CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL INSTITUTO POLITÉCNICO NACIONAL

UNIDAD ZACATENCO
DEPARTAMENTO DE CONTROL AUTOMÁTICO

Diseño, modelado, construcción, análisis, control y percepción en robots humanoides

Tesis que presenta

Rafael Stanley Nuñez Cruz

Para Obtener el Grado de

Doctor en Ciencias

En la especialidad de

Control Automático

Director de la Tesis: Dr. Juan Manuel Ibarra Zannatha

Ciudad de México Abril 2017

CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL INSTITUTO POLITÉCNICO NACIONAL

UNIDAD ZACATENCO
DEPARTAMENTO DE CONTROL AUTOMÁTICO

Resumen

Diseño, modelado, construcción, análisis, control y percepción en robots humanoides

por Rafael Stanley Nuñez Cruz

El propósito de esta tesis es mostrar el desarrollo del robot humanoide *Johnny*, el cual esta diseñado con el objetivo de aprovechar la energía de manera eficiente durante el caminado, esta tarea se plantea como un problema de optimización, para resolverla fue necesario identificar los parámetros involucrados en el consumo energético, los cuales dependen del diseño mecánico, las características del caminado, el tipo de control usado, etc.

También fue necesario definir adecuadamente un criterio de optimización y la metodología usada para encontrar una solución, dadas las características particulares del problema se tuvo que hacer uso de técnicas no convencionales de optimización.

Una vez encontrada una solución al problema planteado se continuó a la etapa de fabricación y prueba del prototipo, también se realizaron comparaciones del consumo energético y se implementaron leyes de control para mantener el equilibrio usando diferentes sensores.

Finalmente se desarrollaron aplicaciones de teleoperación y visión artificial que complementan el trabajo realizado.

CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL INSTITUTO POLITÉCNICO NACIONAL

UNIDAD ZACATENCO
DEPARTAMENTO DE CONTROL AUTOMÁTICO

Abstract

Design, Modeling, Construction, Analysis, Control and Perception in Humanoid Robots

by Rafael Stanley Nuñez Cruz

The purpose of this thesis is to show the development of the humanoid robot *Johnny*, which is designed with the objective of efficient energy consumption during walking, this task is presented as an optimization problem, to solve it it was necessary to identify the parameters involved in the energy consumption, which depend on the mechanical design, the characteristics of the walk, the type of control used, etc.

It was also necessary to adequately define an optimization criterion and the methodology used to find a solution, given the particular characteristics of the problem we had to make use of unconventional optimization techniques.

Once a solution to the problem was found, the production and testing of the prototype was continued, comparisons of the energy consumption were made and control laws were implemented to keep balance using different sensors.

Finally, teleoperation and artificial vision applications were developed to complement the work done.

Agradecimientos

Al CINVESTAV, compañeros, profesores y a todas
las personas involucradas en mi formación, a
Elba Antonio compañera de vida pero
sobre todo a mi familia, la base de
lo que soy y de lo que puedo ser.

Al Consejo Nacional de Ciencia y Tecnología
por el apoyo económico a través de la beca
de posgrado que se me otorgó.

Índice general

Capítulo 1

Introducción

1.1. Objetivo

El objetivo de este proyecto es el desarrollo de un robot humanoide capaz de exhibir un caminado eficiente y estable, se pretende que tenga una morfología lo más parecida posible a la del ser humano en cuanto a sus proporciones y cuyo consumo total de energía sea lo más bajo posible. Alcanzar este objetivo involucra gran trabajo en las áreas principales de la robótica humanoide:

Diseño Mecánico: La estructura mecánica de un robot humanoide es un factor determinante del consumo energético, por lo que es muy importante la realización de estudios bibliográficos e investigación personal sobre este tema para dar con una configuración mecánica eficiente.

Modelado Matemático: El diseño de un robot humanoide eficiente requerirá contar con el modelado matemático de su comportamiento dinámico considerando la mayor cantidad de fenómenos que sea posible. Para lo cual será necesario modelar las distintas etapas del caminado incluyendo etapas de soporte simple, colisiones, almacenamiento y liberación de energía, etc.

Control: Un aspecto básico en el desarrollo de un caminado eficiente será el diseño de referencias articulares así como de las leyes de control locales y externas que permitan al robot desplazarse usando la menor cantidad de energía y asegurando el equilibrio tanto en simulación como en una plataforma real.

Análisis de Estabilidad: Será necesario hacer uso de un criterio de estabilidad adecuado para las características del caminado bípedo, incluyendo las etapas discontinuas causadas por las colisiones con el piso.

Construcción: Para comprobar que los conceptos desarrollados son funcionales en plataformas reales, se construirá el robot humanoide lo que involucra la manufactura de eslabones, desarrollo de un sistema de captura de información sensorial y de un sistema embebido para la percepción y el sistema de control.

Percepción: Finalmente se desarrollarán aplicaciones basadas en visión artificial y sensores inerciales que permitan al robot resolver alguna necesidad del mundo real como mapeo del entorno o identificación de objetos.

1.2. Estado del Arte

Los robots humanoides están teniendo un auge importante en muchas áreas de la Ingeniería y están haciendo su aparición en diversos ámbitos de la actividad humana que van desde competiciones deportivas hasta actividades auxiliares en entornos médicos.

Estos robots se denominan robots de servicio, por oposición a los ahora ya clásicos robots industriales, y se caracterizan por tener una gran movilidad, por su autonomía y, sobre todo, por ser amigables con el ser humano y muy seguros.

Una parte fundamental de los robots de servicio es que son de tipo antropomorfo y se les denomina robots humanoides. La creciente importancia de dichos robots está motivada, por un lado, en su capacidad de desenvolverse en los entornos diseñados para el ser humano y utilizando sus herramientas y, por otro lado, en la expectativa de que sean capaces de asistir al ser humano en una gran cantidad de tareas cotidianas.

Además, existe la posibilidad de que este tipo de robots lleguen a ser capaces de generar conocimiento a partir de sus grandes capacidades de percepción y a su arquitectura mecánica semejante a la del ser humano, como el proyecto iCUB [Vernon, Metta y Sandini, 2007].

Es así que desde hace una decena de años han venido apareciendo prototipos comerciales baratos de pequeño formato (Robonova, Kondo, Bioloid, entre otros muchos), prototipos medianos de altas prestaciones (Qrio, Hoap, Nao, Darwin) que, en paralelo con los grandes proyectos con presupuestos exorbitantes (ASIMO [Sakagami y col., 2002], HUBO [Oh y col., 2006], HRP [Kaneko y col., 2015], Atlas [Atmeh y col., 2014]) tienen arquitecturas mecánicas robustas y requieren de importantes cantidades de energía para realizar sus movimientos.

Este tipo de robots principalmente fueron diseñados desde el punto de vista de los robots industriales, su caminado se basa en la restricción del balance dinámico en todos los instantes durante el caminado.

FIGURA 1.1: GRANDES PROYECTOS DE ROBÓTICA HUMANOIDE

Otra estrategia de diseño es el caminado basado en ciclos límite, de esta forma es posible remover la restricción de balance en cada instante de tiempo y considerar la estabilidad del ciclo de caminado. Un ejemplo de robots basados en esta estrategia es el robot Flame [Hobbelen, Boer y Wisse, 2008], proyecto que comenzó con el desarrollo de plataformas pasivas (sin control ni actuadores) y que fue evolucionado haciendo uso de elementos elásticos lo que permitió desarrollar caminados fluidos, eficientes energéticamente y muy parecidos a los humanos.

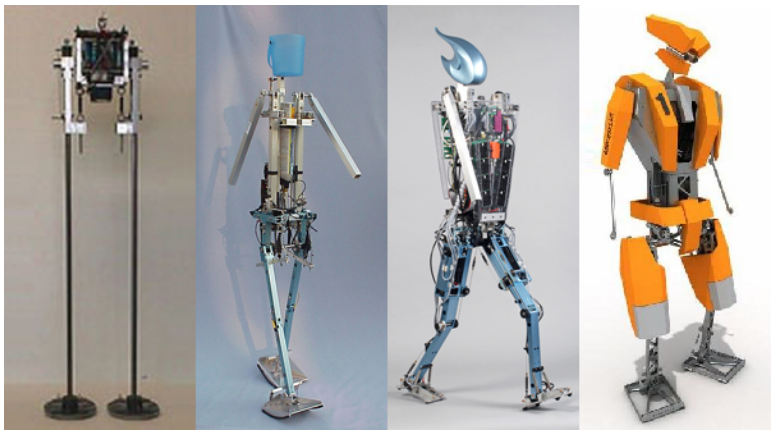


FIGURA 1.2: ROBOT HUMANOIDE *Flame*

Particularmente en nuestro laboratorio durante mas de cinco años de utilizar robots humanoides comerciales de pequeño formato (Robonova, Bioloid, Nao y DarwinOP) y de haber construido nuestro propio prototipo (AH1N1) [Hunter, Cisneros e Ibarra, 2011] hemos desarrollado algunos proyectos como el modelado geométrico, cinemático y dinámico de dichos prototipos, también se ha trabajado en el control del comportamiento cuyo objetivo es proporcionar una cierta autonomía en la toma de decisiones en tareas complejas como la de jugar fútbol.

FIGURA 1.3: ROBOTS PREVIAMENTE USADOS EN NUESTRO EQUIPO

Esta experiencia nos ha permitido confirmar personalmente que estos mecanismos son voraces consumidores de energía, llegando a consumir de manera instantánea picos de hasta 30 Ampéres.

En este contexto, en septiembre de 2009 iniciamos un proyecto sobre caminantes bípedos pasivos cuyo principal objetivo es el desarrollo de arquitecturas mecánicas cuya dinámica natural sea el caminado, cuya energía provenga tan solo del efecto de la gravedad sobre dicha estructura y cuyas proporciones sean lo más parecidas posible a las del ser humano.

La primera fase de este proyecto culminó con el desarrollo de tres diferentes prototipos [Núñez e Ibarra, 2011; Núñez, 2012] basados en los trabajos de McGeer [McGeer, 1990], los

cuales se desarrollaron considerando las diferentes arquitecturas mecánicas y técnicas de caminado encontradas en la literatura.

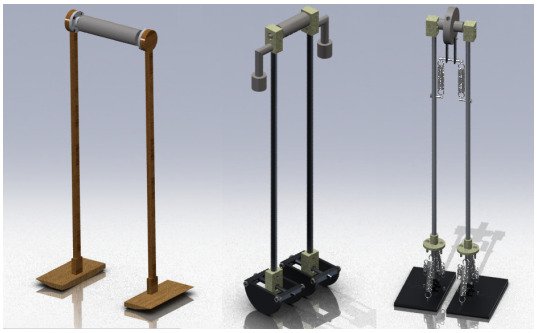


FIGURA 1.4: CDP'S DESARROLLADOS PREVIAMENTE

Con base en los resultados obtenidos en esta primera fase del proyecto ahora se pretende desarrollar un robot humanoide con un mínimo consumo de energía al cual denominamos *Johhny* y es el resultado principal de este trabajo de tesis.

FIGURA 1.5: ROBOT HUMANOIDE *Johhny*

1.3. Enfoque

Se propone abordar el desarrollo de este prototipo como un problema de optimización, inicialmente para identificar cuales son las variables de optimización se estudiaron el modelo matemático de la caminata, las leyes de control usados en sistemas cíclicos y el diseño de trayectorias de caminado.

Posteriormente se define el criterio de optimización y las restricciones del problema. A continuación se construye un simulador para resolver el modelo dinámico del caminado mediante integraciones numéricas, debido a que no será posible obtener una expresión explicita del criterio de optimización elegido habrá que hacer uso de técnicas de

optimización también numéricas como el caso de algoritmos genéticos o la computación evolutiva para encontrar la solución óptima.

Una vez que se encuentra la solución optimizada se procede a la construcción del prototipo, así como a la programación de las trayectorias y las leyes de control definidas.

Después se diseña un lazo de control externo que permita al robot mantener el equilibrio ante perturbaciones usando mediciones inerciales y angulares en sus eslabones.

Tras obtener un caminado estable se compara la eficiencia del prototipo construido con respecto a algún robot comercial de características parecidas.

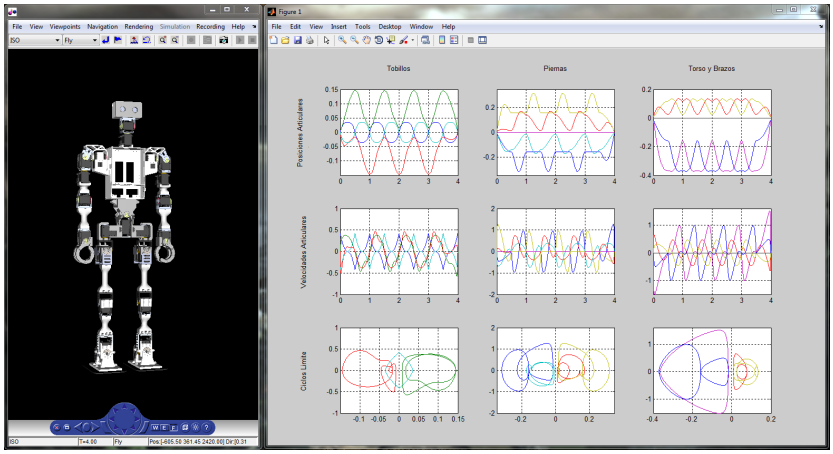


FIGURA 1.6: SIMULADOR DINÁMICO

1.4. Logros

- Se desarrolló un simulador numérico capaz de modelar las 18 articulaciones involucradas durante el caminado, el cual permite modelar la perdida de energía que ocurre durante la transferencia del punto de apoyo y permite definir trayectorias paramétricas para la posición de la cadera, los brazos y el pie oscilante, encargándose internamente de la cinemática inversa para construir las referencias articulares.
- Se diseñó un vínculo entre los programas ®Matlab y ®SolidWorks que permite modificar, mediante programación, algunas dimensiones de los eslabones como longitudes y diámetros y devolver automáticamente la información actualizada de la ubicación de su centro de masa y tensor de inercia.
- Se implementaron distintos algoritmos de optimización numérica como algoritmos genéticos multiobjetivo con o sin restricciones y otros como el método Simplex con el fin de resolver el problema de optimización del caminado usando el simulador y el vínculo antes mencionados.
- Se construyó el prototipo *Johhny* usando distintos métodos de manufactura como impresión 3D y corte CNC mediante láser y fresadora, se utilizaron motores dynamixel MX-28 para el movimiento de las articulaciones.
- Se embarcó, en el robot, el trazador de trayectorias para la generación de referencias articulares en los motores, lo que permite al sistema actualizar las señales de referencia en linea y enviarlas a los actuadores correspondientes.

- Se diseñaron lazos de control externo para mantener el equilibrio durante el caminado usando información inercial y lecturas angulares en las articulaciones.
- Se diseñaron aplicaciones de teleoperación y visión artificial usando dispositivos ®Android que permiten interactuar con el robot.

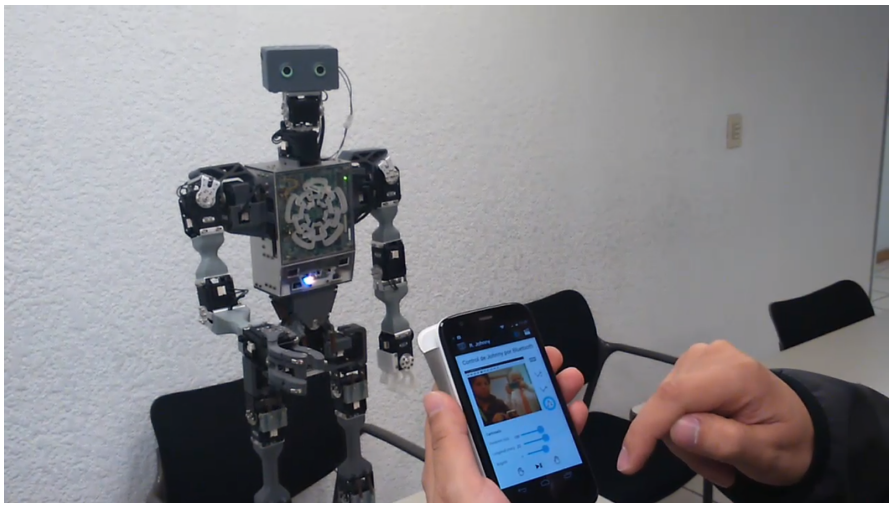


FIGURA 1.7: TELEOPERACIÓN

1.5. Artículos Publicados

Durante el trabajo de tesis se realizaron las siguientes publicaciones sobre el trabajo parcial desarrollado, las cuales fueron presentadas en distintos ámbitos:

Año	Articulo	Revista/Congreso
2017	Efficient mechanical design and limit cycle stability for a humanoid robot: An application of genetic algorithms	Neurocomputing
2016	Active disturbance rejection control for humanoid stable walking	13th Int. conf. on electrical engineering, computing sc. and automatic control
2016	Inertial-Visual odometry on mobile devices	Open conference on future trends in robotics
2015	Optimal design for a humanoid robot based on passive dynamic walkers and genetic algorithms	12th Int. conf. on electrical engineering, computing sc. and automatic control
2015	Johhny: caminado parametrizado y teleoperación mediante dispositivos móviles	AMRob Journal, robotics theory and applications
2014	Johnny: Desarrollo de un robot humanoide óptimo basado en caminantes dinámicos pasivos	AMRob Journal, robotics theory and applications
2013	Diseño de estrategias de control para caminantes basados en bípedos pasivos	AMRob Journal, robotics theory and applications

TABLA 1.1: PUBLICACIONES

1.6. Organización de la Tesis

Cada uno de los capítulos siguientes pretende resolver alguno de los objetivos particulares definidos en la sección 1.1 y contienen su propio estudio bibliográfico cuando sea el caso.

En el capítulo 2, se estudia el modelo matemático de los caminantes pasivos, ya que la estructura mecánica del robot estará basada en este tipo de máquinas, también se estudia el modelo del caminado bípedo y las particularidades del modelado del robot humanoide, que incluye brazos y torso, finalmente se definirá el concepto de estabilidad que se utilizará a lo largo de la tesis.

En el capítulo 3 se hace una comparativa de las leyes de control comúnmente usadas en el manejo de sistemas cíclicos, lo cual nos permite identificar los parámetros de optimización relacionados a las leyes de control usadas y las trayectorias de referencia.

En el capítulo 4 se define el caminado parametrizado que habrá de regir el comportamiento de los eslabones de la unidad motriz (piernas) y la unidad pasajera (torso y brazos).

En el capítulo 5 se muestran distintos enfoques usados para definir el problema de optimización de la caminata dependiendo de las variables de optimización elegidas y el criterio de optimización seleccionado. También se definen los métodos de optimización usados en cada uno de estos enfoques y los resultados obtenidos.

En el capítulo 6 se habla de la construcción del prototipo *Johhny*, de los métodos de manufactura de sus eslabones, los materiales usados y detalles de su ensamble. También se mencionan algunos aspectos generales de la plataforma informática que permite la comunicación con los actuadores y los sensores usados.

En el capítulo 7 se considera el uso de sensores para generar un lazo de control externo y que permita controlar el equilibrio del caminando, se analizan el tratamiento de las señales y el uso de modelos reducidos que permitan identificar y contrarrestar perturbaciones externas.

En el capítulo 8 se muestra el diseño de los experimentos que permita comparar la eficiencia energética de nuestro prototipo con respecto a alguna plataforma comercial así como los resultados obtenidos.

En el capítulo 9 se describen algunas aplicaciones implementadas en el robot que permiten realizar tareas básicas de teleoperación y de visión artificial. Se muestran detalles de los algoritmos usados así como de su implementación y resultados.

En el capítulo 10 se habla finalmente de las conclusiones del trabajo de tesis así como del trabajo futuro en esta área.

Bibliografía del Capítulo 1

Atmeh, G. y col. (2014). «Implementation of an adaptive, model free, learning controller on the Atlas robot». En: *American Control Conference*, págs. 2887-2892.

Hobbelen, D., T. Boer y M. Wisse (2008). «System overview of bipedal robots Flame and TUlip tailor made for Limit Cycle Walking». En: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, págs. 2486-2491.

Hunter, M., R. Cisneros y J.M. Ibarra (2011). «Mechanical design and kinematic analysis of the AH1N1 humanoid robot». En: *Proc. of the IEEE 21st International Conference on Electronics, Communications and Computers, CONIELECOMP*, págs. 177-183.

Kaneko, K. y col. (2015). «Humanoid robot HRP-2Kai -Improvement of HRP-2 towards disaster response tasks-». En: *IEEE-RAS 15th International Conference on Humanoid Robots*, págs. 132-139.

McGeer, T. (1990). «Passive Dynamic Walking». En: *International Journal of Robotics Research* 9, págs. 62-82.

Núñez, R.S. (2012). «Diseño, análisis y construcción de un robot bípedo pasivo». México city, México: CINVESTAV, Departamento de Control Automático.

Núñez, R.S. y J.M. Ibarra (2011). «Desarrollo de Bípedos Pasivos». En: *Memorias del CoMRob 2011, XIII Congreso Mexicano de Robótica de la AMRob*, págs. 69-74.

Oh, J. y col. (2006). «Design of Android type Humanoid Robot Albert HUBO». En: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, págs. 1428-1433.

Sakagami, Y. y col. (2002). «The intelligent ASIMO: system overview and integration». En: *IEEE/RSJ International Conference on Intelligent Robots and Systems* 3, págs. 2478-2483.

Vernon, D., G. Metta y G. Sandini (2007). «The iCub cognitive architecture: Interactive development in a humanoid robot». En: *IEEE 6th International Conference on Development and Learning*, págs. 122-127.

Capítulo 2

Modelos Matemáticos

2.1. Caminantes Pasivos

Un caminante dinámico pasivo (CDP) es una máquina con dos piernas diseñada para caminar de manera estable, sin necesidad de contar con actuadores ni sistema de control. El movimiento de esta clase de máquinas articuladas es un fenómeno producido principalmente por el efecto de la gravedad sobre sus eslabones así como por el equilibrio entre las energías potencial y cinética del mecanismo. Los caminantes dinámicos pasivos exhiben un caminado estable cuando se encuentran descendiendo por un plano inclinado y le han sido proporcionadas las condiciones iniciales de posición y velocidad adecuadas.

McGeer, pionero en el estudio de los CDP [McGeer, 1990a], comenzó construyendo un caminante bípedo con movimiento bidimensional (2D) exclusivamente sobre el plano sagital cuyos pies tienen redondeada la planta, estudiándolo para demostrar la capacidad de los CDP de caminar de manera estable sin control ni actuación. La metodología de análisis utilizada consiste en encontrar las condiciones iniciales de posición y velocidad adecuadas para producir un ciclo límite para un conjunto dado de parámetros físicos del modelo de caminata utilizado. Si no se consigue el ciclo límite buscado debe definirse un nuevo conjunto de parámetros físicos a fin de continuar la búsqueda. Para ello se modifican ligeramente los valores de algunas dimensiones físicas como la longitud de las piernas, la distribución de masas o el radio de la curvatura de la planta de los pies, entre otros.

La mayoría de las veces es posible encontrar más de un conjunto de parámetros que producen una caminata estable, de modo que cada una de estas posibles soluciones producirá un mecanismo actuado diferente. Cada una de estas soluciones requiere, en general, una cantidad de energía también diferente para caminar sobre una plataforma horizontal.

Después del trabajo seminal de McGeer aparecieron resultados de estudios sobre nuevos prototipos de CDP. Por ejemplo, McGeer analizó un nuevo prototipo que consistió en agregar una nueva articulación a cada pierna a manera de rodilla que permitía acortar la longitud total de la pierna para evitar chocar con el suelo en la fase de balanceo [McGeer, 1990b].

Coleman and Ruina construyeron un prototipo bípedo con pies cuya suela redondeada en dos ejes que le permite al CDP movimiento 3D en los planos sagital y frontal en donde los pies poseen una barra estabilizadora que permite bajar el centro de gravedad incrementando la estabilidad de la marcha [Coleman y Ruina, 1998].

Wisse diseñó un prototipo de CDP con pies cilíndricos y tobillos articulados que permitió la posterior construcción de un humanoide neumático [Wisse, Schwab y van der Linde, 2001]. Por su parte Collins desarrolló una sofisticada máquina caminante cuyas piernas se movían de manera coordinada con el balanceo del torso y de los brazos [Collins, Wisse y Ruina, 2001]. Por su parte, Narukawa desarrolló un CDP con pies planos que usa resortes

en los tobillos para modificar eficientemente la frecuencia del ciclo límite de la marcha [Narukawa y col., 2009].

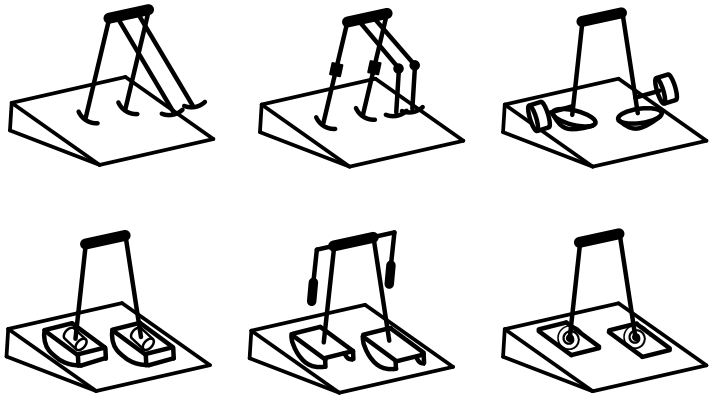


FIGURA 2.1: ALGUNAS ESTRUCTURAS DE CDP'S

En la FIGURA 2.1 se muestran algunas de estas arquitecturas usadas en los CDPs. Del estudio de trabajos previos en el área de diseño y análisis de CDP hemos extraído algunas características importantes a considerar en el diseño de nuestro humanoide tales como el uso de pies planos, tobillos articulados, resortes en todas las articulaciones de la unidad locomotora, masas adicionales en los eslabones de las piernas, torso articulado en la cintura y brazos. El total de articulaciones del humanoide desarrollado son 26 distribuidas como sigue: 6 gdl en cada pierna, 4 gdl en cada brazo, uno mas en cada mano (agarre), dos en el cuello y dos en la cintura. La cadena cinemática correspondiente se muestra en la FIGURA 2.3.

2.2. Modelo de la Caminata

La marcha del caminante se modela como la interconexión de etapas continuas con eventos discretos. En la FIGURA 2.2 se muestra un ciclo de caminado formado por cinco etapas, en donde el vector de estado al final de cada etapa se usa como condiciones iniciales de la siguiente etapa. A continuación se describen estas cinco etapas.

I) *Fase de Soporte Simple.* Aquí el caminante está apoyado en una sola pierna mientras que la otra se balancea hacia adelante, este movimiento es gobernado por un sistema de ecuaciones diferenciales no lineales las cuales se pueden obtener aplicando la metodología de Euler-Lagrange o la de Newton-Euler considerando la cadena cinemática arborecente de la FIGURA 2.3. Este modelo se integra hasta que se detecta la condición de doble soporte.

II) *Rodilla bloqueada.* En los CDPs las rodillas poseen un mecanismo de bloqueo para evitar la hiperextensión, cuando esto ocurre se produce una colisión que genera una discontinuidad en las velocidades articulares. Durante la fase de soporte simple se tiene que detectar el bloqueo de la rodilla y recalcular las velocidades, usando la conservación del momento angular, antes de continuar con la integración del modelo.

III) *Apoyo del talón.* En los robots equipados con pies planos, el pie correspondiente a la pierna balanceándose toca el suelo con el talón antes de apoyar la planta del

pie completamente en el suelo. Generalmente, este fenómeno no se incluye en el modelado, sin embargo, cuando los tobillos cuentan con resortes es necesario modelar la conversión de energía cinética en potencial producida por la compresión de los mismos. Esto se hace disminuyendo la velocidad articular del tobillo del pie apoyado en el piso de tal forma que la disminución de energía cinética sea igual a la energía potencial almacenada en los resortes.

IV) *Fase de doble soporte.* Cuando el pie que se balancea toca completamente el suelo con toda su superficie ocurre otra colisión y el punto de apoyo se transfiere ahora a este pie. Este evento puede considerarse instantáneo de modo que la posición del pie es la misma antes y después del impacto cambiando solo la velocidad. Esta colisión también se modela usando la conservación del momento angular. Cabe mencionar que las ecuaciones de movimiento deben reescribirse pues el referencial de base ha pasado al pie opuesto.

V) *Levantado del talón.* Este es el fenómeno opuesto a la etapa III. Al momento en que el nuevo pie oscilante se despega del suelo los resortes del tobillo liberan la energía potencial almacenada empujando el pie, y por tanto al robot, hacia arriba y adelante. Este fenómeno se modela como una conversión de energía potencial en cinética, para lo cual se aumenta la velocidad articular del tobillo del pie apoyado en el piso de tal forma que el aumento de energía cinética sea igual a la energía potencial almacenada en los resortes.

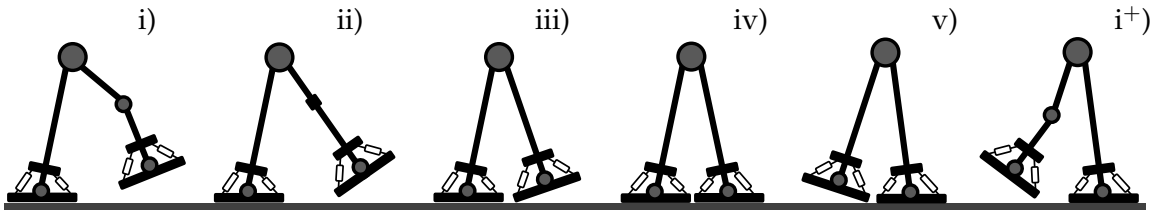


FIGURA 2.2: LAS 5 ETAPAS DEL CAMINADO

La secuencia de estas cinco etapas constituyen un ciclo completo de caminado, en donde el vector de condiciones finales de cualquiera de ellas se constituye en el vector de condiciones iniciales de la siguiente. En muchos casos es difícil representar el modelo del movimiento en una forma cerrada de la forma:

$$x^{k+1} = M(x^k) \tag{2.1}$$

En donde $x^k = [q^k \dot{q}^k]$ representa el vector de estado formado por la posición angular y la velocidad en cada articulación al inicio del paso k.

Para obtener las trayectoria articulares del caminante dadas unas condiciones iniciales x0, debe evaluarse la ECUACIÓN 2.1 mediante integraciones numéricas del sistema no lineal correspondiente en función de la etapa actual dentro del ciclo de caminado, en la forma explicada anteriormente.

2.2.1. Modelado de la Etapa de Soporte Simple

En esta fase se considera que el robot esta formado por una cadena cinemática arborescente donde el marco de referencia inercial está asociado al pie de apoyo, esta cadena

consta de tres ramificaciones: la primera para el pie que se balancea (indicada en color rojo) y las otras dos para los brazos (indicadas en verde y azul).

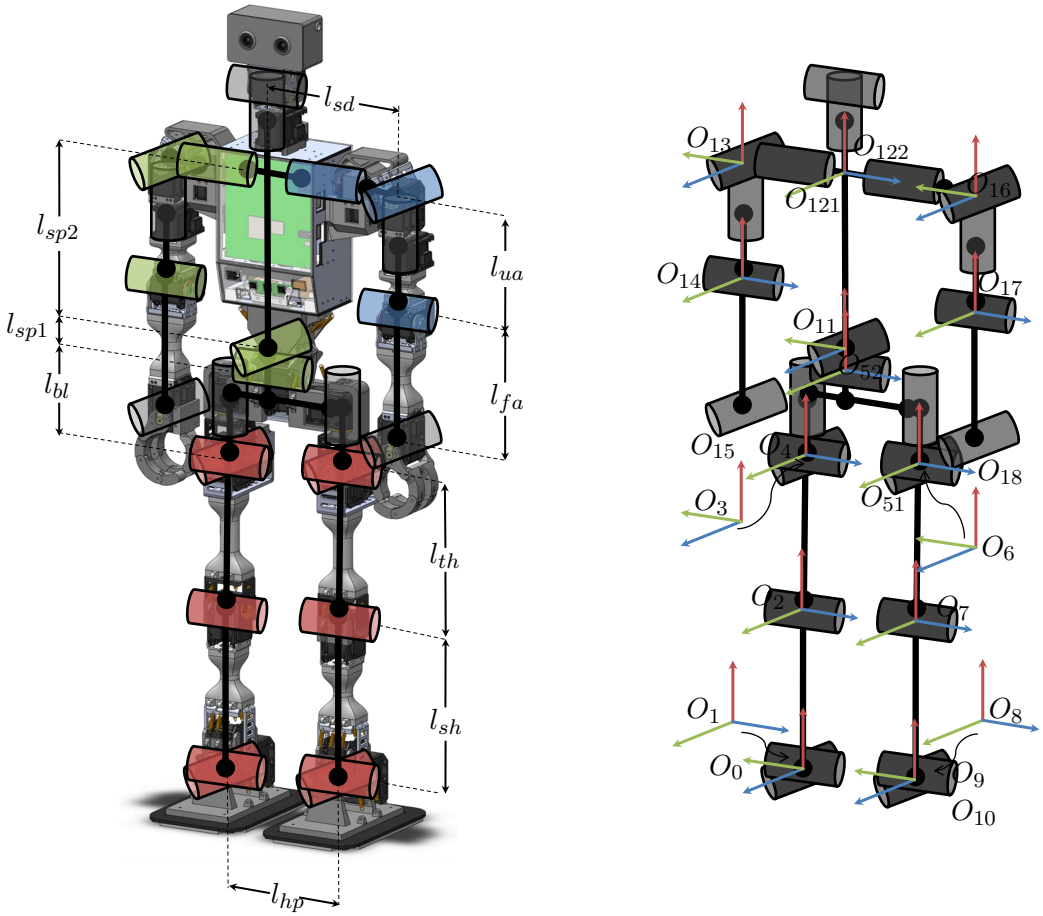


FIGURA 2.3: CADENA CINEMÁTICA DE JOHNNY

Para calcular el modelo cinemático directo se utilizó la convención para definir los ejes de coordenadas mostrada en [*Robot Modeling and Control*]:

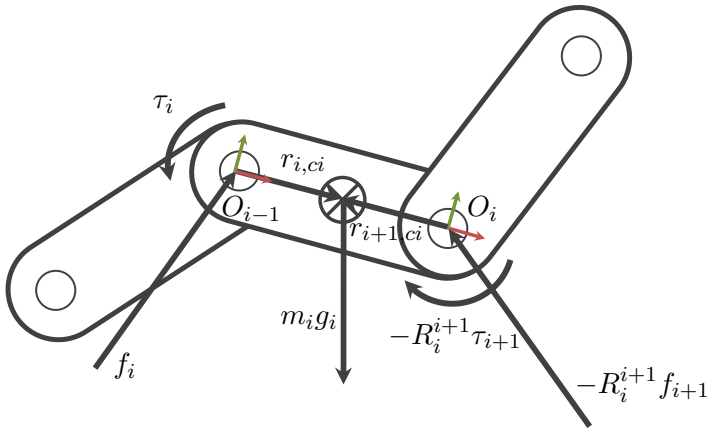


FIGURA 2.4: DIAGRAMA DE CUERPO LIBRE DEL ESLABÓN i

Usando esta convención y a las longitudes mostradas en la FIGURA 2.3 la siguiente tabla de parametros de Denavitt-Hartenberg define el modelo cinemático del robot:

$O_{i-1} - O_i$	θ	d	a	α
$O_0 - O_1$	q_1	0	0	$\pi/2$
$O_1 - O_2$	q_2	0	l_{sh}	0
$O_2 - O_3$	q_3	0	l_{th}	$-\pi/2$
$O_3 - O_4$	q_4	0	0	$\pi/2$
$O_4 - O_{51}$	q_5	l_{hp}	0	0
$O_{51} - O_6$	q_6	0	0	$-\pi/2$
$O_6 - O_7$	q_7	0	$-l_{th}$	$\pi/2$
$O_7 - O_8$	q_8	0	$-l_{sh}$	0
$O_8 - O_9$	q_9	0	0	$-\pi/2$
$O_9 - O_{10}$	q_{10}	0	$-l_{ft}$	0
$O_4 - O_{52}$	q_5	$l_{hp}/2$	l_{bl}	0
$O_{52} - O_{11}$	q_{11}	0	l_{sp1}	$-\pi/2$
$O_{11} - O_{121}$	q_{12}	0	l_{sp2}	$\pi/2$
$O_{121} - O_{13}$	q_{13}	$-l_{sd}$	0	$-\pi/2$
$O_{13} - O_{14}$	q_{14}	0	$-l_{ua}$	$\pi/2$
$O_{14} - O_{15}$	q_{15}	0	$-l_{fa}$	$-\pi/2$
$O_{11} - O_{122}$	q_{12}	0	l_{sp2}	$\pi/2$
$O_{122} - O_{16}$	q_{16}	l_{sd}	0	$-\pi/2$
$O_{16} - O_{17}$	q_{17}	0	$-l_{ua}$	$\pi/2$
$O_{17} - O_{18}$	q_{18}	0	$-l_{fa}$	$-\pi/2$

TABLA 2.1: TABLA DE PARÁMETROS DE D-H

Las masas, longitudes al centro de masa y los tensores de inercia de cada eslabón se obtienen mediante algún software de CAD, con lo cual se obtiene toda la información necesaria para evaluar el algoritmo de Newton-Euler.

Es importante señalar que la formulación de Euler-Lagrange no es apropiada para su uso en simulaciones numéricas cuando el número de articulaciones es muy grande, esto se debe a que el orden de complejidad de esta formulación es $O(n^4)$, por otro lado la forma recursiva de las ecuaciones de Newton-Euler es de orden $O(n)$ [Fu, Gonzalez y Lee, 1987].

Considerando que $n = 17$ para el modelo usado, evaluar la formulacion de Euler-Lagrange requiere realizar $17^3 = 4913$ veces mas operaciones que la formulación de Newton-Euler.

Esto pudo comprobarse empíricamente ya que el modelo generado con Euler-Lagrange ni siquiera pudo ser evaluado en una implementación hecha en Matlab porque la memoria requerida para lograrlo superaba la capacidad de la disponible de una laptop convencional.

2.2.2. Modelado de las Colisiones

La relación entre las velocidades articulares antes y despues de una colisión se encuentra usando la conservación del momento angular. El momento angular del eslabón i con respecto al punto de contacto cp se calcula de la siguiente manera:

$$M_{cp}^{i} = I_{iner}^{i}\omega_{iner}^{cm_i} + P_{cp}^{cm_i} \times m_i v_{iner}^{cm_i} \tag{2.2}$$

Donde $P_{cp}^{cm_i}$ representa la posición del centro de masa i con respecto al punto de contacto cp.

Representando las velocidades angulares y lineales del centro de masa i en función de el Jacobiano de velocidades angulares $J_{\omega,i}$ y lineales $J_{v,i}$ la representación del momento angular se puede reescribir como:

$$M_{cp}^{i} = I_{iner}^{i}J_{\omega,i}\dot{q} + P_{cp}^{cm_i} \times m_i J_{v,i}\dot{q} \tag{2.3}$$

Para este sistema es posible expresar el momento angular en forma matricial $M_{cp}^{i} = A_i(q)\dot{q}$. Donde es:

$$A_i(q) = I_{iner}^{i}J_{\omega,i} + m_i \left[P_{cp}^{cm_i} \times J_{v,i,1} \;\; P_{cp}^{cm_i} \times J_{v,i,2} \;\; \ldots \;\; P_{cp}^{cm_i} \times J_{v,i,n}\right] \tag{2.4}$$

En donde $J_{v,i,n}$ representa la columna n del Jacobiano de velocidades lineales del centro de masa i.

Igualando el momento angular antes y después del impacto (superíndices - y + respectivamente) es posible obtener la expresión para calcular las velocidades después del impacto:

$$\dot{q}^{+} = (A^{+}(q^{+}))^{-1}A^{-}(q^{-})\dot{q}^{-} \tag{2.5}$$

Cabe señalar que en la implementación de este algoritmo se utilizó la función `pinv()` que es la pseudo inversa de una matriz, ya que por lo general la matriz A no es cuadrada.

2.3. Estabilidad de Ciclos Límite

De acuerdo con el paradigma de caminado basado en ciclos límite [Hobbelen y Wisse, 2007], con el fin de encontrar caminados más eficientes, naturales, rápidos y robustos, es necesario reducir las restricciones artificiales añadidas al utilizar otros criterios de estabilidad como aquellos basados en el punto de Momento Cero [Vukobratovic y Borovac, 2004].

La definición más genérica de caminado estable, sin restricciones artificiales, es *evitar la caída*. El uso de esta definición implica evaluar todas las posibles trayectorias de caminado y evaluar si el caminante cae o no. Para utilizar esta definición de una manera práctica, proponemos el uso de algoritmos genéticos, considerando la tarea como un problema de optimización.

La definición formal de este tipo de estabilidad es el siguiente: *El caminado basado en ciclos límite es una secuencia de pasos nominales periódicos que es estable en su conjunto, pero no es localmente estable en cada instante de tiempo.*

En este tipo de estabilidad, también llamada *Cíclica* u *Orbital*, la secuencia de pasos nominales se refiere a un movimiento periódico del caminante observado en ausencia de perturbaciones. Este movimiento periódico es estable cuando trayectorias suficientemente cercanas al movimiento nominal se aproximan al ciclo límite a través de múltiples iteraciones como se muestra FIGURA 2.5.

Para analizar esta definición de estabilidad se considera el mapeo $M(x)$ definido en la ECUACIÓN 2.1.

El mapeo $M(x)$ representa un movimiento periódico cuando se cumple la restricción $r_1(x^*) = 0$, donde:

$$r_1(x) = M(x) - x \tag{2.6}$$

El mapeo $M(x)$ generalmente se le conoce como mapeo de Poincaré y al vector de estados x^* se le conoce como punto fijo y en este caso pertenece al conjunto $\sum$ de vectores de condiciones iniciales.

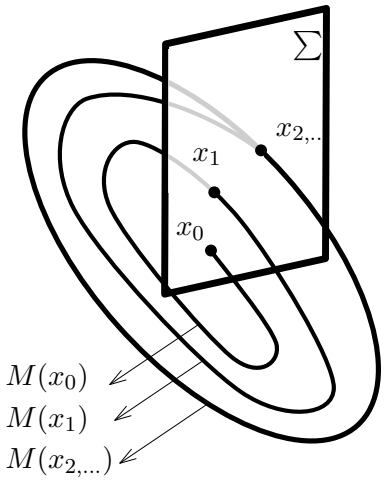


FIGURA 2.5: CICLO LÍMITE

La estabilidad de un punto fijo usando el esquema de la dinámica no lineal puede ser definida como [Coleman, 1998]: El punto fijo x^* es estable si, para cualquier $\epsilon > 0$, existe un $\delta > 0$ tal que siempre que $|x^0 - x^*| < \epsilon$, $|x^n - x^*| < \delta$ para cualquier $n > 0$.

Es posible obtener un criterio de estabilidad para un punto fijo mediante la linearización de $M(x)$ y considerando pequeñas perturbaciones. La linearización de $M(x)$ en x^* es:

$$M(x^k) = M(x^*) + J(x^*)(x^k - x^*) \tag{2.7}$$

Donde $J(x^*) = \frac{\partial M}{\partial x}(x^*)$ es la matriz Jacobiana de M.

Considerando pequeñas perturbaciones de la forma:

$$x^k = x^* + \Delta x_k \tag{2.8a}$$

$$x^{k+1} = x^* + \Delta x_{k+1} \tag{2.8b}$$

Substituyendo ECUACIÓN 2.8 en ECUACIÓN 2.7 es posible formular la dinámica de la perturbación Δx.

$$\Delta x_{k+1} = J(x^*)\Delta x_k \tag{2.9}$$

Entonces un punto fijo x^* será estable si la magnitud de todos los eigenvalores σ_i de $J(x^*)$ son menores a 1. En este caso las pequeñas perturbaciones decreceran paso a paso. La restricción de que un punto fijo sea estable se puede reescribir como $r_2(x) < 1$ donde:

$$r_2(x) = \max_i(|\sigma_i|) \tag{2.10}$$

Según este análisis, la estabilidad del ciclo de caminado se puede obtener mediante la definición de movimientos periódicos de manera que las condiciones iniciales del paso son puntos fijos estables. Por lo general el método de Newton-Rapshon o algoritmos similares se utilizan para encontrar las raíces de la ecuación ECUACIÓN 2.6 para obtener puntos fijos, sin embargo, debe tenerse en cuenta que el algoritmo puede fallar si el valor inicial no es lo suficientemente cercano a la solución o si no hay puntos fijos para ese conjunto de parámetros.

En el caso de un mecanismo de muchos grados de libertad, tales como el robot humanoide que estamos considerando, será muy difícil obtener movimientos periódicos dadas únicamente las condiciones iniciales del paso, por lo que decidimos asignar trayectorias de referencia periódicas por lo que las condiciones de la ECUACIÓN 2.6 se cumplen si el robot es capaz de seguir esas trayectorias sin caer.

Después de que se haya definido una trayectoria de referencia, el mapeo $M(x)$ se puede calcular mediante integración numérica del modelo y el uso de un control de retroalimentación, entonces la matriz Jacobiana también puede calcularse numéricamente para evaluar si las referencias producen un ciclo de la marcha estable. Este procedimiento requiere un modelo preciso de los actuadores de modo que los pares de control en la simulación pueden representar la capacidad de los actuadores reales.

Bibliografía del Capítulo 2

Coleman, M. (1998). «A stability study of a three dimensional passive-dynamic model of humanoid gait». Cornell University.

Coleman, M.J. y A. Ruina (1998). «An uncontrolled walking toy that cannot stand still». En: *Physical Review Letters*, págs. 3658-3661.

Collins, S.H., M. Wisse y A. Ruina (2001). «A three-dimensional Passive Dynamic Walking Robot with Two Legs and Knees». En: *International Journal of Robotics Research* 20, págs. 607-615.

Fu, K. S., R.C. Gonzalez y C.S.G. Lee (1987). *Robotics: Control, sensing, vision, and intelligence*. Mcgraw-Hill Book Company.

Hobbelen, D. y M. Wisse (2007). *Humanoid Robots, Human-like Machines: Chapter Limit Cycle Walking*. I-Tech Education y Publishing.

McGeer, T. (1990a). «Passive Dynamic Walking». En: *International Journal of Robotics Research* 9, págs. 62-82.

— (1990b). «Passive Dynamic Walking with knees». En: *Proc. IEEE Conf. Robotics and Automation*, págs. 1640-1645.

Narukawa, T. y col. (2009). «Design and Construction of a Simple 3D Straight-Legged Passive Walker with Flat Feet and Ankle Springs». En: *JSME Journal of System Design and Dynamics*.

Spong, M., S. Hutchinson y M. Vidyasagar. *Robot Modeling and Control*. JOHN WILEY y SONS, INC.

Vukobratovic, M. y B. Borovac (2004). «Zero-Moment Point thirty five years of its Life». En: *International Journal of Humanoid Robotics* 1, págs. 157-173.

Wisse, M., A.L. Schwab y R.Q. van der Linde (2001). «A 3D passive dynamic biped with yaw and roll compensation». En: *Robotica* 19.

Capítulo 3

Control articular

El propósito de este capítulo es mostrar la manera en que se puede optimizar la cantidad de energía usada por un robot diseñado en base a un caminante pasivo en el seguimiento de una trayectoria de caminado de referencia. Para lograrlo se analizan estrategias de control aplicadas originalmente a sistemas cíclicos, las cuales son comparadas con la finalidad de hacer una propuesta original al replantear la función objetivo a minimizar. Se presenta la metodología de diseño así como los resultados de simulación obtenidos.

3.1. Control para robots humanoides basados en caminantes pasivos

La premisa del diseño de robots basado en bípedos pasivos es obtener un caminante que consuma la mínima cantidad de energía cuando se le agreguen actuadores y se le haga caminar sobre un plano horizontal, sin embargo esto podrá o no lograrse dependiendo de la estrategia de control usada.

Acostumbrados a los problemas de regulación, en un inicio se propuso usar las trayectorias de los caminantes pasivos como señales de referencia para sus contrapartes activas [Sukuzi y col., 2001]. En este enfoque se considera que sólo es necesario aplicar el par producido por el campo gravitacional rotado en las articulaciones del robot para que éste replique las trayectorias del caminante pasivo. Además se agrega una retroalimentación de estados para darle al caminante robustez ante perturbaciones.

Sin embargo, no es posible asegurar que las trayectorias del caminante pasivo sean las óptimas para el caminante activo, debido a la diferencia en la inclinación del plano en el que se mueven los caminantes.

Debido a que la caminata es la dinámica natural de los bípedos pasivos incluso es posible controlar cada una de las articulaciones de sus contrapartes activas de manera local [Anderson y col., 2005] con una trayectoria de referencia en función del tiempo, de tal forma que al seguir las mismas consignas en cada paso se logra un movimiento periódico y estable.

Otro método implementado en este ámbito es el presentado en [Tedrake, Zhang y Seung, 2005], en donde haciendo uso de algoritmos de aprendizaje, es posible hacer un seguimiento de las trayectorias del caminante pasivo y al mismo tiempo disminuir la energía usada si ésta se incluye en la función objetivo, sin embargo de nuevo se restringe el objetivo de control al seguimiento de las señales dadas.

Gracias a la estabilidad inherente de los caminantes pasivos es posible controlar las versiones activas, incluso en lazo abierto, con una máquina de estados [Collins y Ruina, 2005] y lograr caminatas que utilizan una cantidad mínima de energía. Sin embargo, con este

tipo de técnicas se pierde versatilidad para tareas que requieran modificar características del caminado como velocidad o dirección de la caminata.

En el ámbito de la optimización de trayectorias de caminado hay trabajos muy completos, algunos de los cuales incluyen el uso de algoritmos evolutivos para las trayectorias óptimas, en el caso del control usando ZMP [Vukobratovic y Borovac, 2004].

3.2. Control de sistemas cíclicos

La naturaleza cíclica de la caminata resulta evidente, por lo que la aplicación de técnicas de control que saquen provecho de las características periódicas de los sistemas para reducir el costo energético de los actuadores resulta por demás beneficioso a nuestros objetivos.

Para la implementación de los algoritmos en este capítulo, se considera el sistema conocido como *Van der Pol System in Reverse Time* en donde $x(t) \in \mathcal{R}^n$ con $n = 2$ el cual es un oscilador inestable en lazo abierto definido por las ecuaciones:

$$\dot{x_1} = -x_2$$
$$\dot{x_2} = x_1 - \varepsilon(1 - x_1^2)x_2 + \tau$$

Con la finalidad de decidir de una manera objetiva la estrategia a aplicar en el modelo obtenido para el caminante actuado se hizo una comparativa del desempeño de los 7 algoritmos presentados usando este modelo de prueba.

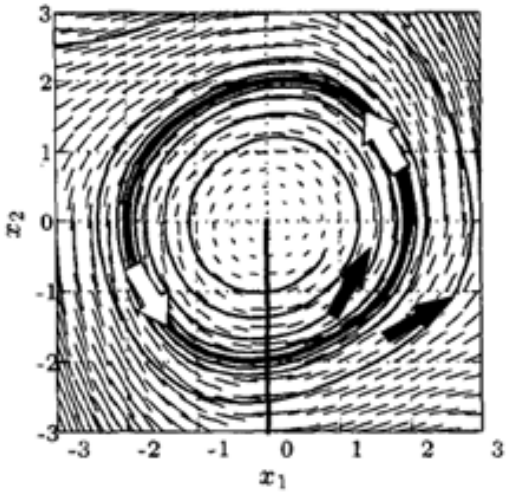


FIGURA 3.1: SISTEMA VAN DER POL

Con la finalidad de que los resultados de los distintos algoritmos pudieran ser comparables, se utilizaron las mismas condiciones iniciales y la misma trayectoria de referencia:

$$x(0) = \begin{bmatrix} 0 \\ -0.5 \end{bmatrix} \quad x_d(t) = \begin{bmatrix} 2\sin\omega t \\ -2\omega\cos\omega t \end{bmatrix}$$

El parámetro de la planta se se definió como $\varepsilon = 0.15$ y la frecuencia de las señales de referencia es $\omega = 0.8$.

3.2.1. Control por Retroalimentación de Estados

El enfoque tradicional [*Optimal Control Theory: An Introduction*] de regulación usando retroalimentación de estados puede aplicarse exitosamente al problema de seguimiento de una referencia periódica mediante la correcta selección de las ganancias de retroalimentación. Sin embargo, encontrar la ganancia óptima que minimice la energía consumida no es una tarea fácil.

El uso de técnicas como LQR (Linear–quadratic regulator) tan sólo ayudará a encontrar una aproximación ya que el modelo lineal aproximado sólo tendrá cierta validez para estados suficientemente cercanos al punto de interés.

En este caso la ley de control $\tau_r(t)$ tiene la forma de la ECUACIÓN 3.2, la cual corresponde a un control PD ya que se puede considerar que el vector de estados $x(t)$ se forma por la posición y la velocidad del sistema.

$$\tau_r(t) = -\begin{bmatrix} k_1 & k_2 \end{bmatrix} e(t) \tag{3.2}$$

Donde $e(t) = x(t) - x_d(t)$ es el error de seguimiento entre el estado de la planta y el estado deseado.

Con la finalidad de que la comparativa a realizarse fuera válida, se utilizaron las mismas ganancias en todas las consignas que utilizarán una retroalimentación de estados como parte de su ley de control, de esta forma se definen: $k_1 = 0.1$ y $k_2 = 4$.

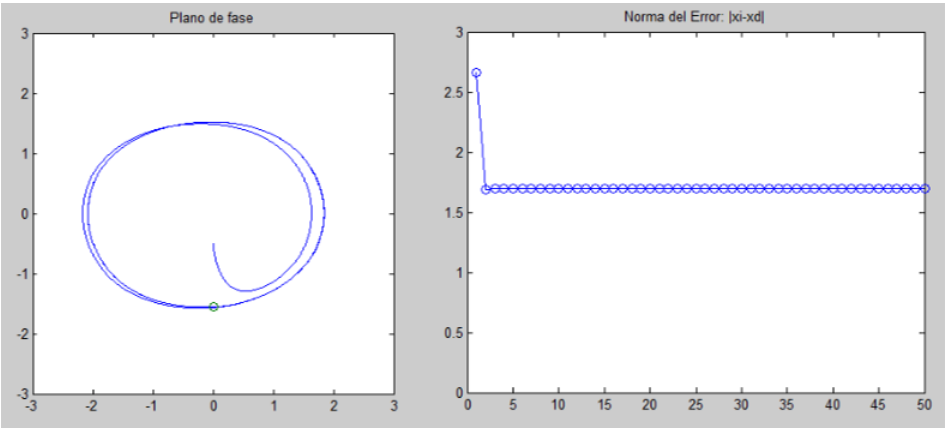


FIGURA 3.2: CONTROL POR RETROALIMENTACIÓN DE ESTADOS

En la FIGURA 3.2 se muestra, del lado izquierdo, el plano de fase de x_1 con respecto a x_2 y del lado derecho se muestra la magnitud del error al inicio de cada ciclo $e_k(0)$. Esta convención se mantiene para las figuras contenidas en este capítulo donde se muestren los resultados de la implementación de distintos algoritmos.

Este tipo de control no aprovecha las características cíclicas de la planta o de la señal de referencia ya que fue diseñado para un sistema de características mas generales, sin embargo, adicionalmente a este enfoque clásico se pueden encontrar en la literatura alternativas que fueron diseñadas específicamente para plantas cíclicas, es decir aquellas en las que sea posible identificar la frecuencia ω o el periodo T de la planta o señal de referencia.

3.2.2. Control usando Técnicas de Aprendizaje

En este método [Heinzinger, Fenwick y Miyazaki, 1992] el objetivo de control consiste en seguir una trayectoria y a la vez minimizar alguna función por aprendizaje (que puede ser el error de seguimiento o el par utilizado, incluso una combinación de estas). Para lograrlo se hace una secuencia de intentos en la que el error obtenido en los intentos anteriores se retroalimenta en la señal de control, por lo general en estos intentos se permite la reinicialización de las condiciones iniciales.

La ley de control correspondiente, mostrada en la ECUACIÓN 3.3, se forma por una retroalimentación de estados y un término de aprendizaje.

$$\tau_l(t) = \tau_r(t) + \tau_l(t) \tag{3.3}$$

En la implementación, el término de aprendizaje $\tau_l(t)$ corresponde a la actualización de la señal de control en función de los errores de seguimiento registrados en los ciclos anteriores:

$$\tau_l(t) = \tau_l(t-T) - \begin{bmatrix} k_{l1} & k_{l2} \end{bmatrix} (x(t-T) - x_d(t))$$

En este caso se tomaron las ganancias: $k_{l1} = 0.05$ y $k_{l2} = 1.2$.

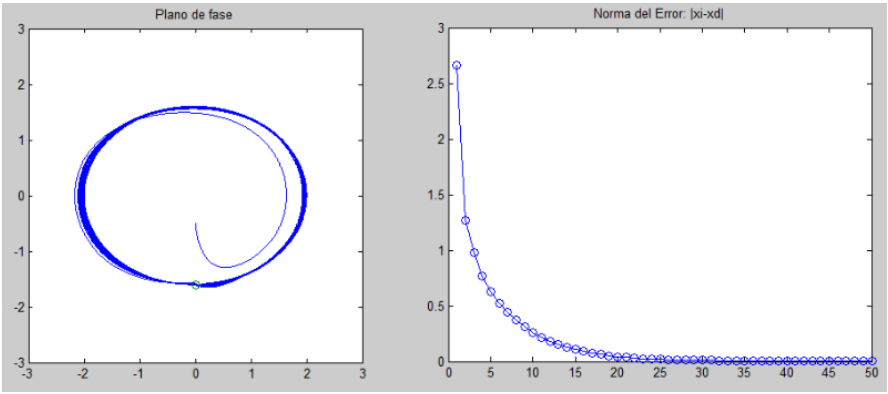


FIGURA 3.3: CONTROL POR APRENDIZAJE

3.2.3. Control Repetitivo

Por otro lado este enfoque [Tohru y Nakano, 1985] propone añadir en el lazo de control un término que corresponde al error de seguimiento retardado, generalmente el del ciclo anterior, esto puede apreciarse en la FIGURA 3.4, de modo que este enfoque puede considerarse dentro de la categoría de los esquemas de control por aprendizaje.

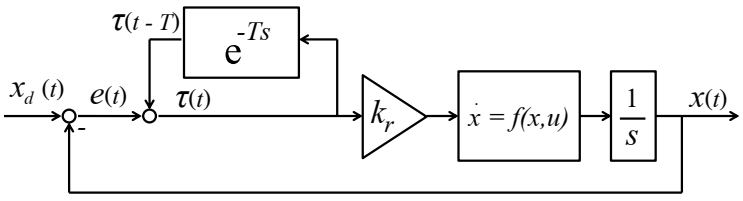


FIGURA 3.4: ESQUEMA DEL CONTROL REPETITIVO

Para este caso la ley de control está dada por:

$$\tau(t) = \begin{bmatrix} k_{r1} & k_{r2} \end{bmatrix} \tau_r(t) \tag{3.4}$$

Donde el termino repetitivo $\tau_r(t)$ está dado por:

$$\tau_r(t) = \tau_r(t - T) - e(t) \tag{3.5}$$

Una desventaja de este control es que la compensación del error en un ciclo, debido a diferencias en condiciones iniciales o perturbaciones, se propaga a los siguientes ciclos, sin embargo también se añade robustez.

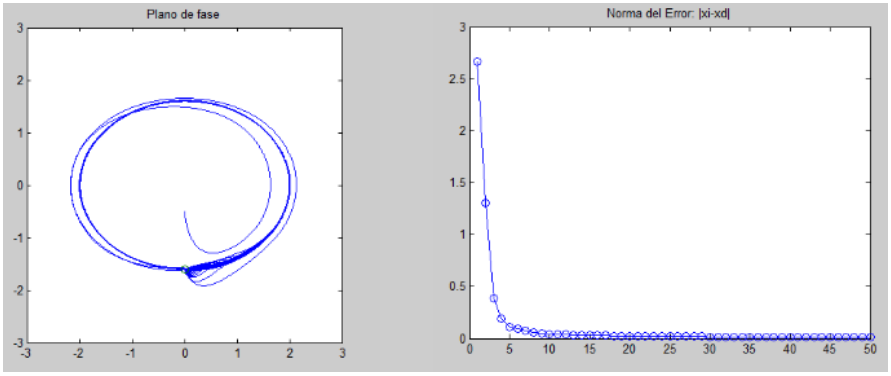


FIGURA 3.5: CONTROL REPETITIVO

3.2.4. Control cíclico

En este tipo de control [Lucibello, 1998] se requiere que el sistema sea operado sobre un ciclo definido sobre una secuencia finita de puntos de equilibrio, en la que las condiciones iniciales y finales de cada ciclo sean las mismas. Cada uno de los puntos usados para definir la trayectoria debe estar ligado a un instante de tiempo. La ley de control se forma por una retroalimentación de estados más una combinación lineal de señales que deben proporcionarse previamente, las constantes usadas en esta combinación son actualizadas al final de cada ciclo en función del error medio de los ciclos anteriores.

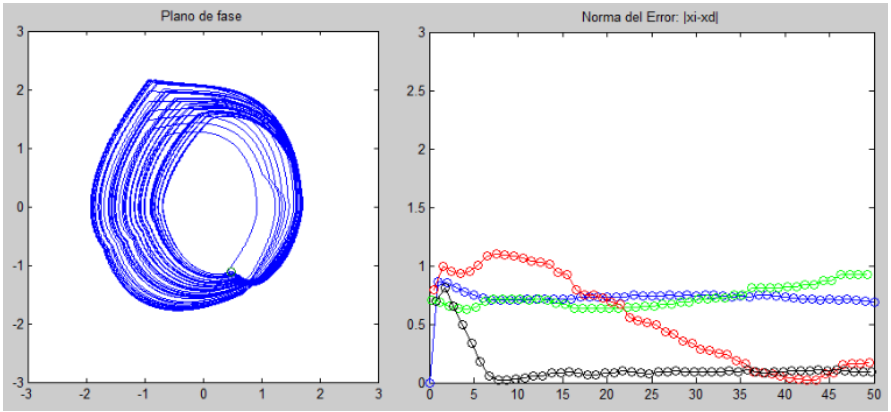


FIGURA 3.6: CONTROL CÍCLICO

Esta estrategia de control fue diseñada para sistemas lineales por lo que su aplicación en caminantes requiere de una aproximación lineal alrededor de cada uno de los puntos usados para definir la trayectoria.

Dada la restricción de utilizar puntos de equilibrio en la definición de la trayectoria a seguir, este esquema no funcionó adecuadamente en el sistema de Van der Pol, ya que para este caso nunca se cumple que $\dot{x} = 0$.

3.2.5. Control mediante Optimización de Rigidez Adaptable

En el contexto de la minimización de la energía se encuentran los trabajos sobre optimización de la rigidez de elementos elásticos [Uemura, Kanaoka y Kawamura, 2007] para el seguimiento de una señal periódica.

Esta estrategia hace uso de las técnicas de control adaptable para variar la rigidez de resortes montados en las articulaciones del sistema con la finalidad de que la frecuencia natural de la planta y de la señal de referencia sean igualadas. De esta forma sólo se requiere que los actuadores induzcan al sistema la energía necesaria para compensar las pérdidas por fricción o calor.

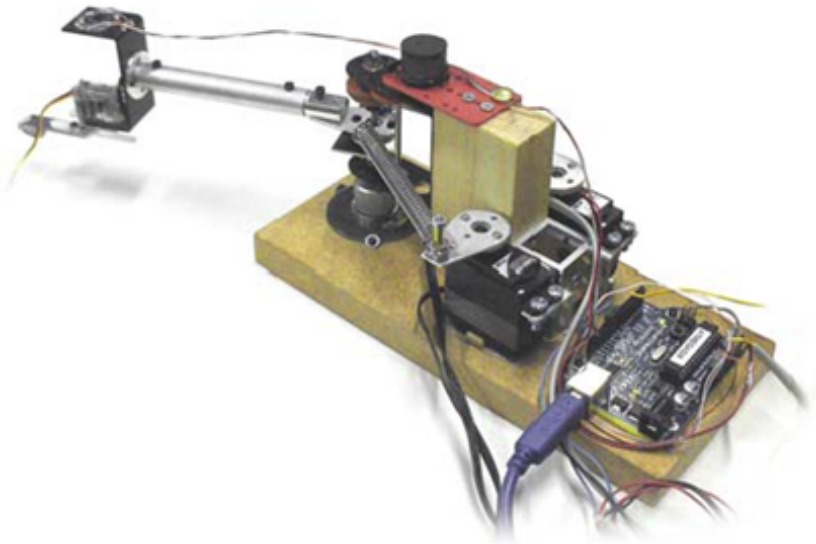


FIGURA 3.7: MECANISMO CON RIGIDEZ AJUSTABLE

El efecto de añadir los resortes de rigidez variable $K(t)$ se ve reflejado como un par externo en la señal de control, la cual queda descrita por la ECUACIÓN 3.6:

$$\tau_s(t) = \tau_r(t) + K(t)\, x_1(t) \tag{3.6}$$

Para encontrar la rigidez óptima, es decir, aquella que produce el menor requerimiento de energía para seguir una trayectoria cíclica, se introduce la dinámica de la rigidez en función del error de seguimiento.

$$\dot{K}(t) = k_a\, x_1(t) \begin{bmatrix} k_{s1} & k_{s2} \end{bmatrix} e(t)$$

En este caso $k_{s1} = 0.6$ y $k_{s2} = 1$ y la constante de adaptación $k_a = 0.1$.

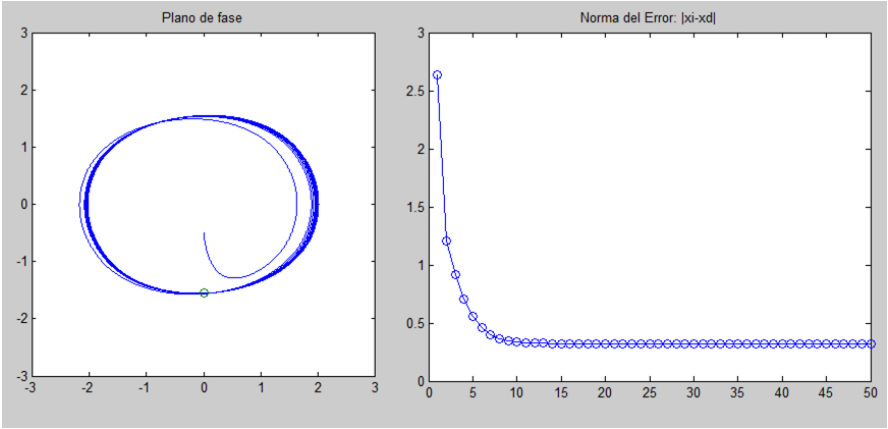


FIGURA 3.8: CONTROL POR OPTIMIZACIÓN DE RIGIDEZ

3.2.6. Control mediante Optimización Temporal y de Rigidez

Una variante del algoritmo anterior consiste en ajustar la frecuencia de la señal de referencia [Nakanishi, Rawlikand y Vijayakumar, 2011], previo al paso de la optimización de la rigidez, para encontrar la frecuencia óptima obteniendo una reducción aún mayor.

La frecuencia óptima ω^* de la señal de referencia será aquella que permita al sistema usar la menor cantidad de energía en la tarea de seguimiento, en un péndulo esto es equivalente a igualar la frecuencia de la señal de referencia con la frecuencia natural del péndulo.

Para obtener ω^* se utiliza la regla de gradiente descendente, mostrada en ECUACIÓN 3.7:

$$\omega_i = \omega_{i-1} - k_g \left(\frac{J_{i-1} - J_{i-2}}{\omega_{i-1} - \omega_{i-2}} \right) \tag{3.7}$$

En este caso $k_g = 0.005$ y J_i es el valor de la función de costo del control en el ciclo i, el cual se forma con tres términos cuadráticos: uno que depende del error de seguimiento, otro que depende del error entre las condiciones iniciales entre dos ciclos consecutivos y el último que depende de la energía aplicada por el controlador.

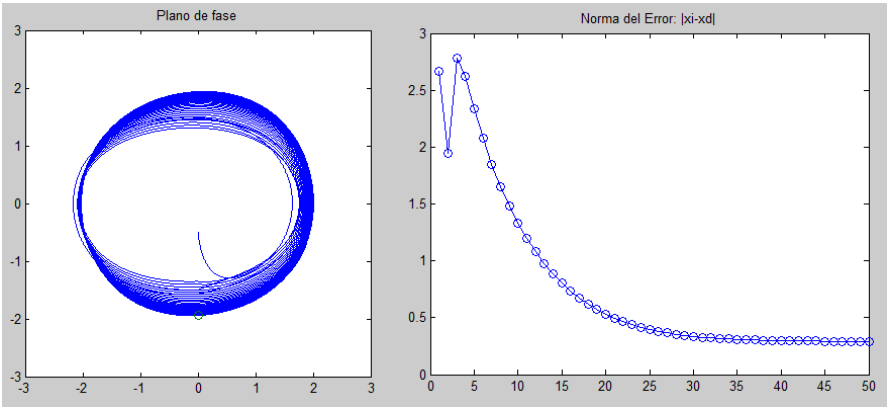


FIGURA 3.9: CONTROL POR OPTIMIZACIÓN TEMPORAL Y RIGIDEZ

En la gráfica podemos observar que el error en las condiciones iniciales produce perturbaciones en la adaptación.

3.2.7. Control Periódico

Una estrategia que incluso ya se ha implementado en modelos de caminantes es el control por estabilización periódica [Ohta y Yamakita, 2001]. Primero hay que estabilizar un ciclo mediante una retroalimentación con el método LQR, después se minimiza la energía usada modificando las condiciones iniciales del ciclo usando una ley de adaptación.

Con la finalidad de conservar algo de la naturaleza pasiva del sistema, el control sólo se aplica dentro una región V definida en función del estado:

$$V = \{x \in X \mid 0 < x_1 < 1,\ x_2 < 0\}$$

En este enfoque no se busca que el sistema siga una trayectoria definida, sólo que al final del ciclo se repitan las condiciones iniciales para mantener un movimiento periódico.

La función de costo a minimizar está definida por:

$$J = \frac{1}{2}\int_0^T \left[e(t)^T\, Q\, e(t) + \tau(t)^T\, R\, \tau(t)\right]\ dt$$

Para la implementación se usó:

$$Q = \begin{bmatrix} 1.1 & 0 \\ 0 & 60 \end{bmatrix}\quad R = \begin{bmatrix} 0.07 & 0 \\ 0 & 1 \end{bmatrix}$$

La aproximación lineal del sistema queda descrita de la forma:

$$\dot{z} = A\, z + B\, v$$

donde: $A = \frac{\partial f}{\partial x}$ y $B = \frac{\partial f}{\partial \tau}$ que al evaluarse se obtiene:

$$A = \begin{bmatrix} 0 & 1 \\ 1 + 2\varepsilon\, x_1 x_2 & -\varepsilon(1 - x_1^2) \end{bmatrix}\quad B = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

La ley de adaptación para este caso se reduce a la ECUACIÓN 3.8:

$$x_{d,i+1} = x_{d,i} - \varepsilon_a \int_0^T \left[-e^{\psi t}\psi^{-1}B - \psi^{-1}A\right] \times R\ e(t)\ dt \tag{3.8}$$

Donde $\psi = A + B$. En este caso se utilizó $\varepsilon_a = 0.05$

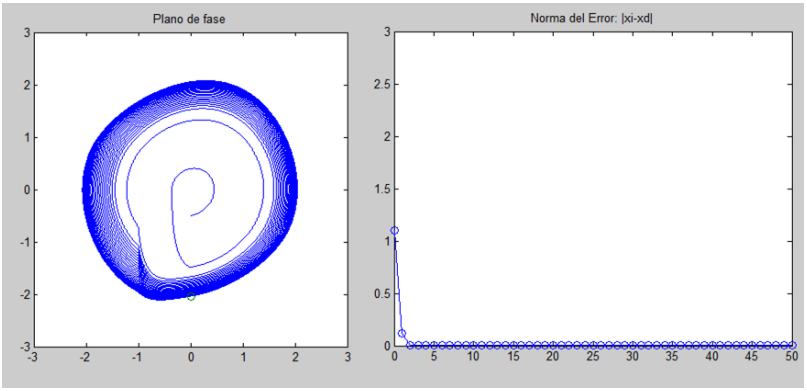


FIGURA 3.10: CONTROL PERIÓDICO

3.3. Comparación de Estrategias de Control

A continuación se muestra una tabla donde se resumen los resultados de la implementación de los algoritmos mencionados:

Algoritmo	Error	Costo	Convergencia
Retro de estados	1.7	2	3
Learning	< 0.05	2.25	10
Repetitive	0.05	2.25	20
Cyclic	$\sim 0.7, \sim 0.2$	> 3	> 50
Stifness	0.3	0.2	10
Stifness w temporal	< 0.05	0.25	30
Periodic	~ 0	< 0.1	20

TABLA 3.1: COMPARACIÓN

Con la información obtenida mediante esta comparación se decidió usar un algoritmo para el control de la caminata con las siguientes características:

- Será un esquema de control de regulación, es decir, se tendrá que proporcionar un conjunto de trayectorias articulares a seguir y se corregirá el error de seguimiento haciendo uso de una retroalimentación de estados.

- Las ganancias del controlador deben sintonizarse de tal forma que se minimice la función objetivo propuesta, se utilizará como ganancia inicial la obtenida mediante el método LQR a partir de una aproximación lineal del sistema (como en el caso reportado en [Ohta y Yamakita, 2001]).

- Las señales de referencia deben ajustarse de tal forma que contribuyan a minimizar la función objetivo, inicialmente se tomarán como referencia las trayectorias del caminante pasivo (como en [Sukuzi y col., 2001]) y después se ajustará la frecuencia y la amplitud de estas señales (como en [Nakanishi, Rawlikand y Vijayakumar, 2011]).

- El algoritmo debe evitar la necesidad de reinicializar el sistema en cada iteración, por lo que las trayectorias deben modificarse cuidando de conservar la continuidad.

Bibliografía del Capítulo 3

Anderson, S.O. y col. (2005). «Powered Bipeds Based on Passive Dynamic Principles». En: *IEEE-RAS International Conference on Humanoid Robots*, págs. 110-116.

Collins, S. y A. Ruina (2005). «A Bipedal Walking Robot with Efficient and Human-Like Gait». En: *IEEE International Conference on Robotics and Automation*, págs. 1983-1988.

Heinzinger, G., D. Fenwick y B.P. Miyazaki (1992). «Stability of learning Control with Disturbances and Uncertain Initial Conditions». En: *IEEE Transactions on Automatic Control* 37, págs. 110-114.

Kirk, D.E. *Optimal Control Theory: An Introduction*. Dover Publications.

Lucibello, P. (1998). «Cyclic control of linear systems with aplication to a flexible arm». En: *IEEE Proceedings - Control Theory and Applications*, págs. 19-24.

Nakanishi, J., K. Rawlikand y S. Vijayakumar (2011). «Stiffness and Temporal Optimization in Periodic Movements: An Optimal Control Approach». En: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, págs. 718-724.

Ohta, H. y M. Yamakita (2001). «Periodic Stabilizing of Control Systems with Collisions - Aplication to Walking Robots». En: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, págs. 980-985.

Sukuzi, S. y col. (2001). «Biped Walking Robot Control with Passive Walker Model by new VSC servo». En: *Proceedings of the American Control Conference*, págs. 107-112.

Tedrake, R., T.W. Zhang y H.S. Seung (2005). «Learning to Walk in 20 Minutes». En: *Proceedings of the Fourteenth Yale Workshop on Adaptive and Learning Systems*.

Tohru, S.H. y M. Nakano (1985). «Synthesis of Repetitive Control systems and its applications». En: *IEEE International Conference on Decision and Control*, págs. 1387-1392.

Uemura, M., K. Kanaoka y S. Kawamura (2007). «A new Control Method Utilizing Stiffness Adjustment of Mechanical Elastic Elements for serial link systems». En: *IEEE International Conference on Robotics and Automation*, págs. 1437-1442.

Vukobratovic, M. y B. Borovac (2004). «Zero-Moment Point thirty five years of its Life». En: *International Journal of Humanoid Robotics* 1, págs. 157-173.

Capítulo 4

Diseño de Trayectorias de Caminado

En esta sección se presenta el procedimiento usado en la generación de patrones de caminado parametrizado implementado en *Johnny* un robot humanoide cuyo diseño fue concebido para minimizar el consumo energético durante el caminado.

La generación de patrones de caminado bípedo es uno de los aspectos más importantes en el área de la robótica humanoide. A lo largo de los años han surgido muchas propuestas basadas en modelos reducidos como el Péndulo Lineal Invertido [Kajita y col., 2014] o los Puntos de Captura [Pratt y col., 2006]. Los robots humanoides cuyos caminados siguen esta corriente a menudo exhiben movimientos considerados como poco naturales e ineficientes, aunque sumamente estables. Recientemente, el concepto de Estabilidad del Ciclo Límite permite al diseñador relajar las restricciones en las posturas instantáneas que puede alcanzar el humanoide para expandir la variedad de patrones de caminado de acuerdo al objetivo planteado.

Los caminantes pasivos [Collins y col., 2005] son un excelente ejemplo de la aplicación de este concepto de estabilidad, en el cual la generación de patrones de caminado se reduce a determinar el conjunto de condiciones iniciales que producen trayectorias cíclicas robustas ante perturbaciones.

A diferencia de los conceptos de estabilidad basados en ZMP (*Zero Moment Point*) [Vukobratovic y Borovac, 2004], en los que se evalúa cada punto de la trayectoria de caminado, cuando se utilizan los conceptos de ciclo límite sólo se evalúa la diferencia entre las condiciones iniciales de ciclos consecutivos para determinar la estabilidad del ciclo completo. Por lo tanto este es un esquema de generación de trayectorias fuera de línea, en el cual es necesario contar con un simulador dinámico realista que permite predecir el comportamiento del robot ante distintas trayectorias cíclicas.

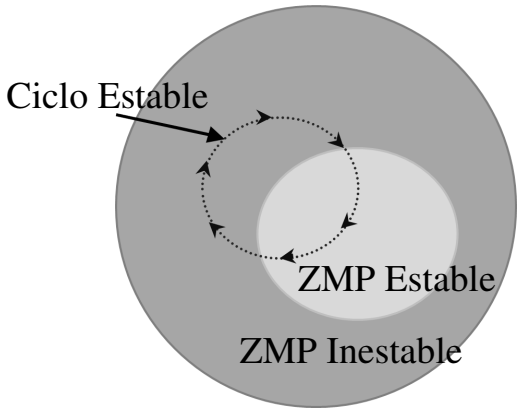


FIGURA 4.1: CRITERIOS DE ESTABILIDAD

Asi, el simulador mecánico debe basarse en un modelo dinámico muy completo: capaz de reproducir el comportamiento de los actuadores, la influencia de la distribución de las masas en el robot, así como el efecto de diversos fenómenos como el impacto de los pies con el suelo o las restricciones mecánicas de sus eslabones. De este modo, el simulador dinámico permitirá determinar si el robot es capaz de seguir las trayectorias definidas.

El caminado parametrizado propuesto, por lo tanto, es puramente cinemático y se define de manera que las condiciones articulares iniciales se repitan al inicio de cada paso. Entonces la estabilidad del caminado depende de los parámetros de caminado así como de los parámetros mecánicos del robot en cuestión y las ganancias de los controladores locales.

En el caso de *Johnny* [Núñez e Ibarra, 2014], el robot humanoide desarrollado en nuestro laboratorio [Núñez e Ibarra, 2013; Núñez e Ibarra, 2011] , el objetivo no es solo generar patrones de caminado estables sino, además, encontrar el conjunto de parámetros mecánicos óptimos para la construcción del robot. Para lograr este objetivo se utilizaron algoritmos genéticos, método de optimización heurística basada en la evolución. Con la finalidad de proveer al algoritmo genético de un espacio de búsqueda representativo, se desarrolló el caminado parametrizado que se presenta en las secciones siguientes. Inicialmente se definen las trayectorias de caminado en el espacio de trabajo y después se hace uso de la cinemática inversa para encontrar las trayectorias articulares.

4.1. Optimización del Modelo del Péndulo Invertido

En el diseño mecánico del robot humanoide *Johnny* se busca aprovechar las características de los caminantes pasivos para minimizar el consumo energético. De la misma manera, las trayectorias de la cadera, los pies y las manos buscan emular el comportamiento pasivo de estos mecanismos. Por lo tanto, como primer paso, se calculan las trayectorias de la cadera de un caminante pasivo simplificado, es decir un péndulo invertido simple sin actuación.

Usando la notación de la FIGURA 4.2, el modelo de este péndulo en el plano XZ es:

$$mr^2\ddot{\theta} - mgr\sin\theta = \tau$$

El modelo linealizado con $\tau = 0$ para ángulos pequeños $\sin\theta = \theta$ y tomando $\omega^2 = \frac{g}{r}$ es:

$$\ddot{\theta} - \omega^2\theta = 0 \tag{4.1}$$

La solución de esta ecuación para la posición y la velocidad es:

$$\theta = a_1 e^{\omega\, t} + a_2 e^{-\omega\, t} \tag{4.2a}$$

$$\dot{\theta} = a_1\omega e^{\omega\, t} - a_2\omega e^{-\omega\, t} \tag{4.2b}$$

Multiplicando ambos lados de la ECUACIÓN 4.1 por $\dot{\theta}$ e integrando con respecto al tiempo se obtiene la ecuación de la energía orbital:

$$\frac{1}{2}\dot{\theta}^2 - \frac{\omega^2}{2}\theta^2 = E$$

Minimizando la energía orbital, se encuentra la siguiente relación:

$$\dot{\theta} = -\omega\,\theta$$

La energía orbital es constante durante el movimiento del péndulo por lo que hacer $E = 0$ implica que, cuando el péndulo pase por la vertical, no tendrá suficiente energía para continuar el movimiento. Agregar un δ de energía permite pasar de la vertical y completar la trayectoria del paso en función del tiempo requerido, por lo que se puede escribir una relación entre las condiciones iniciales de la siguiente manera:

$$\dot{\theta}_0 = -\omega\,\theta_0 + \delta \tag{4.3}$$

Para conseguir un movimiento cíclico, las condiciones iniciales deben repetirse en cada paso. Las posiciones angulares iniciales y finales deben tener la misma magnitud y sentido contrario, la velocidad final debe ser ligeramente mayor que la inicial porque el impacto que se produce al cambiar de punto de apoyo genera un decremento instantáneo de la velocidad angular, lo que nos lleva a las siguientes restricciones:

$$\begin{aligned} \theta(0) &= -\theta_0 \qquad & \dot{\theta}(0) &= \dot{\theta}_0 \\ \theta(t_z) &= \theta_0 \qquad & \dot{\theta}(t_z) &= \frac{1}{\eta}\dot{\theta}_0 \end{aligned} \tag{4.4}$$

Donde t_z es la duración de la zancada y $0 \leq \eta < 1$ es una constante que describe la energía perdida durante la colisión.

Para definir por completo la trayectoria del péndulo linearizado, descrita por la ECUACIÓN 4.2 es necesario encontrar los parámetros a_1, a_2 y δ usando la frecuencia natural ω del péndulo y las restricciones dadas por la ECUACIÓN 4.3 y la ECUACIÓN 4.4.

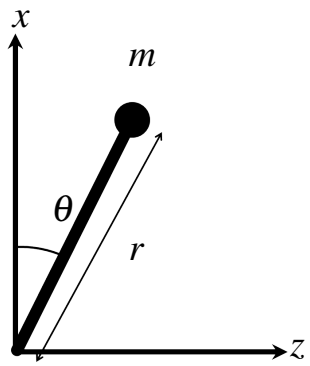


FIGURA 4.2: PÉNDULO INVERTIDO SIN ACTUACIÓN

Evaluando la ECUACIÓN 4.2 en $t = 0$ y $t = t_z$ se tiene el siguiente sistema de ecuaciones:

$$-\theta_0 = a_1 + a_2 \tag{4.5a}$$

$$\omega\theta_0 + \delta = a_1\omega - a_2\omega \tag{4.5b}$$

$$\theta_0 = a_1 e^{\omega\, t_z} + a_2 e^{-\omega\, t_z} \tag{4.5c}$$

$$\frac{1}{\eta}(\omega\theta_0 + \delta) = a_1\omega e^{\omega\, t_z} - a_2\omega e^{-\omega\, t_z} \tag{4.5d}$$

De la ECUACIÓN 4.5.a y la ECUACIÓN 4.5.b se puede encontrar que:

$$a_1 = \frac{\delta}{2\omega}, \qquad a_2 = -\frac{\delta}{2\omega} - \theta_0$$

Y de la ECUACIÓN 4.5.c y la ECUACIÓN 4.5.d se puede encontrar que:

$$\delta = \frac{\omega\theta_0(\eta+1)}{\eta e^{\omega t_z} - 1}$$

Sustituyendo estos resultados se tiene que la trayectoria que lleva al péndulo de $-\theta_0$ a θ_0 en un tiempo de t_z, esta dada por:

$$\theta(t) = \frac{\theta_0(\eta+1)}{2\eta e^{\omega t_z} - 2} e^{\omega t} - \left(\frac{\theta_0(\eta+1)}{2\eta e^{\omega t_z} - 2} - \theta_0 \right) e^{-\omega t}$$

El ángulo inicial θ_0 se define en términos de la extensión de la pierna P_l y el tamaño de la zancada L_z, como:

$$\theta_0 = \arcsin\left(\frac{L_z}{2P_l}\right)$$

La trayectoria de la cadera $\vec{C}$ en coordenadas espaciales, de acuerdo al eje de coordenadas de la FIGURA 4.4, está dada entonces por:

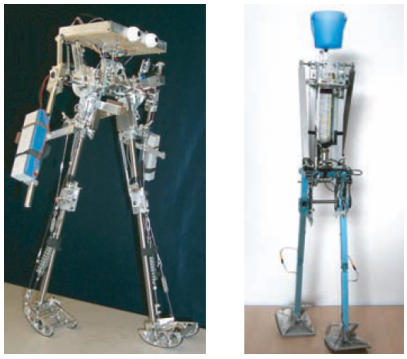
$$\begin{aligned} C_x(t) = &\quad P_l \cos(\theta(t)) \\ C_z(t) = &\quad P_l \sin(\theta(t)) \end{aligned}$$

4.2. Diseño de Trayectorias para la Unidad Pasajera

En el diseño de los primeros robots humanoides basados en caminantes pasivos se buscaba enfatizar el bajo consumo energético que estas máquinas eran capaces de alcanzar, por lo que a menudo contaban con pocos actuadores, con el fin de aprovechar al máximo las características de sus contrapartes pasivas. Así el movimiento de los brazos en la unidad pasajera estaba acoplado mecánicamente al movimiento de las piernas.

En el caso del robot construido por la Universidad de Cornell [Collins y col., 2005], mostrado en la FIGURA 4.3, cada pierna forma un solo eslabón con el brazo opuesto, es decir no hay movimiento relativo, y el torso se mantiene siempre perpendicular al piso.

Otro ejemplo es el robot *Denise* de la Universidad de Delft [Collins y col., 2005], mostrado en la FIGURA 4.3, el cual cuenta con transmisiones de cadenas que acoplan el movimiento de cada pierna al brazo opuesto y también mantienen el torso perpendicular al piso.



Cornell Delft

FIGURA 4.3: ROBOTS BASADOS EN CAMINANTES PASIVOS

En el caso de *Johnny*, el consumo energético es un factor importante pero también lo es la capacidad del robot para realizar diferentes aplicaciones, por lo que se decidió usar actuadores independientes en cada articulación y sincronizar las trayectorias correspondientes durante el caminado. Ahora que se ha definido la trayectoria de la cadera, se procede a definir la trayectoria de las extremidades faltantes: el torso, los pies y las manos, acoplando los movimientos correspondientes.

En la FIGURA 4.4 se pueden observar las trayectorias de los eslabones del humanoide durante la etapa de soporte simple. En el origen se encuentra el tobillo de la pierna apoyada en el piso, la cual se indica con una línea doble, al igual que el correspondiente brazo acoplado, a los cuales nos referiremos como *estáticos*, debido a que durante esta etapa se encuentran siempre sobre el origen. Por otro lado la pierna que se levanta y se mueve en la dirección del caminado se muestran con líneas sólidas, al igual que el correspondiente brazo acoplado, posteriormente nos referiremos a estas extremidades como *oscilantes*. Como en los ejemplos previos, inicialmente la orientación del torso se considera perpendicular al piso. Con respecto a la posición del pie oscilante, las trayectorias en el plano XZ están dadas por:

$$P_x(t) = \frac{P_x^{max}}{H_x^{max} - H_x^{min}}(H_x(t) - H_x^{min})$$
$$P_z(t) = 2H_z(t)$$

Como se puede observar, la trayectoria del pie oscilante es una transformación de la trayectoria de la cadera, que tiene como parámetro la altura máxima P_x^{max} del pie, la cual ocurre a la mitad del paso. $H_{x,z}^{min}$ y $H_{x,z}^{max}$ son el valor mínimo y máximo que alcanza la cadera en las componentes x y z respectivamente y que sirven para escalar el movimiento del pie oscilante.

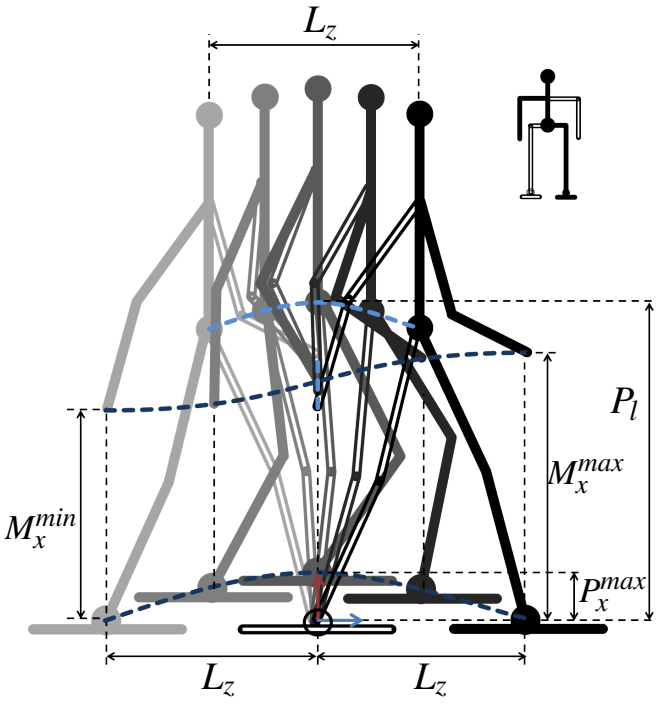


FIGURA 4.4: PARÁMETROS DE CAMINADO

En el caso de la mano oscilante M^o, la componente en el eje z es la misma que la de la pierna oscilante. En el eje y, se define una trayectoria para ir desde una altura mínima M_x^{min} a una máxima M_x^{max}.

$$M_x^o(t) = Pol_3(M_x^{min}, 0, M_x^{max}, 0, t, t_z)$$
$$M_z^o(t) = P_z(t)$$

En donde $x = Pol_3(x_0, \dot{x}_0, x_\tau, \dot{x}_\tau, t, \tau)$ se usa para denotar a la trayectoria generada con un polinomio de tercer grado con condiciones iniciales $(x_0, \dot{x}_0)$ y condiciones finales $(x_\tau, \dot{x}_\tau)$ para el rango de tiempo $0 \leq t \leq \tau$. Por su parte la componente de la mano estática M^e, en el eje z es nula ya que el pie estático siempre está apoyado en el origen. En el eje y la mano se mueve desde M_x^{max} a M_x^{min} para completar el ciclo:

$$M_x^e(t) = Pol_3(M_x^{max}, 0, M_x^{min}, 0, t, t_z)$$
$$M_z^e(t) = 0$$

Una vez definidas las trayectorias de los eslabones en el espacio, se procede a calcular la cinemática inversa de cada extremidad para encontrar las referencias articulares de cada actuador.

4.3. Trayectorias Articulares

Las extremidades, en el caso de *Johnny*, fueron diseñadas para que la cinemática inversa fuera especialmente fácil de calcular, en la FIGURA 2.3 se puede observar que cada extremidad es un manipulador de 2 gdl para el caso de los brazos y 3 gdl para el caso de las piernas, en el plano lateral XZ.

Como se puede observar en la FIGURA 4.5, cada extremidad forma un triángulo, cada uno de los lados es de longitud conocida, pues dos de sus lados están formados por eslabones del robot y el tercer lado está definido por la posición del efector final (cadera, pie oscilante o manos).

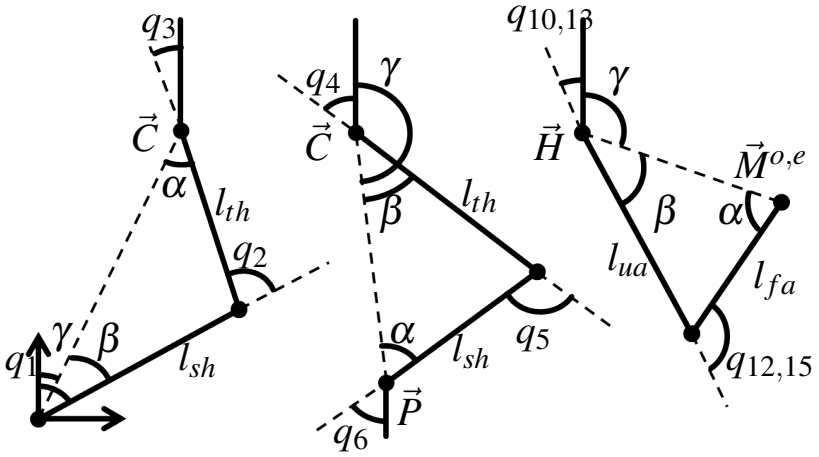


FIGURA 4.5: CINEMÁTICA INVERSA POR EXTREMIDADES

Los desplazamientos angulares de cada articulación se pueden calcular en función de los ángulos internos α y β, usando el teorema del coseno, el ángulo γ está en función de las coordenadas del efector final:

$$\alpha(t) = \arccos\left(\frac{B^2 + C(t)^2 - A^2}{2BC(t)}\right)$$
$$\beta(t) = \arccos\left(\frac{A^2 + C(t)^2 - B^2}{2AC(t)}\right)$$
$$\gamma(t) = \text{atan2}(C(t)_z, C(t)_x)$$

Donde A, B y C, para cada extremidad, tienen los valores dados en la TABLA 4.1, donde los vectores $\vec{C}(t)$, $\vec{P}(t)$, $\vec{M}^o(t)$ y $\vec{M}^e(t)$ representan los vectores de posición de la cadera, pie oscilante, mano oscilante y mano estática, definidos en la sección anterior.

Extremidad	**A**	**B**	**C(t)**
Cadera	l_{th}	l_{th}	$\lvert\vec{C}(t)\rvert$
Pierna Oscilante	l_{th}	l_{sh}	$\lvert\vec{P}(t) - \vec{C}(t)\rvert$
Brazo Oscilante	l_{ua}	l_{fa}	$\lvert\vec{M}^o(t) - \vec{H}(t)\rvert$
Brazo Estático	l_{ua}	l_{fa}	$\lvert\vec{M}^e(t) - \vec{H}(t)\rvert$

TABLA 4.1: PARÁMETROS DE LA CINEMÁTICA INVERSA

El vector $\vec{H}(t)$ representa la posición del hombro en el plano XZ, el cual es igual a la posición de la cadera mas la longitud del torso, es decir $\vec{H}(t) = \vec{C}(t) + l_{sd}\hat{i}$. Conociendo los valores de $\alpha(t)$, $\beta(t)$ y $\gamma(t)$ en cada uno de los triángulos, podemos ver a partir de la FIGURA 4.5 que los desplazamientos angulares están dados por:

$$q_1(t) = \gamma(t) + \beta(t) \qquad q_4(t) = \gamma(t) - \beta(t) - \pi$$
$$q_2(t) = -\alpha(t) - \beta(t) \qquad q_5(t) = \alpha(t) + \beta(t)$$
$$q_3(t) = \alpha(t) - \gamma(t) \qquad q_6(t) = -\alpha(t) - \gamma(t) + \pi$$

$$q_{10}(t) = \gamma(t) + \beta(t) - \pi \quad q_{13}(t) = \gamma(t) + \beta(t) - \pi$$
$$q_{12}(t) = -\alpha(t) - \beta(t) \qquad q_{15}(t) = -\alpha(t) - \beta(t)$$

Por simplicidad, hasta este punto las trayectorias en el espacio de trabajo se definieron únicamente en el plano XZ, ya que las trayectorias de las articulaciones que generan movimiento fuera de este plano se definen directamente en el espacio articular.

Ya que *Johnny* es un caminante 3D para generar el caminado es necesario introducir un balanceo lateral en los tobillos:

$$q_0(t) = q_7(t) = q_0^{max} \sin\left(\pi\frac{t}{t_z}\right)$$

Donde q_0^{max} es la inclinación lateral máxima del tobillo.

En el caso de *Johnny*, la orientación del torso se puede manipular por medio de las articulaciones q_8 y q_9 de manera independiente al movimiento de las piernas. El movimiento frontal q_8 es especialmente útil para contrarrestar los pares producidos por el movimiento de la pierna oscilante en la cadera. Por otro lado, el movimiento lateral q_9 sirve para disminuir

el par que se necesita producir en los tobillos para hacer oscilar el caminante lateralmente. Estos movimientos se generan por medio de funciones sinusoidales, de la siguiente forma:

$$q_8(t) = q_8^{max} \sin(2\pi \frac{t}{t_z}), \qquad q_9(t) = q_9^{max} \sin(\pi \frac{t}{t_z})$$

El movimiento en el eje y del brazo estático $M_y^e(t)$ solo se usa para mantener el brazo en la dirección de la gravedad y evitar generar pares innecesarios en las articulaciones q_{10} y q_{13}, sin embargo no es posible hacer lo mismo para el brazo oscilante ya que este chocaría con el torso.

$$q_{10}(t) = (q_9^{max} + q_0^{max}) \sin\left(\pi \frac{t}{t_z}\right), \qquad q_{13}(t) = 0$$

Con respecto a las trayectorias de la cabeza, no se considera que su movimiento afecte el consumo energético, de manera significativa, por lo que se mantiene estática. De esta forma, la estrategia de caminado propuesta tiene los siguientes parámetros: P_t, L_z, η, P_x^{max}, M_x^{max}, M_x^{min}, q_0^{max}, q_8^{max}, q_9^{max}.

4.4. Diseño de Trayectorias para la Orientación de los Pies

Con respecto al caminado omnidireccional, en el caso especifico de Johnny, cada pierna cuenta con un grado de libertad (q_{16} y q_{17}) que permite orientar directamente el pie oscilante sobre el plano YZ, por lo que el movimiento de las piernas anteriormente mencionado sigue siendo válido.

Para cambiar la dirección de caminado solo es necesario modificar el ángulo del pie oscilante y tener cuidado de restablecer a cero dicho ángulo en la pierna apoyada en el piso. De esta forma se obtienen patrones de caminado como el mostrado en laFIGURA 4.6.

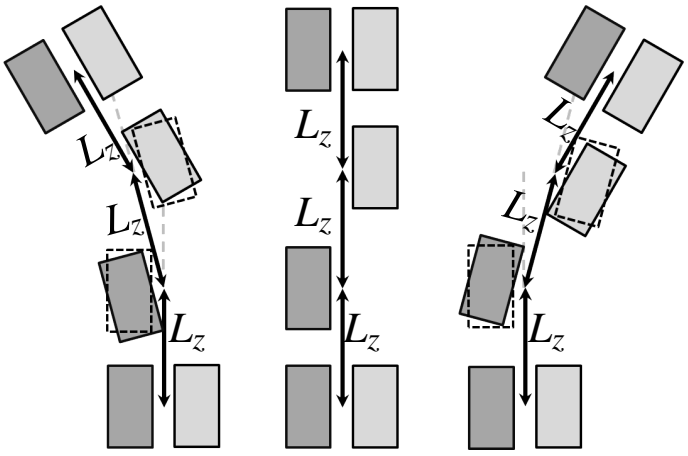


FIGURA 4.6: CAMINADO OMNIDIRECCIONAL

Resultados

El caminado parametrizado desarrollado es capaz de reproducir el movimiento de caminantes pasivos de manera satisfactoria, aprovechando la energía generada de manera eficiente. Los parámetros del caminante *Johnny* fueron encontrados gracias a un proceso de optimización basado en algoritmos genéticos.

El caminado omnidireccional permite al robot caminar con diferente radio de giro siguiendo consignas en línea.

Bibliografía del Capítulo 4

Collins, S. y col. (2005). «Efficient Bipedal Robots Based on Passive Dynamic Walkers». En: *Science Magazine* 307, págs. 1082-1085.

Kajita, S. y col. (2014). *Introduction to Humanoid Robotics*. Springer.

Núñez, R.S. y J.M. Ibarra (2011). «Desarrollo de Bípedos Pasivos». En: *Memorias del CoMRob 2011, XIII Congreso Mexicano de Robótica de la AMRob*, págs. 69-74.

— (2013). «Diseño de estrategias de control para caminantes basados en bípedos pasivos». En: *Memorias del CoMRob 2013, XV Congreso Mexicano de Robótica de la AMRob*.

— (2014). «Johnny: Desarrollo de un Robot Humanoide Óptimo basado en caminantes dinámicos pasivos». En: *Memorias del XVI Congreso Mexicano de Robótica*.

Pratt, J. y col. (2006). «Capture Point: A Step toward Humanoid Push Recovery». En: *Proceedings of the 2006 6th IEEE-RAS International Conference on Humanoid Robots*, págs. 200-207.

Vukobratovic, M. y B. Borovac (2004). «Zero-Moment Point thirty five years of its Life». En: *International Journal of Humanoid Robotics* 1, págs. 157-173.

Capítulo 5

Diseño óptimo

En los capítulos anteriores se estudió el modelo matemático de la caminata, se compararon distintas leyes de control y se definieron trayectorias de caminado, basados en este análisis se determinó que, en el caso mas general, el consumo energético se puede describir de la siguiente forma:

$$E = f(G_p, C_p, M_p, A_p) \tag{5.1}$$

En donde G_p representa el vector de parámetros que definen las trayectorias de caminado, C_p es el vector de parámetros que definen el control de las articulaciones, M_p representa el vector de parámetros mecánicos que definen al caminante y A_p representa el vector de parámetros inherentes a la arquitectura empleada y que no podemos manipular como el coeficiente de fricción de las articulaciones, la eficiencia de los motores, la energía consumida por el sistema de cómputo, etc.

Para simplificar la notación se utiliza el vector $S_p = [G_p\, C_p\, M_p]$ para nombrar el conjunto de todos los parámetros sintonizables, por lo que la dependencia de la energía de este vector se escribirá como $E(S_p)$.

Para evaluar el consumo energético descrito por la ECUACIÓN 5.1 se sustituyen los parámetros de vector S_p y se integra el modelo matemático descrito en el capítulo 2, de las señales de par obtenidas para cada articulación se hace la conversión a consumo energético considerando la eficiencia de los actuadores usados.

Utilizar como objetivo de optimización la ECUACIÓN 5.1 no produce un caminado óptimo, lo que se obtiene es la solución trivial en la cual no hay consumo de energía: el robot *estático*. Un criterio de optimización más adecuado, en el sentido del consumo energético, consiste en utilizar el Costo Específico de Transportación, C_{tr} el cual se define por la expresión siguiente:

$$C_{tr}(S_p) = \frac{E(S_p)}{m_r(S_p)\; g\; d_{tr}(S_p)} \tag{5.2}$$

Donde m_r es la masa del robot, g es el valor de la aceleración de la gravedad y d_{tr} es la distancia recorrida en el trayecto. De esta forma el problema de optimización abordado en este trabajo se puede plantear de la siguiente manera:

$$\begin{gathered} \min_{S_p} C_{tr}(S_p) \\ \text{sujeto a } |r_1(x(S_p))| \leq \delta \text{ y } |r_2(x(S_p))| < 1 \end{gathered} \tag{5.3}$$

En donde δ es la diferencia máxima permitida entre las condiciones iniciales de pasos consecutivos, r_1 y r_2 son las restricciones de estabilidad dadas por la ECUACIÓN 2.6 y la ECUACIÓN 2.10.

Para resolver el problema de encontrar el mínimo de una función real diferenciable se cuenta con una amplia variedad de métodos diferenciales de optimización. Sin embargo, cuando no se dispone de la derivada de la función cuyo mínimo se busca, se recurre a métodos de búsqueda directos [Kolda, Lewis y Torczon, 2003] o a métodos de la Computación Evolutiva [*Introduction to Evolutionary Computing*].

Antes de abordar el problema mas general el cual es minimizar la ECUACIÓN 5.2 con respecto a todos los parámetros en S_p para el modelo de un robot humanoide, se decidió empezar con un problema simplificado: La optimización de un caminante pasivo con actuadores, en base a la experiencia obtenida en una segunda etapa se abordó el problema completo, a continuación se muestran los detalles de ambos procesos de optimización:

- **Primer Enfoque:** Inicialmente se consideró el modelo matemático del tercer prototipo de caminante pasivo construido durante la tesis de maestría [Núñez, 2012], al cual se le agregaron actuadores. En el vector G_p se consideraron la amplitud y frecuencia de sus trayectorias y en C_p las ganancias de un controlador PID, se consideró que no era necesario optimizar la arquitectura mecánica debido a la eficiencia inherente de los caminantes pasivos. En esta primer tentativa de solución el proceso de optimización se realizó mediante el uso del método *Simplex* de [Nelder y Meald, 1965].
- **Segundo Enfoque:** Considerando que durante la conversión física de un caminante pasivo a uno actuado la eficiencia energética se puede degradar (por las modificaciones a los parámetros mećanicos que involucra el agregar actuadores, sensores y equipo de procesamiento), en esta ocasión se decidió usar un enfoque mas general y considerar el modelo del humanoide completo, en el vector G_p se incluyeron los parámetros necesarios para generar las trayectorias de todas las extremidades y en M_p se incluyeron la longitud y masa de todos los eslabones. Se decidió abordar el problema utilizando el algoritmo genético NSGA-II [Deb y col., 2002] como método de optimización global con lo que se obtuvieron los resultados que nos permitieron construir el prototipo *Johhny*.

5.1. Primer Enfoque

En este problema simplificado se utilizó el modelo matemático de un caminante pasivo, el cual posee tres grados de libertad, al que se le agregaron actuadores, por lo que el problema se restringió bajo las siguientes consideraciones:

- **Parámetros de Caminado** G_p: En este vector incluimos A y F que son ganancias para modificar la amplitud y la frecuencia, respectivamente, de las trayectorias articulares del caminante pasivo, como condiciones iniciales se tiene que $A_0 = 1$ y $F_0 = 1$ lo que produce las trayectorias originales.
- **Parámetros del Controlador** C_p: En este vector se incluyen las ganancias de retroalimentación P_i y D_i de la articulación i de un controlador PD, las ganancias iniciales se obtienen por el método LQR usando una aproximación lineal en algún punto intermedio de la trayectoria del caminante pasivo.
- **Parámetros Mecánicos** M_p: Aquí se considera que los parámetros mecánicos ya son lo suficientemente cercanos a los valores óptimos (debido al diseño de los caminantes pasivos) por lo que no se requiere de su sintonización.

Con estas restricciones el vector de parámetros sintonizables se define como:

$$S_p = [P_1, P_2, P_3, D_1, D_2, D_3, A, F] \tag{5.4}$$

5.1.1. Algoritmo de Optimización

Los métodos de búsqueda directa no son bien aceptados por la comunidad de optimización matemática básicamente por tres razones: estos métodos fueron desarrollados de forma heurística, no se han desarrollado pruebas de convergencia para ellos y algunas veces la velocidad de convergencia puede ser muy lenta.

No obstante estos métodos han demostrado ser bastante eficientes para aplicaciones en las comunidades de ingeniería y ciencia aplicada particularmente en los campos de la química, la ingeniería mecánica y en la medicina.

Específicamente se hará uso del método conocido como Algoritmo Simplex de Nelder-Mead. Una iteración de este método puede describirse de la forma mostrada por el ALGORITMO 1 del segundo apéndice según la referencia [Lagarias y col., 1998].

El algoritmo de [Nelder y Meald, 1965] fue propuesto como un método para minimizar una función escalar de variable real $f(x)$ para $x \in \mathcal{R}^n$, usando únicamente evaluaciones de la función a minimizar para formar un simplex.

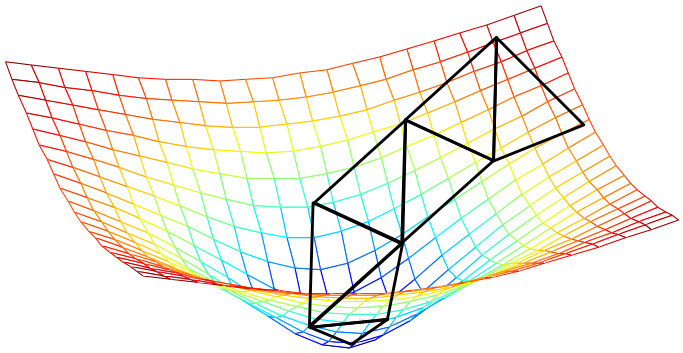


FIGURA 5.1: MÉTODO SIMPLEX

Un simplex es una figura geométrica cuya cantidad de vértices es igual a uno más que la dimensión del vector de estados del sistema. El simplex representa una aproximación tangencial a la superficie que representa la salida del sistema. En el caso de un sistema de dimensión dos un simplex sería equivalente a un triángulo.

Este método de búsqueda forma parte de los algoritmos de "operación evolutiva" o EVOP por sus siglas en inglés y consiste en desplazar el simplex inicial mediante cuatro operaciones que dependen de cuatro coeficientes: reflexión (ρ), expansión (χ), contracción (γ) y encogimiento (σ).

La FIGURA 5.2 muestra el efecto de las operaciones de ***Reflejar*** *i),* **Expandir** ii), ***Contraer*** *iii) y* **Encoger** iv), el simplex original se muestra en líneas punteadas. Estas operaciones permiten expandir el simplex en las direcciones favorables y comprimirlo en las direcciones desfavorables.

En cada iteración de este algoritmo el vértice que produzca la evaluación con mayor magnitud es reflejado y escalado de acuerdo a las operaciones antes mencionadas. En el caso

de que el nuevo vértice produzca una evaluación mayor que todos los vértices existentes se procede a hacer un encogimiento del simplex.

Las modificaciones al simplex inicial producen nuevos simplex cuya posición tiende al mínimo de la función y cuya forma se adapta a la superficie local de la función.

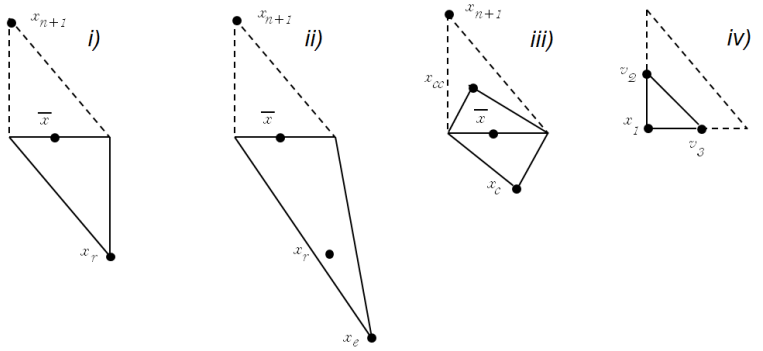


FIGURA 5.2: OPERACIONES BÁSICAS DEL MÉTODO SIMPLEX

5.1.2. Implementación

En el caso de nuestra aplicación los simplex están formados por los 8 parámetros de la ECUACIÓN 5.4; las ganancias PD del simplex inicial se tomaron por medio del algoritmo LQR usando una aproximación lineal del sistema en el punto medio de la trayectoria obtenida por el bípedo pasivo; por su parte la amplitud y frecuencia iniciales se tomaron unitarias ya que la trayectoria original es la del caminante pasivo.

Los coeficientes usados en la implementación de este método fueron: $\rho = 1, \chi = 1.65$ y $\gamma = 0.8$, la operación de encogimiento no se implementó ya que un análisis del algoritmo permite darse cuenta de que las iteraciones que no utilizan esta operación resultan en la modificación de un solo vértice, por lo que el algoritmo puede reescribirse para que sólo sea necesaria una evaluación por iteración, de tal forma que el algoritmo pueda implementarse en línea para optimizar la caminata del robot.

5.1.3. Resultados

En la FIGURA 5.3 y la FIGURA 5.4 se aprecia la convergencia de los parámetros y la convergencia del costo de transportación por paso obtenidos mediante la aplicación del método simplex durante la simulación de la caminata. La disminución del costo de transportación equivale a un $35\,\%$ y la velocidad de la caminata sigue siendo muy parecida.

El método propuesto produce mejores resultados que los métodos vistos debido a que considera un mayor número de variables de decisión en el planteamiento del problema.

De acuerdo a los datos de la FIGURA 5.3 la solución del algoritmo sugiere que la articulación 3, que describe el movimiento de la pierna oscilante con respecto a la cadera, necesita de una actuación tan pequeña que casi podría considerarse como una articulación pasiva. Sin embargo, no es conveniente eliminar por completo la actuación en esa articulación ya que las ganancias de retroalimentación obtenidas mediante este método sólo sirven para mantener al caminante dentro del ciclo límite. Para llevar al caminante de un estado de reposo $\dot{x} = 0$ a las condiciones del ciclo límite $x = x^*$ será necesario inducir al sistema una cantidad mayor de energía.

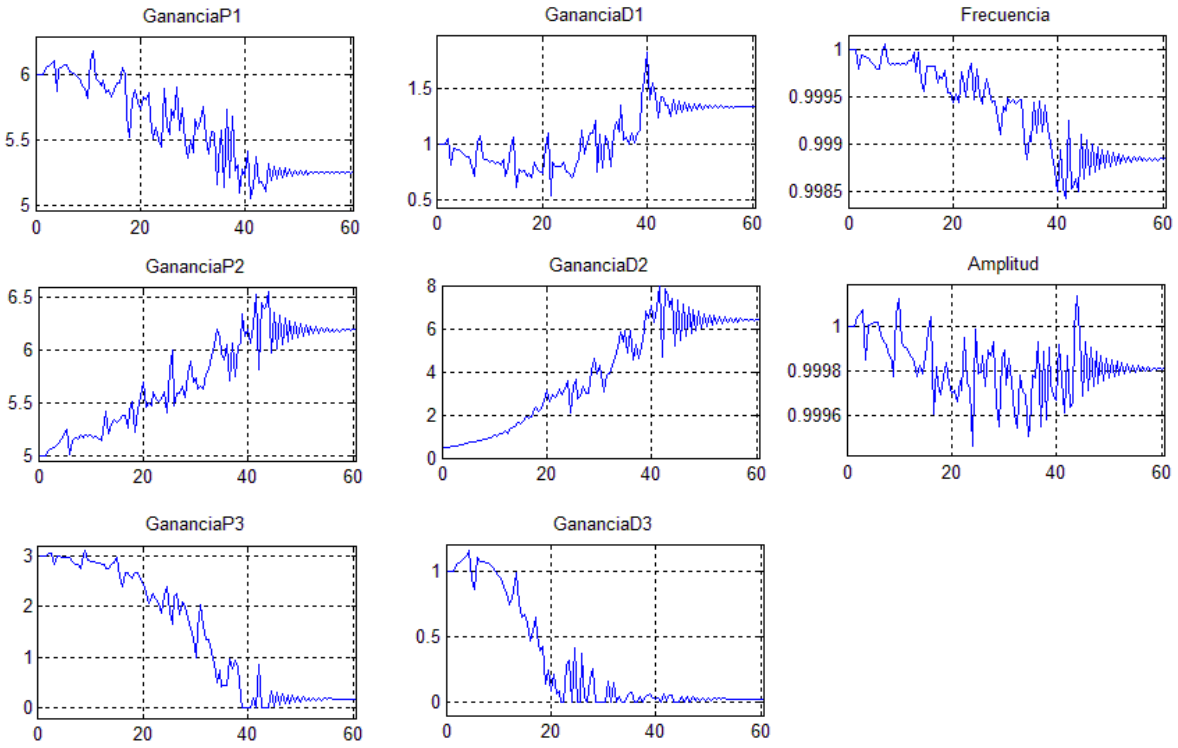


FIGURA 5.3: CONVERGENCIA DE LOS PARÁMETROS

En la FIGURA 5.4 también puede apreciarse el costo de transportación de otros mecanismos [Collins y Ruina, 2005], los datos del caminante propuesto se basan en la suposición de un $50\,\%$ de eficiencia en la transmisión de los motores.

El uso del método simplex en la búsqueda del lazo de retroalimentación y de las trayectorias óptimas de caminado presenta grandes ventajas con respecto a otros esquemas de control, principalmente en el ahorro en el consumo energético del caminante, conservando el caminado natural de los caminantes pasivos.

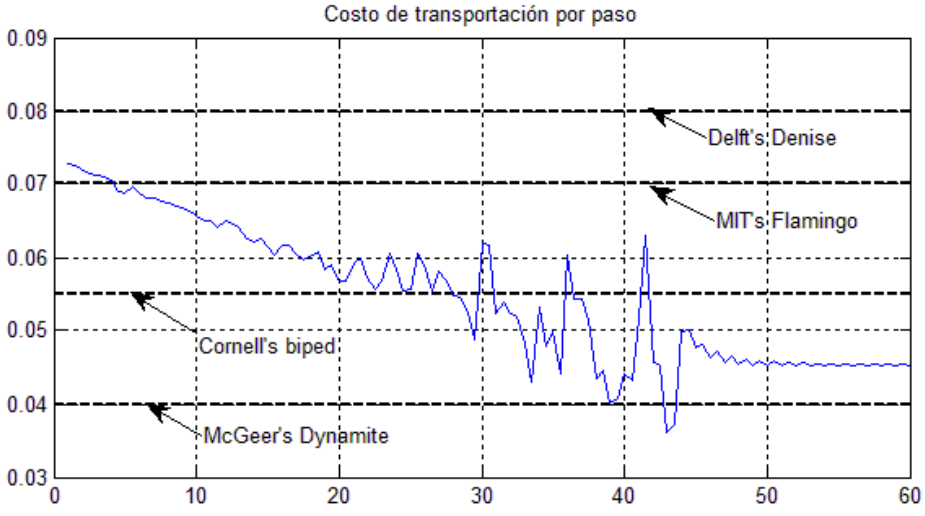


FIGURA 5.4: COSTO DE TRANSPORTACIÓN CON EL PRIMER MÉTODO

A continuación se muestra una iteración de método Simplex, considerese que en la descripción se omitió el superindice k.

Entrada: $V_k = [x_1^k\ x_2^k\ ...\ x_{n+1}^k]$ Simplex de entrada

Salida: $V_{k+1} = [x_1^{k+1}\ x_2^{k+1}\ ...\ x_{n+1}^{k+1}]$ Simplex de salida

1: ***Ordenar.*** Ordena los $n+1$ vertices tal que satisfagan $f(x_1) \leq ... \leq f(x_{n+1})$
2: ***Reflejar.*** Evaluar el *punto reflejado* $f_r = f(x_r)$, donde $x_r = \overline{x} + \rho(\overline{x} - x_{n+1})$ y $\overline{x} = \sum_{i=1}^{n} x_i/n$ es el centroide de los n mejores puntos.
3: **if** $f_1 \leq f_r < f_n$ **then**
4: **Return**(V_{k+1}) Formar V_{k+1} a partir de V_k y reemplazando x_{n+1} por x_r
5: **else if** $f_r < f_1$ **then**
6: ***Expandir.*** Evaluar el *punto expandido* $f_e = f(x_e)$ donde : $x_e = \overline{x} + \rho\chi(\overline{x} - x_{n+1})$
7: **if** $f_e < f_r$ **then**
8: **Return**(V_{k+1}) Formar V_{k+1} a partir del V_k y reemplazando x_{n+1} por x_e
9: **else**
10: **Return**(V_{k+1}) Formar V_{k+1} a partir del V_k y reemplazando x_{n+1} por x_r
11: **end if**
12: **else if** $f_n \leq f_r$ **then**
13: **if** $f_n \leq f_r < f_{n+1}$ **then**
14: ***Contraer hacia afuera.*** Evaluar el *punto contraído hacia afuera* $f_c = f(x_c)$, donde : $x_c = \overline{x} + \rho\gamma(\overline{x} - x_{n+1})$
15: **if** $f_c \leq f_r$ **then**
16: **Return**(V_{k+1}) Formar V_{k+1} a partir del V_k y reemplazando x_{n+1} por x_c
17: **else**
18: ***Encoger.*** Evalua f en los n puntos $v_i = x_i + \sigma(x_i - x_1)$, con $i = 2, ..., n+1$
19: **Return**(V_{k+1}) Formar V_{k+1} a partir de $x_1, v_2, ..., v_{n+1}$
20: **end if**
21: **else**
22: ***Contraer hacia adentro.*** Evaluar el *punto contraído hacia adentro* $f_{cc} = f(x_{cc})$, donde: $x_{cc} = \overline{x} - \rho\gamma(\overline{x} - x_{n+1})$
23: **if** $f_{cc} < f_{n+1}$ **then**
24: **Return**(V_{k+1}) Formar V_{k+1} a partir del V_k y reemplazando x_{n+1} por x_{cc}
25: **else**
26: ***Encoger.*** Evalua f en los n puntos $v_i = x_i + \sigma(x_i - x_1)$, con $i = 2, ..., n+1$
27: **Return**(V_{k+1}) Formar V_{k+1} a partir de $x_1, v_2, ..., v_{n+1}$
28: **end if**
29: **end if**
30: **end if**

Algoritmo 1: Una iteración del método Simplex

5.2. Segundo Enfoque

Para el problema mas general de optimizar el diseño de un robot humanoide completo se hicieron las siguientes consideraciones:

- **Parámetros de Caminado** G_p: Este vector incluye los parámetros que determinan las trayectorias de caminado, es decir: duración del paso t_f, amplitud lateral q_1^{max} y frontal q_2^{max} de los movimientos del tobillo, amplitud de la extensión de rodilla q_5^{max}, amplitud lateral q_8^{max} y frontal q_9^{max} del movimiento del antebrazo y la amplitud de la extensión del codo q_{10}^{max}.

- **Parámetros del Controlador** C_p: Para asegurar el seguimiento de las trayectorias deseadas se implementa un controlador local PID en cada una de las n articulaciones del humanoide, cuyas ganancias proporcional k_p^i, integral k_i^i y derivativa k_d^i con $i = 1, 2, \ldots n$, forman el vector de parámetros del Controlador C_p.

- **Parámetros Mecánicos** M_p: En este vector deben incluirse los parámetros D-H asociados a la cadena cinemática correspondiente: longitudes del muslo l_{th} y de la espinilla l_{sh}, longitudes del brazo l_{ua} y del antebrazo l_{fa}, posición del hombro l_{sd} y tamaño de la cadera l_{hp}. Para modificar la distribución de masas en las piernas se agregaron sendos discos metálicos de densidad conocida en los muslos y piernas del humanoide, por lo que también deberán optimizarse los parámetros diámetro $d_m^{1,2}$ de dichos discos y su posición $l_m^{1,2}$. Finalmente, en el vector de parámetros mecánicos del humanoide se incluye la rigidez k_s^i de los resortes torsionales que permiten modificar la frecuencia natural de 7 de las 10 articulaciones de las piernas que entran en acción en cada paso.

Una vez que se especifica el vector de parámetros S_p ya se puede evaluar el modelo de caminado a fin de obtener las trayectorias; también es posible obtener las señales de control para cada una de las n articulaciones del humanoide y calcular el consumo energético en cada zancada. Además, es posible calcular características de la zancada como la distancia recorrida y la velocidad de desplazamiento y, finalmente, con la ayuda de software de ingeniería tipo CAD es posible calcular algunas propiedades dinámicas del prototipo como los tensores de inercia de cada uno de los eslabones de la cadena cinemática, la altura y el peso total del humanoide.

5.2.1. Algoritmo de Optimización

Este segundo problema de optimización es más general que aquellos que puede resolver el método N-M, por lo que aquí resulta ineficiente. Por tal motivo decidimos enfocar nuestra atención en los métodos de la Computación Evolutiva (CE), cuyas técnicas de optimización se inspiran en la evolución biológica, de donde toma los mecanismos de combinación, mutación y selección del más apto. De entre las alternativas de optimización que ofrece la CE seleccionamos a los Algoritmos Genéticos (AG) propuestos en los años 70 por John Henry Holland [Holland, 1992].

Un AG es un mecanismo iterativo de búsqueda dirigida, pero basada en cálculo de probabilidades, que permite encontrar soluciones a problemas de optimización que convergen en probabilidad al óptimo, es decir, que en la medida en que se realizan más iteraciones, la probabilidad de encontrar la solución óptima se acerca a la unidad (la certeza).

Así, los AGs representan una buena alternativa a problemas de optimización en donde la función de costo es no derivable o cuya derivada se obtiene de manera complicada. Estos algoritmos son llamados genéticos porque hacen evolucionar una población de individuos sometiéndola a acciones aleatorias (operadores genéticos) similares a las que caracterizan a la evolución biológica como la mutación, la cruza, las recombinaciones y la selección de los individuos más aptos, según un cierto criterio predefinido. Dependiendo de la forma en que se aplican los operadores genéticos (cruzamiento, mutación), de cómo se realiza la selección del más apto y de cómo se decide el reemplazo de los individuos para formar la nueva población, es que se tiene la diversidad de AGs que se encuentran en la literatura especializada [Holland, 1992; Deb y col., 2002]. Un AG básico consta de los pasos siguientes:

1. **Inicialización.** Generación aleatoria de una población inicial de individuos (posibles soluciones).
2. **Evaluación.** Cálculo de la aptitud de cada uno de estos individuos para averiguar qué tan buena es la solución que representan.
3. **Selección.** Elección probabilista de los mejores individuos de esta población, quienes serán cruzados para dar origen a la siguiente población.
4. **Cruzamiento.** Aplicación de operadores de recombinación para producir una nueva población. Principal operador genético que representa la reproducción sexual, opera sobre dos cromosomas a la vez para generar dos descendientes combinando las características de ambos cromosomas padres.
5. **Mutación.** Operador que cambia aleatoriamente parte de la información genética de los nuevos individuos y que permite alcanzar zonas del espacio de búsqueda que no estaban cubiertas por la población actual.
6. **Condición de paro.** Continuar en el paso 2, a fin de reemplazar a la población anterior, hasta que se satisfaga un cierto criterio de paro.

A la iteración formada por los pasos del 2 al 6 se le denomina generación. En los AGs cada individuo se representa por un cromosoma: vector binario también denominado genotipo, formados a su vez por elementos denominados genes.

5.2.2. Implementación

En nuestra aplicación el genotipo se obtiene convirtiendo los valores del vector de parámetros, $S_{p,i} = [G_{p,i}\ \ C_{p,i}\ \ M_{p,i}]$, en cadenas binarias con los 62 parámetros cuantizados y concatenados (Ver TABLA 5.2).

Inicialización. Para producir la población inicial es necesario calcular la longitud total L_{tot} del genotipo, la cual depende de la cantidad de parámetros a optimizar, de sus rangos y de la resolución de su cuantización; luego se producen T_{pop} (tamaño de la población) genotipos de dicha longitud y se llenan de manera aleatoria con unos y ceros.

Evaluación. La aptitud F_i de cada individuo se calcula con base en el valor de la función para cada individuo, una aptitud grande representa una mejor solución. Para el caso de la minimización de la energía, la aptitud del i-ésimo individuo se calcula como:

$$F_i = \frac{1}{C_{tr}(S_{p,i})} \tag{5.5}$$

Para evaluar a cada individuo es necesario decodificar cada uno de los segmentos (genes) de su genotipo para obtener el valor del vector P correspondiente, mientras que la función de restricción se evalúa directamente.

Selección. El método de selección elegido se denomina selección por torneo binario y consiste en clasificar todos los individuos en una secuencia aleatoria, compararlos dos a dos y elegir al mejor de los dos, el cual devendrá un padre. Este procedimiento se realiza dos veces a fin de obtener T_{pop} padres.

El criterio que permite decidir cuál de los individuos comparados es mejor depende de su aptitud F_i así como del valor de la restricción sobre las condiciones iniciales del ciclo límite que asegura la marcha y que aparece en la ecuación 6. En este contexto se dice que una solución es factible si la función de restricción arroja el resultado "cierto" y no lo es en caso de obtener el resultado "falso". Así, cuando las dos soluciones comparadas tienen una aptitud importante se presentan los tres casos mostrados en la TABLA 5.1, en cada uno de los cuales la acción a tomar se elige con la regla denominada Superior of the Feasible [Deb y col., 2002], la cual es muy útil aún en casos de optimización multi-objetivo y que se describe a continuación:

Caso	Acción
Ambos son factibles	Elegir la solución más apta
Solo uno es factible	Elegir la solución factible
Ninguno es factible	Elegir la solución que menos viole la restricción

TABLA 5.1: SELECCIÓN POR TORNEO BINARIO CON LA REGLA SF

Cruzamiento. El operador de recombinación es el proceso mediante el cual se generan nuevos individuos a partir de los padres previamente seleccionados. Aquí se usa un operador básico denominado "mezcla de dos puntos", el cual consiste en generar los descendientes (hijos) tomando algunos genes de los padres, de uno de ellos o de ambos dependiendo del intervalo definido por los dos puntos elegidos aleatoriamente, para formar su genotipo. El parámetro que regula el cruzamiento $p_{cross} \in [0, 1]$ el porcentaje de cruzas esperados regulado mediante la variable aleatoria $r \in [0, 1]$; una vez seleccionados los padres, cuando $r \leq p_{cross}$, se ejecuta el operador de cruzamiento, de lo contrario lo hijos serán idénticos a los padres.

Mutación. El parámetro $p_{mut} \in [0, 1]$ determina el porcentaje de mutación y funciona de manera similar al porcentaje de cruzamiento; esto es, por cada gene se produce un número aleatorio $r \in [0, 1]$ y, cuando $r \leq p_{mut}$, se invierten los dígitos del gene correspondiente. Este operador permite garantizar que se pueden obtener todas las soluciones posibles al problema de optimización con el paso de las generaciones.

Uno de los efectos de los operadores de cruzamiento y mutación es que los individuos de la nueva generación pueden tener una aptitud menor que la de sus padres, lo cual puede evitarse mediante el proceso denominado elitismo. Así, cuando ocurre esta pérdida en la aptitud, el elitismo recupera al padre más apto para sustituir a alguno de los hijos menos aptos, elegido aleatoriamente.

Condición de paro. Los AGs deberán continuar sus iteraciones hasta que se alcance el óptimo. Pero, como dicho valor se desconoce, el AG se detiene cuando se satisfaga un cierto criterio de paro, que puede ser cuando no haya cambios significativos en la aptitud de una

población o cuando se haya realizado una cierta cantidad de iteraciones G_{max}. Esta última es la opción utilizada en nuestra implementación.

En los AGs se involucran varios procesos aleatorios por lo que cada vez que se ejecutan los resultados obtenidos pueden presentar ligeras variaciones; además, su desempeño depende tanto del problema de optimización a resolver como de la selección de sus parámetros: L_{tot}, T_{pop}, G_{max}, p_{cross} y p_{mut}.

5.2.3. Resultados

Los parámetros seleccionados para el AG fueron: $T_{pop} = 32$ individuos, $G_{max} = 500$ generaciones, $p_{cross} = 0.8$ y $p_{mut} = 0.01$; mientras que para la restricción se utilizó el umbral $\delta = 0.015$.

El espacio de búsqueda de los AGs es discreto y se construye con el conjunto de valores admisibles de todos los parámetros del vector S_p, que deben optimizarse. Para ello deben definirse los límites inferior y superior de cada una de esos 62 parámetros y la resolución usada para cuantizarlos, los cuales se muestran en la TABLA 5.2 junto con los valores que obtuvo el sistema de optimización.

La implementación del sistema de optimización del diseño mecánico y del controlador del humanoide requiere tener un enlace entre el software de CAD y Matlab, pues en cada una de las iteraciones se ejecutan las siguientes tareas:

- Para cada uno de los individuos $S_{p,i} = [G_{p,i}\, C_{p,i}\, M_{p,i}]$ de la población actual se calculan todas las variables físicas del humanoide (centros de gravedad, tensores de inercia, masas, etc.) con ayuda del software CAD.
- Se evalúa la función de costo en Matlab con ayuda del Modelo Dinámico completo del humanoide, así como las trayectorias articulares, los controladores, se mide el consumo energético, la distancia recorrida, el tiempo empleado y, por tanto, la velocidad de caminado
- Se genera una nueva población usando los operadores genéticos antes mencionados.

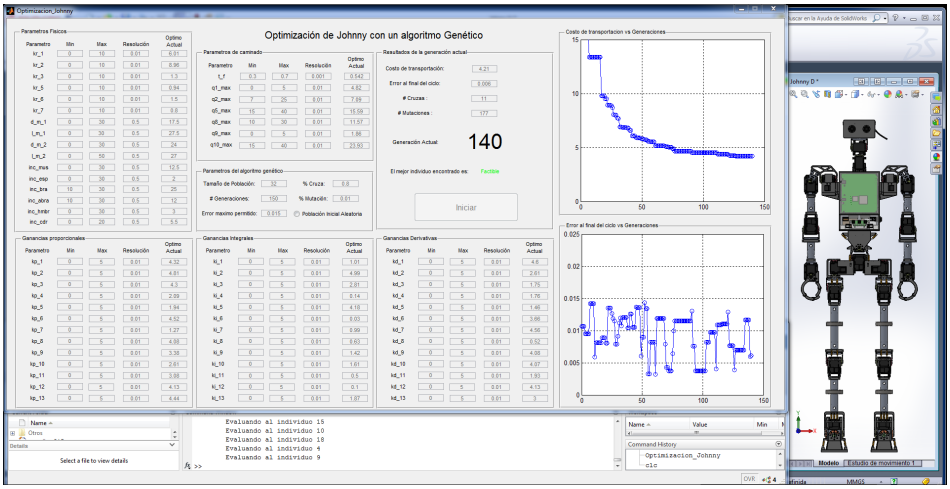


FIGURA 5.5: IMPLEMENTACIÓN DEL ALGORITMO GENÉTICO

Por su parte, los valores óptimos encontrados para las ganancias de los controladores PID de las 13 articulaciones involucradas en el proceso de optimización son:

$$
\begin{aligned}
k_p &= [\ 1.67\ 4.00\ 4.02\ 3.54\ 2.71\ 1.03\ 1.36\ 2.13\ 3.76\ 0.39\ 3.02\ 1.36\ 0.39\ ] \\
k_i &= [\ 4.35\ 0.77\ 4.48\ 0.62\ 3.04\ 4.86\ 4.56\ 0.70\ 2.78\ 2.93\ 1.34\ 4.50\ 0.51\ ] \\
k_d &= [\ 4.94\ 3.23\ 4.53\ 3.02\ 4.34\ 2.75\ 4.91\ 1.12\ 1.42\ 1.47\ 4.32\ 0.72\ 4.67\ ]
\end{aligned}
$$

Con estos valores del vector P, encontrados por el AG utilizado, el costo de transportación obtenido es de $C_{tr} = 3.72$, mientras que el error al final del ciclo es de $r(x) = 0.01$rad. La altura total del humanoide es de 890mm, su masa total es de $m_r(P) = 3.782$kg y la longitud del paso es de $d_{tr}(P) = 96$mm los cuales recorre en 0.633s, lo que representa una velocidad de avance de 0.15m/s.

Con la aplicación de técnicas de optimización provenientes de la Computación Evolutiva fue posible obtener un diseño mecánico óptimo de un humanoide con 22 articulaciones a partir de la estructura propuesta: un tórax con dos brazos, dos piernas, cintura y cuello.

Parametro	Valor min	Valor max	Resolución	Valor Óptimo	Unidades
l_{th}	174.2	204.2	0.5	202.2	mm
l_{sh}	189.7	219.7	0.5	189.7	mm
l_{ua}	129.4	149.4	0.5	129.4	mm
l_{fa}	109.7	129.7	0.5	128.2	mm
l_{sd}	200.5	230.5	0.5	200.5	mm
l_{hp}	127	147	0.5	127	mm
d_m^1	20	50	0.5	23.5	mm
l_m^1	85.32	115.32	0.5	107.32	mm
d_m^2	20	50	0.5	49	mm
l_m^2	59.53	109.53	0.5	101.53	mm
k_s^1	0	10	0.001	3.71	$\frac{g}{s^2} \times 10^9$
k_s^2	0	10	0.001	9.68	$\frac{g}{s^2} \times 10^9$
k_s^3	0	10	0.001	0.59	$\frac{g}{s^2} \times 10^9$
k_s^5	0	10	0.001	0.03	$\frac{g}{s^2} \times 10^9$
k_s^6	0	10	0.001	9.35	$\frac{g}{s^2} \times 10^9$
k_s^7	0	10	0.001	9.29	$\frac{g}{s^2} \times 10^9$
t_f	0.3	0.7	0.001	0.633	s
q_1^{max}	0	5	0.01	4.98	rad
q_2^{max}	7	25	0.01	7.01	rad
q_5^{max}	15	40	0.01	15.01	rad
q_8^{max}	10	30	0.01	10.07	rad
q_9^{max}	0	5	0.01	1.02	rad
q_{10}^{max}	15	40	0.01	16.2	rad

TABLA 5.2: ESPACIO DE BÚSQUEDA DEL AG Y VALORES ÓPTIMOS

Analizando la TABLA 5.2 se ve que el espacio de búsqueda resultante cuenta con 1.1517×10^{164} posibles soluciones! Lo cual justifica plenamente la aplicación de técnicas de optimización evolutivas, además de que no se cuenta con una función real diferenciable,

debido a las discontinuidades provocadas por las colisiones. Además, la metodología propuesta permite la optimización multicriterio.

Cabe mencionar que la optimalidad obtenida se refiere a un consumo energético mínimo con un recorrido máximo, tal como lo especifica el criterio dado por la ECUACIÓN 5.2, mientras que se mantiene la importante restricción de estabilidad en la marcha representada por la ECUACIÓN 2.6, a saber, que las condiciones iniciales y finales del ciclo límite de la marcha deben ser iguales, para lograr continuidad entre los pasos del robot.

Con los resultados de esta optimización se construyó el prototipo correspondiente, que hemos denominado Johnny, utilizando servomotores Dynamixel y un sistema computacional basado en una fitPC2 y la tarjeta controladora CM700 de Robotis. Además, utilizando las ganancias óptimas en sus controladores locales ya se hizo caminar de manera exitosa.

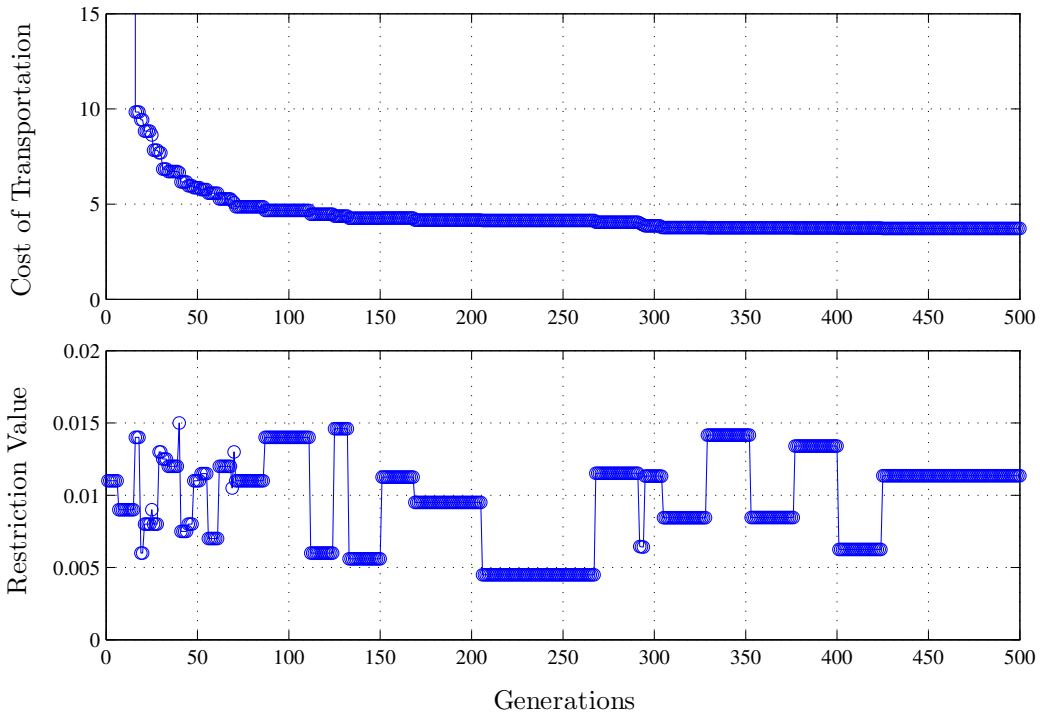


FIGURA 5.6: COSTO DE TRANSPORTACIÓN USANDO EL SEGUNDO MÉTODO

En la FIGURA 5.6 se muestra la gráfica del costo de transportación con respecto a las generaciones durante el proceso evolutivo. También se muestra el valor de la restricción dada por la ECUACIÓN 2.6, la gráfica de la segunda restricción (la cual tiene que ver con la estabilidad) no se muestra ya que, debido al lazo de retroalimentación, esta siempre se cumple.

Bibliografía del Capítulo 5

Collins, S. y A. Ruina (2005). «A Bipedal Walking Robot with Efficient and Human-Like Gait». En: *IEEE International Conference on Robotics and Automation*, págs. 1983-1988.

Deb, K. y col. (2002). «A fast and elitist multiobjective genetic algorithm: NSGA-II». En: *Evolutionary Computation, IEEE Transactions* 6, págs. 181-197.

Eiben, A.E. y J.E. Smith. *Introduction to Evolutionary Computing*. Springer, Natural Computing Series.

Holland, J.H. (1992). *Adaptation in Natural and Artificial Systems*. University of Michigan Press re-issued by MIT Press.

Kolda, T.G., R.M. Lewis y V. Torczon (2003). «Optimization by Direct Search: New perspectives on Some Classical and Modern Methods». En: *SIAM Review* 45, págs. 384-482.

Lagarias, J.C. y col. (1998). «Convergence properties of the Nelder - Mead Simplex Method in low dimensions». En: *SIAM Journal of Optimization* 9, págs. 112-147.

Nelder, J.A. y R. Meald (1965). «A simplex method for function minimization». En: *Computer Journal* 7, págs. 308-313.

Núñez, R.S. (2012). «Diseño, análisis y construcción de un robot bípedo pasivo». México city, México: CINVESTAV, Departamento de Control Automático.

Capítulo 6

Construcción de Johnny

En este capítulo se dan algunos detalles sobre la construcción del prototipo *Johnny*, los procesos de manufactura de sus partes mecánicas y el sistema electrónico que lo conforma. Por último, se muestra de manera general la estructura computacional diseñada para controlar al robot.

6.1. Estructura mecánica

Como se explicó en el capitulo 5 el diseño mecánico de *Johnny* es un diseño paramétrico generado en un programa de CAD, solamente después de terminar con el proceso de optimización se conocieron las dimensiones reales que tendría el robot, las cuales están dadas por la TABLA 5.2.

En el diseño original de *Johnny*, mostrado en la FIGURA 6.1.A, los eslabones de las extremidades están formados por un ensamble de tubos de aluminio, conectores de plástico y brackets unidos a los ejes del servomotor.

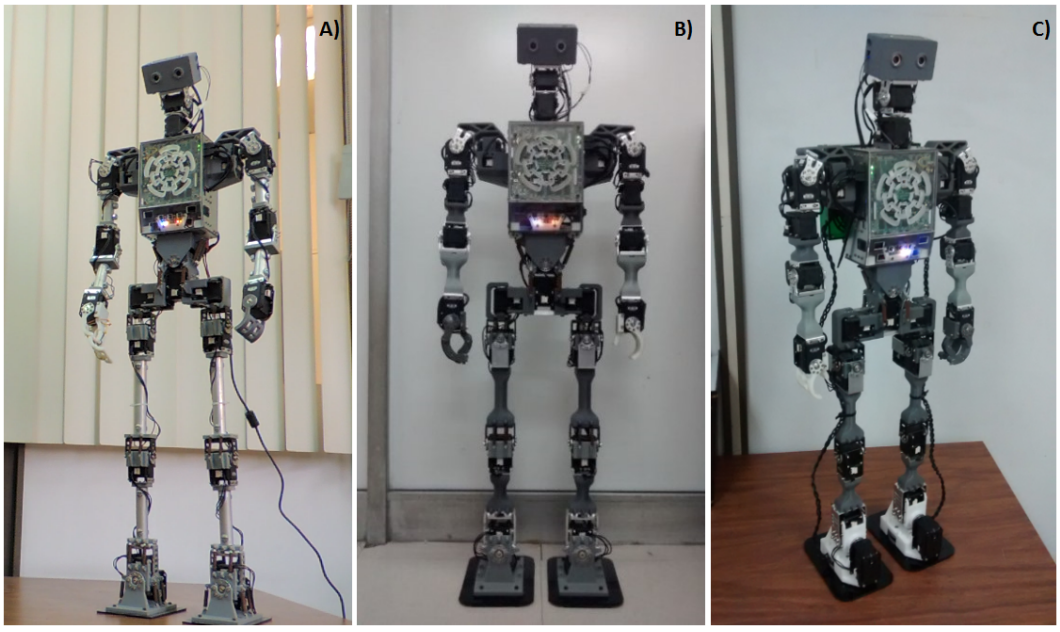


FIGURA 6.1: ASPECTO DEL HUMANOIDE JOHNNY

Las múltiples uniones en este tipo de ensamble producían movimientos relativos entre partes que deberían formar un solo eslabón, lo que producía errores de posición y vibraciones indeseadas en la cadena cinemática.

Por esta razón se decidió reemplazar los tubos de aluminio y sus conectores por piezas manufacturadas mediante impresión 3D, las cuales se muestran en la FIGURA 6.2, los nuevos eslabones se diseñaron de tal forma que su masa y tensor de inercia fueran lo más parecidos a las partes originales a fin de conservar la optimalidad original.

FIGURA 6.2: PARTES MANUFACTURADAS EN IMPRESIÓN 3D

Se conservaron los brackets para unir estas piezas al eje de los servomotores, para poder ensamblar y desensamblar las piezas en caso de ser necesario. Estos brackets se cortaron de láminas de aluminio mediante el uso de fresa CNC (FIGURA 6.3) y se les dio la forma deseada con una dobladora. El resultado de esta primera modificación se muestra en la FIGURA 6.1.B.

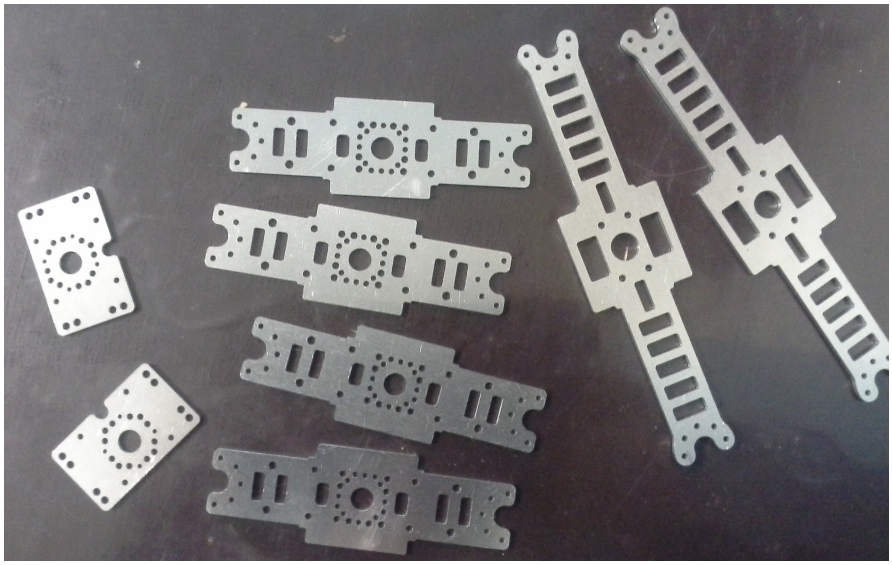


FIGURA 6.3: PARTES MANUFACTURADAS CON CORTE CNC

Mas adelante fue necesario modificar la estructura mecánica de los pies para poder añadir los sensores Optoforce, los cuales miden la fuerza y par en los pies, para hacer mediciones del equilibrio. También se agregó un soporte en la espalda para sostener las tarjetas de adquisición de datos de estos sensores a manera de mochila. El resultado de las últimas modificaciones se muestra en la FIGURA 6.1.C y la FIGURA 6.4.

FIGURA 6.4: MODIFICACIONES A LOS PIES PARA INCLUIR LOS SENSORES

Cabe señalar que estas modificaciones alteran la dinámica del robot por lo que se degrada el desempeño óptimo calculado inicialmente, sin embargo la eficiencia energética sigue siendo bastante buena como se puede ver en el capítulo 7.

6.2. Estructura electrónica

La estructura electrónica de *Johnny* esta basada en aquella que tiene el robot comercial *DARwIn-OP* (FIGURA 6.5), la cual esta formada por una computadora fit-PC2i. Esta computadora cuenta con conexion Wi-Fi, 4 puertos USB y sistema operativo Linux, una tarjeta controladora CM730 de Robotis, una botonera, un micrófono, una bocina y dos ventiladores.

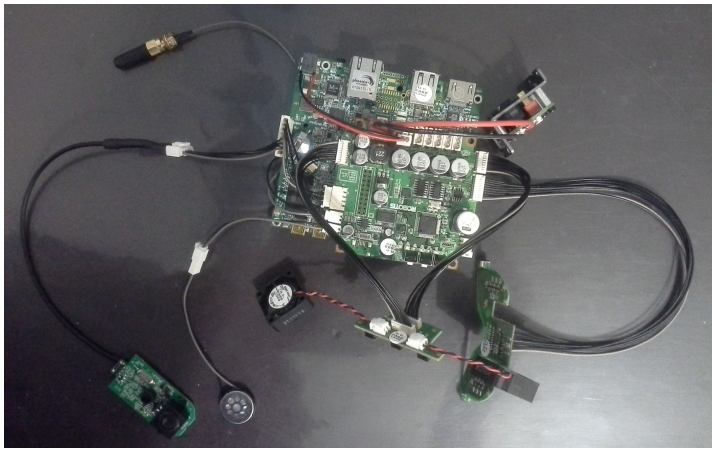


FIGURA 6.5: ELECTRÓNICA

La tarjeta CM730 cuenta con un giróscopo y un acelerómetro los cuales se utilizaron para hacer pruebas de control del equilibrio (Capítulo 8). Esta tarjeta está ubicada en el pecho, en la FIGURA 6.6 se muestra las orientación de estos sensores.

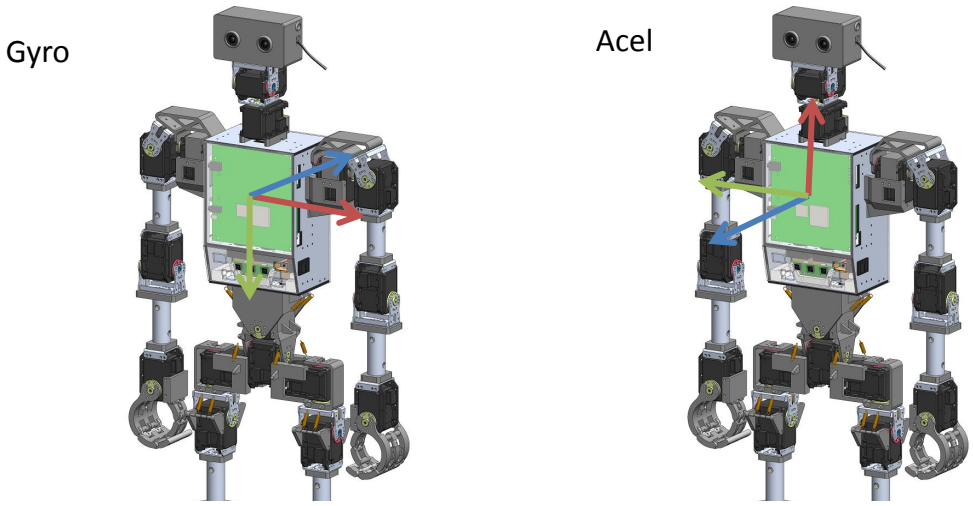


FIGURA 6.6: UBICACIÓN DE LA IMU

Con respecto a los actuadores, se utilizaron los servomotores MX-28T y MX-64T de Robotis los cuales cuentan con un protocolo de comunicación serie asíncrona tipo *half duplex*. Estos actuadores cuentan con encoder magnético, control interno PID con ganancias ajustables por el usuario y control en modo posición o en modo par (solo la versión Mx-64T), también permite conocer la posición y velocidad actuales, entre otras funciones.

FIGURA 6.7: ACTUADORES DE ROBOTIS

A este sistema se añadieron los mencionados sensores de fuerza y par de la marca *Optoforce*, de los cuales se proporciona mayor información en el capitulo 8 en el que se muestra su uso para la medición del equilibrio.

Johnny también cuenta con dos webcams en la cabeza que permiten la implementación de algoritmos de visión artificial mediante *OpenCV*.

6.3. Estructura computacional

La arquitectura computacional con la que funciona *Johnny* es un aporte propio que se desarrolló específicamente para este robot. Su programación está implementada en lenguaje C++ y hace uso de librerías adicionales para tareas especificas como *OpenCV* usada para visión artificial, *libserial* para comunicación serie y las librerías de *Robotis* usadas para la comunicación con la tarjeta CM-730 y los actuadores.

En la FIGURA 6.8 se muestra la jerarquía de las clases implementadas que conforman la estructura computacional de *Johnny*.

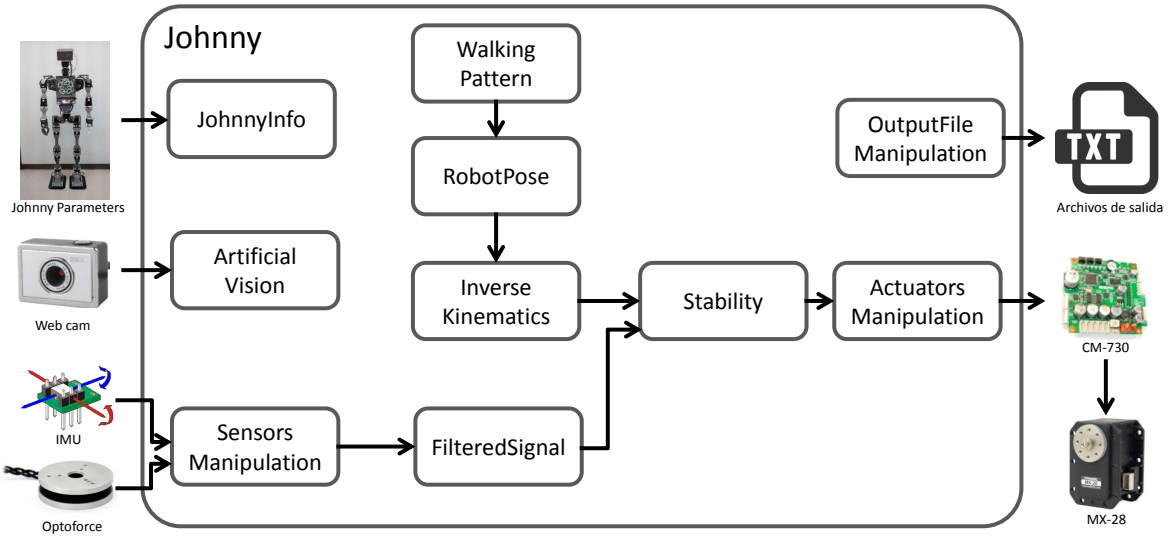


FIGURA 6.8: ESTRUCTURA COMPUTACIONAL

La clase *Johnny* es la que tiene mas alto nivel en la jerarquía y es la única que el usuario necesita importar para comunicarse con el robot, en el lado izquierdo de la figura se muestran las entradas de esta clase, que incluyen los parámetros del robot y los sensores, del lado derecho tenemos las salidas, que son las señales para manipular los actuadores por medio de la tarjeta controladora CM730 y los tres archivos de salida que tienen la información de los sensores, las trayectorias articulares y las del espacio de trabajo.

Las clases de mas bajo nivel, mostradas dentro de la clase *Johnny* se compilaron mediante archivos independientes lo cual permitirá, a los estudiantes que continúen trabajando con el robot, reusar las partes de código que necesiten de manera mas fácil.

Finalmente, cuando se decida actualizar esta plataforma al ambiente ROS (Robot Operating System) la implementación de este sistema mediante nodos y tópicos resultará sumamente sencilla, debido a la estructura modular de esta arquitectura computacional.

Capítulo 7

Comparación de la eficiencia energética

En este capítulo se muestran los resultados de comparación de la eficiencia energética de *Johnny* con respecto al DARwin-OP, robot humanoide comercial fabricado por Robotis, también se muestra la manera de normalizar los datos obtenidos con la finalidad de hacer una comparación válida entre el desempeño de distintos robots.

7.1. Definición del Costo de Transportación

La eficiencia energética de un caminado bípedo es comúnmente medida mediante una cantidad adimensional conocida como costo específico de transportación C_{tr} el cual describe la energía E usada para que un sistema recorra una distancia d por unidad de masa m y esta dado por la expresión siguiente:

$$C_{tr} = \frac{E}{m\ g\ d} \tag{7.1}$$

Este criterio se definió originalmente para comparar la eficiencia energética en animales y medios de transporte [Tucker, 1975], hoy en día también es muy común su uso para la comparación de sistemas robóticos como se muestra en la FIGURA 7.1

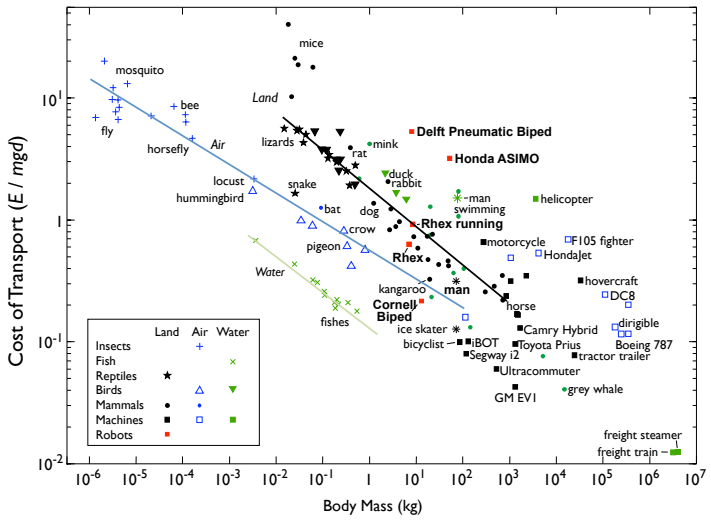


FIGURA 7.1: COSTOS DE TRANSPORTACIÓN TOMADOS DE [TUCKER, 1975]

Existen dos variantes del costo de transportación: el costo específico de transportación total y el costo específico de transportación mecánico. El primero se denomina de ese modo porque para su cálculo se considera la energía total consumida por el mecanismo al caminar, incluyendo la que consume su computadora y su sistema de control usado para generar y controlar sus trayectorias de caminado. Por su parte, el segundo solamente considera la energía consumida por los actuadores durante el caminado. Este valor es útil cuando se requiere la comparación entre mecanismos en donde el consumo energético de las arquitecturas computacionales es tan diferente que no permite hacer comparaciones válidas.

En este capítulo utilizaremos el costo específico de transportación total al cual llamaremos simplemente costo de transportación, la razón es que tanto nuestro robot *Johnny* como el DARwin-OP cuenta con la misma arquitectura computacional por lo que la comparación es válida.

7.2. Medición del Costo de Transportación

Para evaluar la ECUACIÓN 7.1 y así poder comparar el costo de transportación de los humanoides *Johnny* y DARwin-OP se desarrollo un circuito para medir la corriente de alimentación que consumen dichos robots durante el caminado. La corriente $I(t)$ medida permite calcular la carga eléctrica definida como $Q = \int_0^{t_f} I(t)dt$ en donde t_f es el tiempo que le toma al caminante recorrer la distancia d. Una vez obtenida la carga Q se puede calcular la energía consumida mediante la relación $E = QV$ donde V, el voltaje suministrado, es constante. Finalmente la masa m y la distancia recorrida d son de fácil medición.

Para medir la corriente $I(t)$ que consume el robot se utilizó el sensor ACS714 el cual se basa en el efecto Hall para producir una señal de voltaje proporcional a la corriente que pasa entre dos de sus terminales, este sensor produce una señal de salida de 187 mV por amper.

Para hace el almacenamiento y postprocesamiento de estas señales se utilizó una tarjeta Arduino Nano, la cual cuenta con entradas analógicas que permiten hacer la conversión digital con una resolución de 10 bits. También cuenta con un puerto de comunicación serie que permite transmitir los valores digitales a una computadora.

En la FIGURA 7.2 se puede observar el sencillo circuito que sirve para realizar la adquisición de datos descrita.

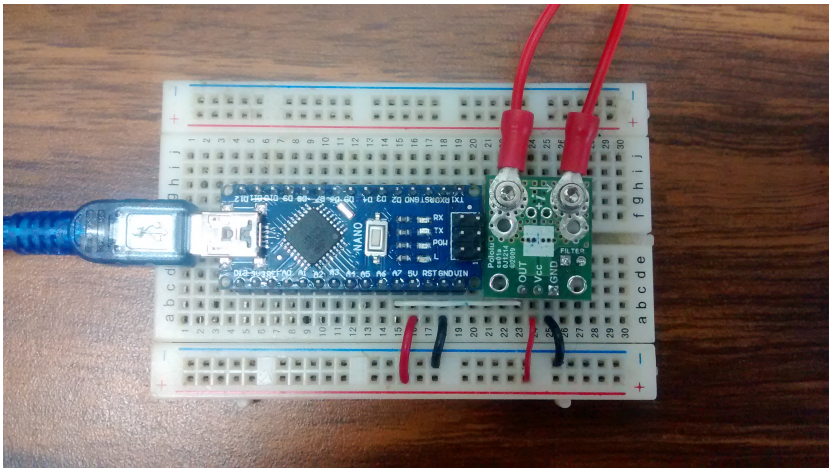


FIGURA 7.2: SISTEMA DE MEDICIÓN DE CORRIENTE

Haciendo uso de Matlab se transformaron las señales provenientes del sensor a valores de corriente, tras considerar la ganancia del sensor y la resolución del convertidor analógico digital se obtuvieron las gráficas de corriente mostradas en la FIGURA 7.3, las cuales fueron capturadas con una frecuencia de muestreo de 100 Hz.

Las gráficas presentadas muestran la corriente que suministra la fuente de alimentación para un recorrido de 1 m, cada una de las pruebas mostradas muestran la variación del consumo de corriente ante variaciones de la longitud l (mm) y duración t (ms) del paso.

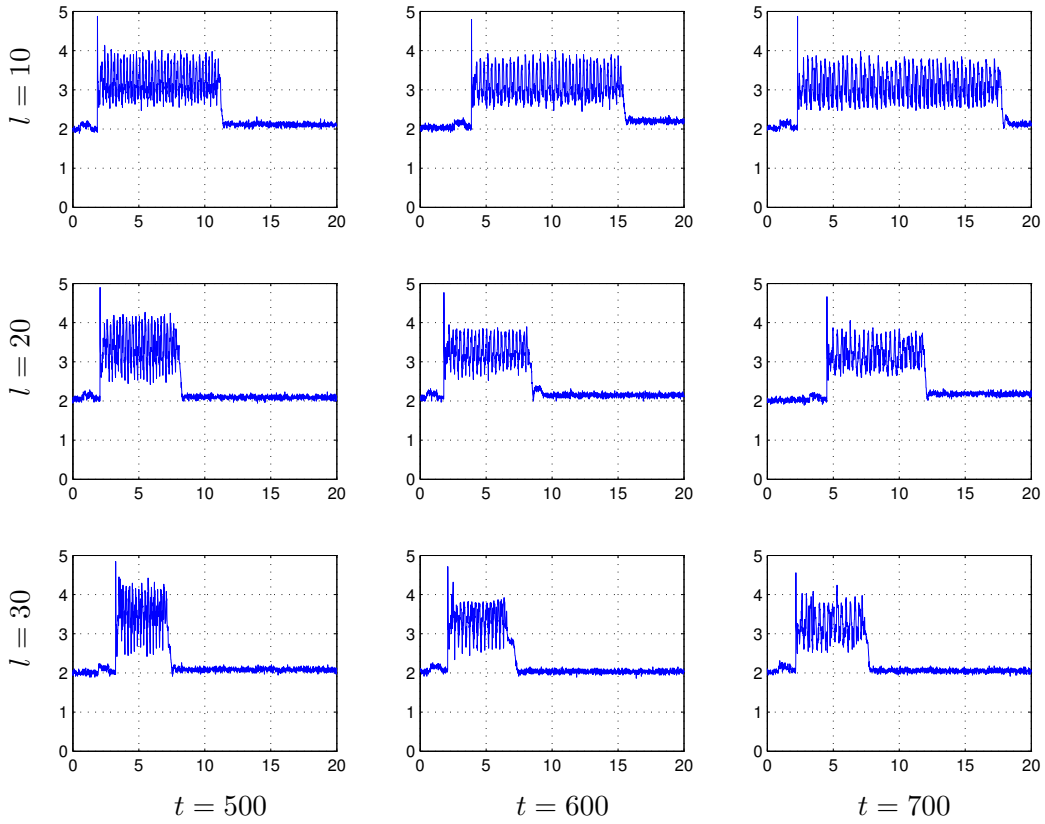


FIGURA 7.3: SEÑALES DE CORRIENTE MEDIDAS

En estas gráficas es posible identificar cuando es que el robot estaba caminando y cuando estaba en la posición de reposo, debido al aumento del consumo de corriente, de esta forma también podemos apreciar el aumento de la velocidad de caminado al aumentar la longitud y/o disminuir la duración del paso.

Utilizando esta información se programó en Matlab el cálculo de la carga eléctrica para evaluar la energía consumida, para cada uno de las pruebas realizadas, con la finalidad de evaluar el costo de transportación dado por la ECUACIÓN 7.1.

A continuación se muestran las mediciones del costo de transportación de los robots *Johnny* y *DARwin-OP* para diferentes valores de longitud y duración del paso, cada punto en la gráfica mostrada representa el promedio de al menos 5 mediciones con el mismo conjunto de parámetros.

También se agregó una línea de tendencia que permite interpolar el costo de transportación en puntos intermedios.

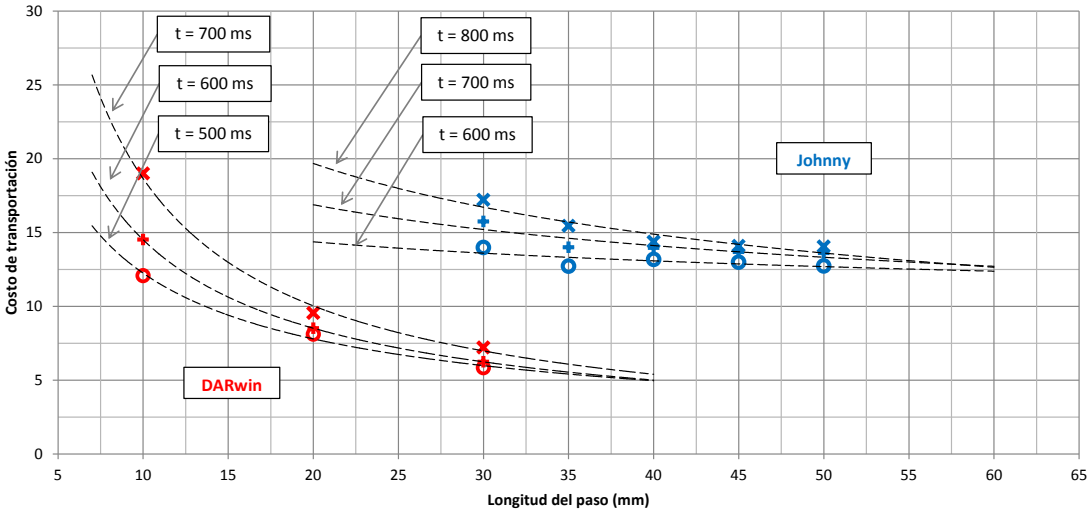


FIGURA 7.4: VARIACIÓN DEL COSTO DE TRANSPORTACIÓN

Como se puede ver en la FIGURA 7.4 el costo de transportación tiende a disminuir al aumentar la velocidad del paso, en parte debido a que se puede recorrer una mayor distancia en menor tiempo.

Sin embargo no es posible aumentar la velocidad de caminado arbitrariamente porque existen varias limitantes como por ejemplo la velocidad máxima que pueden alcanzar los actuadores y por otro lado al aumentar la velocidad aumentan los pares generados por las fuerzas de coriolis provocando rotaciones del pie sobre el piso y haciendo mas difícil controlar la estabilidad de caminado.

Las mediciones del costo de transportación se obtuvieron utilizando los siguientes parámetros para cada uno de los robots:

Parametro	**Johnny**	**DARwin-OP**
Voltaje	12v	12v
Masa	3.1Kg	5.5Kg
Distancia total	1m	1m
Duración paso min	600ms	500ms
Duración paso max	800ms	700ms
Longitud paso min	30mm	10mm
Longitud paso max	50mm	40mm

TABLA 7.1: CONDICIONES DE LAS PRUEBAS

En la siguiente sección mostraremos la normalización utilizada para hacer la comparación entre la eficiencia energética de ambos robots.

7.3. Resultados

Para que la comparación entre distintos caminantes sea válida es necesario que la medición del costo de transportación se realice a la misma velocidad de caminado, esta velocidad debe estar normalizada con el objetivo de eliminar las variaciones debido a las dimensiones de los robots .

Hay diferentes formas de normalizar la velocidad de caminado, en el área de análisis clínico se utiliza la velocidad adimensional [Stansfield, Hillman y Hazlewood, 2003], en el área de la robótica bípeda a este mismo concepto se le conoce como número de Froude, debido a su analogía con la dinámica de fluidos. La velocidad adimensional se define como $F_r = \frac{v}{\sqrt{gl}}$ donde v es la velocidad de caminado, g la aceleración de la gravedad y l la longitud de las piernas del robot.

Tomando los datos de la FIGURA 7.4 y calculando el número de Froude para cada punto, se obtuvieron las gráficas de la FIGURA 7.5, usando estas gráficas ya es posible comparar los correspondientes costos de transportación de las pruebas realizadas en *Johnny* y *DARwin-OP*, de esta forma podemos concluir que la eficiencia energética de *Johnny* es superior, en su rango de movimiento.

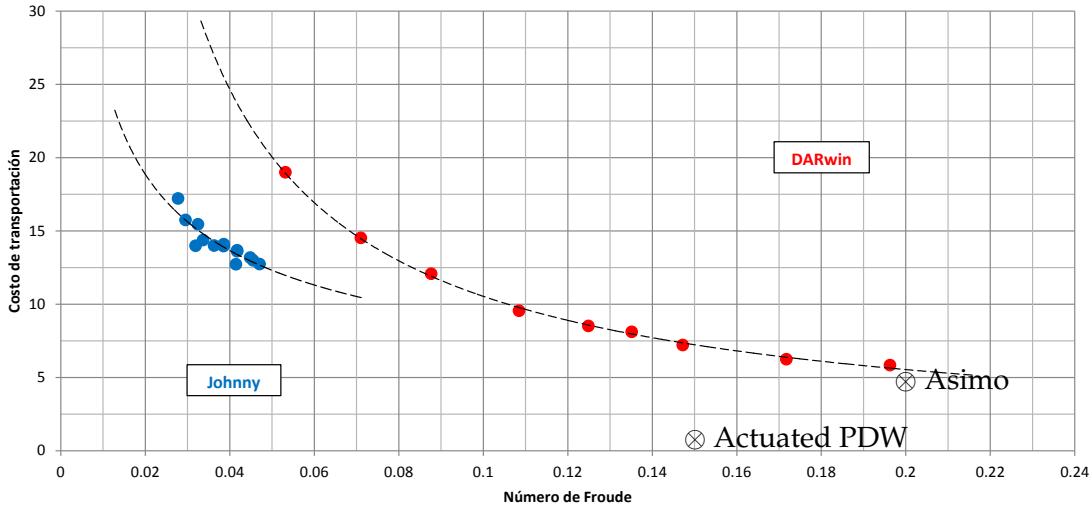


FIGURA 7.5: COMPARACIÓN DEL COSTO DE TRANSPORTACIÓN

Adicionalmente se añadieron los costos de transportación reportados en [Hobbelen y Wisse, 2007] sobre el robot Asimo y las versiones actuadas de los caminantes pasivos.

Bibliografía del Capítulo 7

Hobbelen, D. y M. Wisse (2007). *Chapter 14: Limit Cycle Walking, from the book: Humanoid Robots, Human-like Machines*. I-Tech Education y Publishing.

Stansfield, B.W., S.J. Hillman y M.E. Hazlewood (2003). «Normalisation of gait data in children». En: *Gait and Posture* 17, págs. 81-87.

Tucker, V.A. (1975). «The Energetic Cost of Moving About». En: *American Scientist*.

Capítulo 8

Control del equilibrio

En este capítulo se muestra el diseño del lazo de control externo que permite al robot mantener el equilibrio ante variaciones en la inclinación del piso o perturbaciones externas.

La estimación del equilibrio se obtiene mediante dos tipos de mediciones: la inclinación del torso o el centro de presión, para el primer caso se utiliza el giróscopo y el acelerómetro de la unidad de medición inercial (IMU) en el pecho del robot y en el segundo caso la medición se obtiene mediante el uso de los sensores de fuerza y par colocados en los pies del robot.

El equilibrio se corrige al variar la posición del torso y los brazos sin necesidad de modificar las trayectorias de caminado. En la siguiente sección se muestra la ley de control propuesta y las variaciones hechas a la implementación estándar. Posteriormente se explica el procesamiento aplicado a las mediciones de la central inercial y de los sensores de fuerza para obtener la estimación del equilibrio, finalmente se muestran los resultados obtenidos en las pruebas con el robot real.

8.1. Ley de control

Para mantener el equilibrio se propone hacer un control de velocidad en el torso utilizando un controlador PID [Astrom y Hagglund, 1995]. Se parte de la representación estándar de este algoritmo en tiempo continuo:

$$U(t) = K\left(e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau + T_d\frac{de(t)}{dt}\right) \tag{8.1}$$

En la que U es la variable de control y $e = y_r - y$ es el error entre la salida de la planta y y la señal de referencia y_r. Los parámetros de este controlador se conocen como ganancia proporcional K, tiempo integral T_i y tiempo derivativo T_d.

El controlador PID se forma de la suma de tres elementos: la acción proporcional P, la acción integral I y la acción derivativa D. Para hacer la implementación de este algoritmo en una computadora digital es necesario discretizar cada uno de estos elementos.

La acción proporcional en tiempo continuo es:

$$P(t) = K(y_r(t) - y(t))$$

Esta acción se implementa simplemente remplazando las variables continuas por las señales muestreadas:

$$P(t_k) = K(y_r(t_k) - y(t_k)) \tag{8.2}$$

En donde t_k representa un instante muestreado, es decir, el momento en el que la computadora adquiere una muestra de la señal analógica.

La acción integral está dada por:

$$I(t) = \frac{K}{T_i} \int_0^t e(\tau) d\tau$$

Existen varias formas de aproximar esta ecuación, en este caso se muestra la aproximación de Tustin, también conocida como aproximación trapezoidal:

$$I(t_k) = I(t_{k-1}) + \frac{Ks}{T_i} \left(\frac{e(t_k) - e(t_{k-1})}{2} \right) \tag{8.3}$$

En la que s representa el tiempo de muestreo.
Finalmente la acción derivativa está dada por:

$$D(t) = KT_d \frac{de(t)}{dt}$$

La cual se aproxima de forma similar:

$$D(t_k) = KT_d \left(\frac{e(t_k) - e(t_{k-1})}{s} \right) \tag{8.4}$$

Limitación de la acción derivativa

En implementaciones de control en el mundo real es común tener ruido a alta frecuencia en las mediciones de la planta. En estos casos la acción derivativa puede aumentar de manera inesperada provocando grandes dificultades. Este fenómeno se puede limitar al agregar un filtro de primer orden en la acción derivativa de la siguiente forma:

$$D(t) = -\frac{T_d}{N} \frac{dD(t)}{dt} - KTd \frac{de(t)}{dt}$$

Donde la constante de tiempo del filtro es $\frac{T_d}{N}$.
Esta nueva acción derivativa se puede aproximar como:

$$D(t_k) = \frac{T_d}{T_d + Ns} D(t_{k-1}) - \frac{KT_dN}{T_d + Ns} \left(e(t_k) - e(t_{k-1}) \right) \tag{8.5}$$

Saturación continua

Los actuadores reales tienen límites físicos y en ocasiones las variables de control pueden llegar a alcanzarlos cuando el error se hace muy grande, por lo tanto es importante considerar una función de saturación en nuestra señal de control para evitar estropear los actuadores.

La función de saturación más comúnmente usada tiene la siguiente forma:

$$U_{sat}(U) = \begin{cases} U_{max} & \text{si } U > U_{max} \\ -U_{max} & \text{si } U < -U_{max} \\ U & \text{en otro caso} \end{cases} \tag{8.6}$$

Donde U_{max} es el valor máximo al que limitamos la señal de control.

Sin embargo esta función no es continua lo que puede provocar oscilaciones no deseadas cuando la señal de control llega a la zona de saturación. Otra alternativa continua puede obtenerse usando la función $tanh$ de la siguiente forma:

$$U_{sat}(U) = U_{max} \tanh(hU) \tag{8.7}$$

Una alternativa más es la función logística o curva sigmoide, la cual se puede sintonizar para obtener el comportamiento deseado:

$$U_{sat}(U) = \frac{2U_{max}}{1 + e^{-hU}} - U_{max} \tag{8.8}$$

En ambos casos h es una ganancia usada para ajustar la curva de saturación al comportamiento deseado.

Efecto *windup* en la acción integral

Cuando la señal de control alcanza el límite de la saturación se rompe el lazo de retroalimentación ya que el actuador mantendrá su valor máximo independientemente del comportamiento de la planta.

Bajo estas condiciones la acción integral continuará creciendo, llegando a hacerse muy grande. A este fenómeno se le conoce como *windup*, la consecuencia de este fenómeno es que la variable de control tardará mucho tiempo en regresar a los límites del actuador incluso cuando la señal de referencia haya disminuido.

Para corregir este comportamiento se puede hacer uso de filtros o algoritmos que reinicien automáticamente la acción integral. Cuando la implementación del control se realiza en un sistema computacional la solución es bastante sencilla, se le conoce como integración condicional. En este método se apaga la acción integral cuando se detectan ciertas condiciones, las cuales pueden ser simplemente cuando el error de seguimiento es alto o cuando se detecta la saturación.

Ponderación de la referencia

En la ECUACIÓN 8.1 la señal de control es generada en función del error e entre la señal de referencia y_r y la salida del sistema y. Una estructura de control mas flexible se obtiene al utilizar la referencia y la salida del sistema de manera independiente de la siguiente forma:

$$U(t) = K\left(e_p(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau + T_d\frac{de_d(t)}{dt}\right) \tag{8.9}$$

En donde $e_p = by_r - y$ es el error retroalimentado a la parte proporcional y $e_d = cy_r - y$ es el error retroalimentado a la parte derivativa.

En este caso valores pequeños de los parámetros b y c atenúan el sobretiro ante cambios bruscos en la señal de referencia o para condiciones iniciales muy diferentes.

A un controlador con $b = 0$ y $c = 0$ se le llama I-PD y al controlador con $b = 1$ y $c = 0$ se le conoce como PI-D. Agregar estos parámetros permite generar mas trayectorias distintas a la señal de referencia, lo que permite satisfacer requerimientos más agresivos en el desempeño del controlador, aunque hace mas difícil la tarea de sintonización.

Error cuadrático

La formulación estándar del control PID dado por la ECUACIÓN 8.1 introduce el error de manera lineal en el algoritmo, sin embargo en ocasiones uno desearía tener ganancias aún más altas cuando el error es grande y ganancias aún más bajas cuando el error es pequeño, esto puede lograrse utilizando el error cuadrático que se define como $e_{sq} = e|e|$.

El error cuadrático generalmente solo se usa en la acción proporcional, es útil para reducir el efecto de perturbaciones de baja frecuencia, ya que aunque estas no se pueden filtrar, el controlador les dará poca amplificación cuando el error sea pequeño sin comprometer el desempeño cuando el error sea grande.

Este tipo de error también se utiliza cuando es necesario que las señales de control sean suaves y es posible mantener la salida de la planta cerca de la señal de referencia.

Implementación

A continuación se muestra la implementación considerando las variaciones mencionadas:

```
1  class PID{
2    bool isSaturated;
3    double e_m1, e_d_m1, I_m1, D_m1;
4  public:
5    double K, T_i, T_d, s, b, c, h, N, U_max, U_tol;
6
7    PID(){
8      e_m1   = 0.0;
9      e_d_m1 = 0.0;
10     I_m1   = 0.0;
11     D_m1   = 0.0;
12     isSaturated = false;
13   }
14   double control(double y_r, double y){
15     double e_p  = b * y_r - y;
16     double e_sq = e_p * abs(e_p);
17     double e    = y_r - y;
18     double e_d  = c * y_r - y;
19     double P    = K * e_sq;
20     double I    = I_m1;
21     if(isSaturated == false)
22       I = I_m1 + K * s / T_i * (e - e_m1) / 2.0;
23     double D = T_d * D_m1/(T_d + N * s)
24                - K * T_d * N * (e_d - e_d_m1) / (T_d + N * s);
25     double U = P(k) + I(k) + D(k);
26     double U_sat = U_max * tanh(h * U);
27     isSaturated = abs(U_sat - U_max) < U_tol;
28     e_m1   = e;
29     e_d_m1 = e_d;
30     I_m1   = I;
31     D_m1   = D;
32     return(U_sat);
33   }
34 }
```

FIGURA 8.1: Implementación del algoritmo PID

8.2. Uso de sensores inerciales

La unidad de medición inercial (IMU), descrita en la sección 6.2, permite conocer la velocidad angular y la aceleración lineal del pecho del robot. Con cada una de estas mediciones es posible obtener una aproximación de la inclinación del pecho.

Esta inclinación puede utilizarse como una medida del equilibrio del robot. En la figura FIGURA 8.2 se muestra una prueba realizada en *Johnny* en la que se usa el algoritmo mostrado en la sección anterior para controlar el movimiento de las articulaciones del pecho y los brazos ante perturbaciones inducidas al golpear el pecho del robot.

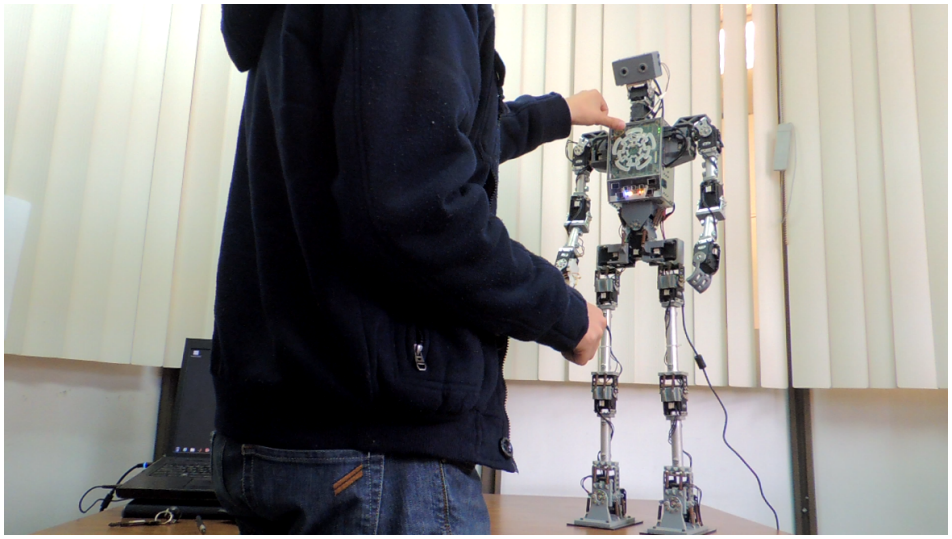


FIGURA 8.2: PRUEBA DE EQUILIBRIO USANDO IMU

La inclinación del pecho se puede estimar a partir de las velocidades angulares mediante simple integración, sin embargo, debido a la poca resolución del gyroscopo y el ruido del sensor, las señales obtenidas así presentan deriva como se puede ver en la FIGURA 8.3.

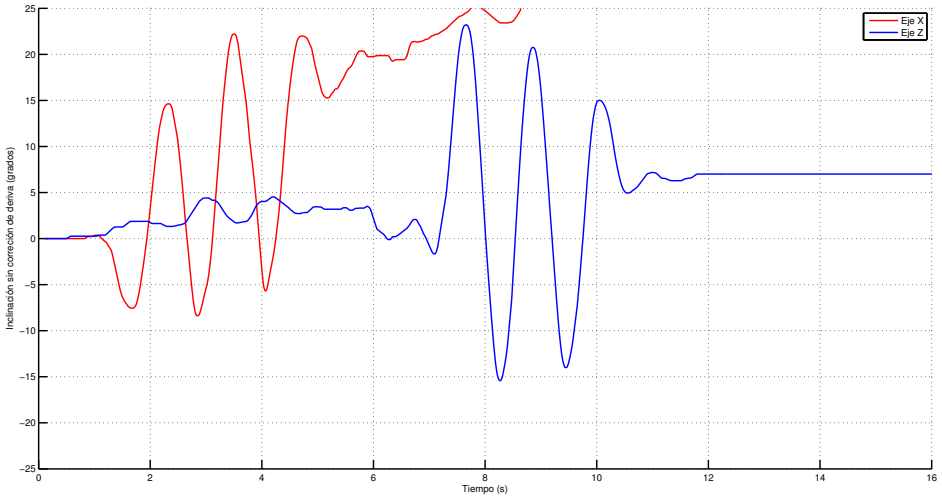


FIGURA 8.3: MEDICIONES SIN CORRECCIÓN DE DERIVA

Es posible eliminar la deriva usando integración condicionada o cancelarla directamente si se tiene un estimado de ella como se muestra en la FIGURA 8.4, sin embargo la deriva tiende a variar con respecto al tiempo por lo que es muy difícil cancelarla de manera dinámica.

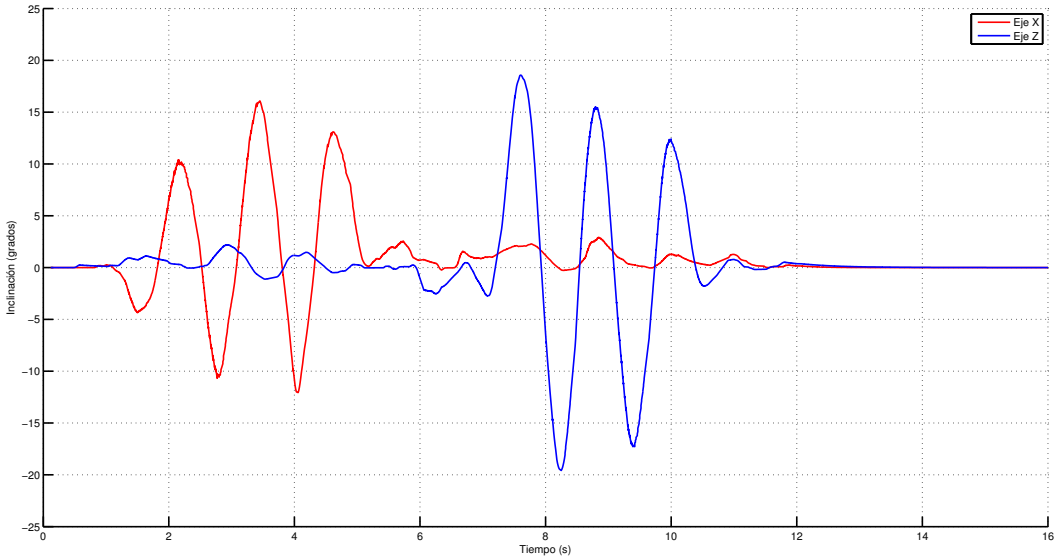


FIGURA 8.4: RESULTADOS DE LA PRUEBA USANDO GYROSCOPO

Como alternativa podemos hacer uso de las señales del acelerómetro, debido a que en todo momento estamos experimentando la aceleración de la gravedad, y esta siempre apunta al centro de la tierra, es posible obtener la inclinación del sensor en base a las componentes del vector de la gravedad medido en cada uno de sus ejes, de esta forma se obtuvieron las señales mostradas en la FIGURA 8.5.

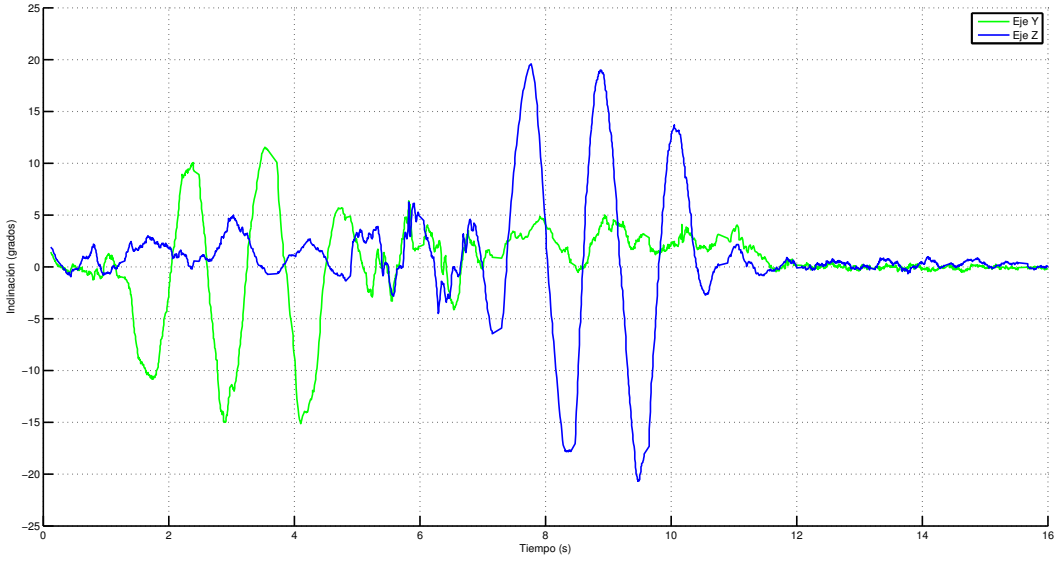


FIGURA 8.5: RESULTADOS DE LA PRUEBA USANDO ACELERÓMETRO

El problema de este método es que las aceleraciones debido al movimiento del robot alteran esta medición, por lo que sería necesario un procesamiento mas complejo que permita separar las mediciones de ambas aceleraciones para obtener una mejor estimación.

Como conclusión se puede decir que el uso de unidades de medición inercial puede usarse satisfactoriamente para controlar el equilibrio en tareas básicas, sin embargo los problemas que presenta el tratamiento de sus señales impide sus uso para tareas mas complejas como el caminado, afortunadamente se dotó al robot *Johnny* de otro tipo de sensores para este fin.

8.3. Uso de sensores de fuerza y par

El criterio de estabilidad mas usado en la actualidad es el *Zero Moment Point* (ZMP), el cual se define como el punto en el piso donde el par de volqueo, debido a la gravedad y las fuerzas inerciales, es igual a cero. El par de volqueo (*tipping moment*) es la componente del par que es tangencial a la superficie de soporte.

Por otro lado el llamado centro de presión corresponde a este mismo punto, cuando el robot está en equilibrio, y éste se define como el punto donde el par generado por la suma de las fuerzas ejercidas (normales al piso) en la superficie de contacto es igual a cero.

El centro de presión C_p se puede calcular con la expresión siguiente:

$$C_p = \frac{n \times M}{|F|} \tag{8.10}$$

Donde n es un vector normal a la superficie de contacto y los valores M y F son el par y la fuerza, respectivamente, que son ejercidos por la reacción del piso en la superficie de contacto.

El robot *Johnny* cuenta con un sensor Optoforce en cada una de las plantas de los pies, estos sensores permiten medir fuerza y par en función de la deformación de una membrana la cual se mide mediante sensores ópticos.

FIGURA 8.6: SENSORES OPTOFORCE

En la ECUACIÓN 8.10 se considera que el robot esta apoyado en un solo punto, por lo tanto para encontrar el centro de presión en la etapa de doble soporte solo es necesario hacer

la suma vectorial del centro de presión obtenido para cada pie con respecto a algún punto deseado.

$$C_p^0 = d_0^d + C_p^d + d_0^i + C_p^i \tag{8.11}$$

En la FIGURA 8.7 y la FIGURA 8.8 se muestran experimentos de pruebas de regulación en el eje x y z, en estas figuras se pueden apreciar el valor articular de los actuadores de la cadera, la inclinación del torso medida con los acelerómetros y la medición del ZMP/CoP.

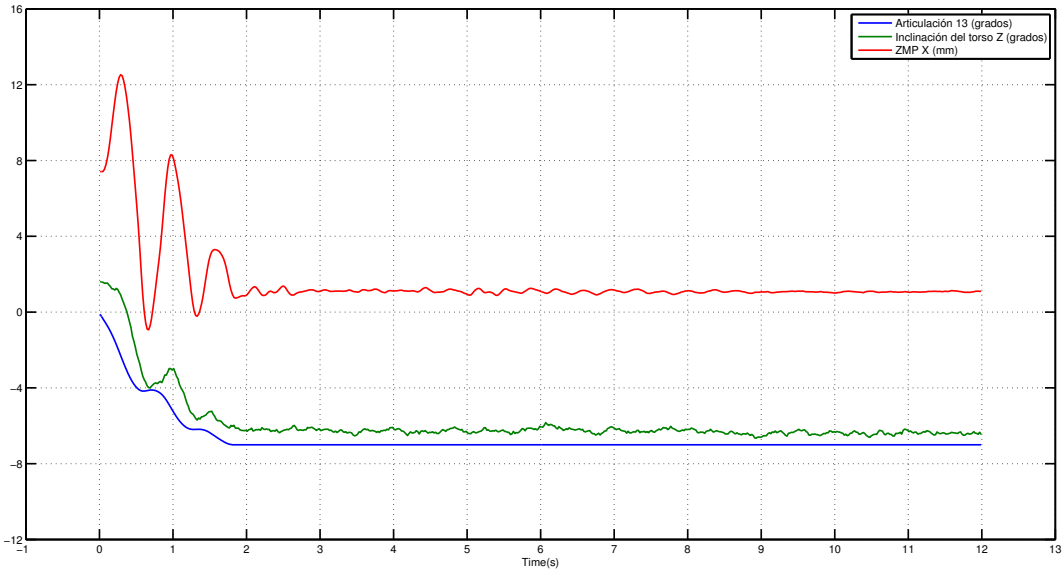


FIGURA 8.7: PRUEBAS DE REGULACIÓN USANDO OPTOFORCE (EJE X)

Las oscilaciones que se observan se deben a vibraciones provocadas por el movimiento de articulaciones no controladas en las extremidades. Un manejo de todas las articulaciones podría ayudar a mejorar este aspecto.

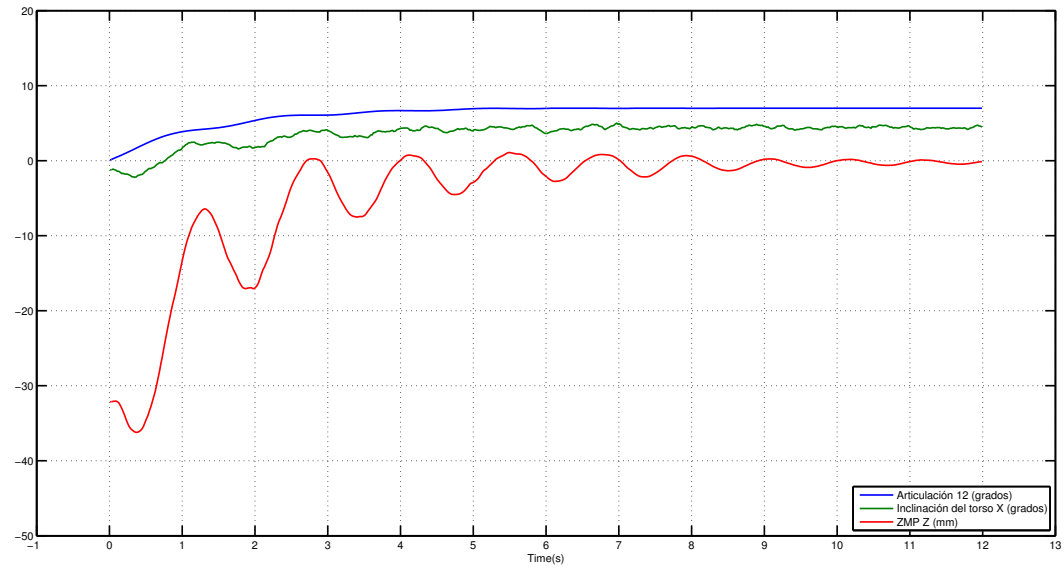


FIGURA 8.8: PRUEBAS DE REGULACIÓN USANDO OPTOFORCE (EJE Z)

En la FIGURA 8.9 y la FIGURA 8.10 se muestran experimentos de pruebas de seguimiento en el eje x y z, en estas figuras se pueden apreciar el valor articular de los actuadores de la cadera, la inclinación del torso medida con los acelerómetros y la medición del ZMP/CoP en el eje correspondiente. También se muestra la señal de referencia.

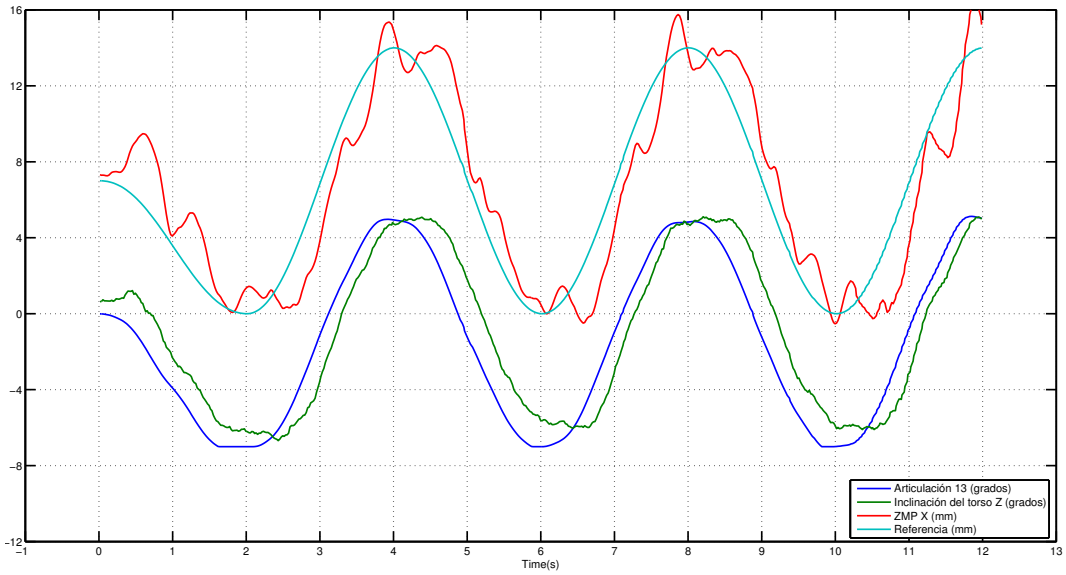


FIGURA 8.9: PRUEBAS DE SEGUIMIENTO USANDO OPTOFORCE (EJE X)

Las oscilaciones que se observan se deben a vibraciones provocadas por el movimiento de articulaciones no controladas en las extremidades. Un manejo de todas las articulaciones podría ayudar a mejorar este aspecto.

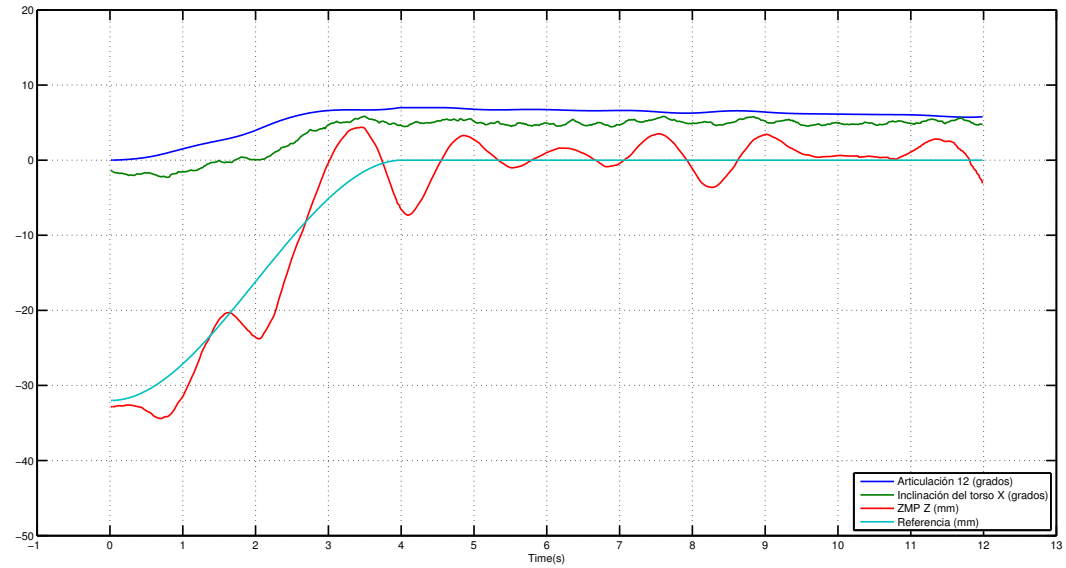


FIGURA 8.10: PRUEBAS DE SEGUIMIENTO USANDO OPTOFORCE (EJE Z)

En la FIGURA 8.11 se puede observar otra prueba realizada durante el caminado, la señal de referencia es la posición esperada del ZMP/CoP durante el caminado, también se puede apreciar la señal medida utilizando los sensores.

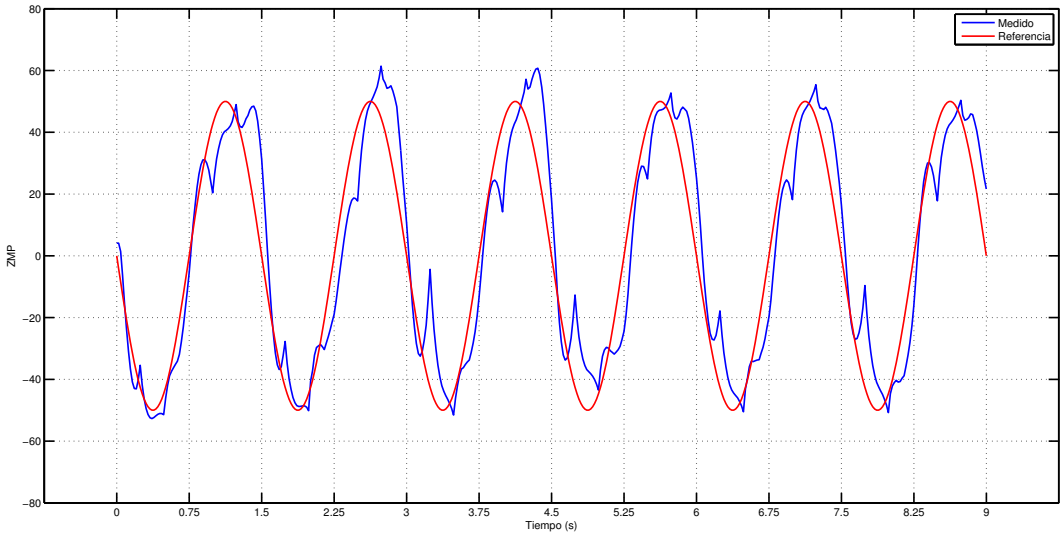


FIGURA 8.11: CONTROL DURANTE EL CAMINADO

Las discontinuidades observadas se deben al impacto producido al inicio de la etapa de doble soporte.

Bibliografía del Capítulo 8

Astrom, K.J. y T. Hagglund (1995). *PID Controllers: Theory, Design, and Tunning*. International Society of Automation.

Capítulo 9

Aplicaciones

9.1. Plataforma de Teleoperación

En esta sección se presenta una interfaz básica de teleoperación que permite al usuario interactuar con *Johnny* para modificar los parámetros de caminado así como hacer manipulación remota de objetos con retroalimentación visual.

La teleoperación es una tarea a la que comúnmente se enfrentan los diseñadores de robots humanoides [Sian y col., 2003]. En ocasiones cuando las aplicación requiere toma de decisiones rápidas en situaciones desconocidas o peligrosas, es preferible que un usuario se involucre para evitar resultados inesperados [O'Flaherty y col., 2013].

La teleoperación es una tarea desafiante ya que requiere, además del robot a operar, una interfaz de usuario y un medio de comunicación que permita al usuario intercambiar información de manera rápida y sin pérdida de datos.

Hoy en día las tecnologías móviles nos brindan una excelente posibilidad de interacción entre seres humanos y robots. Los llamados teléfonos inteligentes son una plataforma que permite desarrollar interfaces de usuario cómodas y versátiles al integrar sensores, protocolos de comunicación y dispositivos de entrada y salida de datos en un ambiente que nos resulta cada vez mas cotidiano.

En el caso de *Johnny* el proceso de teleoperación consiste en generar una comunicación *maestro-esclavo*, entre un celular con sistema operativo Android y la computadora embebida con Ubuntu en el robot, respectivamente. La interfaz gráfica desarrollada permite al usuario generar consignas articulares para los actuadores de la unidad pasajera así como activar secuencias preprogramadas en el robot.

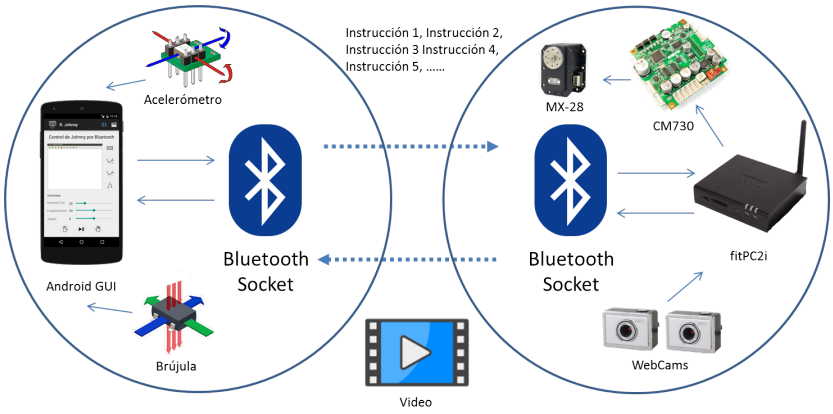


FIGURA 9.1: ARQUITECTURA DE TELEOPERACIÓN

La comunicación es bidireccional y se usa programación de hilos para enviar y recibir información continuamente. La programación del teléfono se hizo en el ambiente de desarrollo Android Studio, el cual tiene acceso a todas las librerías usadas por el sistema operativo Android, lo que permite generar la interfaz de usuario, visualización de imágenes y establecer el protocolo de comunicación Bluetooth. La programación del robot se hizo en el lenguaje C++, bajo el sistema operativo Ubuntu, cabe mencionar que el sistema informático de *Johnny* esta basado en el del robot comercial Darwin-OP de Robotis ®.

Este sistema se ha ido complementando con librerías diseñadas por nuestro equipo para implementar la generación de las trayectorias de caminado y aplicaciones de visión artificial, y también con librerías de código abierto como OpenCV y BlueZ, siendo este último el protocolo oficial de comunicación mediante Bluetooth en Linux.

9.1.1. Referencias para Extremidades

En el modo de teleoperación de extremidades, la interfaz permite manipular el ángulo de referencia de cada uno de los actuadores. Al seleccionar la cabeza, el brazo izquierdo o el derecho, se tiene acceso a las perillas que manipulan cada grado de libertad. Cuando el valor de alguna perilla se modifica la interfaz dispara una rutina que transmite una instrucción mediante la conexión Bluetooth. Cada instrucción está formada por cuatro dígitos, los dos primeros son el número de identificación del actuador y los últimos dos muestran el desplazamiento de la barra, el cual está definido en un rango de 10 a 90 unidades.

Cuando estos datos se reciben en la computadora del robot se escalan al rango de movimiento de cada actuador y se envían mediante la tarjeta controladora CM730 a cada servomotor, el lazo de control PID de cada articulación se encarga de llevarla a la posición de referencia. El dato escalado también se muestra en la interfaz gráfica a la izquierda de la barra correspondiente. Mediante la interfaz gráfica, el usuario puede elegir entre dos modos de operación para generar las referencias usadas en la teleoperación de las extremidades:

Espacio articular: En este modo se manipulan directamente las perillas lineales de cada actuador en la unidad pasajera. Los valores de cada articulación quedan almacenados en variables internas de la aplicación por lo que es posible manipular diferentes extremidades y los valores articulares se mantienen en todos los actuadores. En la Fig. 9.2 se muestran capturas de pantalla de la interfaz gráfica mostrando los valores de las diferentes articulaciones en cada extremidad, la cual es indicada con un aro azul.

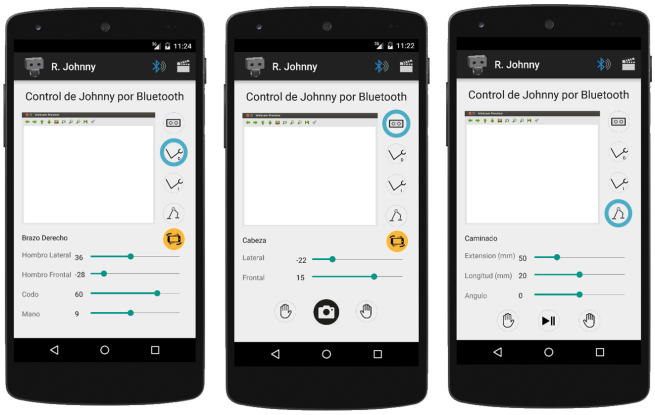


FIGURA 9.2: INTERFAZ DE TELEOPERACIÓN

Espacio de trabajo: Este modo se activa por medio de un botón amarillo que se encuentra en la interfaz de usuario, al presionarlo se activa el acceso a la información que brindan el acelerómetro y la brújula del teléfono. La información del acelerómetro permite conocer los ángulos de *Pitch* y *Roll* mientras que con la brújula se puede saber el ángulo de *Yaw*. Cada medición se hace con respecto a la posición que tenga el celular al momento de presionar el botón.

Para facilitar al usuario el proceso de teloperación, las mediciones en los sensores se consideraran incrementales en lugar de absolutas. De esta forma al inclinar el celular la correspondiente referencia aumenta o disminuye su valor y al regresarlo a la posición original la referencia se mantiene en su posición actual.

En el caso de la cabeza los ángulos de *Pitch* y *Roll* se toman directamente como el *Tilt* y *Pan* de la cabeza es decir las articulaciones q_{20} y q_{21} respectivamente.

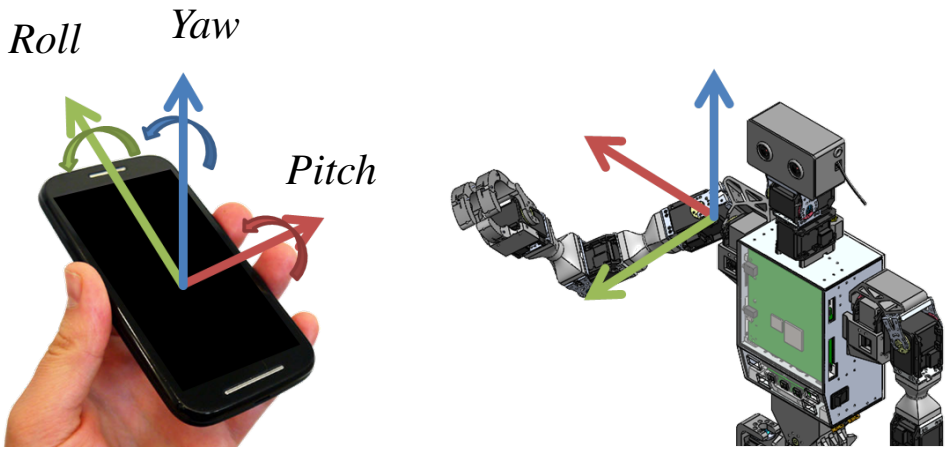


FIGURA 9.3: CONVENCIÓN DE ÁNGULOS

En el caso de los brazos, los sensores definen el vector de posición de la mano con respecto al hombro en el espacio de trabajo, para que este proceso le parezca mas intuitivo al usuario, los sensores de *Pitch* y *Roll* definen la orientación y el sensor de *Yaw* definirá la magnitud de dicho vector, de la manera en la que se muestra en la FIGURA 9.3.

9.1.2. Retroalimentación Visual

Gracias a la capacidad de procesamiento embebida en el robot *Johnny* es posible enviar imágenes tomadas desde las cámaras empotradas en la cabeza y enviarlas a la interfaz de teleoperación de manera inalámbrica. Dada la reducida velocidad de transferencia que se puede alcanzar por medio de una conexión por Bluetooth es necesario que las imágenes se compriman en algún tipo de formato, en esta aplicación se utilizó el formato de compresión *.JPG* con lo cual se alcanza una frecuencia de transmisión de imágenes de 12fps en una resolución de 320x240 pixels. El único inconveniente de utilizar la compresión para el envío de imágenes, es que el tamaño de los archivos no es constante. Por lo que es necesario transmitir la longitud del archivo a enviar antes de transmitir la imagen en sí.

El proceso de enviar y recibir imágenes se implementó en paralelo al proceso de teleoperación de las articulaciones, esto se logro usando programación de hilos (*Threads*).

En la FIGURA 9.4 se muestra el diagrama de flujo del hilo usado para transmitir imágenes.

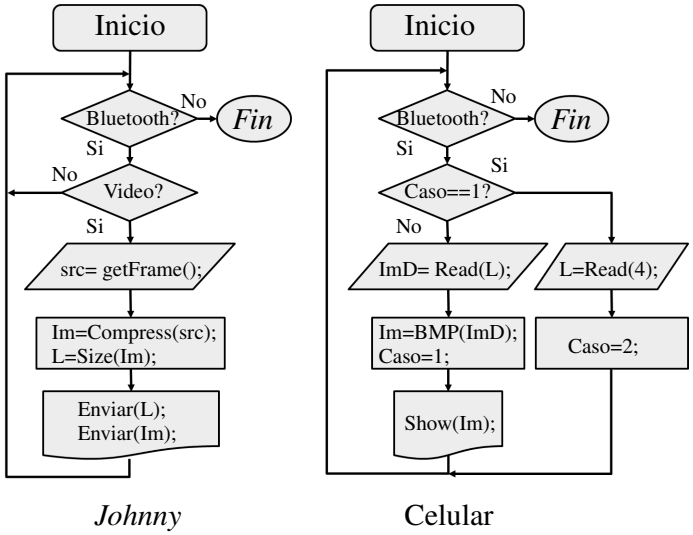


FIGURA 9.4: DIAGRAMA DE ENVIO DE IMÁGENES

9.1.3. Resultados

La interfaz gráfica desarrollada permite al usuario generar trayectorias de manera intuitiva para manipular objetos o para desplazarse usando el caminado bípedo. Mediante la retroalimentación de imágenes *Johnny* es capaz de manipular objetos o evadir obstáculos durante el caminado.

El sistema de comunicación es robusto y permite detectar errores de comunicación para restablecer la comunicación automáticamente cuando las imágenes tienen errores o cuando las instrucciones enviadas al robot son inconsistentes.

Como trabajo futuro se pretende expandir las capacidades de la interfaz gráfica para darle al usuario la capacidad de seleccionar objetos en las imágenes y que el robot pueda alcanzarlas de manera autónoma. También se quiere implementar la localización de puntos en un mapa y que el robot sea capaz de llegar a ellos detectando y esquivando obstáculos.

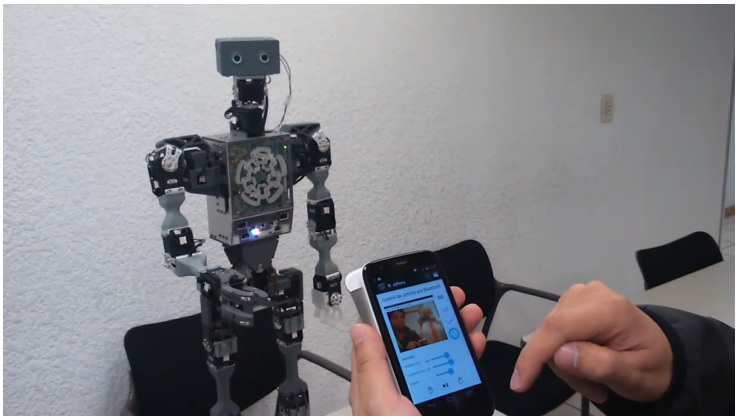


FIGURA 9.5: TELEOPERACIÓN DE JOHNNY

9.2. Odometría Visual

La odometría visual es el proceso de estimación de la pose de un vehículo, de manera incremental, en función de los cambios observados, producidos por el movimiento, en las imágenes tomadas por las cámaras a bordo del vehículo.

Este algoritmo funciona mediante la detección de puntos y su seguimiento en imágenes sucesivas. La ubicación de los puntos en pares de imágenes se usa para estimar el movimiento relativo de la cámara. En la FIGURA 9.6 se muestran las partes que componen el algoritmo y se explican a continuación.

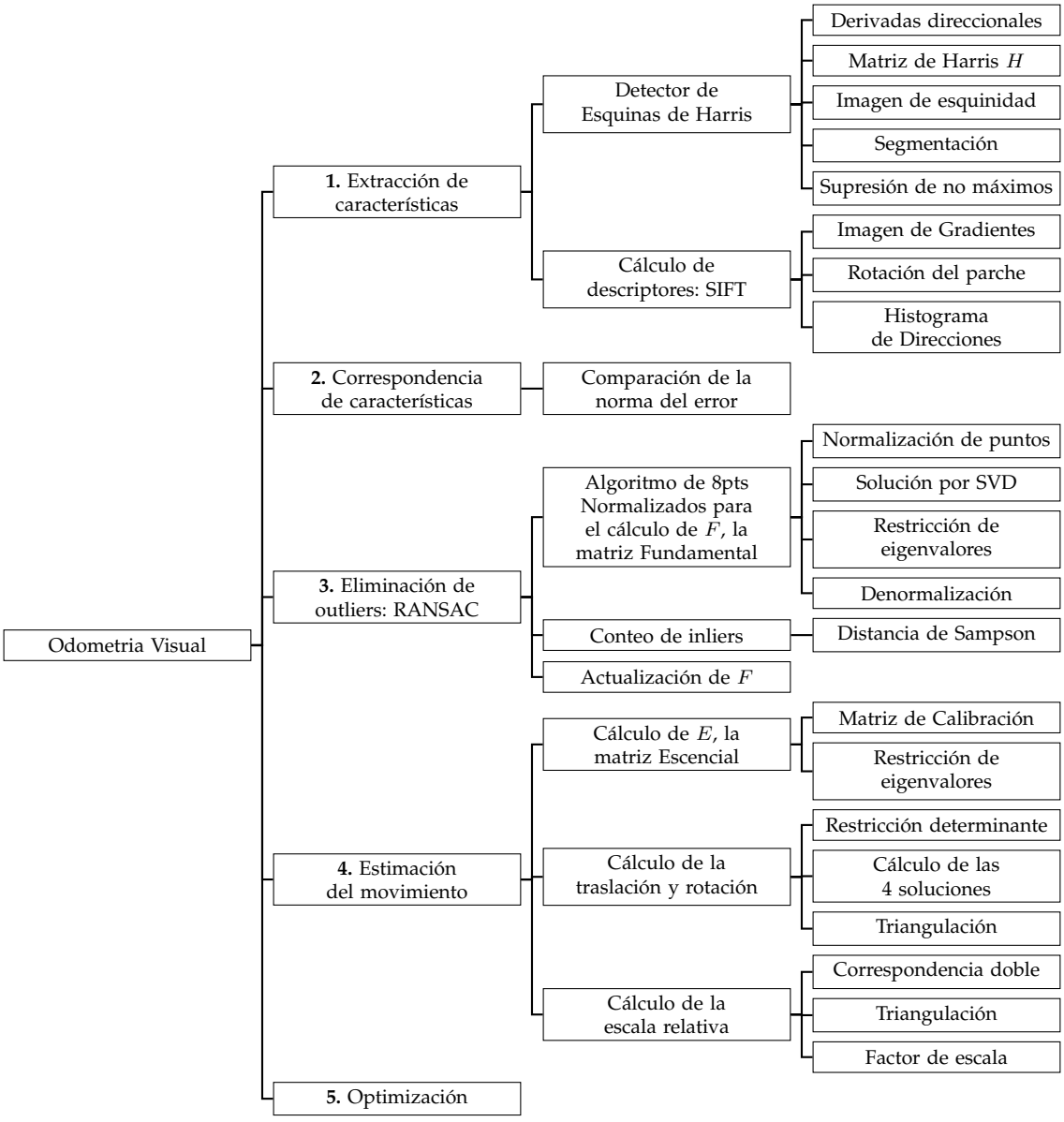


FIGURA 9.6: ESTRUCTURA DEL ALGORITMO DE VO

9.2.1. Extracción de características

Para esta parte del algoritmo se uso el detector de esquinas de Harris ya que es uno de los algoritmos que cuenta con mayor repetitividad y precisión en la localización. El hecho de que este detector no sea invariante a escala o a transformaciones afines no representó un gran problema debido a que esta implementación se pensó para funcionar a cadencia imagen, por lo que se disminuye el efecto de estos fenómenos.

Aunque comúnmente se les llama esquinas a los puntos de interés que detecta el algoritmo de Harris, en realidad estos puntos son intersecciones de contornos en los cuales se produce una gran variación de intensidad luminosa en todas las direcciones.

Considerando un parche (u, v), centrado en el pixel $I(x, y)$, la variación de luminosidad se mide de la siguiente manera:

$$E(x,y) = \sum_{u,v} w(u,v)[I(x+u, y+v) - I(u,v)]^2$$

Donde $w(u, v)$ es una función que proporciona una ganancia para producir una suma ponderada, por lo general es una función gausiana. La aproximación de primer orden para el pixel desplazado se calcula como:

$$I(x+u, y+v) \approx I(u,v) + xI_x(u,v) + yI_y(u,v)$$

Donde I_x, I_y representan las imágenes de las derivadas direccionales con respecto a x, y respectivamente.

Usando la aproximación la variación de luminosidad se puede reescribir como:

$$E(x,y) = \sum_{u,v} w(u,v)[xI_x(u,v) + yI_y(u,v)]^2$$

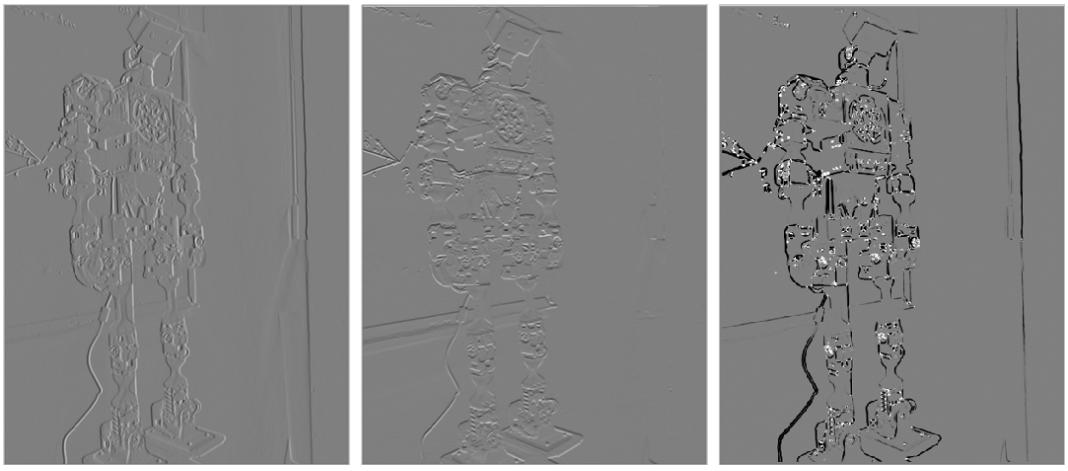


FIGURA 9.7: DERIVADAS I_x, I_y Y ESQUINIDAD

De forma matricial se puede escribir como:

$$E(x,y) = \begin{bmatrix} x & y \end{bmatrix} H(x,y) \begin{bmatrix} x \\ y \end{bmatrix}$$

Donde:

$$H(x,y) = \begin{bmatrix} \sum_{u,v} w(u,v) I_x(u,v)^2 & \sum_{u,v} w(u,v) I_x(u,v) I_y(u,v) \\ \sum_{u,v} w(u,v) I_x(u,v) I_y(u,v) & \sum_{u,v} w(u,v) I_y(u,v)^2 \end{bmatrix}$$

Analizando los valores propios λ_1, λ_2 de $H(x,y)$ se puede ver que cuando ambos son grandes el punto $I(x,y)$ corresponde a una esquina, si $\lambda_1 >> \lambda_2$ ó $\lambda_1 << \lambda_2$ corresponde a un borde y si ambos son pequeños corresponde a una superficie lisa.

La medida de esquinidad de un punto se define de la siguiente manera:

$$R(x,y) = \det(H(x,y)) - k(\text{trace}(H(x,y)))^2 = \lambda_1\lambda_2 - k(\lambda_1 + \lambda_2)^2$$

La constante k, determinada de forma empírica, tiene un valor entre 0.04 y 0.06. La segmentación se implementó usando el umbral thr y se añadió una etapa de supresión de no máximos, así se evita que un punto de interés genere mas de una característica. Usando la ventana (u,v) el punto I(x,y) es una esquina si $R(x,y) > thr$ y $R(x,y) == \text{máx}_{u,v}\, R(u,v)$

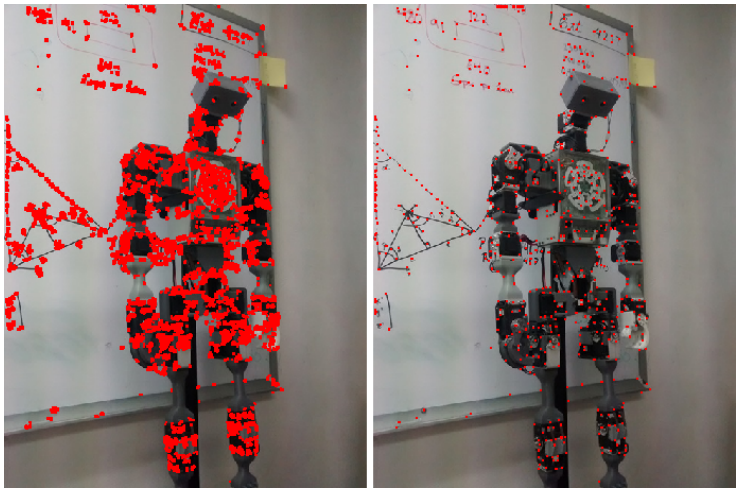


FIGURA 9.8: SEGMENTACIÓN Y SUPRESIÓN DE NO-MAXIMOS

A continuación se presenta el algoritmo de detección de esquinas de Harris:

Data: Imagen de Intensidad I
Result: Imagen Segmentada F
for $\forall(x,y)$ **do**
  $I_x(x,y) = \sum_{u,v} G_x(u,v) I(u,v)$; `// Operador Sobel`
  $I_y(x,y) = \sum_{u,v} G_y(u,v) I(u,v)$; `// Operador Sobel`
  $H(x,y) = \begin{bmatrix} \sum_{u,v} w(u,v) I_x(u,v)^2 & \sum_{u,v} w(u,v) I_x(u,v) I_y(u,v) \\ \sum_{u,v} w(u,v) I_x(u,v) I_y(u,v) & \sum_{u,v} w(u,v) I_y(u,v)^2 \end{bmatrix}$;
  $R(x,y) = \det(H(x,y)) - k(\text{trace}(H(x,y)))^2$; `// Imagen de Esquinidad`
  if $R(x,y) > thr$ ***and*** $R(x,y) == \text{máx}_{u,v}\, R(u,v)$ **then**
    $F(x,y) = 1$; `// Segmentación, No-Máximos`
  else
    $F(x,y) = 0$;

Algoritmo 2: DETECCIÓN DE ESQUINAS DE HARRIS

Cálculo de descriptores: SIFT

El descriptor SIFT se construye con cuatro histogramas de la orientación del gradiente. Este descriptor es estable ante cambios de iluminación, rotación, escala y cambios en el punto de vista de hasta 60°. La imagen G, de la dirección del gradiente, se calcula como $\theta(x, y) = \arctan(\frac{I_x(x,y)}{I_y(x,y)})$.

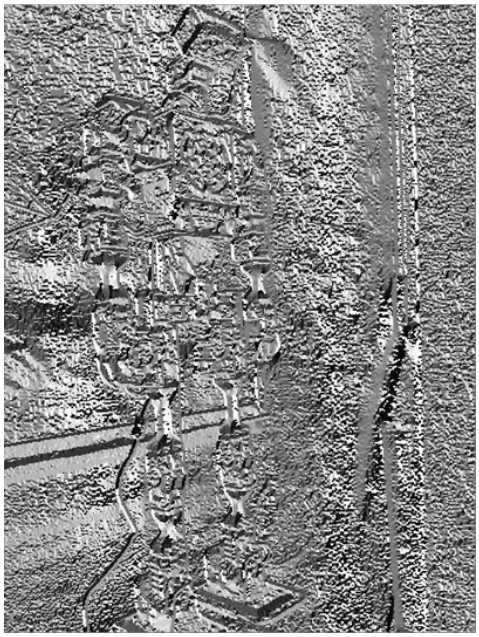


FIGURA 9.9: IMAGEN DE DIRECCIONES DEL GRADIENTE

La orientación del parche (u, v) en la imagen G se obtiene seleccionado la dirección con el pico mas alto en el histograma de orientaciones del parche. Después todas las direcciones del parche se rotan con la dirección previamente calculada.

Para definir el descriptor del punto $I(x, y)$ se divide en cuatro partes el parche rotado en G, para cada parte se calcula un histograma tomando ocho posibles direcciones los cuatro histogramas se concatenan formando el descriptor que tiene 32 elementos.

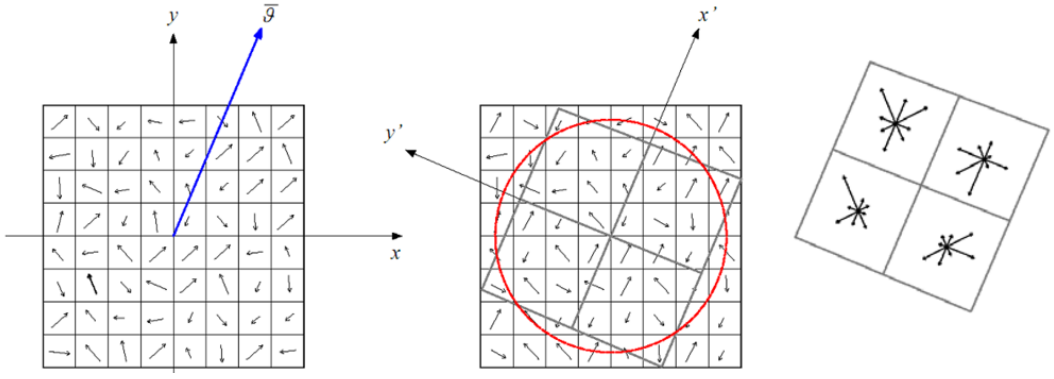


FIGURA 9.10: DESCRIPTOR SIFT

9.2.2. Correspondencia de características

Para emparejar las características de dos imágenes se comparan los descriptores de la primer imagen con los de la segunda. La medida de similitud al usar el descriptor SIFT es la norma euclidiana, si la diferencia entre dos descriptores es menor a algún umbral se considera que los puntos son correspondientes.

Para evitar que un punto en la primer imagen se empareje con mas de un punto en la segunda imagen, se lleva un registro de la disponibilidad, la cual se actualiza cuando un punto es emparejado por primera vez y se verifica para futuros emparejamientos.

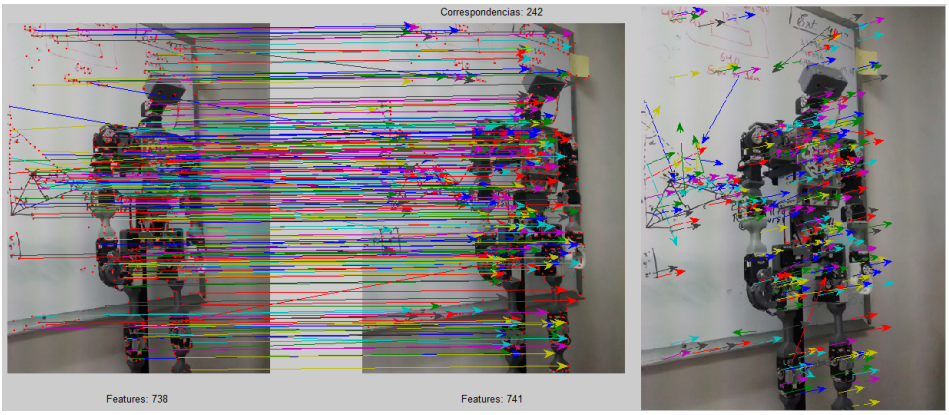


FIGURA 9.11: CORRESPONDENCIA ENTRE DOS IMÁGENES

Otra restricción al emparejamiento es la distribución espacial, de acuerdo al movimiento de la cámara los pixels correspondientes se encontrarán dentro de una región, la cual puede predecirse por medio de mediciones sensoriales o considerando velocidad constante. En esta ocasión el área de búsqueda es un circulo centrado en el punto de la primer imagen.

9.2.3. Eliminación de Outliers: RANSAC

En la FIGURA 9.11 se pueden distinguir falsas correspondencias u *outliers* que son parejas de pixeles que cumplen con el criterio de similitud pero no representan al mismo punto. Durante la estimación del movimiento los *outliers* introducen error, el cual se acumula en cada iteración, por lo que es importante eliminar la mayor cantidad de ellos.

El algoritmo RANSAC, resumido a continuación, es un método para seleccionar el modelo que se ajusta mejor a la mayor cantidad de inliers.

Data: Conjunto de datos S
Result: El modelo al que se ajusta la mayor cantidad de puntos
$S_{max} = 0$;
while $k < N$ ***and*** *size*(S_{max})$< T$ **do**
    Seleccionar aleatoriamente s puntos y ajustarles un modelo;
    Encontrar S_k puntos cuya distancia al modelo es menor al umbral t;
    if *size*(S_k) $<$ *size*(S_{max}) **then**
        $S_{max} = S_k$;
Reestimar el modelo usando todos los puntos en S_{max};

Algoritmo 3: RANSAC

Algoritmo de 8pts Normalizados para el cálculo de F, la matriz Fundamental

El modelo mencionado en el algoritmo RANSAC, en el caso de VO, corresponde a la matriz fundamental F, la cual es la representación algebraica de la geometría epipolar, esta se obtiene a partir del mapeo de puntos correspondientes y la correspondiente linea epipolar.

La matriz fundamental $F \in \mathcal{R}^3$ es una matriz homogénea de rango 2 que satisface la siguiente ecuación:

$$\mathbf{x}'^T F \mathbf{x} = 0 \tag{9.1}$$

Para todos los puntos correspondientes $\mathbf{x} \leftrightarrow \mathbf{x}'$. Escribiendo $\mathbf{x} = (x, y, 1)^T$ y $\mathbf{x}' = (x', y', 1)^T$ cada pareja de puntos correspondientes produce una ecuación lineal con respecto a los elementos de F de la siguiente forma:

$$x'xf_{11} + x'yf_{12} + x'f_{13} + y'xf_{21} + y'yf_{22} + y'f_{23} + x'f_{31} + y'f_{32} + f_{33} = 0.$$

Para un conjunto de n puntos y escribiendo los elementos de F en forma de columna f se obtiene el siguiente sistema de ecuaciones lineales:

$$Af = \begin{bmatrix} x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x'_1 & y'_1 & 1 \\ : & : & : & : & : & : & : & : & : \\ x'_nx_n & x'_ny_n & x'_n & y'_nx_n & y'_ny_n & y'_n & x'_n & y'_n & 1 \end{bmatrix} f = 0$$

No es posible determinar la escala de f por lo que para encontrar la solución A debe tener a lo mucho rango 8, si el rango es exactamente 8 la solución es única y se puede encontrar mediante métodos lineales. En general esto no se cumple, debido al ruido, y la matriz A se considera de rango 9. En este caso se puede encontrar la solución que minimize el error, la cual es el vector que corresponde al menor valor singular de A, es decir la ultima columna de V en la descomposición en valores singulares de la matriz $A = UDV^T$. Este vector minimiza $||Af||$ sujeto a $||f|| = 1$.

A continuación se muestra el algoritmo de 8pts Normalizados para el cálculo de la matriz fundamental F.

Data: $n \geq 8$ Correspondencias $\mathbf{x} \leftrightarrow \mathbf{x}'$
Result: La matriz fundamental F
Encontrar las matrices T y T' que normalizan los puntos $\mathbf{x}$ y $\mathbf{x}'$ en traslación y escala;
Normalizar usando: $\hat{\mathbf{x}}_i = T\mathbf{x}_i$ y $\hat{\mathbf{x}}'_i = T\mathbf{x}'_i$;
Formar la matriz $\hat{A}$ usando las correspondencias $\hat{\mathbf{x}}_i \leftrightarrow \hat{\mathbf{x}}'_i$;
Determinar $\hat{F}$ usando **SVD**($\hat{A}$);
Reemplazar $\hat{F}$ por $\hat{F}'$ tal que **det**$\hat{F}'$=0 usando **SVD**($\hat{F}$);
Denormalizar haciendo $F = T'^T \hat{F}' T$;

Algoritmo 4: ALGORITMO DE 8PTS NORMALIZADOS PARA F

La normalización sugerida es una traslación y escalamiento, para cada imagen, de tal forma que el centroide de los puntos coincida con el origen y que la distancia promedio de cada punto al origen sea $\sqrt{2}$. Nótese que la condición de singularidad debe ser inducida antes de la normalización.

Conteo de inliers

La distancia mencionada en el algoritmo RANSAC, en el caso de VO, corresponde a la distancia de Sampson, la cual es una aproximación del error de reproyección y se muestra a continuación.

$$d_i^2 = \frac{(\mathbf{x}_i'^T F \mathbf{x}_i)^2}{(F\mathbf{x}_i)_1^2 + (F\mathbf{x}_i)_2^2 + (F^T\mathbf{x}_i')_1^2 + (F^T\mathbf{x}_i')_1^2}$$

Dada una estimación de F, esta distancia mide que tanto satisfacen la geometria epipolar el par de puntos correspondientes $\mathbf{x} \leftrightarrow \mathbf{x}'$.

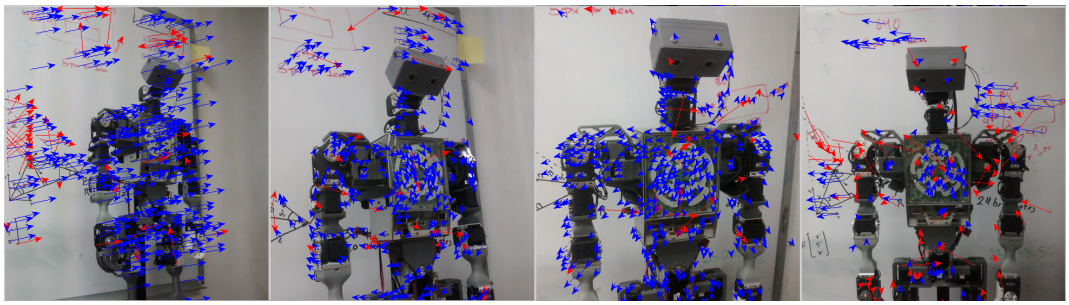


FIGURA 9.12: ELIMINACIÓN DE OUTLIERS

9.2.4. Estimación del movimiento

En el caso de que se usen cámaras calibradas el algoritmo normalizado de 8 puntos calcula la matriz Escencial E en lugar de la matriz fundamental F, sin embargo la matriz E satisface la condición adicional de que los dos valores singulares distintos de cero deben ser iguales y es necesario inducir esta restricción antes de la denormalización.

Cálculo de E, la matriz Escencial

La matriz Fundamental F y la matriz Esencial E cumplen con la siguiente relación:

$$E = K'^T F K$$

Donde K y K' son las matrices de calibración de los puntos en la primera y la segunda imagen respectivamente. Si se utiliza esta relación para el cálculo de E es necesario forzar a que sus dos valores singulares sean iguales, esto se puede hacer fácilmente descomponiéndola mediante $E = UDV^T$ y reescribiéndola como:

$$E' = U \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^T \tag{9.2}$$

Cálculo de la traslación y rotación

Conociendo la matriz Esencial la posición relativa de las cámaras pueden ser obtenida salvo un factor de escala. Si se asume que la primera cámara esta orientada y posicionada

en el origen, entonces tomando la matriz esencial descompuesta en la forma de la ECUACIÓN 9.2, tenemos que el vector de traslación t y la matriz de Rotación R están dadas por:

$$t \equiv \pm[u_{13}, u_{23}, u_{33}]^T \quad R \equiv UWV^T \text{ or } R \equiv UW^TV^T$$

Donde $W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Es posible ver que las 4 soluciones existentes cumplen con la restricción de la geometría epipolar, tal como se muestra en la FIGURA 9.13.

La única realización posible es aquella en la que el punto esta frente a ambas cámaras, por lo que solo se necesita triangular un punto para determinar en cual solución la componente Z es positiva para cada cámara.

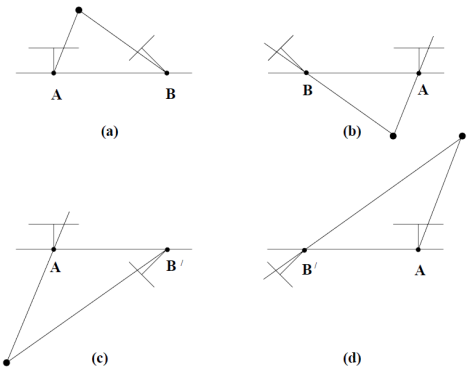


FIGURA 9.13: LAS 4 POSIBLES SOLUCIONES OBTENIDAS A PARTIR DE E

Si denotamos por $P_2 = [R_k t_k]$ a la matriz que determina la posición de la segunda cámara, con $P_1 = [\text{diag}(1, 1, 1)\ 0]$ la matriz de la cámara en el origen y por $\mathbf{X} = [x, y, z, s]^T$ al vector formado por las coordenadas del punto triangulado y su escala, el signo de la profundidad del punto 3D con respecto a la cama P_i esta dada por:

$$\text{depth}_i = \text{sign}(P_i(3,:)\mathbf{X}\mathbf{X}_4\text{det}(P_i(:, 1:3)))$$

De esta forma la solución correcta sera aquella para la cual $\text{depth}_1 = \text{depth}_2 = +1$

Cálculo de la escala relativa

Como ya se mencionó no es posible obtener la escala del vector de traslación t, de hecho la matriz U de la cual se obtiene t esta formada por vectores unitarios por lo que a su vez t es solo un vector unitario en la dirección del movimiento. Sin embargo es posible calcular la escala relativa para las siguientes transformaciones. Una manera de hacerlo es triangular puntos en dos pares subsecuentes de imágenes: $\mathbf{X}_{k-1}, \mathbf{X}_k$ de puntos correspondientes en 3D.

Como la distancia entre los puntos 3D de la escena real no cambia es posible obtener la escala relativa r a partir de la razón entre las distancias observadas entre una pareja de imágenes y otra de la siguiente forma:

$$r = \frac{||\mathbf{X}_{k-1,i} - \mathbf{X}_{k-1,j}||}{||\mathbf{X}_{k,i} - \mathbf{X}_{k,j}||}$$

Para añadir robustez se toman muchas parejas de puntos y la escala se toma como la media. La escala relativa r obtenida se multiplica al desplazamiento t obtenido de la segunda pareja de imágenes.

9.2.5. Optimización

El algoritmo de Odometría Visual calcula la posición de la cámara concatenando las transformaciones obtenidas a partir de dos frames sucesivos, sin embargo también es posible calcular las transformaciones entre el frame actual y los últimos n frames o contra cualquier otro frame pasado. Estas transformaciones se pueden usar entonces como restricciones adicionales para optimizar la trayectoria.

La optimización de la trayectoria puede hacerse minimizando la siguiente función de costo:

$$\sum_{e_{ij}} ||C_i - T_{e_{ij}} C_j||^2$$

Donde C_k es la posición con respecto al origen y $T_{k,k-1}$ es la transformación entre frames. Esta función de costo es no-lineal por lo que se deben usar algoritmos de optimización no lineales como el de Levenberg-Marquardt.

9.2.6. Resultados

El algoritmo se implementó en Matlab y actualmente se esta trabajando en la optimización del tiempo de ejecución para poder implementarlo de nuevo, en tiempo real, en algún dispositivo móvil. El algoritmo funciona adecuadamente para movimientos de rotación y traslaciones así como cambios de perspectiva, el drift de la señal es pequeño como se menciona en la literatura aunque aún falta implementar la optimización de la trayectoria.

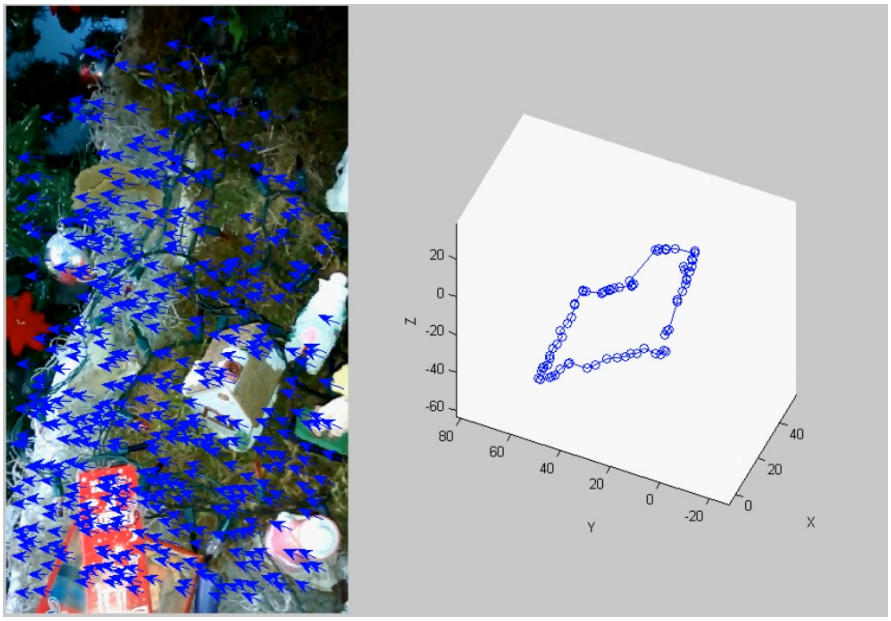


FIGURA 9.14: TRAYECTORIA OBTENIDA USANDO VO

9.3. Algoritmo de Fusión de Señales Inerciales y Visuales

La fusión de información proveniente de cámaras digitales y sensores inerciales se ha vuelto muy popular en el área de la Robótica con aplicaciones en navegación autónoma [Tkocz y Janscheck, 2014] y generación de mapas [Tanskanen y Kolev, 2013], debido al tamaño pequeño y peso de estos sensores, así como de su bajo consumo de energía, estos algoritmos son la opción preferida para ambientes en los que no se tiene acceso a señal de GPS, inclusive las implementaciones exitosas en robots Humanoides, MAV's [Anderse y Taylor, 2007] y UAV's son cada vez mas comunes.

La fusión de datos Visuales-Inerciales esta motivada por las propiedades complementarias de estos dos tipos de sensores [Martinelli, 2014]: La información visual produce buenas estimaciones en movimientos de baja velocidad pero su precisión se degrada en movimientos de alta velocidad debido a la difuminación producida, por otro lado los sensores inerciales ofrecen buenas estimaciones en movimientos de alta velocidad pero tienden a acumular drift devido a la integración del offset presente en los sensores. El algoritmo de fusión de datos seleccionado debe considerar estas deficiencias para producir una solución útil y precisa.

Hoy en día los teléfonos inteligentes comerciales están provistos de estos dos tipos de sensores lo que los convierte en plataformas atractivas para nuevas aplicaciones como seguimiento de personas y reconstrucción 3D. Por otro lado la poca memoria y las capacidades de procesamiento de estos dispositivos limita la complejidad de los algoritmos que se pueden implementar para su funcionamiento en línea.

En esta sección proponemos un algoritmo de odometría inercial asistido por información visual que puede ser implementada en una amplia variedad de dispositivos móviles como un robot humanoide, MAV o un teléfono inteligente. Por lo tanto consideraremos un arquitectura que incluya una cámara monocular y una central inercial (IMU) que cuente con un acelerómetro y un gyroscopo de tres ejes.

Se considera que la velocidad de muestreo de la IMU es alta en comparación con la velocidad de la cámara, de tal forma que sea posible obtener suficiente información para estimar el movimiento en el intervalo requerido para capturar dos imanes sucesivas.

9.3.1. Descripción del Algoritmo

El algoritmo propuesto estima el movimiento de un dispositivo de manera incremental usando las señales de la IMU y la información visual. De tal forma que la IMU es el sensor principal que nos permite estimar la pose del dispositivo mediante integración, mientras que la información visual es usada para refinar la estimación inicial y calcular de manera adaptable el offset de la IMU, como se muestra en la FIGURA 9.15.

Para cada iteración la entrada del algoritmo es un par de imágenes y las señales inerciales grabadas durante el correspondiente intervalo de tiempo transcurrido. La iteración comienza calculando el cambio de posición X y la pose θ del dispositivo.

La Matriz Fundamental F_0 se calcula utilizando este cambio de posición. Usando la restricción epipolar es posible definir una ventana de búsqueda para simplificar el proceso de encontrar las correspondencias $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$ lo que reduce el tiempo de cómputo y las falsas correspondencias. Opcionalmente se puede añadir el algoritmo de RANSAC para eliminar posible outliers.

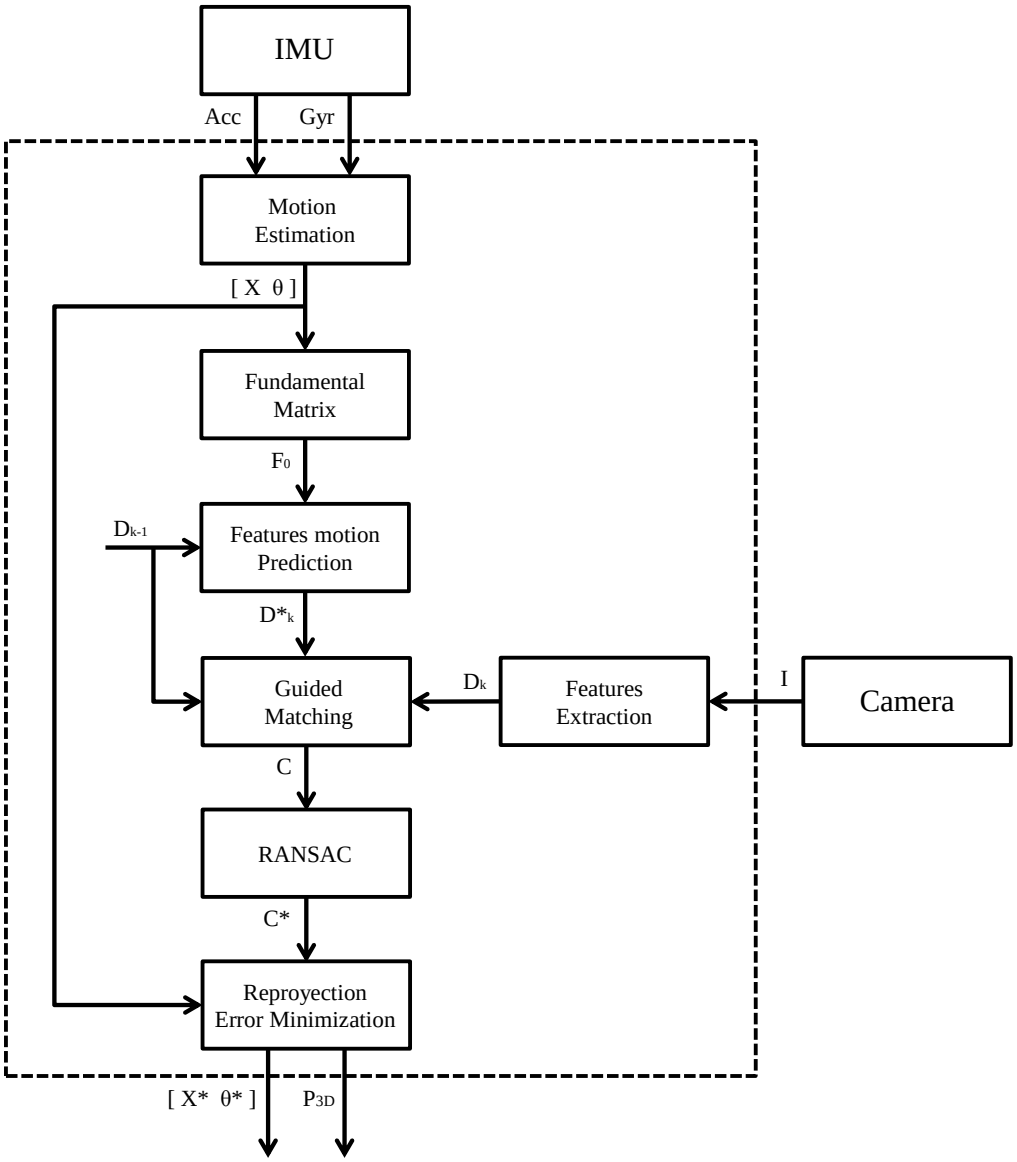


FIGURA 9.15: ALGORITMO DE FUSIÓN INERCIAL-VISUAL

La fusión sensorial se logra por medio de la minimización del error de reproyección, el cual es la distancia entre las coordenadas de cada característica de la imagen y la proyección de la posición 3D de la característica al plano imagen.

El error de reproyección puede describirse en función de la posición de las cámaras, entonces usando la estimación inicial [X θ] obtenida de las mediciones inerciales es posible obtener la estimación refinada [X^* θ^*]. Usando la diferencia entre estas estimaciones es posible estimar la desviación de las señales provenientes de la IMU para corregir los correspondientes offsets. La salida de cada iteración es la estimación refinada de la posición de la cámara y las características detectadas.

9.3.2. Etapa de Calibración

Dado que se seleccionó la IMU como sensor principal, es importante identificar los correspondientes valores de los offsets para obtener una aproximación robusta. Inicialmente estimaremos estos valores mediante una etapa de calibración. Se le pedirá al usuario de la aplicación que alinee la pantalla del dispositivo con respecto a cada uno de los tres ejes y que mantenga el dispositivo quieto por algunos segundos, como se ve en la FIGURA 9.16, los valores registrados durante esta etapa se utilizan para identificar el valor promedio de los offsets de cada una de las seis variables de la IMU.

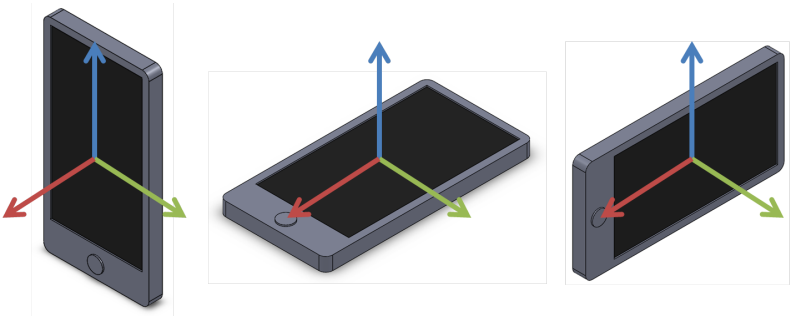


FIGURA 9.16: ETAPA DE CALIBRACIÓN

9.3.3. Estimación de Movimiento a partir de Señales Inerciales

Las señales inerciales generalmente están contaminadas por ruido η y offsets Δ que producen deriva cuando son integrados, representamos esta situación como:

$$Acc = \ddot{X} + \Delta Acc + \eta Acc + G$$
$$Gyr = \dot{\theta} + \Delta Gyr + \eta Gyr$$

La mayoría de los filtros actuales eliminan el ruido usando algún tipo de filtro, usualmente estos filtros usan la medición actual y los valores previamente calculados para calcular una medición filtrada para cada medición del sensor (filtros pasa bajas o filtro de Kalman).

En nuestra implementación el movimiento se calcula únicamente después de que un nuevo frame proveniente de la cámara esta disponible, para ese entonces es posible acceder a un vector de varias mediciones inerciales, toda esta información se puede usar para eliminar el ruido de la señal al mismo tiempo usando por ejemplo un algoritmo de ajuste de curvas.

El valor de los offsets es difícil de compensar en las mediciones de la IMU debido a que este depende del tiempo y la temperatura del sensor, cuando se usa un filtro de Kalman el offset puede ser estimado con cada iteración, pero únicamente en el caso de que se tenga disponible el modelo de la incertidumbre [Aksoy y Aydin, 2014]. En nuestra implementación el valor de los offsets se calculan utilizando el error entre la pose inicial y la refinada.

En el caso de las señales de los acelerómetros también es necesario compensar el vector de la gravedad G, este se puede estimar rotando el vector de la gravedad, en el referencial global, con respecto al referencial de la IMU, el cual se obtiene de la pose θ del dispositivo.

$$G = (R_x(\theta_1)R_y(\theta_2)R_z(\theta_3))^T g\hat{z}$$

Donde $R_i(\theta)$ representa la matriz de rotación en el eje i un ángulo θ, g es la magnitud de la gravedad y $\hat{z}$ es un vector unitario en la dirección del eje z. Después de estas consideraciones la posición y la orientación del dispositivo se puede obtener integrando las señales $\ddot{X}$ y $\dot{\theta}$

9.3.4. Refinación usando Información Visual

La Matriz Fundamental puede obtenerse por distintos métodos, por ejemplo de la manera mostrada en la sección 9.2.3, también puede obtenerse a partir del movimiento relativo de las cámaras, en este caso consideramos que la primer cámara esta ubicada en el origen de tal forma que la posición de la segunda cámara esta definida por X y la matriz de rotación corresponde a $R(\theta) = R_x(\theta_1)R_y(\theta_2)R_z(\theta_3)$, de esta forma la matriz fundamental esta dada por:

$$F = [X]_\times R(\theta)$$

Donde el operador $[v]_\times$ representa la matriz antisimétrica de 3x3 del vector v usada para representar el producto cruz como una multiplicación de matrices.

La relación entre las coordenadas homogéneas de un par de correspondencias $\{\mathbf{x} \leftrightarrow \mathbf{x}'\}$ y la matriz fundamental se describió en la ECUACIÓN 9.1. El término $F\mathbf{x}$ describe la linea epipolar sobre la cual el punto correspondiente $\mathbf{x}'$ debe estar, esta relación permite definir una ventana de búsqueda basad en la intersección de una ventana centrada en las coordenadas del punto $\mathbf{x}$ con radio r y la región formada por todos los puntos cuya distancia a la línea epipolar es $\leq d_min$. En esta región la correspondencia se hace en base a la similitud de sus descriptores.

Aunque el uso del algoritmo de correspondencias guiadas reduce la aparición de OutLiers debido a restricciones geometricas, es recomendable añadir el algoritmo RANSAC para eliminar falsas correspondencias provocadas por oclusión o cambios en luminosidad.

Es importante la correcta eliminación de outliers para que sea posible asumir que la diferencia entre la estimación inicial de la pose y la refinada se debe a errores en la estimación del offset.

Minimización del Error de Reproyección

El algoritmo usado para hacer el refinamiento de la pose, el cual se encarga de hacer la fusión sensorial esta basado en el algoritmo *Gold Standard* [Hartley y Zisserman, 2004] pero el error de proyección es minimizado con respecto a la posición relativa de las cámaras, de la siguiente manera:

Entrada: $[X_0 \ \theta_0]$, la aproximación inicial de la pose relativa de las cámaras
$\{\mathbf{x} \leftrightarrow \mathbf{x}'\}$ el vector de parejas de correspondencias
Salida: $[X^* \ \theta^*]$ la aproximación refinada de la pose relativa de las cámaras

1: Definir las matrices C =$[I|0]$ y C' =$[R(\theta)|X]$
2: Usar las correspondencias $\{\mathbf{x} \leftrightarrow \mathbf{x}'\}$ y las matrices C, C' para encontrar los puntos 3D P_i^{3D} por triangulación.
3: Calcular los puntos reproyectados: $\hat{x}_i = CP_i^{3D}$ y $\hat{x}'_i = C'P_i^{3D}$
4: Minimizar el error de reproyección $r_e = \sum_i d(x_i, \hat{x}_i)^2 + d(x'_i, \hat{x}'_i)^2$ usando el algoritmo de Levenberg-Marquat con respecto a $[X_k \ \theta_k]$

Algoritmo 5: FUSIÓN SENSORIAL

Las coordenadas 3D de las características P_i^{3D*} pueden ser usadas para crear un mapa del entorno visto durante la trayectoria recorrida, la corrección de la pose puede ser usada para refinar la estimación de la pose de la IMU de la siguiente manera:

$$\Delta Acc^k = \Delta Acc^{k-1} - \alpha(X^* - X_0)$$
$$\Delta Gyr^k = \Delta Gyr^{k-1} - \alpha(\theta^* - \theta_0)$$

Donde $\alpha > 0$ es una constante usada como ganancia de retroalimentación para regular la estimación del offset en el modelo de la IMU.

9.3.5. Resultados

El algoritmo propuesto fue implementado en un dispositivo Android comercial, se definió la velocidad de muestreo de la cámara a 1fps y la de la IMU a 20fps, el algoritmo fue programado usando hilos, lo cual permite que el algoritmo pueda grabar las señales de la IMU mientras procesa la información de la iteración anterior.

Los resultados obtenidos fueron obtenidos moviendo el dispositivo sobre una superficie texturizada, la gráfica muestra la trayectoria estimada cuyo error es menor a 5 % de la longitud total de la trayectoria. Estos resultados fueron obtenidos a una velocidad relativamente baja ya que la presencia de imágenes borrosas causa divergencia en el algoritmo.

El algoritmo propuesto muestra una alternativa novedosa en la fusión de información visual e inercial, aunque el algoritmo funciona bien es necesario optimizar la implementación para aumentar la velocidad de procesamiento, también es necesario detectar imágenes borrosas para desecharlas y evitar que el error se transfiera a la estimación de los offsets. Otra mejora sería agregar las mediciones de los magnetómetros para obtener una mejor estimación de la pose [Tian y Zhang, 2013]. También se esta trabajando en la construcción del mapa usando la pose estimada de los puntos 3D.

Bibliografía del Capítulo 9

Aksoy, Y. y A Aydin (2014). «Uncertainty modeling for efficient visual odometry via inertial sensor on mobile devices». En: *Proceedings of the International Conference on Image Processing*, págs. 3397-3401.

Anderse, E.D. y C.N. Taylor (2007). «Improving MAV Pose Estimation Using Visual Information». En: *Proceedings of the International Conference on Intelligent Robots and Systems*, págs. 3745-3750.

Hartley, R. y A. Zisserman (2004). *Multiple View Geometry in Computer Vision*. New York, USA: Cambridge University Press.

Martinelli, A. (2014). «Visual-inertial structure from motion: observability vs minimum number of sensors». En: *Proceedings of the International Conference on Robotics and Automation*, págs. 1020-1027.

O'Flaherty, R. y col. (2013). «Humanoid Robot Teleoperation for Tasks with Power Tools». En: *Proceedings of the 2013 IEEE International Conference on Technologies for Practical Robot Applications*.

Sian, N.E. y col. (2003). «Whole Body Teleoperation of a Humanoid Robot Integrating Operators Intention and Robots Autonomy An Experimental Verification». En: *Proceedings of 2003 lEEE/RSJ International Conference on Intelligent Robots and Systems*, págs. 1651-1656.

Tanskanen, P. y K. Kolev (2013). «Live Metric 3D Reconstruction on Mobile Phones». En: *Proceedings of the International Conference on Computer Vision*, págs. 65-72.

Tian, Y. y J. Zhang (2013). «Adaptive-Frame-Rate Monocular Vision and IMU Fusion for Robust Indoor Positioning». En: *Proceedings of the International Conference on Robotics and Automation*, págs. 2257-2262.

Tkocz, M. y K. Janscheck (2014). «Closed-Form Metric Velocity and Landmark Distance Determination utilizing Monocular Camera Images and IMU Data in the Presence of Gravity». En: *Proceedings of the International Conference on Automatic Robot Systems and Competitions*, págs. 8-13.

Capítulo 10

Conclusiones y trabajo futuro

El desarrollo del robot humanoide *Johnny*, como base de este tema de tesis, fue de gran utilidad para profundizar adecuadamente en la mayoría de las áreas que conforman la robótica, el hecho de que el alcance del proyecto considerará la construcción e implementación física de los conceptos estudiados permitió experimentar una gran cantidad de contrariedades que por lo general se pasan por alto en trabajos puramente teóricos.

La optimización del diseño mecánico de *Johnny* representó una tarea bastante complicada, las ecuaciones que rigen su comportamiento durante el caminado son bastante complicadas y no es posible determinar una expresión cerrada sobre su eficiencia la cual se pueda optimizar por métodos convencionales.

Se tuvo que recurrir a técnicas heurísticas de la computación evolutiva para realizar la tarea de optimización lo que involucró el profundizar en áreas ajenas al control automático.

Por otro lado lograr una estimación correcta de la cantidad de energía consumida durante el caminado involucró realizar un modelo bastante completo que pudiera representar con exactitud la mayor cantidad de fenómenos del mundo real, para que la solución obtenida cumpliera con las expectativas.

Al construir el modelo matemático del caminado se pudo confirmar que la formulación de Euler-Lagrange no es apropiada para su uso en simulaciones, cuando el número de articulaciones es muy grande, esto se debe a que el orden de complejidad de esta formulación es $O(n^4)$, por esta razón fue necesario replantear el modelo de forma recursiva con las ecuaciones de Newton-Euler el cual es de orden $O(n)$

La eficiencia energética de un robot humanoide depende de una gran cantidad de factores, se hizo un análisis profundo, por medio de estudio bibliográfico y experiencia personal, con la finalidad de considerar la mayor cantidad de factores involucrados.

Durante el mencionado análisis se encontró que el lazo de control articular, que se utiliza para lograr que los eslabones sigan las trayectorias de caminado, juega un papel muy importante, por lo que se estudiaron diferentes formas en las que este control se puede optimizar para favorecer el consumo energético.

La construcción del robot también representó todo un reto involucrando la selección de materiales y procesos de manufactura, las restricciones dadas por los elementos electrónicos usados y la capacidad de los actuadores, también se requirió de la capacidad de adaptarse a las necesidades cambiantes del proyecto.

El control de la estabilidad de un robot humanoide también representa una tarea bastante complicada, el simple hecho de definir un criterio de estabilidad apropiado es ya un tema que se ha investigado por varios años, de tal forma que se hizo necesario definir un criterio para el diseño de las trayectorias nominales de caminado (ciclos límite), y utilizar otro criterio para controlar el equilibrio ante perturbaciones externas (zmp).

Finalmente un proyecto de robótica tan ambicioso como este no estaría completo sin la implementación de aplicaciones que mostraran la utilidad y versatilidad de estos sistemas, por esta razón se construyó una plataforma de teleoperación que permite la manipulación de la pose de las extremidades del robot por distintos métodos y también se desarrollaron aplicaciones de visión artificial sobre odometría y fusión sensorial que pueden aplicarse en todo tipo de sistemas robóticos moviles.

Aunque se abordaron meticulosamente los temas principales de la robótica humanoide aún queda un largo camino por recorrer en el proyecto del robot *Johnny*, como trabajo futuro se propone la implementación de algoritmos avanzados de control, como por ejemplo: en la detección y corrección de perturbaciones, control del equilibrio utilizando modelos completos, cálculo de fuerzas y pares para generación de trayectorias con múltiples puntos de contacto, entre muchas tareas más.

Para compensar la degradación de la eficiencia energética debido a las modificaciones mecánicas también se propone, como trabajo futuro, el estudio e implementación de técnicas heurísticas de optimización para modificar los parámetros de caminado y las ganancias de los controladores en linea. De esta forma se podría encontrar un nuevo conjunto de parámetros óptimo que compense el peso extra de los sensores Optoforce en los pies del robot, la masa de la mochila y demas modificaciones al diseño original.