidades y los objetivos de cada proyecto se han diseñado robots como el descrito en [Grieco et al., 1998] que tiene una configuración tipo SCARA, o el proyecto «Design of a Wall-Climbing Hexapod for Advanced Maneuvers» [Palmer et al., 2009] que son prototipos bio-inspirados así como la mayoría de los robots escaladores. Teniendo en cuenta los trabajos citados y otros ya estudiados en el estado del arte se quiere diseñar un prototipo que permita cumplir con las necesidades del proyecto propuesto y cumpla con los objetivos planteados. Para ello se hizo una recopilación de las diferentes configuraciones para cada una de las patas del robot a diseñar y se seleccionaron las más adecuadas que cumplan con los requisitos que son: desplazamiento en superficies verticales, transición de superficie horizontal a vertical y seguimiento de trayectorias.

La característica principal del prototipo es su movilidad bio-inspirada, lo que significa que su morfología tiene características parecidas a las de un ser vivo, en este caso, se tomó como referencia la cucaracha, ya que tiene la habilidad de caminar en superficies inclinadas y desplazarse en cualquier dirección, incluyendo la transición entre superficies ortogonales [Palmer et al., 2009]. La configuración de las patas del robot y los grados de libertad de cada una de ellas, se hizo observando el movimiento de las patas que realiza la cucaracha al desplazarse, como se muestra en la figura 4.1.

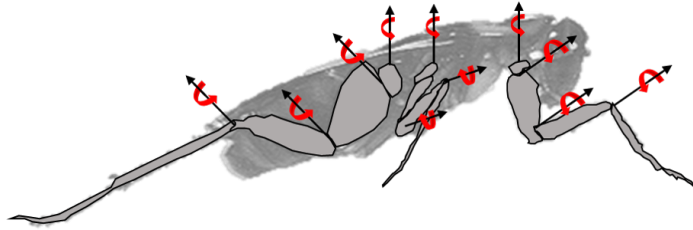


Figura 4.1: Configuración de las patas de la cucaracha.

El diseño de la configuración consiste en un mecanismo de seis cadenas cinemáticas abiertas unidas en un tórax rígido teniendo como característica que las dos patas delanteras junto con las dos patas traseras, ejercen la función de «Empujar-jalar», mientras que las patas de en medio ayudan al avance y permiten realizar la estrategia de avance trípode alterno. Las patas delanteras y traseras están formadas por cuatro articulaciones rotacionales, las tres primeras se encargan de ubicar la ventosa en el espacio y la última articulación la orienta como se explicará más adelante. Las dos patas de en medio, están formadas por tres articulaciones rotacionales, donde la última articulación, al igual que las otras patas, se encarga de orientar la ventosa. En la figura 4.2 se puede ver el esquemático de la configuración de las patas, las cuales están ubicadas de forma simétrica en el tórax.

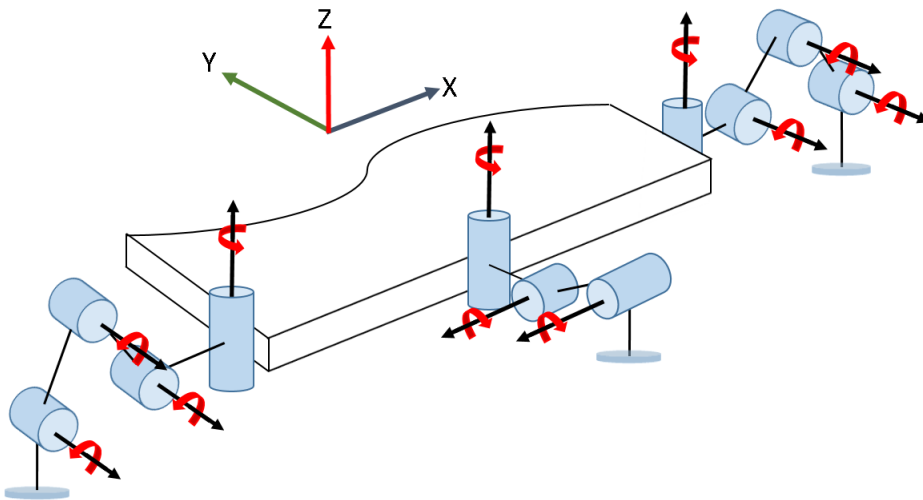


Figura 4.2: Esquemático de configuración de las patas.

4.2. Sistema de sujeción

Este robot es biológicamente inspirado en la morfología de la cucaracha que a diferencia de la configuración mencionada en el artículo “Toward a Rapid and Robust

Attachment Strategy for Vertical Climbing” [Palmer et al., 2010] , con estrategia de agarre hacia adentro con sus siglas en inglés (DIG), utiliza clavos como sistema de sujeción. En este proyecto se va a implementar un sistema de sujeción tipo ventosa, por lo que es necesario asegurar que la superficie de la ventosa llegue paralela a la superficie como se muestra en la figura 4.3 para asegurar que el vacío creado dentro de la ventosa no se pierda al momento que haya contacto con la superficie de apoyo.

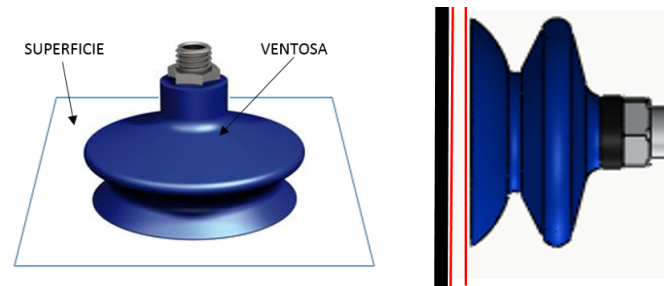


Figura 4.3: Ventosa paralela al plano de sujeción.

El diseño de las patas fue elaborado teniendo en cuenta que el robot se va a desplazar en superficies planas y totalmente lisas. Para lograr que la superficie de la ventosa llegue paralela a la superficie de apoyo, es necesario orientarla en cada punto de la trayectoria. Para lograr esto se utilizó un actuador que se encargará de mantener la ventosa con la orientación adecuada en cualquier instante de tiempo. Para comprobar la necesidad de este último actuador, se realizó el cálculo de los ángulos en los casos extremos en que se puede ubicar la pata del robot, en este caso sería la extensión máxima y la contracción máxima a la que podría llegar el mecanismo. En la figura 4.4 se pueden observar las posiciones máximas y mínimas calculadas con un programa de análisis geométrico llamado GeoGebra.

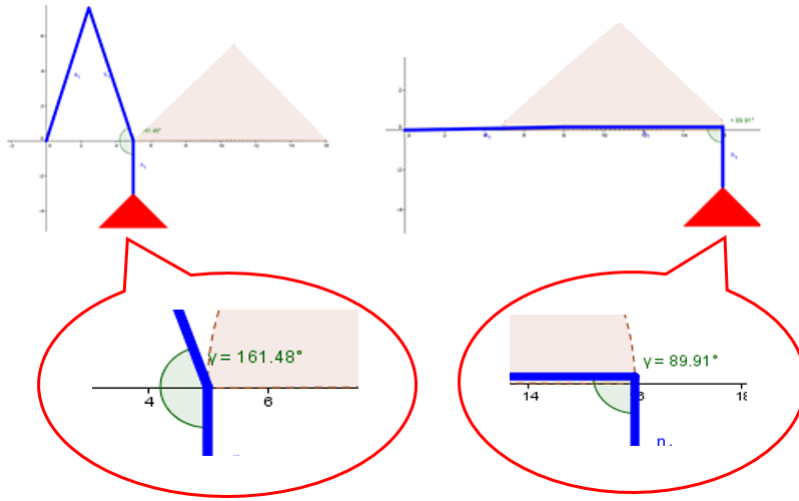


Figura 4.4: Recorrido angular de la ventosa.

El recorrido angular relativo entre los dos últimos eslabones del mecanismo es más de 70 grados, dependiendo del avance lineal con cada paso. El recorrido angular es directamente proporcional a la longitud deseada en el avance por cada paso.

4.3. Análisis cinemático de posición

Para la cinemática de posición se realizó el modelo geométrico inverso, de esta manera se encontraron los ángulos de cada motor tomando como referencia la base de cada cadena cinemática abierta. En el diseño del robot existen dos configuraciones de patas, las delanteras y traseras conforman el primer grupo y las laterales el segundo grupo, para cada una se hizo un análisis cinemático diferente. Para calcular el modelo geométrico en robots móviles es necesario tener un marco de referencia global, así como un marco de referencia para poder calcular el desplazamiento de todo el mecanismo en el espacio, de esta manera, se obtuvieron los siguientes marcos de referencia:

- Marco de referencia *Global*.
- Marco de referencia *Tórax*: se encuentra justo en el centro del tórax.
- Marcos de referencia *Locales*: indican el origen de cada una de las cadenas cinemáticas.

Para encontrar el modelo geométrico inverso es necesario conocer la ubicación del efector final, que en este caso será la ventosa. Es importante mencionar que la ventosa

es flexible, por lo que se tomará como referencia el centro de la ventosa y tendrán como nombre para el primer grupo de patas «ap» y para el segundo grupo de patas «as».

La ubicación en el espacio de cada uno de los marcos de referencia es conocida (Marco de referencia *Global*, *Tórax* y cada uno de los marcos de referencia *Locales*), así como los *ap* y *as*. En la figura 4.5 se puede observar la configuración de una de las patas que componen el primer grupo, donde «L» es el marco de referencia local y «ap» es la ubicación de la ventosa.

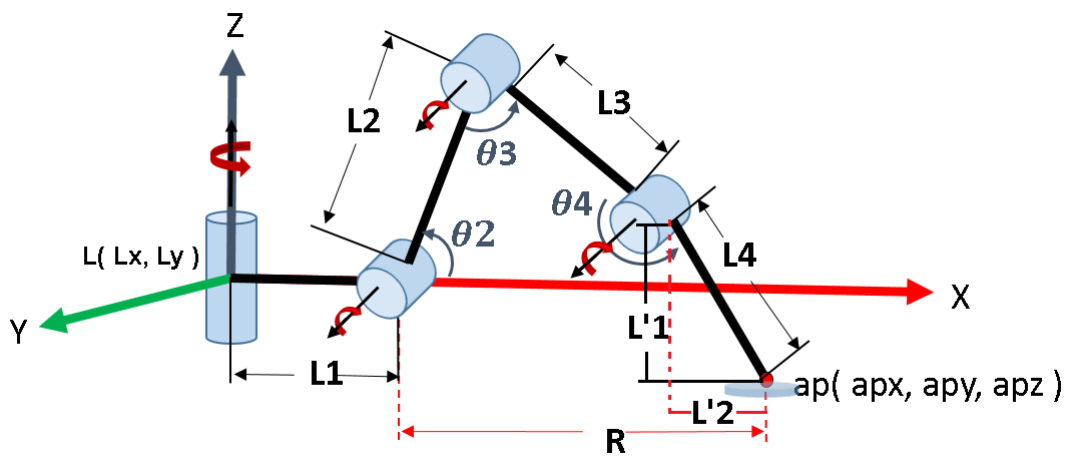


Figura 4.5: Modelo geométrico inverso «Patas Grupo 1».

Donde L_1, L_2, L_3, L_4 , son las longitudes de los eslabones, $L'1$ y $L'2$, son los catetos que componen la longitud del eslabón L_4 . $\theta_1, \theta_2, \theta_3, \theta_4$, son los ángulos y «ap» son las coordenadas de la ventosa. En la figura 4.6 se puede observar más claramente el ángulo θ_1 desde la vista superior. El eslabón L_4 es igual para las seis patas, lo que indica que $L'1$ y $L'2$ tienen los mismos valores en todas las ecuaciones.

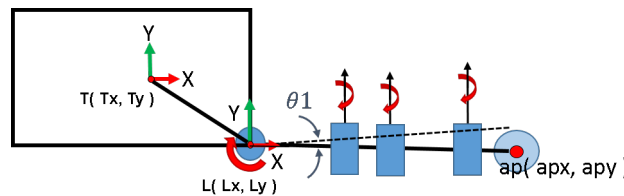


Figura 4.6: Vista superior «Patas grupo 1».

Las ecuaciones del modelo geométrico inverso para esta configuración son las siguientes:

$$R = \left(\sqrt{apx^2 + apy^2} \right) - L^2 - L1 \quad (4.1)$$

$$\theta_1 = \cos^{-1} \left(\frac{apy - Ly}{apx - Lx} \right) \quad (4.2)$$

$$\theta_2 = \cos^{-1} \left(\frac{R^2 + apz^2 + 2apzL'1 + L1^2 + L2^2 - L3^2}{2 * \left(\sqrt{R^2 + (apz + L'1)^2} \right) * L2} \right) \quad (4.3)$$

$$\theta_3 = \cos^{-1} \left(\frac{L2^2 + L3^2 + R^2 + apz^2 + (apz + L'1)^2}{2 * L3 * L2} \right) \quad (4.4)$$

$$\theta_4 = \cos^{-1} \left(\frac{-L2^2 + L3^2 + R^2 + (apz + L'1)^2}{2R^2 + 2 * (apz + L'1)^2 + 2L3} \right) + \cos^{-1} \left(\frac{R}{apz + L'2} \right) \quad (4.5)$$

Para el modelo geométrico inverso del segundo grupo de patas (patas laterales), se realizó el mismo procedimiento, en la figura 4.7 se ven más detalladamente las articulaciones y los ángulos a calcular.

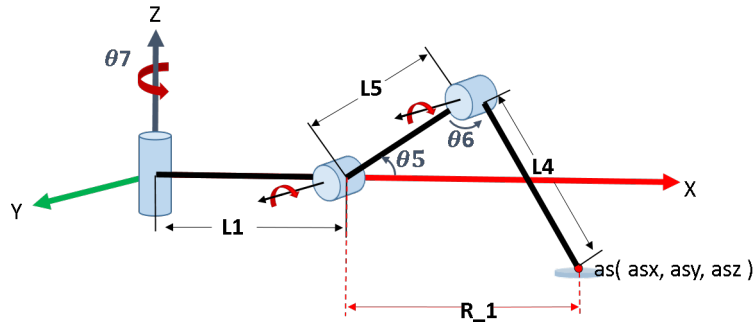


Figura 4.7: Modelo geométrico inverso «grupo 2».

Donde $L1$, $L5$, $L4$, son las longitudes de los eslabones θ_5 , θ_6 , θ_7 , son los ángulos a calcular y as corresponde a la ubicación del centro de la ventosa en el espacio. En la figura 4.8 se puede observar la descripción gráfica y la ubicación del ángulo θ_7 .

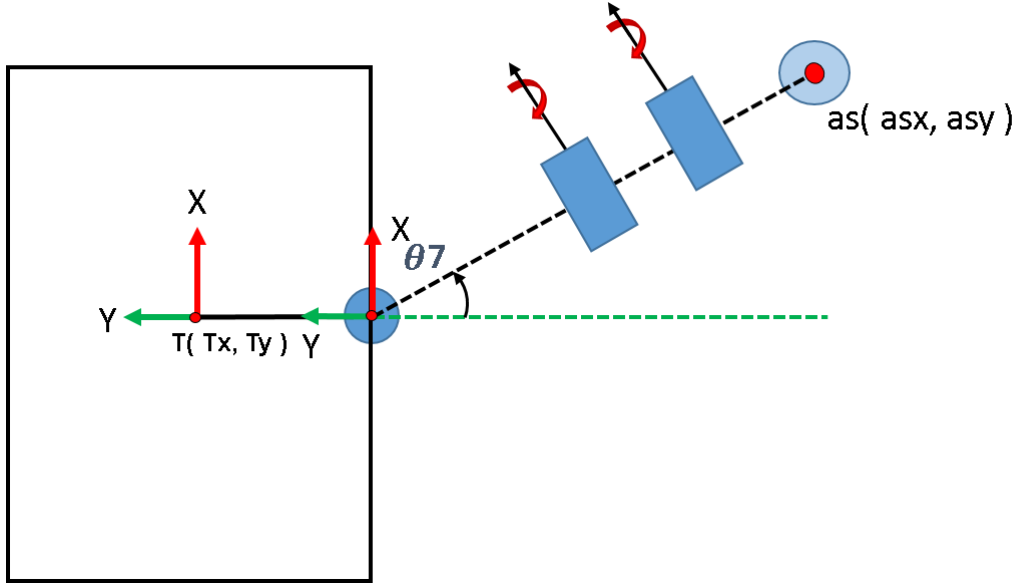


Figura 4.8: Vista superior «patas grupo 2».

Las ecuaciones correspondientes al modelo geométrico inverso para la anterior configuración son las siguientes:

$$R1 = \sqrt{asx^2 + asy^2} - L1 \quad (4.6)$$

$$\theta5 = \cos^{-1} \left(\frac{\left(\sqrt{asx^2 + asy^2} - L1 - L'2 \right)^2 - (R1 - L'2)^2}{2 * \left(\sqrt{asx^2 + asy^2} - L1 - L'2 \right) * L5} \right) \quad (4.7)$$

$$\theta6 = 90 - \theta5 \quad (4.8)$$

$$\theta7 = \cos^{-1} \left(\frac{asx}{asy} \right) \quad (4.9)$$

Conociendo la ubicación del marco de referencia local y los puntos as y ap , podemos calcular cada uno de los ángulos de las articulaciones por medio del modelo geométrico inverso.

4.4. Equilibrio de momentos

Para el cálculo de equilibrio de momentos, se consideró la posición de contracción máxima de la pata y la contracción mínima donde se puede evaluar el par que requieren los motores para el equilibrio estático

Las dimensiones de los eslabones se determinaron suponiendo un peso máximo del robot de 2.5Kg y asumiendo que cada pata soporta un tercio del peso debido a la secuencia de locomoción a implementar llamado trípede alterno. Este cálculo se incluyó en el diseño para tener una idea de los momentos que se iban a generar en los motores y conocer el par necesario para el desplazamiento del robot en superficie vertical. Sabiendo que $M_{(Par-motor)} = \vec{r} \times \vec{F}$, $\sum M = 0$ y que las fuerzas netas externas deben ser iguales a cero se procedió a hacer el cálculo de los momentos con respecto al eje Y.

Para el primer instante de tiempo, con la pata 1 totalmente estirada se tiene que $M2 + M3 + M4 = 0$; para el mismo instante de tiempo donde la pata 5 está totalmente retraída se realizó el cálculo nuevamente sabiendo que $M6 + M7 + M8 = 0$. Por último para calcular el par requerido en las articulaciones con respecto al eje Z del tórax decimos que $M1 + M5 + M9 = 0$.

En la siguiente tabla se puede observar los resultados del cálculo de equilibrio de momentos en cada a motor:

RESULTADO DE EQUILIBRIO DE MOMENTOS	
M2	0.0919 kg*cm
M3	0.76 kg*cm
M4	1.19 kg*cm
M6	0.919 kg*cm
M7	2.163 kg*cm
M8	1.19 kg*cm
M1	1.84 kg*cm
M5	1.84 kg*cm
M9	7.326 kg*cm

Podemos concluir que el máximo par requerido para el desplazamiento del robot en superficie vertical, es de 7.362kg*cm y se encuentra en el motor número 9, en la figura 4.9 se muestra el nombre de cada articulación.

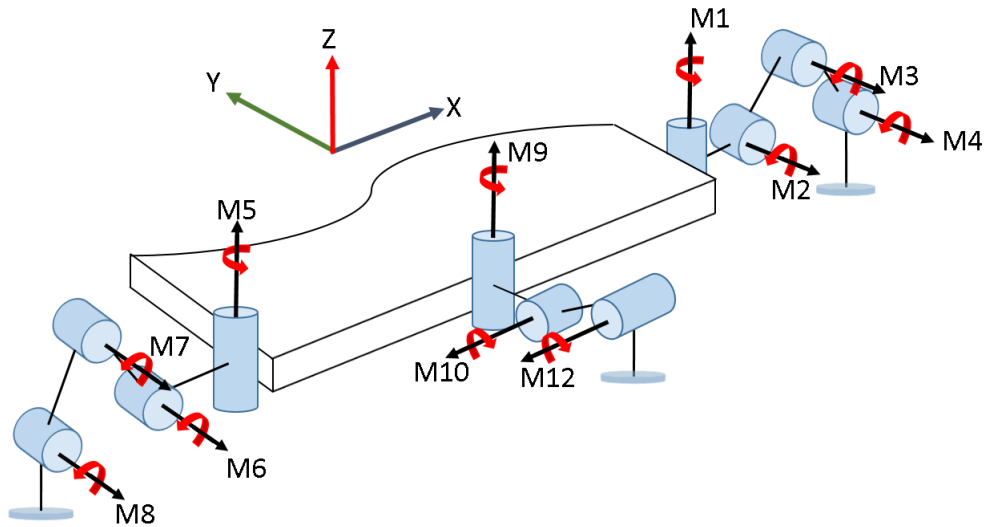


Figura 4.9: Esquemático con ubicación de los motores.

4.5. Modelado CAD

4.5.1. Modelado CAD SolidWorks

El diseño se apoyó con la elaboración del prototipo en un software de diseño asistido por computador (CAD). El software utilizado se llama Solidworks, en el cual se modelaron las partes del robot pieza por pieza y se realizó el ensamble correspondiente. Con la ayuda de este programa se analizaron posibles colisiones entre los eslabones y se definieron las medidas de cada parte a fabricar. En la figura 4.10 se muestra las vistas frontal, superior, lateral e isométrica del robot.

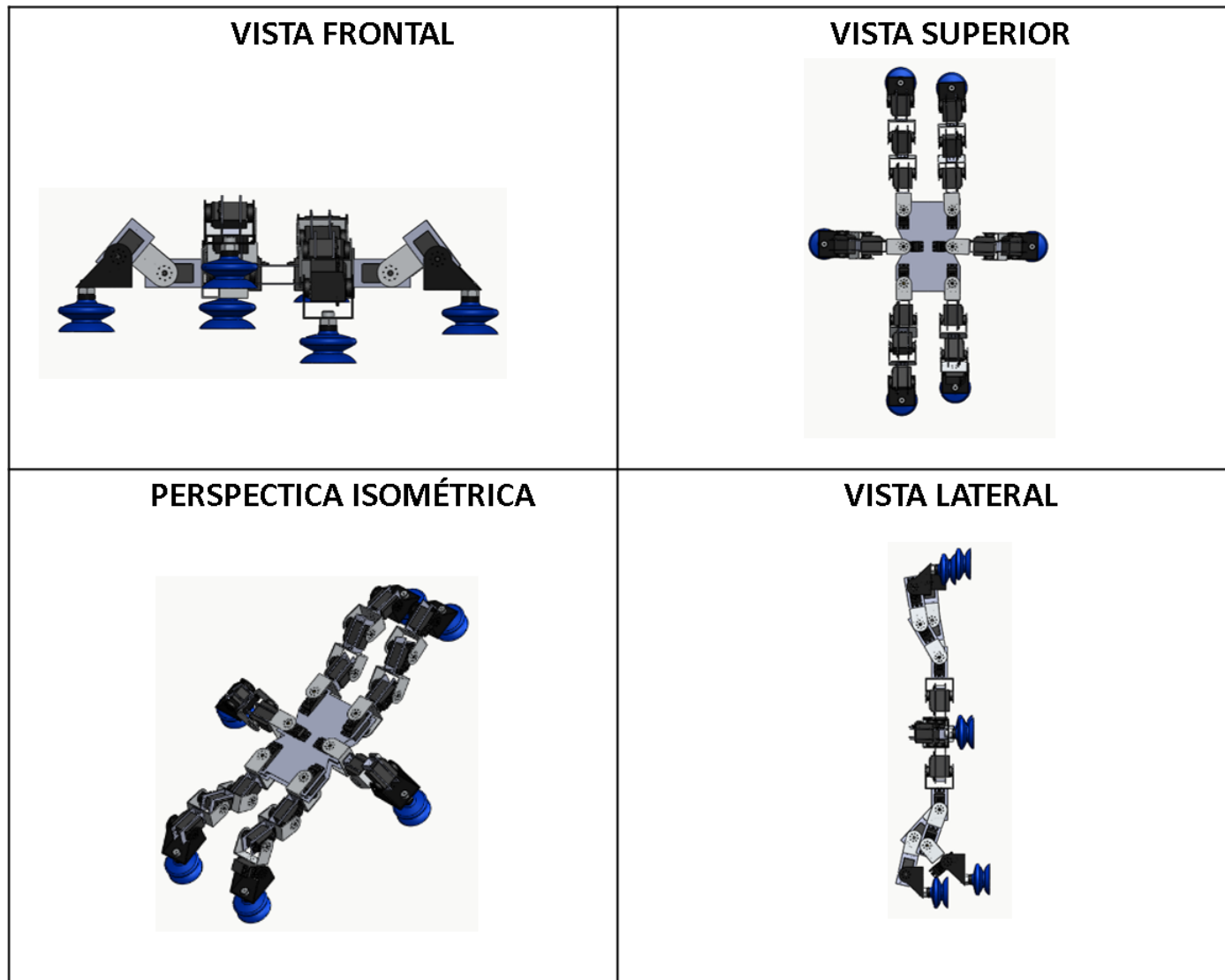


Figura 4.10: Modelado CAD.

Además de los análisis de movilidad realizados en SolidWorks, también se realizó la animación de la estrategia de locomoción llamada trípode alterno, donde se comprobó que el robot podría realizar los movimientos para ejecutar la secuencia de marcha. Otra animación realizada en SolidWorks, es la animación de la trayectoria individual que tienen las patas, en la figura 4.11 se pueden observar tres pasos en la secuencia de locomoción, donde la forma que sigue cada pata individualmente es triangular. Donde a) en la figura 4.11 corresponde al tiempo cero, b), corresponde al tiempo final de apoyo y c) corresponde al tiempo de máxima elevación de la pata.

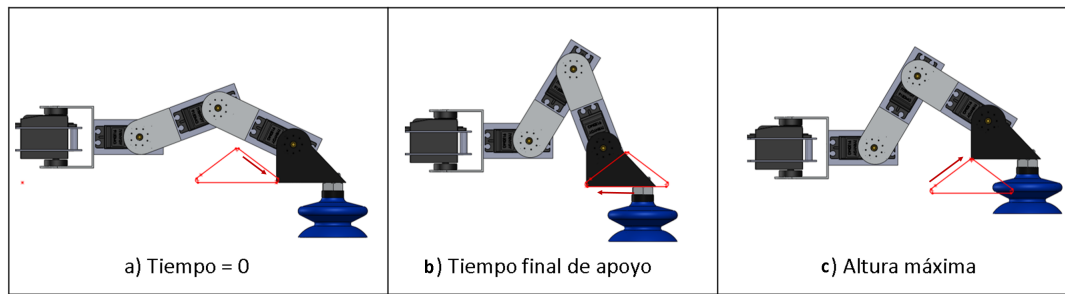


Figura 4.11: Trayectoria del paso.

La secuencia de locomoción es trípede alterno y para facilitar la comprensión de dicha secuencia se han separado las patas en dos grupos, los cuales se pueden apreciar en la figura 4.12.

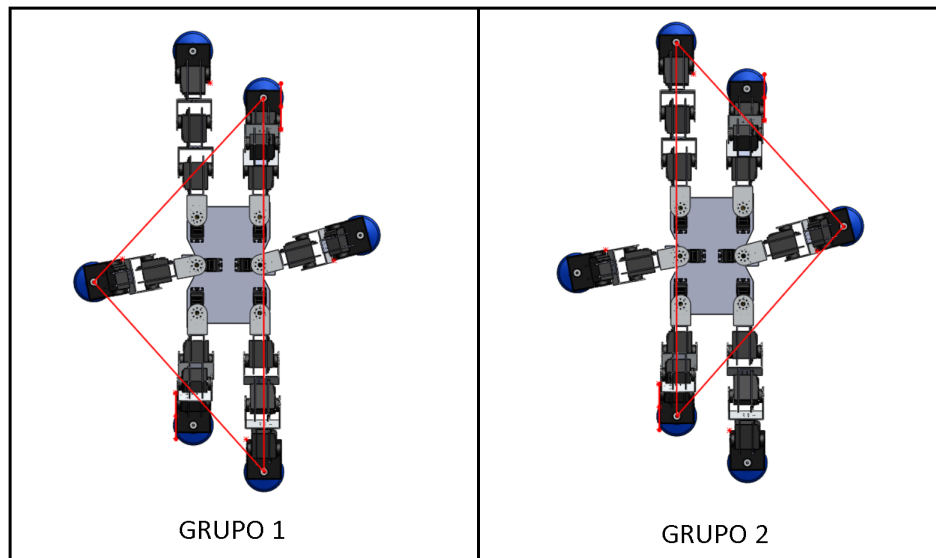


Figura 4.12: Grupos de patas.

4.5.2. Modelado CAD ADAMS

Además del diseño en SolidWorks, también se realizó un análisis en el programa Adams, donde se importó el ensamble desde SolidWorks y se especificó para cada una de las piezas el material y se definió en cada articulación, los motores reales que tiene el prototipo, con el fin de realizar los siguientes análisis:

- Análisis de movilidad «Traslación»: En este análisis se verificó que el robot tuviera la capacidad de trasladar el tórax en los tres ejes «X, Y, Z», para ello

se trasladó el tórax, dejando tres de las patas fijas a la superficie de apoyo. En la figura 4.13 se puede observar las pruebas realizadas de movilidad del tórax.

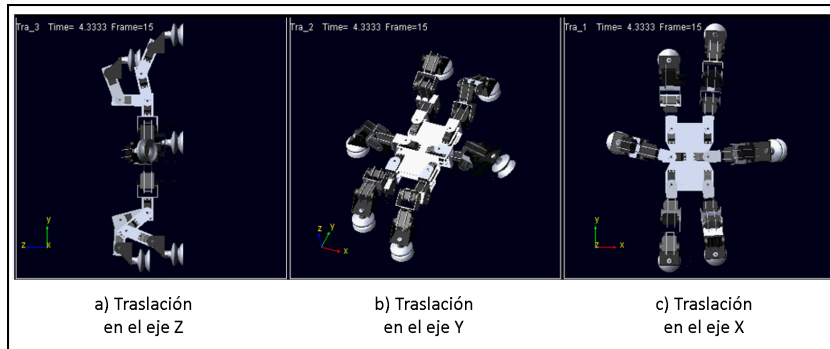


Figura 4.13: Pruebas de traslación del tórax.

- Análisis de movilidad «Rotación»: De la misma manera que para el análisis de traslación, se dejaron tres patas fijas y se comprobó que el tórax tuviera la capacidad de rotar en los tres ejes.

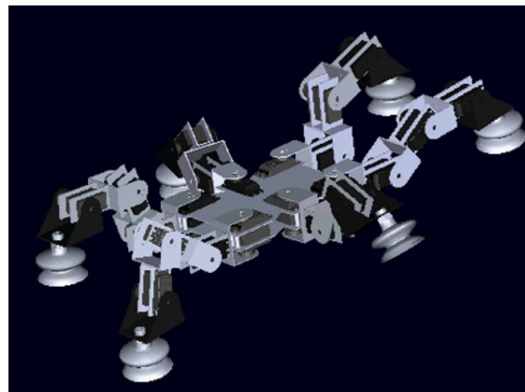


Figura 4.14: Rotación del tórax.

- Análisis de par en los motores: Para el análisis de par en los motores, se hizo una simulación de un paso completo del robot, obteniendo las curvas del par en cada uno de los motores en función del tiempo. En la figura 4.15 se puede ver la posición inicial y la posición final en el avance del tórax con respecto al tiempo.

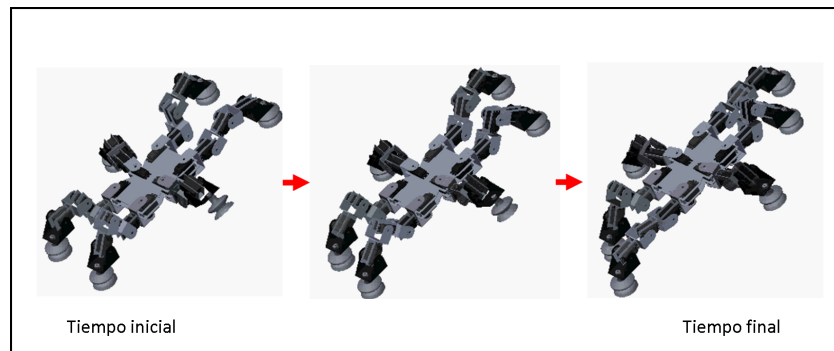


Figura 4.15: Simulación de un paso.

Una de las herramientas que ofrece este programa, es poder hacer análisis dinámicos, considerando las densidades reales de ciertos materiales, momentos de inercia y fuerzas que interactúan en el mecanismo durante un movimiento. En la figura 4.16 se puede observar las gráficas de par de todos los motores.

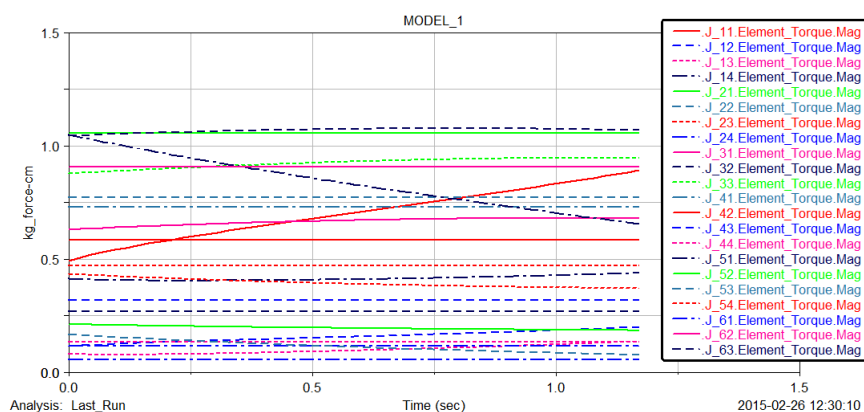


Figura 4.16: Gráfica de par en los motores para un paso en superficie vertical.

Cada una de las líneas graficadas corresponde al par realizado por motores del robot, donde el par máximo no supera un valor de $1.2\text{kg}\cdot\text{cm}$.

4.6. Selección de componentes

Para la selección de componentes se analizaron las alternativas de componentes comerciales y se realizó un diagrama donde se muestra de manera general cada parte que compone el sistema, en la figura 4.17 se puede observar dicho diagrama y a continuación se explicará cada una de ellas.

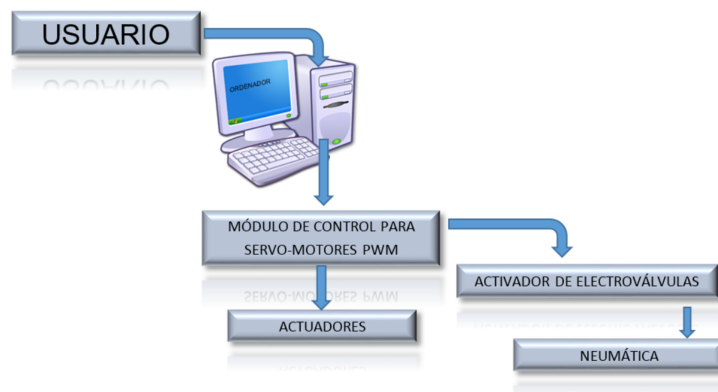


Figura 4.17: Diagrama de componentes.

- Ordenador: Se implementará como herramienta para ejecutar el software que nos permita manipular los servomotores por medio del módulo de control, de igual manera se enviarán señales indicadoras para cada una de las electroválvulas y sincronizar la marcha con la activación y desactivación de las electroválvulas.
- Módulo de control: Se manipularán los motores por medio de un módulo de control PWM llamado Mini Maestro de 24 canales, el cual está diseñado para el uso dedicado de control de servo motores con características tales como una interfaz USB incorporada y con control de secuencias de comandos interna. La resolución $0.25\mu\text{s}$ con una función de velocidad y la aceleración por control de la frecuencia del pulso que puede variarse hasta 333 Hz en cada canal. Es un dispositivo compacto y versátil. En la figura 4.18 , se muestra el dispositivo con el cual se va a realizar el control de los servo-motores [Pololu, 2014].



Figura 4.18: Mini Maestro 24 canales [Pololu, 2014].

- Actuadores: Como actuadores se escogieron los servomotores Turnigy, Modelo TGY-S901D, los cuales tienen un módulo de control interno para facilitar su

manipulación. Estos motores se activan con una señal PWM y su par es de hasta 13 kg*cm, con un voltaje de operación de 6.0V - 7.2V, velocidad de 6.54 rad/seg - 7.48 rad/seg, dimensiones de 40.8X20.1X37.6mm, peso 58g, con engranes metálicos y rodamientos tipo «cojinete de esferas». Además tienen un soporte en el lado opuesto al eje de rotación, lo que facilita su acoplamiento. En la figura 4.19 se puede ver el motor seleccionado.



Figura 4.19: Motor Turnigy.

- Electroválvulas: Las electroválvulas que se usarán serán las VP342-5DZ1-02A de FESTO, que por medio de la señal del módulo de control, se activarán y desactivarán permitiendo el flujo de aire a los generadores de vacío de manera adecuada, para lograr la adherencia necesaria en las ventosas. En la figura 4.20 se puede ver una imagen de la electroválvula usada.



Figura 4.20: Electroválvula FESTO.

- Conector RAP con rosca exterior: Acople FESTO para manguera 6mm.
- Generador de vacío: VN-05-M-I3-PQ2 FESTO, Generador de vacío basado en el principio de Venturi.



Figura 4.21: Generador de vacío FESTO.

- Ventosa: Las ventosas usadas permiten adherirse a la superficie por medio de vacío, se usará la ventosa de FESTO VASB-55-1/4-SI, con fuelle para adaptarse a las pequeñas irregularidades de la superficie de la cual se sujeta. En la figura 4.22 se puede observar la ventosa a usar en el robot.



Figura 4.22: Ventosa FESTO.

4.7. Activación de electroválvulas

Para activar las electroválvulas, se requiere una corriente de 375mA a 12v, por lo que estas electroválvulas tienen una bobina que es la encargada de accionar su mecanismo. En vista que el módulo Mini Maestro puede manejar salidas digitales y la corriente máxima soportada por el puerto de salida es de 20mA a 5v, se usaron estos datos para diseñar un circuito de activación para las electroválvulas amplificando el voltaje y a la vez proporcionando la corriente necesaria para activar el efector final. Este circuito consiste de un optoacoplador 4n35 y dos transistores en configuración Darlington para suministrar la corriente necesaria. El transistor final es un Tip31, el cual soporta una corriente máxima de hasta 1 A, lo cual es tres veces mayor a la corriente requerida por la electroválvula. En la figura 4.23 se puede ver el esquemático del circuito diseñado.

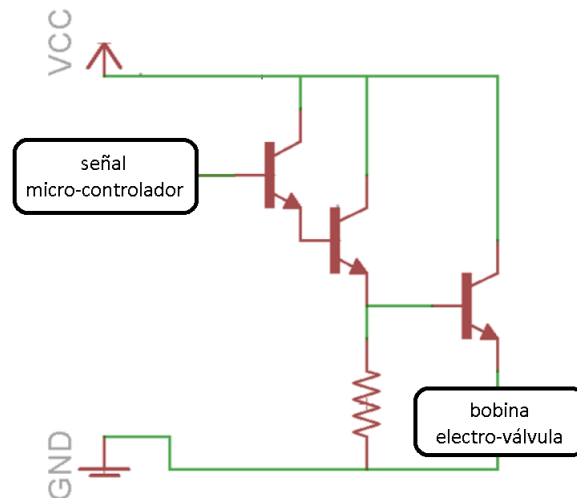


Figura 4.23: Esquemático optoacoplador.

Este circuito demanda unicamente 0.5mA, lo cual es conveniente ya que el micro-controlador no se calentará por corrientes excesivas.

4.8. Construcción

Los robots móviles, debido a que pueden estar en ambientes peligrosos o corrosivos, necesitan proteger sus circuitos de control y electrónica, por este motivo, fue necesario utilizar una caja metálica la cual se ubicó en la parte central-delantera del tórax en donde se encuentra la electrónica del robot. Esta caja fue forrada internamente con un material aislante, con el fin de evitar daños a los circuitos electrónicos. Finalmente se decidió introducir el cableado de cada uno de los motores entre las dos láminas que componen el tórax, para evitar su rose en la superficie por la cual se desplaza.

Ya construido el robot como se puede ver en la Figura 4.24, se realizaron pruebas de funcionamiento y se verificó que las medidas de cada pieza fueran las deseadas.

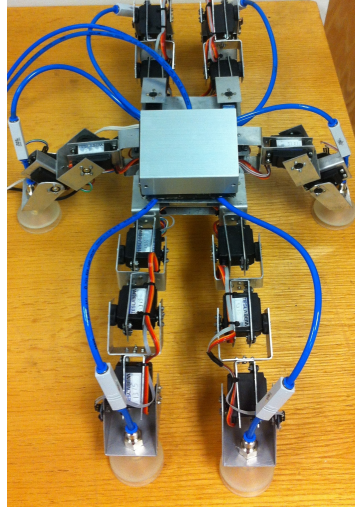


Figura 4.24: Ensamblaje del Robot.

Cada una de los eslabones del robot fue construido a mano con el fin de hacer ajustes a cada una de las piezas en caso de ser necesario.

4.9. Alimentación Servo-Motores

Los servo-motores utilizados en el robot tienen un par de 12 Kg*cm, su consumo de corriente es de 350 mA y su voltaje de alimentación es de 6V, este valor de corriente de consumo proporcionado por el fabricante “TURNIGY”, puede variar dependiendo de la carga de cada motor, por este motivo es de gran importancia considerar una fuente de alimentación que pueda proporcionar la corriente necesaria a los 22 servo-motores que se están utilizando en el robot. Es importante tener en cuenta que cada motor puede aumentar su consumo hasta 500 mA, por este motivo es recomendable no utilizar velocidades máximas en los servo-motores y realizar movimientos donde la carga se encuentre lo más uniformemente distribuida entre cada una de las patas que se encuentran sujetas a la superficie, además, una de las consecuencias por no evitar el excesivo consumo de corriente en los servo-motores es el efecto llamado “Armónicos”, lo que ocasiona caídas de voltaje en la fuente y hace que los motores no funcionen correctamente.

Experimentalmente se comprobó que cada motor no funciona adecuadamente cuando el voltaje de alimentación disminuye los 3.5V y si tenemos en cuenta el efecto mencionado en el documento de Arrillaga, J and Garmendia [Arrillaga et al., 1994], donde mencionan que los armónicos generan una distorsión en la señal, pueden ocurrir caídas de tensión en pequeños instantes de tiempo donde los servo-motores pierden

control. Dando solución a este problema se decidió conectar directamente cada motor a la tarjeta de control Mini Maestro de Pololu, la cual trae un regulador de voltaje para los motores y reduce la distorsión generada por cada uno de ellos.

Capítulo 5

Descripción de la estrategia de locomoción

5.1. Concepto

La estrategia de locomoción a implementar será la de trípode alterno, ya que corresponde a uno de los métodos que ofrece mayor seguridad durante el desplazamiento. Además de la influencia que tiene la estrategia de locomoción en la velocidad de desplazamiento como lo mencionan en los artículos «"Complex-order dynamics in hexapod locomotion"» [Silva et al., 2006] y "Continuous free-crab gaits for hexapod robots on a natural terrain with forbidden zones" [Estremera et al., 2010], el algoritmo consiste básicamente en integrar una secuencia de locomoción con un método que permita controlar la velocidad del robot, creando un método general para que un robot se desplace en función del tiempo modificando únicamente el modelo geométrico inverso del robot que se quiere controlar.

Para explicar mejor la secuencia de locomoción realizada por el robot se hizo una descripción gráfica de la fase de apoyo de cada pata en función del tiempo, en la figura 5.1 se puede observar más detalladamente la fase de apoyo o ventosa fija y la fase de reubicación o ventosa móvil.

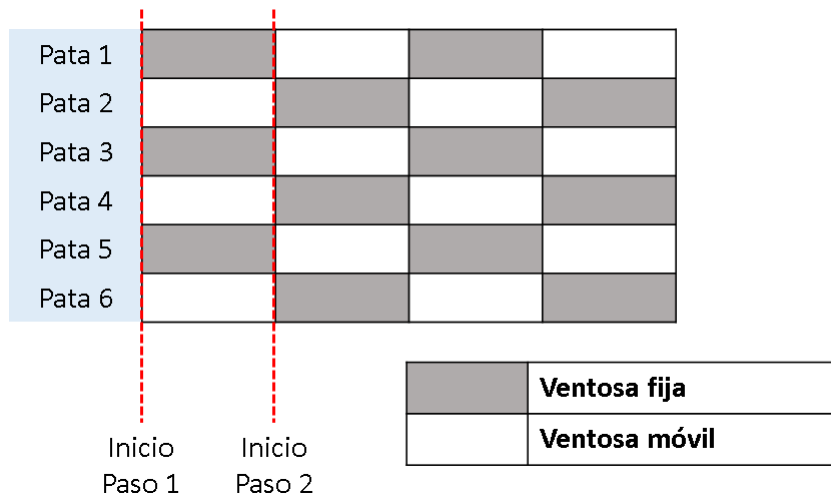


Figura 5.1: Diagrama de locomoción.

Donde el inicio del paso 1, es la misma posición inicial del robot y el inicio del paso 2, es cuando el segundo trío de ventosas se adhieren a la superficie. En la figura 5.2 se puede observar el inicio de cada paso.

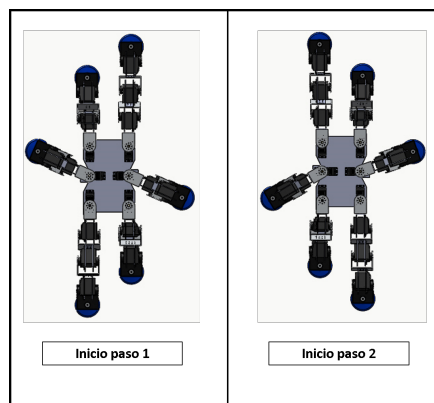
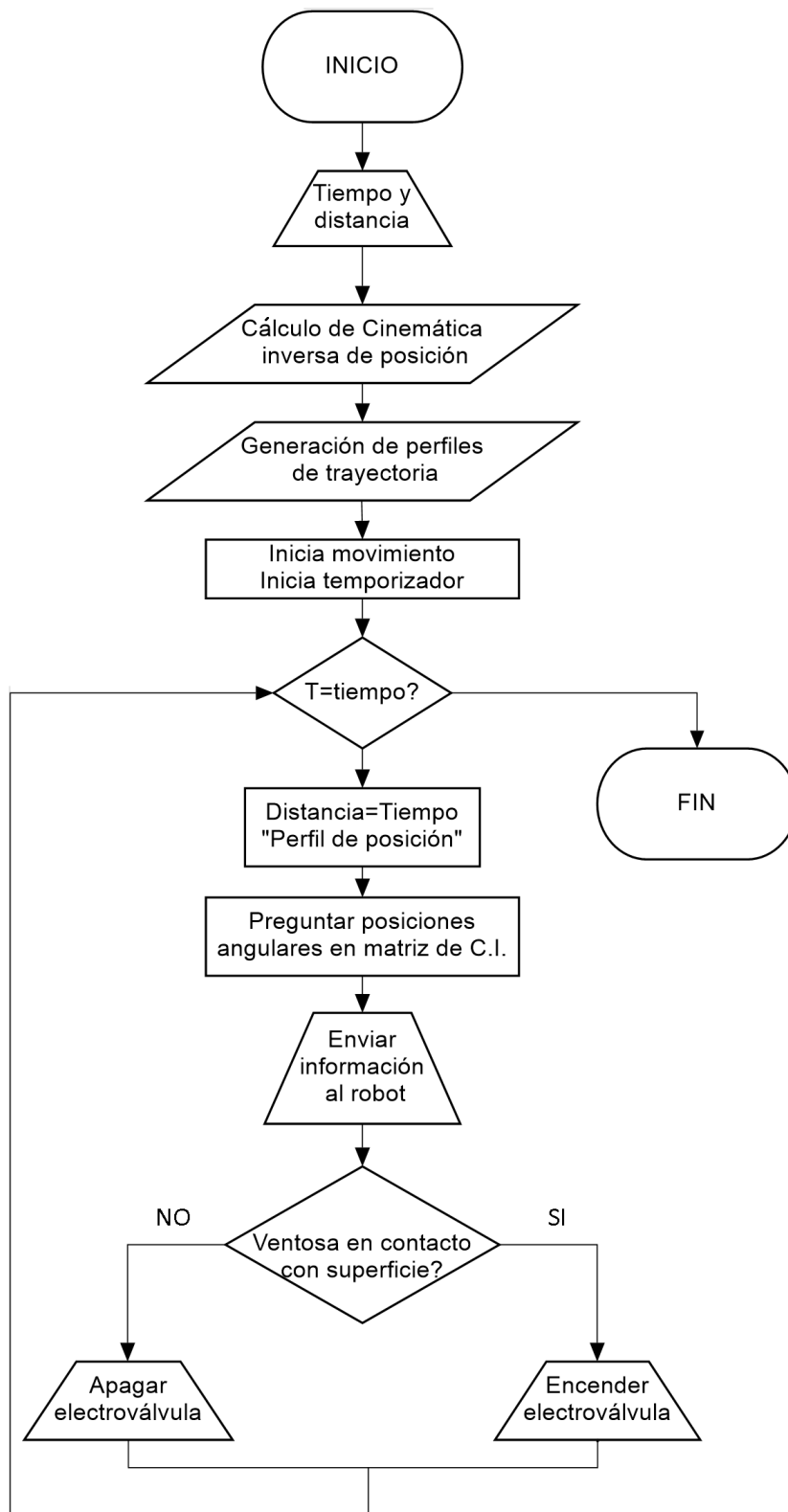


Figura 5.2: Inicio de paso con cada trío de ventosas.

5.2. Descripción del algoritmo por medio de diagrama de flujo

El método que se usa para realizar la locomoción del Wall-Bot, consiste en unos pasos básico que se describen en el siguiente diagrama de flujo.



5.3. Programación

La programación del algoritmo ha sido desarrollada en Matlab debido a la capacidad de cálculo que posee y a la fácil implementación de sistemas de comunicación con dispositivos periféricos. En general existen muchas maneras de hacer que un robot se desplace por cualquier superficie, lo importante es mantener un desplazamiento controlado y asegurando que el robot ejecute acciones deseadas por el usuario. Es necesario mencionar que los requerimientos básicos para generar los perfiles de la trayectoria son requeridos por el programa y las características físicas del robot están implícitas en la cinemática inversa.

Como primera medida, el programa inicia solicitando dos valores numéricos, “[Distancia, Tiempo]”, los cuales son datos necesarios para generar la trayectoria, donde «Distancia» es un valor en centímetros del desplazamiento deseado por usuario y «Tiempo» es el tiempo total que debe durar esta acción. Para generar la trayectoria se supone un caso muy específico, donde el robot se desplaza en línea recta y con una altura del tórax constante, además se establece como parámetro general posición, velocidad y aceleración inicial igual a cero.

Habiendo hecho las pruebas necesarias de cada una de las partes del programa principal se integran en un solo programa encontrando dos principales inconvenientes; el primero, es el cálculo de la C.I. y la conversión simultánea de grados al lenguaje MCC y el segundo inconveniente es la falta de una rutina de inicialización.

5.3.1. Cálculo de la cinemática inversa

La cinemática inversa de posición proporciona el ángulo que debe tener cada articulación conociendo la posición del efector final. Para este caso particular, la altura del tórax permanece constante a medida que se desplaza en línea recta. En total se están utilizando 22 motores y por lo que el tórax debe tener una orientación y posición respecto a un sistema de referencia, como salida obtenemos los 22 ángulos de nuestro sistema que ubican el tórax del robot de acuerdo a las condiciones especificadas.

El robot tiene un alcance de 9cm en cada una de sus patas, lo que debemos considerar para saber cuanto se puede desplazar en cada paso que realice, además, va a utilizar el método llamado trípede alterno, lo que indica que avanzará los 9cm, con tres patas a la vez. Como el desplazamiento se realizará en línea recta, únicamente se considera la variación en el plano en una dirección y para este caso particular se va a desplazar en línea recta.

Se introducen las ecuaciones de cinemática inversa a MATLAB y se considera el desplazamiento de cada pata, variando la posición del efector en pasos de 0.1cm. para obtener los valores de los ángulos y acomodarlos en una matriz llamada Matriz de Cinemática Inversa, “MCI” con el fin de obtener toda la información necesaria antes de ejecutar el programa, ver Anexo 1. Esta matriz se compone por un vector columna con la información de la distancia avanzada y el resto de valores son los ángulos correspondientes para cada pata de acuerdo al orden que se muestra en la figura 5.3

Distancia	Pata 1	Pata 2	Pata 3	Pata 4	Pata 5	Pata 6
0.1	0000	0000	0000	0000	0000	0000
0.2	0000	0000	0000	0000	0000	0000
⋮						
n	0000	0000	0000	0000	0000	0000

Figura 5.3: Matriz «MCI».

La matriz generada MCI contiene la información de todas las articulaciones del robot para cada 0.1cm que avance el robot, cabe mencionar que en función de la distancia, es posible modificar algunos parámetros como altura del tórax o rotación del mismo en cualquiera de sus tres ejes en el caso que se necesite modificarlos debido a los requerimientos del usuario.

5.3.2. Uso de la MCI en función del tiempo

La MCI tiene en su primera fila el indicador de la distancia recorrida, de esta manera se puede saber los ángulos de cada articulación para una distancia X recorrida. El principal objetivo de generar esta matriz, es poder enlazar la trayectoria generada anteriormente con los ángulos de los motores que están implícitos en la MCI. Si tomamos el perfil de posición calculado para una distancia X sabemos la distancia que debe haber recorrido el tórax del Wall-Bot en cualquier instante de tiempo. En la figura 5.4 se puede ver la relación de posición en función del tiempo.



Figura 5.4: Relación MCI y tiempo.

De esta manera se puede saber la distancia que debe haber recorrido el robot según el perfil de posición para cualquier instante de tiempo. Como ya se mencionó antes, la MCI tiene como primer vector columna el componente de distancia y el perfil de posición, según la trayectoria generada, también tiene como eje Y la distancia. Así que de esta manera se puede relacionar la MCI y el perfil de posición. MATLAB posee una herramienta de tiempo llamada TIC - TOC, que se usa como cronómetro para poder preguntar el tiempo cuando el usuario requiera, con el fin de comparar con el perfil de posición, saber la distancia recorrida y finalmente los ángulos que debe haber en cada articulación para un instante de tiempo específico.

Para el ejemplo mostrado en la figura 5.4, en el instante de tiempo 0.7seg, debe haber recorrido una distancia de 10cm y en la MCI, podemos obtener rápidamente los ángulos para cada articulación a partir de la primera columna.

5.4. Comunicación serie Matlab-MCC

El módulo MCC tiene dos puertos para la comunicación, el Serie RS-232 y el USB, en la figura 5.5, se puede observar el PinOut del módulo de control Pololu [Pololu, 2014] que se usa para controlar los servo-motores. El puerto de serial RS-232, es utilizado generalmente para la comunicación con dispositivos que no tienen comunicación USB, por lo tanto, se hará énfasis en la comunicación USB, ya que se pretende hacer los cálculos desde un computador y enviar la información a la tarjeta de control Pololu vía USB.

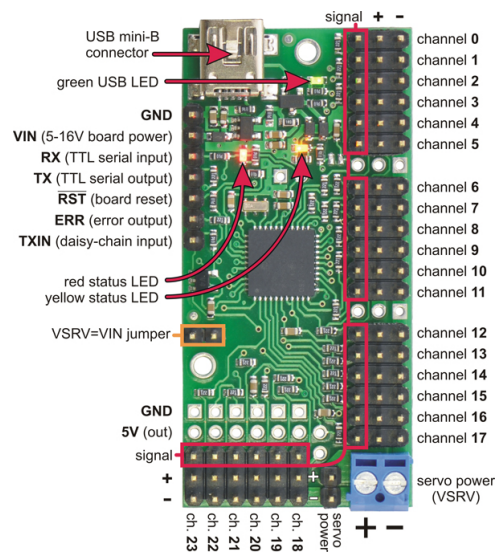


Figura 5.5: PinOut del Mini Maestro Pololu.

Para realizar la comunicación USB desde el computador es necesario instalar un controlador que se encuentra en la página de POLOLU [Pololu, 2014] y se especifican claramente los pasos necesarios para su instalación. Una vez instalado el controlador y el programa recomendado por el fabricante, puede hacerse uso de este módulo, teniendo en cuenta que debe aparecer en el administrador de dispositivos del PC, un puerto de comunicación serial virtual llamado “COM”. A continuación se mencionan tres maneras con las que se pueden manipular servo-motores desde la tarjeta Mini Maestro de Pololu.

5.4.1. Maestro Control Center “Frames secuencia”

El programa Maestro Control Center cuenta con una herramienta de fácil uso que permite modificar la posición de cada motor a partir de botones deslizantes, esta herramienta es muy útil si se quiere hacer una secuencia de movimientos relativamente corta y no es necesario conocer especificaciones técnicas de los motores, ya que la tarjeta se encarga de realizar la modulación del ancho de pulso para cada posición deseada. El método de programación utilizado en este método es experimental y no es recomendable usarlo si se quiere realizar rutinas de movimientos demasiado largas. En la figura 5.6 se muestra la interfaz gráfica del MCC y este método de programación fue utilizado para encontrar la posición inicial de cada motor.

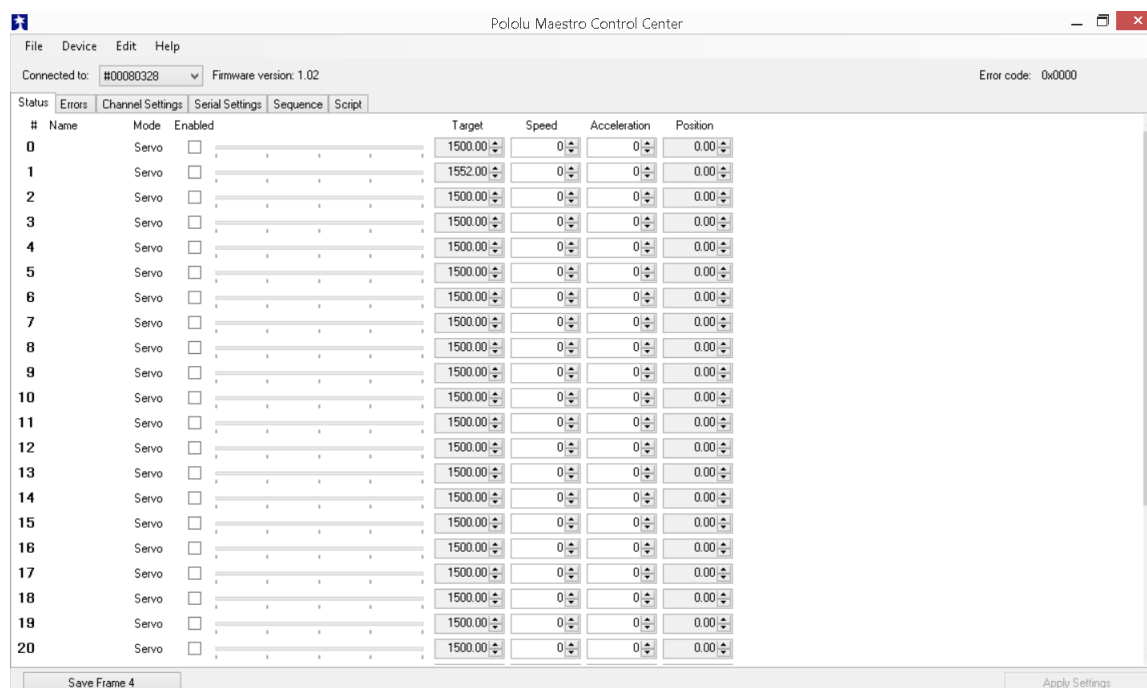


Figura 5.6: Interfaz del MCC.

En esta interfaz de programación se puede modificar posición, velocidad y aceleración, aunque la velocidad y la aceleración de cada motor es modificada por medio de un valor adimensional mostrado en un indicador propio de cada motor. Además se muestra un valor que indica la posición y que varía dependiendo el tipo de servo-motor.

Además de modificar la posición de los servo-motores desde esta interfaz, también indica la posición interpretada por el módulo MCC y si cada motor se encuentra activo o en reposo.

5.4.2. Protocolo de comandos MCC

El módulo MCC usa comandos propios de comunicación serie, el cual debe ser configurado previamente en la pestaña “Serial Settings” del MCC como USB Dual Port, en la interfaz de usuario. Para la comunicación serie de este módulo hay diferentes comandos que pueden usarse, para más información ver directamente en la página del fabricante Pololu [2014]. El comando usado para la comunicación serie se reduce a una sola línea de código, donde se debe enviar una cadena de caracteres en el siguiente orden: 0xAA, numdispos, 0x1F, numcanal, destinoLB, destinoHB. Como primer byte 0xAA (170 Dec.), seguido de un byte de datos con el número de dispositivo (se especifica en la interfaz de MCC), el siguiente byte indica la escritura de posición para cada motor, número de canales a usar, canal de inicio, 1 destino LB, 1 destino HB, 2 destino LB, 2 destino HB,.... n destino LB, n destino HB.

Además de canales de salida PWM, se puede configurar cada pin del microcontrolador como salida digital de una manera muy sencilla. Desde la interfaz MCC, en la pestaña “Chanel Seettings”, hay una opción para configurar cada pin, ya sea, como salida, entrada o Servo. El valor numérico que activa la salida digital en alto debe ser superior a “24” desde Matlab y desde el MCC debe ser superior a 1400. Usando estas salidas digitales, se puede activar y desactivar las electro válvulas, ya que por medio de la cinemática directa se sabe la posición de cada ventosa..

5.4.2.1. Bits de destino

El protocolo serial envía cadenas de datos de 8 bits, por lo que es necesario hacer un ajuste de datos. En la pestaña de estatus en el MCC, en la columna Target, hay un indicador de la posición de cada motor, el cual fue tomado como referencia para saber cómo enviar los datos de posición a los motores. El ajuste consiste en tomar el valor mostrado en el indicador MCC y multiplicarlo por 4, convertirlo en binario y separar

el valor en dos grupos de 8 bits, donde se debe identificar el bit más significativo y el bit menos significativo. En la figura 5.7 se puede observar la conversión y la separación de los bits para completar la trama de datos a enviar por el puerto serie.

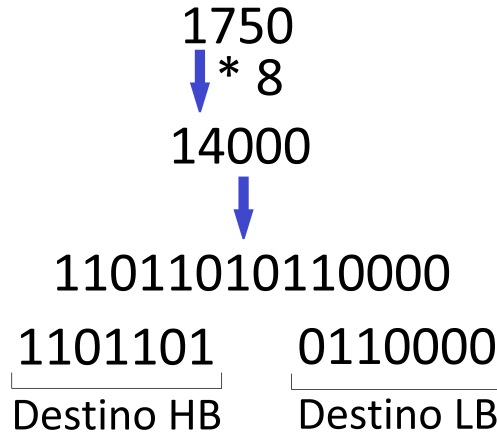


Figura 5.7: Ajuste de bits para comunicación serial.

El comando: “0xFF” - “direccioncanal” - “8-bit destino”, ajusta el canal con un valor de 8 bits de acuerdo al valor del rango del protocolo de comunicación serial.

5.4.3. Programación MCC desde Matlab

Para aplicar un algoritmo de locomoción con movimientos controlados es necesario realizar los cálculos pertinentes como los que se mencionaron en el capítulo anterior, la cinemática inversa que en conjunto con una trayectoria planificada conforman las bases fundamentales del algoritmo de locomoción, sin embargo, es necesario transmitir esta información a la tarjeta de control para realizar los movimientos deseados. Para que el algoritmo funcione correctamente, hay que establecer una posición inicial o “Home Position” que indica a nuestro programa, la posición actual en que se encuentra cada motor.

Como se mencionó anteriormente, la tarjeta de MCC de Pololu se encarga de hacer la modulación de ancho de pulso que requieren los motores para realizar cualquier acción, por lo tanto, hay que hacer una traducción de valores entre los ángulos obtenidos por la C.I. y el valor adimensional que requiere el la tarjeta de control, para esto se hizo una parametrización experimental de los rangos máximos y mínimos de cada motor, observando el valor numérico mostrado en la interfaz MCC.

Para poder enviar los datos calculados desde Matlab, es necesario tener en cuenta los siguientes aspectos:

- *Conversión de ángulos al lenguaje MCC:* Una vez identificado los rangos máximos y mínimos de cada motor, es necesario encontrar una relación entre el ángulo que se obtiene de la C.I. y el valor que hay que enviar a través del puerto serial. Para esto se utilizó la ecuación de una recta de pendiente « m » (ver eq. 5.1), donde la pendiente se obtiene a partir de los rangos máximos y mínimos obtenidos experimentalmente y los ángulos máximos y mínimos calculados [Thomas et al., 1959].

$$(Y - Y_1) = m * (X - X_1) \quad (5.1)$$

Para mejor comprensión, a continuación se muestra un ejemplo de la conversión de valores de uno de los motores:

$$m = \left(\frac{531-2072}{141-64} \right) = \frac{-1541}{-77} = -20,012$$

$$y - 2072 = -20,012 (x - 64)$$

$$y = -20,012x + 3352,83$$

Donde el valor numérico 3352.83, es el punto de origen d la recta y m es la pendiente. En la figura 5.8 se muestra la descripción gráfica del ejercicio desarrollado.

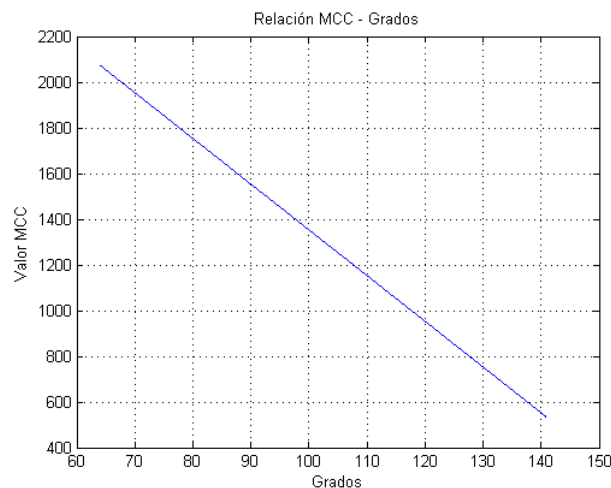


Figura 5.8: Relación MCC - Grados.

Este cálculo fue realizado para cada uno de los motores y poder determinar el valor numérico MCC, para cada posición angular requerida.

- *Envío de datos desde matlab al módulo pololu:* Para poder enviar los datos desde Matlab, se revisó el puerto virtual COM creado por los controladores del MCC, y se estableció como una variable global el comando serial, “s2=serial('COM6')”. Matlab tiene un comando de escritura para comunicación serial llamado “fwrite” el cual se encarga de enviar los datos por el COM virtual la cadena de datos con las posiciones de cada uno de los motores. La cadena de datos enviada tiene la siguiente estructura:

```
fwrite(s2,[170, 12, 31, 22, 0, ... ,1destLB,1destHB], 'uint8')
```

5.5. Cálculos fuera de línea

Un inconveniente cuando se hacen cálculos en línea es que estamos limitados a la capacidad de procesamiento que se esté usando. Por este motivo se decidió hacer los cálculos de cinemática inversa y de conversión de grados a MCC fuera de línea, esto quiere decir que antes de empezar la rutina de movimientos, se hacen todos los cálculos necesarios para que el robot se pueda mover, generando la matriz de ángulos de valores decimales en lenguaje MCC. El tamaño de esta matriz varía dependiendo la distancia que se va a desplazar el robot.

5.6. Secuencia de inicialización

Antes de ejecutar cualquier programa, hay que tener en cuenta que para que un robot pueda tener una alta repetitividad, la calibración que se necesita debe tener una precisión absoluta como lo explican en el artículo «Calibration of A 2-DOF Planar Parallel Robot: Home Position Identification and Experimental Verification» [Ding et al., 2005] y aunque para nuestro caso no se necesita una calibración tan precisa si es necesario encontrar la posición inicial de cada motor, en el caso del Wall-Bot, tiene 22 motores y para cada motor es necesario identificar el punto de partida o posiciones iniciales y finales. Esta identificación de posiciones iniciales se hizo experimentalmente, tomando los rangos máximos y mínimos de cada articulación teniendo en cuenta las posiciones que debe tener cada pata en el máximo alcance y la contracción máxima.

Como ya se mencionó anteriormente, hay que evitar consumo de corriente excesiva y una causa de estas corrientes es la activación de todos los motores al mismo tiempo. Si se ejecuta la rutina de movimientos sin saber la posición de cada motor, el robot va a tratar de ubicar todos los motores muy rápidamente lo que ocasiona una caída de tensión debido a los armónicos producidos [Arrillaga et al., 1994] y por lo tanto

la pérdida de control de los actuadores. Para solucionar este problema se hizo una rutina de inicialización, que consiste en ubicar manualmente cada articulación en una posición aproximada a la que se quiere llegar por medio de código. Esta posición se le llama estado de relajación, donde ninguna de las extremidades del robot se encuentra haciendo contacto con la superficie. Los motores se activan por medio de Matlab, uno a uno, evitando corrientes excesivas. Ya conociendo la posición de cada motor, se procede a ubicar cada extremidad en la posición de inicio, que es el el valor que debe tomar cada articulación en el momento justo antes de iniciar la secuencia de movimientos.

Capítulo 6

Validación

Para comprobar el funcionamiento del robot y el algoritmo de locomoción se realizaron las siguientes pruebas.

6.1. Simulación Matlab

Para comprobar el modelo geométrico inverso del robot, se realizó una simulación en el entorno de Matlab, en la figura 6.1-a) se observa cómo se generan los perfiles de una trayectoria y la posición inicial de una pata en tiempo cero o extensión máxima, en la figura 6.1-b) se observa media contracción de la pata, donde el paso está a la mitad de su recorrido y en la figura 6.1-c) vemos la posición final para un tiempo final, con la contracción máxima de la pata, donde el punto rojo, indica la distancia recorrida en función del tiempo.

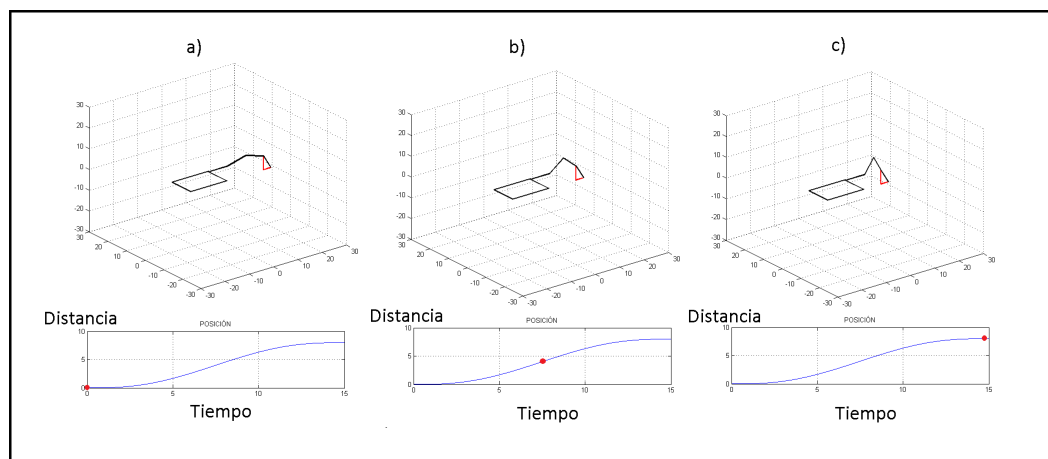


Figura 6.1: Simulación seguimiento de perfil posición.

Luego de haber comprobado el funcionamiento de las ecuaciones con una sola

pata, se procedió a hacer la simulación de las seis patas con las medidas reales que tiene el robot, la figura 6.2 muestra las tres posiciones más importantes que revelan la secuencia de locomoción que son: a) posición inicial, b) posición media y c) posición final.

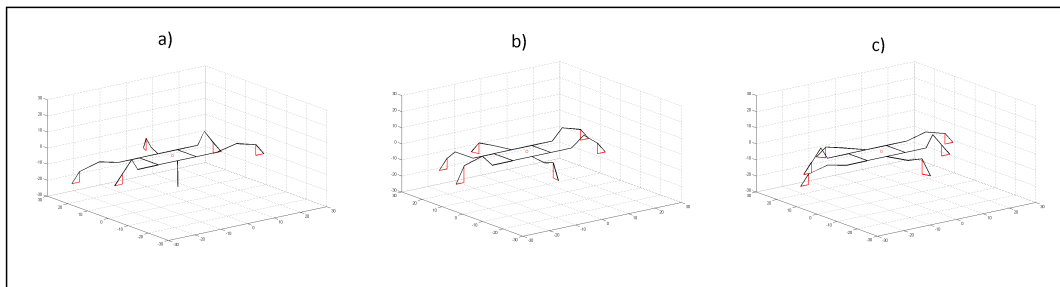


Figura 6.2: Simulación de secuencia de locomoción con medidas reales.

Una descripción numérica del comportamiento en cada una de las articulaciones del robot, es una tabla que se muestra a continuación donde se especifica la secuencia de movimiento de cada motor. En la tabla se tienen cinco cuadros que describen el comportamiento de cada motor al realizar un ciclo de avance que está compuesto por: 1) Ventosa sujeta a la superficie y 2) Re-ubicación de ventosa.

		SECUENCIA DE MOVIMIENTO				
		CUADRO 1	CUADRO 2	CUADRO 3	CUADRO 4	CUADRO 5
PATA 1	M1	0	0	0	0	0
	M2	17.9387	46.4186	64.1739	66.4967	17.9387
	M3	141.0051	82.7148	43.7617	88.8557	141.0051
	M4	19.4974	48.6426	68.1191	45.5721	19.4974
PATA 2	M5	-37.1042	-10.8855	21.0375	-10.1755	-37.1042
	M6	30.1891	30.1891	30.1891	4.5886	30.1891
	M7	90.0114	85.6114	90.1114	85.6114	90.0114
	M8	0	0	0	0	0
PATA 3	M9	65.182	66.7991	17.5156	65.8282	65.182
	M10	43.7141	89.2277	141.0725	89.8211	43.7141
	M11	68.143	45.3861	19.4637	45.0895	68.143
PATA 4	M12	0	0	0	0	0
	M13	18.3545	66.1271	63.1614	46.884	18.3545
	M14	140.9528	90.1931	43.823	81.7406	140.9528
	M15	19.5236	44.9035	68.0885	49.1297	19.5236
PATA 5	M16	21.0375	-10.1755	-37.1042	-10.8855	21.0375
	M17	4.5886	4.5886	4.5886	4.5886	4.5886
	M18	90.0114	85.6114	90.1114	85.6114	90.0114
	M19	0	0	0	0	0
PATA 6	M20	71.1039	66.7991	21.4118	50.8666	71.1039
	M21	43.7141	89.2277	141.0725	82.7148	43.7141
	M22	68.143	45.3861	19.4637	48.6426	68.143

Figura 6.3: Tabla de posiciones angulares.

La posición del robot en los cuadros desde el 1 al 5 se puede ver en las figuras del 6.4 al 6.8 respectivamente, donde los cuadros 1,2 y 3, componen la etapa de ventosa sujeta a la superficie, y los cuadros 4 y 5, la etapa de re-ubicación de la ventosa del trípode 1.

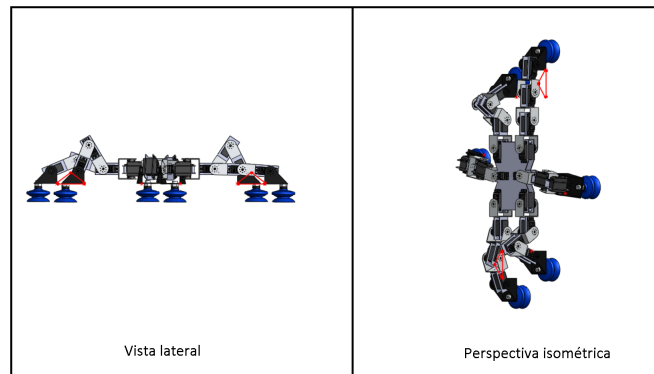


Figura 6.4: Cuadro 1.

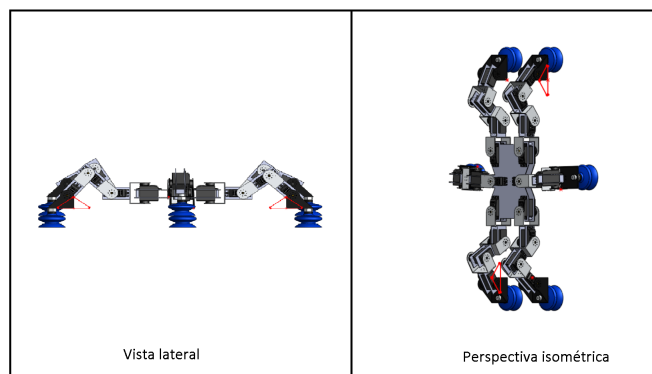


Figura 6.5: Cuadro 2.

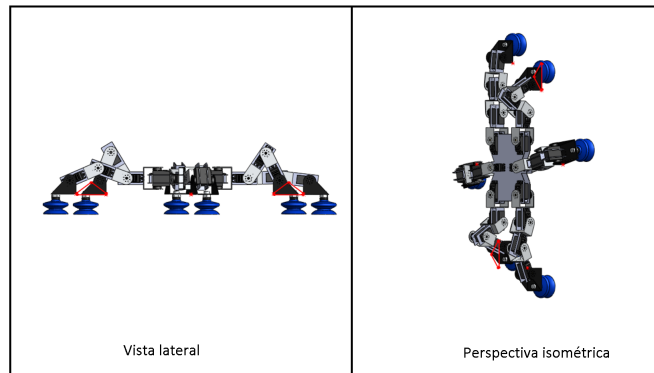


Figura 6.6: Cuadro 3.

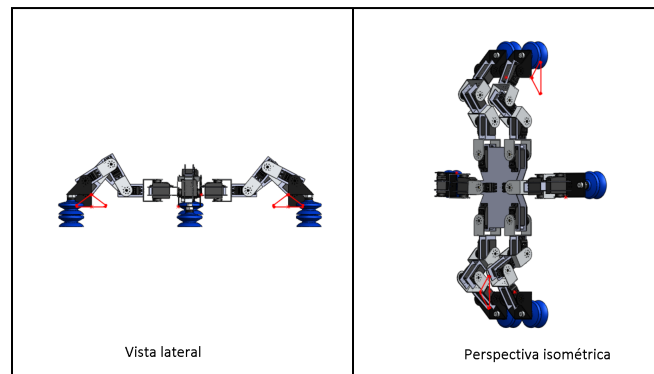


Figura 6.7: Cuadro 4.

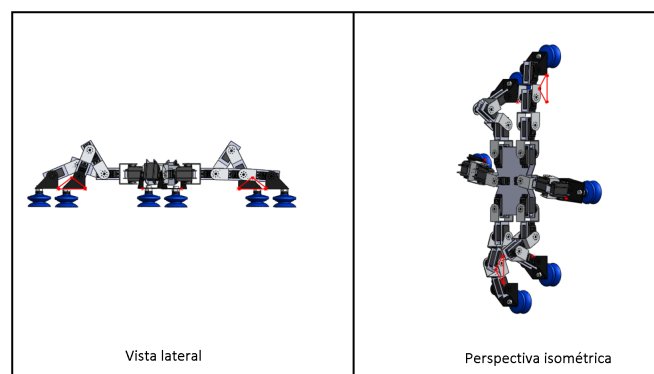


Figura 6.8: Cuadro 5.

6.2. Pruebas en SolidWorks

Una de las herramientas de SolidWorks es el análisis de movimiento, donde también se pueden extraer características del prototipo como colisiones entre eslabones o simulaciones de movimientos. Con el fin de comprobar si el robot está en capacidad de realizar el cambio de superficie de horizontal a vertical, se dividió en etapas los movimientos que el robot necesita para ejecutar esta tarea, etapa 1, ubicación frente a la superficie vertical, ver figura 6.9:

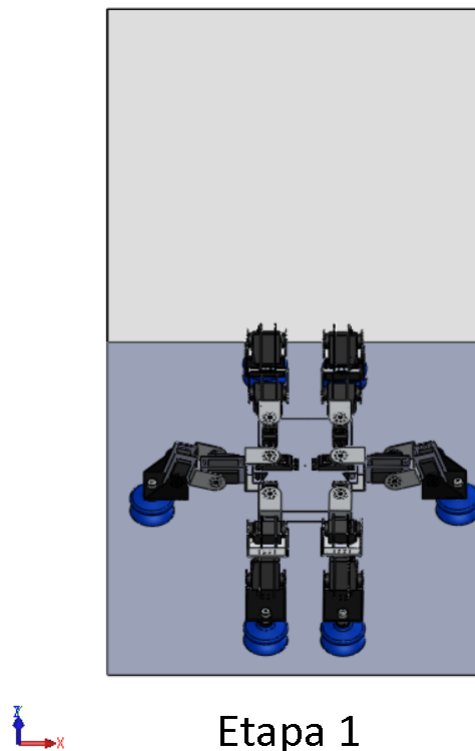


Figura 6.9: Etapa 1, «ubicación frente a la superficie vertical».

Una vez encontrada una superficie vertical y ubicado frente a ella, se eleva el tórax hasta su altura máxima.

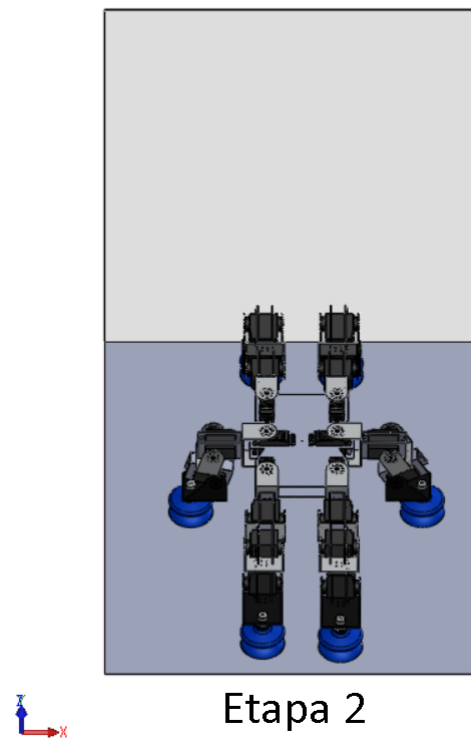


Figura 6.10: Etapa 2, «elevación tórax».

Posteriormente se ubican las ventosas en la superficie vertical.

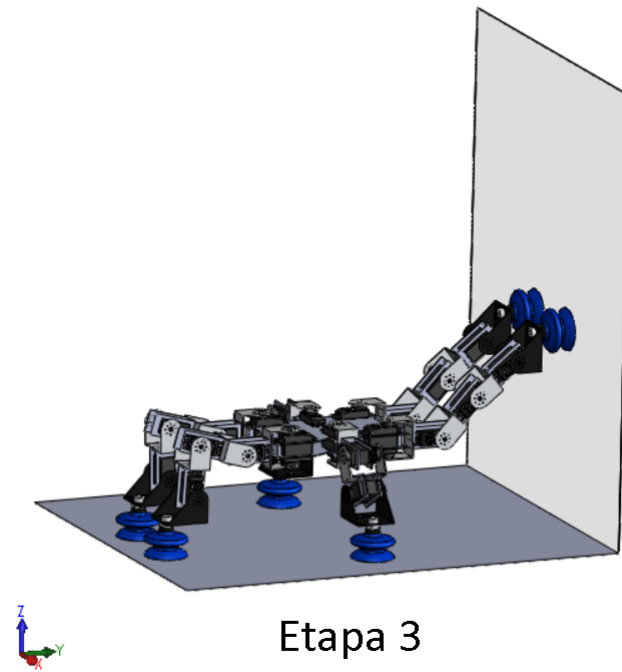


Figura 6.11: Etapa 3 «ubicación de las ventosas en la pared».

Ya con cuatro ventosas fijas «las delanteras y traseras», se reorienta el tórax.

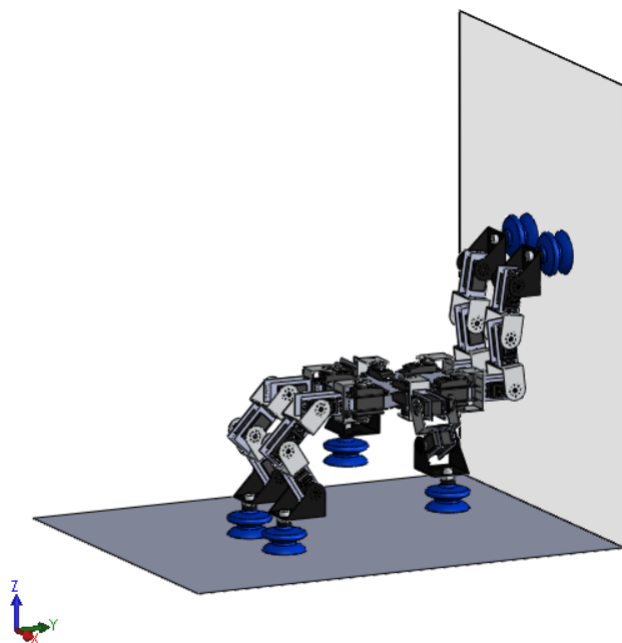


Figura 6.12: Etapa 4, «re-ubicación de las ventosas».

El siguiente paso es orientar el tórax paralelo con la superficie vertical.

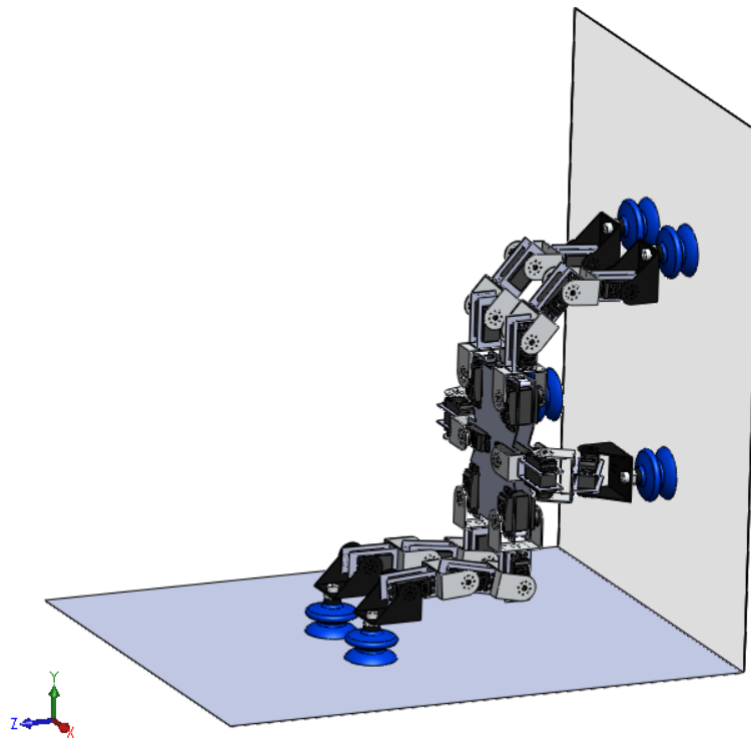


Figura 6.13: Etapa 5, «orientación tórax».

Se acomodan nuevamente las ventosas y se acerca el tórax a la pared hasta que las patas traseras toquen la superficie.

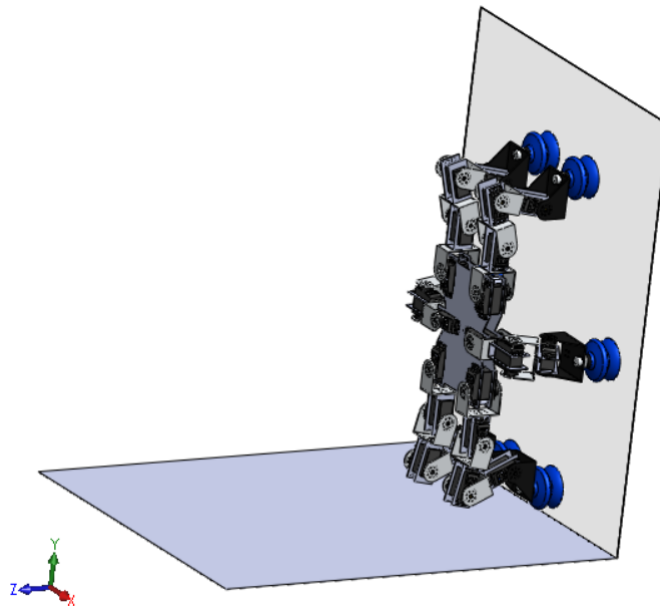


Figura 6.14: Etapa 6, «ubicación de ventosas».

6.3. Pruebas en ADAMS

ADAMS es un programa de diseño asistido por computador que sirve para realizar análisis de fuerzas, par y movilidad entre otros. Este programa se utilizó para conocer el par requerido en cada una de las articulaciones del robot en función del tiempo. El primer análisis consiste en verificar, por medio de una simulación la capacidad que tiene el robot de trasladar el tórax en sus tres ejes, por lo tanto, se realizó una simulación de traslación del tórax para cada eje «Z, X, Y», dejando tres patas fijas y sujetas a la superficie. En esta simulación se puede apreciar el par en cada motor cuando se realiza el movimiento. Estas traslaciones se describen a continuación:

- Traslación en el eje Z: Teniendo como referencia el sistema de coordenadas usado en SolidWorks, se realiza la primera simulación en ADAMS, haciendo una traslación en el eje Z, lo cual provoca una separación o un acercamiento a la superficie de apoyo. En la figura 6.15 observan los dos puntos máximos que puede alcanzar el robot en el eje Z.

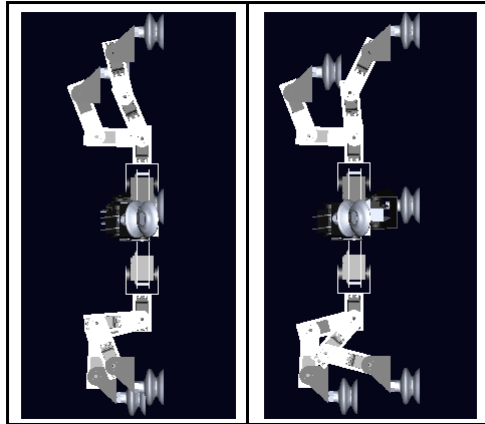


Figura 6.15: Traslación en el eje Z.

Además de comprobar la capacidad que tiene el robot de desplazar el tórax en el eje Z, se calcula el par en cada uno de los motores, en la figura 6.16 se muestra la gráfica del par requerido para realizar este movimiento.

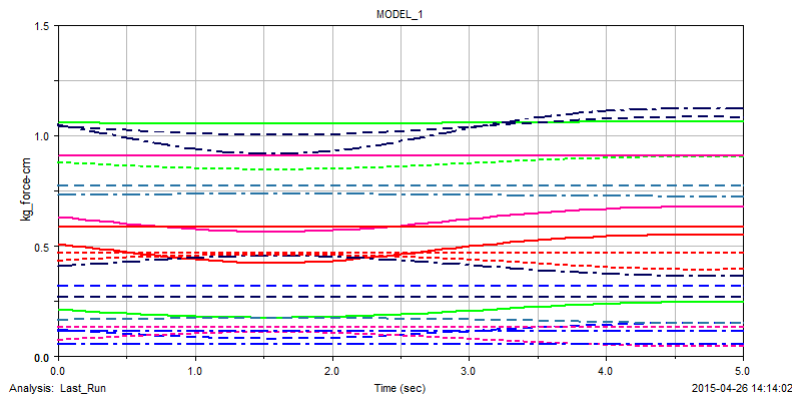


Figura 6.16: Gráfica de par en traslación del eje Z.

La traslación en el eje Z consiste en separar o acercar el tórax de la superficie de apoyo, permitiendo encontrar el punto de menor par para cada motor. Con el fin de facilitar la visualización del par de los motores se seleccionó la pata que tiene los valores de par más elevados. En la figura 6.17 se observa la gráfica de los valores calculados por ADAMS.

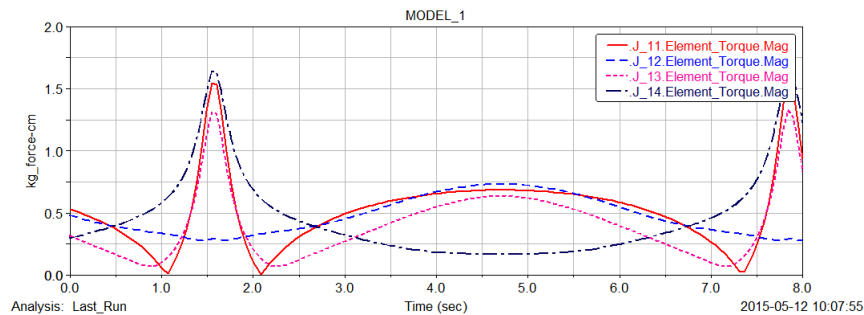


Figura 6.17: Gráfica de par «Pata 1» en traslación en eje Z.

De igual manera se comprobó el par en las siguientes patas de apoyo, que corresponde a la pata 3 y pata 5. En la figura 6.18 y figura 6.19 se puede observar el comportamiento del par en dichas patas al momento de realizar la traslación descrita anteriormente.

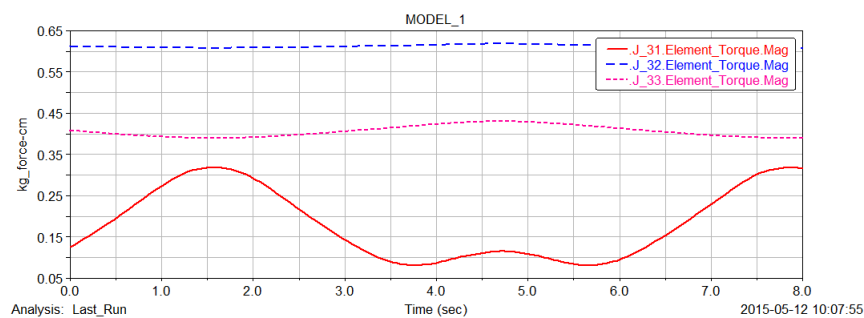


Figura 6.18: Gráfica de par «Pata 3» en traslación en eje Z.

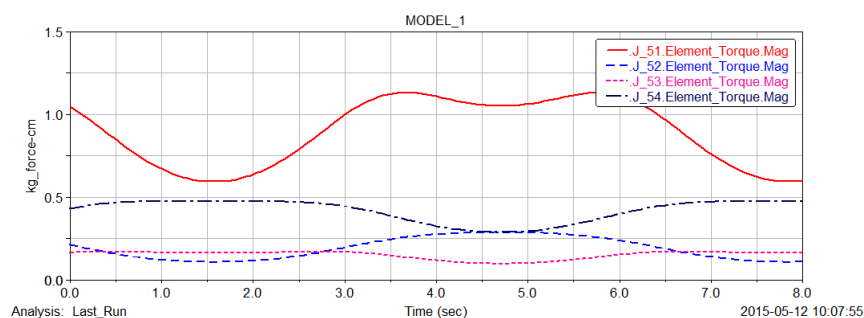


Figura 6.19: Gráfica de par «Pata 5» en traslación en eje Z.

Durante la etapa de diseño es importante considerar el par máximo en cada uno de los motores a la hora de desplazarse o incluso realizar alguna maniobra. Habiendo analizado las gráficas de par donde se realizó un movimiento oscilatorio de traslación a lo largo del eje Z, se observó un aumento en el par entre el segundo 1 y 2 para la pata

1, 3 y 5, pero también se puede observar un punto en el que el par es relativamente pequeño para todos los motores en el segundo 0.5. Con esta simulación se demuestra que el robot está tiene la capacidad de modificar su postura con el el objetivo de encontrar el punto en el que los motores tengan un par mínimo.

- Traslación en el eje X: En la segunda simulación se traslada el tórax del robot en el eje X, como se observa en la figura 6.20.

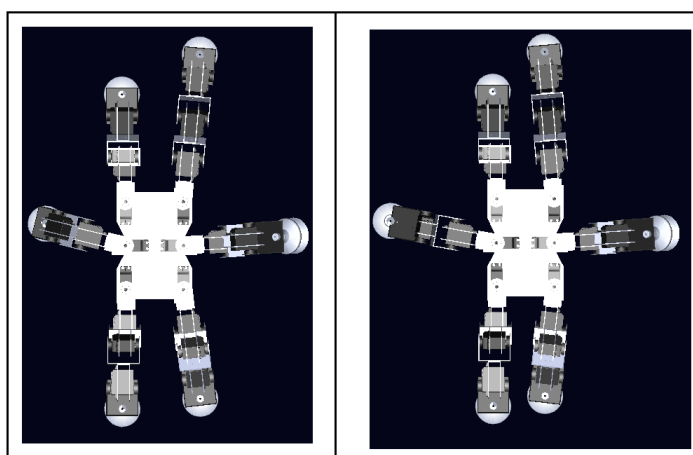


Figura 6.20: Traslación en el eje X.

Para realizar este movimiento se se calcula el par de cada motor dando como resultado la gráfica que se muestra en la figura 6.16:

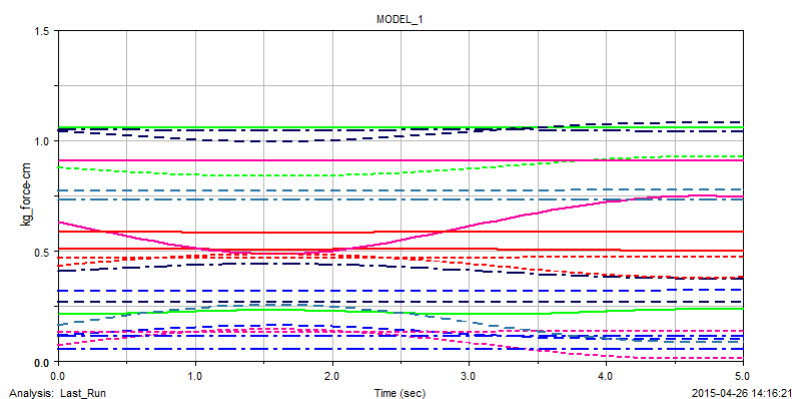


Figura 6.21: Gráfica de par en traslación del eje X.

De la misma manera se realizó el análisis para la traslación en el eje Y, este desplazamiento es considerando el más importante, ya que el avance del robot depende de la distancia que se desplace en este eje. En la figura 6.22, se ve la posición inicial y final del robot al desplazarse en el eje Y.

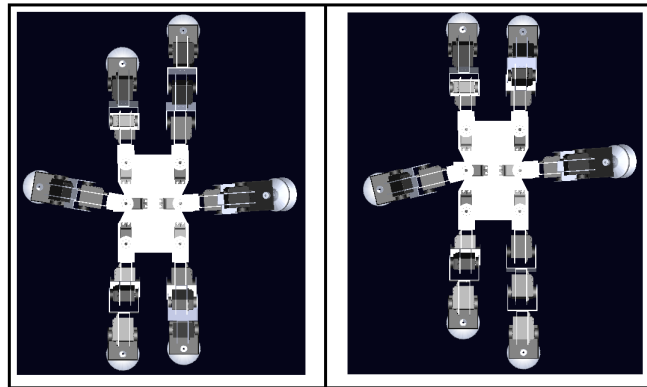


Figura 6.22: Traslación en el eje Y.

De la misma manera se se hace la gráfica del par en cada motor como se observa en la figura 6.23, para comprobar que los motores podrán soportar la carga al realizar este movimiento.

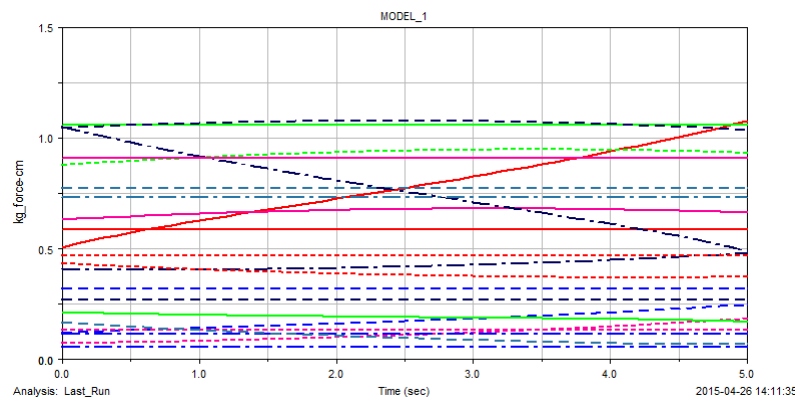


Figura 6.23: Gráfica de par en traslación del eje Y.

Además de las traslaciones en los tres ejes, se realizó una simulación para verificar que el tórax puede rotar en sus tres ejes, calculando el par en cada motor como se muestra en la figura 6.24.

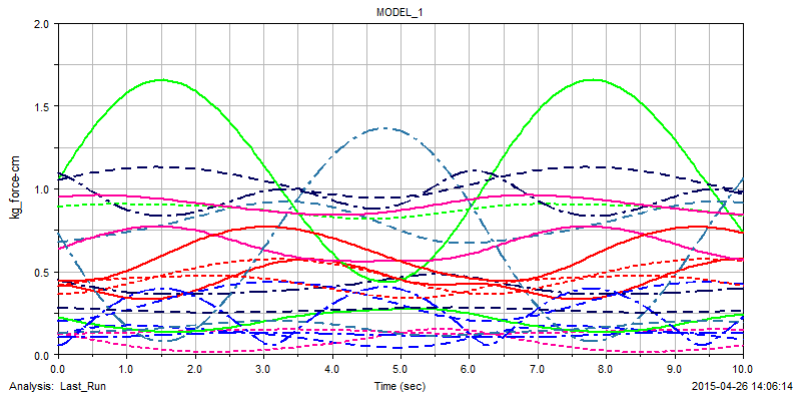


Figura 6.24: Rotación del tórax en tres ejes.

Las pruebas realizadas en ADAMS revelan que el robot es capaz de realizar traslación y rotación en sus tres ejes sin que el par supere el 60 % de la capacidad de los motores escogidos. El máximo valor fue de 1.7 kg*cm y cada motor puede alcanzar un máximo de 12 kg*cm.

Con el propósito de comprobar que el robot tiene la capacidad de desplazarse en superficie horizontal como se muestra en la figura 6.25-a) y vertical como se muestra en la figura 6.25-b), se realizó un análisis de par por medio de ADAMS.

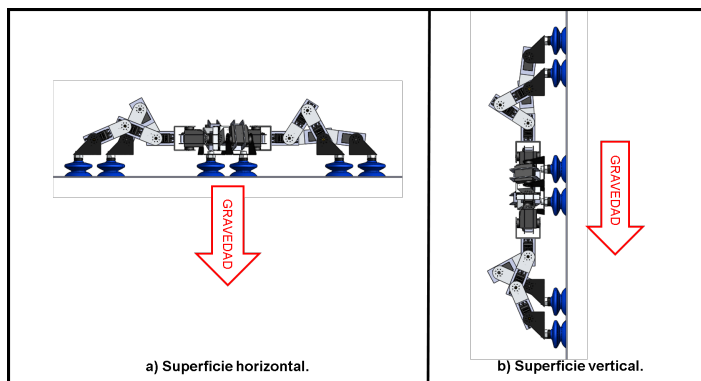


Figura 6.25: Horizontal - Vertical.

En la figura 6.26-a) se muestra el par en los motores para desplazamiento en superficie vertical y en la figura 6.26-b) se muestra el resultado del análisis de par en superficie vertical.

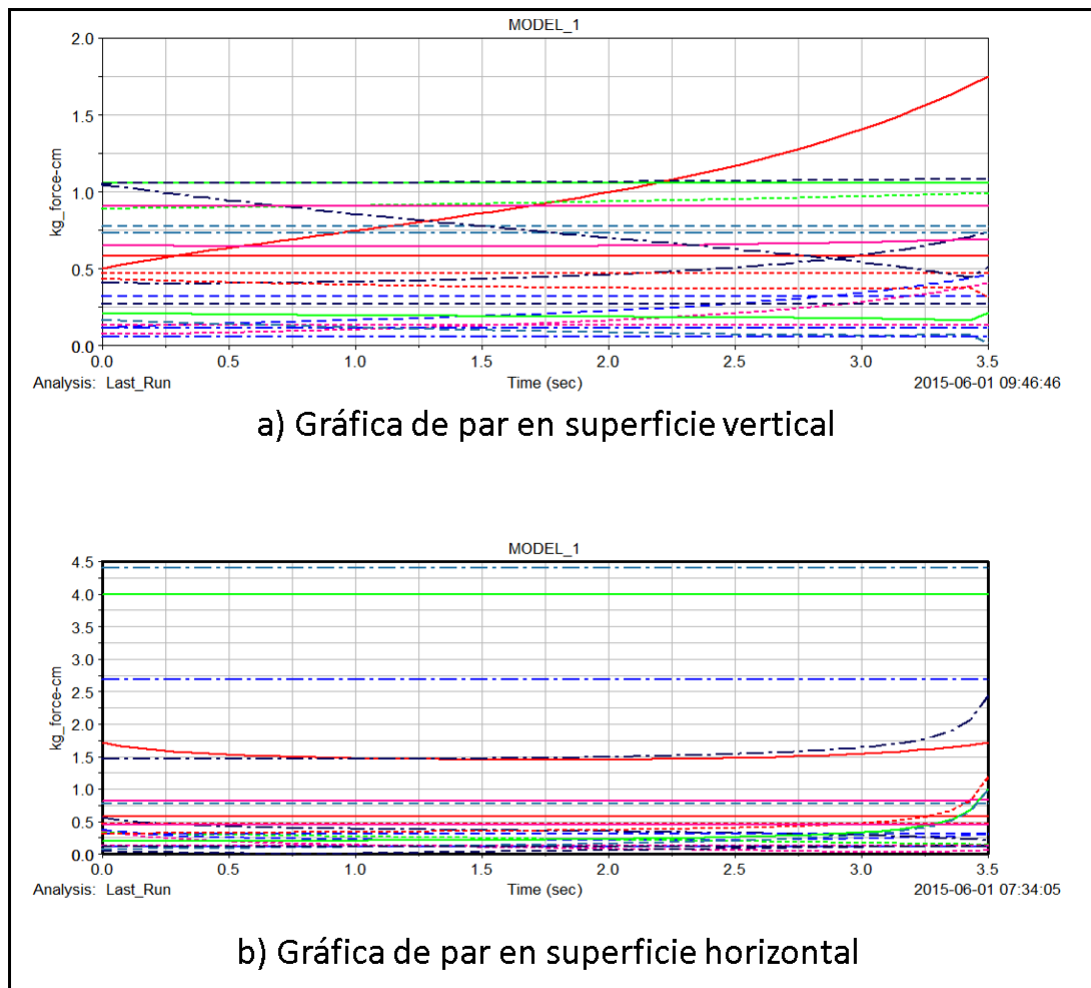


Figura 6.26: Comparación de par Horizontal - Vertical.

Según los resultados obtenidos con el análisis en ADAMS, el par en superficie horizontal es mayor que en superficie vertical, por lo que podemos afirmar que el diseño del robot permite ajustar el centro de masa para obtener un mínimo par en los motores sólo si se encuentra en una superficie vertical.

6.4. Cálculo de distancia por medio de análisis de imágenes

Para calcular la distancia recorrida se utilizó análisis de imágenes, en particular los parámetros intrínsecos de la cámara. Se imprimió un patrón cuadrículado con dimensiones conocidas y se pegó en una superficie plana en el tórax del robot para

medir su desplazamiento. Además se tomaron seis puntos de manera aleatoria para determinar los parámetros intrínsecos de la cámara como se muestra en la figura 6.27.

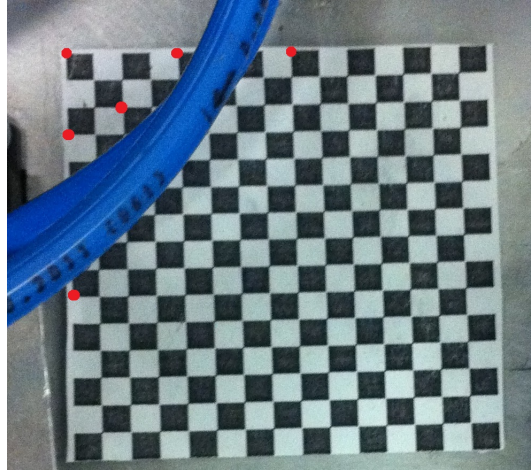


Figura 6.27: Patrón cuadrículado del tórax.

Cada cuadro tiene una medida de 0.4cm y las coordenadas de cada punto son las siguientes: P1=(1, 1), P2=(1, 5), P3=(1, 9), P4=(4, 1), P5=(10, 1), P6=(3, 3).

Por medio del comando «ginput» de Matlab, se obtuvo las coordenadas en píxeles de la imagen, para completar la matriz «A» mencionada en el marco teórico y poder extraer el eigenvector. Una vez calculados los eigenvectores y eigenvalores, se obtiene el eigenvector más pequeño para formar la matriz «m» y con esta información se procede a resolver las ecuaciones de manera simbólica como se muestra a continuación utilizando las ecuaciones 2.15 y 2.16 mencionadas anteriormente y considerando que $Z=0$ porque el desplazamiento del robot es el plano que describen los parámetros extrínsecos.

$$Xs = solve('U = ((m11 * x) + (m12 * y) + m14)((m31 * x) + (m32 * y) + m34)', 'x');$$

$$Ys = solve('V = ((m21 * (-(m14 + m12 * y - U * (m34 + m32 * y)))/(m11 - U * m31))) + (m22 * y) + m24)/((m31 * (-(m14 + m12 * y - U * (m34 + m32 * y)))/(m11 - U * m31))) + (m32 * y) + m34)', 'y');$$

Considerando que los parámetros de la caracterización de la cámara se mantienen constantes, se tomaron los valores en píxeles de un punto de referencia en el patrón que se encuentra en el tórax del robot, en una posición inicial y una posición final como se muestra en la figura 6.28.

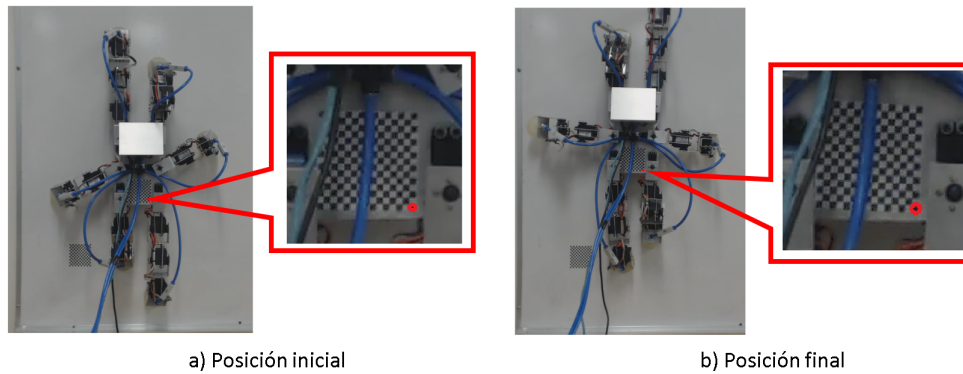


Figura 6.28: Distancia recorrida por el tórax.

Una vez teniendo el valor en pixeles de la distancia recorrida, se hace la conversión de pixeles a milímetros y se calcula la distancia recorrida por medio del teorema de pitágoras que se muestra en la ecuación 6.1.

$$Dist = \sqrt{(X_1 - X_2)^2 - (Y_1 - Y_2)^2} \quad (6.1)$$

La distancia deseada por cada paso es 9cm, y por medio de análisis de imágenes se calculó una distancia recorrida de 8.73cm. Hay que tener en cuenta que la distancia recorrida no es exactamente los 9 cm, ya que hay varios factores que ocasionan diferencia entre los valores deseados y los obtenidos. En el anexo D, se muestra detalladamente el código en Matlab utilizado para encontrar los valores.

Capítulo 7

Conclusiones

- Como resultado del diseño se obtuvo un prototipo capaz de trasladar y rotar el tórax en sus tres ejes, por lo tanto, el robot tiene seis grados de libertad y puede realizar movimientos que le permiten hacer el cambio de superficie horizontal a superficie vertical.
- Se obtuvo un algoritmo de fácil uso y comprensión para la locomoción de robots caminantes, el cual puede adaptarse fácilmente a cualquier robot modificando únicamente el modelo geométrico inverso.
- Gracias a la simulación realizada en ADAMS en el capítulo 6.3, la configuración de las patas del robot permite modificar la separación del tórax con la superficie de apoyo con el fin de encontrar el punto de mínimo de par requerido en los motores y así aumentar la capacidad de carga o disminuir el consumo de energía.
- Gracias al análisis de par en superficies horizontales realizada en la sección 6.3, se puede afirmar que, debido a la configuración del prototipo, el par necesario para realizar la secuencia de locomoción en superficie horizontal es mayor en comparación con el par requerido para la locomoción en superficie vertical.
- Las simulaciones realizadas en ADAMS en el capítulo 6.3 permiten comprobar la capacidad de movimiento del robot, concluyendo que el robot puede desplazarse en cualquier dirección y seguir cualquier tipo de trayectoria de acuerdo al perfil de posición.
- Una de las ventajas del algoritmo diseñado es que permite realizar movimientos controlados de acuerdo a las especificaciones requeridas por la persona que lo esté manipulando, ya sea, variar la distancia de desplazamiento o el tiempo requerido para que se desplace cierta distancia.

Trabajo futuro

Como continuación de este proyecto se propone realizar los siguientes trabajos: el análisis dinámico y pruebas con diferentes tipos de cargas con el fin de encontrar el peso máximo que puede cargar el robot en superficie vertical; incluir en los cálculos el tiempo que tardan las ventosas en liberar el vacío con el fin de acercarse más al comportamiento deseado; incluir una caracterización de los motores para implementar en el algoritmo el uso de los perfiles de velocidad y aceleración; finalmente, la exploración de diferentes secuencias de locomoción para comparar cual puede seguir un perfil de posición con mayor precisión.

Referencias

- J Arrillaga, Jesús Arrillaga Garmendia, and Luis Ignacio Eguíluz Morán. *Armonicos en sistemas de potencia*. Ed. Universidad de Cantabria, 1994.
- Alan T Asbeck and Kim. Climbing walls with microspines. In *IEEE ICRA*, 2006.
- Anibal Ollero Baturone. *Robótica: manipuladores y robots móviles*. Marcombo, 2001.
- George Beekman. *Introducción a la computación*. Pearson Educación, 1999.
- Paolo Boscariol, Michael a. Henrey, Yasong Li, and Carlo Menon. Optimal Gait for Bioinspired Climbing Robots Using Dry Adhesion: A Quasi-Static Investigation. *Journal of Bionic Engineering*, 10(1):1–11, January 2013. ISSN 16726529. doi: 10.1016/S1672-6529(13)60193-6. URL <http://linkinghub.elsevier.com/retrieve/pii/S1672652913601936>.
- Alexander Cerón Correa. *Sistemas roboticos teleoperados*. Universidad Militar Nueva Granada, 2005.
- Qingyong Ding, Lining Sun, Junhong Ji, and Leming Zhang. Calibration of a 2-DOF planar parallel robot: home position identification and experimental verification. In *Mechatronics and Automation, 2005 IEEE International Conference*, volume 1, pages 510–515. IEEE, 2005.
- J. Estremera, J.a. Cobano, and P. Gonzalez 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*, 58(5):700–711, May 2010. ISSN 09218890. doi: 10.1016/j.robot.2009.11.004. URL <http://linkinghub.elsevier.com/retrieve/pii/S0921889009002012>.
- Juan Carlos Grieco, Manuel Prieto, Manuel Armada, and P Gonzalez de Santos. A six-legged climbing robot for high payloads. In *Control Applications, 1998. Proceedings of the 1998 IEEE International Conference on*, volume 1, pages 446–450. IEEE, 1998.

Shigeo Hirose and Mikio Sato. Coupled drive of the multi-DOF robot. In *Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on*, pages 1610–1616. IEEE, 1989.

Bing L. Luk, David S. Cooke, Stuart Galt, Arthur a. Collie, and Sheng Chen. Intelligent legged climbing service robot for remote maintenance applications in hazardous environments. *Robotics and Autonomous Systems*, 53(2):142–152, November 2005. ISSN 09218890. doi: 10.1016/j.robot.2005.06.004. URL <http://linkinghub.elsevier.com/retrieve/pii/S0921889005001016>.

Luther R Palmer, Eric D Diller, and Roger D Quinn. Design of a wall-climbing hexapod for advanced maneuvers. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 625–630. IEEE, 2009.

Luther R Palmer, Eric D Diller, and Roger D Quinn. Toward a rapid and robust attachment strategy for vertical climbing. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 2810–2815. IEEE, 2010.

Robotics & Electronics Pololu. Mini Maestro 24-Channel USB Servo Controller (Assembled), 2014. URL <http://www.pololu.com/product/1356>.

J.M. Porta and E. Celaya. Reactive free-gait generation to follow arbitrary trajectories with a hexapod robot. *Robotics and Autonomous Systems*, 47(4):187–201, July 2004. ISSN 09218890. doi: 10.1016/j.robot.2004.04.001. URL <http://linkinghub.elsevier.com/retrieve/pii/S0921889004000600>.

I. Pretto, S. Ruffieux, C. Menon, a.J. Ijspeert, and S. Cocuzza. A Point-Wise Model of Adhesion Suitable for Real-Time Applications of Bio-Inspired Climbing Robots. *Journal of Bionic Engineering*, 5:98–105, September 2008. ISSN 16726529. doi: 10.1016/S1672-6529(08)60079-7. URL <http://linkinghub.elsevier.com/retrieve/pii/S1672652908600797>.

Fernando Martínez Ramírez. Control embebido de un vehículo guiado automáticamente mediante redes neuronales artificiales. Master’s thesis, Centro Nacional de Investigación y Desarrollo Tecnológico, 2004.

Nacho Herrero Reder. Robot Hexapodo basado en 68HC11, 2012. URL http://webpersonal.uma.es/~iherrero/index.php?option=com_content&view=article&id=51:hexapodo&catid=40:oldpfc&Itemid=66.

- Isidro Ramos Salavert and Miguel A Fernández Graciani. *Vida artificial*, volume 10. Univ de Castilla La Mancha, 1995.
- XOCHITL YAMILE Sandoval-Castro, MARIO A Gracia-Murillo, JONNY PAUL Zavala-De Paz, and EDUARDO Castillo-Castaneda. Hex-piderix: A six-legged walking climbing robot to perform inspection tasks on vertical surfaces. In *Proc. of the 16th Int. Conf. CLAWAR-2013, Sydney*, pages 399–407. World Scientific, 2013.
- Josef Schmitz and Holk Cruse. Behaviour-based modelling of hexapod locomotion : linking biology and technical application q. 33:237–250, 2004. doi: 10.16/j.asd.2004.05.004.
- Manuel F. Silva, J.a. Tenreiro Machado, and Ramiro S. Barbosa. Complex-order dynamics in hexapod locomotion. *Signal Processing*, 86(10):2785–2793, October 2006. ISSN 01651684. doi: 10.1016/j.sigpro.2006.02.024. URL <http://linkinghub.elsevier.com/retrieve/pii/S0165168406000636>.
- Leonardo Enrique Solaque Guzman et al. Flotilla de robots para trabajos en robótica cooperativa. 2014. Robot tipo oruga.
- Mark W Spong, Seth Hutchinson, and Mathukumalli Vidyasagar. *Robot modeling and control*, volume 3. Wiley New York, 2006.
- SuperRobotica. Robot todo terreno robot terrestre 4x4", 10 2012. URL <http://www.superrobotica.com/Images/S300250big.JPG>. <http://www.superrobotica.com/Images/S300250big.JPG>.
- George Brinton Thomas, Julio Porcel Moleón, and Luis Bravo Aguilar. *Desi*. Aguilar, 1959.
- Emanuele Trucco and Alessandro Verri. *Introductory techniques for 3-D computer vision*, volume 201. Prentice Hall Englewood Cliffs, 1998.
- Zhiying Wang, Xilun Ding, Alberto Rovetta, and Alessandro Giusti. Mobility analysis of the typical gait of a radial symmetrical six-legged robot. *Mechatronics*, 21(7): 1133–1146, October 2011. ISSN 09574158. doi: 10.1016/j.mechatronics.2011.05.009. URL <http://linkinghub.elsevier.com/retrieve/pii/S0957415811001115>.
- David Wettergreen and Chuck Thorpe. Developing planning and reactive control for a hexapod robot. In *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, volume 3, pages 2718–2723. IEEE, 1996.

Tomoaki Yano, Shinji Numao, and Yukio Kitamura. Development of a self-contained wall climbing robot with scanning type suction cups. In *Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on*, volume 1, pages 249–254. IEEE, 1998.

Apéndice A

Anexos

Anexo A. Programa principal

```
% M-file to compute a quintic polynomial reference trajectory
% q0 = initial position  v0 = initial velocity  ac0 = initial acceleration
q1 = final position
% v1 = final velocity  ac1 = final acceleration  t0 = initial time  tf =
final time
% fwrite(s2,[170, 12, 31, 1, 22, 0, 50],'uint8'); % ENCENDER VENTOSA 1
% fwrite(s2,[170, 12, 31, 1, 22, 0, 1],'uint8'); % APAGAR VENTOSA 1
% fwrite(s2,[170, 12, 31, 1, 23, 0, 50],'uint8'); % ENCENDER VENTOSA 2
% fwrite(s2,[170, 12, 31, 1, 23, 0, 1],'uint8'); % APAGAR VENTOSA 2
% fclose(instrfind)
clear all;close all;clc

cont=0;

% _____ INICIO DE GENERACIÓN DE
TRAYECTORIA _____
d = input('initial data = [Distancia , Tiempo] =');
q0 = 0; v0= 0; ac0 = 0; q1 = d(1); v1 = 0; ac1=0; to=0 ; tf=d(2);
t = linspace(to,tf,100*(tf-to));
c = ones(size(t));
%*****%
M = [ 1 to to^2 to^3 to^4 to^5;          %
      0 1 2*to 3*to^2 4*to^3 5*to^4;      %
      0 0 2 6*to 12*to^2 20*to^3;         %
      1 tf tf^2 tf^3 tf^4 tf^5;          %
      0 1 2*tf 3*tf^2 4*tf^3 5*tf^4;      %
      0 0 2 6*tf 12*tf^2 20*tf^3];        %
%*****%
b=[q0; v0; ac0; q1; v1; ac1];
a = inv(M)*b; % qd = position trajectory % vd = velocity trajectory % ad =
acceleration trajectory
qd = a(1).*c + a(2).*t +a(3).*t.^2 + a(4).*t.^3 +a(5).*t.^4 + a(6).*t.^5;
vd = a(2).*c +2*a(3).*t +3*a(4).*t.^2 +4*a(5).*t.^3 +5*a(6).*t.^4;
ad = 2*a(3).*c + 6*a(4).*t +12*a(5).*t.^2 +20*a(6).*t.^3;

%~~~~~ LLAMAR FUNCIÓN DE CINEMÁTICA INVERSA ~~~~~
pasos=ceil(d(1)/18);
posi_1=cinem_inv(pasos);
%~~~~~

% _____ FIN DE GENERACIÓN DE TRAYECTORIA _____

%***** SECUENCIA DE INICIALIZACIÓN *****%
ini_robot(1) %
%*****%

x=t;          % N+1=11 points
y=qd;        % Runge function
xspline=(to:0.0001:tf); % Finer spacing in x
yspline = spline(x,y,xspline);

s2=serial('COM3');
fopen(s2);
```

```

fwrite(s2,[170, 12, 31, 1, 22, 0, 50],'uint8'); % ENCENDER VENTOSA 1
fwrite(s2,[170, 12, 31, 1, 23, 0, 1],'uint8'); % APAGAR VENTOSA 2
%*****
Dist=1:ql;
m = Dist(mod(Dist, 9) == 0);
a= length (m);
n=1;
xv=0;
tic
%*****
while (1)

    while toc > to && toc < tf
        t_1=toc;
        t1=(round(t_1*10000))/10000;
        tr=find(xspline == t1);% tr es la posición del vector donde
se encuentra el valor de Y en el tiempo real
        Dr=yspline(tr); % Dr es la distancia donde debe estar el
centro del torax en el tiempo real
        t_r=(tr/10000)+to;
%_____capturar angulos de la matriz de Cinemática Inversa_____
        if Dr>=0.1
            D_r =round(Dr*10)/10;
            if D_r>0
                posi=posi_1(D_r*10, :);%tomo el vector de ángulos
"posición D_r en el tiempo t_r"
            else
                posi=posi_1(1, :);
            end
        end

%_____

        if m==0
            m=1;
        end

b=m(n);% igualo b a la posición en m
% encender ventosas*****
if D_r==(b-0.2)
    if xv==0;
        fwrite(s2,[170, 12, 31, 1, 23, 0, 50],'uint8'); % ENCENDER VENTOSA 2
        xv=1;
    end
    if xv==2
        fwrite(s2,[170, 12, 31, 1, 22, 0, 50],'uint8'); % ENCENDER VENTOSA 1
        xv=3;
    end
end
if D_r==(b+0.1)
    if n< length(m+1) %pregunta si n es de la dimensión de m para no sumar más
posiciones
        n=n+1; end
    if xv==1;
        fwrite(s2,[170, 12, 31, 1, 22, 0, 1],'uint8'); % APAGAR VENTOSA 1
        xv=2;
    end
end

```

```

    if xv==3;
    fwrite(s2,[170, 12, 31, 1, 23, 0, 1], 'uint8'); % APAGAR VENTOSA 2
    xv=0;
    end
end
%***** fin de activación de electroválvulas *****

%***** ENVIAR POR SERIAL A LA TARJETA POLOLU
*****
    fwrite(s2,[170, 12, 31, 22, 0, ...
    posi(27), posi(28), posi(29), posi(30), posi(25), posi(26), posi(5) ,
posi(6), ...% 0-3
    posi(41), posi(42), posi(23), posi(24), posi(37), posi(38), posi(39),
posi(40), ...% 4-7
    posi(43), posi(44), posi(31), posi(32), posi(33), posi(34), posi(1) ,
posi(2), ...% 8-11
    posi(35), posi(36), posi(21), posi(22), posi(17), posi(18), posi(19),
posi(20), ...% 12-15
    posi(3) , posi(4) , posi(9) , posi(10), posi(11), posi(12), posi(13),
posi(14), ...% 16-19
    posi(15), posi(16), posi(7) , posi(8)], 'uint8') % 20-21

%*****
*****
                                end
%*****

                                end
    if toc >= tf
        break
    end
    end

fclose(s2);

disp('Fin de la trayectoria')

```


Anexo B. Subrutina de inicialización Matlab

```
function ini_robot(sec)

%0xAA,numdispos,0x1F,numdestinos,1numcanal,1destLB,1destHB,2destLB,2destHB***
** posición
% 0xAA,numdispos,0x07,numcanal,velocidadLB,velocidadHB***** velocidad

% clear all; close all; clc;
% fclose(instrfind)
s2=serial('COM3');
fopen(s2);

fwrite(s2,[170, 12, 31, 24, 0, ...
          0, 0, 0, 0, 0, 0, 0, 0, 0,...% 1 - 4
          0, 0, 0, 0, 0, 0, 0, 0, 0,...% 5 - 8
          0, 0, 0, 0, 0, 0, 0, 0, 0,...% 9 - 12
          0, 0, 0, 0, 0, 0, 0, 0, 0,...% 13 - 16
          0, 0, 0, 0, 0, 0, 0, 0, 0,...% 17 - 20
          0, 0, 0, 0, 0, 0, 0, 0], 'uint8')      % 21 - 24

pause()

frame_1=[];frame_2=[];frame_3=[];frame_4=[];frame_5=[];frame_6=[];frame_7=[];
%_____

frame_1=[3297 5246 8443 3063 8911 4736 7616 2985 7040 7312 6650 7552 ...
        8521 4544 5402 8911 6806 4516 7274 7040 5504 4778];

frame_2=[8576 7168 4096 9088 2816 4736 7616 7424 4736 7296 9457 7552 ...
        2560 5632 9856 2816 2560 4480 9223 1792 5504 6016];

frame_3=[9379 5168 5402 9691 2049 4736 7616 6572 6494 7296 9344 7552 ...
        2560 4155 8521 2595 3609 4480 9691 1737 5504 4778];

frame_4=[3840 7936 6416 3712 2048 4736 7616 6784 6528 8320 5504 7552 ...
        5632 3584 7936 2432 5090 6016 8064 2688 5504 7296];

frame_5=[3843 7976 8755 3730 2127 4736 7616 6867 6646 8408 5558 7552 ...
        5714 3609 7945 2523 6572 6117 8134 2751 5504 7412];
%%%%%%%%%%_____

%_____
%multiplicación de cada frame *8
```

```

% frame_1=frame1.*4;frame_2=frame2.*4;frame_3=frame3.*4;frame_4=frame4.*4;
% frame_5=frame5.*4;frame_6=frame6.*4;frame_7=frame7.*4;
%
%-----
%conversión decimal - binario
frame1=dec2bin(frame_1);frame2=dec2bin(frame_2);frame3=dec2bin(frame_3);
frame4=dec2bin(frame_4);frame5=dec2bin(frame_5);
%
%-----

frame=[frame1; frame2; frame3; frame4; frame5];
num=length(frame)/22;
var=0;
cont=0;
pos=zeros(44,1);
%*****

%*****
for j=1:num
    var=var+22;
    l=0;
    cont=cont+1;
    for i=(var-21):var

        l=l+1;ll=l*2;

pos(ll)=bin2dec(frame(i,(1:7)));
pos(ll-1)=bin2dec(frame(i,(8:14)));
end
posi=pos';

if cont == 1
    for ki=1:22
fwrite(s2,[170, 12, 31, ki, 0, 0, posi(2), 0, posi(4)...
    , 0, posi(6), 0, posi(8), 0, posi(10), 0, posi(12)...
    , 0, posi(14), 0, posi(16), 0, posi(18), 0, posi(20)...
    , 0, posi(22), 0, posi(24), 0, posi(26), 0, posi(28)...
    , 0, posi(30), 0, posi(32), 0, posi(34), 0, posi(36)...
    , 0, posi(38), 0, posi(40), 0, 0, 0, posi(44)],'uint8')
pause(0.1)
    end
end

if cont ==2
    for i=1:22
        v=10;
        if i== 7 || 14 || 18 ||10
            v=50;
        end
        fwrite(s2,[170, 12, 7, i, v, 0],'uint8')
        pause(0.01)
    end
end

fwrite(s2,[170, 12, 31, 22, 0, 0, posi(2), 0, posi(4)...

```

```

, 0, posi(6), 0, posi(8), 0, posi(10), 0, posi(12)...
, 0, posi(14), 0, posi(16), 0, posi(18), 0, posi(20)...
, 0, posi(22), 0, posi(24), 0, posi(26), 0, posi(28)...
, 0, posi(30), 0, posi(32), 0, posi(34), 0, posi(36)...
, 0, posi(38), 0, posi(40), 0, posi(42), 0, posi(44)], 'uint8')

pause(2)

end

if cont ==2
    for i=1:22
        v=5;
        if i== 19 || 12 || 0 || 7 || 14 || 16 || 2 || 3
            v=50;
        end
        fwrite(s2,[170, 12, 7, i, v, 0], 'uint8')
        pause(0.01)
    end
end
% fwrite(s2,[170, 12, 4, 16, 0, 20], 'uint8')
% fwrite(s2,[170, 12, 4, 2, 0, 40], 'uint8')
% pause(0.5)
posi=[ 48 59 100 48 18 29 116 57 101 47 8 62 10 26 60 43 ...
      9 62 91 19 48 26 80 37 123 64 52 29 112 59 88 65 ...
      0 57 0 37 64 59 83 53 79 16 118 51];

for i = 1:22
    fwrite(s2,[170, 12, 31, i, 0, ...
              posi(27), posi(28), posi(29), posi(30), posi(25), posi(26),
posi(5) , posi(6),...% 0-3
              posi(41), posi(42), posi(23), posi(24), posi(37), posi(38),
posi(39), posi(40),...%
                  4-7
              posi(43), posi(44), posi(31), posi(32), posi(33), posi(34),
posi(1) , posi(2),...% 8-11
              posi(35), posi(36), posi(21), posi(22), posi(17), posi(18),
posi(19), posi(20),...%
                  12-15
              posi(3) , posi(4) , posi(9) , posi(10), posi(11), posi(12),
posi(13), posi(14),...% 16-19
              posi(15), posi(16), posi(7) , posi(8)], 'uint8') % 20-21
    end
    pause(5)
    for i=1:24
        fwrite(s2,[170, 12, 7, i, 0, 0], 'uint8')
        pause(0.01)
    end

%%% POSICIÓN DE MARCHA %%%%%%%%%%%

fclose(s2);

end

```

Anexo C. Subrutina de Cinemática inversa de posición

```
function [posi_1]=cinem_inv(pasos)

% pasos=1;

grad=[];
mat=[];
dist=0;dist_p=0;
for j=1:pasos
L1=7.8;L2=7.8;L3=7.8;L5=5;
L_1=3.6;L_2=3;L_3=3;
L4=sqrt(L_2^2+(L_1+L_3)^2);
PI_1x=0; PI_1y=4.75; PI_1z=0;%posiciones iniciales
J1x=PI_1x;J1z=PI_1z;J1y=PI_1y;%valores de la junta 1
J2x=L1; J2y=4.75; J2z=0;
a_z= -7; a_y=4.75;% ax=18;
a_z1=a_z+1;
a_z2=a_z+1;
INx=-7.25;
INy=-4.75;
INz=0;
vec=(16.6:0.1:25.5);
vec_2=(25.5:-0.1:16.6);
    dist_3=0;
    dist_2=0;
    cont=0;

% _____secuencia 1_____
for i=1:90

clc;
dist_p=dist_p+1;
dist=dist+0.1; mat(dist_p,1)=dist;

dist_2=dist_2+0.1;

%*****PATA NUMERO 1*****
    ax=vec_2(i); ay=4.75; az=L_1+L_3+a_z;
    R=ax-L_2-L1;
    J4x=R+L1; J4y=4.75; J4z=az;
    R_1=sqrt(R^2+J4z^2);

%    theta_2=acosd((L3^2+L2^2-R_1^2)/(2*L2*L3)); %punto y coma-
    phi_1=asind(J4z/R_1);
    theta_1=acosd((L2^2+R_1^2-L3^2)/(2*L2*R_1));
%    theta_3=theta_1;

    cond=theta_1-phi_1;
    J3z=(sind(cond)*L2)-(J4z);
    J3x=(cosd(cond)*L2)+L1;
    J3y=4.75;
```

```

        theta_2=180-(theta_1*2);
        angle=theta_1+phi_1;
        mat(dist_p,(2:5))=[0 angle theta_2 theta_1 ];

%*****

%*****PATA NUMERO 6*****
ay1=-4.75;O3y=-4.75;O4y=-4.75;O2y=-4.75;O1y=-4.75;
cont=cont+1;
ax=vec(i);
    if cont<45 a_z1=a_z1+0.1;    end
    if cont>45 a_z1=a_z1-0.1;    end
    az1=L_1+L_3+a_z1;
R=ax-L_2-L1;
J4x=R+L1; J4y=4.75; J4z=az1;
R_1=sqrt(R^2+J4z^2);
phi_1=acosd((R^2+R_1^2-J4z^2)/(2*R*R_1));
theta_1=acosd((R_1^2+L2^2-L3^2)/(2*R_1*L2));
J3z=sind(theta_1+phi_1)*L2;
J3x=(cosd(theta_1+phi_1)*L2)+L1;

        theta_2=180-(theta_1*2);
        angle=theta_1+phi_1;
        mat(dist_p,(20:23))=[0 angle theta_2 theta_1 ];

%*****

%_____PATA NUMERO 3_____
P1x=-14.5;P2x=-14.5-L1;
P4x=-R-14.5-L1;
P4z=az;
ax1=-R-14.5-L1-L_2;
ax=vec(i); ay=4.75; az=L_1+L_3+a_z;
    R=ax-L_2-L1;
    J4z=az;
    R_1=sqrt(R^2+J4z^2);
    phi_1=asind(J4z/R_1);
    theta_1=acosd((L2^2+R_1^2-L3^2)/(2*L2*R_1));
    cond=theta_1-phi_1;
    P3x=(-cosd(cond))*L2-14.5-L1;
    theta_2=180-(theta_1*2);
    angle=theta_1+phi_1;
    mat(dist_p,(9:12))=[0 angle theta_2 theta_1 ];

%_____PATA NUMERO 4_____

P1x=-14.5;P2x=-14.5-L1;
P4z=az;
ax1=-R-14.5-L1-L_2;
ax=vec_2(i); ay=4.75; az=L_1+L_3+a_z;
    R=ax-L_2-L1;
    R_1=sqrt(R^2+J4z^2);
    J4z=az1;
    theta_2=acosd((L3^2+L2^2-R_1^2)/(2*L2*L3)); %punto y coma-

```

```

phi_1=asind(J4z/R_1);
theta_1=acosd((L2^2+R_1^2-L3^2)/(2*L2*R_1));
theta_3=theta_1;
cond=theta_1-phi_1;
M4x=-R-14.5-L1;
ax3=M4x-L_2;
P3x=(-cosd(cond))*L2-14.5-L1;
P3z=sind(theta_1+phi_1)*L2;
theta_2=180-(theta_1*2);
angle=theta_1+phi_1;
mat(dist_p,(13:16))=[0 angle theta_2 theta_1 ];
%***** PATA NUMERO 5 *****
INx=-7.25;INy=-4.75; INz=0;
N4z=-7; RT1=(L_1+L_3)+N4z;
RT=sqrt((L5)^2-(RT1^2));
RU=L1+RT+L_2;
theta_4=acosd((RT^2+L5^2-RT1^2)/(2*RT*L5));
theta_5=(180-(90+theta_4))+4.7;

theta_6=atand((vec_2(i)-22.5)/L1);

N2x=(sind(theta_6)*(RU-L_2-RT))+INx;
N2y=-(cosd(theta_6)*(RU-L_2-RT))+INy;
N3x=(sind(theta_6)*(RU-L_2))+INx;
N3y=-(cosd(theta_6)*(RU-L_2))+INy;
N3z=RT1;
N4x=(sind(theta_6)*(RU))+INx;
N4y=-(cosd(theta_6)*(RU))+INy;

if dist_2 < 4.6
    theta_5=theta_5-dist_2;
end
if dist_2==4.5
    theta__5=theta_5;
end
if dist_2 > 4.5
    dist_3=dist_3+0.1;
    theta_5=theta__5+dist_3;
end
mat(dist_p,(17:19))=[ theta_6 theta_4 theta_5 ];

%***** PATA NUMERO 2 *****
INx=-7.25;INy=4.75;INz=0;
RT1=(L_1+L_3)+N4z;
theta_6=atand((vec(i)-22.5)/L1);
N2x=(sind(theta_6)*L1)+INx;
N2y=(cosd(theta_6)*L1)+INy;
N3x=(sind(theta_6)*(L1+RT))+INx;
N3y=(cosd(theta_6)*(L1+RT))+INy;
N3z=RT1+3;
N4z_2=N4z+2;
N4x=(sind(theta_6)*(RU))+INx;
N4y=(cosd(theta_6)*(RU))+INy;

theta_4=acosd((RT^2+L5^2-N3z^2)/(2*RT*L5));

```

```

mat(dist_p,(6:8))=[ theta_6 theta_4 theta_5 ];

%*****
end
%
_____

cont=0;
theta_5=0;
theta__5=0;
dist_2=0; dist_3=0;
%_____secuencia 2_____

for i=1:90
    clc;
    dist_2=dist_2+0.1;
%*****PATA NUMERO 1*****
dist_p=dist_p+1;
dist=dist+0.1; mat(dist_p,1)=dist;
cont=cont+1;
ax=vec(i); ay=4.75;
    if cont<45 a_z1=a_z1+0.1; end
    if cont>45 a_z1=a_z1-0.1; end
    az1=L_1+L_3+a_z1;
R=ax-L_2-L1;
J4x=R+L1; J4y=4.75; J4z=az1;
R_1=sqrt(R^2+J4z^2);
phi_1=acosd((R^2+R_1^2-J4z^2)/(2*R*R_1));
theta_1=acosd((R_1^2+L2^2-L3^2)/(2*R_1*L2));
theta_2=180-(theta_1*2);
J3z=sind(theta_1+phi_1)*L2;
J3x=(cosd(theta_1+phi_1)*L2)+L1;

    angle=theta_1+phi_1;
    mat(dist_p,(2:5))=[0 angle theta_2 theta_1 ];

%*****

%*****PATA NUMERO 6*****
ay1=-4.75;O3y=-4.75;O4y=-4.75;O2y=-4.75;O1y=-4.75;

ax=vec_2(i);
    az=a_z+(L_1+L_3);
R=ax-L_2-L1;
J4x=R+L1; J4y=4.75; J4z=az;
R_1=sqrt(R^2+J4z^2);
phi_1=acosd((R^2+R_1^2-J4z^2)/(2*R*R_1));
theta_1=acosd((R_1^2+L2^2-L3^2)/(2*R_1*L2));
J3z=sind(theta_1+phi_1)*L2;
J3x=(cosd(theta_1+phi_1)*L2)+L1;
    theta_2=180-(theta_1*2);
    angle=theta_1+phi_1;
    mat(dist_p,(20:23))=[0 angle theta_2 theta_1 ];

```

```

%***** PATA NUMERO 3
P1x=-14.5;P2x=-14.5-L1;
P4x=-R-14.5-L1;
ax1=-R-14.5-L1-L_2;
ax=vec_2(i); ay=4.75; az=L_1+L_3+a_z; J4z=az1;
R=ax-L_2-L1;
R_1=sqrt(R^2+J4z^2);
phi_1=asind(J4z/R_1);
theta_1=acosd((L2^2+R_1^2-L3^2)/(2*L2*R_1));
cond=theta_1-phi_1;
P3x=((-cosd(cond))*L2)-14.5-L1;

theta_2=180-(theta_1*2);
angle=theta_1+phi_1;
mat(dist_p,(9:12))=[0 angle theta_2 theta_1 ];
%
NUMERO 4
P1x=-14.5;P2x=-14.5-L1;

ax1=-R-14.5-L1-L_2;
ax=vec(i); ay=4.75; az=L_1+L_3+a_z; J4z=az;
R=ax-L_2-L1;
P4x=-R-14.5-L1;
R_1=sqrt(R^2+J4z^2);
phi_1=asind(J4z/R_1);
theta_1=acosd((L2^2+R_1^2-L3^2)/(2*L2*R_1));
cond=theta_1-phi_1;
P3x=((-cosd(cond))*L2)-14.5-L1;
theta_2=180-(theta_1*2);
angle=theta_1+phi_1;
ax3=P4x-L_2;
mat(dist_p,(13:16))=[0 angle theta_2 theta_1 ];

%***** PATA NUMERO 5 *****
INx=-7.25;INy=-4.75; INz=0;
N4z=-7; RT1=(L_1+L_3)+N4z;
RT=sqrt((-L5)^2-(RT1^2));
RU=L1+RT+L_2;
PR=RT1+3;
theta_4=acosd((RT^2+L5^2-PR^2)/(2*RT*L5));
theta_5=(90-theta_4)+4.7;

theta_6=atand((vec(i)-22.5)/L1);

N2x=(sind(theta_6)*(RU-L_2-RT))+INx;
N2y=-(cosd(theta_6)*(RU-L_2-RT))+INy;
N3x=(sind(theta_6)*(RU-L_2))+INx;
N3y=-(cosd(theta_6)*(RU-L_2))+INy;

N4x=(sind(theta_6)*(RU))+INx;
N4y=-(cosd(theta_6)*(RU))+INy;
% N4_z=N4z+3;

if dist_2 < 4.6
    theta_5=theta_5-dist_2;
end

```



```

if dist_2==4.5
    theta__5=theta_5;
end
if dist_2 > 4.5
    dist_3=dist_3+0.1;
    theta_5=theta__5+dist_3;
end

mat(dist_p,(17:19))=[ theta_6 theta_4 theta_5 ];

%*****
%***** PATA NUMERO 2 *****

INx=-7.5;INy=4.75;INz=0;
RT1=(L_1+L_3)+N4z;
theta_6=atand((vec_2(i)-22.5)/L1);
N2x=(sind(theta_6)*L1)+INx;
N2y=(cosd(theta_6)*L1)+INy;
N3x=(sind(theta_6)*(L1+RT))+INx;
N3y=(cosd(theta_6)*(L1+RT))+INy;
N3z=RT1;
N4z_2=N4z;
N4x=(sind(theta_6)*(RU))+INx;
N4y=(cosd(theta_6)*(RU))+INy;
theta_4=acosd((RT^2+L5^2-RT1^2)/(2*RT*L5));
mat(dist_p,(6:8))=[ theta_6 theta_4 theta_5 ];

end

end

%***** CONVERTIR A MCC *****

%***** CALCULAR PENDIENTE DE CADA MOTOR *****

vec_pend=[];
vec_or=[];
vec_Y=[];
vec_Y_MCC=[];
posi_1=[];

%_____ VALORES DE CADA MOTOR EN MÁXIMOS Y MÍNIMOS EN
MCC_____
vec_pos_1=[1900 1624 935 1652 1024 1660 974 1391 2013 629 843
1204 ...
          2072 941 1916 2102 1684 1350 1904 1700 531 1662 ];

vec_pos_2=[1900 862 2445 1152 1420 1900 810 1391 1409 1995 1584
1204 ...
          1292 2361 1170 1674 2100 1155 1904 824 2072 941];

vec_pos=vec_pos_1-vec_pos_2;

```

```

%
mat_1=mat;
mat_1(:,1)=[];
% VALORES DE CADA MOTOR EN MÁXIMOS Y MÍNIMOS EN
GRADOS

vec_grad_1=mat_1(1, :);
vec_grad_2=mat_1(90, :);
vec_grad_2(6)=4.5886;vec_grad_2(17)=30.1891;
vec_grad_2(18)=60.0109;vec_grad_2(7)=60.0109;
vec_grad=vec_grad_1-vec_grad_2;
% vec_grad()=;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%cambiar los valores en este vector
% para que no de inf

%
];

for i= 1:22
vec_pend(i)=(vec_pos(i)/vec_grad(i));
ind=find(isnan(vec_pend));%encontrar las posiciones del vector
donde hay NaN
vec_pend(ind)=0; % igualar estas posiciones "ind" a cero
vec_or(i)= (vec_pend(i)*vec_grad_1(i)+(vec_pos_1(i)*-1);
%CALCULAR PUNTO DE ORIGEN
ind_=find(isnan(vec_or));
vec_or(ind_)=0;
end

%*****
inv=[1 1 1 1 -1 -1 1 1 1 1 1 1 1 1 1 1 1 1 1 1];
vec_pend=vec_pend.*inv;
%*****
mat_1=mat;
mat_1(:,1)=[];

%***** Pasar de ángulos a MCC*****
lar=length(mat(:, 1));
for i=1:lar
vec_1=mat_1(i, :);
for j=1:22
vec_Y_MCC(i,j)=
vec_pend(j)*vec_1(j)+(vec_or(j)*-1);

end
frame_1=vec_Y_MCC(i, :).*4;

frame_=dec2bin(frame_1);
frame=frame_;
var=length(frame);
l=0;
l_v=length(frame(1, :));
pos=zeros(44,1);
for k=(var-21):var
l=l+1;ll=l*2;
pos(ll)=bin2dec(frame(k, 1:(l_v-7)));

```

```
pos(l1-1) =bin2dec(frame(k, (l_v-6):l_v));  
end  
posi_1(i,:)=pos';
```

```
end
```

```
%*****
```

```
end
```

```

close all; clear all; clc
%***** Lllamar imagen de patrón cuadrulado*****
% Pat=imread('D:\CICATA\documentación tesis de
maestría\Tesis\Tesis\video_robot\1.png');
% image(Pat)
%*****
%***** PUNTOS ALEATORIAMENTE ESCOGIDOS *****
[u1,v1]=ginput(1);
[u2,v2]=ginput(1);
[u3,v3]=ginput(1);
[u4,v4]=ginput(1);
[u5,v5]=ginput(1);
[u6,v6]=ginput(1);
%*****
%*****
% P1=(1, 1), P2=(1, 5), P3=(1, 9), P4=(4, 1), P5=(10, 1), P6=(3, 3)
%*****
%***** VALORES NUMÈRICOS DE LOS PUNTOS ESCOGIDOS*****
x1=0.3;y1=0.3;z=0;
x2=0.3;y2=1.5;
x3=0.3;y3=2.7;
x4=1.2; y4=0.3;
x5=3; y5=0.3;
x6=0.9;y6=0.9;
%*****
%***** COORDENADAS EN LA IMAGEN DE CADA PUNTO *****
u1=684.466; v1=786.4 ;
u2=703.731; v2=787.065;
u3=722.569; v3=787.729;
u4=684.894; v4=801.026;
u5=684.252; v5=830.610;
u6=694.312; v6=797.037;
%*****
%***** MATRIZ "A" *****
A=[x1,y1,0,1,0,0,0,0,-u1*x1,-u1*y1,0,-u1;0,0,0,0,x1,y1,0,1,-v1*x1,-v1*y1,0,-
v1;x2,y2,0,1,0,0,0,0,-u2*x2,-u2*y2,0,-u2;0,0,0,0,x2,y2,0,1,-v2*x2,-v2*y2,0,-
v2;x3,y3,0,1,0,0,0,0,-u3*x3,-u3*y3,0,-u3;0,0,0,0,x3,y3,0,1,-v3*x3,-v3*y3,0,-
v3;x4,y4,0,1,0,0,0,0,-u4*x4,-u4*y4,0,-u4;0,0,0,0,x4,y4,0,1,-v4*x4,-v4*y4,0,-
v4;x5,y5,0,1,0,0,0,0,-u5*x5,-u5*y5,0,-u5;0,0,0,0,x5,y5,0,1,-v5*x5,-v5*y5,0,-
v5;x6,y6,0,1,0,0,0,0,-u6*x6,-u6*y6,0,-u6;0,0,0,0,x6,y6,z,1,-v6*x6,-v6*y6,0,-
v6];
%***** CÁLCULO DE EIGENVALORES Y EIGENVECTORES*****
[u,s,v]=svd(A);
%***** EIGENVECTOR SELECCIONADO *****
m11= -5.170631133432642e-16;
m12= -4.816410415348961e-17;
m13= 1.176692552129060e-06;
m14= 9.476695805328921e-16;
m21= -5.60246855232800e-16;
m22= 3.953480993170910e-18;
m23= 9.99999999993077e-01;
m24= 9.166034313447224e-16;
m31= -7.153090018335236e-19;
m32= -3.117824972592062e-20;
m33= 0;
m34= 1.246642462805041e-18;
%*****

```

```

% ***** LLAMAR IMÁGENES DE POSICIÓN INICIAL Y FINAL*****
Im=imread('D:\CICATA\documentación tesis de
maestría\Tesis\Tesis\video_robot\1.png');
image(Im)
Im2=imread('D:\CICATA\documentación tesis de
maestría\Tesis\Tesis\video_robot\2.png');
image(Im2)
%***** EXTRACCIÓN DE PUNTOS *****
D1U= 952; D1V= 650;
D2U= 927; D2V= 549;

% ***** CAMBIO DE PÍXELES A MILÍMETROS *****
Y1=-(D1V*(m34 - (m31*(m14 - D1U*m34)))/(m11 - D1U*m31)) - m24 + (m21*(m14 -
D1U*m34))/(m11 - D1U*m31)/(D1V*(m32 - (m31*(m12 - D1U*m32)))/(m11 - D1U*m31))
- m22 + (m21*(m12 - D1U*m32))/(m11 - D1U*m31);
X1=-(m14 + m12*Y1 - D1U*(m34 + m32*Y1))/(m11 - D1U*m31);
Y2=-(D2V*(m34 - (m31*(m14 - D2U*m34)))/(m11 - D2U*m31)) - m24 + (m21*(m14 -
D2U*m34))/(m11 - D2U*m31)/(D2V*(m32 - (m31*(m12 - D2U*m32)))/(m11 - D2U*m31))
- m22 + (m21*(m12 - D2U*m32))/(m11 - D2U*m31);
X2=-(m14 + m12*Y2 - D2U*(m34 + m32*Y2))/(m11 - D2U*m31);

%**** CÁLCULO DE LA DISTANCIA POR MEDIO DEL TEOREMA DE PITÁGORAS *****
DIST =sqrt((X1-X2)^2+(Y1-Y2)^2);

```