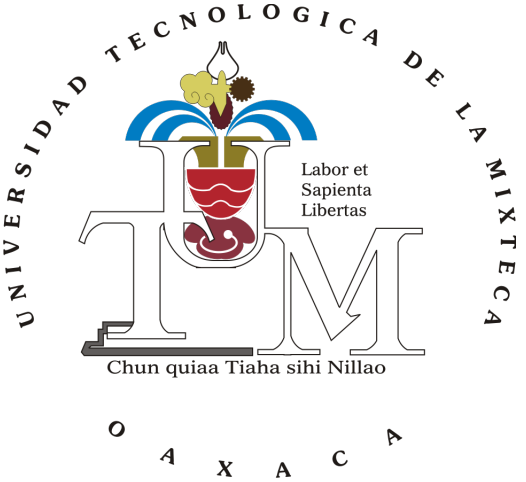
UNIVERSIDAD TECNOLÓGICA DE LA MIXTECA



" DESARROLLO DE UN ROBOT TIPO BALLBOT PARA APLICACIONES DE CONTROL "

TESIS

PARA OBTENER EL TÍTULO DE INGENIERO EN MECATRÓNICA

PRESENTA:

RAFAEL ADRIAN GARCÍA GARCÍA

DIRECTOR DE TESIS:

DR. MANUEL ARIAS MONTIEL

HUAJUAPAN DE LEÓN, OAXACA, MÉXICO, JULIO DEL 2014

Tesis presentada en Julio de 2014
ante los sinodales:

M.C. Fermín Hugo Ramírez Leyva
M.C. Jorge Luis Barahona Ávalos
Dr. Jesús Linares Flores

Director de tesis:

Dr. Manuel Arias Montiel

Este trabajo de tesis se lo dedico a mi familia, la cual siempre ha estado conmigo apoyándome y motivándome. Gracias por todos los sacrificios que han hecho para que este logro fuera posible.

A mi padre:
Rafael Genaro García Bohorquez

A mi madre:
María Vianey García Soriano

A mi hermano:
Jose Manuel García García

A mi hermana:
Andrea Catalina García García

Agradecimientos

Sin duda alguna han existido varias personas a lo largo de mi vida, las cuales me han permitido convivir con ellas e intercambiar distintos puntos de vista, así como me han enseñado diferentes lecciones de vida. Así también, siempre me han apoyado y motivado a ser una mejor persona, y nunca dejar de lado mis metas. Aunque no quisiera excluir a nadie, agradezco enormemente a las siguientes personas.

A mi padre, quien siempre nos ha enseñado a mis hermanos y a mí que todo sacrificio tiene su recompensa, ahora se que algunas decisiones que toma son para nuestro propio bien. Gracias por estar conmigo en todo momento, así como enseñarme que todo es posible, solo es cuestión de decidirse y hacer las cosas.

A mi madre, quien siempre está al pendiente de mi salud y seguridad, además de ser una mujer con mucha perseverancia y decisión. Gracias por apoyarme en todo y consentirme siempre, también por los sermones que siempre me das y que no siempre hago caso.

A mi hermano, con quien he vivido grandes aventuras (*travesuras*) las cuales de solo recordar me da un ataque de risa. Es agradable echar la reta contigo en *Gears of war* o *Call of duty* aunque aclaro que siempre te dejo ganar para que no te sientas mal. Sigue echándole ganas a la escuela que ya vas a la mitad de la carrera y por cierto ya no desarmes ni descompongas mis cosas.

A mi hermana, a quien he cuidado desde pequeña y me ha enseñado a ser responsable en algunas situaciones. Espero ser una buena guía en tu formación académica y disculpame por tantos zapes. Aprovecho también para felicitarte y decirte que le eches ganas ahora que decidiste estudiar en la UTM.

A mi abuelo Marcelo (*papá chelo*), con el cual paso tiempo escuchando sus historias y consejos mientras jugamos cartas o me invita unos tragos.

A mi abuela Catalina (*mamá cata*), quien siempre me recuerda como era de pequeño, me da su apoyo y consejos para ser una mejor persona, y a pesar de mi edad sigue consintiéndome.

A mi abuela Genoveva (*geno*), quien me ha apoyado desde siempre y ha estado conmigo en las clausuras, además de sus consejos siempre me ofrece de sus ya famosas memelitas o

empanadas especiales para mi.

A todos mis tíos, quienes me han apoyado en la mayoría de lo posible, me dan palabras de aliento y se los agradezco de todo corazón. En especial a mi tía Esther quien seguirá siendo como una segunda madre para mí, pues aún está conmigo.

A mis tíos y padrinos Hipólito y Juliana, quienes también han hecho lo posible por acompañarme en mi ceremonia de graduación y esta vez en el examen de titulación, muchas gracias.

A mis primos, con los cuales he pasado grandes momentos como: platicar, jugar, hacer travesuras. Aunque ahora ya cada quien tiene diferentes caminos, siempre que nos juntamos hacemos buenas fiestas.

A mi novia Saibet, a quien desde que conocí hace mas de 6 años siempre ha sido una buena amiga, compaẽra, confidente, y me ha alentado en todos mis proyectos, nunca dejándome desistir. Gracias por tu amor y soportarme tanto tiempo.

A todos mis compañeros, con los cuales he tenido grandes experiencias e intercambiado diferentes puntos de vista a lo largo de todos estos años. A mis compañeros de la universidad, tanto de mecatrónica como de las demás áreas con quienes conviví estos 6 años y creamos un lazo de amistad, grupos de trabajo y fuimos precursores de algunas leyendas y fiestas en Acatlima. A todos ustedes les deseo un gran futuro y mucho éxito en su vida, espero que en unos años podamos reunirnos nuevamente y recordar todos esos momentos.

A mi director de tesis el Dr. Manuel Arias Montiel, con quien desde que platiqué mi tema inicial de tesis me brindó ayuda e información para darle seguimiento, lo cual fue de gran utilidad al replantear el tema principal. Gracias por sus comentarios, observaciones, el apoyo y tiempo invertido para que este trabajo se pudiera finalizar. Así como alentarme para que fuera posible realizar una publicación y espero en un futuro volver a trabajar con usted.

A mis sinodales: el M.C. Hugo Leyva, gracias por aceptar colaborar en este proyecto y por sus observaciones para que el documento quedara lo más completo posible; al M.C. Jorge Barahona, gracias por apoyarme a pesar de la gran carga de trabajo que tenia y que ahora como jefe de la carrera de mecatrónica se esfuerza por realizar cambios necesarios para mejorar el equipamiento de los laboratorios y promocionar la oferta educativa. Aún así gracias por el tiempo en revisar y realizar las correcciones del documento; al Dr. Jesús Linares, gracias por sus comentarios los cuales sirvieron para ampliar el contenido de la tesis e incursionar en una área que no conocía pero resultó ser de gran ayuda, así como sus observaciones durante las pláticas y exposiciones.

A la Dra. Patricia Gallegos del instituto de minería, el cual parecia más un laboratorio de mecatrónica en periodo de proyectos finales. Gracias por todo su apoyo durante la carrera, y también por los dulces y el café que nos invitaba cuando pasábamos a saludarla o molestarla

con algún favor.

A todos los profesores de los cuales he aprendido que no solo basta con tener suficientes conocimientos, si no el saber como transmitir estos hacia los demás.

A la secretaría de Educación Pública, a través de la Coordinación Nacional de Becas de Educación Superior por el apoyo otorgado para la realización de esta tesis.

Por último, me queda decirles, gracias totales!...

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

Cuenta la historia que aproximadamente en 1581 Galileo Galilei se encontraba en la catedral de Pisa, cuando el movimiento de vaivén de una lámpara colgante captó su atención. Auxiliándose únicamente con su pulso, comenzó a medir el periodo de los movimientos de péndulo de la lámpara. Como resultado de sus mediciones, Galileo dedujo que el tiempo de oscilación era independiente de la amplitud de las oscilaciones, fenómeno llamado isocronismo [65, 67]. Esto lo llevó a realizar más experimentos similares. Es así como comienza el estudio del péndulo como sistema físico, dando origen a numerosos estudios teóricos y prácticos con aplicaciones dentro de la relojería, física, pedagogía, filosofía, tecnología, cultura y sociedad. A partir de este sistema físico, varias personalidades dentro de los campos de la física, matemáticas, filosofía, astronomía, fundamentaron las bases de sus investigaciones como lo son: Huygens y su propuesta de un nuevo estándar de longitud, la formulación de la fuerza centrífuga, así como la primer formulación del periodo de oscilación de un péndulo para un ángulo pequeño; Hooke y su ley del movimiento armónico simple; Newton y el apoyo que obtuvo mediante la formulación de Hooke y el uso del péndulo para demostrar su segunda y tercera ley en su obra "Principia"; Leibnitz, Bernouli, Euler y nuevamente Newton, Huygens y Hooke, quienes abordaron un nuevo problema conocido como la braquistrócona y el cual fue el origen de lo que hoy se conoce como cálculo variacional, problema que pretende encontrar los máximos y mínimos de una función ante determinados parámetros y en el que posteriormente muchos más investigadores contribuyeron; Foucault, quien diseñó un péndulo para demostrar la rotación de la tierra, etc. Es así como el péndulo fue un elemento crucial dentro de la mecánica clásica para el establecimiento de las leyes de colisión, conservación, determinación del valor de la aceleración de la gravedad, y asentar la física de Newton [50]. Algunas aplicaciones industriales del péndulo son: el sistema gobernador de vapor de Watt utilizado para regular los motores de vapor, la implementación de teleféricos, manipuladores industriales y las plataformas utilizadas para la recuperación en alta mar de petróleo y gas [5].

Los principios físicos por los que se rige el movimiento de un péndulo simple, constituyen una herramienta importante para lo que se conoce como alfabetización científica, que busca hacer que el estudiante entienda la relación que tiene la ciencia con las matemáticas, la cultura y su contexto social. Para lograr el objetivo anterior, se requiere que el alumno estudie, comprenda e implemente los pasos del método científico. En los últimos cincuenta años han

aparecido diversos artículos relacionados con el péndulo en las principales revistas científicas, en las cuales se muestran las propiedades y campos de aplicación de un péndulo, así como sus diversas variantes entre las cuales destacan: el péndulo de Kater, el péndulo cicloidal, el péndulo elástico, el péndulo bifilar, el péndulo blackwood, el doble péndulo, el péndulo de torsión, y el péndulo invertido. El péndulo tambien sirve de base para modelar otros sistemas, como por ejemplo, la caminata y el balanceo; ambos han sido extensamente discutidos en la literatura y son un ejemplo del uso del péndulo y el entendimiento de sus propiedades como lo son: el periodo natural, la energía cinética, la energía potencial, la conservación de momentos y el movimiento armónico simple [24].

Una variante del péndulo conocida como el péndulo invertido, constituye un sistema de amplio estudio en diversas áreas de investigación. Dentro del área de control automático dicho sistema adquirió gran interés debido a su naturaleza no lineal e inestable, así como a la facilidad para introducir perturbaciones en dicho sistema. Otro factor importante es el hecho de que este sistema se asemeja al comportamiento de otros fenómenos físicos complejos y que se pueden modelar de la misma manera. Así pues, se han desarrollado diversas leyes de control para estabilizar al sistema, ya sea abordándolo desde su forma no lineal o linealizándolo alrededor de un punto de operación [27].

En la actualidad existen diversas configuraciones del péndulo invertido, por ejemplo: el péndulo invertido sobre un carro, donde la restricción de movimiento de ambos se da en un solo plano; el péndulo invertido sobre dos ruedas, sistema que tiene la capacidad de girar sobre un punto sin desplazarse ofreciéndole mayor maniobrabilidad; el péndulo esférico invertido sobre un móvil omnidireccional en el plano $x - y$ que controla el giro libre del péndulo sobre la base del móvil; y el péndulo invertido sobre una esfera, donde el péndulo hace contacto con la esfera mediante unas ruedas de control y la esfera a su vez directamente con la superficie, lo que le permite equilibrarse en un menor espacio.

Estos modelos del péndulo invertido se han implementado en diversas áreas, tanto militares, sociales y de la industria privada, como lo son: el control de cohetes; el modelo del control de posición de un propulsor primario espacial para despegues; los robots bípedos; el modelado del movimiento de los pies del ser humano; los vehículos terrestres; dispositivos de transporte personal; los sistemas antisísmicos en construcciones; o bien; los sistemas de absorción de vibraciones mecánicas [27].

Una robopelota o ballbot (péndulo invertido sobre una esfera) es un robot móvil basado en un equilibrio activo y por lo tanto es “dinámicamente estable”, es decir, que solo se mantendrá en posición vertical si se realizan correcciones continuas en su postura. El ballbot mantiene su equilibrio sobre una sola rueda esférica (es decir, una bola). A través de su único punto de contacto con el suelo, el ballbot es omnidireccional y por lo tanto excepcionalmente ágil, maniobrable y natural en movimiento en comparación con otros vehículos de tierra. Su estabilidad dinámica permite mejorar la navegabilidad en ambientes estrechos, llenos de gente y dinámicos. El ballbot funciona sobre el mismo principio que el de un péndulo invertido [33].

Una de las plataformas experimentales que se han usado para el estudio del ballbot es el kit de *LEGO Mindstorms NXT®*. Este kit es ampliamente utilizado en diversas áreas de la educación y la investigación, abarcando una serie de aplicaciones, que van desde demostraciones de sobremesa a la investigación de posgrado, laboratorios y proyectos. Algunas de estas áreas son inteligencia artificial, sistemas embebidos, sistemas de control, robótica y sistemas operativos. El kit de *LEGO Mindstorms NXT®* es de bajo costo comparado con otros kits de robótica, de fácil reconfiguración, reprogramable, polivalente (puede ser usado con distintos fines) y robusto, lo que lo hace muy adecuado para su uso en la enseñanza e investigación [9, 13, 25, 38, 39, 63, 66, 78].

El ballbot al poder ser implementado en un kit de *LEGO Mindstorms NXT®*, permite analizar su comportamiento, comparar resultados reportados en la literatura y realizar mejoras tanto en su estructura, como en las técnicas de control y navegación aplicadas.

1.1. Antecedentes

1.1.1. Péndulo invertido

El péndulo invertido, es un sistema bastante conocido dentro del área de ingeniería, esto debido a su naturaleza no lineal, inestable, fácilmente perturbable y el problema que lleva consigo el lograr estabilizarlo en su punto de equilibrio no estable. Además, la amplia aplicación de la tecnología que se deriva de este sistema inestable ha despertado el interés de varios investigadores en las áreas de control y robótica en todo el mundo.

En años recientes, los investigadores han aplicado la similitud del modelo dinámico de un péndulo invertido móvil a diversos problemas como el diseño de robots humanoides caminantes, sillas de ruedas robóticas y los sistemas personales de transporte. Por ejemplo: el segway; que se utiliza con fines comerciales y el prototipo del proyecto VECNA B.E.A.R; robot que se puede utilizar para operaciones militares [17]. Dichos proyectos han propiciado en los últimos años que los robots humanoides sobre ruedas ganen popularidad a nivel comercial y gubernamental, debido a su gran movilidad. Así también, el problema de control se puede simular y aplicar en un salón de clases para enseñar a los estudiantes la necesidad de controlar los movimientos de un robot inestable sobre ruedas así como las propiedades y operaciones fundamentales empleadas en robótica móvil.

Entre los trabajos reportados en la literatura existente, y que están relacionados con el péndulo invertido, pueden citarse los siguientes: el péndulo invertido sobre un carro[3, 26], el péndulo con disco inercial[8, 71], el péndulo esférico invertido [28, 48, 79], el péndulo de Furuta[4, 23], el pendubot[7, 21, 70], el acrobot [10, 14, 69], el sistema bola-viga [2, 29, 59], el sistema de despegue y aterrizaje vertical de un avión (VTOL, *Vertical Take-Off and Landing*, por sus siglas en inglés) [30, 49], y el sistema oscilador traslacional con actuador rotacional (TORA, *Translational Oscillator with Rotational Actuator*, por sus siglas en inglés) [37, 51, 75]. Dentro del área de control automático el péndulo invertido ha mantenido durante al menos

cincuenta años su utilidad para ilustrar casi todas las técnicas que han surgido en esta disciplina y que se han reportado en diferentes medios de divulgación como se muestra en la Tabla 1.1.

Tabla 1.1: Técnicas de diseño de control ilustradas por el péndulo invertido. Tabla tomada de [11].

Técnica de Control	**Libros**	**Recopilaciones**	**Artículos de Investigación**
Control Bang-Bang		•	•
Control con Lógica Difusa	•	•	•
Control con Redes Neuronales	•	•	•
Control PID Adaptativo	•	•	•
Control Robusto	•	•	•
Control Basado en Energía	•	•	•
Control Híbrido	•	•	•
Control por Modos Deslizantes	•	•	•
Control Óptimo en el Tiempo	•	•	•
Control Predictivo	•	•	•
Control de Perturbación Singular	•	•	•
Control Feed-forward		•	•

En años recientes una nueva configuración del péndulo invertido ha logrado acaparar la atención de la comunidad científica. Se trata de una configuración de péndulo invertido conocida como ballbot, un robot que se adapta a diferentes entornos, ademas de que posee características que lo hacen robusto. Existen líneas de investigación relacionadas con este tipo de robot en las cuales se busca mejorar la estructura mecánica del robot y los esquemas de control aplicados al mismo, dejando abierta una amplia gama de aplicaciones.

1.1.2. Robots tipo ballbot

Origen ballbot

En la Universidad de Carnegie Mellon bajo la dirección del doctor Ralph Hollis y los ingenieros George Kantor y Umashankar Nagarajan, se desarrolló un robot equilibrista capaz de balancearse sobre una bola de boliche, como se muestra en la Figura 1.1, al que llamaron "ballbot" [33]. Estos investigadores consideran que para lograr un desempeño productivo de los robots, dentro de la vida cotidiana de los seres humanos, estos deben de contar con una estructura robusta. Los autores señalan que, la altura suficiente de los robots para interactuar eficazmente en entornos humanos tiene un centro de gravedad alto y se debe acelerar y desacelerar lentamente, así como evitar las rampas empinadas, para evitar que se caiga. Para contrarrestar este problema, los robots estáticamente estables (robots que viajan sobre bases de apoyo de tres o cuatro ruedas y que mantiene a los robots verticales incluso

en reposo) tienden a tener cuerpos grandes con distancias anchas entre ejes, lo que limita en gran medida su movilidad a través de las puertas y alrededor de los muebles o las personas [33].

El tamaño de los robots en última instancia, afecta su movilidad. Con el fin de resolver dicho problema, al doctor Hollis se le ocurrió un nuevo diseño que mejora la estructura global del robot así como su movilidad. Hollis y su equipo construyeron un robot de cinco pies de alto, ágil y delgado. El objetivo del robot es equilibrarse en sí mismo sobre la parte superior de una rueda esférica. Hollis compara el diseño de la estructura de su robot con la de un bolígrafo gigante o un payaso de circo tratando de equilibrarse en lo alto de una pelota. El ballbot funciona sobre el mismo principio que el de un péndulo invertido [33]. El verdadero reto de Hollis fue mantener el robot autobalanceado en una posición vertical estable, puesto que el sistema en lazo abierto no es estable. Hollis encontró que la mejor manera de resolver este problema era la implementación dentro de su diseño de nuevos sensores avanzados y algoritmos de control óptimos. Los sensores que se incorporaron incluyen un giroscopio y un acelerómetro. Hollis implementó un algoritmo de control denominado "regulador lineal cuadrático" (LQR, *Linear Quadratic Regulator*, por sus siglas en inglés), técnica con la cual logró mantener el ballbot en un estado vertical estable en lazo cerrado. El LQR se basa en la teoría de control óptimo. El objetivo principal al utilizar técnicas de control óptimo en el sistema es minimizar el esfuerzo para estabilizar el ballbot en la posición vertical.

Figura 1.1: Proyecto ballbot junto a su creador, Ralph Hollis de la Universidad de Carnegie Mellon (CMU). Imagen tomada de http://www.msl.ri.cmu.edu/projects/ballbot/ .

La estructura del ballbot utiliza cuatros rodillos para actuar sobre la esfera. Además posee una unidad de medición inercial que aporta información sobre los ángulos y la velocidad de inclinación, como se puede apreciar en la Figura 1.2. Utiliza un mecanismo de viraje para la rotación alrededor del eje vertical y tiene instaladas tres extremidades para que tenga estabilidad estática cuando se apague. Su sistema de control está dividido en varias etapas: control de

estabilidad, control de estacionamiento, control de trayectorias, control contra perturbaciones intensas. Además, gracias a su sistema de tres extremidades es capaz de moverse sólo desde el arranque hasta el apagado sin intervención externa. El sistema contra perturbaciones es capaz de determinar, en función de la intensidad de la perturbación, la intención de la persona u objeto que la ha producido y actuar en consecuencia. Una de las debilidades de este robot es la incapacidad que tiene de subir por escaleras.

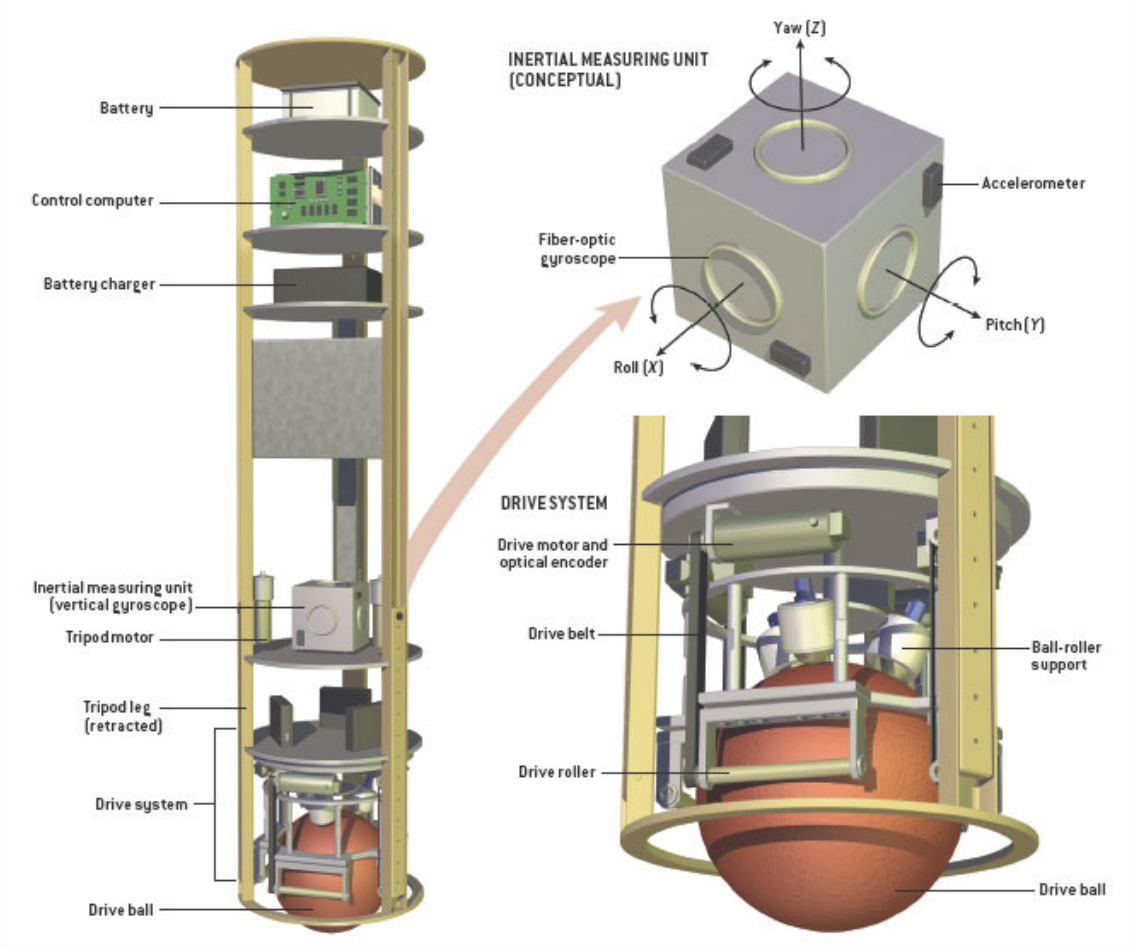


Figura 1.2: Estructura del ballbot (CMU), con sus componentes principales. Imagen tomada de [33].

Proyectos ballbot

Aunque varios robots tipo ballbot se han desarrollado en los últimos años, ninguno de ellos ha sido capaz de utilizar todo el potencial de agilidad que caracteriza a un sistema ballbot.

Como se ha mencionado, el primer ballbot fue desarrollado en el año 2006, en la Universidad de Carnegie Mellon (CMU) en los EE.UU (Figura 1.1) [33]. Otro ballbot denominado BALLIP (Ball Inverted Pendulum) fue desarrollado por el Dr. Masaaki Kumagai, director del

departamento de robótica de la Universidad de Tohoku Gakuin (TGU) en Japón, el cual se aprecia en la Figura 1.3 [42]. La estrategia elegida en este proyecto es utilizar tres ruedas omnidireccionales en lugar de rodillos para actuar sobre la esfera, tal como se observa en la Figura 1.4. Entre sus funciones están volver al punto donde se encontraba antes de una perturbación, control del robot con un dedo y diseño robusto frente a variaciones de peso. Posteriormente, un tercer prototipo surgió como un proyecto estudiantil en 2009 en la Universidad de Adelaida (UA) en Australia, éste se muestra en la Figura 1.5 [22].

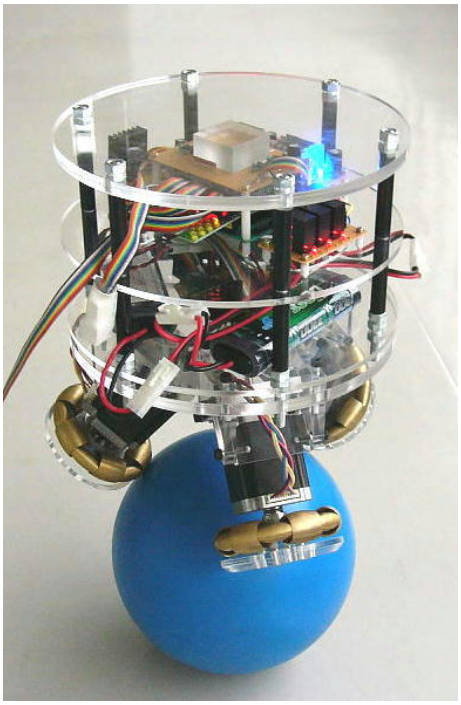


Figura 1.3: Ballbot desarrollado por Masaaki Kumagai y Takaya Ochiai en la Universidad de Tohoku Gakuin. Imagen tomada de [42].

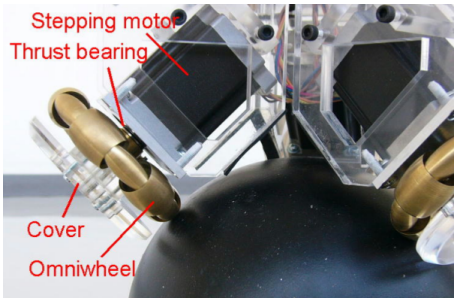


Figura 1.4: Uso de ruedas omnidireccionales en lugar de rodillos en el ballbot (TGU). Imagen tomada de [42].

La segunda versión del ballbot (CMU) se estabilizó con un controlador PID (proporcional integral derivativo) en cascada. El ballbot TGU utiliza un controlador PD (proporcional

Figura 1.5: Ballbot desarrollado por Justin Fong y Simon Uppill en la Universidad de Adelaida. Imagen tomada de http://sites.mecheng.adelaide.edu.au/robotics/robotics_projects.php?wpage_id=44&title=61&browsebyyear=2009.

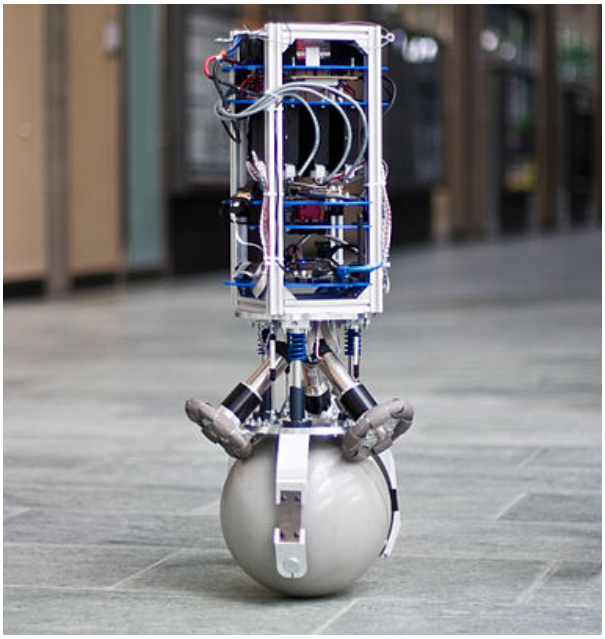


Figura 1.6: Proyecto Rezero, creado por estudiantes de la ETH Zurich. Imagen tomada de [74].

derivativo). El ballbot UA se controla con un controlador LQR que utiliza una parte integral adicional. Es importante mencionar que todos estos proyectos desarrollaron un controlador basado en un modelo planar. Se pueden apreciar las características de los ballbots antes mencionados en la Tabla 1.2.

Tabla 1.2: Especificaciones de ballbots existentes. Tabla tomada de [20].

Universidad	**CMU**	**TGU**	**UA**
Año	2006-2010	2006-2008	2009
Peso	45 Kg	11 Kg	34.5 Kg
Altura	150 cm	50 cm	160 cm
Máximo ángulo de inclinación	$<1°$	$<5°$	$<5°$
Máxima velocidad	<0.1 m/s	<0.3 m/s	<0.1 m/s
Rotación alrededor de su propio eje	si	si	no
Controlador implementado	LQR/PID	PD	LQRPI

Posteriormente, alumnos de la Universidad ETH Zurich desarrollaron el proyecto "Rezero", el primer ballbot que no sólo se centra en la estabilidad y el balanceo suave, sino que también apunta a una gran agilidad y alta aceleración, moviéndose de manera natural, como se observa en la Figura 1.6. Entre las múltiples funciones que posee este robot se pueden mencionar: un control de estabilidad y trayectoria, un sistema de seguimiento por láser, orbitación alrededor de un objeto y control remoto del robot. El desarrollo de líneas de investigación en torno a los robots tipo ballbot, apunta así a la producción de robots guías, de transporte, fotógrafos, juguetes y otros servicios limitados a la imaginación [74].

Con el desarrollo de estos prototipos y la constante investigación sobre temas relacionados, han surgido diversas publicaciones en el campo de control automático y robótica móvil, véase [1, 6, 20, 22, 41, 44, 45, 48, 52, 53, 55, 60, 68, 76, 79, 80].

1.1.3. LEGO Mindstorms NXT®

Lego Mindstorms NXT® es un kit con elementos básicos de robótica, que va desde la unión de piezas a la programación de acciones de una forma interactiva y orientada para niños. Este kit es fabricado por la empresa *Lego®*. El kit NXT de la Figura 1.7 es una versión mejorada a partir de *Lego Mindstorms RCX®*, que generalmente se considera la predecesora y precursora de los bloques programables de *Lego®*.

El componente principal o ladrillo inteligente contiene: un microcontrolador principal ARM7 (AT91SAM7S256) de 32 bits a 48 MHz, que incluye 256 Kb de memoria Flash y 64 Kb de RAM externa, encargado de ejecutar el software; un microcontrolador para los sensores y motores AVR (ATmega48) de 8 bits a 8 MHz, con 4 Kb de memoria Flash y 512 bytes de RAM. El ladrillo NXT dispone de cuatro entradas para los sensores y tres salidas de energía para los motores como se muestra en la Figura 1.8. Los sensores que trae de fábrica son: sensor de

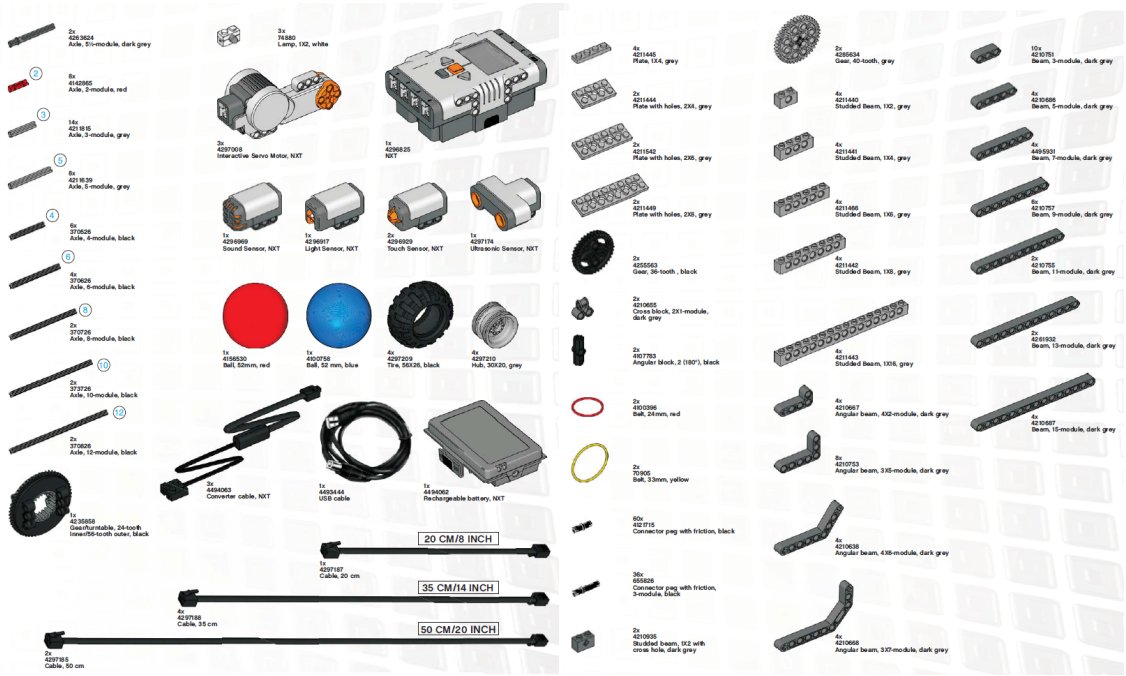


Figura 1.7: Elementos del kit *LEGO Mindstorms NXT®*. Imagen tomada de http://cache.lego.com/downloads/education/9797_LME_UserGuide_US_low.pdf.

contacto, sensor de luz, sensor ultrasónico y sensor de sonido. Existen también otros sensores y accesorios no oficiales como los sensores de giro, sensores de aceleración, sensor de temperatura, sensor de fuerza, ruedas omnidireccionales, pelotas electrónicas, bloques *wi-Fi*, entre otros, los cuales son distribuidos por fabricantes como : *HiTechnic*, *Dexter Industries* y *Mindsensors*.

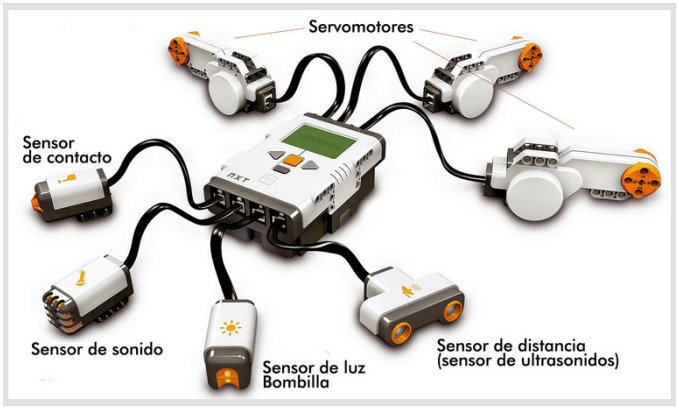


Figura 1.8: Ladrillo inteligente NXT con sus respectivos sensores y motores. Imagen tomada de http://legomindstorms.es/.

El ladrillo NXT puede comunicarse con otros bloques NXT, computadoras, palms, teléfonos móviles y otros aparatos con conectividad bluetooth ya que posee una interfaz *bluetooth*

compatible con la clase II V2.0. Así como una conexión directa con la computadora mediante la interfaz de *USB* que posee. Además, el ladrillo contiene un display LCD (100x64 pixeles), 4 botones para navegar en la interfaz de usuario, una bocina interna y su alimentación es mediante 6 pilas AA o una batería recargable. Una consulta mas extensa acerca del funcionamiento del kit *LEGO Mindstorms NXT®* se encuentra disponible en [35, 46].

Respecto al software, *Lego®* proporciona de fábrica un lenguaje de programación gráfico muy sencillo basado en *National Instruments LabVIEW®* denominado NXT-G, como se observa en la Figura 1.9, esto debido a que originalmente el kit está orientado hacia niños. Gracias a que el firmware es de código abierto y *Lego®* proporciona la documentación necesaria para desarrollar software y hardware, se han desarrollado diversas herramientas computacionales para la programación de las acciones del ladrillo. Algunos lenguajes y entornos de programación que admite son: $C^{\#}$, *Robolab*, *RobotC®*, *leJOS NXJ*, *Matlab-Simulink®*, *LabVIEW®* y *nxtOSEk*.

Un diagrama a bloques de la estructura principal del ladrillo NXT se puede observar en la Figura 1.10.

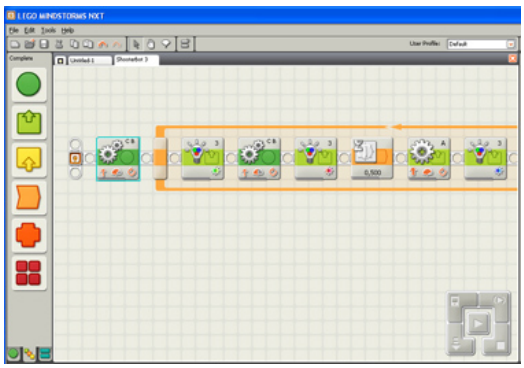


Figura 1.9: Entorno de programación NXT-G. Imagen tomada de `http://mindstorms.lego.com/en-us/history/default.aspx`.

El uso del kit de *LEGO Mindstorms NXT®* es altamente aceptado dentro de la comunidad de investigadores, tanto para realizar demostraciones en la docencia, como para abordar nuevas líneas de investigación, esto debido a que el kit es económico, reprogramable, polivalente y robusto. Los siguientes trabajos son una muestra de las aplicaciones a nivel de investigación que han usado el Lego Mindstorm como plataforma experimental [9, 13, 25, 38, 39, 63, 66, 78].

LEGO NXT ballbot

El NXT ballbot es una versión diseñada por Yorihisa Yamamoto [78] sobre un kit de *LEGO Mindstorms NXT®* del ballbot desarrollado por Ralph Hollis de la Universidad de Carnegie Mellon. El NXT ballbot mantiene su equilibrio sobre una rueda esférica mediante dos sensores giroscópicos que proporcionan la velocidad angular para medir la inclinación y

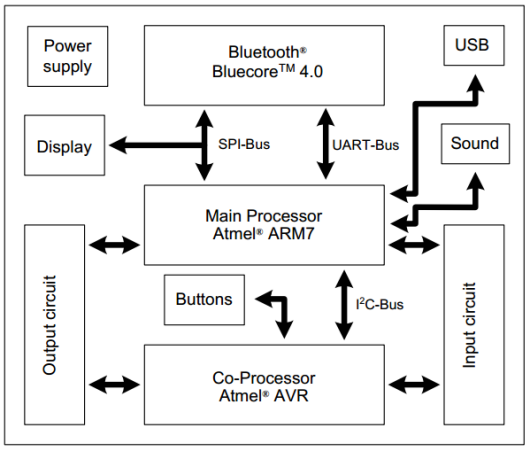


Figura 1.10: Diagrama a bloques de la estructura principal del ladrillo NXT. Imagen tomada de http://zeus.nyf.hu/~simona/NXT_Hardware.pdf.

dos motores que se comportan como actuadores, como se puede apreciar en la Figura 1.11. La construcción estándar utiliza piezas de *Lego*® que se incluyen dentro de la caja del kit *LEGO Mindstorms NXT*® educativo o versión básica, a excepción de los giroscopios. El kit de *Lego*® en sí mismo se utiliza como un banco de pruebas para calibrar los sensores, actuadores y realizar la estimación de varios parámetros desconocidos, tales como el momento de inercia de los motores y sus coeficientes de fricción [66].

Figura 1.11: Prototipo NXT ballbot desarrollado por Yorihisa Yamamoto y basado en el ballbot de Ralph Hollis. Imagen tomada de [78].

1.2. Planteamiento del problema

El uso de robots inteligentes que interactúen con el entorno y ayuden en las actividades diarias se ha convertido en una realidad, pero aún faltan resolver varios problemas fundamentales en cuanto a la percepción robótica, modelado de su entorno, razonamiento automatizado, la manipulación de objetos y su locomoción. Es así como ingenieros e investigadores se han dado a la tarea de desarrollar nuevas técnicas de control, programación, y navegación, con el fin de resolver dichos problemas y construir un robot lo más completo posible para interactuar con los humanos de manera natural en un ambiente cotidiano.

Con base en estas necesidades y en la revisión realizada en la literatura del problema de estabilización de un robot dinámicamente estable en movimiento, en este proyecto se pretende abordar el tema de modelado, construcción y control de un péndulo esférico invertido, basado en el diseño del robot tipo LEGO NXT ballbot. Para llevar a cabo dicho proyecto, será necesario desarrollar el modelo dinámico del sistema, la caracterización de los sensores y actuadores a emplear, diseñar e implementar diversas leyes de control a fin de realizar una comparativa en términos de su desempeño, así como desarrollar una interfaz mediante el uso de algún paquete de programación como *Labview®*, *Matlab-Simulink®* o *LeJOS NXJ*.

1.3. Justificación

El control del equilibrio durante el movimiento de los robots móviles con estabilidad dinámica en lazo cerrado es muy importante dentro del campo de la robótica móvil y el control automático. Debido a esto han surgido diversas leyes de control las cuales pretenden optimizar los recursos con los cuales se logre estabilizar un sistema, además de ser lo suficientemente rápidas y confiables en su desempeño. El problema radica en que no existe una metodología única para sintonizar dichos controladores, por lo que se comparará la técnica del regulador lineal cuadrático (LQR) propuesto en la literatura revisada y se implementará dicha técnica con ayuda de una herramienta que mejore la sintonización del controlador, como lo es la técnica de incertidumbre acotada en los datos (BDU, *Bounded Data Uncertainties*, por sus siglas en inglés) o desigualdades matriciales lineales (LMI, *Linear Matrix Inequalities*, por sus siglas en inglés). Posteriormente se pretende emplear el kit de *LEGO Mindstorms NXT®* bajo el modelo de un LEGO NXT ballbot para implementar dichos controladores.

Al final del proyecto se espera contar con un sistema el cual tenga un comportamiento óptimo en su desempeño comparado con las técnicas tradicionales del control automático, que sirva a estudiantes y profesores para llevar a la práctica conceptos teóricos de ingeniería de control y robótica móvil de una manera fácil e interactiva, empleando una plataforma accesible y polivalente como lo es el kit *LEGO Mindstorms NXT®*.

Al ser la carrera de mecatrónica un área multidisciplinaria, en este proyecto se pretende poner en práctica conocimientos adquiridos en materias como: control automático, robótica, sistemas digitales, programación, diseño de elementos de máquinas, diseño asistido por com-

putadora, etc., considerando que en dichas materias, el presente trabajo se podría mostrar como un ejemplo práctico de la aplicación de los conocimientos teóricos.

1.4. Hipótesis

- El controlador a diseñar para el sistema completo en 3D se puede lograr mediante el diseño de controladores independientes para los dos sistemas de planos separados e idénticos (sagital y coronal).
- La técnica mediante incertidumbre acotada en los datos (BDU) o desigualdad matricial lineal (LMI) para sintonizar el regulador lineal cuadrático (LQR) dará como resultado un controlador con mejor desempeño en términos de robustez que el reportado en la literatura.
- Un controlador mediante la técnica del LQR tendrá un mejor desempeño del error en estado estable, así como una demanda en el esfuerzo de control de manera óptima, en contraste a un controlador mediante asignación de polos.
- Una vez simulado el comportamiento de dichos controladores, estos podrán ser implementados en el LEGO NXT ballbot.

1.5. Objetivos

1.5.1. Objetivo General

Modelar, construir y controlar un péndulo invertido esférico basado en el prototipo LEGO NXT ballbot, comparar el controlador tipo regulador lineal cuadrático (LQR) propuesto en la literatura con un controlador clásico como el de asignación de polos en lazo cerrado y mejorar el desempeño del LQR mediante técnicas de sintonización. Posteriormente implementar dichos controladores en el LEGO NXT ballbot.

1.5.2. Objetivos Específicos

El problema general se puede agrupar en los siguientes objetivos específicos, que son:

- Desarrollar el modelo dinámico del sistema mediante la formulación de Euler-Lagrange y linealizarlo alrededor de su punto de equilibrio para obtener las ecuaciones de estado de forma lineal.
- Realizar el ensamblado físico y virtual (*Solidworks*®) del prototipo LEGO NXT ballbot, así como realizar pruebas experimentales para identificar los parámetros físicos de los motores. Esto para estimar los parámetros físicos requeridos en el modelado del sistema, comparándolos con los reportados en la literatura.

- Realizar simulaciones numéricas del algoritmo de control tipo regulador lineal cuadrático (LQR) reportado en la literatura para el LEGO NXT ballbot y diseñar dicho algoritmo con alguna técnica de sintonización que permitan mejorar el desempeño en su estabilización, así como diseñar un controlador clásico mediante asignación de polos. Realizar comparaciones entre dichos controladores.

- Implementar los controladores diseñados en el LEGO NXT ballbot, desarrollando una interfaz gráfica para monitorear los procesos.

- Documentar todos los pasos elaborados y publicar los resultados obtenidos en congresos, publicaciones, talleres, etc.

1.6. Metodología de Desarrollo

Para el desarrollo del presente proyecto se propone la siguiente metodología:

- Instalación de todos los paquetes de programación y complementos necesarios para utilizar de forma adecuada el kit de *LEGO Mindstorms NXT®* y su interacción con otros paquetes de programación como: *Matlab-Simulink®* y *JAVA®*. Se revisará la documentación existente y se asignará una computadora del laboratorio de mecatrónica para dichas pruebas.

- Desarrollar el modelo matemático mediante la metodología de Euler-Lagrange y linealizarlo alrededor de su punto de equilibrio.

- Ensamblar las piezas del kit de *LEGO Mindstorms NXT®* para construir el prototipo ballbot de Yorihisa Yamamoto [78].

- Ensamblar las piezas de *LEGO®* en un software de CAD, en este caso *Solidworks®*, para determinar los parámetros físicos del sistema (estructura), para ello, se empleará una librería existente de las piezas de *LEGO®* proporcionada por la Universidad de Carnegie Mellon, las piezas faltantes se modelarán en *Solidworks®* para proceder a realizar el ensamble de la estructura del LEGO NXT ballbot.

- Determinar los parámetros físicos internos de los actuadores, mediante el análisis de su diagrama eléctrico y estimaciones experimentales aplicadas para un motor de DC común. Para ello se muestra un banco de pruebas con el *LEGO Mindstorms NXT®* en [66].

- Realizar la caracterización y calibración de los sensores giroscópicos mediante un algoritmo de estimación de sesgo en línea.

- Realizar las simulaciones numéricas del controlador propuesto en la literatura, diseñar nuevamente el mismo controlador empleando técnicas de sintonización para mejorar el desempeño del controlador, así como diseñar un controlador clásico mediante asignación de polos.

- Comparar los resultados de los diversos controladores.
- Realizar una interfaz para monitorear los procesos durante la implementación física de los algoritmos de control sobre el LEGO NXT ballbot.
- Realizar pruebas finales y redactar el documento final.

1.7. Estructura de la Tesis

Este trabajo de tesis está organizado de la siguiente manera: en el Capítulo 1 se presenta una introducción y antecedentes al tema que se plantea; se establecen el planteamiento y la justificación del problema a abordar, se especifican los objetivos a cumplir a fín de resolver las hipótesis propuestas, y por último se describe la metodología a seguir durante el desarrollo del trabajo. En el Capítulo 2 se resume la teoría que da respaldo al diseño de controladores óptimos y robustos, y que se utilizará en el diseño e implementación de los esquemas de control a utilizar en el NXT ballbot. En el Capítulo 3 se presenta un análisis de los elementos que conforman la estructura del sistema LEGO NXT ballbot, posteriormente, se aborda la estimación de parámetros del sistema que se emplean al final del capítulo para obtener el modelo dinámico del sistema NXT ballbot mediante la metodología de Euler-Lagrange, finalmente se desarrolla la linealización del modelo alrededor de un punto de equilibrio, y se establece su representación en un modelo politópico para incluir incertidumbres en los parámetros. En el Capítulo 4 se describe el procedimiento para obtener las ganancias de diferentes controladores propuestos, y se realiza el análisis temporal de la respuesta del sistema en lazo cerrado con las ganancias obtenidas, a fin de comprobar que cumplan con la estabilización del sistema NXT ballbot. El Capítulo 5 presenta los resultados en simulación de los controladores en el modelo del sistema NXT ballbot implementado en *Simulink*®. En el Capítulo 6 se muestran las respuestas experimentales de los diferentes controladores diseñados, así como se realiza una comparativa entre las diferentes respuestas que genera cada controlador experimentalmente y su respectiva respuesta en simulación. Por último, en el Capítulo 7 se presentan las conclusiones obtenidas a lo largo del desarrollo del trabajo de tesis, se plantean mejoras en el sistema NXT ballbot para obtener un mejor desempeño y se mencionan posibles trabajos futuros apartir del trabajo realizado.

Capítulo 2

Control óptimo y robusto

2.1. Teoría moderna de control

El enfoque del control clásico es aplicable a sistemas con características de una entrada y una salida (SISO, *Single Input Single Output*, por sus siglas en inglés), lineales o que pueden ser linealizados y que son invariantes en el tiempo [15]. Pero debido a los avances tecnológicos y el requerimiento de sistemas con mayor precisión y capaces de realizar tareas más complejas y sofisticadas, estos sistemas pueden contar con múltiples entradas y múltiples salidas (MIMO, *Multiple Input Multiple Output*, por sus siglas en inglés), así como pueden ser variantes en el tiempo [56].

Es así como surge en los años de 1960, la teoría moderna de control, la cuál es aplicable a sistemas SISO y MIMO, que pueden ser lineales o no lineales e invariantes o variantes en el tiempo. Otra característica del control moderno es su aproximación en el dominio temporal, mientras que la teoría de control convencional es una aproximación en el dominio de la frecuencia compleja [56].

2.1.1. Espacio de estados

La teoría moderna de control se basa en el espacio de estados, definiendo como el estado de un sistema dinámico al conjunto de variables más pequeño (variables de estado) el cual conociendo un tiempo inicial t_0, junto con las variables de entrada determinan completamente el comportamiento del sistema para cualquier tiempo $t \geq t_0$ [15].

Las variables de estado constituyen al menor conjunto de variables que son requeridas para describir la naturaleza dinámica del sistema, y estas no necesariamente necesitan ser físicamente medibles, ya que si el sistema es observable entonces las variables que no se pueden medir son estimadas. La importancia de disponer de las mediciones de las variables de estado, o estimar algunas de ellas, reside en que las leyes de control óptimo requieren de realimentar todas las variables de estado. La manera en la cual las variables de estado cambian como una función del tiempo puede ser a través de una trayectoria en un espacio n-dimensional, llamado espacio de estados [15, 56].

Formulación de espacio de estados para sistemas en tiempo contínuo

Un sistema dinámico lineal e invariante en el tiempo (LTI, *Linear Time Invariant*, por sus siglas en inglés), en tiempo continuo es descrito por

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t), \quad \text{ecuación de estado} \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t). \quad \text{ecuación de salida} \end{aligned} \tag{2.1.1}$$

Con condiciones iniciales $\mathbf{x}(t = t_0) = \mathbf{x}(t_0)$. Aquí, $\mathbf{x}(t) \in \mathbb{R}^n$ es un vector de *estado*, $\mathbf{u}(t) \in \mathbb{R}^r$ es un vector de *control* , y $\mathbf{y}(t) \in \mathbb{R}^p$ es un vector de *salida* y las diversas matrices $\mathbf{A}, \mathbf{B}, \ldots$, son de dimensión apropiada [54].

2.1.2. Controlabilidad y observabilidad

Los conceptos de controlabilidad y observabilidad fueron introducidos por Kalman (1960), y juegan un rol importante en el control de sistemas SISO y MIMO [15].

Se dice que un sistema es controlable en el tiempo t_0 si se puede transferir desde un estado inicial $\mathbf{x}(t_0)$ a cualquier otro estado $\mathbf{x}(t)$, mediante un vector de control $\mathbf{u}(t)$ sin restricciones, en un intervalo de tiempo finito [15, 56].

Se dice que un sistema es observable en el tiempo t_0 si, con el sistema en el estado $\mathbf{x}(t_0)$, es posible determinar este estado a partir de mediciones de la salida $\mathbf{y}(t)$ y de la entrada $\mathbf{u}(t)$ [15, 56].

El sistema (2.1.1) con el par $(\mathbf{A} \in \mathbb{R}^{n\times n}, \mathbf{B} \in \mathbb{R}^{n\times r})$ se dice de *estado completamente controlable* si cualquiera de las siguientes condiciones se cumple

1. El rango de la matriz de controlabilidad

$$\mathbf{Q}_c = \left[\mathbf{B}\ \mathbf{AB}\ \mathbf{A^2B} \cdots \mathbf{A}^{n-1}\mathbf{B}\right], \tag{2.1.2}$$

   es n (rango de fila completa), o

2. El *Gramiano de controlabilidad*

$$\begin{aligned} \mathbf{W}_c(t) &= \int_0^t e^{\mathbf{A}\tau}\mathbf{B}\mathbf{B}^T e^{\mathbf{A}^T\tau}\, d\tau, \\ &= \int_0^t e^{\mathbf{A}t-\tau}\mathbf{B}\mathbf{B}^T e^{\mathbf{A}^T t-\tau}\, d\tau, \end{aligned} \tag{2.1.3}$$

   es no singular para cualquier $t > 0$ [15, 54, 56].

El sistema (2.1.1) con el par $(\mathbf{A} \in \mathbb{R}^{n\times n}, \mathbf{C} \in \mathbb{R}^{p\times n})$ es *completamente observable* si cualquiera de las siguientes condiciones se cumple

1. El rango de la matriz de observabilidad

$$\mathbf{Q}_o = \left[\mathbf{C}\ \mathbf{CA}\ \mathbf{CA^2} \cdots \mathbf{CA}^{n-1}\right]^T, \tag{2.1.4}$$

es n (rango de columna completa), o

2. El *Gramiano de Observabilidad*

$$\mathbf{W}_o(t) = \int_0^t e^{\mathbf{A}^T\tau}\mathbf{C}^T\mathbf{C}e^{\mathbf{A}\tau}\, d\tau, \tag{2.1.5}$$

es no singular para cualquier $t > 0$ [15, 54, 56].

2.1.3. Retroalimentación de variables de estado

Cualquier sistema en el que la magnitud de salida es monitoreada y comparada con la entrada, usando cualquier diferencia para llevar al sistema hasta que la salida sea igual a la entrada es llamado sistema de control en lazo cerrado o retroalimentado [15].

Del sistema descrito por las ecuaciones de estado y salida (2.1.1), se selecciona una ley de control de la forma

$$\mathbf{u} = (\mathbf{r} - \mathbf{Kx}), \tag{2.1.6}$$

donde, $\mathbf{r}(t)$ es un vector de variables de estado deseado y $\mathbf{K}$ se conoce como la matriz de ganancia de realimentación de estado.

Sustituyendo la ecuación (2.1.6) dentro de la ecuación (2.1.1) se obtiene

$$\dot{\mathbf{x}} = \mathbf{Ax} + \mathbf{B}(\mathbf{r} - \mathbf{Kx}),$$

o agrupando términos

$$\dot{\mathbf{x}} = (\mathbf{A} - \mathbf{BK})\mathbf{x} + \mathbf{Br}. \tag{2.1.7}$$

En la ecuación (2.1.7) la matriz $(\mathbf{A} - \mathbf{BK})$ es la matriz del sistema en lazo cerrado.

Para el sistema descrito por la ecuación (2.1.1), y usando la ecuación característica dada por

$$|(s\mathbf{I} - \mathbf{A})| = 0. \tag{2.1.8}$$

Las raíces de la ecuación (2.1.8) son los polos o eigenvalores en lazo abierto. Para el sistema en lazo cerrado descrito por la ecuación (2.1.7), la ecuación característica es

$$|(s\mathbf{I} - \mathbf{A} + \mathbf{BK})| = 0. \tag{2.1.9}$$

Las raíces de la ecuación (2.1.9) son los polos o eigenvalores en lazo cerrado [15].

2.2. Diseño de sistemas óptimos

La optimización es una característica muy deseable hoy en día, el ser humano busca la manera de realizar su trabajo, emplear recursos y administrar su tiempo de la manera más óptima posible. La optimización puede ser clasificada como optimización estática y optimización dinámica [54].

- **Optimización estática:** Se centra en controlar una planta bajo condiciones de estado estable, es decir, las variables del sistema no cambian con respecto al tiempo. La planta es descrita por ecuaciones algebraicas. Las técnicas utilizadas son cálculo elemental, multiplicadores de Lagrange, programación lineal y no lineal.

- **Optimización dinámica:** Centrada en el control óptimo de plantas bajo condiciones dinámicas, es decir, las variables del sistema cambian respecto al tiempo y por lo tanto el tiempo es involucrado en la descripción del sistema. La planta es entonces descrita por ecuaciones diferenciales. Las técnicas utilizadas son técnicas de búsqueda, programación dinámica, cálculo variacional y el principio de Pontryagin.

2.2.1. Control óptimo

El principal objetivo del control óptimo es determinar una ley de control, tal que la dinámica del sistema alcance un objetivo o siga a una variable de estado (o trayectoria) haciendo que opere dentro de ciertas limitantes físicas de control o en los estados del sistema y que al mismo tiempo optimice (maximizando o minimizando) un criterio de desempeño elegido (índice de desempeño o función de costo) [15, 54].

En la formulación de un problema de control óptimo se requiere:

- Un modelo matemático del proceso a ser controlado(generalmente en la forma de variables de estado).

- Una especificación del índice de desempeño, en base a la tarea a realizar.

- Un enunciado de condiciones límite y las restricciones físicas en los estados y/o controles.

Índice de desempeño

El índice de desempeño (PI, *Performance Index*, por sus siglas en inglés) a optimizar puede tomar varias formas dependiendo el tipo de problema que se aborde, como se describe a continuación.

***Índice de desempeño para sistemas de control óptimo en tiempo*:** En este problema se intenta transferir un sistema desde un estado inicial arbitrario $\mathbf{x}(t_0)$ a un estado final especificado $\mathbf{x}(t_f)$ en un tiempo *mínimo*. El correspondiente índice de desempeño se describe como

$$J = \int_{t_0}^{t_f} dt = t_f - t_0 = t^*,$$

donde $(*)$ indica condición óptima.

***Índice de desempeño para sistemas de control óptimo en combustible*:** Considerando el problema de una nave espacial. Siendo $u(t)$ el empuje de un motor de cohete y suponiendo que la magnitud $|u(t)|$ del empuje es proporcional a la razón de cambio en el consumo de combustible. Con el fin de *minimizar* el gasto total de combustible, la formulación del índice de desempeño se describe como

$$J = \int_{t_0}^{t_f} |u(t)|\, dt.$$

Para varios controles, podría escribirse como

$$J = \int_{t_0}^{t_f} \sum_{i=1}^{m} R_i\, |u_i(t)|\, dt,$$

donde R es un factor de ponderación.

***Índice de desempeño para sistemas de control de energía mínima*:** Considerando $u_i(t)$ como la corriente en el lazo i-ésimo de una red eléctrica. Entonces $\sum_{i=1}^{m} u_i^2(t) r_i$ (donde, r_i es la resistencia del lazo i-ésimo) es la potencia total o la *tasa* total de gasto de energía de la red. Entonces, para la minimización de la energía total consumida, se tiene un criterio de desempeño como

$$J = \int_{t_0}^{t_f} \sum_{i=1}^{m} u_i^2(t) r_i dt,$$

o en general

$$J = \int_{t_0}^{t_f} \mathbf{u}^T(t) \mathbf{R} \mathbf{u}(t) dt,$$

donde, $\mathbf{R}$ es una matriz *definida positiva* y (T) denota transpuesta.

Del mismo modo, se puede pensar en la minimización de la integral del error al cuadrado de un sistema de seguimiento. Por lo que se tiene

$$J = \int_{t_0}^{tf} \mathbf{x}^T(t)\mathbf{Q}\mathbf{x}(t)dt,$$

donde, $\mathbf{x}_d(t)$ es el valor deseado, $\mathbf{x}_a(t)$ es el valor actual, y $\mathbf{x}(t) = \mathbf{x}_a(t) - \mathbf{x}_d(t)$, es el error. Aquí $\mathbf{Q}$ es una matriz de ponderación, cuál puede ser *semidefinida positiva*.

***Índice de desempeño para sistemas de control terminal*:** En un problema de objetivo terminal, el interés es reducir al mínimo el error entre la posición objetivo deseada $\mathbf{x}_d(t_f)$ y la posición objetivo actual $\mathbf{x}_a(t_f)$ al final de la maniobra o en el tiempo final t_f. El error terminal (final) es $\mathbf{x}(t_f) = \mathbf{x}_a(t_f) - \mathbf{x}_d(t_f)$. Teniendo cuidado de los valores positivos y negativos de error y factores de ponderación, la función de costo se describe de la siguiente manera

$$J = \mathbf{x}^T(t_f)\mathbf{F}\mathbf{x}(t_f),$$

cuál también es llamada *función de costo terminal*. Aquí, $\mathbf{F}$ es una matriz *semidefinida positiva*.

***Índice de desempeño para sistemas de control general*:** Combinando las formulaciones anteriores, se tiene el siguiente índice de desempeño en forma general

$$J = \mathbf{x}^T(t_f)\mathbf{F}\mathbf{x}(t_f) + \int_{t_0}^{t_f} \left[\mathbf{x}^T(t)\mathbf{Q}\mathbf{x}(t) + \mathbf{u}^T(t)\mathbf{R}\mathbf{u}(t)\right] dt, \tag{2.2.1}$$

donde, $\mathbf{R}$ es una matriz *definida positiva*, y $\mathbf{Q}$ y $\mathbf{F}$ son matrices *semidefinidas positivas*. Como nota, las matrices $\mathbf{Q}$ y $\mathbf{R}$ pueden ser variantes en el tiempo.

Restricciones

Los vectores de control $\mathbf{u}$ y estado $\mathbf{x}$ siempre se encuentran limitados a la hora de implementar físicamente un controlador, esto se debe a las características propias de los sensores y actuadores empleados. Es por esto que es necesario conocer estas restricciones físicas, ya sea de corriente, voltaje, velocidades máximas, etc. y representarlas mediante

$$\mathbf{U}_- \leq \mathbf{u}(t) \leq \mathbf{U}_+, \quad \text{y} \quad \mathbf{X}_- \leq \mathbf{x}(t) \leq \mathbf{X}_+,$$

donde, $+$ y $-$ indican el valor máximo y mínimo que las variables pueden alcanzar.

2.2.2. Regulador cuadrático lineal (LQR)

El diseño de control tipo regulador cuadrático lineal (LQR) para un sistema de tipo (2.1.1) que se encuentra inicialmente desplazado del equilibrio, implica seleccionar valores apropiados para la matriz de ganancia de retroalimentación $\mathbf{K}$ en la ley de control lineal, de tal manera que el sistema retorne a su estado de equilibrio y que la función de costo se reduzca al mínimo [15, 47, 56].

La ley de control se describe mediante

$$\mathbf{u} = -\mathbf{K}\mathbf{x}. \tag{2.2.2}$$

Mientras que eligiendo la función de costo descrita por la ecuación (2.2.1), y considerando que para que J sea finita, es necesario que $\mathbf{x}(t_f) \to 0$, con lo que se descarta el primer término y se obtiene la función de costo siguiente

$$J = \int_0^\infty \left(\mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{u}^T\mathbf{R}\mathbf{u}\right) dt. \tag{2.2.3}$$

La función de costo es una función de la matriz de estados, $\mathbf{x} \in \mathbb{R}^n$, y la matriz de entradas, $\mathbf{u} \in \mathbb{R}^r$, en el caso MIMO. $\mathbf{Q} \in \mathbb{R}^{n\times n}$ y $\mathbf{R} \in \mathbb{R}^{r\times r}$ son matrices simétricas definidas positivas. La matriz de ponderación $\mathbf{Q}$ establece qué estados tendrán mayor prioridad a ser controlados que otros, la matriz $\mathbf{R}$ pondera la cantidad de acción de control a ser aplicada dependiendo de qué tan grande sea la desviación en el estado x_i. Esta ponderación en el costo de optimización, limita la magnitud de la señal de control.

Para determinar la matriz $\mathbf{K}$, primero se debe resolver $\mathbf{P} \in \mathbb{R}^{n\times r}$ de la siguiente ecuación de Riccati en su forma de matriz reducida

$$\mathbf{A}^T\mathbf{P} + \mathbf{P}\mathbf{A} - \mathbf{P}\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{P} + \mathbf{Q} = 0, \tag{2.2.4}$$

donde $\mathbf{P}$ es una matriz definida positiva que siempre existe cuando el sistema en lazo cerrado es estable (es decir, la matriz $(\mathbf{A} - \mathbf{B}\mathbf{K})$ es estable). Con la solución de $\mathbf{P}$, la matriz $\mathbf{K}$ se puede obtener a partir de

$$\mathbf{K} = \mathbf{R}^{-1}\mathbf{B}^T\,\mathbf{P}. \tag{2.2.5}$$

Finalmente, el costo mínimo es dado por

$$J = \frac{1}{2}\mathbf{x}^T\mathbf{P}\mathbf{x}.$$

2.3. Diseño de sistemas robustos

Un problema en el diseño de sistemas de control es que los sistemas físicos no se pueden modelar de manera precisa, así como su comportamiento puede cambiar repentinamente y a su vez estár sujetos a perturbaciones en el entorno en que operan. Para abordar estos problemas se han diseñado estrategias de control robusto que aseguran una estabilidad y desempeño robusto del sistema ante estas incertidumbres y perturbaciones.

2.3.1. Control robusto

El modelo dinámico de un sistema siempre será una representación aproximada del sistema físico real debido a que este presenta algunas de las siguientes características [18]:

- Cambios en los parámetros
- Dinámicas sin modelar
- Tiempos de retardo sin modelar
- Cambios en el punto de equilibrio (punto de operación)
- Ruido en los sensores
- Perturbaciones impredecibles

El objetivo de diseño de sistemas robustos es asegurar el desempeño del sistema a pesar de las imprecisiones y cambios en el modelo. Un sistema es *robusto* cuando el sistema presenta cambios aceptables en el desempeño debido a cambios en el modelo o imprecisiones. Un paso para el diseño de sistemas robustos es obtener un modelo convexo de la dinámica del sistema en el que se consideren las incertidumbres y posibles perturbaciones.

Diversos problemas dentro de la teoría de sistemas lineales y control robusto son formulados como optimización convexa mediante desigualdades matriciales lineales (LMI, *Linear Matrix Inequalities*, por sus siglas en inglés).

Convexidad

Un concepto importante dentro de las LMIs es la convexidad, por lo que se resumen algunos conceptos que serán mencionados posteriormente [31].

Definición 2.1 *Se dice que un conjunto A de $\mathbb{R}^n$ es convexo si, dados dos puntos cualesquiera de A, el segmento que los une está totalmente contenido en el conjunto, es decir, si la combinación convexa $(1-\lambda)x + \lambda y \in A$ para $x, y \in A$ y $0 \leq \lambda \leq 1$.*

Otro concepto interesante es el de *cono convexo* y *envoltura convexa*, que se definen a continuación.

Definición 2.2 *Un cono (convexo) es un subconjunto A de $\mathbb{R}^n$ que es convexo, no vacío y tal que si x está en A, entonces x también está en A para todo $\lambda \geq 0$.*

Definición 2.3 *Dado un conjunto arbitrario A, se define la envoltura convexa de A, y se representa por conv A, como la intersección de todos los subconjuntos convexos de $\mathbb{R}^n$ que contienen a A.*

Por último, se tiene que

Definición 2.4 *La envoltura convexa de un número finito de puntos se denomina politopo (convexo), en general, o polígono (convexo), en el caso del plano euclidiano.*

2.3.2. Desigualdades matriciales lineales (LMI)

El control LMI, es una estrategia que permite encontrar una solución a la vez que asegura el cumplimiento de un gran número de requisitos de diseño, donde las técnicas clásicas fallan en la obtención de una solución analítica. Esta técnica de control proporciona una solución numérica que puede ser obtenida a través de modernos algoritmos de optimización con el fin de asegurar el cumplimiento de los requerimientos de diseño y criterios de optimización como: restricción en la ubicación de polos, nivel de rechazo a perturbaciones y limitaciones en el esfuerzo de control. De esta manera se determinan las ganancias del controlador robusto [58].

Definición

Una LMI es una restricción de la forma

$$F(x) \triangleq F_0 + \sum_{i=1}^{m} x_i F_i > 0, \tag{2.3.1}$$

donde $\mathbf{x} = (x_1, \ldots, x_m)^T \in \mathbb{R}^m$ es el vector de m variables, $\mathbf{F}_i = \mathbf{F}_i^T \in \mathbb{R}^{n\times n}, i = 0, \ldots, m$ son matrices simétricas dadas, El simbolo de desigualdad $>$ significa que la matriz $\mathbf{F}(\mathbf{x})$ es definida positiva, es decir, $\mathbf{u}^T\ \mathbf{F}(x)\mathbf{u} > 0$ para toda $\mathbf{u} \in \mathbb{R}^n$ no cero [12, 19, 61].

Ademas existen LMIs no estrictas, de la forma $\mathbf{F}(\mathbf{x}) \geq 0$, donde $\geq$ significa que la matriz $\mathbf{F}(\mathbf{x})$ es semidefinida positiva, y LMIs de la forma $\mathbf{F}(\mathbf{x}) < 0$ cual obviamente es equivalente a $-\mathbf{F}(\mathbf{x}) > 0$.

Un conjunto de restricciones, o múltiples LMIs, tales que $\mathbf{F}^{(1)}(\mathbf{x}) > 0, \ldots, \mathbf{F}^{(p)}(\mathbf{x}) \geq 0$, pueden ser agrupados dentro de una única LMI:

$$\begin{bmatrix} F_1(x) & 0 & \ldots & 0 \\ 0 & F_2(x) & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \ldots & F_p(x) \end{bmatrix} > 0. \tag{2.3.2}$$

Propiedades

Propiedad 2.1 *Dada una matriz no singular* $\mathbf{T} = \mathbf{T}^T \in \mathbb{R}^{n\times n}$*, la LMI* $\mathbf{F}(\mathbf{x}) > 0$ *es equivalente a* $\mathbf{T}^T\mathbf{F}(\mathbf{x})\mathbf{T} > 0$.

Propiedad 2.2 (Complemento de Schur) *Dadas las siguientes matrices:* $\mathbf{Q}(\mathbf{x}) = \mathbf{Q}(\mathbf{x})^T \in \mathbb{R}^{n\times n}, \mathbf{R}(\mathbf{x}) = \mathbf{R}(\mathbf{x})^T \in \mathbb{R}^{m\times m}$ *y* $\mathbf{S}(\mathbf{x}) \in \mathbb{R}^{n\times m}$*, cuales dependen de* $(\mathbf{x})$*, la LMI*

$$\begin{bmatrix} \mathbf{Q}(\mathbf{x}) & \mathbf{S}(\mathbf{x}) \\ \mathbf{S}(\mathbf{x})^T & \mathbf{R}(\mathbf{x}) \end{bmatrix} > 0, \tag{2.3.3}$$

es equivalente a las dos LMIs

$$\begin{cases} \mathbf{R}(\mathbf{x}) & > & 0 \\ \mathbf{Q}(\mathbf{x}) - \mathbf{S}(\mathbf{x})\mathbf{R}(\mathbf{x})^{-1}\mathbf{S}(\mathbf{x})^T & > & 0, \end{cases}$$

y a las dos LMIs

$$\begin{cases} \mathbf{Q}(\mathbf{x}) & > & 0 \\ \mathbf{R}(\mathbf{x}) - \mathbf{S}(\mathbf{x})^T\mathbf{Q}(\mathbf{x})^{-1}\mathbf{S}(\mathbf{x}) & > & 0. \end{cases}$$

Problemas estándar que implican LMIs

Existen dos principales clases de problemas de optimización con restricciones expresadas como LMIs.

Problemas de factibilidad Este problema consiste en encontrar una $\mathbf{x} \in \mathbb{R}^m$, solución a la LMI $\mathbf{F}(\mathbf{x}) > 0$, o determinar que esta LMI es infactible, es decir, que tal $\mathbf{x}$ no existe.

Problemas de eigenvalor El problema del eigenvalor consiste en minimizar los eigenvalores de una matriz que dependa de alguna variable, sujeta a una restricción LMI:

$$\begin{array}{ll} \text{minimizar} & \lambda \\ \text{sujeto a} & \lambda\mathbf{I} - \mathbf{A}(\mathbf{x}) > 0, \mathbf{B}(\mathbf{x}), \end{array}$$

tales problemas aparecen frecuentemente en la forma equivalente de minimizar una función lineal de $\mathbf{x}$ sujeto a una LMI:

$$\begin{array}{ll} \text{minimizar} & \mathbf{c}^T\mathbf{x} \\ \text{sujeto a} & \mathbf{F}(\mathbf{x}) > 0, \end{array}$$

conocido también como programa semidefinido.

Inclusiones diferenciales

Una inclusión diferencial (DI, *Differential Inclusion*, por sus siglas en inglés) se describe por:

$$\dot{\mathbf{x}} \in \mathbf{F}(\mathbf{x}(t), t), \quad \mathbf{x}(0) = x_0, \tag{2.3.4}$$

donde $\mathbf{F}$ es un conjunto de funciones con valores en $\mathbb{R}^n \times \mathbb{R}_+$. Cualquier $\mathbf{x} : \mathbb{R}_+ \to \mathbb{R}^n$ que satisface la DI (2.3.4) es llamada *solución* o *trayectoria* de la DI (2.3.4). Por supuesto, puede haber varias soluciones para una DI dada.

Un caso particular de DI es la lineal (LDI) y está dada por

$$\dot{\mathbf{x}} \in \Omega\mathbf{x}, \quad \mathbf{x}(0) = x_0, \tag{2.3.5}$$

donde Ω es un subconjunto de $\mathbb{R}^{n\times n}$.

Una generalización de la LDI descrita anteriormente para sistemas con múltiples entradas y salidas se describe de la siguiente manera

$$\begin{array}{rcll} \dot{\mathbf{x}} & = & \mathbf{A}(t)\mathbf{x}+\mathbf{B}(t)\mathbf{u}, & \mathbf{x}(0)=x_0 \\ \mathbf{y} & = & \mathbf{C}(t)\mathbf{x}+\mathbf{D}(t)\mathbf{u}, & \end{array} \tag{2.3.6}$$

donde $\mathbf{x}:\mathbb{R}_+\to\mathbb{R}^n$ es referida como el estado, $\mathbf{u}:\mathbb{R}_+\to\mathbb{R}^{n_u}$ es referida como la entrada de control, $\mathbf{y}:\mathbb{R}_+\to\mathbb{R}^{n_y}$ es referida como la salida del sistema.

Las matrices en (2.3.6) satisfacen:

$$\begin{bmatrix} \mathbf{A}(t) & \mathbf{B}(t) \\ \mathbf{C}(t) & \mathbf{D}(t) \end{bmatrix} \in \Omega, \tag{2.3.7}$$

para todo $t\geq 0$, donde $\Omega\subseteq\mathbb{R}^{(n+n_y)\times(n+n_u)}$.

Sistemas lineales invariantes en el tiempo (LTI) Cuando Ω es un conjunto unitario, la LDI se reduce al sistema lineal invariante en el tiempo (LTI)

$$\begin{array}{rcll} \dot{\mathbf{x}} & = & \mathbf{A}\mathbf{x}+\mathbf{B}\mathbf{u}, & \mathbf{x}(0)=x_0 \\ \mathbf{y} & = & \mathbf{C}\mathbf{x}+\mathbf{D}\mathbf{u}, & \end{array} \tag{2.3.8}$$

donde

$$\Omega=\left\{\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}\right\}. \tag{2.3.9}$$

LDIs politópicos (PLDI) Estos sistemas se pueden ver como una extensión del sistema LTI con matrices dependientes linealmente de parámetros inciertos, es decir

$$\begin{array}{rcll} \dot{\mathbf{x}} & = & \mathbf{A}(\rho)\mathbf{x}+\mathbf{B}(\rho)\mathbf{u}, & \mathbf{x}(0)=x_0 \\ \mathbf{y} & = & \mathbf{C}(\rho)\mathbf{x}+\mathbf{D}(\rho)\mathbf{u}, & \end{array} \tag{2.3.10}$$

donde $\rho=(\rho_1,\ldots,\rho_{n_p})$ es el vector de agrupación de n_p parámetros inciertos del sistema. Cada parámetro incierto ρ_i se encuentra limitado entre un valor mínimo y un valor máximo $[\rho_{i_{\text{mín}}},\rho_{i_{\text{máx}}}]$. Las posibles combinaciones de los valores mínimos y máximos del vector ρ define un politopo convexo con 2^{n_p} vértices $\{v_1,\ldots,v_L\}$, que puede ser representado mediante la siguiente combinación convexa

$$\rho\in \mathrm{C}_0\{v_1,\ldots,v_L\}=\left\{\sum_{i=1}^{L}\vartheta_i v_i,\quad \vartheta_i>0,\quad \sum_{i=1}^{L}\vartheta_i=1\right\},$$

donde C_0 es la envolvente convexa del conjunto de vértices $\in\mathbb{R}^{n_p}$.

Cuando la dependencia es lineal sobre ρ, el sistema de matrices $\Omega(\rho)$ se puede representar mediante el siguiente politopo convexo

$$\Omega(\rho) \in \mathrm{C}_0 \left\{\zeta_1, \ldots, \zeta_L\right\} = \left\{ \sum_{i=1}^{L} \vartheta_i \zeta_i, \quad \vartheta_i > 0, \quad \sum_{i=1}^{L} \vartheta_i = 1 \right\},$$

siendo

$$\Omega(\rho) \in \mathrm{C}_0 = \left\{ \begin{bmatrix} \mathbf{A}_1 & \mathbf{B}_1 \\ \mathbf{C}_1 & \mathbf{D}_1 \end{bmatrix}, \ldots, \begin{bmatrix} \mathbf{A}_L & \mathbf{B}_L \\ \mathbf{C}_L & \mathbf{D}_L \end{bmatrix} \right\}, \tag{2.3.11}$$

donde los vértices ζ_i corresponden a la imágenes vi, es decir $\zeta_i = \Omega(v_i)$. Una representación gráfica del sistema politópico se muestra en la Figura 2.1. Esto permite evaluar todos los estados intermedios incluidos en el politopo.

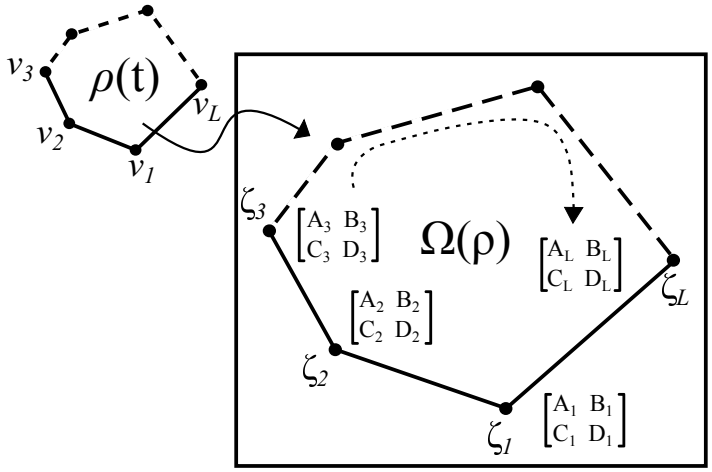


Figura 2.1: Representación gráfica de la incertidumbre politópica.

Estabilidad cuadrática

Considerando las inclusiones diferenciales del tipo

$$\dot{\mathbf{x}} = \mathbf{A}(t)\mathbf{x}, \quad \mathbf{A}(t) \in \Omega, \tag{2.3.12}$$

donde Ω tiene alguna de las formas como en (2.3.9) o (2.3.11), una condición suficiente para que cualquier trayectoria de una LDI converja al origen (cero), cuando $t \to \infty$, es la existencia de una función cuadrática $V(\mathbf{x}) = \mathbf{x}^T\mathbf{P}\mathbf{x}, \quad \mathbf{P} > 0$ que decrece a lo largo de cualquier trayectoria no cero (no trivial) de la LDI (2.3.12). Si existe tal $\mathbf{P}$, se dice que la LDI (2.3.12) es cuadráticamente estable y llamamos a V como función cuadrática de Lyapunov.

Desde que

$$\frac{d}{dt}V(\mathbf{x}) = \mathbf{x}^T(\mathbf{A}(t)^T\mathbf{P} + \mathbf{P}\mathbf{A}(t))\mathbf{x},$$

se tiene que una condición necesaria y suficiente para la estabilidad cuadrática del sistema (2.3.12) es

$$\mathbf{P} > 0, \quad \mathbf{A}^T\mathbf{P} + \mathbf{P}\mathbf{A} < 0 \quad \forall \mathbf{A} \in \Omega. \tag{2.3.13}$$

Multiplicando la segunda desigualdad en (2.3.13) a la izquierda y derecha por $\mathbf{P}^{-1}$, y definiendo una nueva variable $\mathbf{Q} = \mathbf{P}^{-1}$, se puede reescribir (2.3.13) como

$$\mathbf{Q} > 0, \quad \mathbf{Q}\mathbf{A}^T + \mathbf{A}\mathbf{Q} < 0 \quad \forall \mathbf{A} \in \Omega. \tag{2.3.14}$$

Esta doble desigualdad es una condición equivalente para la estabilidad cuadrática. Para los sistemas LTI y PLDI se tiene

- Sistemas LTI:
$$\mathbf{P} > 0, \quad \mathbf{A}^T\mathbf{P} + \mathbf{P}\mathbf{A} < 0.$$
- Sistemas Politópicos PLDI:
$$\mathbf{P} > 0, \quad \mathbf{A}_i^T\mathbf{P} + \mathbf{P}\mathbf{A}_i < 0, \quad i = 0, \ldots, L.$$

o análogamente

$$\mathbf{Q} > 0, \quad \mathbf{A}\mathbf{Q} + \mathbf{Q}\mathbf{A}^T < 0.$$

$$\mathbf{Q} > 0, \quad \mathbf{Q}\mathbf{A}_i^T + \mathbf{A}_i\mathbf{P} < 0, \quad i = 0, \ldots, L.$$

2.3.3. Síntesis de retroalimentación de estados para LDIs

El problema de síntesis de retroalimentación de estados es encontrar una matriz $\mathbf{K}$ tal que la LDI (2.3.6) en lazo cerrado satisfaga ciertas propiedades o especificaciones de diseño, por ejemplo, estabilidad.

Estabilizabilidad cuadrática

El sistema (2.3.6) se dice que es estabilizable cuadŕaticamente (vía retroalimentación de estados lineal) si existe una ganancia de retroalimentación de estados $\mathbf{K}$, tal que el sistema en lazo cerrado sea cuadráticamente estable. La estabilizabilidad cuadrática puede ser expresada como un problema LMI.

Sistemas LTI: Fijando la matriz $\mathbf{K}$. El sistema LTI (2.3.8) es cuadráticamente estable si y solo si existe $\mathbf{P} > 0$ tal que

$$(\mathbf{A} - \mathbf{B}\mathbf{K})^T \mathbf{P} + \mathbf{P} (\mathbf{A} - \mathbf{B}\mathbf{K}) < 0,$$

o equivalentemente, si existe $\mathbf{Q} > 0$, tal que

$$\mathbf{Q}\left(\mathbf{A} - \mathbf{BK}\right)^T + \left(\mathbf{A} - \mathbf{BK}\right)\mathbf{Q} < 0. \tag{2.3.15}$$

Ninguna de estas dos expresiones es conjuntamente convexa en $\mathbf{K}$ y $\mathbf{P}$ ó $\mathbf{Q}$, pero con un simple cambio de variables, se obtiene una expresión equivalente que tiene la propiedad de ser una LMI. Haciendo $\mathbf{Y} = \mathbf{KQ}$, tal que para $\mathbf{Q} > 0$ tenemos $\mathbf{K} = \mathbf{YQ}^{-1}$.

Sustituyendo en (2.3.15) se tiene

$$\mathbf{AQ} + \mathbf{QA}^T - \mathbf{BY} - \mathbf{Y}^T\mathbf{B}^T < 0, \tag{2.3.16}$$

la cual es una LMI en $\mathbf{Q}$ y $\mathbf{Y}$.

Por lo tanto, el sistema (2.3.8) es cuadráticamente estable, si y solo si existe $\mathbf{Q} > 0$ y $\mathbf{Y}$, tal que la LMI (2.3.16) se cumple. Si esta LMI es factible, entonces la función cuadrática $V(\mathbf{x}) = \mathbf{x}^T\mathbf{Q}^{-1}\mathbf{x}$ demuestra la estabilidad cuadrática del sistema (2.3.8) con retroalimentación de estados $\mathbf{u} = \mathbf{YQ}^{-1}\mathbf{x}$.

Sistemas politópicos LDI: Para la PLDI (2.3.10), el mismo argumento se aplica. La estabilidad cuadrática es equivalente a la existencia de $\mathbf{Q} > 0$ y $\mathbf{Y}$ con

$$\mathbf{A}_i\mathbf{Q} + \mathbf{QA}_i^T - \mathbf{B}_i\mathbf{Y} - \mathbf{Y}^T\mathbf{B}_i^T < 0, \quad i = 1, \ldots, L. \tag{2.3.17}$$

La estabilizabilidad cuadrática también se puede interpretar como una elipsoide invariante. Se dice que el elipsoide

$$\varepsilon = \left\{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{x}^T\mathbf{Q}^{-1}\mathbf{x} \leq 1\right\}$$

es capaz de cumplirse para el sistema (2.3.6), si existe una ganancia de retroalimentación de estados $\mathbf{K}$, tal que ε sea invariante para el sistema (2.3.6) con $\mathbf{u} = -\mathbf{Kx}$.

Restricciones en la entrada de control

Cuando la condición inicial es conocida, se puede encontrar una cota superior en la norma de entrada de control $\mathbf{u}(t) = -\mathbf{Kx}(t)$ como sigue. Se elije $\mathbf{Q} > 0$ y $\mathbf{Y}$ cuales satisfacen la condición de estabilidad cuadrática (cualquiera (2.3.16),(2.3.17)), y ademas $\mathbf{x}(0)^T\mathbf{Q}^{-1}\mathbf{x}(0) \leq 1$. Esto implica que $\mathbf{x}(t)$ pertenece a ε para todo $t \geq 0$, así se tiene que

$$\begin{aligned}
\max_{t\geq 0} \|\mathbf{u}(t)\| &= \max_{t\geq 0} \left\|-\mathbf{YQ}^{-1}\mathbf{x}(t)\right\|, \\
&\leq \max_{\mathbf{x}\in\varepsilon} \left\|-\mathbf{YQ}^{-1}\mathbf{x}\right\|, \\
&= \lambda_{max}\left(\mathbf{Q}^{-\frac{1}{2}}\mathbf{Y}^T\mathbf{YQ}^{-\frac{1}{2}}\right).
\end{aligned}$$

Por lo tanto, la restricción $\|\mathbf{u}(t)\| \leq \mu$ se garantiza para todo el tiempo $t \geq 0$, si las LMIs

$$\begin{bmatrix} 1 & \mathbf{x}(0)^T \\ \mathbf{x} & \mathbf{Q} \end{bmatrix} \geq 0, \qquad \begin{bmatrix} \mathbf{Q} & \mathbf{Y}^T \\ \mathbf{Y} & \mu^2\mathbf{I} \end{bmatrix} \geq 0 \tag{2.3.18}$$

se mantienen, donde $\mathbf{Q} > 0$ y $\mathbf{Y}$ satisface las condiciones de estabilizabilidad (2.3.16),(2.3.17).

Ubicación de polos para sistemas politópicos

La estabilidad es un requerimiento mínimo para los sistemas de control robusto. Sin embargo, en la práctica es usual evaluar el desempeño de la respuesta transitoria del sistema de control, como lo son: el tiempo de establecimiento, tiempo de subida y amortiguamiento, entre otros. Estos indicadores están relacionados a la localización de los polos del sistema en lazo cerrado en ciertas regiones del plano complejo.

Antes de enunciar los requerimientos para la ubicación de los polos en determinadas regiones del plano complejo, se define lo que es una región LMI [16].

Definición 2.5 (Regiones LMI) *Un subconjunto $\mathcal{D}$ del plano complejo se denomina región LMI si existe una matriz simétrica $\mathbf{L} \in \mathbb{R}^{m\times m}$ y una matriz $\mathbf{M} \in \mathbb{R}^{m\times m}$, tales que*

$$\mathcal{D} = \left\{ z \in \boldsymbol{C} : \quad \mathbf{L} + z\mathbf{M} + \bar{z}\mathbf{M}^T < 0 \right\},$$

donde la función

$$f_{\mathcal{D}}(z) = \mathbf{L} + z\mathbf{M} + \bar{z}\mathbf{M}^T,$$

es llamada la función característica de $\mathcal{D}$.

Así se dice también que, una región LMI es un subconjunto del plano complejo que es representable por una LMI en z y $\bar{z}$, o equivalentemente, una LMI en $x = \mathrm{Re}(z)$ y $y = \mathrm{Im}(z)$. Como consecuencia, las regiones LMI son convexas. Ademas, las regiones LMI son simétricas en relación al eje real del plano complejo.

Una región interesante para propósitos de control es el conjunto $S(\alpha, r, \theta)$ de numeros complejos $x + jy$, tal que

$$x < -\alpha < 0, \quad |x + jy| < r, \quad \tan\theta x < -|y|, \tag{2.3.19}$$

como se muestra en la Figura 2.2, donde la región que confina a los polos en lazo cerrado asegura una tasa mínima de decaimiento α, una mínima tasa de amortiguamiento $\zeta = \cos\theta$, y una máxima frecuencia natural no amortiguada $w_d = r\sin\theta$.

El siguiente teorema permite restringir la ubicación de polos en lazo cerrado en la región $S(\alpha, r, \theta)$.

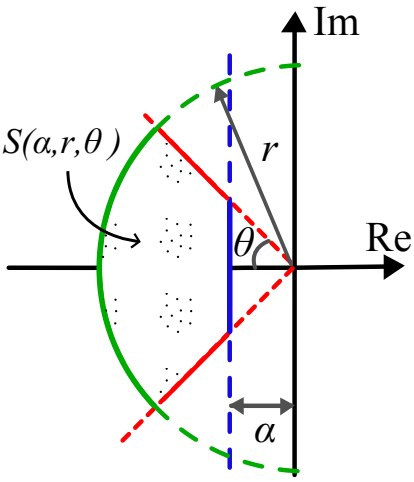


Figura 2.2: Región de ubicación de polos $S(\alpha, r, \theta)$.

Teorema 2.1 *Los polos en lazo cerrado del sistema* (2.3.10) *con una retroalimentación de estados* $\mathbf{u} = -\mathbf{Kx}$ *se ubican dentro de la región* $S(\alpha, r, \theta)$ *si existen matrices* $\mathbf{W} = \mathbf{W}^T \in \mathbb{R}^{n\times n} > 0$ *y* $\mathbf{Y} \in \mathbb{R}^{m\times n}$ *que satisfacen*

$$\mathbf{A}_i\mathbf{W} + \mathbf{W}\mathbf{A}_i^T - \boldsymbol{B}_i\mathbf{Y} - \mathbf{Y}^T\mathbf{B}_i^T + 2\alpha\mathbf{W} \;<\; 0, \tag{2.3.20}$$

$$\begin{bmatrix} -r\mathbf{W} & \mathbf{W}\mathbf{A}_i^T - \mathbf{Y}^T\mathbf{B}_i^T \\ \mathbf{A}_i\mathbf{W} - \mathbf{B}_i\mathbf{Y} & -r\mathbf{W} \end{bmatrix} \;<\; 0, \tag{2.3.21}$$

$$\begin{bmatrix} \sin\theta(\mathbf{A}_i\mathbf{W} + \mathbf{W}\mathbf{A}_i^T - \boldsymbol{B}_i\mathbf{Y} - \mathbf{Y}^T\mathbf{B}_i^T) & \cos\theta(\mathbf{A}_i\mathbf{W} - \mathbf{W}\mathbf{A}_i^T - \boldsymbol{B}_i\mathbf{Y} + \mathbf{Y}^T\mathbf{B}_i^T) \\ \cos\theta(-\mathbf{A}_i\mathbf{W} + \mathbf{W}\mathbf{A}_i^T + \boldsymbol{B}_i\mathbf{Y} - \mathbf{Y}^T\mathbf{B}_i^T) & \sin\theta(\mathbf{A}_i\mathbf{W} + \mathbf{W}\mathbf{A}_i^T - \boldsymbol{B}_i\mathbf{Y} - \mathbf{Y}^T\mathbf{B}_i^T) \end{bmatrix} \;<\; 0, \tag{2.3.22}$$

siendo $\mathbf{K} = \mathbf{Y}\mathbf{W}^{-1}$ *la ganancia de retroalimentación de estados.*

Para una consulta mas extensa, así como ejemplos acerca de regiones LMIs y ubicación de polos se puede consultar las siguientes referencias [16, 57, 72].

Control LQR mediante LMIs

El problema del LQR se puede extender a sistemas politópicos mediante restricciones LMI, lo que permite encontrar una solución para sistemas con incertidumbre en sus parámetros, mientras que el LQR clásico solo es válido para sistemas sin incertidumbres. Para este caso, sea el sistema lineal invariante en el tiempo con incertidumbres paramétricas descrito por

$$\dot{\mathbf{x}} \;=\; \mathbf{A}(\rho)\mathbf{x} + \mathbf{B}(\rho)\mathbf{u}, \qquad \mathbf{x}(0) = x_0 \tag{2.3.23}$$

el problema consiste en encontrar una ley de control mediante retroalimentación de estados $\mathbf{u} = -\mathbf{Kx}$, tal que minimice la siguiente función de costo

$$J = \int_0^\infty \left(\mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{u}^T\mathbf{R}\mathbf{u}\right) dt.$$

Sustituyendo la ley de control en la función de costo, esta se reescribe como

$$J = \int_0^\infty \left(\mathbf{x}^T\left(\mathbf{Q} + \mathbf{K}^T\mathbf{R}\mathbf{K}\right)\mathbf{x}\right) dt.$$

Al emplear el operador de traza $\mathbf{Tr}(\cdot)$, cual satisface $\mathbf{Tr}(A^TBC) = \mathbf{Tr}(BCA^T)$, la función de costo es equivalente a

$$\begin{aligned} J &= \int_0^\infty \mathbf{Tr}\left(\left(\mathbf{Q} + \mathbf{K}^T\mathbf{R}\mathbf{K}\right)\mathbf{x}\mathbf{x}^T\right) dt, \\ &= \mathbf{Tr}\left(\left(\mathbf{Q} + \mathbf{K}^T\mathbf{R}\mathbf{K}\right)\mathbf{P}\right), \end{aligned}$$

donde $\mathbf{P} = \int_0^\infty \left(\mathbf{x}\mathbf{x}^T\right) dt$ es una matriz simétrica definida positiva que satisface

$$\left(\mathbf{A}_i - \mathbf{B}_i\mathbf{K}\right)\mathbf{P} + \mathbf{P}\left(\mathbf{A}_i - \mathbf{B}_i\mathbf{K}\right)^T + \mathbf{I} = 0, \quad i = 1, \ldots, L. \tag{2.3.24}$$

donde el sub índice i representa cada uno de los vértices del politopo [40, 43, 58].

La ganancia de retroalimentación óptima $\mathbf{K}$ puede determinarse mediante la minimización de la siguiente expresión:

$$\begin{aligned} \underset{\mathbf{P},\mathbf{K}}{\text{mín}} \quad & \mathbf{Tr}\left(\mathbf{Q}\mathbf{P}\right) + \mathbf{Tr}\left(\mathbf{R}^{\frac{1}{2}}\mathbf{K}\mathbf{P}\mathbf{K}^T\mathbf{R}^{\frac{1}{2}}\right) \\ \text{sujeto a} \quad & \\ & \left(\mathbf{A}_i - \mathbf{B}_i\mathbf{K}\right)\mathbf{P} + \mathbf{P}\left(\mathbf{A}_i - \mathbf{B}_i\mathbf{K}\right)^T + \mathbf{I} < 0. \end{aligned} \tag{2.3.25}$$

Sin embargo, (2.3.25) no es conjuntamente convexa en $\mathbf{P}, \mathbf{K}$, por lo que se introduce una nueva variable $\mathbf{Y} = \mathbf{K}\mathbf{P}$, reescribiendo a (2.3.25) como

$$\begin{aligned} \underset{\mathbf{P},\mathbf{K}}{\text{mín}} \quad & \mathbf{Tr}\left(\mathbf{Q}\mathbf{P}\right) + \mathbf{Tr}\left(\mathbf{R}^{\frac{1}{2}}\mathbf{Y}\mathbf{P}^{-1}\mathbf{Y}^T\mathbf{R}^{\frac{1}{2}}\right) \\ \text{sujeto a} \quad & \\ & \mathbf{A}_i\mathbf{P} + \mathbf{P}\mathbf{A}_i^T - \mathbf{B}_i\mathbf{Y} - \mathbf{Y}^T\mathbf{B}_i^T + \mathbf{I} < 0. \end{aligned} \tag{2.3.26}$$

Así también, el término no lineal $\left(\mathbf{R}^{\frac{1}{2}}\mathbf{Y}\mathbf{P}^{-1}\mathbf{Y}^T\mathbf{R}^{\frac{1}{2}}\right)$ se puede reemplazar por una segunda variable auxiliar $\mathbf{X}$ tal que

$$\mathbf{X} > \mathbf{R}^{\frac{1}{2}}\mathbf{Y}\mathbf{P}^{-1}\mathbf{Y}^T\mathbf{R}^{\frac{1}{2}}, \tag{2.3.27}$$

la cual empleando el complemento de Schur, se reescribe como

$$\mathbf{X} - \left(\mathbf{R}^{\frac{1}{2}}\mathbf{Y}\mathbf{P}^{-1}\mathbf{Y}^T\mathbf{R}^{\frac{1}{2}}\right) > 0 \leftrightarrow \begin{bmatrix} \mathbf{X} & \mathbf{R}^{\frac{1}{2}}\mathbf{Y} \\ \mathbf{Y}^T\mathbf{R}^{\frac{1}{2}} & \mathbf{P} \end{bmatrix} > 0. \tag{2.3.28}$$

Por lo tanto, la formulación completa del problema LQR para sistemas politópicos queda definida como

$$\begin{aligned} \min_{\mathbf{P},\mathbf{K},\mathbf{X}} \quad & \mathbf{Tr}\left(\mathbf{QP}\right) + \mathbf{Tr}\left(\mathbf{X}\right) \\ \text{sujeto a} \quad & \\ & \mathbf{P} > 0, \\ & \mathbf{A}_i\mathbf{P} + \mathbf{P}\mathbf{A}_i^T - \mathbf{B}_i\mathbf{Y} - \mathbf{Y}^T\mathbf{B}_i^T + \mathbf{I} < 0, \\ & \begin{bmatrix} \mathbf{X} & \mathbf{R}^{\frac{1}{2}}\mathbf{Y} \\ \mathbf{Y}^T\mathbf{R}^{\frac{1}{2}} & \mathbf{P} \end{bmatrix} > 0. \end{aligned} \tag{2.3.29}$$

Una vez que dichas restricciones son resueltas, la ganancia de retroalimentación de estados óptima se determina mediante $\mathbf{K} = \mathbf{Y}\mathbf{P}^{-1}$.

Capítulo 3

Modelado dinámico y estimación de parámetros del sistema NXT ballbot

En este capítulo se describirá la importancia de modelar un sistema dinámico. Se analizarán los elementos que describen el funcionamiento del sistema ballbot. Se presentarán las pruebas experimentales para obtener los valores de los parámetros físicos que intervienen en el modelado de los elementos del sistema. Al final se realizará el modelado dinámico del sistema NXT ballbot, así como la linealización del sistema alrededor de un punto de operación para su representación en espacio de estados y la descripción del modelo politópico del sistema.

3.1. Modelado matemático de sistemas dinámicos

El modelo matemático de un sistema dinámico se define como un conjunto de ecuaciones representativas de la dinámica de dicho sistema. Este conjunto de ecuaciones no es único ya que depende de la manera en que se aborda el comportamiento del sistema, de la precisión que se requiera, y de la formulación de ecuaciones a partir de las leyes físicas que modelan el comportamiento del sistema [56]. Para los sistemas mecánicos en algunas ocasiones se emplea la formulación mediante Newton, Euler-Lagrange y el hamiltoniano, dependiendo de las restricciones o facilidades del sistema para su análisis.

Un modelo matemático puede resultar más conveniente que otros, por ejemplo, para realizar un análisis de la respuesta transitoria o la respuesta en frecuencia de sistemas lineales con una entrada y una salida invariantes en el tiempo, la representación mediante la función de transferencia puede ser un método conveniente, mientras que en problemas de control óptimo, es ventajoso usar representaciones en el espacio de estados [56], pues son de gran ayuda en cuanto al costo computacional y su implementación en sistemas digitales.

Para obtener un modelo matemático fiable, es necesario analizar los elementos que conforman al sistema, la relación que existe entre estos elementos, así como conocer los valores de los parámetros que intervienen. A continuación se analizarán los elementos que constituyen al sistema NXT ballbot, los cuales son: sensores, actuadores y la estructura del robot.

3.2. Elementos del sistema NXT ballbot

Antes de analizar el comportamiento de los sensores y actuadores que intervienen en el funcionamiento del sistema NXT ballbot, se describirán las características de los puertos de entrada y salida que emplea el ladrillo NXT para su interacción con estos dispositivos [35, 46].

3.2.1. Puertos NXT

El ladrillo NXT posee cuatro puertos de entrada para los sensores, estos puertos se localizan en la parte inferior del ladrillo NXT, y están enumerados del 1 al 4. Los puertos de salida para los motores son tres y se encuentran en la parte superior del ladrillo NXT, etiquetados como A, B y C. Si se observa a través del conector en un extremo de un cable del NXT, se puede observar seis pines correspondientes a los cables que interaccionan con el ladrillo NXT. Estos cables están codificados por los colores: blanco, negro, rojo, verde, amarillo y azul.

La función de estos cables depende de si son usados en una entrada para sensor o una salida para motor. aún así su función está sujeta al tipo de sensor conectado.

Puertos de entrada *LEGO Mindstorms NXT®*

La Figura 3.1 muestra detalles esquemáticos de la conexión del puerto 1 del ladrillo. Los esquemas de los puertos 2, 3 y 4 son idénticos.

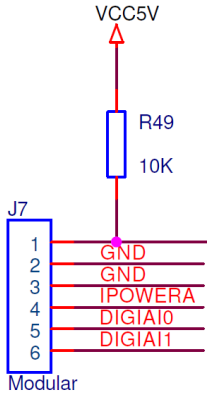


Figura 3.1: Esquemático de los pines de entrada del *LEGO Mindstorms NXT®*. Imagen tomada de [46].

En la Tabla 3.1 se resumen los colores y nombres de los pines de un puerto de entrada para sensor del ladrillo NXT. Como se observa el puerto de entrada tiene la capacidad de manejar sensores analógicos o digitales.

Pin 1 - [AN]

Este pin se utiliza para dos propósitos: Como una entrada analógica, usada para la compatibilidad con los sensores del antiguo RCX; O como una fuente de voltaje (9V o 7.2V,

Tabla 3.1: Colores y nombres de los pines de entrada. Tabla tomada de [35].

Número de Pin	Color	Nombre
1	Blanco	AN
2	Negro	GND
3	Rojo	GND
4	Verde	4.3V POWER
5	Amarillo	DIGI0
6	Azul	DIGI1

dependiendo de las baterías), para el sensor de luz RCX y el sensor ultrasónico NXT.

Cuando el pin se utiliza como una entrada analógica, la señal es conectada a un convertidor analógico-digital de 10 bits. El valor de la señal de entrada varía entre 0V y 5V, que se traducen en una valor digital entre 0 y 1023. La frecuencia de muestreo para todos los sensores analógicos es de 333Hz. Este pin es conectado permanentemente a 5V a través de una resistencia pull-up de 10KΩ. La corriente límite aproximada que provee por puerto de entrada es de 14mA. Si la corriente es mayor a este valor, el voltaje decaerá rápidamente.

Pines 2 y 3 - [GND]

Los pines 2 y 3 son pines de tierra. Estos dos pines están unidos dentro del ladrillo NXT y en los sensores de LEGO®. Todas las señales se miden referente a estos pines de tierra.

Pin 4 - [4.3V Power]

Este pin es la fuente principal de alimentación para todos los sensores NXT, y está conectado internamente con todas las fuentes de alimentación tanto de los puertos de entrada como de salida. Esta fuente de alimentación tiene una corriente máxima de salida de 180mA para los siete puertos de entrada y salida, por lo que cada puerto puede emplear 25mA en promedio. En la Tabla 3.2 se muestran los consumos de corriente para algunos sensores NXT estándar y los encoders de los motores NXT.

Tabla 3.2: Consumo de corriente de los sensores y encoders NXT. Tabla tomada de [35]

Dispositivo	Consumo
Tacto	0
Sonido	1.7mA
Sensor de luz (luz apagada)	2.6 a 3mA (dependiendo de la luz)
Sensor de luz (luz encendida)	16.3mA
Sensor ultrasónico	4mA
Encoder de posición	9 a 12mA (dependiendo de la posición)

La Figura 3.2 muestra cómo decae el voltaje de alimentación a medida que la carga aumenta. El voltaje es aproximadamente 5V sin carga, y tan solo 4.3V con una carga considerable. Aún así es capaz de alimentar circuitos integrados de tecnología reciente con una lógica de 5V. Si la potencia requerida es mayor, la corriente de salida decrecerá automáticamente sin ninguna advertencia. Si la señal de alimentación está en corto circuito a tierra, el ladrillo NXT se reiniciará.

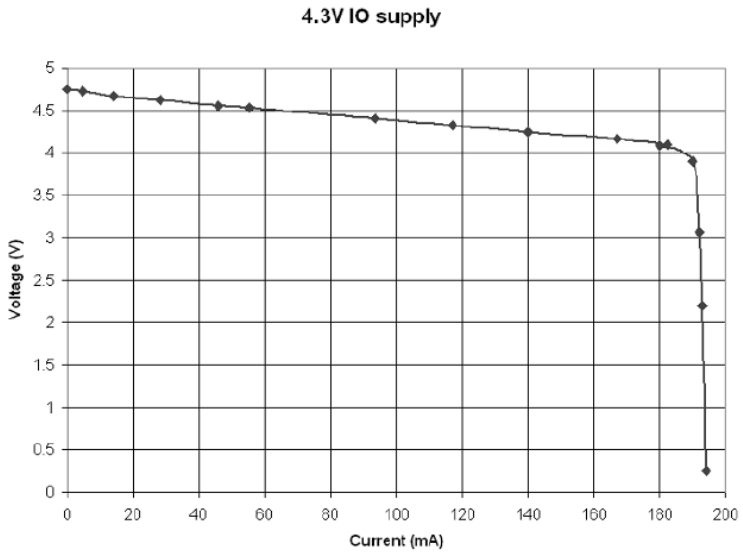


Figura 3.2: Gráfica de voltaje-corriente en el pin 4. Imagen tomada de [35].

Pines 5 y 6 - [DIGI0-DIGI1]

Estos pines son señales digitales a 3.3V y están conectados directamente al microprocesador del NXT. Estos pines se utilizan principalmente para establecer comunicaciones digitales implementadas mediante I^2C, compatibles con una comunicación que corre a 9600 bits/s. El ladrillo NXT posee un circuito de protección para evitar que altos voltajes puedan producir algún daño. Este circuito incluye una resistencia de 4.7KΩ conectada en serie con el puerto, así incluso si el voltaje del sensor es demasiado alto, la corriente será baja.

Puertos de salida *LEGO Mindstorms NXT®*

La Figura 3.3 muestra detalles esquemáticos de la conexión A del ladrillo. Los esquemas de los puertos B y C son idénticos.

La Tabla 3.3 resume los colores y los nombres de los pines de salida del motor NXT.

Pines 1 y 2 [M0 - M1]

Estos pines proporcionan energía al motor. La tensión máxima es la tensión de las pilas (9V para baterías estándar, 7.2V para baterías de Ni-MH). El motor es controlado por un

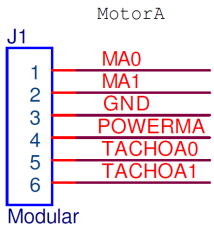


Figura 3.3: Esquemático de los pines de salida del *LEGO Mindstorms NXT®*. Imagen tomada de [46].

Tabla 3.3: Colores y nombres de los pines de salida. Tabla tomada de [35].

Número de Pin	**Color**	**Nombre**
1	Blanco	M0
2	Negro	M1
3	Rojo	GND
4	Verde	4.3V POWER
5	Amarillo	TACHO0
6	Azul	TACHO1

circuito denominado puente H (controladores tipo LB1836M y LB1930M), el que nos permite girar al motor en ambos sentidos. Un puente H está constituido por cuatro transistores , en las siguientes figuras los transistores están etiquetados del 1 al 4.

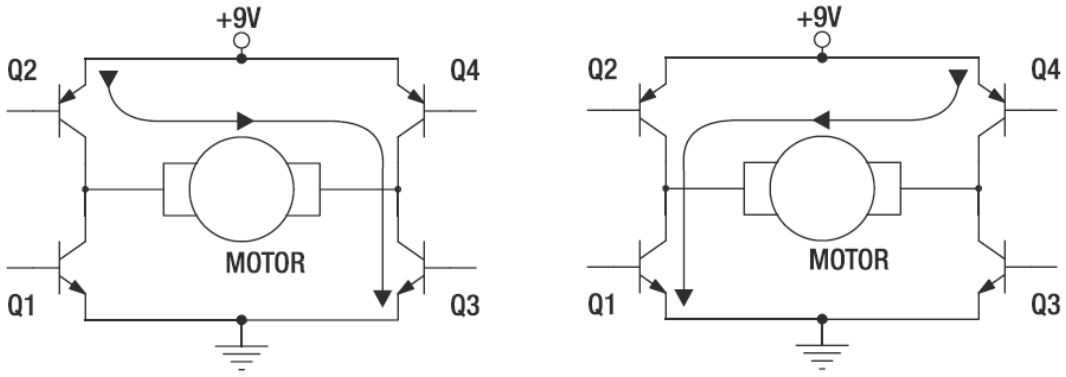
El circuito de control está diseñado de manera que los transistores Q1 y Q2 de un lado y Q3 y Q4 por otro lado no están conduciendo simultáneamente, esto para evitar un corto-circuito. La Figura 3.4(a) muestra el estado de los transistores para la condición de avance, con los transistores Q1 y Q4 abiertos. La Figura 3.4(b) muestra la condición de retroceso, con Q2 y Q3 abiertos. La Figura 3.5 muestra el flujo de corriente para las condiciones de frenado rápido y frenado bajo su propia inercia.

La velocidad del motor está controlada por la técnica de modulación por ancho de pulso (PWM, *Pulse Width Modulation*, por sus siglas en inglés). Esta técnica modifica el ciclo de trabajo de una señal periódica (cuadrada en este caso) para controlar la tensión media que se le aplica al motor NXT y así variar su velocidad, como se muestra en la Figura 3.6.

La corriente disponible en los puertos de salida es de aproximadamente 700mA, y puede alcanzar un pico de corriente de 1A. El controlador tiene una protección térmica, que reduce la corriente cuando se presenta un sobrecalentamiento.

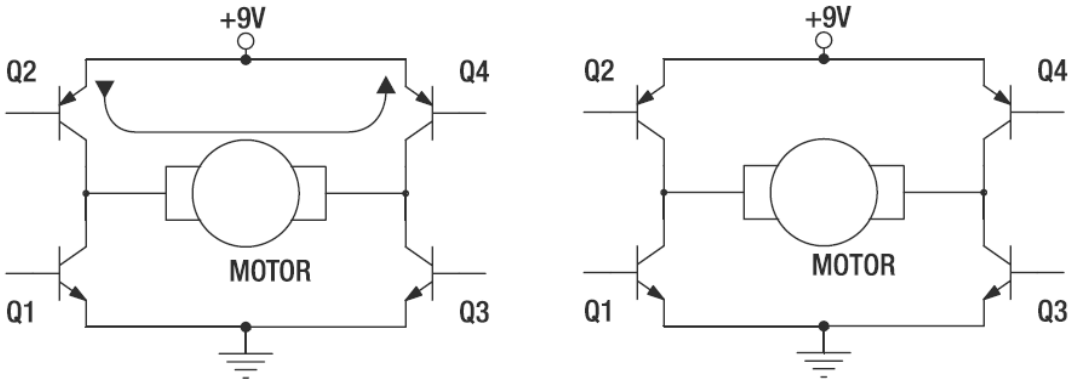
Pin 3 [GND]

Este es el conector de tierra para el codificador óptico.



(a) (b)

Figura 3.4: (a) Configuración para avance del motor; (b) Configuración para retroceso del motor. Imagen tomada de [35]



(a) (b)

Figura 3.5: (a) Configuración para frenado rápido del motor; (b) Configuración para frenado bajo la inercia del motor. Imagen tomada de [35]

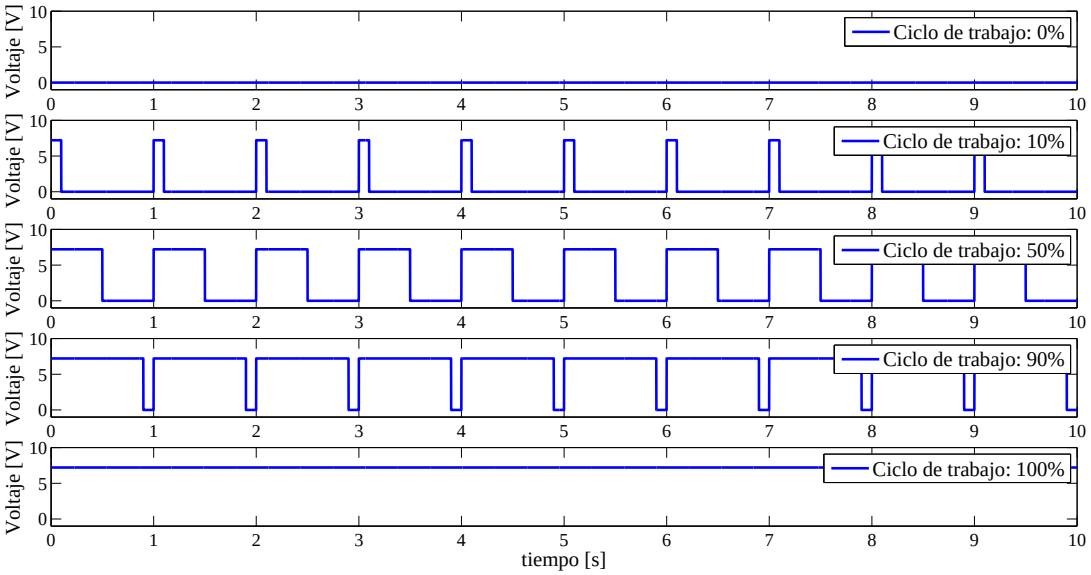


Figura 3.6: Ejemplo de diversos ciclos de trabajo para el Motor NXT.

Pin 4 [4.3V POWER]

Este pin está conectado a la fuente de alimentación de 4.3V que es compartida entre todos los puertos del NXT y se emplea para alimentar al codificador óptico en este caso.

Pines 5 y 6 [TACHO0 - TACHO1]

Las dos entradas se utilizan para la lectura del encoder óptico integrado en los motores NXT. El encoder genera señales en cuadratura, esto permite al NXT determinar la dirección y velocidad del motor. Las dos señales son pulsos rectangulares desfasados, el desfase representa un cuarto de la fase, de ahí el término de cuadratura. La Figura 3.7 ilustra las señales para un motor girando hacia adelante, y la Figura 3.8 para un motor girando hacia atrás. La frecuencia de la señal da la velocidad de rotación del motor. Un ciclo medio de la señal corresponde a un grado de rotación del motor.

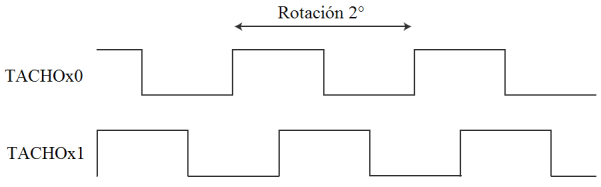


Figura 3.7: Señales en cuadratura para un motor en marcha hacia adelante. Imagen tomada de [35].

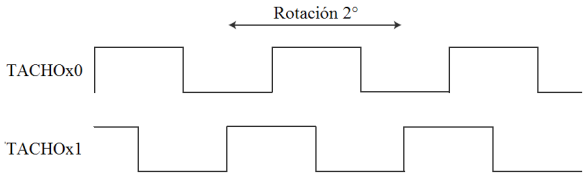


Figura 3.8: Señales en cuadratura para un motor en marcha hacia atrás. Imagen tomada de [35].

3.2.2. Sensores

El sistema NXT ballbot utiliza dos tipos de sensores: un sensor de desplazamiento angular y un sensor de velocidad angular.

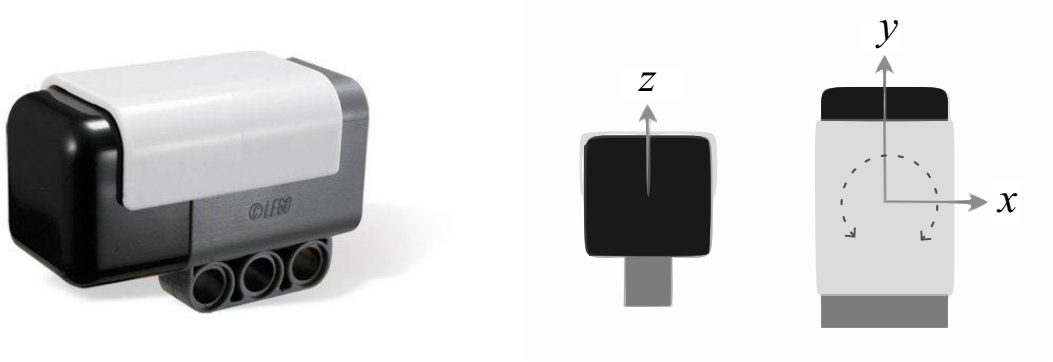
Sensor de desplazamiento angular

El sensor de desplazamiento angular utilizado es un encoder óptico en cuadratura, este sensor se localiza en el eje del motor NXT y mide el desplazamiento del NXT ballbot desde su posición inicial con una resolución de medición de 1°. El encoder cuenta con un diodo emisor y dos fotorreceptores. Uno de los fotorreceptores genera una onda cuadrada para llevar la cuenta de los giros del motor, mientras el otro fotorreceptor que se encuentra desfasado a un cuarto de fase con respecto al primero genera otra onda cuadrada que se utiliza para conocer el sentido de giro del motor. La Figura 3.9 muestra el encoder que posee el motor NXT.

Figura 3.9: Vista del encoder interno en el motor. Imagen tomada de [34].

Sensor de velocidad angular

El sensor de velocidad angular utilizado es un sensor de giro de la marca HiTechnic®, este sensor mide la velocidad angular de rotación del NXT ballbot en el plano $x-y$, así como la dirección de rotación. La Figura 3.10(a) muestra el sensor de giro de HiTechnic® utilizado. El sensor de HiTechnic® consta de un sensor giroscópico de un solo eje basado en un resonador de cuarzo, actuando en este caso sobre el eje z para darnos la velocidad de rotación en grados por segundo, como se muestra en la Figura 3.10(b). El sensor de giro puede medir rotaciones



(a) (b)

Figura 3.10: (a) Sensor de giro HiTechnic® utilizado en el sistema ballbot. Imagen tomada de http://www.hitechnic.com/cgi-bin/commerce.cgi?preadd=action&key=NGY1044; (b) Vista del sensor HiTechnic® en el plano $x-y$ y su eje de rotación z.

de hasta $\pm 360\frac{°}{s}$. El valor de salida nominal de velocidad angular cero es 620, por lo que la medida verdadera se obtiene restando 620 a la medida obtenida.

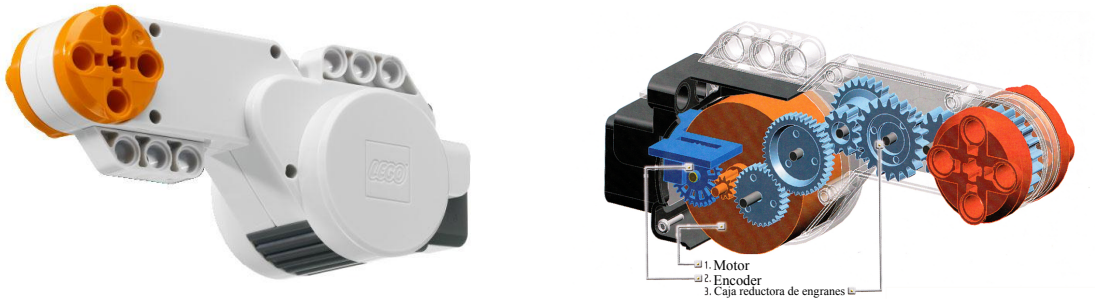
Para el sistema NXT ballbot se utilizan dos de estos sensores, uno para el plano $x-y$ y otro para el plano $z-y$. Cada sensor varía ligeramente en su valor de salida nominal cero, por lo que se recomienda realizar una calibración durante uno o dos segundos mientras el robot se encuentra estacionario, con esto se obtiene un valor de sesgo promedio. Cabe mencionar que incluso realizando esta promediación, el valor del cero nominal del sensor variará durante su operación, esto debido a diversos factores como: vibraciones provocadas por el accionamiento de los actuadores, la temperatura del lugar y el nivel de voltaje suministrado por el ladrillo NXT. El sensor de giro se conecta al ladrillo NXT utilizando la interfaz de sensor analógica, con una frecuencia de muestreo aproximadamente de 300Hz.

3.2.3. Actuadores

Los actuadores encargados de generar los pares de fuerza necesarios para equilibrar al sistema NXT ballbot, son dos motores de corriente contínua incluidos en el kit estándar de *LEGO Mindstorms NXT*®. Estos motores contienen en su interior una caja reductora de engranes con una relación de $n = 48$ para aumentar el par generado por el motor. La Figura 3.11(a) muestra uno de los motores empleados, así como en la Figura 3.11(b) se muestra una vista de los componentes del motor [34].

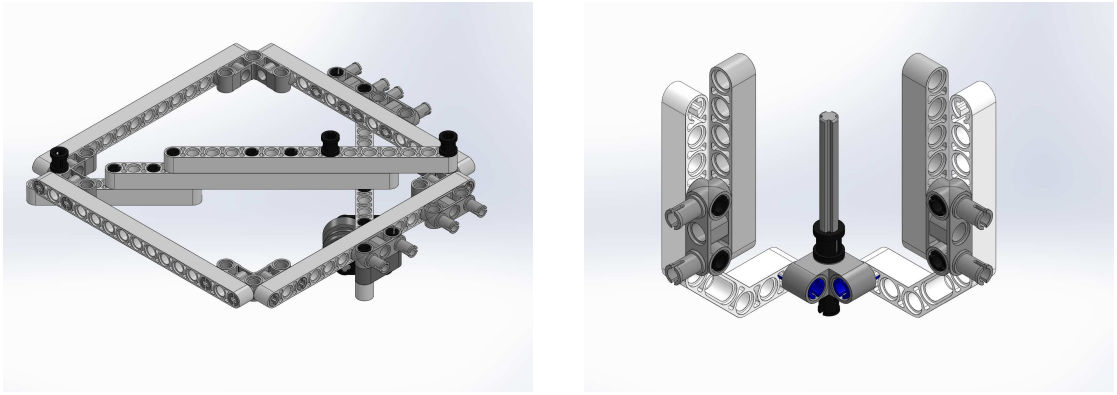
3.2.4. Estructura del sistema NXT ballbot

Mediante las instrucciones de ensamblado para el robot tipo Ballbot diseñado por Yorihisa Yamamoto [77], se han diseñado las piezas necesarias para realizar dicho ensamble en el software de CAD *Solidworks*®. Esto con el fin de obtener información mas completa acerca de los parámetros de la estructura como: volumen total, peso total del robot, centro de masas,



(a) (b)

Figura 3.11: (a) Motor de corriente contínua utilizado en el sistema ballbot; (b) Vista interna del encoder y caja reductora de engranes en el motor. Imagen tomada de [34].



(a) (b)

Figura 3.12: (a) Subensamblaje de la base inferior; (b) Subensamblaje de los cojinetes para los motores.

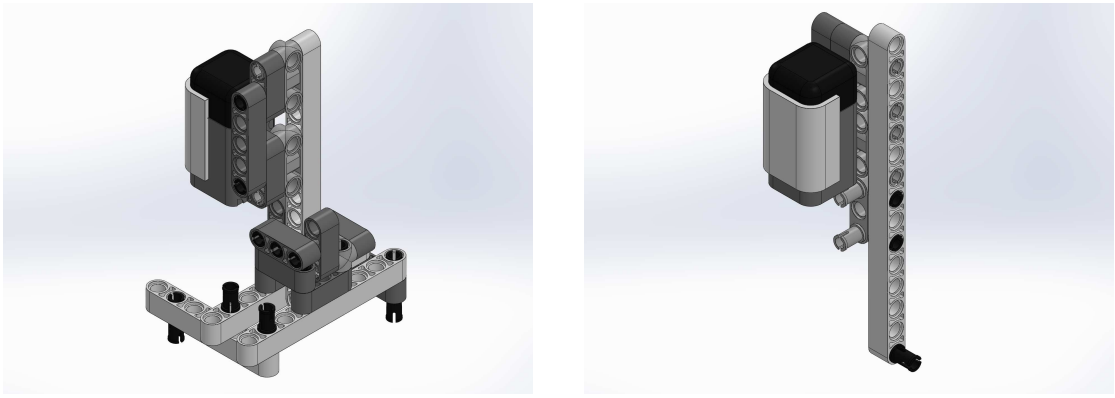
momentos de inercia, etc. Para el diseño de estas piezas se basó en una librería de piezas de *LEGO*® no oficial proporcionada en el sitio de la Universidad de Carnegie Mellon [73], un capítulo de libro sobre las piezas del *LEGO*® [62], así como la ayuda de un vernier electrónico [Mitutoyo Digimatic Vernier Caliper], una báscula electrónica de precisión [Sartorius BL210S] y una báscula electrónica [OHAUS ES Series]. Esto con el fin de comparar medidas y pesos de las piezas, y poder determinar la densidad de cada pieza a diseñar.

Una vez teniendo todas las piezas necesarias, se procedió a realizar el ensamblado del robot en *Solidworks*®. El ensamble completo del NXT ballbot se dividió en siete subensamblajes principales, como se muestra en las Figuras [3.12 - 3.15(a)]. Esto se hizo con el fin de poder realizar modificaciones en el diseño si así se requiriera en un futuro, como el tener una mejor claridad de la ubicación de los componentes que conforman la estructura del NXT ballbot.

Al unir los subensamblajes mostrados anteriormente, se obtiene la estructura del sistema

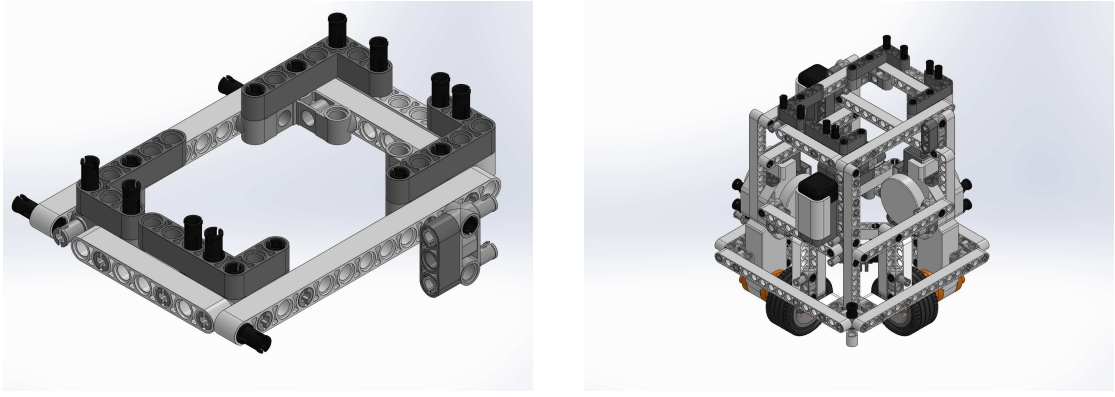
(a) (b)

Figura 3.13: (a) Subensamblaje del motor 1; (b) Subensamblaje del motor 2.



(a) (b)

Figura 3.14: (a) Subensamblaje del sensor 1; (b) Subensamblaje del sensor 2.



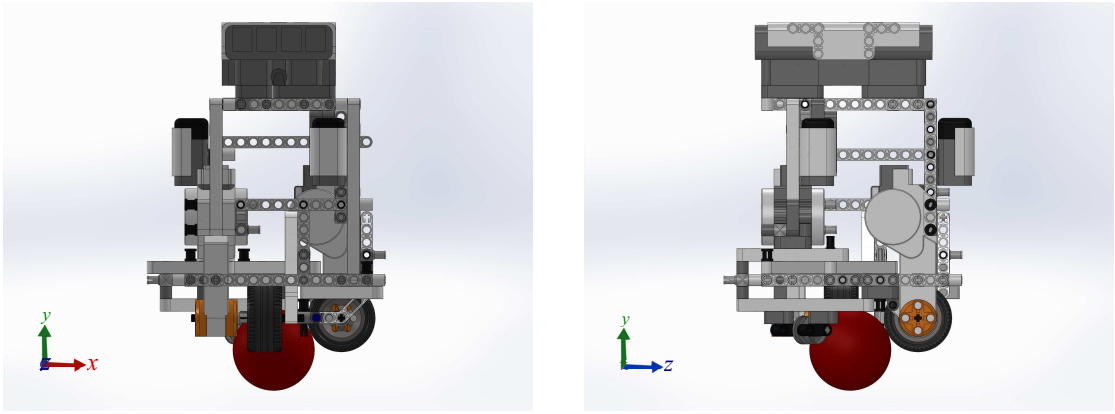
(a) (b)

Figura 3.15: (a) Subensamblaje de la base superior; (b) Ensamblaje de la estructura del sistema NXT ballbot.

(a) (b)

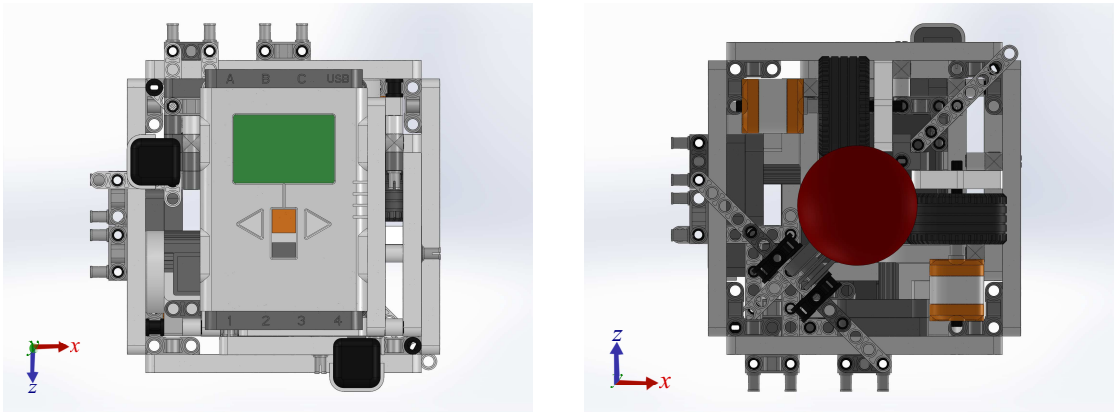
Figura 3.16: (a) Modelo completo del NXT ballbot en 3D diseñado en *Solidworks*®; (b) Render del modelo NXT ballbot en *Solidworks*®.

NXT ballbot como se muestra en la Figura 3.15(b). Por último se coloca el ladrillo NXT en la base superior y la pelota entre las llantas, con lo cual se obtiene el modelo completo en 3D del sistema NXT ballbot mostrado en la Figura 3.16(a). En las Figuras [3.17 - 3.18] se muestran diferentes vistas del modelo en *Solidworks*®. Posteriormente se realizó el ensamble físico del robot siguiendo las mismas instrucciones, las imágenes de este proceso se muestran en [78]. Para finalizar y ver que el modelo diseñado en *Solidworks*® es confiable respecto a las medidas y pesos del modelo real, se han pesado cada una de las piezas que lo integran como se muestra en la Figura 3.19(a), con estas medidas se determinó la densidad del material de cada pieza y se estableció en el modelo de *Solidworks*®. Al final se pesó el sistema completo y se determinó el peso total del modelo en *Solidworks*®, esto se hizo dirigiéndose a la pestaña de calcular y seleccionando la opción de propiedades físicas. En la Tabla 3.4 se resumen las piezas empleadas para la construcción del sistema NXT ballbot, así como su peso correspondiente. Con fines de comparación en las Figuras 3.20(b), 3.19(b) se muestran los parámetros físicos de una viga módulo de 7 y del ensamble completo del sistema en *Solidworks*®.



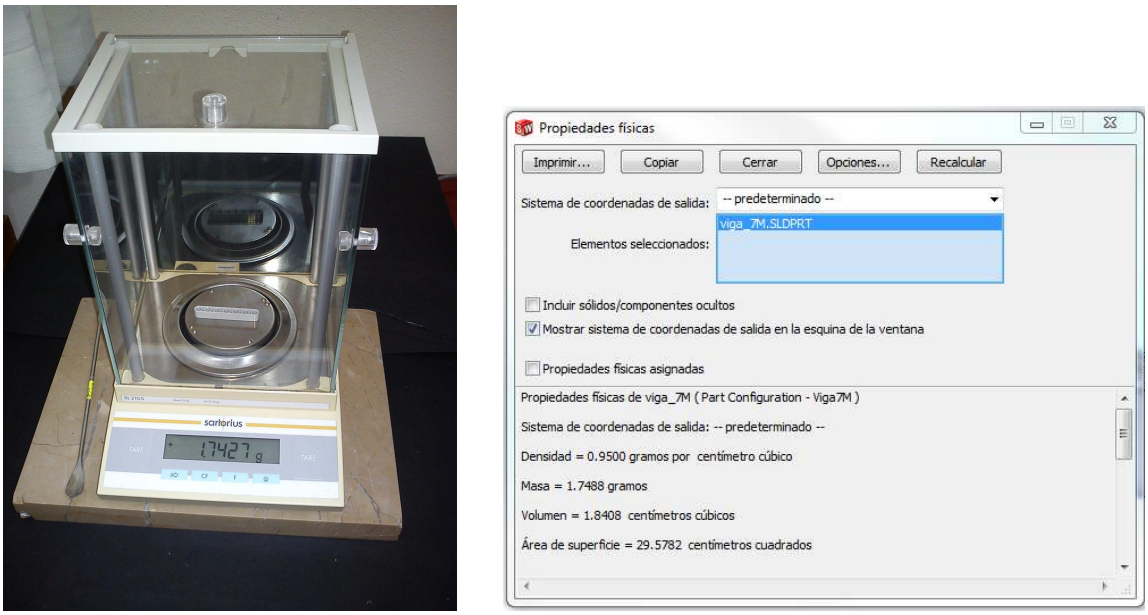
(a) (b)

Figura 3.17: (a) Vista frontal del modelo en el plano x-y de *Solidworks*®; (b) Vista lateral del modelo en el plano z-y de *Solidworks*®.



(a) (b)

Figura 3.18: (a) Vista superior del modelo en *Solidworks*®; (b) Vista inferior del modelo en *Solidworks*®.



(a) (b)

Figura 3.19: (a) Medición del peso de una pieza de la estructura; (b) Medición del peso de una pieza de la estructura en *Solidworks*®

(a) (b)

Figura 3.20: (a) Medición del peso total del NXT ballbot.; (b) Medición del peso total del NXT ballbot en *Solidworks*®.

Tabla 3.4: Piezas empleadas para la construcción del sistema NXT ballbot con su respectivo peso.

Pieza	**Referencia**	**peso [g]**
Ladrillo inteligente NXT*	9841-1	157.416
Motor NXT	9842-1	79.895
Sensor giroscópico *Hitechnic*®	MS1044-1	12.338
Rueda 30x20	56145	4.217
Rueda 18x14	55981	1.816
Neumático 43.2x22	55978	11.115
Pelota	41250	12.774
Eje, módulo de 4	3705	0.578
Eje, módulo de 6	3706	0.878
Eje, módulo de 10	3737	1.4980
Viga, módulo de 3	32523	0.688
Viga, módulo de 5	32316	1.261
Viga, módulo de 7	32524	1.743
Viga, módulo de 9	40490	2.589
Viga, módulo de 11	32525	3.1911
Viga, módulo de 13	32277	3.281
Viga, módulo de 15	32278	4.437
Viga angular, módulo de 3x5	32526	1.744
Viga angular, módulo de 3x7	32009	3.374
Conector	3673	0.151
Conector, módulo de 3	6558	0.285
Conector con cojinete	32054	0.319
Conector con eje	43093	0.255
Cojinete	3713	0.145
Conector perpendicular 2x2	44809	0.646
Conector perpendicular 3x3	55615	1.801
Conector doble, módulo de 3	48989	1.208
Bloque transversal, doble	32184	0.643

* Sin baterías.

3.3. Modelado dinámico de los actuadores

Como ya se mencionó, los actuadores utilizados son los motores de corriente contínua del kit de *LEGO Mindstorms NXT®*. El diagrama eléctrico y mecánico simplificado de un motor de corriente contínua se muestra en la Figura 3.21. En este diagrama intervienen los siguientes parámetros eléctricos: el voltaje de alimentación generado por el PWM del ladrillo NXT e_a, la resistencia de armadura del motor R_a, la inductancia de armadura del motor L_a, la fuerza contra electromotriz generada por el giro del rotor e_b, la corriente de armadura del motor i_a y una resistencia de medición tal que nos permita medir el valor de la corriente de armadura del motor R_{shunt}. Los parámetros mecánicos son: la posición angular del eje del motor θ_e, la posición angular del eje de salida a la salida de la caja reductora de engranes θ_m, el coeficiente de fricción viscosa b_m y el momento de inercia del motor J_m.

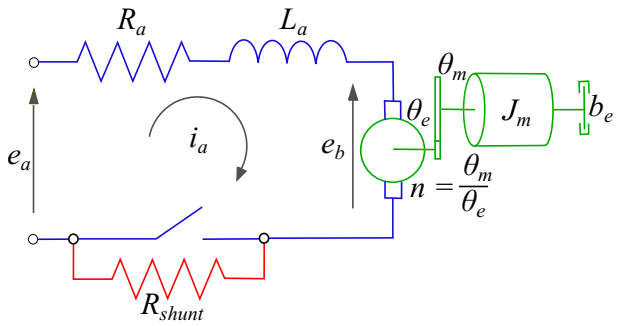


Figura 3.21: Diagrama eléctrico-mecánico de un motor de corriente contínua.

Las ecuaciones que describen la dinámica del motor se obtienen combinando las leyes de Kirchhoff y Newton. En primer lugar, aplicando la segunda ley de voltaje de Kirchhoff al circuito de armadura del motor se tiene que el voltaje de entrada es igual a

$$e_a = \frac{di_a}{dt} L_a + R_a i_a + e_b. \tag{3.3.1}$$

donde la fuerza contra electromotriz se expresa como

$$e_b = K_b \frac{d\theta_e}{dt}. \tag{3.3.2}$$

siendo K_b la constante de fuerza contra electromotriz o simplemente constante eléctrica.

Aplicando la segunda ley de Newton al movimiento rotacional se obtiene que el par del motor τ, es igual a

$$\tau = J \frac{d^2\theta_e}{dt^2} + b_e \frac{d\theta_e}{dt}. \tag{3.3.3}$$

Este par también puede expresarse en términos de la corriente de armadura del motor

como

$$\tau = K_p i_a. \tag{3.3.4}$$

siendo K_p la constante de par del motor. Por lo que es necesario determinar el valor de estos parámetros.

3.3.1. Diseño de la plataforma experimental

Para la obtención de los parámetros eléctricos y mecánicos del motor, se diseñó una plataforma experimental, para lo que se utilizó el siguiente material: medio metro de cable telefónico de 6 hilos, dos conectores telefónicos RJ-12, una pinza ponchadora para cableado estructurado, cuatro terminales (clemas) de 3 tornillos, un protoboard, una resistencia de 1Ω, un multímetro [Agilent 34401A], un osciloscopio [Agilent 54621A] y dos puntas para osciloscopio.

Como se muestra en la Tabla 3.3, la comunicación entre el ladrillo NXT y el motor se da mediante 6 cables de diferente color. Con la finalidad de realizar con mayor facilidad las mediciones de las señales entre estos dispositivos, se elaboró un par de cables semejantes a los del kit de *LEGO Mindstorms NXT*® y no tener que cortar uno de los cables originales del kit.

Para la elaboración de estos cables, se poncharon en los extremos del cable telefónico los conectores RJ-12. Se comparó con el cable original para asegurar la correcta ubicación de los cables en los conectores. Posteriormente se les removió la pestaña de enganche debido a que a pesar de que los puertos del ladrillo son similares a los conectores RJ-12, estos difieren en la ubicación y anchura de la pestaña de enganche, como se puede observar en la Figura 3.22 [35].

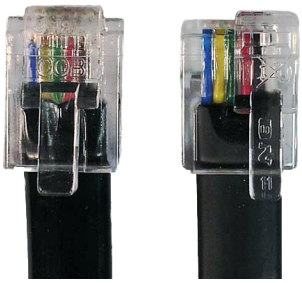
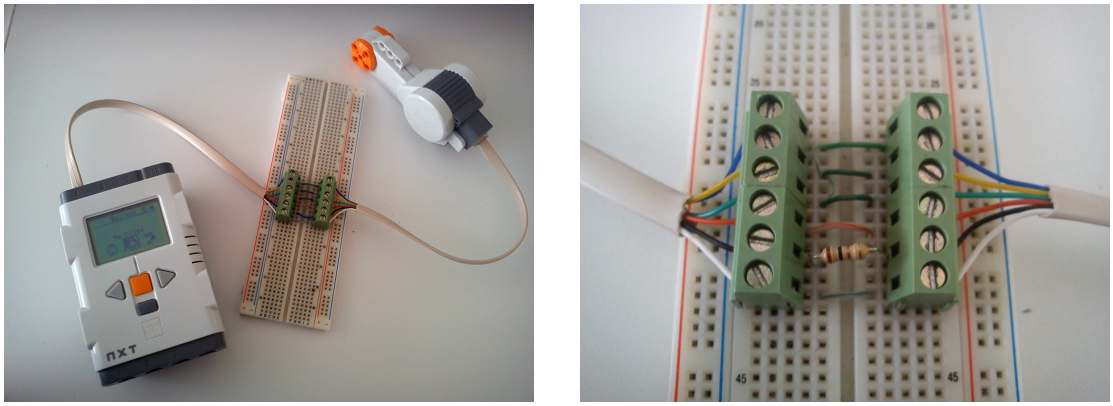


Figura 3.22: Comparación entre el conector telefónico RJ-12 a la izquierda y el conector de LEGO® a la derecha, donde se aprecia la ubicación y anchura de la pestaña de enganche . Imagen tomada de http://www.philohome.com/nxtplug/nxtplug.htm

Para embonar los nuevos conectores a los puertos del ladrillo NXT y del motor, estos se recubrieron con cinta adhesiva para que entraran a presión. A pesar de que para las mediciones que se realizaron, este método fue lo suficientemente aceptable, no se recomienda para pruebas que requieran de movimientos bruscos debido a que los conectores podrián desconectarse



(a) (b)

Figura 3.23: (a) Vista de la plataforma experimental y su conexión entre el ladrillo NXT y el motor; (b) Resistencia de medición R_{shunt} empleada para medir la corriente que circula por el motor.

fácilmente. Para otras versiones en el diseño de estos cables ver [35].

Una vez realizado estos pasos, el primer cable se conecta del ladrillo NXT a las terminales en el protoboard, estas terminales se conectan a unas segundas terminales debido a que después será necesario incluir una resistencia de medición en serie entre estas terminales. El segundo cable se conecta de las segundas terminales del protoboard al motor. La Figura 3.23(a) muestra la plataforma experimental. Cabe aclarar que solo se realizaron las mediciones en un motor, ya que se supuso que los parámetros del segundo motor, aunque no serán iguales, si serán bastante similares [55].

3.3.2. Parámetros eléctricos

El primer parámetro a determinar fue el valor de la resistencia de armadura R_a. Esta medición se realiza con el motor en vacío (sin alimentación) y midiendo con el multímetro entre los pines de alimentación M0 y M1. El valor obtenido fue de 4.6Ω.

El segundo parámetro fue la inductancia de armadura L_a. Para obtener dicho valor se incluyó una resistencia de medición R_{shunt} en serie con el pin M1, con el fin de medir la corriente i_a que circula por el motor. El valor de esta resistencia fue de 1Ω, como se aprecia en la Figura3.23(b).

Para esta prueba se eligieron las configuraciones de control para el motor que se muestran en las Figuras 3.4(a), 3.5(a). Estas configuraciones hacen girar al motor en sentido positivo (condición de avance), manteniendo siempre encendido el transistor Q2 y conmutando los transistores Q3 y Q4 en función del valor del PWM aplicado al puente H. Esto produce un voltaje en el inductor que va de un voltaje positivo a un voltaje 0 y viceversa. Así también se realizó un programa en *Simulink*®, el cual se encarga de hacer girar al motor dado un

determinado ciclo de trabajo en el PWM, para este caso se empleó un ciclo de trabajo de 50 %. El diagrama de bloques del programa en *Simulink*® utilizado se muestra en el Apéndice B.

Una vez descargado el programa al ladrillo NXT, se corre el programa y se espera a que el motor se encuentre operando en régimen permanente. En esta etapa se mide tanto el voltaje de alimentación del motor e_a entre los pines M0 y M1, como el voltaje en los extremos de la resistencia de medición. A este último valor le corresponde un valor equivalente a la corriente que circula por el motor al aplicar la ley de Ohm. La Figura 3.24 muestra el resultado de esta medición.

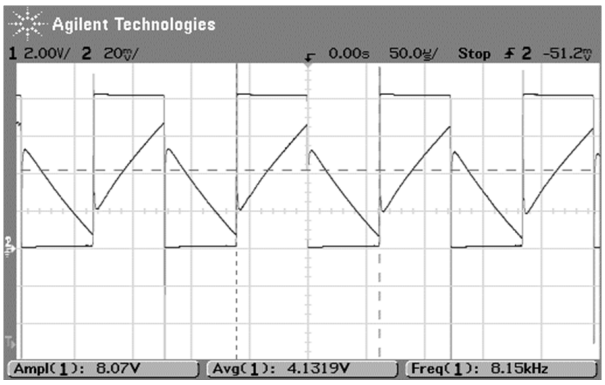


Figura 3.24: Mediciones del voltaje de entrada al motor y el voltaje en las terminales de la resistencia de medición, con un ciclo de trabajo en el PWM de 50 %.

Para tener un mejor manejo de los datos obtenidos, estos se exportaron del osciloscopio a un archivo en *Excel*®. Posteriormente, este archivo se importó a *Matlab*® y se diseñó un programa el cual solicita al usuario seleccionar un punto inicial y final de la gráfica de corriente, esto para determinar su pendiente y despejar el valor de la inductancia de armadura de la ecuación (3.3.1). El valor obtenido de la inductancia de armadura L_a fue de 5.27mH, como se observa en la Figura 3.25.

Por último, para determinar el valor de la constante eléctrica K_b del motor, se realizaron 10 pruebas donde se varió el ciclo de trabajo del PWM de 10 a 100 con un incremento del 10 %. Las mediciones que se realizaron fueron: el voltaje de alimentación del motor e_a, la corriente de armadura i_a, el periodo de la señal del encoder T_{enc}, la velocidad angular del motor ω, la fuerza contra electromotriz e_b y la constante eléctrica K_b correspondiente. Los valores de las mediciones obtenidas se resumen en la Tabla 3.5.

La velocidad del motor se obtiene a través de medir el periodo de la señal del encoder, la cual se midió entre los pines GND y TACHO0, esto debido a que no se tomó en cuenta la dirección de giro, por lo que se descartó la señal del pin TACHO1. Como un periodo de la señal del encoder corresponde a 2° de giro del motor, la velocidad angular del motor en

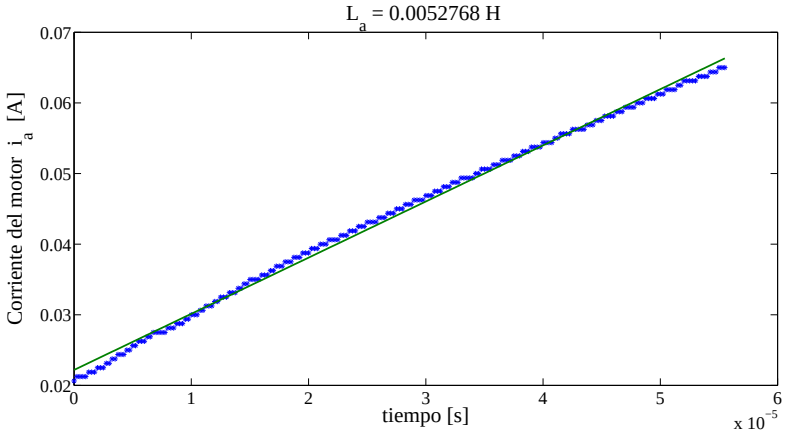


Figura 3.25: Valor de la Inductancia de armadura L_a obtenida, la pendiente de la corriente del inductor se determinó mediante un ajuste por mínimos cuadrados.

Tabla 3.5: Datos obtenidos mediante experimentación para la identificación de parámetros del motor.

PWM [%]	e_a $[V]$	i_a $[A]$	T_{enc} $[s]$	ω $[rad/s]$	e_b $[V]$	K_b $[Vs/rad]$
10	0.944	0.018347	0.0265	1.317229624	0.925653	0.702727135
20	1.7536	0.028887	0.011775	2.964465821	1.724713	0.581795542
30	2.5084	0.034259	0.00778	4.486707589	2.474141	0.551437987
40	3.3379	0.031824	0.00571	6.11323731	3.306076	0.540806095
50	4.1318	0.03634	0.004425	7.888493794	4.09546	0.519168818
60	4.9692	0.03782	0.00368	9.485485065	4.93138	0.519886961
70	5.7058	0.049639	0.003215	10.8574137	5.656161	0.520949202
80	6.5484	0.048575	0.00279	12.5113208	6.499825	0.519515493
90	7.3735	0.04803	0.00247	14.13222066	7.32547	0.518352365
100	8.22	0.050868	0.00223	15.65317715	8.169132	0.521883316

radianes se obtiene mediante la siguiente ecuación

$$\omega = \frac{2\pi}{T_{enc} * 180^{\circ}}. \tag{3.3.5}$$

El valor de la fuerza contra electromotriz e_b se calcula a partir de la ecuación (3.3.1), sin tomar en cuenta el término de la inductancia por considerarse despreciable. Con esta consideración, la fuerza contra electromotriz queda establecida por la siguiente ecuación

$$e_b = e_a - R_a i_a. \tag{3.3.6}$$

Por último, la constante eléctrica K_b para cada ciclo de trabajo se calculó mediante la ecuación (3.3.2). Como se observa en la Tabla 3.5 el valor de K_b difiere levemente en cada ciclo de trabajo, por lo que se realizó un ajuste por mínimos cuadrados. Para la realización de este ajuste se diseño un programa en *Matlab*® el cual importa los datos correspondientes a cada

ciclo de trabajo y los agrupa en vectores, posteriormente despeja los valores de ω y e_b para crear nuevos vectores a los cuales realiza un ajuste por mínimos cuadrados para encontrar el valor de K_b. La Figura 3.26 muestra el resultado del ajuste por mínimos cuadrados del valor de la constante eléctrica K_b con un valor de 0.49Vs. El código fuente de estos programas en *Matlab*® se encuentran en el Apéndice B.

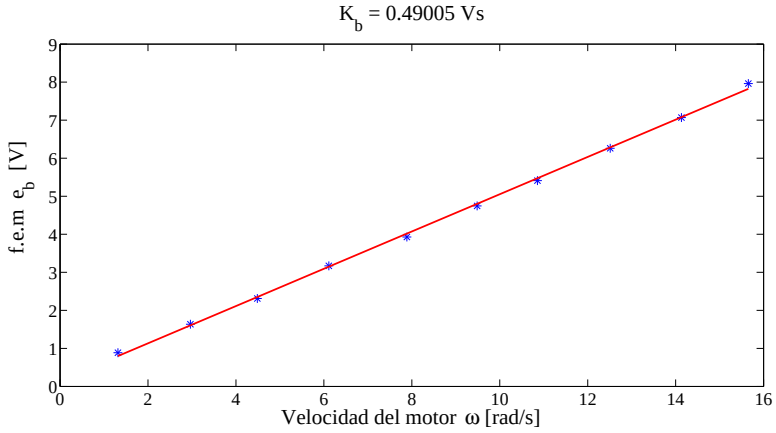


Figura 3.26: Valor de la constante eléctrica K_b obtenida, la pendiente de la fuerza contra electromotriz se determinó mediante un ajuste por mínimos cuadrados.

3.3.3. Parámetros mecánicos

El primer parámetro a determinar fue la constante de par K_p, como el sistema de unidades empleado es el sistema internacional (SI) y teniendo en cuenta que aplicando el principio de conservación de la energía en el caso de un motor de corriente directa nos dice que la potencia eléctrica de entrada es igual a la potencia mecánica de salida mas las perdidas generadas, se asume la hipótesis de que el valor de dicha constante de par y el valor de la constante eléctrica K_b determinada anteriormente son iguales, con esto se obtiene que K_p es igual a 0.49Nm/A. Este valor y las mediciones de la corriente de armadura obtenidas anteriormente para cada ciclo de trabajo del PWM se sustituyen en la ecuación (3.3.4) para determinar el par del motor, el cual a su vez se sustituye en la ecuación (3.3.3), donde como resultado de realizar las mediciones cuando el motor operaba en régimen permanente se obtuvo una velocidad angular constante y a su vez una aceleración angular cero. La ecuación (3.3.3) queda simplificada de la siguiente manera

$$K_p i_a = b_e \omega. \tag{3.3.7}$$

De esta ecuación se despeja b_e para cada ciclo de trabajo. Nuevamente, se diseñó un programa en *Matlab*® para realizar un ajuste por mínimos cuadrados obteniendo como resultado la gráfica de la Figura 3.27 donde b_e es igual a 0.914mNm, además se determinó el valor del par resistente T_r interno del motor como resultado de dicho ajuste.

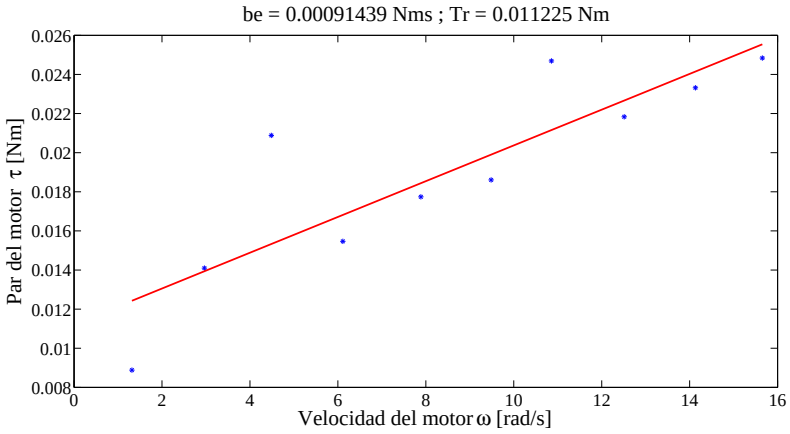


Figura 3.27: Gráfica de ajuste de la fricción viscosa b_e obtenida mediante mínimos cuadrados.

Como último parámetro queda el valor de la inercia del motor, para el cual se elije ahora la configuración de control del motor que muestran las Figuras 3.4(a), 3.5(b). Con esta configuración, cuando se le pida al motor dar un factor de servicio de 0 %, en vez de cerrar el circuito para recircular la corriente, este se detendrá bajo su propia inercia al dejar el circuito abierto.

Para ello, se estableció un ciclo de trabajo del 100 % para el PWM y una vez alcanzada su máxima velocidad se estableció un ciclo de trabajo del 0 % hasta que el motor se detuvo por completo. Una vez teniendo estos datos, de igual manera se realizó un ajuste por mínimos cuadrados para determinar la pendiente con la que decaía la velocidad del motor, con lo que se obtuvo el valor para la inercia del motor J_m de 0.00296kgm^2, como se observa en la Figura 3.28.

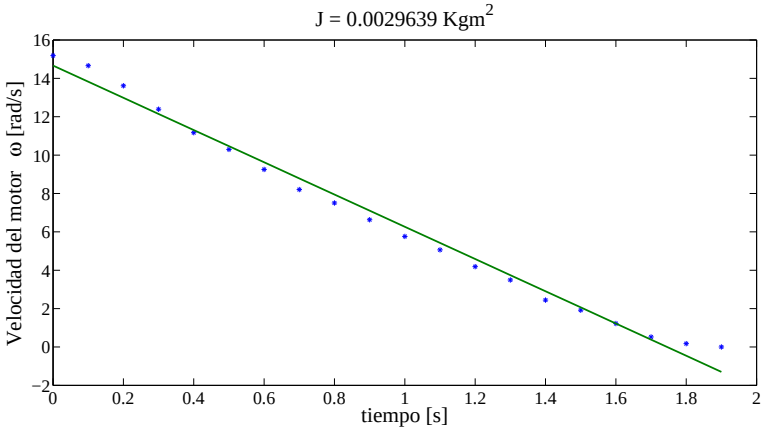


Figura 3.28: Gráfica de ajuste de la inercia del motor J_m obtenida mediante mínimos cuadrados.

Con estas pruebas se termina de obtener todos los parámetros necesarios para realizar el modelado matemático del sistema NXT ballbot completo.

3.4. Modelado dinámico del sistema NXT ballbot

El modelo del sistema NXT ballbot está basado en el modelo del péndulo invertido sobre una pelota esférica, se asume que el movimiento se realiza en el plano $x-y$ y en el plano $z-y$ los cuales están desacoplados y las ecuaciones de movimiento en esos dos planos son idénticas. Bajo esta suposición el sistema NXT ballbot puede ser considerado como dos partes separadas e idénticas para el modelo del péndulo invertido sobre la pelota esférica como se muestra en el diagrama de cuerpo libre de la Figura 3.29, donde además se presenta un sistema de coordenadas para el plano $x-y$. Así también se puede observar la llanta que utiliza el motor de corriente contínua, encargado de accionar el movimiento del ballbot y su relación con la pelota esférica. Se asume que no hay deslizamiento entre la pelota esférica y la llanta del motor.

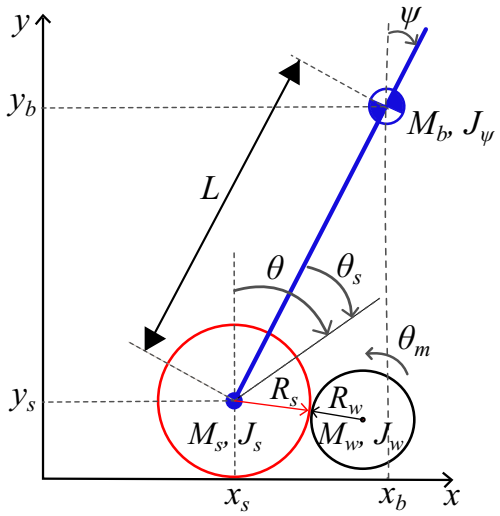


Figura 3.29: Diagrama de cuerpo libre del sistema NXT ballbot basado en el modelo del péndulo invertido sobre una esféra accionada por un motor y su sistema de coordenadas para el pano $x-y$.

De este sistema de coordenadas se observa que:

- ψ *: ángulo de inclinación del cuerpo con respecto a la vertical.*
- θ *: ángulo de la pelota esférica con respecto a la vertical.*
- θ_s *: ángulo de la pelota esférica accionada por la llanta del motor.*
- θ_m *: ángulo de la llanta accionada por el motor*

Como también las siguientes relaciones:

$$\theta = \psi + \theta_s, \tag{3.4.1}$$

$$R_\omega \theta_m = R_s \theta_s. \tag{3.4.2}$$

Las ecuaciones de movimiento del sistema se derivan mediante el método Euler-Lagrange haciendo uso de las coordenadas (x_s, y_s) y (x_b, y_b) establecidas en el sistema de la Figura 3.29. Así para el instante cuando $t = 0$ (tiempo inicial del sistema) y $\theta = 0$ (que representa al robot colocado verticalmente y sin movimiento) estas coordenadas se establecen de la siguiente manera:

$$\begin{aligned}
(x_s, y_s) &= (R_s\theta, y_s)\,, \\
(\dot{x_s}, \dot{y_s}) &= \left(R_s\dot{\theta}, 0\right), \\
(x_b, y_b) &= (x_s + L\sin\psi, y_s + L\cos\psi)\,, \\
(\dot{x_b}, \dot{y_b}) &= \left(R_s\dot{\theta} + L\dot{\psi}\cos\psi, -L\dot{\psi}\sin\psi\right).
\end{aligned}$$

En la mecánica clásica, el lagrangiano de un sistema conservativo se denota mediante $\mathcal{L}$, siendo este la diferencia entre la energía cinética total (T) y la energía potencial total (V) del sistema.

$$\mathcal{L} = T - V. \tag{3.4.3}$$

3.4.1. Ecuaciones de Euler-Lagrange

Del sistema representado en la Figura 3.29 se establece que la energía cinética total está constituida por la energía cinética traslacional (T_1) y la energía cinética rotacional (T_2), las cuales se definen como

$$T_1 = \frac{1}{2}M_s\left(\dot{x}_s^2 + \dot{y}_s^2\right) + \frac{1}{2}M_b\left(\dot{x}_b^2 + \dot{y}_b^2\right), \tag{3.4.4}$$

$$T_2 = \frac{1}{2}J_s\dot{\theta}^2 + \frac{1}{2}J_\psi\dot{\psi}^2 + \frac{1}{2}J_\omega\dot{\theta}_m^2 + \frac{1}{2}J_m\dot{\theta}_m^2. \tag{3.4.5}$$

La energía potencial total del sistema queda definida como

$$V = M_s g y_s + M_b g y_b. \tag{3.4.6}$$

Por lo que el lagrangiano del sistema queda expresado de la siguiente forma

$$\mathcal{L} = T_1 + T_2 - V. \tag{3.4.7}$$

A continuación se definen las coordenadas generalizadas de posición q_j , en este caso se elijen las coordenadas (θ,ψ), lo que nos da las siguientes ecuaciones de Euler-Lagrange

$$\frac{d}{dt}\left(\frac{\partial\mathcal{L}}{\partial\dot{\theta}}\right) - \frac{\partial\mathcal{L}}{\partial\theta} = F_\theta, \tag{3.4.8}$$

$$\frac{d}{dt}\left(\frac{\partial\mathcal{L}}{\partial\dot{\psi}}\right) - \frac{\partial\mathcal{L}}{\partial\psi} = F_\psi. \tag{3.4.9}$$

Para poder evaluar la ecuación del lagrangiano (3.4.3) en las ecuaciones anteriores, primero se establecen en términos de las coordenadas generalizadas (θ y ψ) las ecuaciones que conforman dicho lagrangiano (T_1, T_2 y V) a través de las coordenadas establecidas para el instante $t = 0$.

Definiendo T1 en términos de las coordenadas generalizadas tenemos

$$T_1 = \frac{1}{2}M_s R_s^2 \dot{\theta}^2 + \frac{1}{2}M_b[(R_s\dot{\theta} + L\dot{\psi}\cos\psi)^2 + (-L\dot{\psi}\sin\psi)^2].$$

Luego se desarrolla y agrupan términos

$$\begin{aligned} T_1 &= \frac{1}{2}M_s R_s^2 \dot{\theta}^2 + \frac{1}{2}M_b(R_s^2\dot{\theta}^2 + 2LR_s\dot{\theta}\dot{\psi}cos\psi + L^2\dot{\psi}^2\cos\psi^2 + L^2\dot{\psi}^2\sin\psi^2), \\ T_1 &= \frac{1}{2}M_s R_s^2 \dot{\theta}^2 + \frac{1}{2}M_b(R_s^2\dot{\theta}^2 + 2LR_s\dot{\theta}\dot{\psi}cos\psi + L^2\dot{\psi}^2(\cos\psi^2 + \sin\psi^2)). \end{aligned}$$

Finalmente, la energía cinética traslacional queda definida como

$$T_1 = \frac{1}{2}M_s R_s^2 \dot{\theta}^2 + \frac{1}{2}M_b(R_s^2\dot{\theta}^2 + 2LR_s\dot{\theta}\dot{\psi}cos\psi + L^2\dot{\psi}^2). \tag{3.4.10}$$

Para la energía cinética rotacional T_2, de la ecuación (3.4.2) se tiene que

$$\theta_m = \frac{R_s}{R_\omega}\theta_s.$$

y utilizando la ecuación (3.4.1)

$$\theta_s = \theta - \psi.$$

se sustituyen para obtener la siguiente ecuación

$$\begin{aligned} \theta_m &= \frac{R_s}{R_\omega}(\theta - \psi), \\ \dot{\theta}_m &= \frac{R_s}{R_\omega}(\dot{\theta} - \dot{\psi}), \\ \dot{\theta}_m^2 &= \frac{R_s^2}{R_\omega^2}(\dot{\theta} - \dot{\psi})^2. \end{aligned} \tag{3.4.11}$$

Sustituyendo la ecuación (3.4.11) en (3.4.5), finalmente se obtiene

$$\begin{aligned} T_2 &= \frac{1}{2}J_s\dot{\theta}^2 + \frac{1}{2}J_\psi\dot{\psi}^2 + \frac{1}{2}(J_m + J_\omega)\frac{R_s^2}{R_\omega^2}(\dot{\theta} - \dot{\psi})^2, \\ T_2 &= \frac{1}{2}J_s\dot{\theta}^2 + \frac{1}{2}J_\psi\dot{\psi}^2 + \frac{1}{2}(J_m + J_\omega)\frac{R_s^2}{R_\omega^2}(\dot{\theta}^2 - 2\dot{\theta}\dot{\psi} + \dot{\psi}^2). \end{aligned} \tag{3.4.12}$$

Ahora se define V en términos de las coordenadas generalizadas

$$V = M_s g y_s + M_b g(y_s + L\cos\psi). \tag{3.4.13}$$

Se evalúa la ecuación del lagrangiano (3.4.7) conformado por las ecuaciones (3.4.10), (3.4.12) y (3.4.13) en las ecuaciones de Euler-Lagrange (3.4.8) y (3.4.9), obteniéndose así

$$\begin{aligned}
\frac{\partial L}{\partial \theta} &= 0, \\
\frac{\partial L}{\partial \dot{\theta}} &= M_s R_s^2\dot{\theta} + M_b R_s^2\dot{\theta} + M_b R_s L\dot{\psi}\cos\psi + J_s\dot{\theta} + \frac{R_s^2}{R_\omega^2}(J_m + J_\omega)(\dot{\theta} - \dot{\psi}), \\
\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) &= R_s^2\ddot{\theta}(M_s + M_b) + M_b R_s L\ddot{\psi}\cos\psi + M_b R_s L\dot{\psi}(-\sin\psi)\dot{\psi} + J_s\ddot{\theta} + \\
&\quad \frac{R_s^2}{R_\omega^2}(J_m + J_\omega)(\ddot{\theta} - \ddot{\psi}), \\
\frac{\partial L}{\partial \psi} &= \frac{1}{2}M_b(-2R_s L\dot{\theta}\dot{\psi}\sin\psi) + M_b g L\sin\psi, \\
&= -M_b R_s L\dot{\theta}\dot{\psi}\sin\psi + M_b g L\sin\psi, \\
\frac{\partial L}{\partial \dot{\psi}} &= \frac{1}{2}M_b\left(2L^2\dot{\psi} + 2LR_s\dot{\theta}\cos\psi\right) + J_\psi\dot{\psi} + \frac{1}{2}\frac{R_s^2}{R_\omega^2}(J_m + J_\omega)(2\dot{\psi} - 2\dot{\theta}), \\
&= M_b\left(L^2\dot{\psi} + LR_s\dot{\theta}\cos\psi\right) + J_\psi\dot{\psi} + \frac{R_s^2}{R_\omega^2}(J_m + J_\omega)(\dot{\psi} - \dot{\theta}), \\
\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\psi}}\right) &= M_b(L^2\ddot{\psi} + LR_s\ddot{\theta}\cos\psi - LR_s\dot{\theta}\dot{\psi}\sin\psi) + J_\psi\ddot{\psi} + \frac{R_s^2}{R_\omega^2}(J_m + J_\omega)(\ddot{\psi} - \ddot{\theta}).
\end{aligned}$$

Al agrupar términos, la ecuación de Euler-Lagrange (3.4.8) se define como

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) - \frac{\partial L}{\partial \theta} = F_\theta,$$

$$\left[(M_s + M_b)R_s^2 + \frac{R_s^2}{R_\omega^2}(J_m + J_\omega) + J_s\right]\ddot{\theta} + \left[M_b R_s L\cos\psi - \frac{R_s^2}{R_\omega^2}(J_m + J_\omega)\right]\ddot{\psi} -$$

$$M_b R_s L\dot{\psi}^2\sin\psi = F_\theta. \tag{3.4.14}$$

La ecuación de Euler-Lagrange (3.4.9) se define como

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\psi}}\right) - \frac{\partial L}{\partial \psi} = F_\psi,$$

$$\left[M_b R_s L\cos\psi - \frac{R_s^2}{R_\omega^2}(J_m + J_\omega)\right]\ddot{\theta} + \left[M_b L^2 + J_\psi + \frac{R_s^2}{R_\omega^2}(J_m + J_\omega)\right]\ddot{\psi} - M_b g L\sin\psi = F_\psi. \tag{3.4.15}$$

Así las ecuaciones (3.4.14) y (3.4.15) describen la dinámica del sistema.

Para las fuerzas generalizadas F_θ y F_ψ, al considerar el par y la fricción viscosa del motor de corriente continua del kit de *LEGO Mindstorms NXT®* se obtienen las siguientes ecuaciones

$$F_\theta = k_p i_a - b_e\dot{\theta}_m, \tag{3.4.16}$$
$$F_\psi = -k_p i_a + b_e\dot{\theta}_m, \tag{3.4.17}$$

de donde $\dot{\theta}_m = \frac{R_s}{R_w}(\dot{\theta} - \dot{\psi})$ es la velocidad angular del motor.

Como se mencionó anteriormente, una característica del motor utilizado es que en lugar de corriente, este se controla a través del voltaje aplicado mediante la técnica PWM, por lo que de la ecuación (3.3.1) que describe la dinámica eléctrica del motor, se despeja i_a, y se considera el valor de la inductancia de armadura L_a despreciable, con lo que se obtiene la corriente del motor en términos del voltaje

$$\begin{aligned} i_a &= \frac{e_a - e_b}{R_a}, \\ i_a &= \frac{e_a - k_b\dot{\theta}_m}{R_a}, \\ i_a &= \frac{e_a - k_b k(\dot{\theta} - \dot{\psi})}{R_a}, \end{aligned} \tag{3.4.18}$$

donde $k = \frac{R_s}{R_w}$, sustituyendo la ecuación (3.4.18) en (3.4.16) y (3.4.17), se obtienen las fuerzas generalizadas en términos del voltaje

$$F_\theta = \alpha e_a - \beta(\dot{\theta} - \dot{\psi}), \tag{3.4.19}$$
$$F_\psi = -\alpha e_a + \beta(\dot{\theta} - \dot{\psi}), \tag{3.4.20}$$

donde $\alpha = \frac{k_t}{R_a}; \beta = k(\frac{k_p k_b}{R_a} + b_e)$.

3.4.2. Linealización del sistema NXT ballbot

Como se puede apreciar, las ecuaciones (3.4.14) y (3.4.15) son no lineales ya que las variables son argumentos de funciones trigonométricas o están elevadas al cuadrado. Por lo que se procede a linealizar al sistema alrededor de un punto de equilibrio. En este caso el punto

de equilibrio que nos interesa es cuando el sistema está completamente vertical ($\psi = 0$).

Si se considera el límite cuando $\psi \to 0$ entonces se tiene que ($\sin\psi \approx \psi$, $\cos\psi \approx 1$) y despreciando los términos de segundo orden como $\dot{\psi}^2 \approx 0$, las ecuaciones que rigen el comportamiento del sistema (3.4.14) y (3.4.15) quedan linealizadas alrededor del punto de equilibrio como

$$\left[(M_s + M_b)R_s^2 + \frac{R_s^2}{R_\omega^2}(J_m + J_\omega) + J_s\right]\ddot{\theta} + \left[M_b R_s L - \frac{R_s^2}{R_\omega^2}(J_m + J_\omega)\right]\ddot{\psi} = F_\theta, \tag{3.4.21}$$

$$\left[M_b R_s L - \frac{R_s^2}{R_\omega^2}(J_m + J_\omega)\right]\ddot{\theta} + \left[M_b L^2 + J_\psi + \frac{R_s^2}{R_\omega^2}(J_m + J_\omega)\right]\ddot{\psi} - M_b g L\psi = F_\psi. \tag{3.4.22}$$

3.5. Ecuaciones de estado para el sistema NXT ballbot

Una vez linealizado el modelo alrededor del punto de equilibrio cuando el límite $\psi \to 0$, las ecuaciones (3.4.21) y (3.4.22) se pueden escribir como

$$\mathbf{E}\begin{bmatrix}\ddot{\theta}\\ \ddot{\psi}\end{bmatrix} + \mathbf{F}\begin{bmatrix}\dot{\theta}\\ \dot{\psi}\end{bmatrix} + \mathbf{G}\begin{bmatrix}\theta\\ \psi\end{bmatrix} = \mathbf{H}e_a, \tag{3.5.1}$$

donde

- $\mathbf{E}$: Matriz de inercia,
- $\mathbf{F}$: Matriz de Coriolis y fuerzas centrífugas,
- $\mathbf{G}$: Matriz de energía potencial,
- $\mathbf{H}$: Matriz de torque de entrada,

quedando definidas de la siguiente manera

$$\mathbf{E} = \begin{bmatrix}(M_s + M_b)R_s^2 + k^2(J_m + J_\omega) + J_s & M_b R_s L - k^2(J_m + J_\omega)\\ L M_b R_s - k^2(J_m + J_\omega) & M_b L^2 + J_\psi + k^2(J_m + J_\omega)\end{bmatrix},$$
$$\mathbf{F} = \begin{bmatrix}\beta & -\beta\\ -\beta & \beta\end{bmatrix},$$
$$\mathbf{G} = \begin{bmatrix}0 & 0\\ 0 & -M_b g L\end{bmatrix},$$
$$\mathbf{H} = \begin{bmatrix}\alpha\\ -\alpha\end{bmatrix}.$$

Eligiendo el vector de estado $\mathbf{x}$ y el vector de entrada $\mathbf{u}$ como

$$\mathbf{x} = \begin{bmatrix} \theta & \psi & \dot{\theta} & \dot{\psi} \end{bmatrix}^T,$$
$$\mathbf{u} = e_a,$$

donde $\mathbf{x}^T$ indica la transpuesta de $\mathbf{x}$.

Para expresar las ecuaciones de estado del sistema NXT ballbot de la forma ((2.1.1)), es necesario determinar los elementos de las matrices involucradas, por lo que se despejan de (3.5.1) los estados correspondientes a cada elemento

$$\begin{aligned}
\mathbf{E}\begin{bmatrix}\ddot{\theta}\\ \ddot{\psi}\end{bmatrix} &= \mathbf{H}e_a - \mathbf{F}\begin{bmatrix}\dot{\theta}\\ \dot{\psi}\end{bmatrix} - \mathbf{G}\begin{bmatrix}\theta\\ \psi\end{bmatrix},\\
\begin{bmatrix}\ddot{\theta}\\ \ddot{\psi}\end{bmatrix} &= \mathbf{E}^{-1}\left[\mathbf{H}e_a - \mathbf{F}\begin{bmatrix}\dot{\theta}\\ \dot{\psi}\end{bmatrix} - \mathbf{G}\begin{bmatrix}\theta\\ \psi\end{bmatrix}\right],\\
\begin{bmatrix}\ddot{\theta}\\ \ddot{\psi}\end{bmatrix} &= \frac{1}{det(\mathbf{E})}\begin{bmatrix}E_{22} & -E_{12}\\ -E_{21} & E_{11}\end{bmatrix}\left[\mathbf{H}e_a - \begin{bmatrix}F_{11} & F_{12}\\ F_{21} & F_{22}\end{bmatrix}\begin{bmatrix}\dot{\theta}\\ \dot{\psi}\end{bmatrix} - \begin{bmatrix}G_{11} & G_{12}\\ G21 & G_{22}\end{bmatrix}\begin{bmatrix}\theta\\ \psi\end{bmatrix}\right],\\
\begin{bmatrix}\ddot{\theta}\\ \ddot{\psi}\end{bmatrix} &= \frac{1}{det(\mathbf{E})}\begin{bmatrix}E_{22} & -E_{12}\\ -E_{21} & E_{11}\end{bmatrix}\left[\mathbf{H}e_a - \begin{bmatrix}F_{11}\dot{\theta} + F_{12}\dot{\psi}\\ F_{21}\dot{\theta} + F_{22}\dot{\psi}\end{bmatrix} - \begin{bmatrix}0\\ G_{22}\psi\end{bmatrix}\right],\\
\begin{bmatrix}\ddot{\theta}\\ \ddot{\psi}\end{bmatrix} &= \frac{1}{det(\mathbf{E})}\begin{bmatrix}E_{22} & -E_{12}\\ -E_{21} & E_{11}\end{bmatrix}\left[\mathbf{H}e_a + \begin{bmatrix}-F_{11}\dot{\theta} - F_{12}\dot{\psi}\\ -F_{21}\dot{\theta} - F_{22}\dot{\psi} - G_{22}\psi\end{bmatrix}\right],\\
\begin{bmatrix}\ddot{\theta}\\ \ddot{\psi}\end{bmatrix} &= \frac{1}{det(\mathbf{E})}\begin{bmatrix}E_{22} & -E_{12}\\ -E_{21} & E_{11}\end{bmatrix}\left[\begin{bmatrix}-F_{11}\dot{\theta} - F_{12}\dot{\psi} + H_{11}e_a\\ -F_{21}\dot{\theta} - F_{22}\dot{\psi} - G_{22}\psi + H_{21}e_a\end{bmatrix}\right],\\
\begin{bmatrix}\ddot{\theta}\\ \ddot{\psi}\end{bmatrix} &= \frac{1}{det(\mathbf{E})}\begin{bmatrix}-E_{22}F_{11}\dot{\theta} - E_{22}F_{12}\dot{\psi} + E_{22}H_{11}e_a + E_{12}F_{21}\dot{\theta} + E_{12}F_{22}\dot{\psi} + \\ E_{12}G_{22}\psi - E_{12}H_{21}e_a\\ E_{21}F_{11}\dot{\theta} + E_{21}F_{12}\dot{\psi} - E_{21}H_{11}e_a - E_{11}F_{21}\dot{\theta} - E_{11}F_{22}\dot{\psi} - \\ E_{11}G_{22}\psi + E_{11}H_{21}e_a\end{bmatrix}.
\end{aligned}$$

Agrupando los términos por variables de estado obtenemos

$$\ddot{\theta} = \frac{1}{det(\mathbf{E})}\left[\left(E_{12}F_{21} - E_{22}F_{11}\right)\dot{\theta} + \left(E_{12}F_{22} - E_{22}F_{12}\right)\dot{\psi} + E_{12}G_{22}\psi + \left(E_{22}H_{11} - E_{12}H_{21}\right)e_a\right],$$
$$\ddot{\psi} = \frac{1}{det(\mathbf{E})}\left[\left(E_{21}F_{11} - E_{11}F_{21}\right)\dot{\theta} + \left(E_{21}F_{12} - E_{11}F_{22}\right)\dot{\psi} - E_{11}G_{22}\psi + \left(E_{11}H_{21} - E_{21}H_{11}\right)e_a\right],$$

de donde $\dot{x_3} = \ddot{\theta}$ y $\dot{x_4} = \ddot{\psi}$, ademas $E_{12} = E_{21}$, por lo que al agrupar términos, se obtienen las siguientes matrices

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{E_{12}G_{22}}{E_{11}E_{22}-E_{12}^2} & \frac{E_{12}F_{21}-E_{22}F_{11}}{E_{11}E_{22}-E_{12}^2} & \frac{E_{12}F_{22}-E_{22}F12}{E_{11}E_{22}-E_{12}^2} \\ 0 & \frac{-E_{11}G_{22}}{E_{11}E_{22}-E_{12}^2} & \frac{E_{21}F_{11}-E_{11}F_{21}}{E_{11}E_{22}-E_{12}^2} & \frac{E_{21}F_{12}-E_{11}F_{22}}{E_{11}E_{22}-E_{12}^2} \end{bmatrix},$$

$$\mathbf{B} = \begin{bmatrix} 0 \\ 0 \\ \frac{E_{22}H_{11}-E_{12}H_{21}}{E_{11}E_{22}-E_{12}^2} \\ \frac{E_{11}H_{21}-E_{21}H_{11}}{E_{11}E_{22}-E_{12}^2} \end{bmatrix}, \tag{3.5.2}$$

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{D} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

3.6. Modelo politópico del sistema NXT ballbot

Una vez que se cuenta con los valores de los parámetros físicos del motor empleado en el sistema NXT ballbot, estos datos se comparan con los reportados en la literatura. En la Tabla 3.6 se resumen los valores de los parámetros físicos del motor NXT obtenidos y los reportados en la literatura.

Tabla 3.6: Parámetros físicos del motor NXT reportados en la literatura.

Parámetro	**Experimental**	**Referencia [66]**	**Referencia [36]**	**Referencia [64]**	**Referencia [78]**
$R_a[\Omega]$	4,6	4,6	6,85	5,0012	6,69
$L_a[\mathrm{H}]$	$5{,}27 \times 10^{-3}$	$3{,}7 \times 10^{-3}$	$---$	1×10^{-3}	*despreciable*
$J_m[\mathrm{kgm}^2]$	$2{,}96 \times 10^{-3}$	45×10^{-9}	$---$	$2{,}4589 \times 10^{-6}$	10×10^{-6}
$b_e[\mathrm{Nms}]$	$0{,}914 \times 10^{-3}$	$1{,}85 \times 10^{-3}$	$1{,}1278 \times 10^{-3}$	$38{,}745 \times 10^{-6}$	$2{,}2 \times 10^{-3}$
$K_p[\frac{\mathrm{Nm}}{A}]$	0,49005	0,48	0,3179	0,5246	0,317
$K_b[\frac{\mathrm{Vs}}{rad}]$	0,49005	0,48	0,46389	0,5246	0,468

Una vez que se evalúan las variaciones en los parámetros y con el fin de obtener un sistema politópico simple pero representable, se elijen los dos parámetros con mayor desviación, estos son: la inercía del motor J y la fricción viscosa b_e. Por lo que se establece el siguiente vector

de parámetros inciertos

$$\rho = (b_e \quad J_m) \tag{3.6.1}$$

donde el número de vértices del polítopo resultante es $L = 4$, acotado por sus valores mínimo y máximo

$$b_e \in [b_{e\text{mín}} \quad b_{e\text{máx}}], \quad J_m \in [J_{m\text{mín}} \quad J_{m\text{máx}}]. \tag{3.6.2}$$

Como estos parámetros inciertos se presentan en las matrices $\mathbf{A}$ y $\mathbf{B}$, y estas a su vez dependen linealmente de dichos parámetros, es posible definir un politopo de 4 vértices en el cual se encuentran todos los posibles valores de los parámetros inciertos acotados entre sus valores mínimo y máximo. Antes de presentar los vértices del modelo politópico y para simplificar la notación, dado que las matrices $\mathbf{A}$ y $\mathbf{B}$ dependen linealmente de los elementos de las matrices $\mathbf{E}$ y $\mathbf{F}$, se presentan los valores mínimos y máximos de estas matrices en función de los parámetros inciertos.

$$\begin{aligned}
\mathbf{E}_{\text{mín}} &= \begin{bmatrix} (M_s + M_b)R_s^2 + k^2(J_{m\text{mín}} + J_\omega) + J_s & M_b R_s L - k^2(J_{m\text{mín}} + J_\omega) \\ LM_b R_s - k^2(J_{m\text{mín}} + J_\omega) & M_b L^2 + J_\psi + k^2(J_{m\text{mín}} + J_\omega) \end{bmatrix}, \\
\mathbf{E}_{\text{máx}} &= \begin{bmatrix} (M_s + M_b)R_s^2 + k^2(J_{m\text{máx}} + J_\omega) + J_s & M_b R_s L - k^2(J_{m\text{máx}} + J_\omega) \\ LM_b R_s - k^2(J_{m\text{máx}} + J_\omega) & M_b L^2 + J_\psi + k^2(J_{m\text{máx}} + J_\omega) \end{bmatrix}, \\
\mathbf{F}_{\text{mín}} &= \begin{bmatrix} k\left(\frac{b_{e\text{mín}} + (K_b K_p)}{R_a}\right) & -k\left(\frac{b_{e\text{mín}} + (K_b K_p)}{R_a}\right) \\ -k\left(\frac{b_{e\text{mín}} + (K_b K_p)}{R_a}\right) & k\left(\frac{b_{e\text{mín}} + (K_b K_p)}{R_a}\right) \end{bmatrix}, \\
\mathbf{F}_{\text{máx}} &= \begin{bmatrix} k\left(\frac{b_{e\text{máx}} + (K_b K_p)}{R_a}\right) & -k\left(\frac{b_{e\text{máx}} + (K_b K_p)}{R_a}\right) \\ -k\left(\frac{b_{e\text{máx}} + (K_b K_p)}{R_a}\right) & k\left(\frac{b_{e\text{máx}} + (K_b K_p)}{R_a}\right) \end{bmatrix}.
\end{aligned} \tag{3.6.3}$$

A partir de los elementos de estas matrices, se definen entonces los 4 vértices del modelo

politópico para las matrices $\mathbf{A}$ y $\mathbf{B}$

$$
\begin{aligned}
\mathbf{A}_1 &= \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{E_{\text{mín}12}G_{22}}{E_{\text{mín}11}E_{\text{mín}22}-E_{\text{mín}12}^2} & \frac{E_{\text{mín}12}F_{\text{mín}21}-E_{\text{mín}22}F_{\text{mín}11}}{E_{\text{mín}11}E_{\text{mín}22}-E_{\text{mín}12}^2} & \frac{E_{\text{mín}12}F_{\text{mín}22}-E_{\text{mín}22}F_{\text{mín}12}}{E_{\text{mín}11}E_{\text{mín}22}-E_{\text{mín}12}^2} \\ 0 & \frac{-E_{\text{mín}11}G_{22}}{E_{\text{mín}11}E_{\text{mín}22}-E_{\text{mín}12}^2} & \frac{E_{\text{mín}21}F_{\text{mín}11}-E_{\text{mín}11}F_{\text{mín}21}}{E_{\text{mín}11}E_{\text{mín}22}-E_{\text{mín}12}^2} & \frac{E_{\text{mín}21}F_{\text{mín}12}-E_{\text{mín}11}F_{\text{mín}22}}{E_{\text{mín}11}E_{\text{mín}22}-E_{\text{mín}12}^2} \end{bmatrix}, \\
\mathbf{B}_1 &= \begin{bmatrix} 0 \\ 0 \\ \frac{E_{\text{mín}22}H_{11}-E_{\text{mín}12}H_{21}}{E_{\text{mín}11}E_{\text{mín}22}-E_{\text{mín}12}^2} \\ \frac{E_{\text{mín}11}H_{21}-E_{\text{mín}21}H_{11}}{E_{\text{mín}11}E_{\text{mín}22}-E_{\text{mín}12}^2} \end{bmatrix}, \\
\mathbf{A}_2 &= \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{E_{\text{mín}12}G_{22}}{E_{\text{mín}11}E_{\text{mín}22}-E_{\text{mín}12}^2} & \frac{E_{\text{mín}12}F_{\text{máx}21}-E_{\text{mín}22}F_{\text{máx}11}}{E_{\text{mín}11}E_{\text{mín}22}-E_{\text{mín}12}^2} & \frac{E_{\text{mín}12}F_{\text{máx}22}-E_{\text{mín}22}F_{\text{máx}12}}{E_{\text{mín}11}E_{\text{mín}22}-E_{\text{mín}12}^2} \\ 0 & \frac{-E_{\text{mín}11}G_{22}}{E_{\text{mín}11}E_{\text{mín}22}-E_{\text{mín}12}^2} & \frac{E_{\text{mín}21}F_{\text{máx}11}-E_{\text{mín}11}F_{\text{máx}21}}{E_{\text{mín}11}E_{\text{mín}22}-E_{\text{mín}12}^2} & \frac{E_{\text{mín}21}F_{\text{máx}12}-E_{\text{mín}11}F_{\text{máx}22}}{E_{\text{mín}11}E_{\text{mín}22}-E_{\text{mín}12}^2} \end{bmatrix}, \\
\mathbf{B}_2 &= \mathbf{B}_1, \\
\mathbf{A}_3 &= \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{E_{\text{máx}12}G_{22}}{E_{\text{máx}11}E_{\text{máx}22}-E_{\text{máx}12}^2} & \frac{E_{\text{máx}12}F_{\text{mín}21}-E_{\text{máx}22}F_{\text{mín}11}}{E_{\text{máx}11}E_{\text{máx}22}-E_{\text{máx}12}^2} & \frac{E_{\text{máx}12}F_{\text{mín}22}-E_{\text{máx}22}F_{\text{mín}12}}{E_{\text{máx}11}E_{\text{máx}22}-E_{\text{máx}12}^2} \\ 0 & \frac{-E_{\text{máx}11}G_{22}}{E_{\text{máx}11}E_{\text{máx}22}-E_{\text{máx}12}^2} & \frac{E_{\text{máx}21}F_{\text{mín}11}-E_{\text{máx}11}F_{\text{mín}21}}{E_{\text{máx}11}E_{\text{máx}22}-E_{\text{máx}12}^2} & \frac{E_{\text{máx}21}F_{\text{mín}12}-E_{\text{máx}11}F_{\text{mín}22}}{E_{\text{máx}11}E_{\text{máx}22}-E_{\text{máx}12}^2} \end{bmatrix}, \\
\mathbf{B}_3 &= \begin{bmatrix} 0 \\ 0 \\ \frac{E_{\text{máx}22}H_{11}-E_{\text{máx}12}H_{21}}{E_{\text{máx}11}E_{\text{máx}22}-E_{\text{máx}12}^2} \\ \frac{E_{\text{máx}11}H_{21}-E_{\text{máx}21}H_{11}}{E_{\text{máx}11}E_{\text{máx}22}-E_{\text{máx}12}^2} \end{bmatrix}, \\
\mathbf{A}_4 &= \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{E_{\text{máx}12}G_{22}}{E_{\text{máx}11}E_{\text{máx}22}-E_{\text{máx}12}^2} & \frac{E_{\text{máx}12}F_{\text{máx}21}-E_{\text{máx}22}F_{\text{máx}11}}{E_{\text{máx}11}E_{\text{máx}22}-E_{\text{máx}12}^2} & \frac{E_{\text{máx}12}F_{\text{máx}22}-E_{\text{máx}22}F_{\text{máx}12}}{E_{\text{máx}11}E_{\text{máx}22}-E_{\text{máx}12}^2} \\ 0 & \frac{-E_{\text{máx}11}G_{22}}{E_{\text{máx}11}E_{\text{máx}22}-E_{\text{máx}12}^2} & \frac{E_{\text{máx}21}F_{\text{máx}11}-E_{\text{máx}11}F_{\text{máx}21}}{E_{\text{máx}11}E_{\text{máx}22}-E_{\text{máx}12}^2} & \frac{E_{\text{máx}21}F_{\text{máx}12}-E_{\text{máx}11}F_{\text{máx}22}}{E_{\text{máx}11}E_{\text{máx}22}-E_{\text{máx}12}^2} \end{bmatrix}, \\
\mathbf{B}_4 &= \mathbf{B}_3.
\end{aligned}
\tag{3.6.4}
$$

Este modelo se utilizará para encontrar una ley de control que sea robusto ante estas incertidumbres en los parámetros. Cabe recalcar que se pueden introducir más parámetros con incertidumbres, pero además de incrementar el número de vértices del politopo y debido a que algunos parámetros dentro de las matrices son no lineales, es necesario reformular dichos parámetros mediante transformaciones fraccionales lineales.

Capítulo 4

Control del sistema NXT ballbot

En este capítulo se describirá la interacción del sistema NXT ballbot y el paquete de programación *Matlab-Simulink®* en la implementación de los controladores. Posteriormente, se diseñarán dos controladores para el sistema NXT ballbot con el objetivo principal de mantenerlo estable en su posición vertical. Los dos primeros controladores, los cuales servirán de referencia para comparar el desempeño de los dos controladores a diseñar son los propuestos en la literatura en [78] basado en un controlador de tipo LQR y en [32] basado en un controlador mediante la técnica de asignación de polos en lazo cerrado.

4.1. Interacción sistema NXT ballbot y *Matlab®- Simulink®*

El paquete de programación a emplear para el diseño de los controladores es *Matlab®*, así como también se hará uso de *Simulink®*, donde se llevará a cabo la simulación e implementación del controlador en el sistema NXT ballbot. Para llevar a cabo la interacción entre el sistema NXT ballbot y *Simulink®* es necesario descargar el complemento para la biblioteca de funciones de *Simulink®* denominado *Support Package for LEGO MINDSTORMS NXT Hardware*. Este complemento se encuentra disponible a partir de la versión R2012a.

Una vez que se descarga e instala el complemento, el programa solicita al usuario si desea actualizar la versión en el controlador del ladrillo NXT, ademas de asignarle un nombre al ladrillo el cual servirá para identificarlo durante la comunicación *bluetooth*, durante este proceso se requiere conectar el ladrillo NXT a la computadora. En el Apéndice A se detalla el procedimiento de instalación del complemento y actualización de la versión del ladrillo NXT.

La Figura 4.1 muestra los bloques que contiene el complemento descargado y que permiten interaccionar con los elementos del kit *LEGO Mindstorms NXT®*, como también con los demás bloques de la biblioteca de funciones de *Simulink®*.

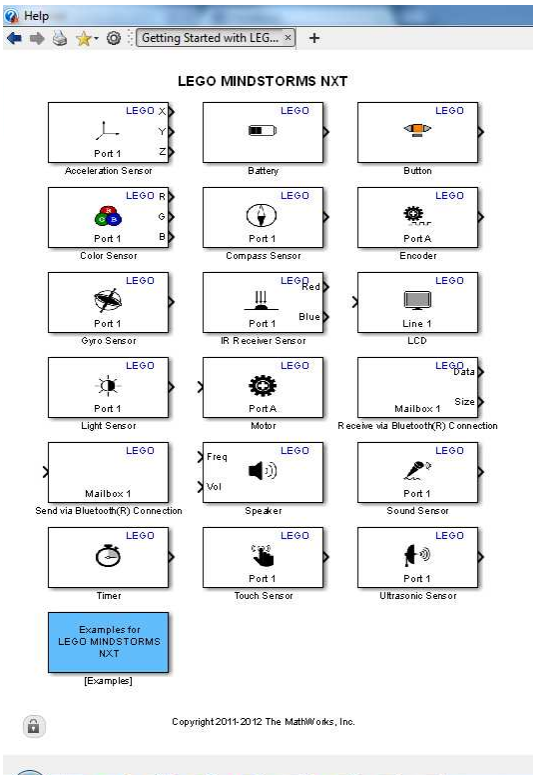


Figura 4.1: Bloques de la biblioteca de funciones instalada en Simulink

4.1.1. Comunicación *bluetooth* (NXT-PC)

Una de las ventajas de *Simulink*® es la capacidad que ofrece de monitorear e interactuar en tiempo real con los datos de los sensores y actuadores del sistema *LEGO Mindstorms NXT*®. Esto se logra mediante una comunicación *bluetooth* entre el ladrillo NXT y la computadora. En el Apéndice A se muestra el proceso de configuración de la comunicación *bluetooth* entre estos dispositivos. Una vez establecida la configuración, se ejecuta el programa en modo *External Mode* y *Simulink*® se encarga de compilar y generar el código ejecutable. La descarga del código ejecutable al ladrillo NXT se da mediante dos opciones, mediante el cable *usb* o mediante la conexión *bluetooth*, esta última tiene el inconveniente de ser mas lenta para códigos de gran tamaño.

4.1.2. Diagrama a bloques del sistema NXT ballbot

El esquema general de los controladores a diseñar se basa en la retroalimentación de estados mas un término integral del error del estado x_1 que en nuestro caso se refiere a la posición agular del sistema NXT ballbot θ, como se muestra en la Figura 4.2. Este esquema se conoce también como servo controlador, el término integral se añade con el fin de que el sistema NXT ballbot sea capaz de seguir una trayectoria ademas del equilibrio vertical. Este esquema se implementa mediante bloques de *Simulink*®, donde los estados del sistema NXT ballbot se determinan mediante un bloque que representa la dinámica de la planta para el

caso de simulación o un bloque que contiene los sensores y actuadores del sistema para el caso de la implementación física. Estos bloques se muestran en la Figura 4.3 , así como un cuarto bloque donde se monitorean y guardan las variables del sistema. En el Apéndice D se muestran a detalle los sub bloques y elementos que contienen cada uno de estos bloques principales.

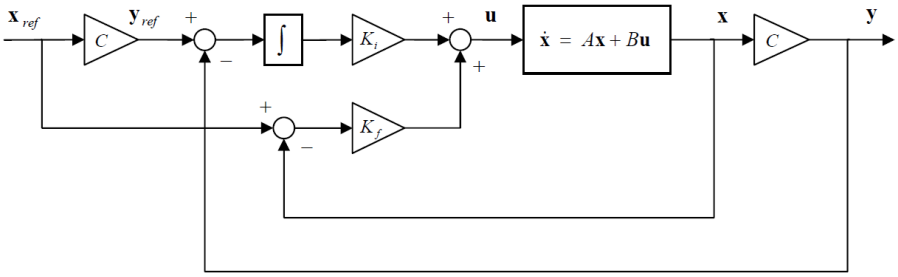


Figura 4.2: Esquema general del servo controlador. Imagen tomada de [78].

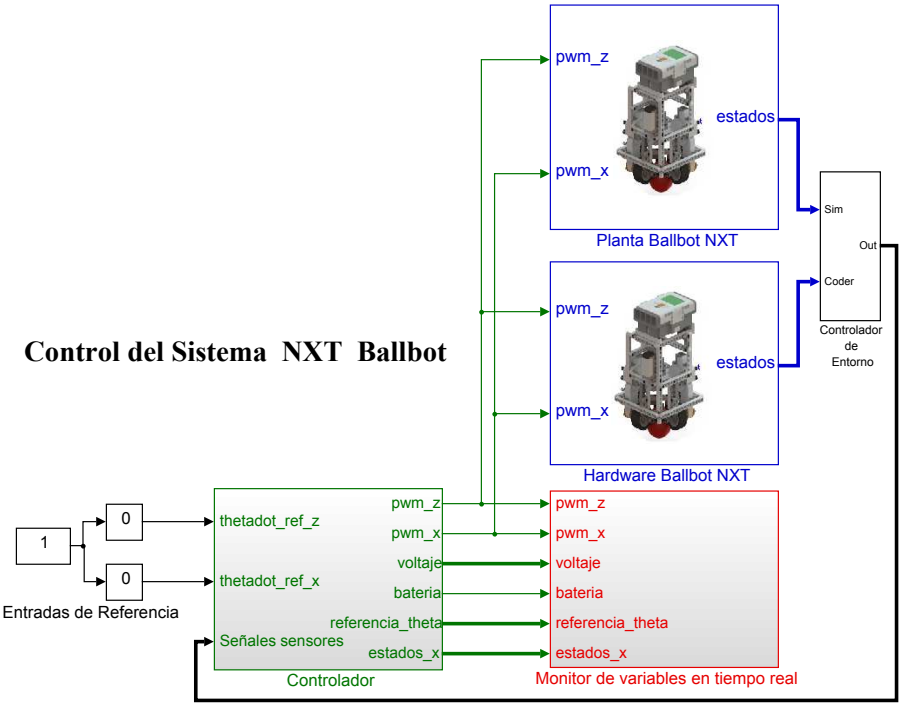


Figura 4.3: Bloques principales del control del sistema NXT ballbot implementado en *Simulink*®.

Al ejecutar el programa en *Simulink*®, para generar el código ejecutable es necesario contar con las ganancias del controlador y los parámetros del sistema y simulación, los cuales carga directamente del archivo creado en *Matlab*®. Al terminar la simulación o prueba experimental los datos de los estados son guardados en el espacio de trabajo de *Matlab*®.

4.2. Análisis del modelo NXT ballbot

En esta sección se presentan los resultados de estabilidad, controlabilidad y observabilidad del sistema NXT ballbot en lazo abierto. Para esto se diseña un programa en *Matlab*® donde se introducen los parámetros del sistema NXT ballbot y se calculan las matrices de estado correspondientes que servirán para crear el modelo del sistema en espacio de estados empleando la función *ss*. A continuación se muestra parte del código para crear el modelo en espacio de estados del sistema NXT ballbot.

Código fuente 4.1: Creación del modelo en espacio de estados del sistema NXT ballbot.

```
estados = {'\theta' '\psi' 'd\theta/dt' 'd\psi/dt' };
entrada = {'u'};
salidas = {'\theta' '\psi' 'd\theta/dt' 'd\psi/dt' };
sys_ss = ss(A,B,C,D,...
            'statename',estados,'inputname',entrada,'outputname',salidas);
```

Dado que durante el desarrollo de esta tesis se obtuvieron los parámetros del sistema mediante experimentación y uso de software para dibujo en 3D *Solidworks*®, se crearán dos modelos del sistema, el primer modelo incluirá los parámetros del sistema reportados en [78], mientras el segundo modelo incluirá los parámetros que se obtuvieron durante el modelado del sistema en esta tesis. Así mismo, cuando se haga referencia al sistema completo (abarcando los modelos 1 y 2) se denominará sistema NXT ballbot mientras que se referirá como modelo 1 y 2 según sea el caso que se analice del sistema NXT ballbot por separado.

4.2.1. Estabilidad

La estabilidad del sistema NXT ballbot en lazo abierto se evalúa mediante la función *pole* , esta función devuelve el valor de los polos del sistema como se muestra a continuación.

Código fuente 4.2: Polos de los modelos 1 y 2 del sistema en lazo abierto.

```
>> polos_lazoabierto = pole(sys_ss1)

polos_lazoabierto =

         0
 -241.8164
    6.1020
   -5.6754

>> polos_lazoabierto = pole(sys_ss2)

polos_lazoabierto =

         0
    5.9600
   -5.7222
```

```
  -14.0300
```

Como se observa, en ambos modelos un polo se encuentra en el semiplano derecho del plano complejo, por lo tanto el sistema NXT ballbot en lazo abierto es inestable en el punto de operación a partir del cual se linealizó el modelo.

4.2.2. Controlabilidad

Para comenzar a diseñar un controlador que haga que el sistema se estabilice, es necesario evaluar la controlabilidad del sistema. Haciendo uso de las funciones *ctrb* y *rank* de matlab, se evalúa el rango de la matriz de controlabilidad de los modelos 1 y 2 del sistema. A continuación se muestra el resultado obtenido en *Matlab*®.

Código fuente 4.3: Controlabilidad de los modelos 1 y 2 del sistema NXT ballbot.

```
>> controlabilidad = rank(ctrb(sys_ss1))

controlabilidad =

     4

>> controlabilidad = rank(ctrb(sys_ss2))

controlabilidad =

     4
```

Como ambas matrices de controlabilidad son de rango completo $n = 4$, el sistema es completamente controlable por lo que es posible diseñar un controlador que logre estabilizar al sistema NXT ballbot.

4.2.3. Observabilidad

Del mismo modo se evalúa la matriz de observabilidad de los modelos del sistema mediante el uso de la función *obsv*. Como resultado se obtiene que las matrices son de rango completo $n = 4$ por lo tanto el sistema es completamente observable.

Código fuente 4.4: Observabilidad de los modelos 1 y 2 del sistema NXT ballbot.

```
>> observabilidad = rank(obsv(sys_ss1))

observabilidad =

     4

>> observabilidad = rank(obsv(sys_ss2))

observabilidad =
```

4

4.3. Control LQR

El controlador propuesto por Yorihisa Yamamoto en [78] se basa en un controlador tipo LQR. Con el fin de que el NXT ballbot pueda seguir una trayectoria deseada, se agregó una nueva variable de estado ξ correspondiente a la integral del error de posición, esto con el fin de que el error en estado estacionario sea cero, quedando la ecuación de estado de la siguiente manera

$$\dot{\mathbf{x}}_a = \hat{\mathbf{A}}\mathbf{x} + \hat{\mathbf{B}}\mathbf{u} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mathrm{r},$$

con

$$\mathbf{x}_a = \begin{bmatrix} \mathbf{x} & \xi \end{bmatrix}^T.$$

donde r representa el valor de la referencia a seguir. Estas matrices se determinan como

$$\begin{aligned} \hat{\mathbf{A}} &= \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ -\mathbf{C} & 0 \end{bmatrix}, \\ \hat{\mathbf{B}} &= \begin{bmatrix} \mathbf{B} \\ 0 \end{bmatrix}. \end{aligned} \tag{4.3.1}$$

La ganancia de retroalimentación aumentada que se obtiene resulta en $\hat{\mathbf{K}} = [\mathbf{K} \quad -K_\xi]$ con $\mathbf{K} \in \mathbb{R}^{m \times n}$ y $K_\xi \in \mathbb{R}$. El autor menciona que los valores de las matrices de ponderación $\mathbf{Q}$ y $\mathbf{R}$ se encontraron mediante el proceso de prueba y error, quedando establecidas de la siguiente manera

$$\begin{aligned} \mathbf{Q} &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 6 \times 10^5 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \times 10^3 \end{bmatrix}, \\ \mathbf{R} &= 6 \times 10^3 \left(\frac{180}{\pi} \right)^2. \end{aligned} \tag{4.3.2}$$

4.3.1. Ganancias del controlador

Capturando las matrices de la ecuaciónes (4.3.1) y (4.3.2) en el código de *Matlab*®, se emplea la función *lqr* que resuelve la ecuación (2.2.4) y devuelve las siguientes ganancias

Código fuente 4.5: Ganancias del controlador LQR para el modelo 1 del sistema.

```
>> K_lqr = lqr(A_hat, B_hat, Q, R)
   k_f = K_lqr(1:4)
   k_i = -K_lqr(5)

K_lqr =

   -0.0150   -1.5698   -0.0270   -0.2325    0.0071

k_f =

   -0.0150   -1.5698   -0.0270   -0.2325

k_i =

   -0.0071
```

Código fuente 4.6: Ganancias del controlador LQR para el modelo 2 del sistema.

```
>> K_lqr = lqr(A_hat, B_hat, Q, R)
   k_f = K_lqr(1:4)
   k_i = -K_lqr(5)

K_lqr =

   -0.0153   -2.0748   -0.0273   -0.3228    0.0071

k_f =

   -0.0153   -2.0748   -0.0273   -0.3228

k_i =

   -0.0071
```

Los primeros cuatro valores $\mathbf{K_{lqr}}(1:4)$ corresponden a la ganancia de retroalimentación $\mathbf{k}_f$ mientras el último valor $\mathbf{K_{lqr}}(5)$ corresponde a la ganancia integral k_i.

Con el fin de comprobar que las ganancias del controlador cumplen con el objetivo de estabilización del sistema NXT ballbot, A partir de la ecuación (2.1.7) se crea un nuevo modelo en espacio de estados del sistema en lazo cerrado como se muestra a continuación

Código fuente 4.7: Creación del modelo en espacio de estados del sistema NXT ballbot en lazo cerrado.

```
estados_cl = {'\theta' '\psi' 'd\theta/dt' 'd\psi/dt' 'error'};
entrada_cl = {'r'};
salidas_cl = {'\theta' '\psi' 'd\theta/dt' 'd\psi/dt' };
sys_clt = ss([A-B*K_lqr(1:4) -B*K_lqr(5); -C(1,:) 0],[zeros(size(B));1],...
             [C zeros(size(B))],D,'statename',estados_cl,...
```

```
'inputname',entrada_cl,'outputname',salidas_cl);
```

Los polos del sistema con la retroalimentación de estados obtenidos son

Código fuente 4.8: Polos del sistema en lazo cerrado (modelo 1) ante el controlador LQR.

```
>> polos_lazocerrado = pole(sys_clt1)

polos_lazocerrado =

   1.0e+02 *

  -2.4185
  -0.0057 + 0.0057i
  -0.0057 - 0.0057i
  -0.0681
  -0.0509
```

Código fuente 4.9: Polos del sistema en lazo cerrado (modelo 2) ante el controlador LQR.

```
>> polos_lazocerrado = pole(sys_clt2)

polos_lazocerrado =

 -14.2200
  -6.8967
  -5.0017
  -0.5736 + 0.5762i
  -0.5736 - 0.5762i
```

Con lo que se observa como todos los polos se localizan en el semiplano izquierdo del plano complejo, haciendo al sistema NXT ballbot estable alrededor del origen del sistema.

4.3.2. Análisis temporal

Con el fin de analizar el desempeño del controlador, se evalúa la respuesta en el tiempo del sistema NXT ballbot ante una entrada escalón unitario, para esto se emplea la función *step* en el código de *Matlab*®. La respuesta del sistema ante la entrada escalón unitario se muestra en las Figuras 4.4 y 4.5.

Los tiempos de establecimiento del estado $x_2 = \psi$ que representa el ángulo de inclinación del cuerpo son $t_s = 6,76$s para el modelo 1, y $t_s = 6,74$s para el modelo 2, los demás valores se resumen en las Tablas (4.1, 4.2). El código fuente completo del controlador LQR se encuentra en el Apéndice C.

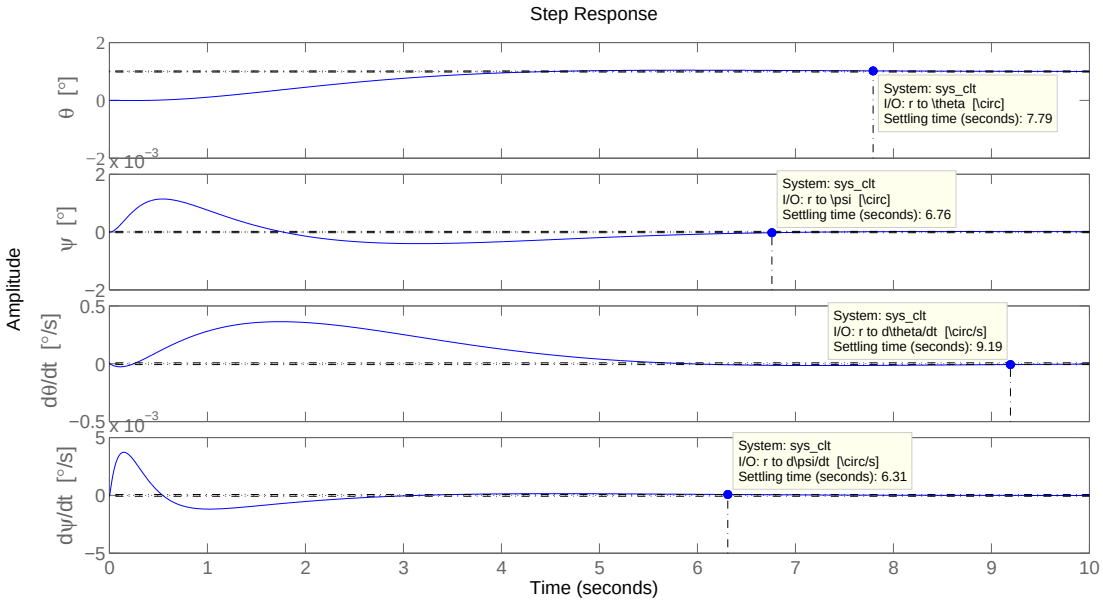


Figura 4.4: Respuesta del sistema ante una entrada escalón unitario para el modelo 1 ante el controlador LQR.

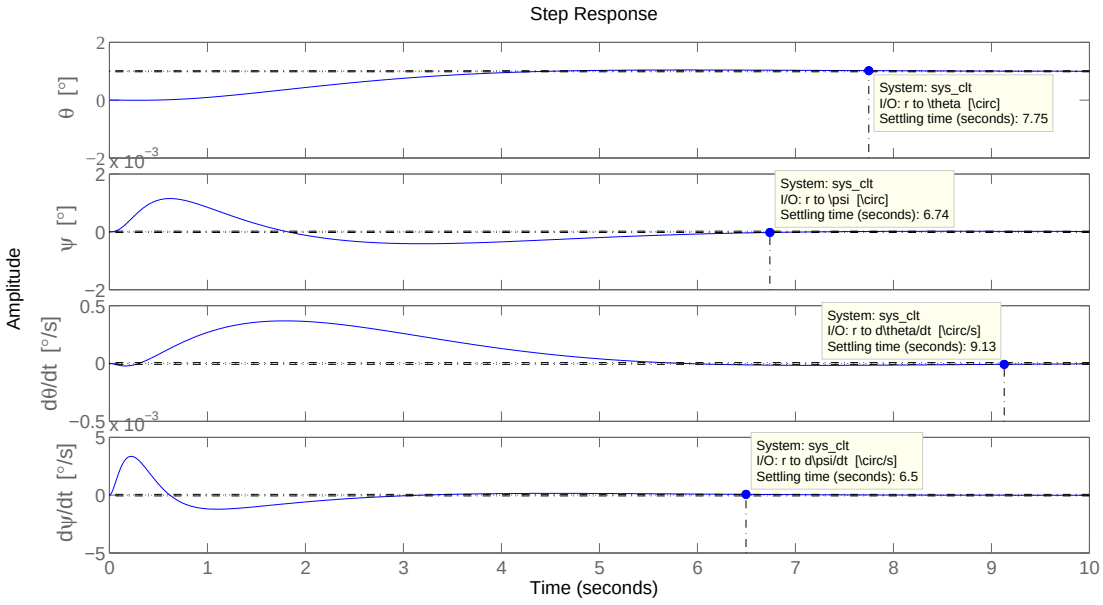


Figura 4.5: Respuesta del sistema ante una entrada escalón unitario para el modelo 2 ante el controlador LQR.

Tabla 4.1: Valores de la respuesta temporal del modelo 1 del sistema ante el controlador LQR

Característica	**Estado θ**	**Estado ψ**	**Estado $\dot{\theta}$**	**Estado $\dot{\psi}$**
Tiempo de levantamiento t_r [s]	2,69	$1{,}82 \times 10^{-12}$	0	0
Tiempo pico t_p [s]	5,9	544×10^{-3}	1,74	148×10^{-3}
Pico de amplitud M_p [°]	1,04	$1{,}14 \times 10^{-3}$	363×10^{-3}	$3{,}73 \times 10^{-3}$
Tiempo de establecimiento t_s [s]	7,79	6,76	9,19	6,31

Tabla 4.2: Valores de la respuesta temporal del modelo 2 del sistema ante el controlador LQR

Característica	**Estado θ**	**Estado ψ**	**Estado $\dot{\theta}$**	**Estado $\dot{\psi}$**
Tiempo de levantamiento t_r [s]	2,65	$5{,}55 \times 10^{-15}$	0	0
Tiempo pico t_p [s]	5,88	616×10^{-3}	1,79	224×10^{-3}
Pico de amplitud M_p [°]	1,04	$1{,}15 \times 10^{-3}$	368×10^{-3}	$3{,}35 \times 10^{-3}$
Tiempo de establecimiento t_s [s]	7,75	6,74	9,13	6,5

4.4. Control mediante asignación de polos

El autor Pablo Tomás [32] propone un controlador mediante la técnica de asignación de polos con el fin de obtener una respuesta mas rápida y un menor soprepaso con respecto al controlador obtenido anteriormente bajo el esquema LQR. Luego de realizar pruebas y correcciones, el autor establece la siguiente ubicación de polos.

$$\begin{aligned} p_1 &= -53 + 2i \\ p_2 &= -53 - 2i \\ p_3 &= -1 \\ p_4 &= -2 \end{aligned}$$

En este caso el autor no añade el término del estado de la integral del error de posición.

4.4.1. Ganancias del controlador

Haciendo uso de la función *place* de *Matlab*®, se obtienen las siguientes ganancias de retroalimentación del controlador.

Código fuente 4.10: Ganancias del controlador mediante asignación de polos para el modelo 1.

```
>> polos_deseados = [-53+2i, -53-2i, -1, -2];
   K_polos = place(A,B,polos_deseados)

K_polos =
```

```
   -0.0075   -1.4776   -0.0226   -0.1324
```

Código fuente 4.11: Ganancias del controlador mediante asignación de polos para el modelo 2.

```
>>    K_polos = place(A,B,polos_deseados)

K_polos =

    -0.1236  -23.7402   -0.2009   -2.3652
```

Como en este caso no se agregó el estado de la integral del error de posición, se procede a crear un nuevo modelo en espacio de estados que representa la dinámica del sistema NXT ballbot en lazo cerrado como se muestra a continuación.

Código fuente 4.12: Creación del modelo en espacio de estados del sistema NXT ballbot en lazo cerrado sin término integral.

```
estados_cl = {'\theta' '\psi' 'd\theta/dt' 'd\psi/dt' };
entrada_cl = {'u'};
salidas_cl = {'\theta' '\psi' 'd\theta/dt' 'd\psi/dt' };
sys_cl = ss(A-B*K_polos,B,C,D,'statename',estados_cl,...
            'inputname',entrada_cl,'outputname',salidas_cl);
```

Para comprobar que efectivamente las ganancias del controlador cumplen con el objetivo de estabilizar al sistema NXT ballbot, se verifica que los polos del modelo en espacio de estados creado coincidan con los polos deseados en ambos modelos.

Código fuente 4.13: Polos del sistema en lazo cerrado de los modelos 1 y 2 ante el controlador por asignación de polos.

```
>> polos_lazocerrado = pole(sys_cl)

polos_lazocerrado =

 -53.0000 + 2.0000i
 -53.0000 - 2.0000i
  -2.0000
  -1.0000
```

Como se observa, los polos de ambos modelos en lazo cerrado coinciden con la ubicación de los polos deseados, garantizando así la estabilidad del sistema NXT ballbot bajo este esquema de control.

4.4.2. Análisis temporal

Nuevamente para analizar el desempeño del controlador y hacer futuras comparaciones con los demás controladores, se evalúa la respuesta en el tiempo del sistema NXT ballbot

ante una entrada escalón unitario. La respuesta de los modelos 1 y 2 ante una entrada escalón unitario se muestra en las Figuras 4.6 y 4.7.

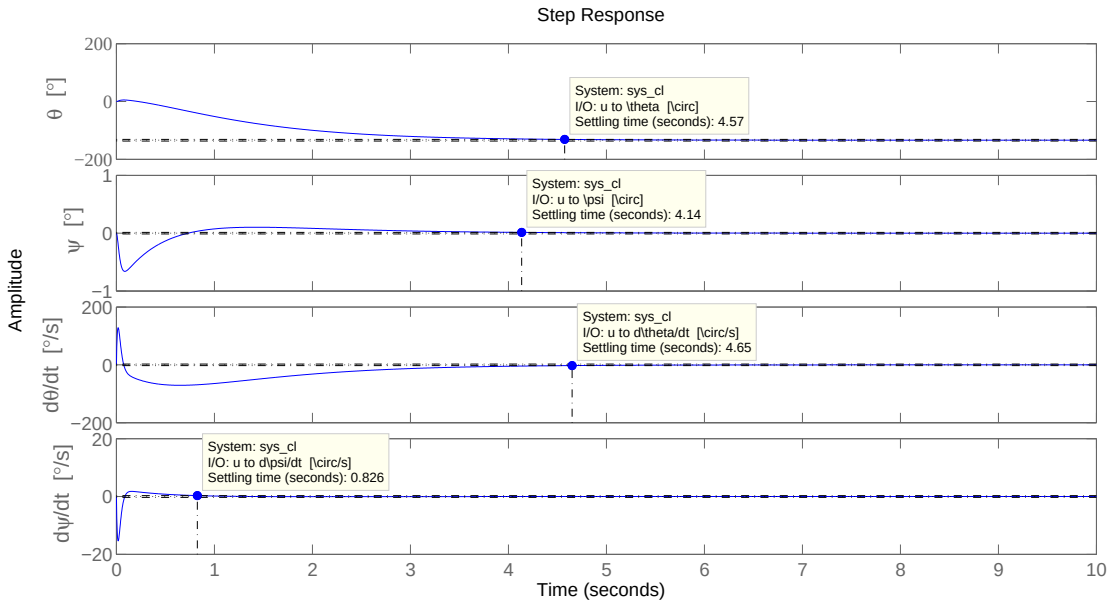


Figura 4.6: Respuesta ante una entrada escalón unitario del modelo 1 del sistema NXT ballbot en lazo cerrado.

El tiempo de establecimiento del estado $x_2 = \psi$ es $t_s = 4{,}14$s para ambos modelos, menores que los obtenidos mediante el controlador LQR. Los demás valores se resumen en la Tablas (4.3, 4.4). El código fuente completo del controlador por asignación de polos se encuentra en el Apéndice C.

Tabla 4.3: Valores de la respuesta temporal del modelo 1 del sistema ante el controlador por asignación de polos.

Característica	**Estado θ**	**Estado ψ**	**Estado $\dot{\theta}$**	**Estado $\dot{\psi}$**
Tiempo de levantamiento t_r [s]	2,54	$2{,}26 \times 10^{-14}$	0	0
Tiempo pico t_p [s]	10	88×10^{-3}	16×10^{-3}	16×10^{-3}
Pico de amplitud M_p [°]	−134	−0,66	128	−15,3
Tiempo de establecimiento t_s [s]	4,57	4,14	4,65	0,826

4.5. Control LQR con LMIs

En esta sección se diseñará un controlador LQR para el modelo politópico del sistema NXT ballbot mediante la formulación de desigualdades matriciales lineales (LMIs), con fines

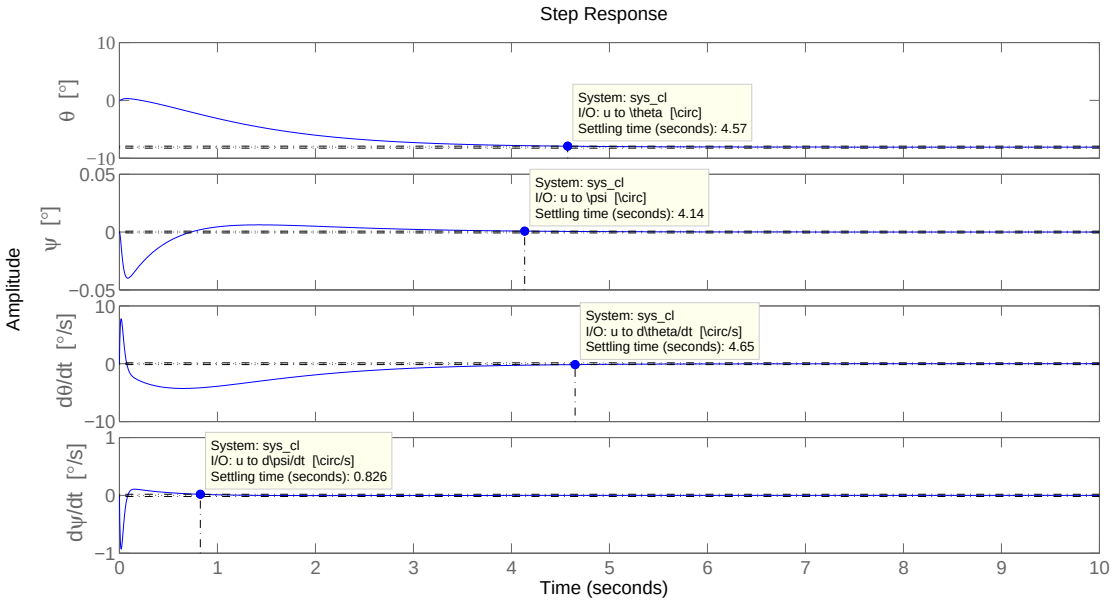


Figura 4.7: Respuesta ante una entrada escalón unitario del modelo 2 del sistema NXT ballbot en lazo cerrado.

Tabla 4.4: Valores de la respuesta temporal del modelo 2 del sistema ante el controlador por asignación de polos.

Característica	**Estado θ**	**Estado ψ**	**Estado $\dot{\theta}$**	**Estado $\dot{\psi}$**
Tiempo de levantamiento t_r [s]	2,54	$2{,}42 \times 10^{-15}$	0	0
Tiempo pico t_p [s]	10	88×10^{-3}	16×10^{-3}	16×10^{-3}
Pico de amplitud M_p [°]	−8,09	$39{,}9 \times 10^{-3}$	7,75	-926×10^{-3}
Tiempo de establecimiento t_s [s]	4,57	4,14	4,65	0,826

de comparación con el controlador LQR presentado anteriormente, se emplearán los mismos valores de las matrices de ponderación. Dado el sistema politópico de la ecuación (2.3.23) y añadiendo nuevamente el término de la integral de error de posición, se reescribe la ecuación de estado como (4.3.1), esta vez con las matrices aumentadas correspondientes a

$$\begin{aligned} \hat{\mathbf{A}} &= \begin{bmatrix} \mathbf{A}_i & \mathbf{0} \\ -\mathbf{C} & 0 \end{bmatrix}, \\ \hat{\mathbf{B}} &= \begin{bmatrix} \mathbf{B}_i \\ 0 \end{bmatrix}, \quad i = 1, \ldots, L. \end{aligned} \tag{4.5.3}$$

Para la solución de este nuevo esquema, solo basta con sustituir $\mathbf{A}_i$ y $\mathbf{B}_i$ por el conjunto de ecuaciones (3.6.4) correspondientes a los vértices del sistema politópico en las formulaciones de optimización (2.3.29). Posteriormente, se procede a capturar este conjunto de ecuaciones

a través del toolbox *LMI Control Toolbox*® de *Matlab*® [19, 61], con lo que se obtendrán las matrices óptimas que satisfacen dichas restricciones.

4.5.1. Ganancias del controlador

Una vez capturadas las formulaciones de optimización correspondiente a cada vértice del sistema politópico en el código de *Matlab*®, se hallan los valores de las matrices óptimas correspondientes al conjunto de restricciones (2.3.29), (2.3.18), el segundo conjunto de restricciones se empleó luego de observar que las ganancias del controlador únicamente con el primer conjunto de restricciones generaba algunas entradas de control que superaban el límite del valor de operación del sistema, con lo que finalmente se obtuvieron las siguientes ganancias

Código fuente 4.14: Ganancias del controlador LQR mediante LMIs para el modelo politópico 1.

```
>> K_lqrlmi = Yopt/(Popt)
   k_f = K_lqrlmi(1:4)
   k_i = -K_lqrlmi(5)

K_lqrlmi =

   -0.0072   -1.3851   -0.0181   -0.2298    0.0025

k_f =

   -0.0072   -1.3851   -0.0181   -0.2298

k_i =

   -0.0025
```

Código fuente 4.15: Ganancias del controlador LQR mediante LMIs para el modelo politópico 2.

```
>> K_lqrlmi = Yopt/(Popt)
   k_f = K_lqrlmi(1:4)
   k_i = -K_lqrlmi(5)

K_lqrlmi =

   -0.0074   -0.9940   -0.0167   -0.1682    0.0023

k_f =

   -0.0074   -0.9940   -0.0167   -0.1682

k_i =

   -0.0023
```

Con el fin de comprobar que las ganancias del controlador cumplen con el objetivo de estabilización del sistema politópico NXT ballbot, se crean cuatro modelos en espacio de estados correspondientes a cada vértice del sistema politópico en lazo cerrado como se muestra a continuación

Creación del modelo en espacio de estados para los vértices del sistema politópico del NXT ballbot en lazo cerrado.

```
estados_cl = {'\theta' '\psi' 'd\theta/dt' 'd\psi/dt' 'error'};
entrada_cl = {'r'};
salidas_cl = {'\theta' '\psi' 'd\theta/dt' 'd\psi/dt' };
syscl1tr = ss([A1-B1*K_lqrlmi(1:4) -B1*K_lqrlmi(5); -Cy 0],...
              [zeros(size(B1)); 1],[C zeros(size(B1))],D,'statename',...
               estados_cl,'inputname',entrada_cl,'outputname',salidas_cl);
syscl2tr = ss([A2-B1*K_lqrlmi(1:4) -B1*K_lqrlmi(5); -Cy 0],...
              [zeros(size(B1)); 1],[C zeros(size(B1))],D,'statename',...
               estados_cl,'inputname',entrada_cl,'outputname',salidas_cl);
syscl3tr = ss([A3-B3*K_lqrlmi(1:4) -B3*K_lqrlmi(5); -Cy 0],...
              [zeros(size(B3)); 1],[C zeros(size(B3))],D,'statename',...
               estados_cl,'inputname',entrada_cl,'outputname',salidas_cl);
syscl4tr = ss([A4-B3*K_lqrlmi(1:4) -B3*K_lqrlmi(5); -Cy 0],...
              [zeros(size(B3)); 1],[C zeros(size(B3))],D,'statename',...
               estados_cl,'inputname',entrada_cl,'outputname',salidas_cl);
```

Los polos del sistema politópico con retroalimentación de estados correspondientes a cada vértice obtenidos son

Código fuente 4.16: Polos del modelo politópico 1 en lazo cerrado ante el controlador LQR mediante LMIs.

```
>> polos_lazocerrado1 = pole(syscl1tr)
   polos_lazocerrado2 = pole(syscl2tr)
   polos_lazocerrado3 = pole(syscl3tr)
   polos_lazocerrado4 = pole(syscl4tr)

polos_lazocerrado1 =

   1.0e+02 *

  -4.4854
  -0.0483
  -0.0087 + 0.0068i
  -0.0087 - 0.0068i
  -0.0084

polos_lazocerrado2 =

   1.0e+02 *

  -4.7330
  -0.0501
```

```
  -0.0072 + 0.0105i
  -0.0072 - 0.0105i
  -0.0057

polos_lazocerrado3 =

  -7.2270
  -1.1641 + 3.8589i
  -1.1641 - 3.8589i
  -0.5579 + 0.3534i
  -0.5579 - 0.3534i

polos_lazocerrado4 =

  -7.4855
  -1.1722 + 3.2877i
  -1.1722 - 3.2877i
  -0.7058 + 0.2516i
  -0.7058 - 0.2516i
```

Código fuente 4.17: Polos del modelo politópico 2 en lazo cerrado ante el controlador LQR mediante LMIs.

```
>> polos_lazocerrado1 = pole(syscl1tr)
   polos_lazocerrado2 = pole(syscl2tr)
   polos_lazocerrado3 = pole(syscl3tr)
   polos_lazocerrado4 = pole(syscl4tr)

polos_lazocerrado1 =

   1.0e+02 *

  -7.4940
  -0.0462
  -0.0043
  -0.0105 + 0.0136i
  -0.0105 - 0.0136i

polos_lazocerrado2 =

   1.0e+02 *

  -7.7401
  -0.0475
  -0.0041
  -0.0088 + 0.0146i
  -0.0088 - 0.0146i

polos_lazocerrado3 =

 -11.6212
  -1.9973 + 2.8166i
```

```
  -1.9973 - 2.8166i
  -1.6442
  -0.4462

polos_lazocerrado4 =

 -12.3198
  -1.6268 + 2.4351i
  -1.6268 - 2.4351i
  -2.2818
  -0.4217
```

Con lo que se observa como todos los polos de los cuatro vértices correspondientes a cada modelo politópico se localizan en el semiplano izquierdo del plano complejo, haciendo al sistema politópico NXT ballbot estable alrededor del origen del sistema.

4.5.2. Análisis temporal

Con el fin de analizar el desempeño del controlador, se evalúa la respuesta en el tiempo del sistema politópico NXT ballbot ante una entrada escalón unitario, las Figuras 4.8 y 4.9 muestran las respuestas de los vértices del sistema politópico.

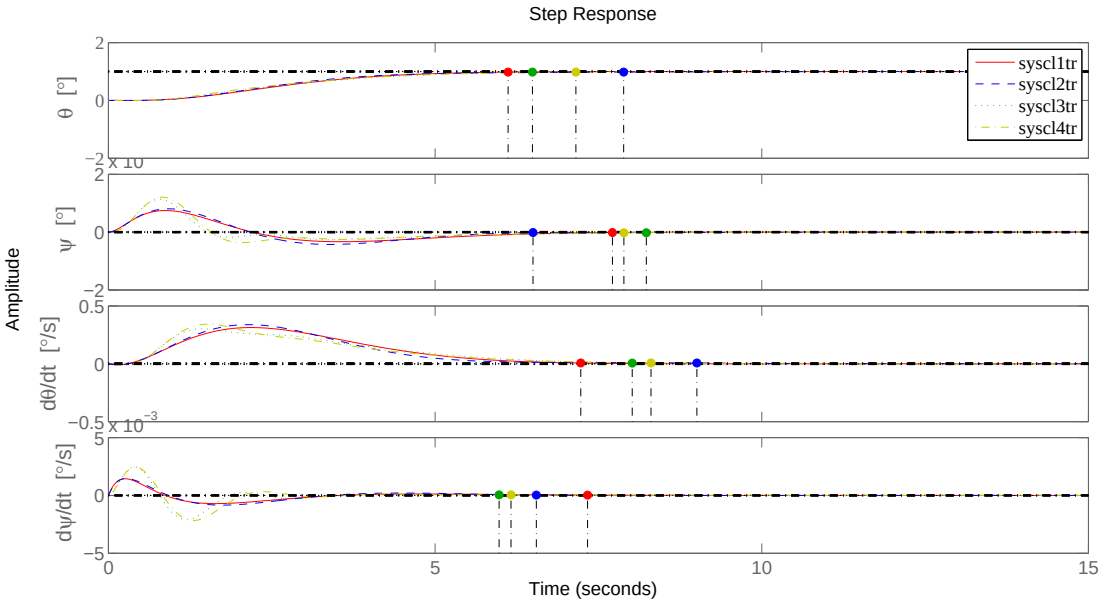


Figura 4.8: Respuesta de los vértices del modelo politópico 1 ante una entrada escalón unitario en lazo cerrado.

Los valores de la respuesta temporal de los vértices del sistema se resumen en las Tablas (4.5, 4.6). El código fuente completo del controlador LQR mediante LMIs se encuentra en el Apéndice C.

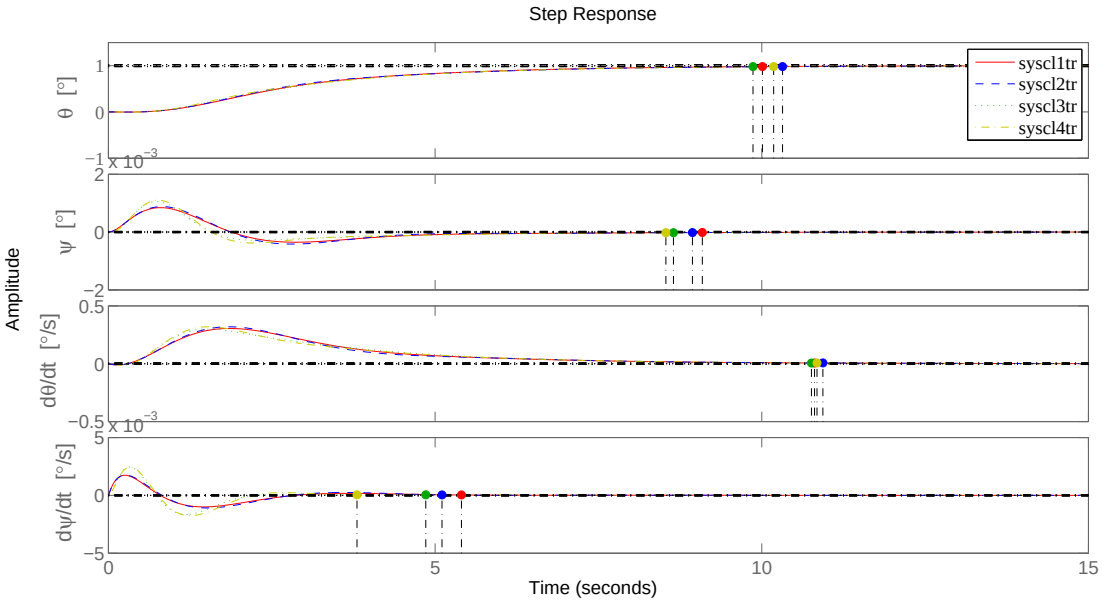


Figura 4.9: Respuesta de los vértices del modelo politópico 2 ante una entrada escalón unitario en lazo cerrado.

4.6. Control mediante asignación de polos y LMIs

El problema de asignación de polos para el modelo politópico del sistema NXT ballbot consiste en ubicar los polos del sistema dentro de una región denominada Región LMI $S(\alpha, r, \theta)$ [16, 58, 72], la cual para el caso del modelo politópico (2.3.23) del sistema NXT ballbot dependerá de las restricciones en tiempo de respuesta y esfuerzo de control especificadas en la Tabla 4.7.

Los valores de estos parámetros establecen prácticamente que el conjunto de restricciones (2.3.20),(2.3.21),(2.3.22) y (2.3.18), establezcan a los polos del sistema politópico en lazo cerrado en el semiplano izquierdo del plano complejo, limitado entre 0 y -2000s^{-1}, esto debido a que con valores menores se encontraba que la solución de las restricciones era infactible o los esfuerzos de control eran muy cercanos a el valor límite del sistema, en particular con los vértices 1 y 2 del modelo politópico 2. Dado que se incluyó dentro del problema de control el objetivo de seguimiento de una trayectoria, nuevamente se agregó el término de la integral del error del estado de posición, empleando así la ecuación de estados (4.5.3).

4.6.1. Ganancias del controlador

Una vez capturadas en *Matlab*® el conjunto de restricciones y resolviendo el problema de factibilidad, se obtuvieron las siguientes ganancias

Tabla 4.5: Valores de la respuesta temporal del modelo politópico 1 del sistema ante el controlador LQR mediante LMIs.

Característica	**Estado θ**	**Estado ψ**	**Estado $\dot{\theta}$**	**Estado $\dot{\psi}$**
Tiempo de levantamiento t_r [s]				
Vértice 1	3,4	$3{,}24 \times 10^{-14}$	0	0
Vértice 2	3,31	$2{,}49 \times 10^{-14}$	0	$1{,}44 \times 10^{-15}$
Vértice 3	3,87	$0{,}111 \times 10^{-14}$	0	$0{,}333 \times 10^{-15}$
Vértice 4	3,93	$0{,}0888 \times 10^{-14}$	0	0
Tiempo pico t_p [s]				
Vértice 1	15	876×10^{-3}	2,19	248×10^{-3}
Vértice 2	15	919×10^{-3}	2,15	276×10^{-3}
Vértice 3	9,09	777×10^{-3}	1,44	395×10^{-3}
Vértice 4	15	848×10^{-3}	1,5	391×10^{-3}
Pico de amplitud M_p [°]				
Vértice 1	1	$0{,}745 \times 10^{-3}$	313×10^{-3}	$1{,}45 \times 10^{-3}$
Vértice 2	1	$0{,}805 \times 10^{-3}$	339×10^{-3}	$1{,}46 \times 10^{-3}$
Vértice 3	1,01	$1{,}14 \times 10^{-3}$	302×10^{-3}	$2{,}47 \times 10^{-3}$
Vértice 4	1	$1{,}21 \times 10^{-3}$	343×10^{-3}	$2{,}46 \times 10^{-3}$
Tiempo de establecimiento t_s [s]				
vértice 1	6,12	7,72	7,23	7,33
vértice 2	7,84	6,5	9,01	6,55
vértice 3	6,49	8,23	8,02	5,98
vértice 4	7,16	7,84	8,31	6,16

Código fuente 4.18: Ganancias del controlador mediante asignación de polos y LMIs para el modelo polítopico 1.

```
>> K_poloslmi = Y/W
   k_f = K_poloslmi(1:4)
   k_i = -K_poloslmi(5)

K_poloslmi =

   -0.0072   -1.6683   -0.0202   -0.2577    0.0025

k_f =

   -0.0072   -1.6683   -0.0202   -0.2577

k_i =

    -0.0025
```

Tabla 4.6: Valores de la respuesta temporal del modelo politópico 2 del sistema ante el controlador LQR mediante LMIs.

Característica	**Estado θ**	**Estado ψ**	**Estado $\dot{\theta}$**	**Estado $\dot{\psi}$**
Tiempo de levantamiento t_r [s]				
Vértice 1	5,07	$2{,}12 \times 10^{-13}$	0	0
Vértice 2	5,22	$1{,}81 \times 10^{-13}$	0	0
Vértice 3	5,13	$4{,}44 \times 10^{-15}$	0	0
Vértice 4	5,25	$3{,}55 \times 10^{-15}$	0	0
Tiempo pico t_p [s]				
Vértice 1	15	798×10^{-3}	1,85	259×10^{-3}
Vértice 2	15	814×10^{-3}	1,84	271×10^{-3}
Vértice 3	15	738×10^{-3}	1,52	323×10^{-3}
Vértice 4	15	767×10^{-3}	1,53	323×10^{-3}
Pico de amplitud M_p [°]				
Vértice 1	1	$0{,}843 \times 10^{-3}$	305×10^{-3}	$1{,}74 \times 10^{-3}$
Vértice 2	1	$0{,}88 \times 10^{-3}$	319×10^{-3}	$1{,}75 \times 10^{-3}$
Vértice 3	1,01	$1{,}06 \times 10^{-3}$	300×10^{-3}	$2{,}46 \times 10^{-3}$
Vértice 4	1	$1{,}1 \times 10^{-3}$	320×10^{-3}	$2{,}45 \times 10^{-3}$
Tiempo de establecimiento t_s [s]				
Vértice 1	10	9,09	10,8	5,4
Vértice 2	10,3	8,94	10,9	5,11
Vértice 3	9,87	8,65	10,8	4,86
Vértice 4	10,2	8,53	10,8	3,81

Tabla 4.7: Parámetros para el controlador mediante asignación de polos y LMIs.

Parámetro	**Valor**
α	0
θ	85°
r	2000 s^{-1}
μ	7 V

Código fuente 4.19: Ganancias del controlador mediante asignación de polos y LMIs para el modelo politópico 2.

```
>> K_poloslmi = Y/W
   k_f = K_poloslmi(1:4)
   k_i = -K_poloslmi(5)

K_poloslmi =

   -0.0056   -1.2246   -0.0187   -0.1874    0.0019
```

```
k_f =

   -0.0056   -1.2246   -0.0187   -0.1874

k_i =

   -0.0019
```

Comprobando que las ganancias del controlador cumplan con el objetivo de estabilización del modelo politópico NXT ballbot, se crean nuevamente cuatro modelos en espacio de estados correspondientes a cada vértice del sistema politópico en lazo cerrado como se muestra a continuación

Código fuente 4.20: Creación del modelo en espacio de estados para los vértices del sistema politópico del NXT ballbot en lazo cerrado.

```
estados_cl = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
              'd\psi/dt  [\circ/s]' 'error'};
entrada_cl = {'r'};
salidas_cl = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
              'd\psi/dt  [\circ/s]'  };
syscl1tr = ss([A1-B1*K_poloslmi(1:4) -B1*K_poloslmi(5); -Cy ...
    0],[zeros(size(B1)); 1],...
              [C zeros(size(B1))],D,'statename',estados_cl,...
              'inputname',entrada_cl,'outputname',salidas_cl);
syscl2tr = ss([A2-B1*K_poloslmi(1:4) -B1*K_poloslmi(5); -Cy ...
    0],[zeros(size(B1)); 1],...
              [C zeros(size(B1))],D,'statename',estados_cl,...
             'inputname',entrada_cl,'outputname',salidas_cl);
syscl3tr = ss([A3-B3*K_poloslmi(1:4) -B3*K_poloslmi(5); -Cy ...
    0],[zeros(size(B3)); 1],...
              [C zeros(size(B3))],D,'statename',estados_cl,...
             'inputname',entrada_cl,'outputname',salidas_cl);
syscl4tr = ss([A4-B3*K_poloslmi(1:4) -B3*K_poloslmi(5); -Cy ...
    0],[zeros(size(B3)); 1],...
              [C zeros(size(B3))],D,'statename',estados_cl,...
             'inputname',entrada_cl,'outputname',salidas_cl);
```

Con lo que se obtuvieron los siguientes polos en lazo cerrado del sistema politópico

Código fuente 4.21: Polos del modelo politópico 1 en lazo cerrado ante el controlador por ubicación de polos en regiones LMI.

```
>> polos_lazocerrado1 = pole(syscl1tr)
   polos_lazocerrado2 = pole(syscl2tr)
   polos_lazocerrado3 = pole(syscl3tr)
   polos_lazocerrado4 = pole(syscl4tr)

polos_lazocerrado1 =

   1.0e+02 *
```

```
  -4.6730
  -0.0582
  -0.0019 + 0.0022i
  -0.0019 - 0.0022i
  -0.0181

polos_lazocerrado2 =

   1.0e+02 *

  -4.9207
  -0.0583
  -0.0024 + 0.0023i
  -0.0024 - 0.0023i
  -0.0131

polos_lazocerrado3 =

  -6.0011
  -2.3760 + 3.9863i
  -2.3760 - 3.9863i
  -0.1857 + 0.2069i
  -0.1857 - 0.2069i

polos_lazocerrado4 =

  -6.0375
  -2.6058 + 3.2800i
  -2.6058 - 3.2800i
  -0.2230 + 0.2111i
  -0.2230 - 0.2111i
```

Código fuente 4.22: Polos del modelo politópico 2 en lazo cerrado ante el controlador por ubicación de polos en regiones LMI.

```
>> polos_lazocerrado1 = pole(syscl1tr)
   polos_lazocerrado2 = pole(syscl2tr)
   polos_lazocerrado3 = pole(syscl3tr)
   polos_lazocerrado4 = pole(syscl4tr)

polos_lazocerrado1 =

   1.0e+02 *

  -7.6286
  -0.0583
  -0.0043 + 0.0043i
  -0.0043 - 0.0043i
  -0.0222

polos_lazocerrado2 =
```

```
   1.0e+02 *

  -7.8751
  -0.0584
  -0.0048 + 0.0044i
  -0.0048 - 0.0044i
  -0.0183

polos_lazocerrado3 =

  -5.5389 + 3.5191i
  -5.5389 - 3.5191i
  -6.1743
  -0.4053 + 0.3937i
  -0.4053 - 0.3937i

polos_lazocerrado4 =

  -6.5948
  -5.5738 + 2.2402i
  -5.5738 - 2.2402i
  -0.4454 + 0.3979i
  -0.4454 - 0.3979i
```

Como se observa, todos los polos se encuentran en el semiplano izquierdo del plano complejo, siendo el polo mas alejado con valor de -787.51 y el mas cercano con -0.19, definiendo así la región LMI que estabiliza al sistema politópico NXT ballbot.

4.6.2. Análisis temporal

Analizando el desempeño del controlador, se evalúa la respuesta en el tiempo del sistema politópico NXT ballbot ante una entrada escalón unitario, las Figuras 4.10 y 4.11 muestran las respuestas de los vértices del sistema politópico.

Los valores de la respuesta temporal de los vértices del sistema se resumen en las Tablas (4.8,4.9). El código fuente completo del controlador por ubicación de polos en regiones LMI se encuentra en el Apéndice C.

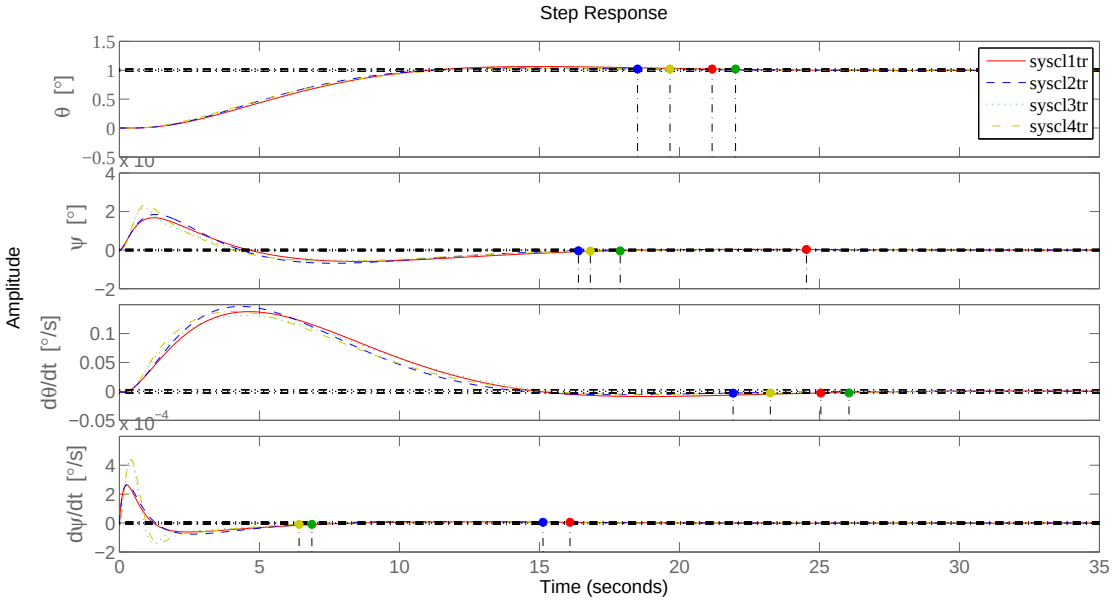


Figura 4.10: Respuesta de los vértices del modelo politópico 1 ante una entrada escalón unitario en lazo cerrado.

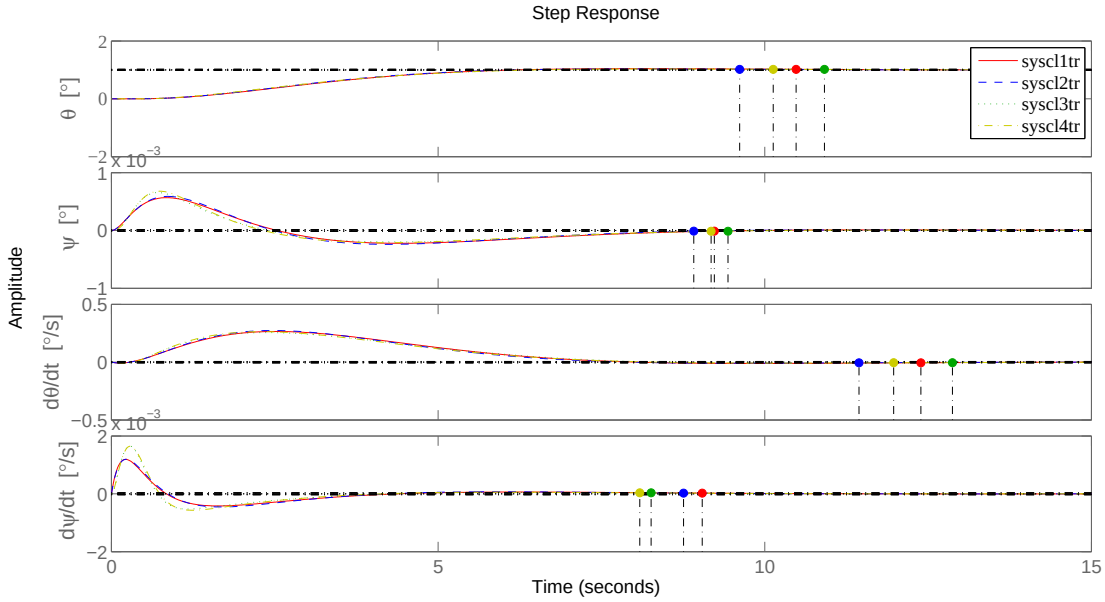


Figura 4.11: Respuesta de los vértices del modelo politópico 2 ante una entrada escalón unitario en lazo cerrado.

Tabla 4.8: Valores de la respuesta temporal del modelo politópico 1 del sistema ante el controlador por ubicación de polos en regiones LMI.

Característica	**Estado θ**	**Estado ψ**	**Estado $\dot{\theta}$**	**Estado $\dot{\psi}$**
Tiempo de levantamiento t_r [s]				
Vértice 1	6,91	$1{,}33 \times 10^{-12}$	0	0
Vértice 2	6,7	$0{,}983 \times 10^{-12}$	0	0
Vértice 3	7,29	$0{,}0{,}14 \times 10^{-12}$	0	0
Vértice 4	7,17	$0{,}0115 \times 10^{-12}$	0	0
Tiempo pico t_p [s]				
Vértice 1	14,9	1,22	4,62	269×10^{-3}
Vértice 2	14,5	1,31	4,33	284×10^{-3}
Vértice 3	15,6	875×10^{-3}	4,44	399×10^{-3}
Vértice 4	15,3	961×10^{-3}	4,04	412×10^{-3}
Pico de amplitud M_p [°]				
Vértice 1	1,06	$0{,}168 \times 10^{-3}$	138×10^{-3}	$0{,}263 \times 10^{-3}$
Vértice 2	1,04	$0{,}185 \times 10^{-3}$	147×10^{-3}	$0{,}266 \times 10^{-3}$
Vértice 3	1,06	$0{,}218 \times 10^{-3}$	131×10^{-3}	$0{,}438 \times 10^{-3}$
Vértice 4	1,04	$0{,}237 \times 10^{-3}$	138×10^{-3}	$0{,}439 \times 10^{-3}$
Tiempo de establecimiento t_s [s]				
Vértice 1	21,2	24,5	25,1	16,1
Vértice 2	18,5	16,4	21,9	15,1
Vértice 3	622	17,9	26,1	6,87
Vértice 4	19,7	10,8	23,2	6,42

Tabla 4.9: Valores de la respuesta temporal del modelo politópico 2 del sistema ante el controlador por ubicación de polos en regiones LMI.

Característica	**Estado θ**	**Estado ψ**	**Estado $\dot{\theta}$**	**Estado $\dot{\psi}$**
Tiempo de levantamiento t_r [s]				
Vértice 1	3,68	$4{,}58 \times 10^{-13}$	0	0
Vértice 2	3,62	$4{,}02 \times 10^{-13}$	0	0
Vértice 3	3,86	$6{,}33 \times 10^{-14}$	0	$1{,}11 \times 10^{-15}$
Vértice 4	3,82	$7{,}21 \times 10^{-14}$	0	$1{,}22 \times 10^{-15}$
Tiempo pico t_p [s]				
Vértice 1	8,05	837×10^{-3}	2,51	221×10^{-3}
Vértice 2	7,93	867×10^{-3}	2,46	221×10^{-3}
Vértice 3	8,4	716×10^{-3}	2,37	283×10^{-3}
Vértice 4	8,37	740×10^{-3}	2,3	293×10^{-3}
Pico de amplitud M_p [°]				
Vértice 1	1,04	567×10^{-3}	265×10^{-3}	$1{,}19 \times 10^{-3}$
Vértice 2	1,03	589×10^{-3}	273×10^{-3}	$1{,}19 \times 10^{-3}$
Vértice 3	1,04	656×10^{-3}	256×10^{-3}	$1{,}65 \times 10^{-3}$
Vértice 4	1,03	679×10^{-3}	263×10^{-3}	$1{,}64 \times 10^{-3}$
Tiempo de establecimiento t_s [s]				
Vértice 1	10,5	9,23	12,4	9,04
Vértice 2	9,62	8,91	11,4	8,76
Vértice 3	10,9	9,44	12,9	8,26
Vértice 4	10,1	9,18	12	8,09

Capítulo 5

Resultados en simulación

En este capítulo se presentan los resultados obtenidos a partir de simulaciones numéricas del sistema en lazo cerrado, empleando los distintos controladores diseñados en el capítulo anterior. Para esto, se ejecutó el modelo del controlador implementado en *Simulink*® en el modo *Normal*, como se describe en el Apéndice E. Las pruebas de cada controlador se realizaron para los modelos 1 y 2, tanto en su modelo nominal como en su representación politópica. Dado que los sensores giroscópicos tienen una desviación media de 2°, se estableció una condición inicial en el estado de posición angular $\psi = 2°$ y un tiempo de simulación de 30 segundos, así como un tiempo de muestreo de $1ms$.

Las primeras simulaciones que se realizaron fueron aplicadas al modelo del sistema con los parámetros del motor reportados por Yamamoto [78], denominado modelo 1 y posteriormente, se realizaron las simulaciones correspondientes al modelo con los parámetros obtenidos mediante pruebas experimentales, al que se denominó modelo 2.

5.1. Controlador LQR

El primer controlador a simular fue el reportado por Yamamoto [78], el cual ya ha mostrado buenos resultados tanto en simulación como experimentalmente, y servirá de referencia para comparar el desempeño de los demás controladores. Las respuestas del sistema en lazo cerrado servirán para analizar el desempeño del controlador LQR en términos de robustez ante incertidumbre en los parámetros, así como la demanda en el esfuerzo de control.

5.1.1. Respuesta del modelo 1

Para analizar la respuesta del controlador en el modelo 1, se realizaron 5 simulaciones. Primeramente se realizó la simulación del controlador aplicado al modelo nominal, el cual contiene los valores reportados por Yamamoto [78]. Para esto se emplearon las matrices de estado $\mathbf{A}$ y $\mathbf{B}$ dadas en la ecuación (3.5.2). Posteriormente, se realizarón 4 simulaciones en el supuesto caso de que el modelo del sistema se encontrara en cada uno de los vértices de su representación politópica. Para esto se emplearon las matrices de estado $\mathbf{A_i}$ y $\mathbf{B_i}$ para

$i = 1, 2, 3, 4$ dadas en la ecuación (3.6.4) dependiendo del vértice del modelo politópico.

Por último, se realizó una gráfica comparativa en la respuesta de los estados $x_1 = \theta$ y $x_2 = \psi$ ante estos 5 casos, como se muestra en la Figura 5.9. Se han tomado estos dos estados debido a que son los de principal interés, ya que nos proporcionan la información del desplazamiento y posición angular del sistema NXT ballbot.

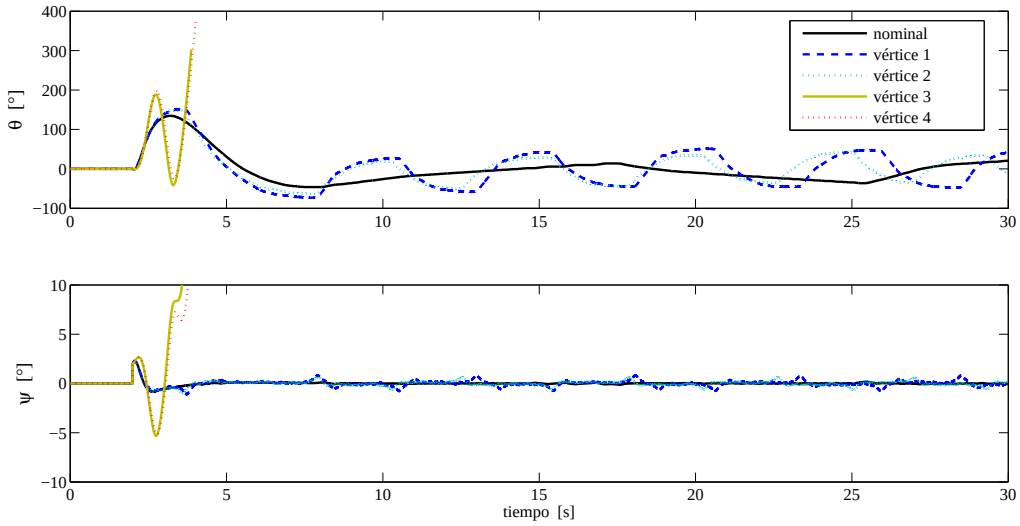


Figura 5.1: Comparativa de la respuesta del sistema en lazo cerrado ante el controlador LQR, aplicado al modelo 1 y su representación politópica.

Como resultado se observa que el controlador LQR cumple con su objetivo de estabilizar al sistema NXT ballbot solo en el modelo nominal y los vértices 1 y 2 del modelo politópico. En cuanto a la respuesta de los estados, el estado θ presenta ligeras oscilaciones en el modelo nominal, no así en en los vértices 1 y 2 donde se puede apreciar más a detalle este comportamiento sub amortiguado, por otra parte, en el estado ψ se observa que la respuesta para los casos en los que el sistema es estable, se encuentra acotada en una región de $\pm 1°$ por lo que se puede concluir que aunque la respuesta es satisfactoria en estos casos, el controlador no cumple con el objetivo de robustez ante incertidumbre en los parámetros.

Por último, una vez evaluada la robustez del controlador, evaluamos el esfuerzo de control requerido para estabilizar al sistema NXT ballbot. Para esto, se resumen en la Figura 5.10, los ciclos de trabajo del PWM requerido en cada caso. En esta gráfica se observa cómo el esfuerzo de control requerido para estabilizar al sistema NXT ballbot en los vértices 1 y 2 es mayor que el requerido en el sistema nominal, mientras que en los vértices 3 y 4 alcanza un valor máximo el cual no es suficiente para lograr el objetivo de estabilización.

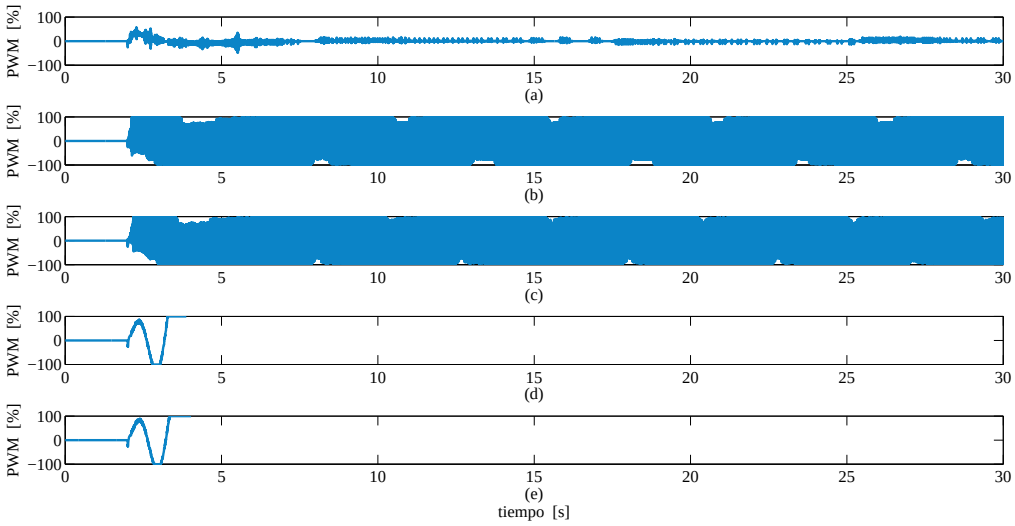


Figura 5.2: Comparativa de las señales que muestran el ciclo de trabajo del PWM que demanda el sistema en lazo cerrado ante el controlador LQR en el modelo 1 y sus respectivos vértices del modelo politópico; (a) nominal, (b) vértice 1, (c) vértice 2, (d) vértice 3, (e) vértice 4.

5.1.2. Respuesta del modelo 2

En el caso del modelo 2, el cual contiene los parámetros físicos del motor NXT obtenidos experimentalmente. Se realizaron las mismas 5 simulaciones tanto en el modelo nominal, como en sus respectivos vértices del modelo politópico. En este caso las matrices $\mathbf{A}$ y $\mathbf{B}$ de la ecuación (3.5.2) difieren por los valores de los parámetros del motor NXT, así como las matrices de estado $\mathbf{A_i}$ y $\mathbf{B_i}$ para $i = 1, 2, 3, 4$ descritas en (3.6.4), y las ganancias del controlador, como se mostró en el capítulo anterior. Análogamente, se realizó una gráfica comparativa para analizar la robustez del controlador, como se muestra en la Figura 5.3.

Como resultado de la simulación, se obtiene esta vez que el controlador cumple con su objetivo de estabilizar al sistema NXT ballbot en el modelo nominal y en los vértices 3 y 4, no así en los vértices 1 y 2 como lo hacía el controlador LQR para el modelo 1. En cuanto a los estados $x_1 = \theta$ y $x_2 = \psi$, en los casos en que el controlador logra estabilizar al sistema, estas respuestas son muy semejantes, sin presentar grandes oscilaciones y en el caso de ψ acotada en una región menor a $\pm 0{,}5°$. Sin embargo, a pesar de una respuesta aceptable del sistema en estos 3 casos, la robustez del controlador ante incertidumbre en los parámetros no se satisface.

Por último, analizamos el esfuerzo de control requerido por el controlador en los casos en que logra estabilizar al sistema NXT ballbot. En la Figura 5.4 , se muestran los diferentes ciclos de trabajo requeridos por parte del sistema ante el controlador LQR para el modelo 2 y sus respectivos vértices del modelo politópico. Como conclusión, se observa que los vértices 3 y 4 mantienen una demanda semejante en el esfuerzo de control que el modelo nominal, mientras que en los vértices 1 y 2 se observa cómo va incrementando la demanda de control

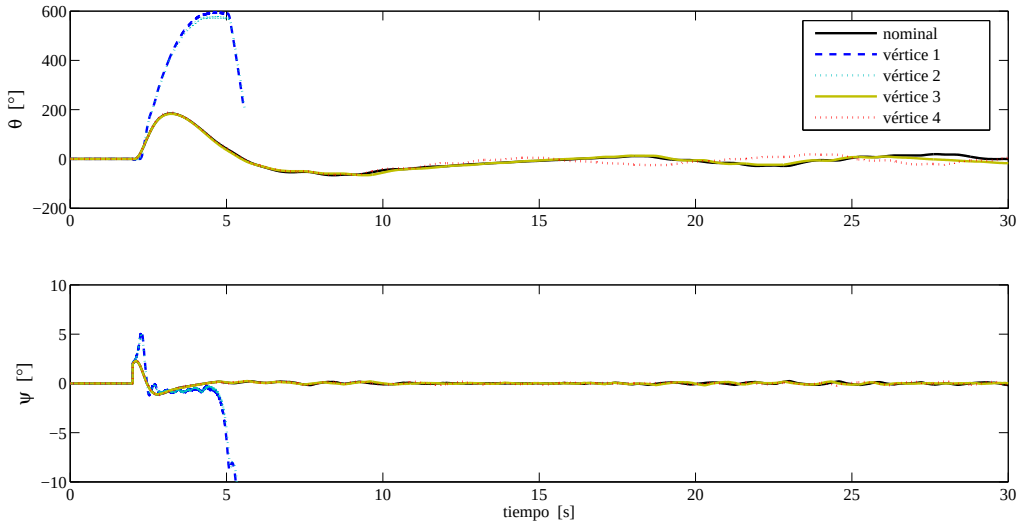


Figura 5.3: Comparativa de la respuesta del sistema en lazo cerrado ante el controlador LQR, aplicado al modelo 2 y su representación politópica.

requerida hasta llegar al punto en que esta no puede ser suministrada.

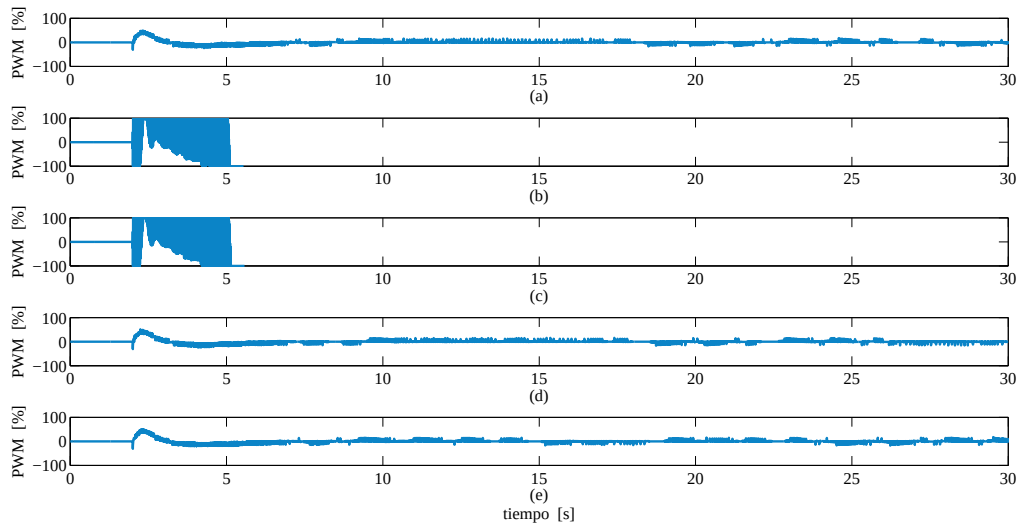


Figura 5.4: Comparativa de las señales que muestran el ciclo de trabajo del PWM del sistema ante el controlador LQR en el modelo 2 y los vértices de su representación politópica, (a) nominal, (b) vértice 1, (c) vértice 2, (d) vértice 3, (e) vértice 4.

5.2. Controlador por asignación de polos

Para la simulación del controlador por asignación de polos, se emplearon los polos establecidos por Pablo Tomás [32]. Cabe recordar que en este caso, el autor no incluye el estado de error de posición ξ, por lo que en el modelo de *Simulink*® la ganancia k_i se establece en 0. Las respuestas del sistema en lazo cerrado servirán para analizar el desempeño del controlador por asignación de polos en términos de robustez ante incertidumbre en los parámetros, así como la demanda en el esfuerzo de control.

5.2.1. Respuesta del modelo 1

Para la simulación en el modelo 1, los resultados de los estados θ y ψ del sistema NXT ballbot se resumen en la Figura 5.5. Como resultado se observa que el controlador por asignación de polos cumple con el objetivo de estabilizar al sistema NXT ballbot únicamente en el modelo nominal y los vértices 1 y 2. En comparación con la respuesta del controlador LQR, esta vez no se presentan oscilaciones. Sin embargo, se puede apreciar como el estado θ en los vértices 1 y 2 no alcanza el valor de referencia 0, esto en parte a la falta de la inclusión del estado ξ para minimizar el error de posición en estado estable. En cuanto al estado ψ en comparación con el controlador LQR, se observa una mejor respuesta, esta vez acotada en una región menor a $\pm 0{,}5°$. Sin embargo, el controlador por asignación de polos tampoco cumple con la característica de ser robusto ante incertidumbre en los parámetros.

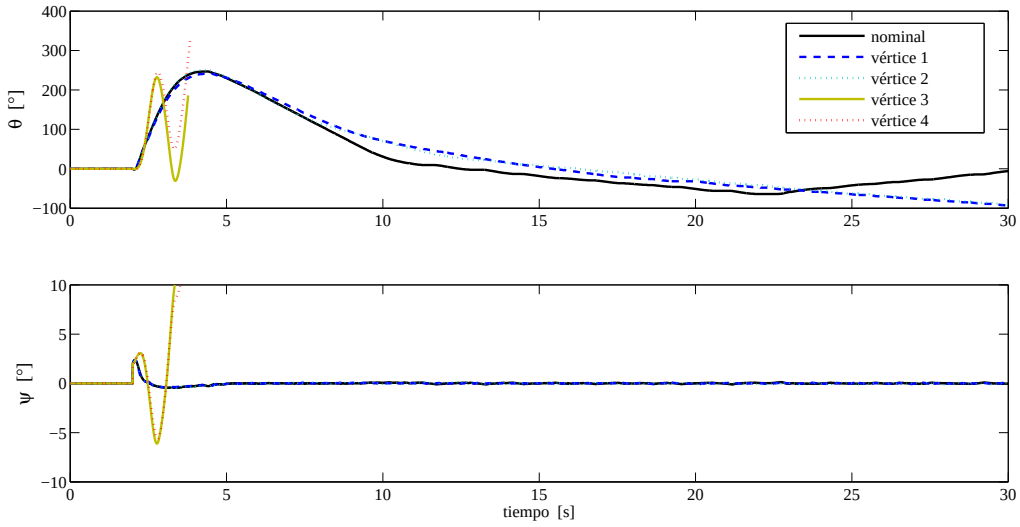


Figura 5.5: Comparativa de los estados del sistema ante el controlador por asignación de polos en el modelo nominal 1 y sus respectivos vértices del modelo politópico.

Por último, se analizan los esfuerzos de control requeridos por el controlador mediante asignación de polos. En la Figura 5.6, se resumen los diferentes ciclos de trabajo para el PWM requeridos. Como conclusión, se observa que el esfuerzo de control requerido en los vértices 1

y 2 es muy semejante al requerido en el modelo nominal. En comparación con el controlador LQR, el controlador mediante asignación de polos para el modelo 1 tiene una demanda menor en el esfuerzo de control.

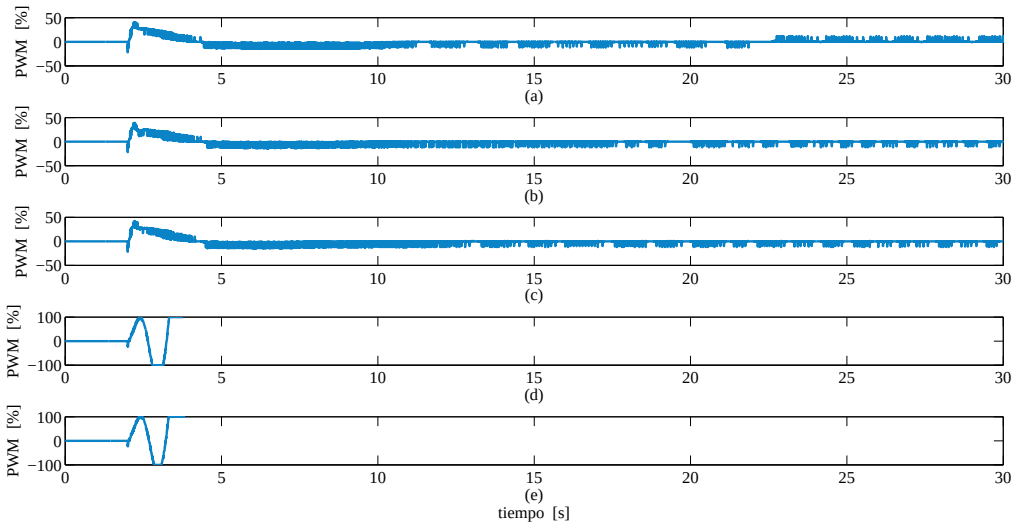


Figura 5.6: Comparativa de las señales que muestran el ciclo de trabajo del PWM del sistema ante el controlador por asignación de polos en el modelo 2 y los vértices de su representación politópica, (a) nominal, (b) vértice 1, (c) vértice 2, (d) vértice 3, (e) vértice 4.

5.2.2. Respuesta del modelo 2

La respuesta en lazo cerrado de los estados θ y ψ para el modelo 2 se resumen en la Figura 5.5. Como resultado se observa que el controlador por asignación de polos cumple con el objetivo de estabilizar por completo al sistema NXT ballbot. Sin embargo, se puede apreciar como el estado θ en los vértices 1 y 2 presenta grandes oscilaciones alrededor del valor de referencia 0, mientras la respuesta del modelo nominal y los vértices 3 y 4 no alcanza por completo el valor de referencia 0, de igual manera por la falta de inclusión del estado ξ. Por otra parte, en comparación con el controlador LQR, los vértices 3 y 4 presentan de igual forma una respuesta muy semejante al modelo nominal. En cuanto al estado ψ en comparación con el controlador LQR en el caso del modelo nominal y los vértices 3 y 4, la respuesta es semejante, nuevamente acotada en una región menor a $\pm 0{,}5°$. Por otra parte, la respuesta en los vértices 1 y 2 presenta oscilaciones alrededordel valor de referencia 0, además de estar acotada en una región de $\pm 5°$. Por lo que se puede concluir que el controlador por asignación de polos para el modelo 2 cumple con la característica de ser robusto ante incertidumbre en los parámetros.

Una vez que el objetivo en la robustez del controlador se ha logrado, se procede a analizar los esfuerzos de control requeridos para la estabilización del sistema NXT ballbot. En la Figura 5.8, se resumen los diferentes ciclos de trabajo para el PWM requeridos. Como conclusión, se observa que el esfuerzo de control requerido en los vértices 3 y 4 es muy semejante al requerido

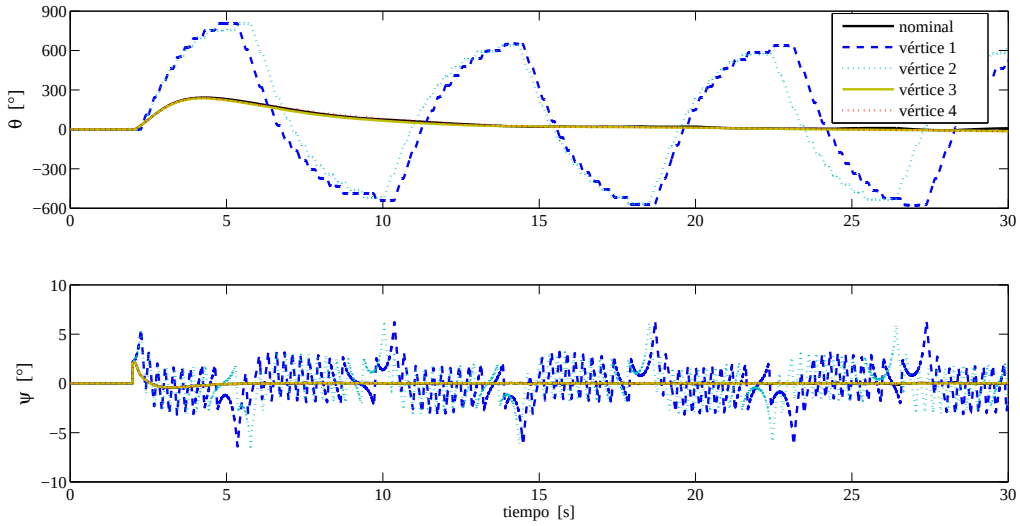


Figura 5.7: Comparativa de los estados del sistema ante el controlador por asignación de polos en el modelo nominal 2 y sus respectivos vértices del modelo politópico.

en el modelo nominal, mientras que el esfuerzo requerido en los vértices 1 y 2 es mucho mayor al nominal, al grado de estar cambiando entre los valores máximo y mínimo durante toda la simulación. En comparación con el controlador LQR y para los casos en los cuales este controlador era capaz de estabilizar al sistema, el controlador mediante asignación de polos para el modelo 2 tiene una demanda mayor en el esfuerzo de control.

5.3. Controlador LQR mediante LMIs

La simulación del controlador LQR mediante LMIs tiene como objetivo comprobar que mediante esta técnica se puede mejorar el desempeño del controlador LQR en términos de robustez ante incertidumbre en los parámetros, así como acotar el esfuerzo de control requerido para estabilizar al sistema en lazo cerrado. Durante la simulación se emplearon los mismos valores que en el controlador LQR, tanto para las matrices de estado (3.5.2), (3.6.4), como las matrices de ponderación (4.3.2). Las cuales en este caso deben satisfacer las formulaciones de optimización (2.3.29) para la robustez del controlador y (2.3.18) para limitar el esfuerzo de control.

5.3.1. Respuesta del modelo 1

La respuesta en lazo cerrado de los estados θ y ψ para el modelo 1 se resumen en la Figura 5.9. Como resultado se observa que el controlador LQR mediante LMIs cumple con el objetivo de estabilizar por completo al sistema NXT ballbot. Aunque la respuesta del estado θ en el modelo nominal presenta ligeras oscilaciones, no ocurre lo mismo para los vértices 1, 2 , 3 y 4, siendo estos dos últimos los que presentan una amplitud menor en las oscilaciones alrededor

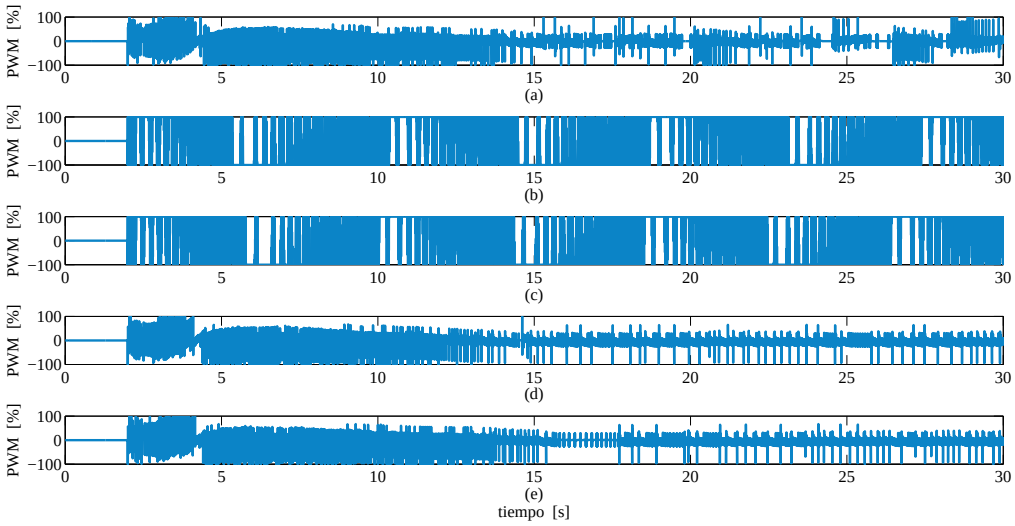


Figura 5.8: Comparativa de las señales que muestran el ciclo de trabajo del PWM del sistema ante el controlador por asignación de polos en el modelo 2 y los vértices de su representación politópica, (a) nominal, (b) vértice 1, (c) vértice 2, (d) vértice 3, (e) vértice 4.

del valor de referencia 0. Por otra parte, en comparación con el controlador LQR, la respuesta del controlador LQR mediante LMIs presenta un valor de sobre impulso de casi el doble. En cuanto al estado ψ en comparación con el controlador LQR, esta vez se encuentra acotada en una región de $\pm 5°$, debido a la respuesta sub amortiguada del sistema en su representación politópica. Con esto se concluye que el controlador LQR mediante LMIs para el modelo 1 cumple con la característica de ser robusto ante incertidumbre en los parámetros.

Por último, en la Figura 5.10 se muestran los diferentes ciclos de trabajo requeridos por el controlador para estabilizar al sistema NXT ballbot. Como conclusión, se observa que los vértices 1 y 2 demandan un esfuerzo de control mayor que en el modelo nominal y los vértices 3 y 4. Por otra parte, comparándolo con el esfuerzo requerido por el controlador LQR, se observa una ligera disminución en la demanda del esfuerzo de control.

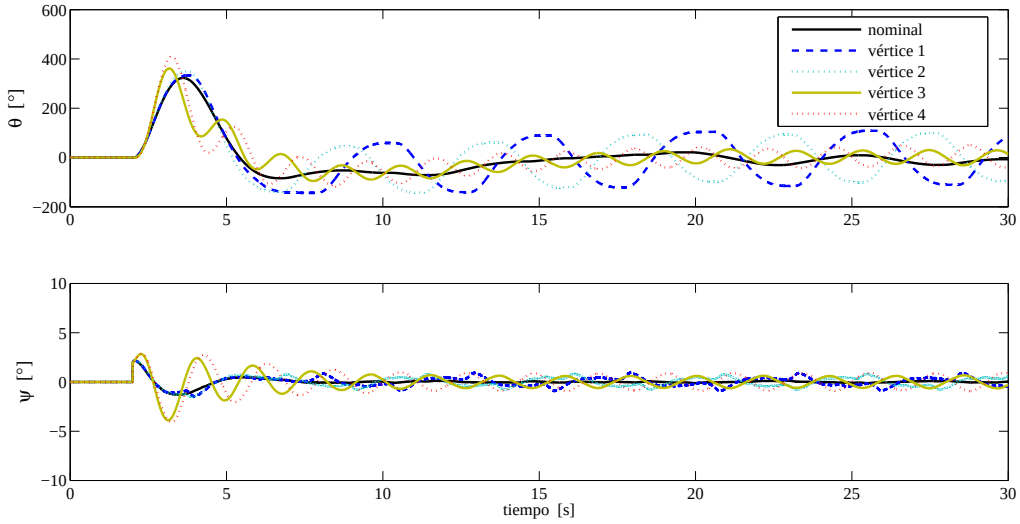


Figura 5.9: Comparativa de los estados del sistema ante el controlador LQR mediante LMIs en el modelo nominal 1 y sus respectivos vértices del modelo politópico.

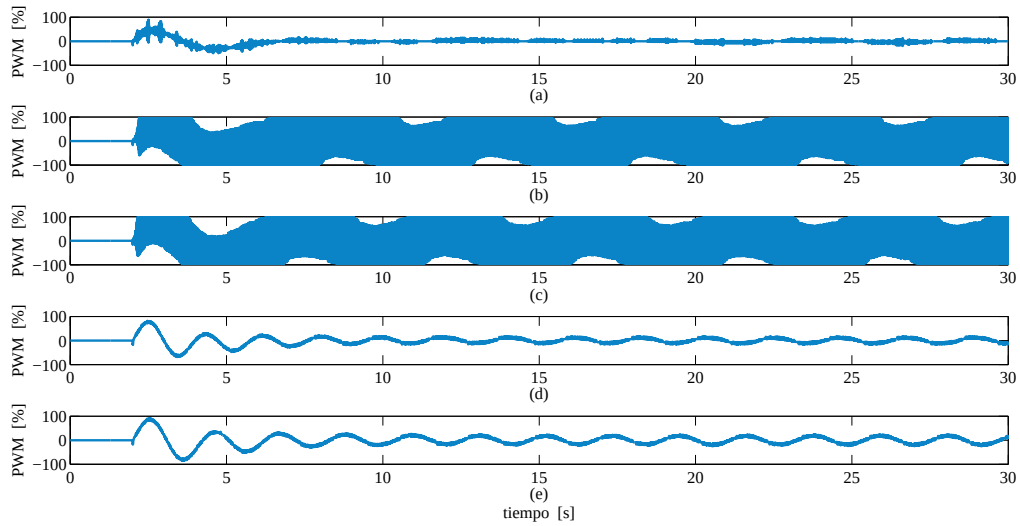


Figura 5.10: Comparativa de las señales que muestran el ciclo de trabajo del PWM del sistema ante el controlador LQR mediante LMIs en el modelo 2 y los vértices de su representación politópica, (a) nominal, (b) vértice 1, (c) vértice 2, (d) vértice 3, (e) vértice 4.

5.3.2. Respuesta del modelo 2

La respuesta en lazo cerrado de los estados θ y ψ para el modelo 2 se resumen en la Figura 5.11. Como resultado se observa que nuevamente el controlador LQR mediante LMIs cumple con el objetivo de estabilizar por completo al sistema NXT ballbot. Esta vez, la respuesta del estado θ tanto en el modelo nominal como en los vértices 3 y 4 presentan ligeras oscilaciones, mientras que en los vértices 1 y 2 las oscilaciones son de mayor amplitud que en el modelo 1. Por otra parte, el sobre impulso en comparación con el controlador LQR es ligeramente mayor. En cuanto a la respuesta del estado ψ en comparación con el controlador LQR, nuevamente se encuentra acotada en una región de $\pm 5°$, debido a la respuesta sub amortiguada del sistema. Con esto se concluye que el controlador LQR mediante LMIs para el modelo 2 cumple con la característica de ser robusto ante incertidumbre en los parámetros.

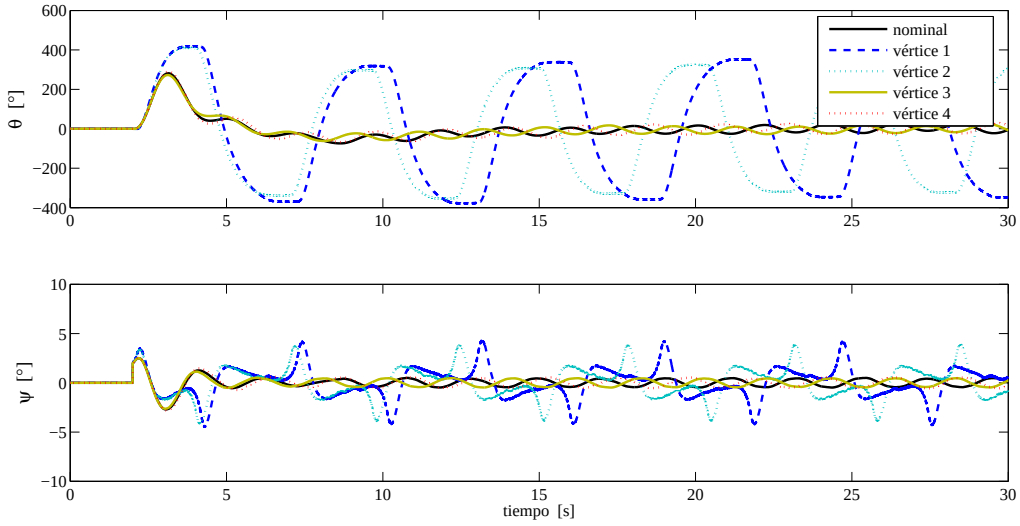


Figura 5.11: Comparativa de los estados del sistema ante el controlador LQR mediante LMIs en el modelo nominal 2 y sus respectivos vértices del modelo politópico.

En la Figura 5.12 se muestran los diferentes ciclos de trabajo requeridos por el controlador para estabilizar al sistema NXT ballbot. En base a estas gráficas, se observa que aunque los vértices 1 y 2 demandan un esfuerzo de control mayor que en el modelo nominal y los vértices 3 y 4, la demanda es mucho menor que el requerido en el modelo 1. Por otra parte, comparándolo con el esfuerzo requerido por el controlador LQR, la demanda en el esfuerzo de control es ligeramente menor.

5.4. Controlador por ubicación de polos en regiones LMI

La simulación del controlador mediante ubicación de polos en regiones LMI tiene como objetivo comprobar que mediante esta técnica se pueden establecer previamente criterios en el

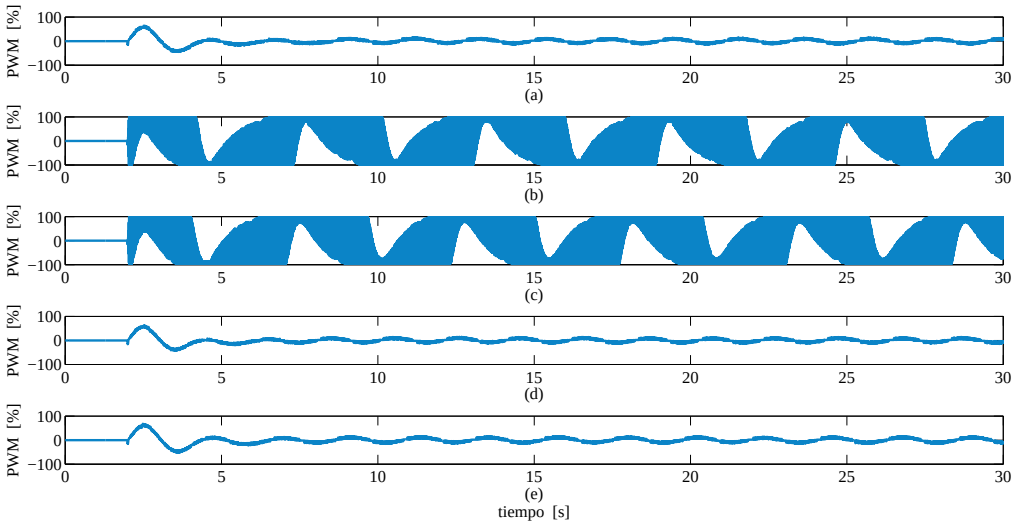


Figura 5.12: Comparativa de las señales que muestran el ciclo de trabajo del PWM del sistema ante el controlador LQR mediante LMIs en el modelo 2 y los vértices de su representación politópica, (a) nominal, (b) vértice 1, (c) vértice 2, (d) vértice 3, (e) vértice 4.

desempeño del controlador como lo son: la respuesta en el tiempo, restricciones en la entrada de control y la robustez ante incertidumbre en los parámetros. En este caso, nuevamente se incluyó el estado ξ con el fin de eliminar el error en estado estable para el estado $x_1 = \theta$.

5.4.1. Respuesta del modelo 1

La respuesta en lazo cerrado de los estados θ y ψ para el modelo 1 se resumen en la Figura 5.13. Como resultado se observa que el controlador mediante ubicación de polos en regiones LMI cumple con el objetivo de estabilizar por completo al sistema NXT ballbot. La respuesta del estado θ en los vértices 1 y 2 presentan oscilaciones de gran tamaño en relación a los controladores antes presentados. Por otra parte, el sobre impulso en comparación con el controlador mediante asignación de polos es aproximadamente el triple. En cuanto a la respuesta del estado ψ, esta se encuentra acotada en una región menor a $\pm 1{,}5°$. Con esto se concluye que el controlador mediante ubicación de polos en regiones LMI para el modelo 1 cumple con la característica de ser robusto ante incertidumbre en los parámetros.

En la Figura 5.14 se muestran los diferentes ciclos de trabajo requeridos por el controlador para estabilizar al sistema NXT ballbot. Comparando los casos en los que el controlador por asignación de polos lograba estabilizar al sistema, la demanda en el esfuerzo de control requerido era menor que el que se requiere con el controlador mediante ubicación de polos en regiones LMI.

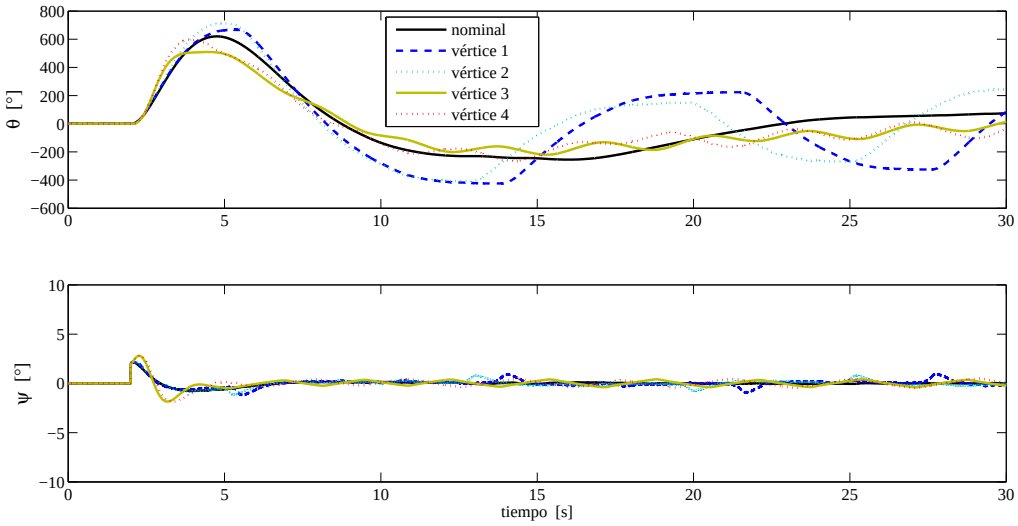


Figura 5.13: Comparativa de los estados del sistema ante el controlador por ubicación de polos en regiones LMI en el modelo nominal 1 y sus respectivos vértices del modelo politópico.

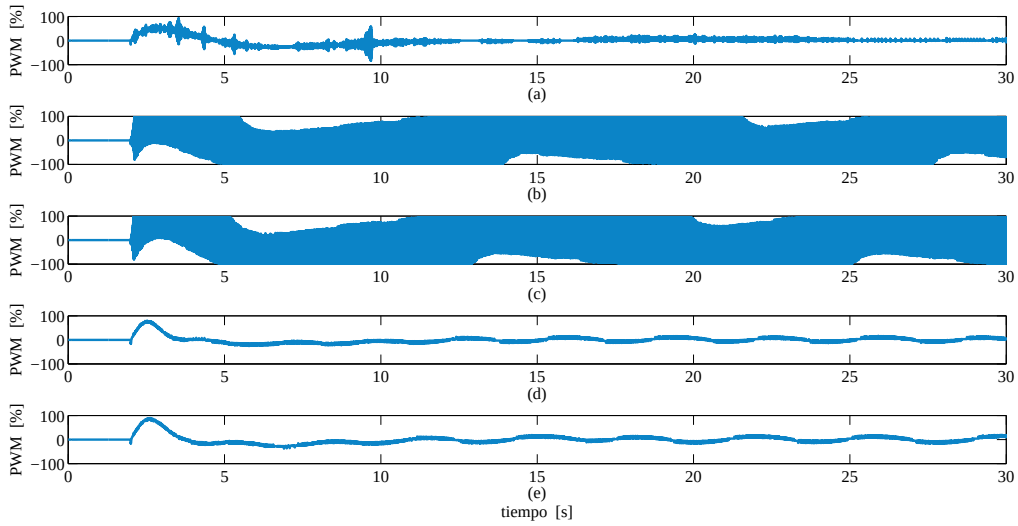


Figura 5.14: Comparativa de las señales que muestran el ciclo de trabajo del PWM del sistema ante el controlador por ubicación de polos en regiones LMI en el modelo 2 y los vértices de su representación politópica, (a) nominal, (b) vértice 1, (c) vértice 2, (d) vértice 3, (e) vértice 4

5.4.2. Respuesta del modelo 2

En la Figura 5.15 se resumen las respuestas en lazo cerrado de los estados θ y ψ para el modelo 2. Como resultado se observa que el controlador mediante ubicación de polos en regiones LMI cumple con el objetivo de estabilizar por completo al sistema NXT ballbot.

Haciendo una comparación con el controlador por asignación de polos para el modelo 2 el cual también logró estabilizar por completo al sistema NXT ballbot. La respuesta del estado θ en los vértices 1 y 2 presentan oscilaciones del mismo tamaño, sin embargo la frecuencia con la que aparecen es menor. En cuanto a la respuesta del estado ψ, esta se encuentra acotada en una región menor a $\pm 5{,}5°$ para los vértices 1 y 2, mientras que para los demás casos la respuesta es satisfactoria. Con esto se concluye que nuevamente el controlador mediante ubicación de polos en regiones LMI ahora para el modelo 2 cumple con la característica de ser robusto ante incertidumbre en los parámetros.

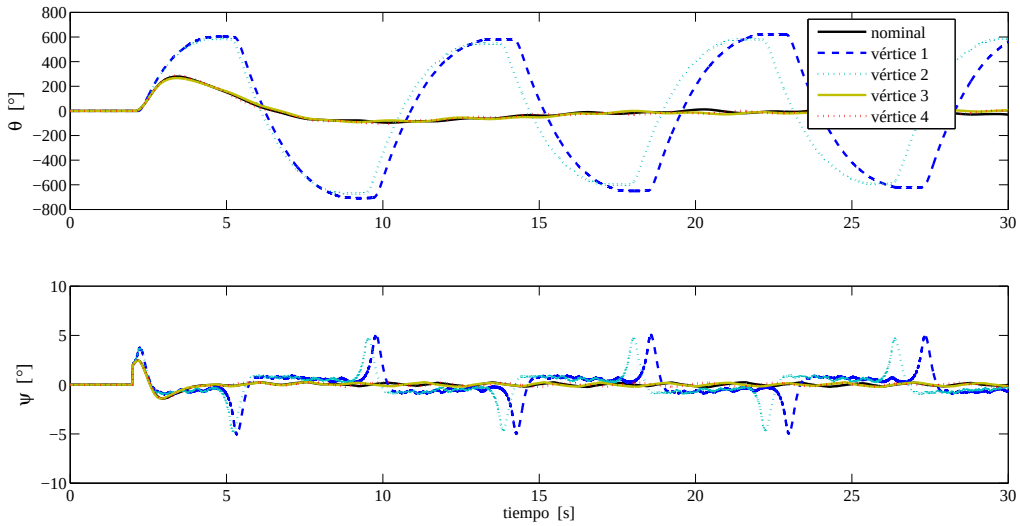


Figura 5.15: Comparativa de los estados del sistema ante el controlador por ubicación de polos en regiones LMI en el modelo nominal y sus respectivos vértices del modelo politópico.

En la Figura 5.16 se muestran los diferentes ciclos de trabajo requeridos por el controlador para estabilizar al sistema NXT ballbot. Comparando los casos en los que el controlador por asignación de polos para el modelo 2 lograba estabilizar completamente al sistema, la demanda en el esfuerzo de control requerido era mayor que el que se requiere con el controlador mediante ubicación de polos en regiones LMI.

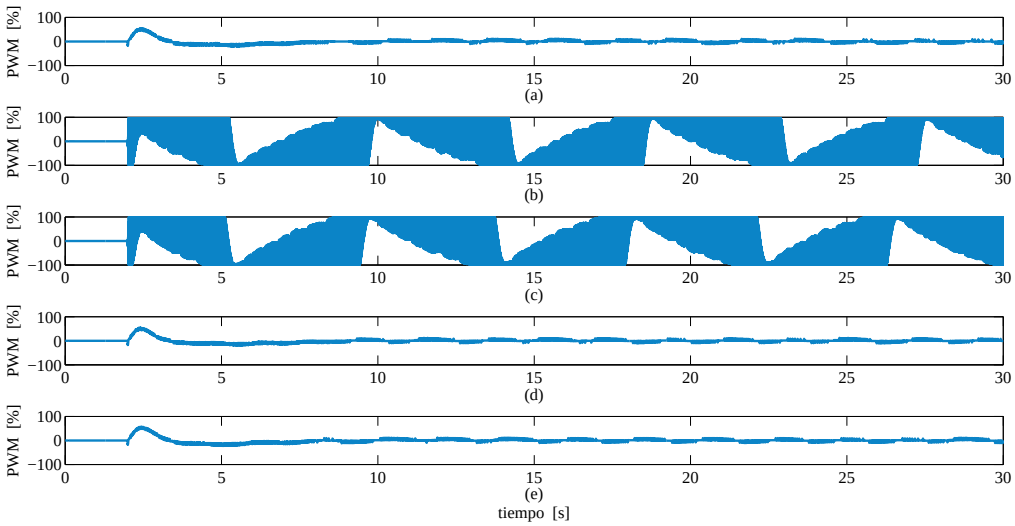


Figura 5.16: Comparativa de las señales que muestran el ciclo de trabajo del PWM del sistema ante el controlador por ubicación de polos en regiones LMI en el modelo 2 y los vértices de su representación politópica, (a) nominal, (b) vértice 1, (c) vértice 2, (d) vértice 3, (e) vértice 4.

Capítulo 6

Resultados experimentales

En esta sección se presentan los resultados obtenidos experimentalmente al ejecutar el modelo en *simulink*® del controlador para el sistema NXT ballbot en el modo *External Mode*, como se muestra en el Apéndice E. Las pruebas de cada controlador se realizaron únicamente para los modelos 1 y 2 en su modelo nominal, puesto que no podemos variar físicamente el valor de los parámetros con incertidumbre elegidos. El tiempo de ejecución se estableció en 30 segundos y un tiempo de muestreo fundamental de $1ms$ con el fin de poder realizar una comparativa con las simulaciones realizadas anteriormente, así como se registraron los resultados tanto para el eje x correspondiente al plano $z-y$ y para el eje z correspondiente al plano $x-y$.

6.1. Controlador LQR

El primer controlador a implementar físicamente fue el LQR. Por lo que a continuación se muestran los resultados obtenidos.

6.1.1. Respuesta del modelo 1

Como se muestra en la Figura 6.1, la respuesta en lazo cerrado del sistema es semejante en ambos planos. El sobreimpulso del estado θ alcanza los 100°, mientras el estado ψ se encuentra acotado en una región menor a ±5°. En este estado se puede apreciar ligeramente como los valores no varían entre ±5° si no que varían en un rango de ±2° pero se van desplazando del origen debido a la suma de la integración del error en la lectura del sensor giroscópico. En comparación con la respuesta en simulación, la trayectoria seguida por ambos estados θ y ψ es muy similar, sin embargo, en esta ocasión se presentan pequeñas oscilaciones alrededor de la trayectoria generada. Así también, la frecuencia con la que se presentan estas oscilaciones es muy grande. Como conclusión, el controlador LQR con los parámetros reportados por Yamamoto [78] en el modelo dinámico de la planta cumple con el objetivo de estabilización.

En cuanto al esfuerzo de control demandado por el sistema, en la Figura 6.2 se muestran los ciclos de trabajo del PWM requeridos en ambos planos. Se puede observar como el ciclo

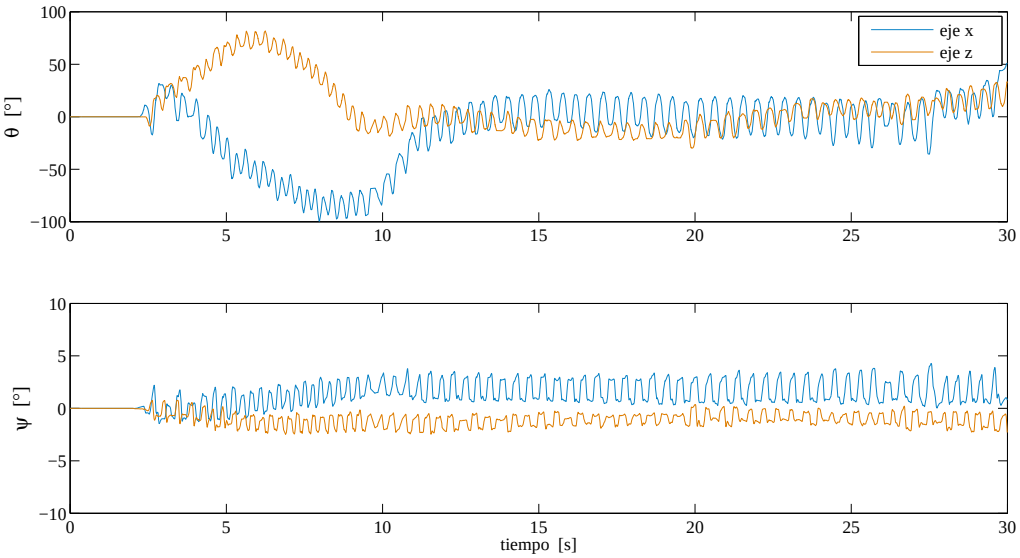


Figura 6.1: Comparativa entre los estados del sistema en los ejes x y z ante el controlador LQR en el modelo 1.

de trabajo del PWM en el plano $z-y$ (Figura 6.2(a)) demanda un mayor esfuerzo de control que en el plano $x-y$ (Figura 6.2(b)). Esta respuesta no se parece mucho al obtenido en la simulación para el modelo nominal, sin embargo, es muy parecida a la obtenida en los vértices 1 y 2 de dicha simulación.

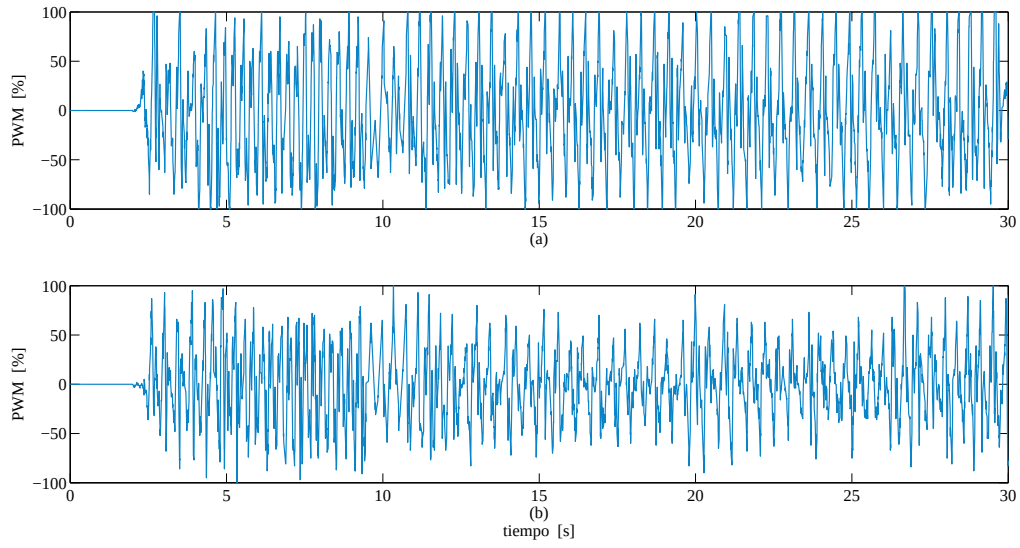


Figura 6.2: Comparativa de las señales que muestran el ciclo de trabajo del PWM del sistema ante el controlador LQR en el modelo 1; (a) eje x, (b) eje z.

6.1.2. Respuesta del modelo 2

En la Figura 6.3 se muestra la respuesta en lazo cerrado obtenida en los estados θ y ψ para ambos planos. El sobre impulso del estado θ en el eje z supera los 100° mientras en el eje x el sobre impulso se encuentra por debajo de este valor, aún así, la respuesta en ambos plano es muy semejante. El estado ψ se encuentra acotado en una región menor a ±4°. Esta vez la desviación del origen es muy pequeña. En comparación con la respuesta en simulación, la trayectoria seguida por ambos estados θ y ψ presenta un comportamiento sub amortiguado además de que siguen apareciendo las pequeñas oscilaciones alrededor de la trayectoria generada. Así también, la frecuencia con la que se presentan estas oscilaciones sigue siendo muy grande. Como conclusión, el controlador LQR con los parámetros obtenidos experimentalmente en el modelo dinámico de la planta cumple con el objetivo de estabilización.

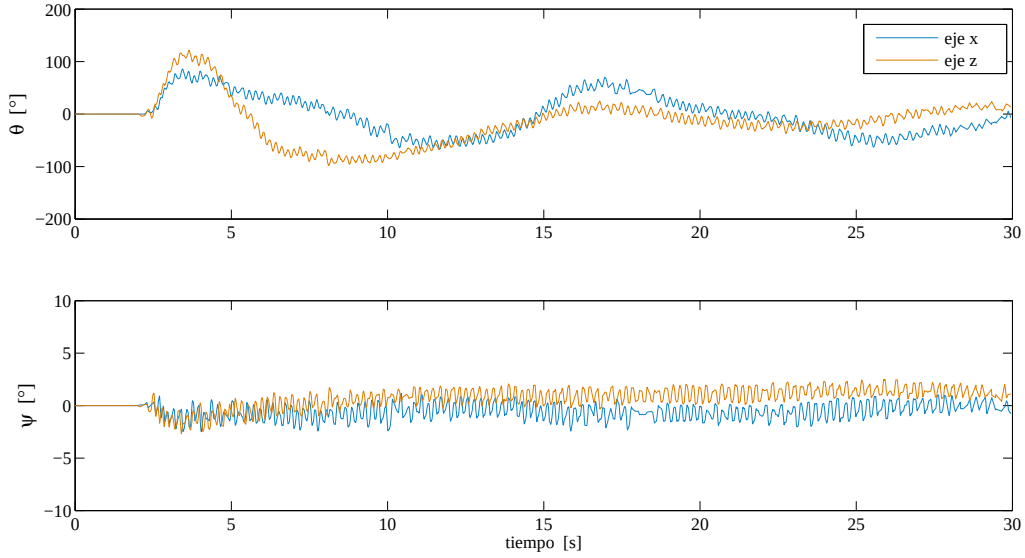


Figura 6.3: Comparativa entre los estados del sistema en los ejes x y z ante el controlador LQR en el modelo 2.

Con respecto al esfuerzo de control requerido por el sistema, en la Figura 6.4 se muestran los ciclos de trabajo del PWM demandados por el sistema NXT ballbot en ambos planos. Esta vez, la demanda del esfuerzo de control en ambos planos es muy semejante, sin llegar a parecerse a alguna respuesta obtenida en la simulación.

6.2. Controlador por asignación de polos

Los resultados de la respuesta en lazo cerrado del sistema NXT ballbot ante el controlador por asignación de polos se presentan a continuación.

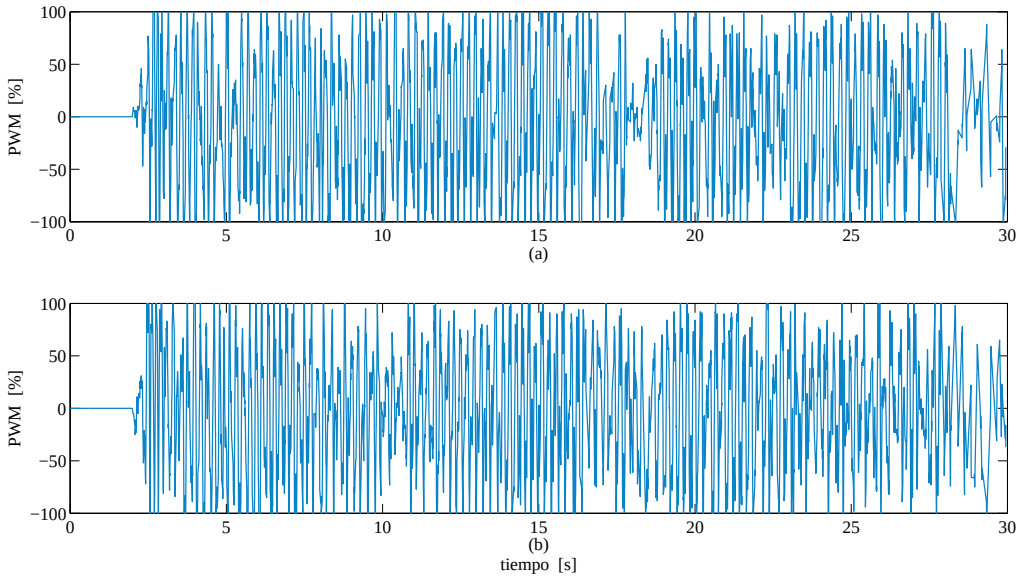


Figura 6.4: Comparativa de las señales que muestran el ciclo de trabajo del PWM del sistema ante el controlador LQR en el modelo 2; (a) eje x, (b) eje z.

6.2.1. Respuesta del modelo 1

En la Figura 6.5 se muestra la respuesta en los estados θ y ψ para ambos planos. El sobre impulso del estado θ en los ejes x y z se encuentran por encima de los 300°. La respuesta en ambos planos es semejante aunque en sentidos contrarios. El estado ψ se encuentra acotado en una región menor a $\pm 4°$. La desviación al origen es muy pequeña, sin embargo, la amplitud de las oscilaciones es mayor que la que se obtuvo con el controlador LQR. En comparación con la respuesta en simulación, la trayectoria seguida por el estado θ es muy parecida, inclusive para el tiempo de simulación no logra alcanzar el valor de referencia $\theta = 0$. Por otra parte, ψ presenta un comportamiento con demasiadas oscilaciones que las que se mostraban en la simulación. Además, las pequeñas oscilaciones alrededor de la trayectoria generada siguen apareciendo, esta vez con una frecuencia menor a las presentadas por el controlador LQR. Como conclusión, el controlador por asignación de polos con los parámetros obtenidos experimentalmente en el modelo dinámico de la planta cumple con el objetivo de estabilización.

Con respecto al esfuerzo de control requerido para estabilizar al sistema NXT ballbot, se registraron los ciclos de trabajo requeridos en el PWM para ambos planos. La Figura 6.6 muestra estos datos, de los cuales se puede observar como el esfuerzo de control requerido en el eje z es menor al que se requiere en el eje x, pero en general, ambos esfuerzos son menores al que requiere el controlador LQR. Comparándo dichos esfuerzos de control con los obtenidos durante la simulación, ninguno de los 3 casos en los que se lograba la estabilización del sistema se parece a los mostrados en la Figura 6.6.

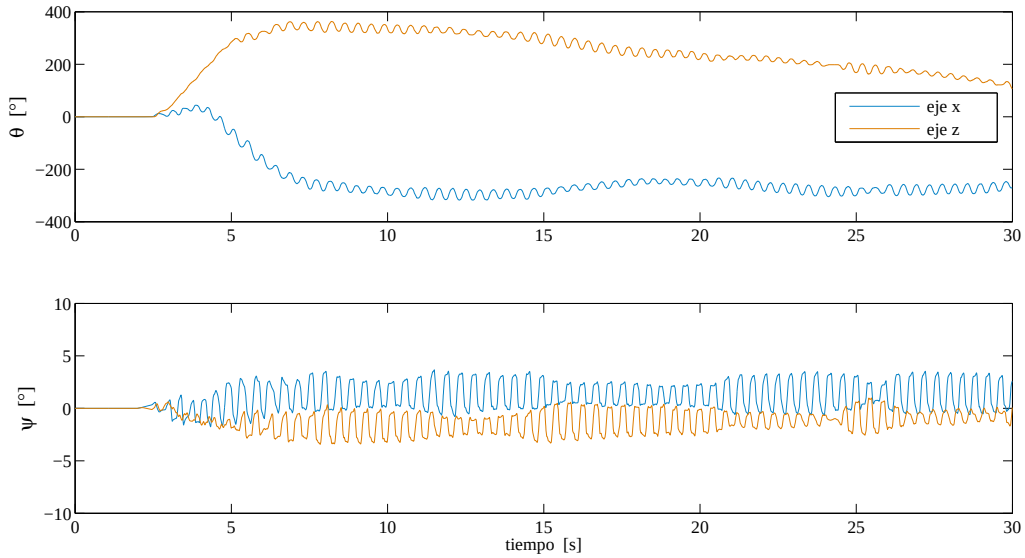


Figura 6.5: Comparativa entre los estados del sistema en los ejes x y z ante el controlador por asignación de polos en el modelo 1.

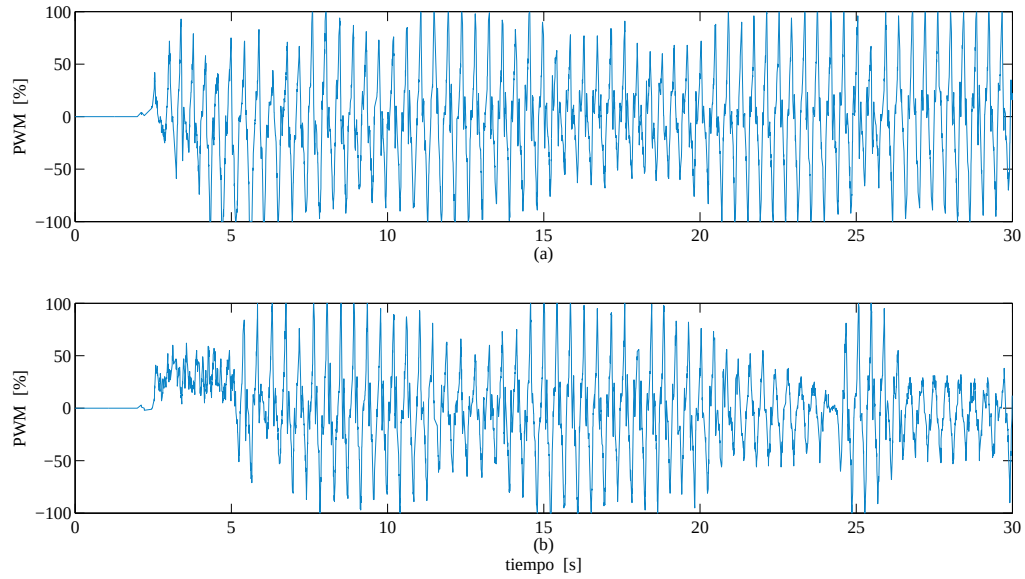


Figura 6.6: Comparativa de las señales que muestran el ciclo de trabajo del PWM del sistema ante el controlador por asignación de polos en el modelo 1; (a) eje x, (b) eje z.

6.2.2. Respuesta del modelo 2

En la Figura 6.7 se muestra la respuesta del sistema NXT ballbot ante el controlador por asignación de polos en el modelo 2. Como se puede apreciar, el controlador no es capaz de

estabilizar al sistema en lazo cerrado. Esta respuesta es completamente diferente a la mostrada en la simulación donde el controlador logró estabilizar completamente al sistema.

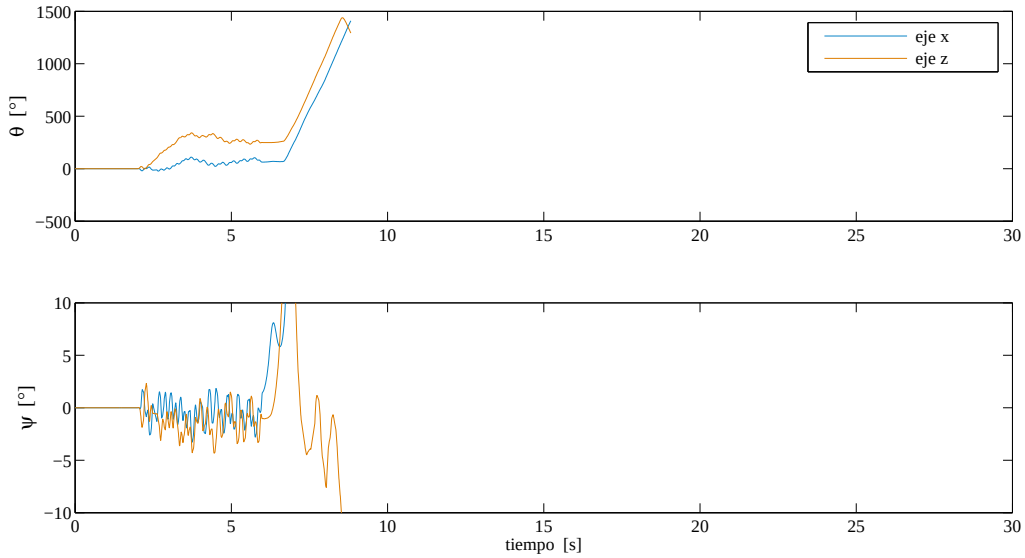


Figura 6.7: Comparativa entre los estados del sistema en los ejes x y z ante el controlador por asignación de polos en el modelo 2.

Analizando la respuesta respecto a la demanda en el esfuerzo de control requerida por el sistema, en la Figura 6.8 se observa como en ambos planos la demanda en el esfuerzo de control llega a un valor máximo el cual no es suficiente para estabilizar al sistema.

6.3. Controlador LQR mediante LMIs

Con el fin de comprobar si la inclusión de las LMIs mejora el desempeño en la respuesta del sistema NXT ballbot, a continuación se presentan los resultados adquiridos experimentalmente.

6.3.1. Respuesta del modelo 1

Como se muestra en la Figura 6.9, la respuesta en lazo cerrado del sistema es semejante en ambos planos. El sobreimpulso del estado θ en el eje x alcanza los 200°, mientras en el eje z alcanza los 600°. En cuanto al estado ψ, este se encuentra acotado en una región entre ±6°. En este estado se puede apreciar como el desplazamiento a partir del origen es más notorio debido a la desviación que se produce en el sensor giroscópico. En comparación con la respuesta en simulación, la trayectoria seguida por ambos estados θ y ψ es muy similar a la presentada en los vértices 1 y 2, presentando un comportamiento subamortiguado. Así también, se observa como desaparecen las pequeñas oscilaciones alrededor de la trayectoria generada, pero en su

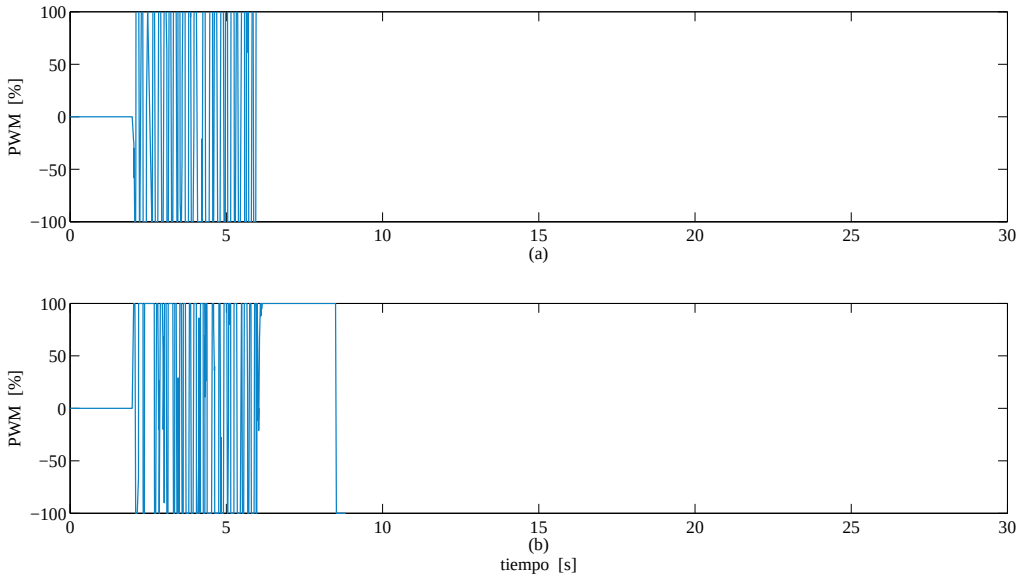


Figura 6.8: Comparativa de las señales que muestran el ciclo de trabajo del PWM del sistema ante el controlador por asignación de polos en el modelo 2; (a) eje x, (b) eje z.

lugar la amplitud de las oscilaciones es mas grande. Como conclusión, el controlador LQR mediante LMIs con los parámetros reportados por Yamamoto [78] en el modelo dinámico de la planta cumple con el objetivo de estabilización.

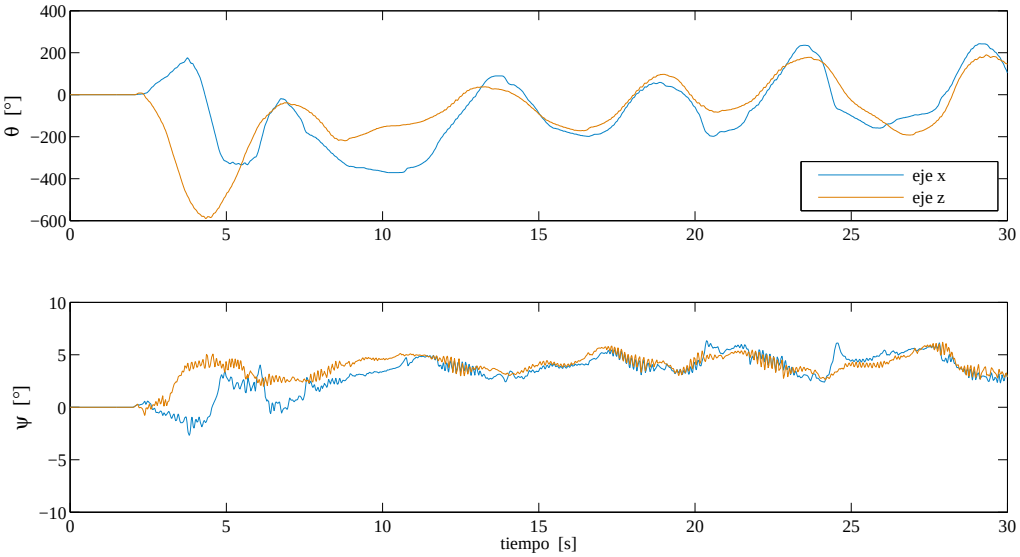


Figura 6.9: Comparativa entre los estados del sistema en los ejes x y z ante el controlador LQR mediante LMIs en el modelo 1.

En cuanto al esfuerzo de control demandado por el sistema, en la Figura 6.10 se muestran los ciclos de trabajo del PWM requeridos en ambos planos. Se puede observar como el ciclo de trabajo del PWM en ambos planos es muy semejante, así como la demanda del esfuerzo de control es mucho menor a la requerida por el controlador LQR normal. En comparación con los resultados obtenidos en la simulación, estos no se asemejan al obtenido experimentalmente, salvo que ambos presentan oscilaciones.

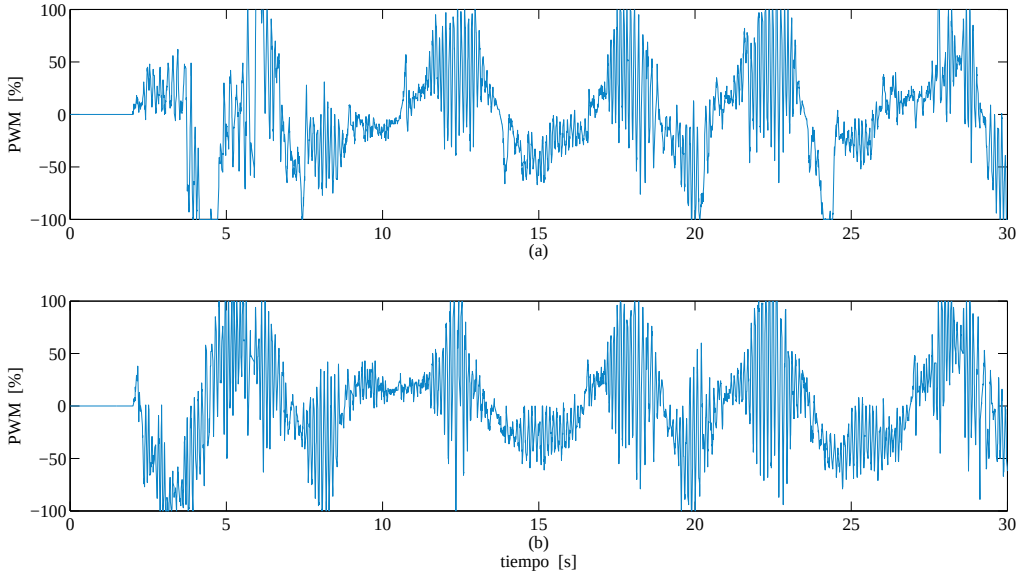


Figura 6.10: Comparativa de las señales que muestran el ciclo de trabajo del PWM del sistema ante el controlador LQR mediante LMIs en el modelo 1; (a) eje x, (b) eje z.

6.3.2. Respuesta del modelo 2

En la Figura 6.11, la respuesta en lazo cerrado del sistema es semejante en ambos planos. El sobreimpulso del estado θ en el eje x alcanza los 55°, mientras en el eje z alcanza los 70°. En cuanto al estado ψ, este se encuentra acotado en una región entre $\pm 3°$. En este estado se puede apreciar como el desplazamiento a partir del origen es menos notorio, estableciéndose alrededor de 1,5°. En comparación con la respuesta en simulación, la trayectoria seguida por ambos estados θ y ψ es muy similar a la presentada en los vértices 3 y 4, presentando un comportamiento ligeramente subamortiguado. Así también, se observa como vuelven a aparecer las pequeñas oscilaciones alrededor de la trayectoria generada. Como conclusión, el controlador LQR mediante LMIs con los parámetros obtenidos experimentalmente en el modelo dinámico de la planta cumple con el objetivo de estabilización.

En cuanto al esfuerzo de control demandado por el sistema, en la Figura 6.12 se muestran los ciclos de trabajo del PWM requeridos en ambos planos. Se puede observar como el ciclo de trabajo del PWM en ambos planos es muy semejante, sin embargo, el esfuerzo de control es mucho mayor al requerido en el modelo 1, pero muy similar al controlador LQR normal. En

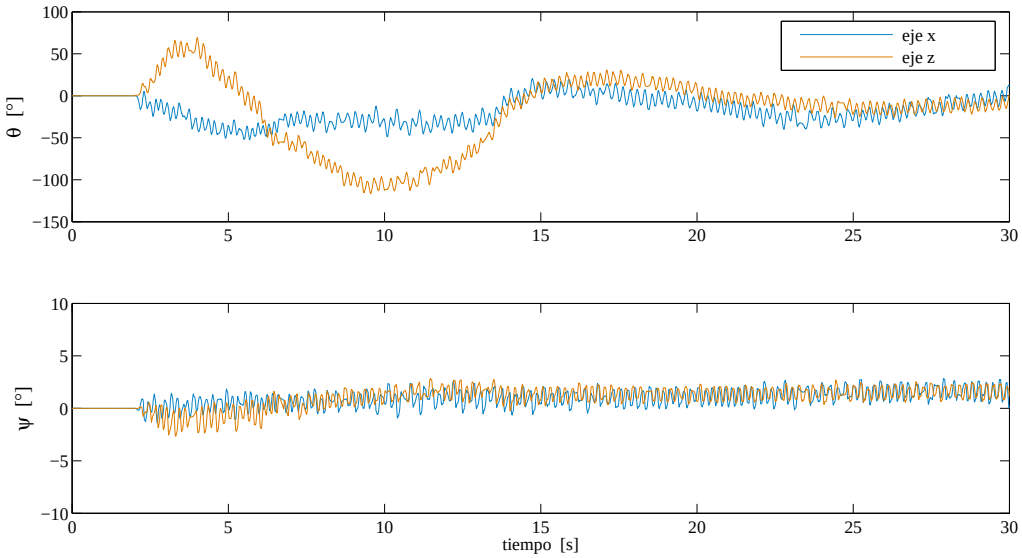


Figura 6.11: Comparativa entre los estados del sistema en los ejes x y z ante el controlador LQR mediante LMIs en el modelo 2.

comparación con los resultados obtenidos en la simulación, estos no se asemejan al obtenido experimentalmente.

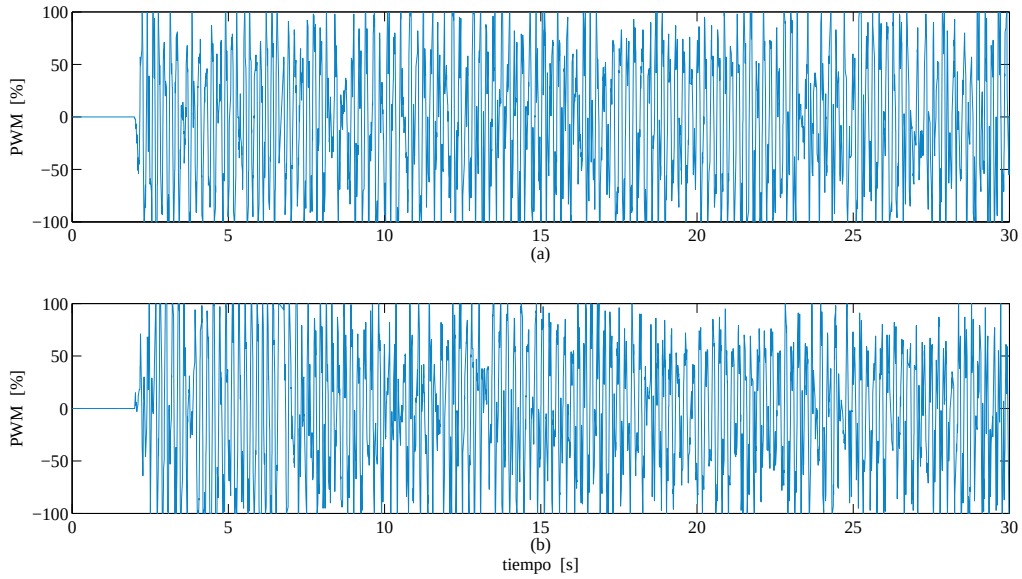


Figura 6.12: Comparativa de las señales que muestran el ciclo de trabajo del PWM del sistema ante el controlador LQR mediante LMIs en el modelo 2; (a) eje x, (b) eje z.

6.4. Controlador por ubicación de polos en regiones LMI

En esta sección se presentan los resultados obtenidos al llevar a cabo la implementación del controlador por ubicación de polos en regiones LMI. Este controlador no estaba contemplado inicialmente, sin embargo, se llevó a cabo para tener una versión del controlador por asignación de polos presentado anteriormente mediante la técnica de las LMI. Por otra parte, las formulaciones para llevar a cabo dicho controlador incluyen ciertos requisitos en el diseño que no se pueden establecer en el controlador LQR mediante LMIs, dado que no existe una metodología exacta para determinar los valores de las matrices de ponderación.

6.4.1. Respuesta del modelo 1

Como se muestra en la Figura 6.13, la respuesta en lazo cerrado del estado θ en ambos planos presentan un comportamiento oscilatorio alrededor del valor de referencia 0, se observa también que la amplitud de las oscilaciones en el eje x es muy grande, alcanzando un sobre impulso por arriba de los 400° que representa casi el doble del valor las oscilaciones en el eje z en el cual el sobre impulso supera los 200°. En cuanto al estado ψ, el comportamiento es muy semejante en ambos planos. Las oscilaciones presentadas son muy pequeñas y se encuentran acotadas en una región de $\pm 5°$, sin embargo, se presentan muchas desviaciones en el valor del cero nominal. En general, se observa como el resultado obtenido en ambos estados tiene un comportamiento más suave que los controladores presentados anteriormente, los cuales presentaban pequeñas oscilaciones al rededor de la trayectoria generada. Comparando la respuesta con la que se obtuvo en la simulación, la respuesta se asemeja a la presentada en los vértices 1 y 2. Como conclusión, el controlador por ubicación de polos en regiones LMI con los parámetros reportados por Yamamoto [78] en el modelo dinámico de la planta cumple con el objetivo de estabilización.

Analizando el esfuerzo de control requerido para estabilizar al sistema NXT ballbot, en la Figura 6.14 se observa como en ambos planos la demanda en el esfuerzo de control es mínima en comparación con la requerida por los controladores presentados anteriormente. Respecto a los valores presentados en la simulación, ninguno coincide con el comprotamiento presentado experimentalmente.

6.4.2. Respuesta del modelo 2

Para el caso del modelo 2, se oberva en la Figura 6.15 como la respuesta en lazo cerrado del sistema es muy semejante a la obtenida en el modelo 1. Sin embargo, el estado θ presenta un desplazamiento inicial lejos del punto de equilibrio así como un sobre impulso que alcanza los 400°. Conforme transcurre el tiempo se observa como la respuesta del estado se va acercando al punto de equilibrio 0. En cuanto al estado ψ, el comportamiento en ambos planos presenta una desviación de 3° en el valor del cero nominal, hecho por el cual la respuesta se encuentra acotada en una región entre $-1°$ y $+7°$. Nuevamente el comportamiento obtenido presenta una trayectoria mas suave que los controladores presentados anteriormente. Como conclusión, el controlador por ubicación de polos en regiones LMI con los parámetros obtenidos

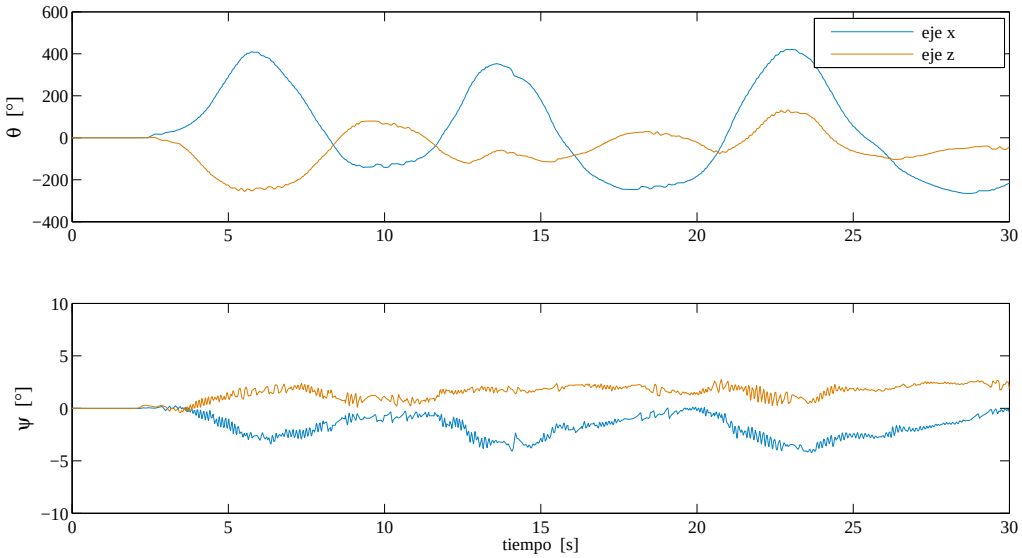


Figura 6.13: Comparativa entre los estados del sistema en los ejes x y z ante el controlador por ubicación de polos en regiones LMI en el modelo 1.

experimentalmente en el modelo dinámico de la planta cumple con el objetivo de estabilización.

Por último, se analiza el esfuerzo de control requerido para estabilizar al sistema. En la Figura 6.16 se muestran los ciclos de trabajo requeridos por el sistema en ambos planos. Observando ambos esfuerzos de control requeridos, se concluye que el controlador en ambos panos demanda un esfuerzo de control muy semejante, incluso el valor de estos esfuerzos es mucho menor que todos los controladores presentados hasta ahora, inclusive este mismo controlador aplicado en el modelo 1. Como conclusión, para el bajo esfuerzo de control demandado por el sistema NXT ballbot, la respuesta obtenida es muy satisfactoria.

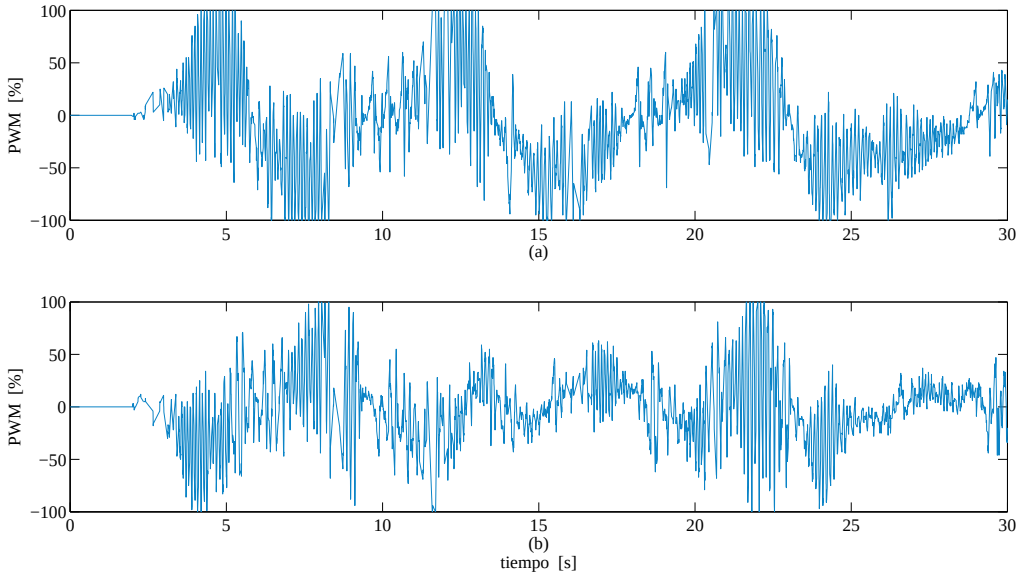


Figura 6.14: Comparativa de las señales que muestran el ciclo de trabajo del PWM del sistema ante el controlador por ubicación de polos en regiones LMI en el modelo 1; (a) eje x, (b) eje z.

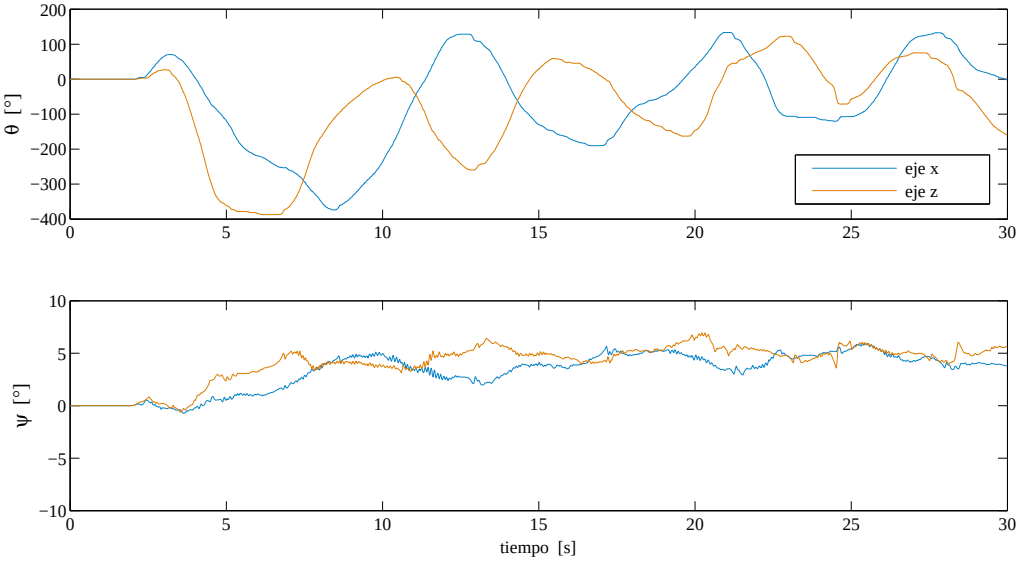


Figura 6.15: Comparativa entre los estados del sistema en los ejes x y z ante el controlador por ubicación de polos en regiones LMI en el modelo 2.

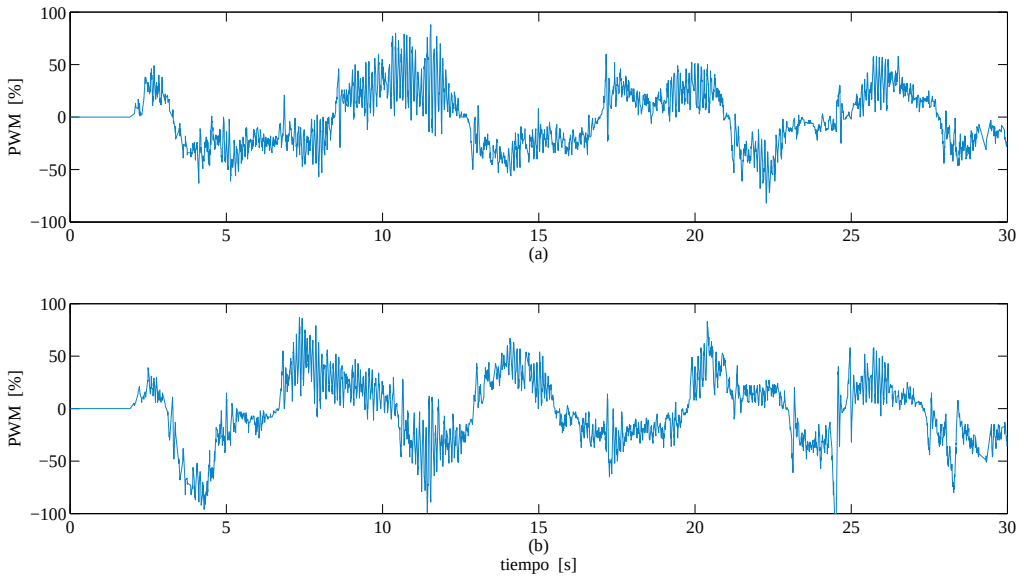


Figura 6.16: Comparativa de las señales que muestran el ciclo de trabajo del PWM del sistema ante el controlador por ubicación de polos en regiones LMI en el modelo 2; (a) eje x, (b) eje z.

Capítulo 7

Conclusiones y trabajo futuro

En este capítulo se resumen las observaciones realizadas a través del desarrollo de esta tesis. Así como también, se proponen nuevos retos y mejoras para darle seguimiento al tema desarrollado.

7.1. Conclusiones

Se comprobó la importancia de modelar matemáticamente el comportamiento de un sistema dinámico, con el fin de poder modificar su comportamiento para obtener una respuesta deseada a través de algoritmos de control. Considerando que un sistema físico no se puede modelar de forma exacta (cualquier modelo es sólo una aproximación del sistema real), siempre existirán incertidumbres en los modelos matemáticos, mas aún, los sistemas reales presentan, en términos generales, comportamientos no lineales. Lo anterior hace que el diseño de sistemas de control basados en modelos dinámicos sea una tarea sumamente compleja. En el caso del sistema NXT ballbot se recurrió a métodos de linealización alrededor de un punto de equilibrio, para tener una aproximación lineal del modelo y poder resolver el problema de control con algoritmos lineales. Esta transformación hace que la imprecisión del modelo sea aún mayor. Además, al caracterizar experimentalmente los actuadores, se observó cómo estos datos difieren según el procedimiento y las condiciones en las que se lleve a cabo, se revisó en la literatura y diversos autores reportan valores diferentes para los motores NXT que se emplearon en el sistema NXT ballbot, por lo que todos estos valores fueron contemplados inicialmente. Sin embargo, la formulación de un modelo politópico que incluya todas las posibles variaciones en los parámetros es prácticamente imposible debido a que el número de vértices del modelo politópico crece con respecto al número de parámetros inciertos a razón de 2^{n_p}. Además, algunos parámetros aparecen de forma no lineal en el modelo, lo cual requiere la aplicación de un tratamiento matemático del modelo mediante fracciones lineales para poder aplicar el enfoque de LMIs. Por otra parte, se presentaron ruidos en las lecturas de los sensores giroscópicos, problema que se abordó mediante una actualización del valor cero nominal y un filtro pasa bajas como se reporta en [78].

Una vez que se obtuvo tanto el modelo nominal como el modelo politópico para los modelos 1 y 2, se realizaron pruebas de estabilidad, observabilidad y controlabilidad. Como resultado de estas pruebas, se observó que todos los modelos no eran estables en lazo abierto, esto al contener al menos un polo en el semiplano derecho del plano complejo. Sin embargo, los modelos cumplían con la condición de observabilidad y controlabilidad, por lo que se buscó diseñar un controlador que estabilizara al sistema en lazo cerrado mediante el esquema básico de retro alimentación de todos los estados, además que compensara la incertidumbre en los parámetros del motor NXT y que el esfuerzo de control requerido para mantener al sistema NXT ballbot verticalmente fuera el mínimo posible. Por último, se compararía el desempeño de este controlador con algún algoritmo de control reportado en la literatura, en este caso el controlador óptimo LQR [78] y un controlador clásico mediante asignación de polos [32].

Para llevar acabo el diseño del controlador se hizo uso de una técnica matemática denominada *desigualdades matriciales lineales* o simplemente LMIs, la cual convirtió el problema de control en un problema de optimización convexa al que se le añadieron diversas restricciones en los requisitos de diseño. En este caso, el principal requisito fue el concepto de estabilidad cuadrática del sistema NXT ballbot, la restricción en el esfuerzo de control, la robustez ante incertidumbre en los parámetros, y en el caso especial del controlador por ubicación de polos en regiones LMIs, que dichas regiones satisfacieran ciertos parámetros en la respuesta transitoria del sistema. Una vez que se formularon las restricciones para los algoritmos de control LQR y ubicación de polos, la solución a estas restricciones para determinar las ganancias de los controladores se obtuvo mediante el paquete de *Matlab*® *LMI Control Toolbox*® que mediante avanzados algoritmos de optimización determinaron si el problema de control era o no factible. Lo que hizo que se modificaran los valores en la restricción de control en el caso del LQR y para el caso de las regiones LMI, se redujo el límite para el coeficiente de amortiguamiento y se amplió el tiempo de respuesta, esto debido a que el esfuerzo requerido para estabilizar al sistema con las condiciones iniciales eran muy grandes y el motor NXT no sería capaz de proporcionar tal demanda. Una vez que se encontró que la solución a las restricciones establecidas era factible, se procedió a realizar un análisis temporal del controlador en el modelo del sistema NXT ballbot capturado en *Matlab*® lo que fue de gran ayuda para determinar si efectivamente se cumplía la estabilización del sistema en lazo cerrado y conocer la nueva ubicación de los polos.

En cuanto a la instalación del *software* de CAD *Solidworks*® no se presentó ningún problema, sin embargo, durante la realización de las piezas faltantes en la librería proporcionada por la universidad Carnegie Mellon [73] se revisó que las dimensiones de las piezas reportadas en [62] no correspondian a las de la librería, hecho por el cuál se diseñaron todas las piezas de la librería con las nuevas medidas. Otro problema que se presentó, ahora durante el ensamblado de las piezas, fue que ciertos ensambles no coincidían para ser posicionados mediante relaciones de posición entre centros, ya que en el ensamble físico estas piezas embonaban a presión, esto se solucionó creando algunas piezas a la medida y en otros casos mediante otras relaciones de posición que aseguraran la ubicación de la pieza. Por otra parte, para la implementación del modelo del sistema NXT ballbot en *Simulink*®, se instalaron dos complementos: el primero denominado *ECROBOT* en el cuál estaba basado el diseño de Yamamoto [78] y el segundo

complemento denominado *support hardware for lego mindstorms* el cuál era un complemento oficial por parte de *Matlab*® a partir de la versión R2012a. Luego de analizar las características que ofrecian ambos complementos, se eligió el segundo complemento, esto debido a que aún no se había desarrollado el controlador para el NXT ballbot con dicho complemento y además ofrecía el monitoreo de señales en tiempo real sin necesidad de guardar los datos y posteriormente graficarlos, como lo hacia el primer complemento. Por lo que se realizó una nueva versión del controlador diseñado por Yamamoto [78], ahora en el complemento oficial de *Matlab*®. En este nuevo modelo del controlador para el sistema NXT ballbot, se cargó el archivo que contenia las ganancias de los controladores diseñados y se realizaron las simulaciones correspondientes para cada modelo. Esto con el fin de conocer de antemano cómo se esperaba que la respuesta experimental se comportara.

Luego de comprobar en las simulaciones que efectivamente los controladores diseñados mediante formulaciones LMI lograban que los controladores fueran robustos y cumplieran con las demás restricciones de diseño en comparación con los controladores LQR [78] y por asignación de polos [32]. Se procedió a realizar las implementaciones físicas de los controladores en el sistema LEGO NXT ballbot. En esta parte se apreció la ventaja de *Simulink*® para monitorear y guardar las señales del sistema en tiempo real mediante la conexión *bluetooth* entre el ladrillo NXT y la PC. Así tambien, que ofrece la capacidad de hacer modificaciones en tiempo real en el controlador. Como establecer cierta entrada de referencia al sistema, lo que resulta muy útil para implementar técnicas de navegación.

Al finalizar la implementación de los controladores diseñados y evaluar las diferentes respuestas del sistema en lazo cerrado, se confirmó la hipótesis planteada de que es posible desacoplar el controlador completo del sistema NXT ballbot en dos controladores independientes y que actuan en los planos idénticos *coronal* $x-y$ y *sagital* $z-y$. Esta conclusión se da una vez que las respuestas en ambos planos resultaron ser muy semejantes y cumplieron con el objetivo de estabilizar al sistema NXT ballbot. La segunda hipótesis planteada acerca del uso de la herramienta LMI para sintonizar el controlador LQR también se cumple, esto al observar como el desempeño del controlador LQR mediante LMIs es mejor en términos de robustez ante incertidumbre en los datos y el esfuerzo de control requerido que el reportado por Yamamoto [78]. En cuanto a la tercera hipótesis planteada, analizando las respuestas únicamente entre el controlador óptimo LQR y un controlador clásico como lo fué el de asignación de polos. Se observa que efectivamente el controlador LQR presenta una mejor respuesta del error en estado estable, puesto que el controlador LQR a pesar de ser capaz de estabilizar al sistema NXT ballbot en los modelos 1 y 2, el estado θ se aproxima a 0. Por otra parte, el controlador por asignación de polos solo estabilizó al modelo 1 del sistema NXT ballbot y el estado θ se encontró muy lejos del valor 0. Por último, es claro que todos los controladores se pudieron implementar en el sistema LEGO NXT ballbot y en la mayoría de los casos la respuesta fue semejante a las simulaciones. Como trabajo adicional y para reforzar la utilidad de emplear la técnica LMI en el diseño de controladores, el controlador por ubicación de polos en regiones LMI que se diseñó mostró mejores respuestas del sistema en lazo cerrado que los mencionados anteriormente si se compara el esfuerzo de control requerido para estabilizar al sistema, hecho por el cual presenta grandes oscilaciones en el estado θ pero sin presentar las

pequeãs oscilaciones a través de dicha trayectoría. Es decir la respuesta del controlador es mas suave al de los demás controladores.

7.2. Trabajos futuros

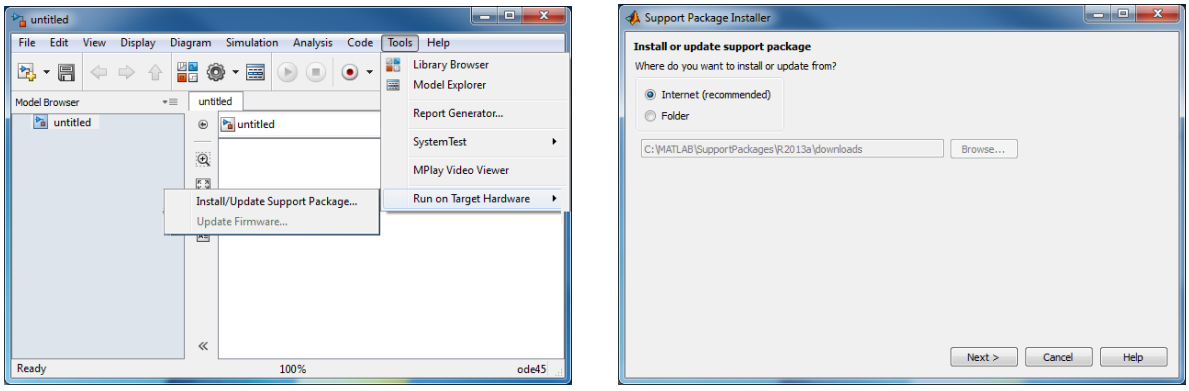
Con el fin de poder darle seguimiento al tema desarrollado en esta tesis, a continuación se enlistan los posibles puntos a mejorar, así como nuevos objetivos a desarrollar.

- Mejorar el filtro que elimina la desviación que presenta el sensor giroscópico para una lectura más confiable.
- Incluir un sensor de aceleración/inclinación para fusionar las lecturas con las proporcionadas por el sensor giroscópico.
- Implementar un observador o filtro de Kalman para eliminar la suma del error que genera la integración del estado $\dot{\psi}$.
- Una vez que se mejoren las lecturas de los sensores y el desempeño de los controladores, se podría abordar el modelo no lineal del sistema.
- Una vez abordado el sistema lineal y no lineal del sistema, se pueden implementar técnicas de navegación en el sistema NXT ballbot.
- Se pueden incluir más parámetros en el modelo politópico, los cuales se puedan variar físicamente.
- Una vez que se cuenta con el modelo en *Solidworks*® se puede abordar una co-simulación entre *Simulink*®-*SimMechanics*® o *Simulink*®-*ADAMS*® para visualizar gráficamente el comportamiento del sistema NXT Balbot sin necesidad de llegar a implementarlo físicamente.
- Desarrollar un modelo a mayor escala del ballbot.
- Equipar al modelo a mayor escala con periféricos que le permitan interactuar con otros dispositivos y que su plataforma sea abierta a cualquier aplicación que pueda incorporarse en un futuro.

Apéndice A

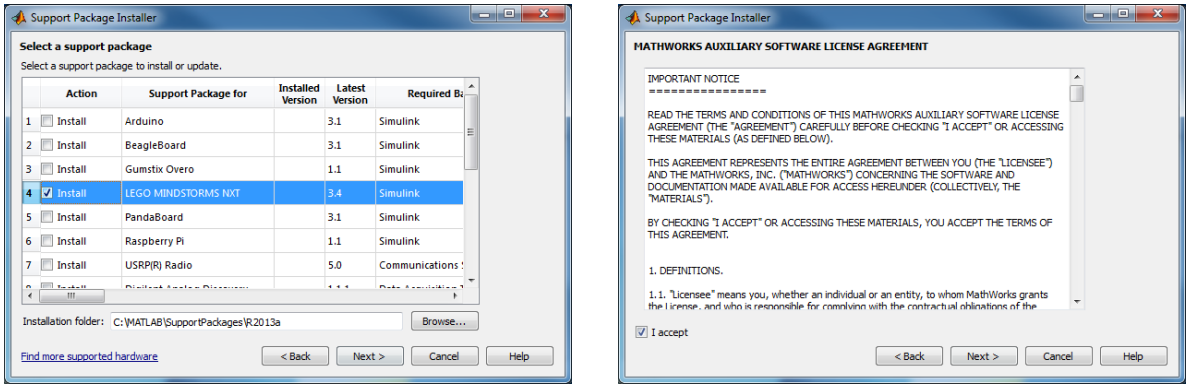
Instalación del soporte del kit *Lego NXT Mindstorms®* para *Simulink®*

A continuación se muestra la serie de pasos para la instalación del paquete *support hardware for lego mindstorms* de *Simulink®*.



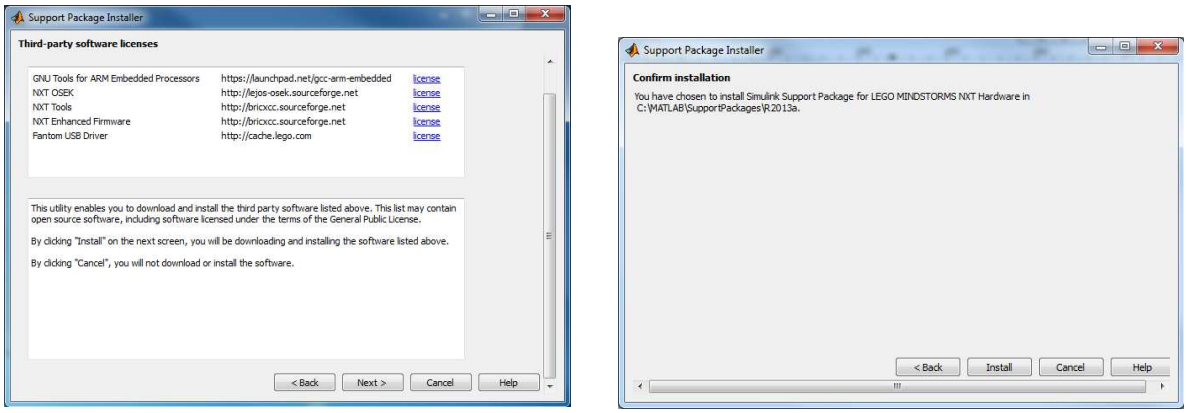
(a) (b)

Figura A.1: (a) Paso de instalación 1; (b) Paso de instalación 2.



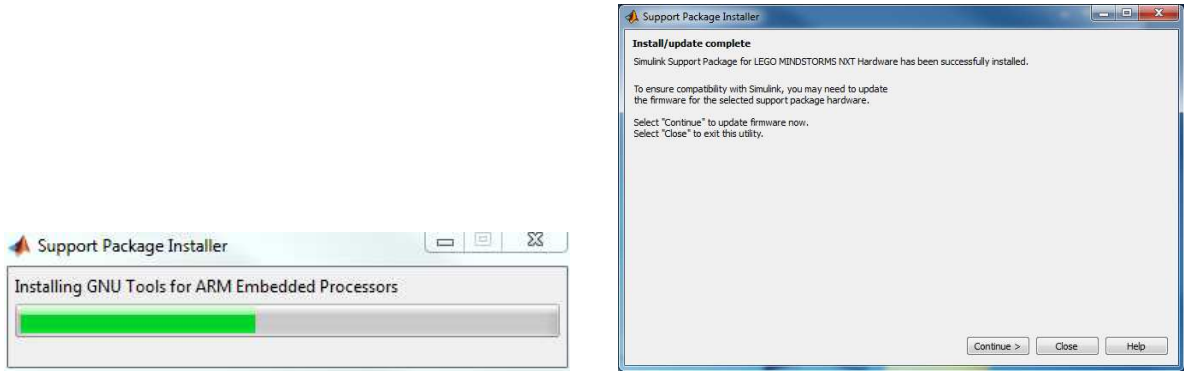
(a) (b)

Figura A.2: (a) Paso de instalación 3; (b) Paso de instalación 4.



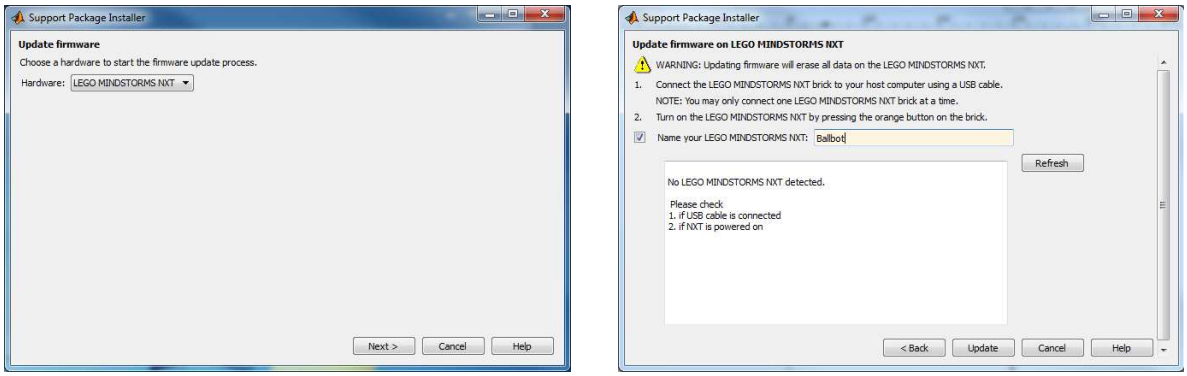
(a) (b)

Figura A.3: (a) Paso de instalación 5; (b) Paso de instalación 6.



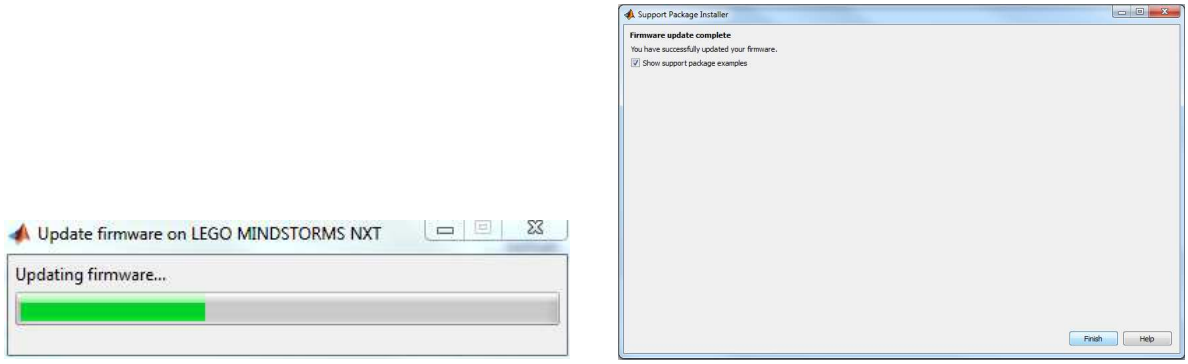
(a) (b)

Figura A.4: (a) Paso de instalación 7; (b) Paso de instalación 8.



(a) (b)

Figura A.5: (a) Paso de instalación 9; (b) Paso de instalación 10.



(a) (b)

Figura A.6: (a) Paso de instalación 11; (b) Paso de instalación 12.

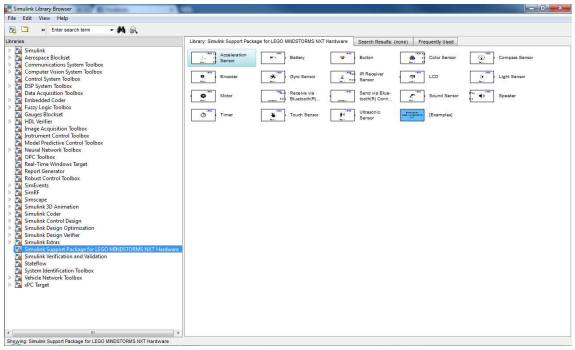
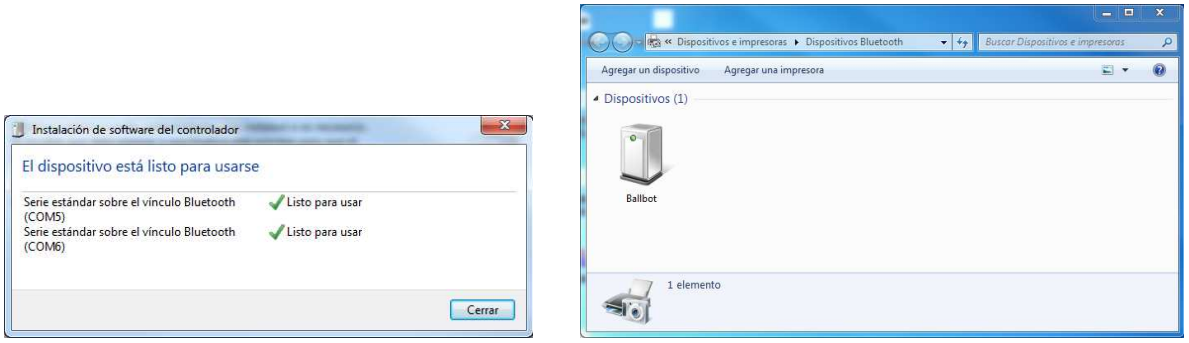


Figura A.7: Librería instalada en Simulink

A.1. Configuración de la comunicación *bluetooth* entre el ladrillo NXT y la PC.

En esta parte se describe cómo emparejar la comunicación entre el ladrillo NXT y la PC donde se ejecuta el modelo del controlador implementado en *Simulink*®. Para comenzar se debe acceder al menu *bluetooth* del ladrillo NXT y posteriormente seleccionar la opción *on* (*encender bluetooth*) y activar el modo *visible*. Una vez hecho esto, en la computadora nos dirigimos al icono de la aplicación *bluetooth* y seleccionamos la opción agregar un nuevo dispositivo, la aplicación buscará nuevos dispositivos que se encuentren en modo visible, tras encontrar al dispositivo NXT, se solicitará una clave de emparejamiento. En este caso se estableció la clave 1111. Tras realizar el emparejamiento se instalarán los controladores del ladrillo NXT para su uso con el *bluetooth* como se muestra en la Figura A.8(a). Para comprobar que se ha instalado correctamente, este deberá aparecer en la sección *Dispositivos Bluetooth* como se muestra en la Figura A.8(b).



(a) (b)

Figura A.8: (a) Instalación de los controladores del ladrillo NXT para su uso con el *bluetooth*; (b) Comprobación de emparejamiento entre el ladrillo NXT y la PC.

Para conocer el número del puerto *COM* a utilizar para el intercambio de datos, nos dirigimos a las propiedades del dispositivo *bluetooth* y en la pestaña de *Servicios* observamos que en este caso el número a utilizar es *COM 5* como se muestra en la Figura A.9.

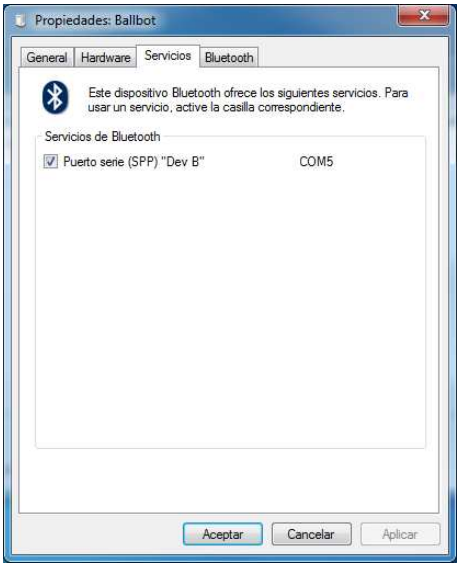


Figura A.9: Propiedades del dispositivo bluetooth para el ladrillo NXT.

Apéndice B

Códigos para la caracterización del motor NXT

El primer modelo en *Simulink*® para la caracterización del motor consta simplemente de un bloque que nos permite establecer el ciclo de trabajo del PWM del motor NXT, el cual variamos desde 10 a 100 % con incrementos de 10. Otro aspecto importante es que se empleó la configuración para frenado de motor denominado en el bloque por *brake*, como se muestra en la figura B.2(a).

El segundo modelo en *Simulink*® nos permite sensar mediante el encoder interno del motor, la velocidad a la que gira el motor NXT y como esta decae al aplicarle un frenado mediante la acción de su propia inercia, para este caso la configuración seleccionada del frenado se denomina *coast*, como se muestra en la Figura B.2(b).

Por último, luego de recolectar los datos de *Simulink*® , estos se exportaron a *Matlab*®, donde se procesaron para determinar los parámetros de los motores mediante el siguiente código en *Matlab*®.

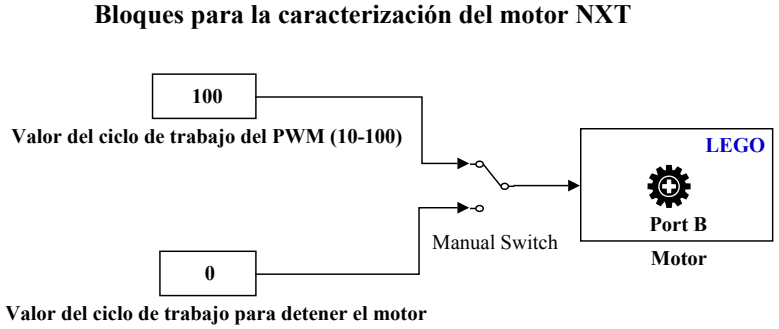
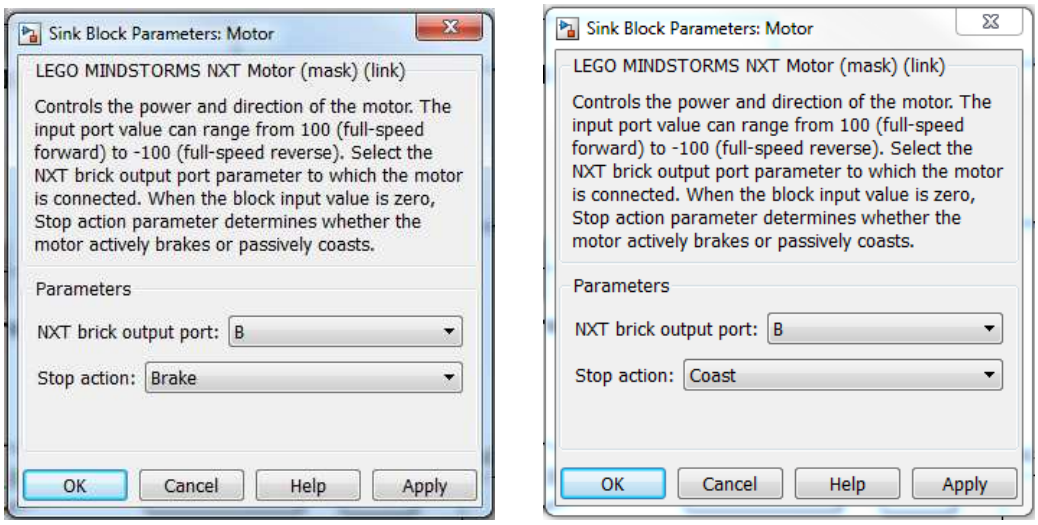


Figura B.1: Modelo en *Simulink*® para asignarle un ciclo de trabajo al PWM del motor.



(a) (b)

Figura B.2: (a) Configuración para el frenado del motor tipo *brake*; (b) Configuración para el frenado del motor tipo *coast*.

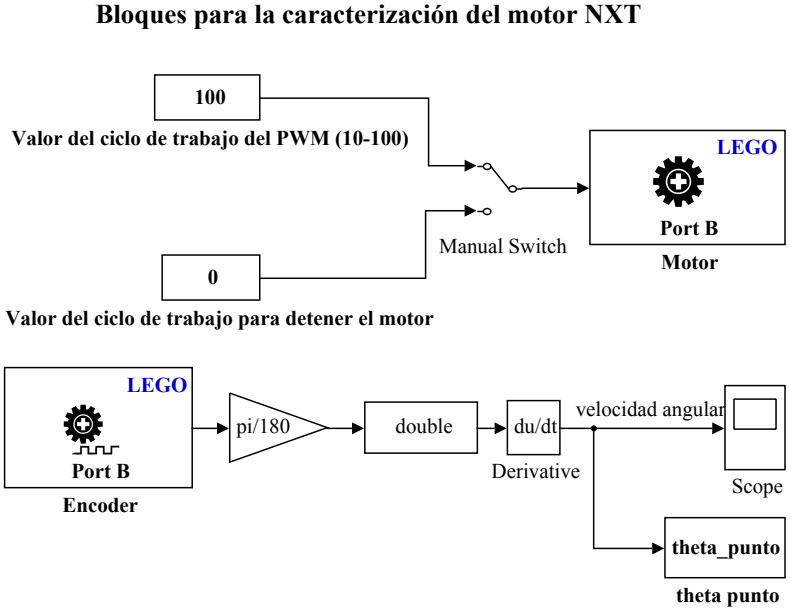


Figura B.3: Modelo en *Simulink*® para asignarle un ciclo de trabajo de 100 al PWM del motor y sensar como decae la velocidad al frenar el motor mediante la acción de su propia inercia .

Código fuente B.1: Determinacion_parametros_NXT.m

```
%% Obtención de los parámetros físicos del motor NXT
% Este archivo carga los datos que se obtuvieron mediante el osciloscopio
% al realizar las diferentes pruebas de caracterización al motor NXT y
% posteriormente muestra los valores de los parámetros luego de una
% regresión lineal mediante mínimos cuadrados.

% Valor de la resistencia interna del motor, medida directamente con un
% multímetro
R=4.6;      %Ohms
% Importación de los datos de las respuestas del motor a determinado ciclo
% de trabajo del PWM
load voltaje10.mat;
load corriente10.mat;
load Tiempo10.mat;
load voltaje20.mat;
load corriente20.mat;
load Tiempo20.mat;
load voltaje30.mat;
load corriente30.mat;
load Tiempo30.mat;
load voltaje40.mat;
load corriente40.mat;
load Tiempo40.mat;
load voltaje50.mat;
load corriente50.mat;
load Tiempo50.mat;
load voltaje60.mat;
load corriente60.mat;
load Tiempo60.mat;
load voltaje70.mat;
load corriente70.mat;
load Tiempo70.mat;
load voltaje80.mat;
load corriente80.mat;
load Tiempo80.mat;
load voltaje90.mat;
load corriente90.mat;
load Tiempo90.mat;
load voltaje100.mat;
load corriente100.mat;
load Tiempo100.mat;
%- Gráfica de las señales de voltaje y corriente del motor
figure1 = figure('Color',[1 1 1]);
axes1 = axes('Parent',figure1,'FontSize',14,'FontName','Times New Roman');
box(axes1,'on');
hold(axes1,'all')
plot(Tiempo10,Voltaje10,'b',Tiempo10,Corriente10*100,'r','Parent',axes1);
xlabel('tiempo  [s]','FontSize',14,'FontName','Times New Roman');
ylabel(['Voltaje del motor  [V]','; Corriente del motor  i_a  ...
    [mA]'],'FontSize',14,'FontName','Times New Roman');
figure2 = figure('Color',[1 1 1]);
axes1 = axes('Parent',figure2,'FontSize',14,'FontName','Times New Roman');
```

```
box(axes1,'on');
hold(axes1,'all')
plot(Tiempo20,Voltaje20,'b',Tiempo20,Corriente20*100,'r','Parent',axes1);
xlabel('tiempo  [s]','FontSize',14,'FontName','Times New Roman');
ylabel(['Voltaje del motor  [V]','; Corriente del motor  i_a  ...
    [mA]'],'FontSize',14,'FontName','Times New Roman');
figure3 = figure('Color',[1 1 1]);
axes1 = axes('Parent',figure3,'FontSize',14,'FontName','Times New Roman');
box(axes1,'on');
hold(axes1,'all')
plot(Tiempo30,Voltaje30,'b',Tiempo30,Corriente30*100,'r','Parent',axes1);
xlabel('tiempo  [s]','FontSize',14,'FontName','Times New Roman');
ylabel(['Voltaje del motor  [V]','; Corriente del motor  i_a  ...
    [mA]'],'FontSize',14,'FontName','Times New Roman');
figure4 = figure('Color',[1 1 1]);
axes1 = axes('Parent',figure4,'FontSize',14,'FontName','Times New Roman');
box(axes1,'on');
hold(axes1,'all')
plot(Tiempo40,Voltaje40,'b',Tiempo40,Corriente40*100,'r','Parent',axes1);
xlabel('tiempo  [s]','FontSize',14,'FontName','Times New Roman');
ylabel(['Voltaje del motor  [V]','; Corriente del motor  i_a  ...
    [mA]'],'FontSize',14,'FontName','Times New Roman');
figure5 = figure('Color',[1 1 1]);
axes1 = axes('Parent',figure5,'FontSize',14,'FontName','Times New Roman');
box(axes1,'on');
hold(axes1,'all')
plot(Tiempo50,Voltaje50,'b',Tiempo50,Corriente50*100,'r','Parent',axes1);
xlabel('tiempo  [s]','FontSize',14,'FontName','Times New Roman');
ylabel(['Voltaje del motor  [V]','; Corriente del motor  i_a  ...
    [mA]'],'FontSize',14,'FontName','Times New Roman');
figure6 = figure('Color',[1 1 1]);
axes1 = axes('Parent',figure6,'FontSize',14,'FontName','Times New Roman');
box(axes1,'on');
hold(axes1,'all')
plot(Tiempo60,Voltaje60,'b',Tiempo60,Corriente60*100,'r','Parent',axes1);
xlabel('tiempo  [s]','FontSize',14,'FontName','Times New Roman');
ylabel(['Voltaje del motor  [V]','; Corriente del motor  i_a  ...
    [mA]'],'FontSize',14,'FontName','Times New Roman');
figure7 = figure('Color',[1 1 1]);
axes1 = axes('Parent',figure7,'FontSize',14,'FontName','Times New Roman');
box(axes1,'on');
hold(axes1,'all')
plot(Tiempo70,Voltaje70,'b',Tiempo70,Corriente70*100,'r','Parent',axes1);
xlabel('tiempo  [s]','FontSize',14,'FontName','Times New Roman');
ylabel(['Voltaje del motor  [V]','; Corriente del motor  i_a  ...
    [mA]'],'FontSize',14,'FontName','Times New Roman');
figure8 = figure('Color',[1 1 1]);
axes1 = axes('Parent',figure8,'FontSize',14,'FontName','Times New Roman');
box(axes1,'on');
hold(axes1,'all')
plot(Tiempo80,Voltaje80,'b',Tiempo80,Corriente80*100,'r','Parent',axes1);
xlabel('tiempo  [s]','FontSize',14,'FontName','Times New Roman');
ylabel(['Voltaje del motor  [V]','; Corriente del motor  i_a  ...
```

```
    [mA]'],'FontSize',14,'FontName','Times New Roman');
figure9 = figure('Color',[1 1 1]);
axes1 = axes('Parent',figure9,'FontSize',14,'FontName','Times New Roman');
box(axes1,'on');
hold(axes1,'all')
plot(Tiempo90,Voltaje90,'b',Tiempo90,Corriente90*100,'r','Parent',axes1);
xlabel('tiempo  [s]','FontSize',14,'FontName','Times New Roman');
ylabel(['Voltaje del motor  [V]','; Corriente del motor  i_a  ...
    [mA]'],'FontSize',14,'FontName','Times New Roman');
figure10 = figure('Color',[1 1 1]);
axes1 = axes('Parent',figure10,'FontSize',14,'FontName','Times New Roman');
box(axes1,'on');
hold(axes1,'all')
plot(Tiempo100,Voltaje100,'b',Tiempo100,Corriente100*100,'r','Parent',axes1);
xlabel('tiempo  [s]','FontSize',14,'FontName','Times New Roman');
ylabel(['Voltaje del motor  [V]','; Corriente del motor  i_a  ...
    [mA]'],'FontSize',14,'FontName','Times New Roman');
%- Obtención de los valores promedio del voltaje de entrada y corriente.
v10=mean(Voltaje10);
i10=mean(Corriente10);
v20=mean(Voltaje20);
i20=mean(Corriente20);
v30=mean(Voltaje30);
i30=mean(Corriente30);
v40=mean(Voltaje40);
i40=mean(Corriente40);
v50=mean(Voltaje50);
i50=mean(Corriente50);
v60=mean(Voltaje60);
i60=mean(Corriente60);
v70=mean(Voltaje70);
i70=mean(Corriente70);
v80=mean(Voltaje80);
i80=mean(Corriente80);
v90=mean(Voltaje90);
i90=mean(Corriente90);
v100=mean(Voltaje100);
i100=mean(Corriente100);
%- Creación de vectores con las respuestas del motor
V_motor=[v10 v20 v30 v40 v50 v60 v70 v80 v90 v100];
I_motor=[i10 i20 i30 i40 i50 i60 i70 i80 i90 i100];
load('T_motor.mat');
%% Determinación de Parámetros del motor NXT
w_motor = (2*pi*ones(size(T_motor))) ./ (T_motor*180); % Velocidad motor
eb = (V_motor-R*I_motor);    % fuerza electromotiz
%- Determinación de kb por regresión lineal
% eb = kb*w_motor + b
W = [w_motor ; ones(size(w_motor))]';
theta_eb = W\eb';
kb = theta_eb(1);            % Constante eléctrica
fem = W*theta_eb;      %fem respecto al valor de la regresión lineal de ke
% Representación gráfica de los valores reales y regresión lineal
figure11 = figure('Color',[1 1 1]);
```

```
149 axes1 = axes('Parent',figure11,'FontSize',14,'FontName','Times New Roman');
150 box(axes1,'on');
151 hold(axes1,'all')
152 plot(w_motor,eb,'*',w_motor,fem,'r','Parent',axes1);
153 xlabel('Velocidad del motor \omega  ...
        [rad/s]','FontSize',14,'FontName','Times New Roman');
154 ylabel('fem  e_b  [V]','FontSize',14,'FontName','Times New Roman');
155 title(['Kb = ' num2str(kb) ' V.s '],'FontSize',14,'FontName','Times New ...
        Roman');
156  %- Constante de armadura (par) kp
157 kp = kb;               %siempre que ambas estén en unidades del SI
158  %- Regresión lineal para determinar el coef. de fric. viscosa be [N.m.s]
159  % Tm = be*w_motor + Tr
160 Tm=kp*I_motor;
161 theta_tm=W\Tm';
162 be=theta_tm(1);
163 Tr=theta_tm(2);
164  % Representación gráfica
165 figure12 = figure('Color',[1 1 1]);
166 axes1 = axes('Parent',figure12,'FontSize',14,'FontName','Times New Roman');
167 box(axes1,'on');
168 hold(axes1,'all')
169 plot(w_motor,Tm,'*',w_motor,W*theta_tm,'r','Parent',axes1);
170 xlabel('Velocidad del motor \omega  ...
        [rad/s]','FontSize',14,'FontName','Times New Roman');
171 ylabel('Par del motor  \tau  [N.m]','FontSize',16,'FontName','Times New ...
        Roman');
172 title(['be = ' num2str(be) ' N.m.s ; Tr = ' num2str(abs(Tr)) ' ...
        N.m'],'FontSize',14,'FontName','Times New Roman')
173  %- Determinación de la Inductancia de armadura La
174  % Parametros del motor con un ciclo de trabajo al 50%
175 Kb = kb;
176 wr = w_motor(5);
177  % Representación gráfica de la corriente para determinar la pendiente
178 figure;
179 stairs(Corriente50)
180 xlabel('Muestras')
181 ylabel('Corriente del motor i_a  [A]')
182 title('Haz zoom y pulsa cualquier tecla para seleccionar el intervalo ...
        temporal')
183 grid
184  % Se selecciona el intervalo para el cálculo
185 pause
186 k = round(ginput(2));
187  % Se determina el intervalo temporal entre ambas velocidades y se calcula
188  % la pendiente mediante una regresión lineal
189 Y = Corriente50(k(1):k(2));
190 X = Tiempo50(k(1):k(2))-Tiempo50(k(1));
191 A = [X ones(size(X))];
192 th1 = A\Y;       % pendiente de la corriente
193 Vdc = mean(Voltaje50(k(1):k(2)));
194  % la pendiente se puede calcular también como
195  % ((Vdc-Kb*w_motor)/R-i(0))/L*R = di/dt,a partir de esta expresión se ...
```

```
    calcula L
L = ((Vdc-Kb*wr)/R-Corriente50(k(1)))*R/abs(th1(1));
% Representación gráfica del resultado
figure13 = figure('Color',[1 1 1]);
axes1 = axes('Parent',figure13,'FontSize',14,'FontName','Times New Roman');
box(axes1,'on');
hold(axes1,'all')
plot(X,Y,'*',X,A*th1,'-','Parent', axes1);
grid;
ylabel('Corriente del motor  i_a  [A]','FontSize',16,'FontName','Times ...
    New Roman');
xlabel('tiempo  [s]','FontSize',14,'FontName','Times New Roman');
title(['L = ' num2str(abs(L)) ' H'],'FontSize',14,'FontName','Times New ...
    Roman');
%- Determinación de la inercia del motor J
% Representación gráfica de la velocidad
load tiempow100.mat;
load w100.mat;
figure14 = figure('Color',[1 1 1]);
axes1 = axes('Parent',figure14,'FontSize',14,'FontName','Times New Roman');
box(axes1,'on');
hold(axes1,'all')
stairs(w100);
xlabel('Muestras','FontSize',14,'FontName','Times New Roman');
ylabel('Velocidad del motor  \omega  ...
    [rad/s]','FontSize',16,'FontName','Times New Roman');
title('Haz zoom y pulsa cualquier tecla para seleccionar el intervalo ...
    temporal');
grid;
% Se selecciona el intervalo para el cálculo
pause;
k = round(ginput(2));
% Se determina el intervalo temporal entre ambas velocidades y se calcula
% la pendiente mediante una regresión lineal
Y2 = w100(k(1):k(2));
X2 = tiempow100(k(1):k(2),1)-tiempow100(k(1),1);
A2 = [X2 ones(size(X2))];
th3 = A2\Y2;         % pendiente velocidad del motor
% la pendiente se puede calcular también como (Tr/Dm+w(0))/Jm*Dm =
% = (Tr+w(0)*Dm)/Jm. A partir de esta expresión se calcula Jm
Jm = (Tr+w100(k(1))*be)/abs(th3(1));
% Representación gráfica del resultado
figure15 = figure('Color',[1 1 1]);
axes1 = axes('Parent',figure15,'FontSize',14,'FontName','Times New Roman');
box(axes1,'on');
hold(axes1,'all')
plot(X2,Y2,'*',X2,A2*th3,'-','Parent', axes1);
grid;
ylabel('Velocidad del motor  \omega  ...
    [rad/s]','FontSize',16,'FontName','Times New Roman');
xlabel('tiempo  [s]','FontSize',14,'FontName','Times New Roman');
title(['J = ' num2str(abs(Jm)) ' Kg.m^2']);
```

Apéndice C

Diseño de los controladores en *Matlab*®

Durante el proceso de diseño de los controladores para el sistema NXT ballbot, se crearon diversos códigos en *Matlab*®, el primero de ellos consiste en un archivo que contiene todos los parámetros físicos del sistema y calcula las matrices de estado, esto será útil para no tener que ingresar todos estos datos en cada código de los controladores.

Código fuente C.1: Parametros_Matrices_NXT_Ballbot.m

```
%% Parámetros NXT Ballbot
% Este archivo contiene los parámetros físicos del sistema NXT Ballbot, así
% como genera las matrices de estado para el sistema nominal y politópico.

% Constante física
g = 9.81;               % aceleración de gravedad [m/seg^2]
% Parámetros sistema NXT Ballbot
Ms = 0.013;             % peso de la esfera [kg]
Rs = 0.026;             % radio de la esfera [m]
Js = 2 * Ms * Rs^2 / 3; % momento de inercia de la esfera [kgm^2]
Mw = 0.015;             % peso de la llanta [kg]
Rw = 0.021;             % radio de la llanta [m]
Lw = 0.022;             % ancho de la llanta [m]
Jw = Mw * Lw^2 / 2;     % momento de inercia de la llanta [kgm^2]
Mb = 0.682;             % peso del cuerpo(incluyendo el peso de la ...
    llanta) [kg]
L = 0.17;               % distancia entre el centro de masa de cuerpo y ...
    el centro de la esfera [m]
Jp = Mb * L^2 / 3;      % momento de inercia en la inclinación (pitch) ...
    del cuerpo [kgm^2]
% Parámetros motor DC reportados por Yamamoto
Jm = 1e-5;              % momento de inercia del motor de DC [kgm^2]
Ra = 6.69;              % resistencia interna del motor de DC [Ohms]
Kb = 0.468;             % constante de FEM en el motor de DC [Vsec/rad]
Kp = 0.317;             % constante de torque en el motor de DC [Nm/A]
be = 0.0022;            % coeficiente de fricción viscosa del motor de ...
    DC [Nms]
k = Rs / Rw;            % relación entre radios
```

```
%%Parámetros motor DC obtenidos experimentalmente
% Jm = 2.9639e-03;        % momento de inercia del motor de DC [kgm^2]
% Ra = 4.6;               % resistencia interna del motor de DC [Ohms]
% Kb = 0.49005;           % constante de FEM en el motor de DC [Vsec/rad]
% Kp = 0.49005;           % constante de torque en el motor de DC [Nm/A]
% be = 0.91439e-03;       % coeficiente de fricción viscosa del motor de ...
    DC [Nms]
% k = Rs / Rw;            % relación entre radios
% Relaciones  para las fuerzas generalizadas en términos de voltaje
a = Kp / Ra * 180 / pi;
b = k * (Kp * Kb / Ra + be);
%% Cálculo de las matrices de espacio de estados del sistema nominal

% Cálculo de los elementos de las matrices E, F, G, H
E = [ (Ms+Mb)*Rs^2+k^2*(Jm+Jw)+Js    Mb*Rs*L-k^2*(Jm+Jw)
       Mb*Rs*L-k^2*(Jm+Jw)          Mb*L^2+Jp+k^2*(Jm+Jw) ];
F = [ b  -b
     -b  +b];
G = [ 0         0
      0   -Mb*g*L ];
H = [ a
     -a ];
detE = det(E);
A32 = E(1,2)*G(2,2);
A33 = (E(1,2)*F(2,1)-E(2,2)*F(1,1));
A34 = (E(1,2)*F(2,2)-E(2,2)*F(1,2));
A42 = -E(1,1)*G(2,2);
A43 = (E(2,1)*F(1,1)-E(1,1)*F(2,1));
A44 = (E(2,1)*F(1,2)-E(1,1)*F(2,2));
B31 = E(2,2)*H(1,1)-E(1,2)*H(2,1);
B41 = E(1,1)*H(2,1)-E(2,1)*H(1,1);
% Cálculo de las matrices A, B, C, D, Cy
A = [ 0    0        1            0
      0    0        0            1
      0 A32/detE A33/detE A34/detE
      0 A42/detE A43/detE A44/detE ];
B =  [    0
          0
       B31/detE
       B41/detE ];
C = eye(4);
Cy = C(1,:);      % selección del estado x1 para seguimiento de trayectoria
D = zeros(4, 1);
% Vector de parámetros inciertos del motor
Jm = [4.5e-8  2.988e-3]; % momento de inercia del motor de DC [kgm^2]
be = [3.8745e-5  2.2e-3];% coeficiente de fricción viscosa del motor de ...
    DC [Nms]
%% Cálculo de las matrices de espacio de estados del sistema politópico

% Cálculo de los elementos de las matrices E, F, G, H
E1 = [ (Ms+Mb)*Rs^2+k^2*(Jm(1,1)+Jw)+Js     Mb*Rs*L-k^2*(Jm(1,1)+Jw)
       Mb*Rs*L-k^2*(Jm(1,1)+Jw)           Mb*L^2+Jp+k^2*(Jm(1,1)+Jw)]; %Emín
E2 = [ (Ms+Mb)*Rs^2+k^2*(Jm(1,2)+Jw)+Js     Mb*Rs*L-k^2*(Jm(1,2)+Jw)
```

```
76          Mb*Rs*L-k^2*(Jm(1,2)+Jw)           Mb*L^2+Jp+k^2*(Jm(1,2)+Jw)]; %Emáx
77  F1 = [ k*(Kp*Kb/Ra+be(1,1))  -k*(Kp*Kb/Ra+be(1,1))
78         -k*(Kp*Kb/Ra+be(1,1))   k*(Kp*Kb/Ra+be(1,1))];     %Fmín
79
80  F2 = [ k*(Kp*Kb/Ra+be(1,2))  -k*(Kp*Kb/Ra+be(1,2))
81         -k*(Kp*Kb/Ra+be(1,2))   k*(Kp*Kb/Ra+be(1,2))];     %Fmáx
82  G = [ 0         0
83         0   -Mb*g*L ];
84  H = [ a
85       -a ];
86  detE1 = det(E1);
87  detE2 = det(E2);
88   % Cálculo de las matrices A,B para el vértice del sistema con Emín-Fmín
89  A3211 = E1(1,2)*G(2,2);
90  A3311 = (E1(1,2)*F1(2,1)-E1(2,2)*F1(1,1));
91  A3411 = (E1(1,2)*F1(2,2)-E1(2,2)*F1(1,2));
92  A4211 = -E1(1,1)*G(2,2);
93  A4311 = (E1(2,1)*F1(1,1)-E1(1,1)*F1(2,1));
94  A4411 = (E1(2,1)*F1(1,2)-E1(1,1)*F1(2,2));
95  B3111 = E1(2,2)*H(1,1)-E1(1,2)*H(2,1);
96  B4111 = E1(1,1)*H(2,1)-E1(2,1)*H(1,1);
97  A1 = [ 0      0           1           0
98         0      0           0           1
99         0 A3211/detE1 A3311/detE1 A3411/detE1
100        0 A4211/detE1 A4311/detE1 A4411/detE1 ];
101 B1 = [      0
102             0
103        B3111/detE1
104        B4111/detE1 ];
105  % Cálculo de las matrices A,B para el vértice del sistema con Emín-Fmáx
106 A3212 = E1(1,2)*G(2,2);
107 A3312 = (E1(1,2)*F2(2,1)-E1(2,2)*F2(1,1));
108 A3412 = (E1(1,2)*F2(2,2)-E1(2,2)*F2(1,2));
109 A4212 = -E1(1,1)*G(2,2);
110 A4312 = (E1(2,1)*F2(1,1)-E1(1,1)*F2(2,1));
111 A4412 = (E1(2,1)*F2(1,2)-E1(1,1)*F2(2,2));
112 B3112 = E1(2,2)*H(1,1)-E1(1,2)*H(2,1);
113 B4112 = E1(1,1)*H(2,1)-E1(2,1)*H(1,1);
114 A2 = [ 0      0           1           0
115        0      0           0           1
116        0 A3212/detE1 A3312/detE1 A3412/detE1
117        0 A4212/detE1 A4312/detE1 A4412/detE1 ];
118 B2 = [      0
119             0
120        B3112/detE1
121        B4112/detE1 ];    % B2 = B1
122  % Cálculo de las matrices A,B para el vértice del sistema con Emáx-Fmín
123 A3221 = E2(1,2)*G(2,2);
124 A3321 = (E2(1,2)*F1(2,1)-E2(2,2)*F1(1,1));
125 A3421 = (E2(1,2)*F1(2,2)-E2(2,2)*F1(1,2));
126 A4221 = -E2(1,1)*G(2,2);
127 A4321 = (E2(2,1)*F1(1,1)-E2(1,1)*F1(2,1));
128 A4421 = (E2(2,1)*F1(1,2)-E2(1,1)*F1(2,2));
```

```
B3121 = E2(2,2)*H(1,1)-E2(1,2)*H(2,1);
B4121 = E2(1,1)*H(2,1)-E2(2,1)*H(1,1);
A3 = [ 0      0           1           0
       0      0           0           1
       0 A3221/detE2 A3321/detE2 A3421/detE2
       0 A4221/detE2 A4321/detE2 A4421/detE2 ];
B3 = [      0
            0
       B3121/detE2
       B4121/detE2 ];
% Cálculo de las matrices A,B para el vértice del sistema con Emáx-Fmáx
A3222 = E2(1,2)*G(2,2);
A3322 = (E2(1,2)*F2(2,1)-E2(2,2)*F2(1,1));
A3422 = (E2(1,2)*F2(2,2)-E2(2,2)*F2(1,2));
A4222 = -E2(1,1)*G(2,2);
A4322 = (E2(2,1)*F2(1,1)-E2(1,1)*F2(2,1));
A4422 = (E2(2,1)*F2(1,2)-E2(1,1)*F2(2,2));
B3122 = E2(2,2)*H(1,1)-E2(1,2)*H(2,1);
B4122 = E2(1,1)*H(2,1)-E2(2,1)*H(1,1);
A4 = [ 0      0           1           0
       0      0           0           1
       0 A3222/detE2 A3322/detE2 A3422/detE2
       0 A4222/detE2 A4322/detE2 A4422/detE2 ];
B4 = [      0
            0
       B3121/detE2
       B4121/detE2 ];    % B4 = B3
```

Luego de tener los parámetros físicos y matrices de estado del sistema, se genera un archivo que realiza el análisis en lazo abierto del sistema para determinar su estabilidad, controlabilidad y observabilidad.

Código fuente C.2: Analisis_Sistema_NXT_Ballbot.m

```
%% Análisis del sistema NXT Ballbot en lazo abierto
% Este archivo carga los datos del modelo NXT Ballbot para crear los
% modelos del sistema en espacio de estados, posteriormente se analizan en
% lazo abierto para determinar su estabilidad, controlabilidad y
% observabilidad.

% Obtención de datos del sistema NXT Ballbot
Parametros_Matrices_NXT_Ballbot;
%% Espacio de estados del sistema en su modelo nominal
estados = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
           'd\psi/dt  [\circ/s]'};
entrada = {'u'};
salidas = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
           'd\psi/dt  [\circ/s]'};
sys_ss_nom = ss(A,B,C,D,...
            'statename',estados,'inputname',entrada,'outputname',salidas);
% Análisis de estabilidad
```

```
18 polos_lazoabierto_nom = pole(sys_ss_nom)     % polos del sistema
19 figure;
20 pzmap(sys_ss_nom);                          % gráfica de los polos del sistema
21  % Análisis de controlabilidad
22 controlabilidad_nom = rank(ctrb(sys_ss_nom)) %rango de la matriz de ...
      controlabilidad
23  % Análisis de observabilidad
24 observabilidad_nom = rank(obsv(sys_ss_nom))  % rango de la matriz de ...
      observabilidad
25
26  %% Espacio de estados del sistema en su modelo politópico vértice 1
27 estados = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
28            'd\psi/dt  [\circ/s]'};
29 entrada = {'u'};
30 salidas = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
31            'd\psi/dt  [\circ/s]'};
32 sys_ss_vert1 = ss(A1,B1,C,D,...
33             'statename',estados,'inputname',entrada,'outputname',salidas);
34  % Análisis de estabilidad
35 polos_lazoabierto_vert1 = pole(sys_ss_vert1)     % polos del sistema
36 figure;
37 pzmap(sys_ss_vert1);                           % gráfica de los polos del ...
      sistema
38  % Análisis de controlabilidad
39 controlabilidad_vert1 = rank(ctrb(sys_ss_vert1)) %rango de la matriz de ...
      controlabilidad
40  % Análisis de observabilidad
41 observabilidad_vert1 = rank(obsv(sys_ss_vert1))  % rango de la matriz de ...
      observabilidad
42
43  %% Espacio de estados del sistema en su modelo politópico vértice 2
44 estados = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
45            'd\psi/dt  [\circ/s]'};
46 entrada = {'u'};
47 salidas = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
48            'd\psi/dt  [\circ/s]'};
49 sys_ss_vert2 = ss(A2,B2,C,D,...
50             'statename',estados,'inputname',entrada,'outputname',salidas);
51  % Análisis de estabilidad
52 polos_lazoabierto_vert2 = pole(sys_ss_vert2)     % polos del sistema
53 figure;
54 pzmap(sys_ss_vert2);                           % gráfica de los polos del ...
      sistema
55  % Análisis de controlabilidad
56 controlabilidad_vert2 = rank(ctrb(sys_ss_vert2)) %rango de la matriz de ...
      controlabilidad
57  % Análisis de observabilidad
58 observabilidad_vert2 = rank(obsv(sys_ss_vert2))  % rango de la matriz de ...
      observabilidad
59
60  %% Espacio de estados del sistema en su modelo politópico vértice 3
61 estados = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
62            'd\psi/dt  [\circ/s]'};
```

```
entrada = {'u'};
salidas = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
           'd\psi/dt  [\circ/s]'};
sys_ss_vert3 = ss(A3,B3,C,D,...
            'statename',estados,'inputname',entrada,'outputname',salidas);
% Análisis de estabilidad
polos_lazoabierto_vert3 = pole(sys_ss_vert3)    % polos del sistema
figure;
pzmap(sys_ss_vert3);                        % gráfica de los polos del ...
    sistema
% Análisis de controlabilidad
controlabilidad_vert3 = rank(ctrb(sys_ss_vert3)) %rango de la matriz de ...
    controlabilidad
% Análisis de observabilidad
observabilidad_vert3 = rank(obsv(sys_ss_vert3))  % rango de la matriz de ...
    observabilidad

%% Espacio de estados del sistema en su modelo politópico vértice 4
estados = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
           'd\psi/dt  [\circ/s]'};
entrada = {'u'};
salidas = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
           'd\psi/dt  [\circ/s]'};
sys_ss_vert4 = ss(A4,B4,C,D,...
            'statename',estados,'inputname',entrada,'outputname',salidas);
% Análisis de estabilidad
polos_lazoabierto_vert4 = pole(sys_ss_vert4)    % polos del sistema
figure;
pzmap(sys_ss_vert4);                        % gráfica de los polos del ...
    sistema
% Análisis de controlabilidad
controlabilidad_vert4 = rank(ctrb(sys_ss_vert4)) %rango de la matriz de ...
    controlabilidad
% Análisis de observabilidad
observabilidad_vert4 = rank(obsv(sys_ss_vert4))  % rango de la matriz de ...
    observabilidad
```

A continuación se enlistan los códigos de los controladores diseñados.

Código fuente C.3: LQR.m

```
%% Controlador LQR
% Este archivo carga los datos del sistema NXT Ballbot para diseñar un
% controlador bajo el esquema LQR, luego de obtener las ganancias del
% controlador se realiza un análisis del sistema en lazo cerrado y
% posteriormente se grafica la respuesta del sistema ante una entrada
% escalón unitario, así como su respuesta ante condiciones iniciales.

% Obtención de datos del sistema NXT Ballbot
Parametros_Matrices_NXT_Ballbot;
%% Diseño y obtención de las ganancias del controlador
%- Matrices aumentadas por la acción integral
A_hat = [ A   zeros(size(B))
```

```
          -Cy              0 ];
B_hat = [ B
          0 ];
% Matrices de ponderación Q y R
Q = [ 1   0   0   0   0
      0  6e5  0   0   0
      0   0   1   0   0
      0   0   0   1   0
      0   0   0   0  1e3 ];     % ponderación de los estados del sistema
R = 6e3*(180 / pi)^2;           % ponderación de la acción de control
% Cálculo de la ganancia del controlador LQR
K_lqr = lqr(A_hat, B_hat, Q, R)
%% Análisis del sistema NXT Ballbot en lazo cerrado
% Espacio de estados del sistema mas el término integral
estados_cl = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
           'd\psi/dt  [\circ/s]' 'error'};
entrada_cl = {'r'};
salidas_cl = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
           'd\psi/dt  [\circ/s]'};
sys_clt = ss([A-B*K_lqr(1:4) -B*K_lqr(5); -C(1,:) 0],[zeros(size(B));1],...
             [C zeros(size(B))],D,'statename',estados_cl,...
             'inputname',entrada_cl,'outputname',salidas_cl);
% Análisis de estabilidad
polos_lazocerrado = pole(sys_clt)     % polos del sistema
% Creación figura
figure1 = figure('Color',[1 1 1]);
% Creación ejes 1
axes1 = axes('Parent',figure1,'FontSize',14,'FontName','Times New Roman');
box(axes1,'on');
hold(axes1,'all')
pzmap(sys_clt);                        % gráfica de los polos del sistema
grid(axes1, 'on');
%% Simulación de la respuesta del sistema ante el controlador
t = 0:0.004:10;
% Respuesta del sistema ante una entrada escalón unitario
% Creación figura
figure2 = figure('Color',[1 1 1]);
% Creación ejes 1
axes1 = axes('Parent',figure2,'FontSize',14,'FontName','Times New Roman');
%xlim(axes1,[0 15]);
box(axes1,'on');
step(sys_clt,t);
% Condiciones iniciales del sistema x(0)=x0 con término integral
x0tr = [0 4 0 0 0]';
r = 0*ones(size(t));
[y,t,x] = lsim(sys_clt,r,t,x0tr);
% Respuesta del sistema ante condiciones iniciales
% Creación figura
figure3 = figure('Color',[1 1 1]);
% Creación ejes 1
axes1 = axes('Parent',figure3,...
    'Position',[0.13 0.583837209302326 0.334659090909091 ...
       0.341162790697674],...
```

```
65          'FontSize',14,...
66      'FontName','Times New Roman');
67  %xlim(axes1,[0 30]);
68  box(axes1,'on');
69  %grid(axes1,'on');
70  hold(axes1,'all');
71  % Plot figura
72  plot(t,x(:,1),'Parent',axes1);
73  ylabel({'\theta [\circ]'},'FontSize',16,'FontName','Times New Roman');
74  % Creación ejes 2
75  axes2 = axes('Parent',figure3,...
76      'Position',[0.570340909090909 0.583837209302326 0.334659090909091 ...
            0.341162790697674],...
77      'FontSize',14,...
78      'FontName','Times New Roman');
79  %xlim(axes2,[0 30]);
80  box(axes2,'on');
81  %grid(axes2,'on');
82  hold(axes2,'all');
83  % Plot figura
84  plot(t,x(:,2),'Parent',axes2);
85  ylabel({'\psi [\circ]'},'FontSize',16,'FontName','Times New Roman');
86  % Creación ejes 3
87  axes3 = axes('Parent',figure3,...
88      'Position',[0.13 0.11 0.334659090909091 0.341162790697674],...
89      'FontSize',14,...
90      'FontName','Times New Roman');
91  %xlim(axes3,[0 30]);
92  box(axes3,'on');
93  %grid(axes3,'on');
94  hold(axes3,'all');
95  % Plot figura
96  plot(t,x(:,3));
97  ylabel({'d\theta/dt  [\circ/s]'},'FontSize',16,'FontName','Times New ...
        Roman');
98  xlabel({'tiempo  [s]'},'FontSize',14,'FontName','Times New Roman');
99  % Creación ejes 4
100 axes4 = axes('Parent',figure3,...
101     'Position',[0.570340909090909 0.11 0.334659090909091 ...
            0.341162790697674],...
102     'FontSize',14,...
103     'FontName','Times New Roman');
104 %xlim(axes4,[0 30]);
105 box(axes4,'on');
106 %grid(axes4,'on');
107 hold(axes4,'all');
108 % Plot figura
109 plot(t,x(:,4),'Parent',axes4);
110 xlabel({'tiempo  [s]'},'FontSize',14,'FontName','Times New Roman');
111 ylabel({'d\psi/dt  [\circ/s]'},'FontSize',16,'FontName','Times New Roman');
112 %- Respuesta de la señal de control
113 % Creación figura
114 figure4 = figure('Color',[1 1 1]);
```

```
% Creación ejes 1
axes1 = axes('Parent',figure4,'FontSize',14,'FontName','Times New Roman');
%xlim(axes1,[0 30]);
box(axes1,'on');
hold(axes1,'all')
% Plot figura
plot(t,-K_lqr*x','Parent',axes1);
xlabel({'tiempo  [s]'},'FontSize',14,'FontName','Times New Roman');
ylabel({'Voltaje  [V]'},'FontSize',14,'FontName','Times New Roman');
```

Código fuente C.4: Polos.m

```
%% Controlador Asignación de Polos
% Este archivo carga los datos del sistema NXT Ballbot para diseñar un
% controlador bajo el esquema de asignación de Polos, luego de obtener las
% ganancias del controlador se realiza un análisis del sistema en lazo
% cerrado y posteriormente se grafica la respuesta del sistema ante una
% entrada escalón unitario, así como su respuesta ante condiciones ...
    iniciales.

% Obtención de datos del sistema NXT Ballbot
Parametros_Matrices_NXT_Ballbot;
%% Diseño y obtención de las ganancias del controlador
%- Cálculo de las ganancias del controlador mediante ubicación de polos
polos_deseados = [-53+2i -53-2i -1 -2];
K_polos = place(A,B,polos_deseados)
%% Análisis del sistema NXT Ballbot en lazo cerrado
%- Espacio de estados del sistema
estados_cl = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
              'd\psi/dt  [\circ/s]'};
entrada_cl = {'u'};
salidas_cl = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
              'd\psi/dt  [\circ/s]'};
sys_cl = ss(A-B*K_polos,B,C,D,'statename',estados_cl,...
           'inputname',entrada_cl,'outputname',salidas_cl);
%- Análisis de estabilidad
polos_lazocerrado = pole(sys_cl)    % polos del sistema
% Creación figura
figure1 = figure('Color',[1 1 1]);
% Creación ejes 1
axes1 = axes('Parent',figure1,'FontSize',14,'FontName','Times New Roman');
box(axes1,'on');
hold(axes1,'all')
pzmap(sys_cl);                      % gráfica de los polos del sistema
grid(axes1, 'on');
%% Simulación de la respuesta del sistema ante el controlador
t = 0:0.004:10;
%- Respuesta del sistema ante una entrada escalón unitario
% Creación figura
figure2 = figure('Color',[1 1 1]);
% Creación ejes 1
axes1 = axes('Parent',figure2,'FontSize',14,'FontName','Times New Roman');
```

```
%xlim(axes1,[0 15]);
box(axes1,'on');
step(sys_cl,t);
%- Condiciones iniciales del sistema x(0)=x0
x0 = [0 4 0 0]';
r = 0*ones(size(t));
[y,t,x] = lsim(sys_cl,r,t,x0);
%- Respuesta del sistema ante condiciones iniciales
% Creación figura
figure3 = figure('Color',[1 1 1]);
% Creación ejes 1
axes1 = axes('Parent',figure3,...
    'Position',[0.13 0.583837209302326 0.334659090909091 ...
       0.341162790697674],...
        'FontSize',14,...
    'FontName','Times New Roman');
%xlim(axes1,[0 30]);
box(axes1,'on');
%grid(axes1,'on');
hold(axes1,'all');
% Plot figura
plot(t,x(:,1),'Parent',axes1);
ylabel({'\theta [\circ]'},'FontSize',16,'FontName','Times New Roman');
% Creación ejes 2
axes2 = axes('Parent',figure3,...
    'Position',[0.570340909090909 0.583837209302326 0.334659090909091 ...
       0.341162790697674],...
    'FontSize',14,...
    'FontName','Times New Roman');
%xlim(axes2,[0 30]);
box(axes2,'on');
%grid(axes2,'on');
hold(axes2,'all');
% Plot figura
plot(t,x(:,2),'Parent',axes2);
ylabel({'\psi [\circ]'},'FontSize',16,'FontName','Times New Roman');
% Creación ejes 3
axes3 = axes('Parent',figure3,...
    'Position',[0.13 0.11 0.334659090909091 0.341162790697674],...
    'FontSize',14,...
    'FontName','Times New Roman');
%xlim(axes3,[0 30]);
box(axes3,'on');
%grid(axes3,'on');
hold(axes3,'all');
% Plot figura
plot(t,x(:,3));
ylabel({'d\theta/dt  [\circ/s]'},'FontSize',16,'FontName','Times New ...
   Roman');
xlabel({'tiempo  [s]'},'FontSize',14,'FontName','Times New Roman');
% Creación ejes 4
axes4 = axes('Parent',figure3,...
```

```
89      'Position',[0.570340909090909 0.11 0.334659090909091 ...
           0.341162790697674],...
90      'FontSize',14,...
91      'FontName','Times New Roman');
92  %xlim(axes4,[0 30]);
93  box(axes4,'on');
94  %grid(axes4,'on');
95  hold(axes4,'all');
96  % Plot figura
97  plot(t,x(:,4),'Parent',axes4);
98  xlabel({'tiempo  [s]'},'FontSize',14,'FontName','Times New Roman');
99  ylabel({'d\psi/dt  [\circ/s]'},'FontSize',16,'FontName','Times New Roman');
100 %- Respuesta de la señal de control
101 % Creación figura
102 figure4 = figure('Color',[1 1 1]);
103 % Creación ejes 1
104 axes1 = axes('Parent',figure4,'FontSize',14,'FontName','Times New Roman');
105 %xlim(axes1,[0 30]);
106 box(axes1,'on');
107 hold(axes1,'all')
108 % Plot figura
109 plot(t,-K_polos*x','Parent',axes1);
110 xlabel({'tiempo  [s]'},'FontSize',14,'FontName','Times New Roman');
111 ylabel({'Voltaje  [V]'},'FontSize',14,'FontName','Times New Roman');
```

Código fuente C.5: LQR_LMIs.m

```
1  %% Controlador LQR-LMIs
2  % Este archivo carga los datos del sistema NXT Ballbot para diseñar un
3  % controlador bajo el esquema LQR mediante LMIs para el modelo politópico
4  % del sistema, luego de obtener las ganancias del controlador se realiza
5  % un análisis del sistema en lazo cerrado y posteriormente se grafica la
6  % respuesta del sistema ante una entrada escalón unitario, así como su
7  % respuesta ante condiciones iniciales.
8
9  % Obtención de datos del sistema NXT Ballbot
10 Parametros_Matrices_NXT_Ballbot;
11 %% Diseño y obtención de las ganancias del controlador
12 %- Matrices aumentadas por la acción integral
13 A1_hat = [ A1   zeros(size(B1))
14           -Cy               0 ];
15 A2_hat = [ A2   zeros(size(B1))
16           -Cy               0 ];
17 A3_hat = [ A3   zeros(size(B3))
18           -Cy               0 ];
19 A4_hat = [ A4   zeros(size(B3))
20           -Cy               0 ];
21 B1_hat = [ B1
22             0 ];
23 B3_hat = [ B3
24             0 ];
25 %- Matrices de ponderación Q y R
```

```
Q = [ 1   0   0   0   0
      0  6e5  0   0   0
      0   0   1   0   0
      0   0   0   1   0
      0   0   0   0  1e3 ];     % ponderación de los estados del sistema
R = 6e3*(180 / pi)^2;           % ponderación de la acción de control
Rsq = R^(1/2);                  %
%- Cálculo de las ganacias del controlador LQR mediante restricciones LMI
% Inicialización de las LMIs
setlmis([]);
% Variables en las LMIs
P = lmivar(1,[5 1]);            % P matriz simétrica de 5x5
X = lmivar(2,[1 1]);            % X variable 1x1
Y = lmivar(2,[1 5]);            % Y vector de 1x5
mu = 7;                         % mu es la limitación en la acción de control
% LMI#1    Matriz simétrica definida positiva
% P > 0
lmiterm([-1 1 1 P],1,1);
% LMIs#2,3,4,5 para A1->A4
% Ai*P + P*Ai' - Bi*Y - Y'*Bi' + I < 0
lmiterm([2 1 1 P],A1_hat,1,'s');        % A1_hat*P + P*A1_hat'
lmiterm([2 1 1 Y],B1_hat,-1,'s');       % -B1_hat*Y - Y'*B1_hat
lmiterm([2 1 1 0],1);                   % I
lmiterm([3 1 1 P],A2_hat,1,'s');        % A2_hat*P + P*A2_hat'
lmiterm([3 1 1 Y],B1_hat,-1,'s');       % -B1_hat*Y - Y'*B1_hat
lmiterm([3 1 1 0],1);                   % I
lmiterm([4 1 1 P],A3_hat,1,'s');        % A3_hat*P + P*A3_hat'
lmiterm([4 1 1 Y],B3_hat,-1,'s');       % -B3_hat*Y - Y'*B3_hat
lmiterm([4 1 1 0],1);                   % I
lmiterm([5 1 1 P],A4_hat,1,'s');        % A4_hat*P + P*A4_hat'
lmiterm([5 1 1 Y],B3_hat,-1,'s');       % -B3_hat*Y - Y'*B3_hat
lmiterm([5 1 1 0],1);                   % I
% LMI#6 para la operación de Trace(.)
%  -                      -
% |   X          R^(1/2)*Y  |
% | Y'*R^(1/2)       P      | > 0      % Rsq = R^(1/2)
%  -                      -
lmiterm([-6 1 1 X],1,1);                % X
lmiterm([-6 2 1 -Y],1,Rsq);             % Y'*R^(1/2)
lmiterm([-6 2 2 P],1,1);                % P
% LMI#7 Restricciones asociadas a la entrada de control
% -                -
%|   1      x(0)'   |
%|                  | ≥ 0    // >
%|   x(0)      P    |
% -                -
x0tr = [0 4 0 0 0]';
lmiterm([-7 1 1 0],1);                  % 1
lmiterm([-7 2 1 0],x0tr);               % x(0)
lmiterm([-7 2 2 P],1,1);                % P
% LMI#8 Restricciones asociadas a la entrada de control
% -                  -
%|   P          Y'    |
```

```
%|                    | ≥ 0    // >
%|   Y       mu^2*I   |
% -                   -
lmiterm([-8 1 1 P],1,1);               % P
lmiterm([-8 2 1 Y],1,1);               % Y
lmiterm([-8 2 2 0],mu^2);              % I
% Descripción interna del sistema LMI
lmisys = getlmis;
% Definiendo el vector "c" a minimizar
n = decnbr(lmisys);          % toma el número de las variables de decisión
c = zeros(n,1);              % crea un vector de ceros de tamaño nx1
% Tr(Q*P) + Tr(X), 'trace' suma los elementos de la diagonal
% Obtención del vector c para resolver mincx
for i=1:n
[Pj, Xj, Yj] = defcx(lmisys, i, P, X, Y);
c(i) = trace(Q*Pj) + trace(Xj);
end
% Opciones del solver para la resolución de las LMIs
opciones = [1e-20, 100, -1, 5, 1];
% se minimiza el objetivo mediante las restricciones LMIs
[copt, xopt] = mincx(lmisys, c, opciones);
% Obtención de las variables de decisión en matrices y la ganancia K
Xopt = dec2mat(lmisys, xopt, X);
Yopt = dec2mat(lmisys, xopt, Y);
Popt = dec2mat(lmisys, xopt, P);
K_lqrlmi = Yopt/(Popt)             %K = Yopt*inv(Popt);
%% Análisis del sistema NXT Ballbot en lazo cerrado
%- Espacio de estados del sistema para A1_hat->A4_hat con término integral
estados_cl = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
              'd\psi/dt  [\circ/s]' 'error'};
entrada_cl = {'r'};
salidas_cl = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
              'd\psi/dt  [\circ/s]' };
syscl1tr = ss([A1-B1*K_lqrlmi(1:4) -B1*K_lqrlmi(5); -Cy 0],...
              [zeros(size(B1)); 1],[C zeros(size(B1))],D,'statename',...
               estados_cl,'inputname',entrada_cl,'outputname',salidas_cl);
syscl2tr = ss([A2-B1*K_lqrlmi(1:4) -B1*K_lqrlmi(5); -Cy 0],...
              [zeros(size(B1)); 1],[C zeros(size(B1))],D,'statename',...
               estados_cl,'inputname',entrada_cl,'outputname',salidas_cl);
syscl3tr = ss([A3-B3*K_lqrlmi(1:4) -B3*K_lqrlmi(5); -Cy 0],...
              [zeros(size(B3)); 1],[C zeros(size(B3))],D,'statename',...
               estados_cl,'inputname',entrada_cl,'outputname',salidas_cl);
syscl4tr = ss([A4-B3*K_lqrlmi(1:4) -B3*K_lqrlmi(5); -Cy 0],...
              [zeros(size(B3)); 1],[C zeros(size(B3))],D,'statename',...
               estados_cl,'inputname',entrada_cl,'outputname',salidas_cl);
% Análisis de estabilidad
polos_lazocerrado1 = pole(syscl1tr)
polos_lazocerrado2 = pole(syscl2tr)
polos_lazocerrado3 = pole(syscl3tr)
polos_lazocerrado4 = pole(syscl4tr)
% gráfica de los polos en lazo cerrado
figure1 = figure('Color',[1 1 1]);
pzmap(syscl1tr);
```

```
grid on;
hold on;
pzmap(syscl2tr);
pzmap(syscl3tr);
pzmap(syscl4tr);
hold off;
%% Simulación de la respuesta del sistema ante el controlador
t = 0:0.004:15;
%- Respuesta del sistema ante una entrada escalón unitario
% Creación figura
figure2 = figure('Color',[1 1 1]);
% Creación ejes 1
axes1 = axes('Parent',figure2,'FontSize',14,'FontName','Times New Roman');
%xlim(axes1,[0 15]);
box(axes1,'on');
step(syscl1tr,'r');
hold on
step(syscl2tr,'b--');
step(syscl3tr,'g:');
step(syscl4tr,'y-.');
%- Condiciones iniciales del sistema x(0)=x0 con término integral
x0tr = [0 4 0 0 0]';
r = 0*ones(size(t));
[y1tr,t,x1tr]=lsim(syscl1tr,r,t,x0tr);
[y2tr,t,x2tr]=lsim(syscl2tr,r,t,x0tr);
[y3tr,t,x3tr]=lsim(syscl3tr,r,t,x0tr);
[y4tr,t,x4tr]=lsim(syscl4tr,r,t,x0tr);
%- Respuesta del sistema ante condiciones iniciales
% Creación figura
figure3 = figure('Color',[1 1 1]);
% Creación ejes 1
axes1 = axes('Parent',figure3,...
    'Position',[0.13 0.583837209302326 0.334659090909091 ...
       0.341162790697674],...
        'FontSize',14,...
    'FontName','Times New Roman');
%xlim(axes1,[0 30]);
box(axes1,'on');
%grid(axes1,'on');
hold(axes1,'all');
% Plot figuras
plot(t,x1tr(:,1),'Parent',axes1,'Color',[1 0 0]);
plot(t,x2tr(:,1),'Parent',axes1,'LineStyle','--','Color',[0 0 1]);
plot(t,x3tr(:,1),'Parent',axes1,'LineStyle',':','Color',[0 1 0]);
plot(t,x4tr(:,1),'Parent',axes1,'LineStyle','-.','Color',[1 1 0]);
ylabel({'\theta  [\circ]'},'FontSize',16,'FontName','Times New Roman');
% Creación ejes 2
axes2 = axes('Parent',figure3,...
    'Position',[0.570340909090909 0.583837209302326 0.334659090909091 ...
       0.341162790697674],...
    'FontSize',14,...
    'FontName','Times New Roman');
%xlim(axes2,[0 30]);
```

```
box(axes2,'on');
 %grid(axes2,'on');
hold(axes2,'all');
 % Plot figuras
plot(t,x1tr(:,2),'Parent',axes2,'Color',[1 0 0]);
plot(t,x2tr(:,2),'Parent',axes2,'LineStyle','--','Color',[0 0 1]);
plot(t,x3tr(:,2),'Parent',axes2,'LineStyle',':','Color',[0 1 0]);
plot(t,x4tr(:,2),'Parent',axes2,'LineStyle','-.','Color',[1 1 0]);
ylabel({'\psi  [\circ]'},'FontSize',16,'FontName','Times New Roman');
 % Creación ejes 3
axes3 = axes('Parent',figure3,...
    'Position',[0.13 0.11 0.334659090909091 0.341162790697674],...
    'FontSize',14,...
    'FontName','Times New Roman');
 %xlim(axes3,[0 30]);
box(axes3,'on');
 %grid(axes3,'on');
hold(axes3,'all');
 % Plot figuras
plot(t,x1tr(:,3),'Parent',axes3,'Color',[1 0 0]);
plot(t,x2tr(:,3),'Parent',axes3,'LineStyle','--','Color',[0 0 1]);
plot(t,x3tr(:,3),'Parent',axes3,'LineStyle',':','Color',[0 1 0]);
plot(t,x4tr(:,3),'Parent',axes3,'LineStyle','-.','Color',[1 1 0]);
ylabel({'d\theta/dt  [\circ/s]'},'FontSize',16,'FontName','Times New ...
    Roman');
xlabel({'tiempo  [s]'},'FontSize',14,'FontName','Times New Roman');
 % Creación ejes 4
axes4 = axes('Parent',figure3,...
    'Position',[0.570340909090909 0.11 0.334659090909091 ...
       0.341162790697674],...
    'FontSize',14,...
    'FontName','Times New Roman');
 %xlim(axes4,[0 30]);
box(axes4,'on');
 %grid(axes4,'on');
hold(axes4,'all');
 % Plot figuras
plot(t,x1tr(:,4),'Parent',axes4,'Color',[1 0 0]);
plot(t,x2tr(:,4),'Parent',axes4,'LineStyle','--','Color',[0 0 1]);
plot(t,x3tr(:,4),'Parent',axes4,'LineStyle',':','Color',[0 1 0]);
plot(t,x4tr(:,4),'Parent',axes4,'LineStyle','-.','Color',[1 1 0]);
ylabel({'d\psi/dt  [\circ/s]'},'FontSize',16,'FontName','Times New Roman');
xlabel({'tiempo [s]'},'FontSize',14,'FontName','Times New Roman');
 %- Respuesta de la señal de control
 % Creación figura
figure4 = figure('Color',[1 1 1]);
 % Creación ejes 1
axes1 = axes('Parent',figure4,'FontSize',14,'FontName','Times New Roman');
 %xlim(axes1,[0 30]);
box(axes1,'on');
hold(axes1,'all')
 % Plot figuras
plot(t,-K_lqrlmi*x1tr','Parent',axes1,'Color',[1 0 0]);
```

```
plot(t,-K_lqrlmi*x2tr','Parent',axes1,'LineStyle','--','Color',[0 0 1]);
plot(t,-K_lqrlmi*x3tr','Parent',axes1,'LineStyle',':','Color',[0 1 0]);
plot(t,-K_lqrlmi*x4tr','Parent',axes1,'LineStyle','-.','Color',[1 1 0]);
xlabel({'tiempo  [s]'},'FontSize',14,'FontName','Times New Roman');
ylabel({'Voltaje  [V]'},'FontSize',14,'FontName','Times New Roman');
```

Código fuente C.6: Polos_LMIs.m

```
%% Controlador Ubicación de Polos en Regiones LMI
% Este archivo carga los datos del sistema NXT Ballbot para diseñar un
% controlador bajo el esquema de ubicación de polos en regiones LMI para el
% modelo politópico del sistema, luego de obtener las ganancias del
% controlador se realiza un análisis del sistema en lazo cerrado y
% posteriormente se grafica la respuesta del sistema ante una entrada
% escalón unitario, así como su respuesta ante condiciones iniciales.

% Obtención de datos del sistema NXT Ballbot
Parametros_Matrices_NXT_Ballbot;
%% Diseño y obtención de las ganancias del controlador
%- Matrices aumentadas por la acción integral
A1_hat = [ A1   zeros(size(B1))
          -Cy               0 ];
A2_hat = [ A2   zeros(size(B1))
          -Cy               0 ];
A3_hat = [ A3   zeros(size(B3))
          -Cy               0 ];
A4_hat = [ A4   zeros(size(B3))
          -Cy               0 ];
B1_hat = [ B1
           0 ];
B3_hat = [ B3
           0 ];
%- Cálculo de las ganacias del controlador mediante restricciones LMI
% Definición de la región LMI W(alfa,r,theta)
disp('Defina la Región LMI W(alfa,r,theta) y la restricción de control mu')
alfa = input ('Introduzca el valor de alfa, ');
r = input ('Introduzca el valor de r, ');
theta = input('Introduzca el valor de theta en grados, ')*(pi/180);
mu = input('Introduzca el valor de mu, ');
% Inicialización de las LMIs
setlmis([]);
% Variables en las LMIs
W = lmivar(1,[5 1]);    % W matriz simétrica de 5x5
Y = lmivar(2,[1 5]);    % Y vector 1x5
X = lmivar(2,[1 1]);    % X variable 1x1
% LMI#1 Matriz simétrica definida positiva W
% W > 0
lmiterm([-1 1 1 W],1,1);
% LMIs#2,3,4,5 restricciones asociadas al parámetro alfa de A1_hat->A4_hat
% A_i*W + W*A_i' - B_i*Y - Y'*B_i' + 2*alfa*W < 0
lmiterm([2 1 1 W],A1_hat,1,'s');        % A1_hat*W + W*A1_hat'
lmiterm([2 1 1 Y],B1_hat,-1,'s');       % -B1_hat*Y - Y'*B1_hat'
```

```
45 lmiterm([2 1 1 W],2*alfa,1);                 % 2*alfa*W
46 lmiterm([3 1 1 W],A2_hat,1,'s');             % A2_hat*W + W*A2_hat'
47 lmiterm([3 1 1 Y],B1_hat,-1,'s');            % -B1_hat*Y - Y'*B1_hat'
48 lmiterm([3 1 1 W],2*alfa,1);                 % 2*alfa*W
49 lmiterm([4 1 1 W],A3_hat,1,'s');             % A3_hat*W + W*A3_hat'
50 lmiterm([4 1 1 Y],B3_hat,-1,'s');            % -B3_hat*Y - Y'*B3_hat'
51 lmiterm([4 1 1 W],2*alfa,1);                 % 2*alfa*W
52 lmiterm([5 1 1 W],A4_hat,1,'s');             % A4_hat*W + W*A4_hat'
53 lmiterm([5 1 1 Y],B3_hat,-1,'s');            % -B3_hat*Y - Y'*B3_hat'
54 lmiterm([5 1 1 W],2*alfa,1);                 % 2*alfa*W
55 % LMIs#6,7,8,9 Restricciones asociadas a r para A1_hat->A4_hat
56 % -                                    -
57 %|   -r*W           A_i*W - B_i*Y      |
58 %|                                      | < 0
59 %| W*A_i' - Y'*B_i'        -r*W        |
60 % -                                    -
61 lmiterm([6 1 1 W],-r,1);                     % -r*W
62 lmiterm([6 1 2 W],A1_hat,1);                 % A1_hat*W
63 lmiterm([6 1 2 Y],B1_hat,-1);                % -B1_hat*Y
64 lmiterm([6 2 2 W],-r,1);                     % -r*W
65 lmiterm([7 1 1 W],-r,1);                     % -r*W
66 lmiterm([7 1 2 W],A2_hat,1);                 % A2_hat*W
67 lmiterm([7 1 2 Y],B1_hat,-1);                % -B1_hat*Y
68 lmiterm([7 2 2 W],-r,1);                     % -r*W
69 lmiterm([8 1 1 W],-r,1);                     % -r*W
70 lmiterm([8 1 2 W],A3_hat,1);                 % A3_hat*W
71 lmiterm([8 1 2 Y],B3_hat,-1);                % -B3_hat*Y
72 lmiterm([8 2 2 W],-r,1);                     % -r*W
73 lmiterm([9 1 1 W],-r,1);                     % -r*W
74 lmiterm([9 1 2 W],A4_hat,1);                 % A4_hat*W
75 lmiterm([9 1 2 Y],B3_hat,-1);                % -B3_hat*Y
76 lmiterm([9 2 2 W],-r,1);                     % -r*W
77 % LMIs#10,11,12,13 Restricciones asociadas a theta para A1_hat->A4_hat
78 % - ...

      -
79 %|  sin(theta)*(A_i*W+W*A_i'-B_i*Y-Y'*B_i')   ...
      cos(theta)*(A_i*W-B_i*Y-W*A_i'+Y'*B_i')   |
80 %| ...

      | < 0
81 %|  cos(theta)*(W*A_i'-Y'*B_i'-A_i*W+B_i*Y)   ...
      sin(theta)*(A_i*W+W*A_i'-B_i*Y-Y'*B_i')   |
82 % - ...

      -
83 lmiterm([10 1 1 W],A1_hat,sin(theta),'s');    % sin(theta)*(A1_hat*W + ...
      W*A1_hat')
84 lmiterm([10 1 1 Y],B1_hat,-sin(theta),'s');   % sin(theta)*(-B1_hat*Y - ...
      Y'*B1_hat')
85 lmiterm([10 2 1 W],(A1_hat),-cos(theta));     % cos(theta)*(-A1_hat)*W
86 lmiterm([10 2 1 W],cos(theta),(A1_hat)');     % W*cos(theta)*(A1_hat)'
87 lmiterm([10 2 1 Y],(B1_hat),cos(theta));      % cos(theta)*(B1_hat)*Y
```

```
88  lmiterm([10 2 1 -Y],-cos(theta),B1_hat');      % -Y'*cos(theta)*B1_hat'
89  lmiterm([10 2 2 W],A1_hat,sin(theta),'s');     % sin(theta)*(A1_hat*W + ...
        W*A1_hat')
90  lmiterm([10 2 2 Y],B1_hat,-sin(theta),'s');    % sin(theta)*(-B1_hat*Y - ...
        Y'*B1_hat')
91  lmiterm([11 1 1 W],A2_hat,sin(theta),'s');     % ...
        sin(theta)*(A2_hat*W+W*A2_hat')
92  lmiterm([11 1 1 Y],B1_hat,-sin(theta),'s');    % sin(theta)*(-B1_hat*Y - ...
        Y'*B1_hat')
93  lmiterm([11 2 1 W],(A2_hat),-cos(theta));      % cos(theta)*(-A2_hat)*W
94  lmiterm([11 2 1 W],cos(theta),(A2_hat)');      % W*cos(theta)*(A2_hat)'
95  lmiterm([11 2 1 Y],(B1_hat),cos(theta));       % cos(theta)*(B1_hat)*Y
96  lmiterm([11 2 1 -Y],-cos(theta),B1_hat');      % -Y'*cos(theta)*B1_hat'
97  lmiterm([11 2 2 W],A2_hat,sin(theta),'s');     % sin(theta)*(A2_hat*W + ...
        W*A2_hat')
98  lmiterm([11 2 2 Y],B1_hat,-sin(theta),'s');    % sin(theta)*(-B1_hat*Y - ...
        Y'*B1_hat')
99  lmiterm([12 1 1 W],A3_hat,sin(theta),'s');     % ...
        sin(theta)*(A3_hat*W+W*A3_hat')
100 lmiterm([12 1 1 Y],B3_hat,-sin(theta),'s');    % sin(theta)*(-B3_hat*Y - ...
        Y'*B3_hat')
101 lmiterm([12 2 1 W],(A3_hat),-cos(theta));      % cos(theta)*(-A3_hat)*W
102 lmiterm([12 2 1 W],-cos(theta),(A3_hat)');     % W*cos(theta)*(A3_hat)'
103 lmiterm([12 2 1 Y],(B3_hat),cos(theta));       % cos(theta)*(B3_hat)*Y
104 lmiterm([12 2 1 -Y],-cos(theta),B3_hat');      % -Y'*cos(theta)*B3_hat'
105 lmiterm([12 2 2 W],A3_hat,sin(theta),'s');     % sin(theta)*(A3_hat*W + ...
        W*A3_hat')
106 lmiterm([12 2 2 Y],B3_hat,-sin(theta),'s');    % sin(theta)*(-B3_hat*Y - ...
        Y'*B3_hat')
107 lmiterm([13 1 1 W],A4_hat,sin(theta),'s');     % ...
        sin(theta)*(A4_hat*W+W*A4_hat')
108 lmiterm([13 1 1 Y],B3_hat,-sin(theta),'s');    % sin(theta)*(-B3_hat*Y - ...
        Y'*B3_hat')
109 lmiterm([13 2 1 W],(A4_hat),-cos(theta));      % cos(theta)*(-A4_hat)*W
110 lmiterm([13 2 1 W],cos(theta),(A4_hat)');      % W*cos(theta)*(A4_hat)'
111 lmiterm([13 2 1 Y],(B3_hat),cos(theta));       % cos(theta)*(B3_hat)*Y
112 lmiterm([13 2 1 -Y],-cos(theta),B3_hat');      % -Y'*cos(theta)*B3_hat'
113 lmiterm([13 2 2 W],A4_hat,sin(theta),'s');     % sin(theta)*(A3_hat*W + ...
        W*A3_hat')
114 lmiterm([13 2 2 Y],B3_hat,-sin(theta),'s');    % sin(theta)*(-B3_hat*Y - ...
        Y'*B3_hat')
115 % LMI#14 Restricciones asociadas a la entrada de control
116 % -                 -
117 %|   1      x(0)'   |
118 %|                  | ≥ 0    // >
119 %|   x(0)      W    |
120 % -                 -
121 x0tr = [0 4 0 0 0 ]';
122 lmiterm([-14 1 1 0],1);          % 1
123 lmiterm([-14 2 1 0],x0tr);       % x(0)
124 lmiterm([-14 2 2 W],1,1);        % W
125 % LMI#15 Restricciones asociadas a la entrada de control
126 % -                   -
```

```
%|   W          Y'    |
%|                    | ≥ 0    // >
%|   Y       mu^2*I   |
% -                   -
lmiterm([-15 1 1 W],1,1);      % W
lmiterm([-15 2 1 Y],1,1);      % Y
lmiterm([-15 2 2 0],mu^2);     % I
% Descripción interna del sistema LMI
lmisys = getlmis;
% Obtención de las variables de decisión en matrices y la ganancia K
[tmin,xfeas] = feasp(lmisys);
W = dec2mat(lmisys,xfeas,W);
Y = dec2mat(lmisys,xfeas,Y);
K_poloslmi = Y/W
%% Análisis del sistema NXT Ballbot en lazo cerrado
%- Espacio de estados del sistema para A1_hat->A4_hat con término integral
estados_cl = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
              'd\psi/dt  [\circ/s]' 'error'};
entrada_cl = {'r'};
salidas_cl = {'\theta  [\circ]' '\psi  [\circ]' 'd\theta/dt  [\circ/s]'...
              'd\psi/dt  [\circ/s]' };
syscl1tr = ss([A1-B1*K_poloslmi(1:4) -B1*K_poloslmi(5); -Cy 0],...
              [zeros(size(B1)); 1],[C zeros(size(B1))],D,'statename',...
               estados_cl,'inputname',entrada_cl,'outputname',salidas_cl);
syscl2tr = ss([A2-B1*K_poloslmi(1:4) -B1*K_poloslmi(5); -Cy 0],...
              [zeros(size(B1)); 1],[C zeros(size(B1))],D,'statename',...
               estados_cl,'inputname',entrada_cl,'outputname',salidas_cl);
syscl3tr = ss([A3-B3*K_poloslmi(1:4) -B3*K_poloslmi(5); -Cy 0],...
              [zeros(size(B3)); 1],[C zeros(size(B3))],D,'statename',...
               estados_cl,'inputname',entrada_cl,'outputname',salidas_cl);
syscl4tr = ss([A4-B3*K_poloslmi(1:4) -B3*K_poloslmi(5); -Cy 0],...
              [zeros(size(B3)); 1],[C zeros(size(B3))],D,'statename',...
               estados_cl,'inputname',entrada_cl,'outputname',salidas_cl);
% Análisis de estabilidad
polos_lazocerrado1 = pole(syscl1tr)
polos_lazocerrado2 = pole(syscl2tr)
polos_lazocerrado3 = pole(syscl3tr)
polos_lazocerrado4 = pole(syscl4tr)
% gráfica de los polos en lazo cerrado
figure1 = figure('Color',[1 1 1]);
pzmap(syscl1tr);
grid on;
hold on;
pzmap(syscl2tr);
pzmap(syscl3tr);
pzmap(syscl4tr);
hold off;
%% Simulación de la respuesta del sistema ante el controlador
t = 0:0.004:30;
%- Respuesta del sistema ante una entrada escalón unitario
% Creación figura
figure2 = figure('Color',[1 1 1]);
% Creación ejes 1
```

```
axes1 = axes('Parent',figure2,'FontSize',14,'FontName','Times New Roman');
 %xlim(axes1,[0 15]);
box(axes1,'on');
step(syscl1tr,'r');
hold on
step(syscl2tr,'b--');
step(syscl3tr,'g:');
step(syscl4tr,'y-.');
 %- Condiciones iniciales del sistema x(0)=x0 con término integral
x0tr = [0 4 0 0 0]';
r = 0*ones(size(t));
[y1tr,t,x1tr]=lsim(syscl1tr,r,t,x0tr);
[y2tr,t,x2tr]=lsim(syscl2tr,r,t,x0tr);
[y3tr,t,x3tr]=lsim(syscl3tr,r,t,x0tr);
[y4tr,t,x4tr]=lsim(syscl4tr,r,t,x0tr);
 %- Respuesta del sistema ante condiciones iniciales
 % Creación figura
figure3 = figure('Color',[1 1 1]);
 % Creación ejes 1
axes1 = axes('Parent',figure3,...
    'Position',[0.13 0.583837209302326 0.334659090909091 ...
       0.341162790697674],...
        'FontSize',14,...
    'FontName','Times New Roman');
 %xlim(axes1,[0 30]);
box(axes1,'on');
 %grid(axes1,'on');
hold(axes1,'all');
 % Plot figuras
plot(t,x1tr(:,1),'Parent',axes1,'Color',[1 0 0]);
plot(t,x2tr(:,1),'Parent',axes1,'LineStyle','--','Color',[0 0 1]);
plot(t,x3tr(:,1),'Parent',axes1,'LineStyle',':','Color',[0 1 0]);
plot(t,x4tr(:,1),'Parent',axes1,'LineStyle','-.','Color',[1 1 0]);
ylabel({'\theta  [\circ]'},'FontSize',16,'FontName','Times New Roman');
 % Creación ejes 2
axes2 = axes('Parent',figure3,...
    'Position',[0.570340909090909 0.583837209302326 0.334659090909091 ...
       0.341162790697674],...
    'FontSize',14,...
    'FontName','Times New Roman');
 %xlim(axes2,[0 30]);
box(axes2,'on');
 %grid(axes2,'on');
hold(axes2,'all');
 % Plot figuras
plot(t,x1tr(:,2),'Parent',axes2,'Color',[1 0 0]);
plot(t,x2tr(:,2),'Parent',axes2,'LineStyle','--','Color',[0 0 1]);
plot(t,x3tr(:,2),'Parent',axes2,'LineStyle',':','Color',[0 1 0]);
plot(t,x4tr(:,2),'Parent',axes2,'LineStyle','-.','Color',[1 1 0]);
ylabel({'\psi  [\circ]'},'FontSize',16,'FontName','Times New Roman');
 % Creación ejes 3
axes3 = axes('Parent',figure3,...
    'Position',[0.13 0.11 0.334659090909091 0.341162790697674],...
```

```
231     'FontSize',14,...
232     'FontName','Times New Roman');
233  %xlim(axes3,[0 30]);
234  box(axes3,'on');
235  %grid(axes3,'on');
236  hold(axes3,'all');
237  % Plot figuras
238  plot(t,x1tr(:,3),'Parent',axes3,'Color',[1 0 0]);
239  plot(t,x2tr(:,3),'Parent',axes3,'LineStyle','--','Color',[0 0 1]);
240  plot(t,x3tr(:,3),'Parent',axes3,'LineStyle',':','Color',[0 1 0]);
241  plot(t,x4tr(:,3),'Parent',axes3,'LineStyle','-.','Color',[1 1 0]);
242  ylabel({'d\theta/dt  [\circ/s]'},'FontSize',16,'FontName','Times New ...
        Roman');
243  xlabel({'tiempo  [s]'},'FontSize',14,'FontName','Times New Roman');
244  % Creación ejes 4
245  axes4 = axes('Parent',figure3,...
246     'Position',[0.570340909090909 0.11 0.334659090909091 ...
           0.341162790697674],...
247     'FontSize',14,...
248     'FontName','Times New Roman');
249  %xlim(axes4,[0 30]);
250  box(axes4,'on');
251  %grid(axes4,'on');
252  hold(axes4,'all');
253  % Plot figuras
254  plot(t,x1tr(:,4),'Parent',axes4,'Color',[1 0 0]);
255  plot(t,x2tr(:,4),'Parent',axes4,'LineStyle','--','Color',[0 0 1]);
256  plot(t,x3tr(:,4),'Parent',axes4,'LineStyle',':','Color',[0 1 0]);
257  plot(t,x4tr(:,4),'Parent',axes4,'LineStyle','-.','Color',[1 1 0]);
258  ylabel({'d\psi/dt  [\circ/s]'},'FontSize',16,'FontName','Times New Roman');
259  xlabel({'tiempo [s]'},'FontSize',14,'FontName','Times New Roman');
260  %- Respuesta de la señal de control
261  % Creación figura
262  figure4 = figure('Color',[1 1 1]);
263  % Creación ejes 1
264  axes1 = axes('Parent',figure4,'FontSize',14,'FontName','Times New Roman');
265  %xlim(axes1,[0 30]);
266  box(axes1,'on');
267  hold(axes1,'all')
268  % Plot figuras
269  plot(t,-K_poloslmi*x1tr','Parent',axes1,'Color',[1 0 0]);
270  plot(t,-K_poloslmi*x2tr','Parent',axes1,'LineStyle','--','Color',[0 0 1]);
271  plot(t,-K_poloslmi*x3tr','Parent',axes1,'LineStyle',':','Color',[0 1 0]);
272  plot(t,-K_poloslmi*x4tr','Parent',axes1,'LineStyle','-.','Color',[1 1 0]);
273  xlabel({'tiempo  [s]'},'FontSize',14,'FontName','Times New Roman');
274  ylabel({'Voltaje  [V]'},'FontSize',14,'FontName','Times New Roman');
```

Apéndice D

Modelo del sistema en *Simulink*®

El modelo del controlador del sistema NXT en *Simulink*® consta de 4 bloques principales: el controlador del sistema, el modelo de la planta del sistema para la simulación, el hardware del sistema para su implementación física, así como la adquisición de las señales en tiempo real.

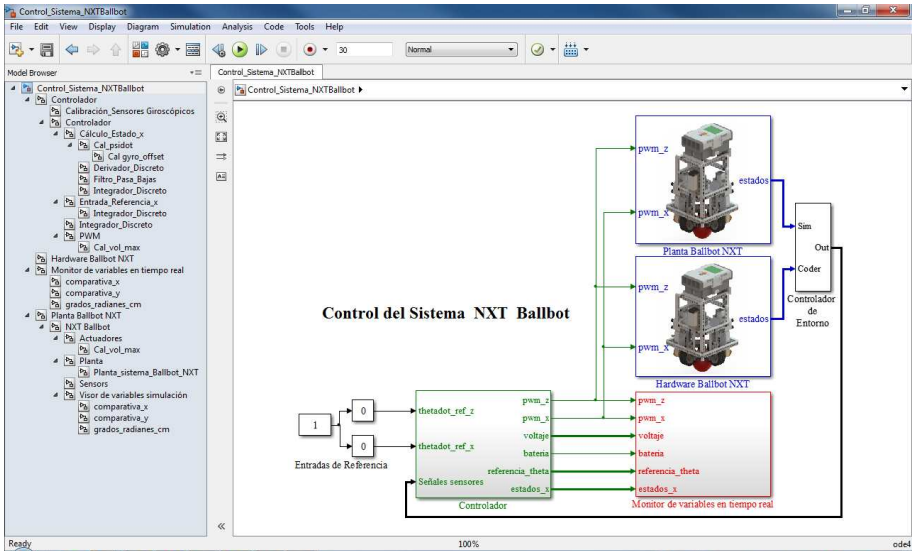


Figura D.1: Estructura del modelo en *Simulink*®

Bloque del controlador del sistema

En esta sección se incluyen los subbloques que constituyen al bloque principal del controlador del sistema NXT ballbot.

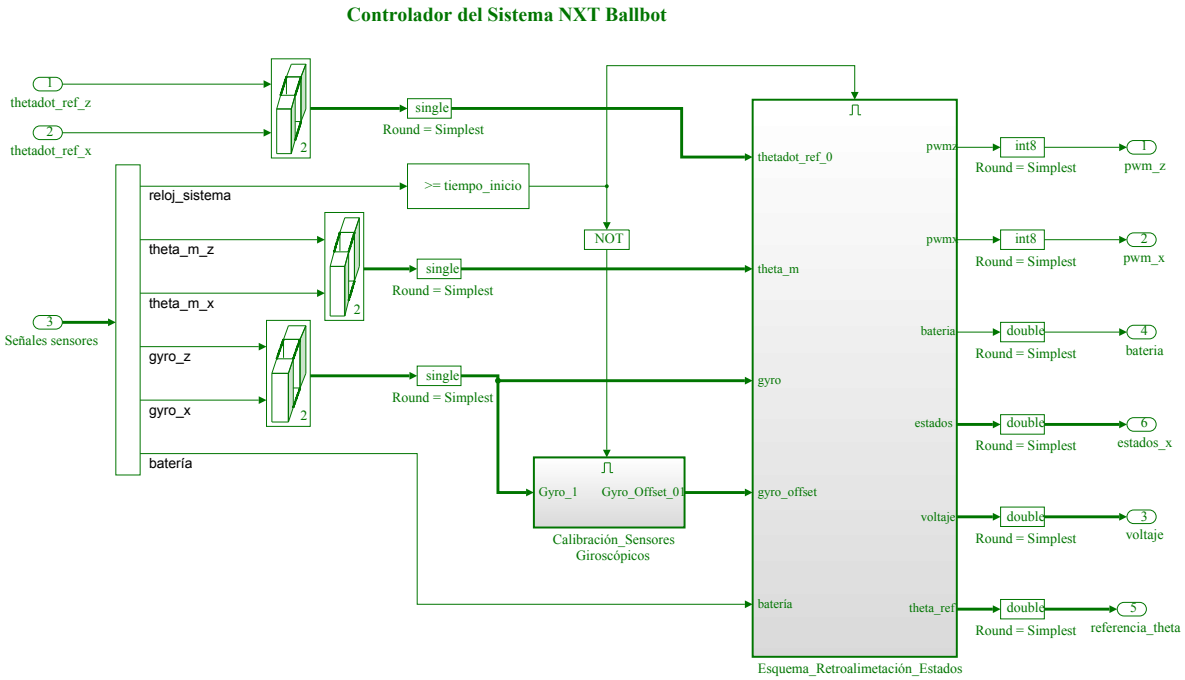


Figura D.2: Contenido del bloque principal del controlador

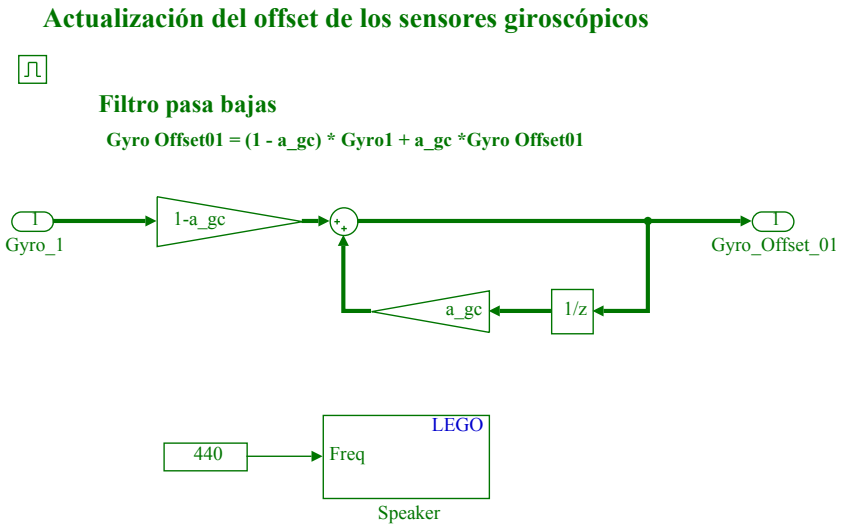


Figura D.3: Bloque de calibración de los sensores giroscópicos.

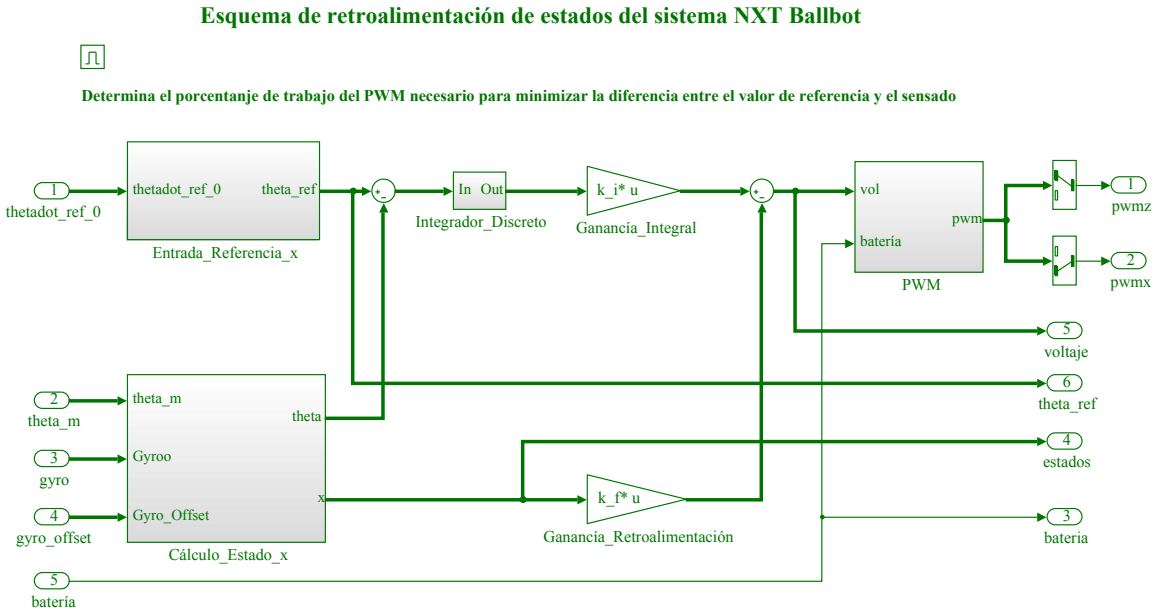


Figura D.4: Bloque de retroalimentación de estados.

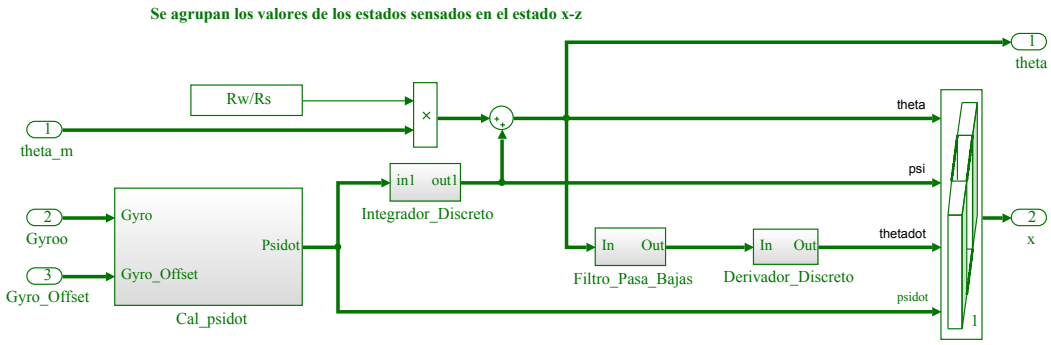


Figura D.5: Cálculo de los vectores de estado.

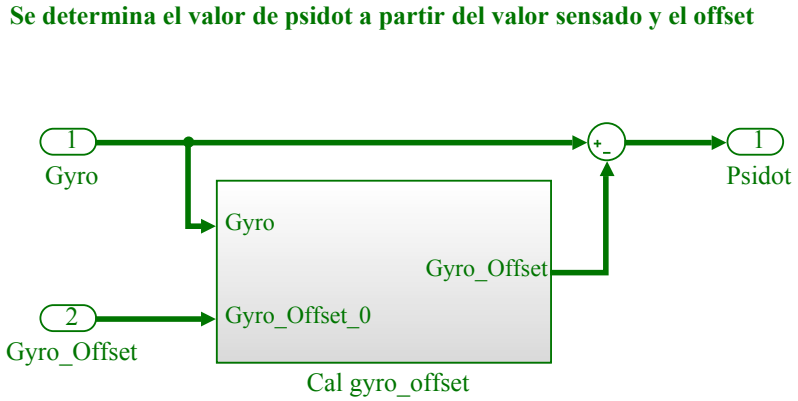


Figura D.6: Cálculo del valor de $\dot{\psi}$.

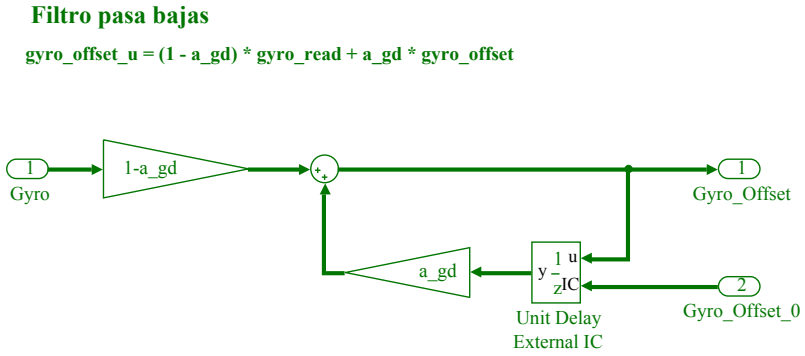


Figura D.7: Filtro pasa bajas para la actualización del offset del sensor.

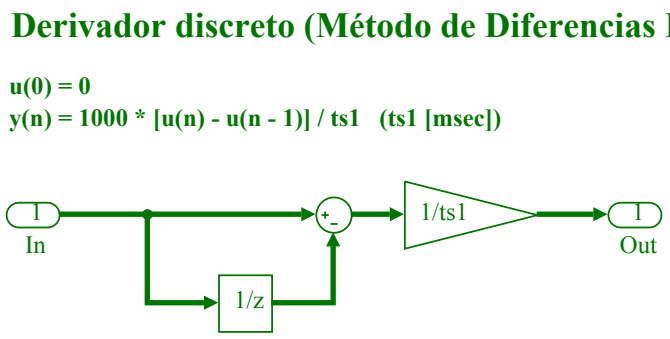


Figura D.8: Derivador discreto.

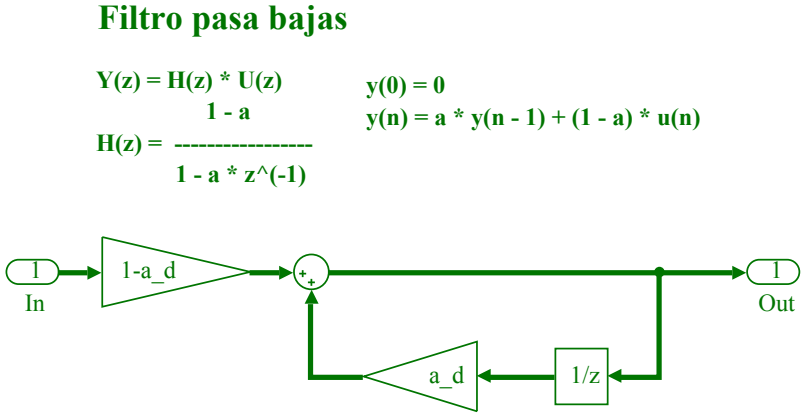


Figura D.9: Filtro pasa bajas.

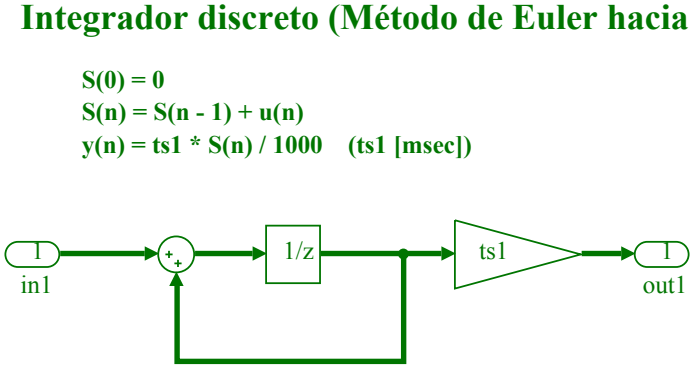


Figura D.10: Integrador discreto.

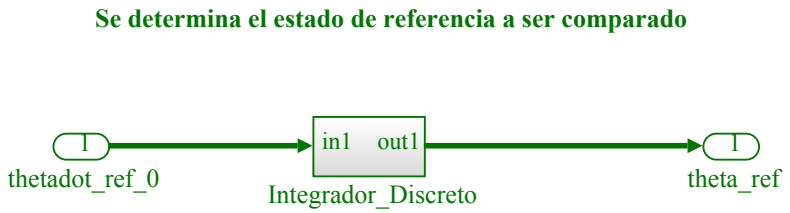


Figura D.11: Cálculo de la entrada de referencia.

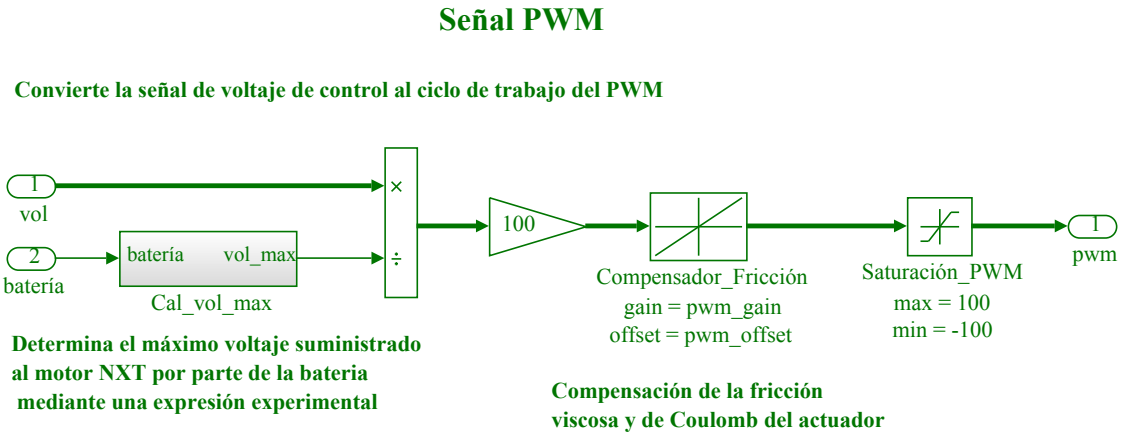


Figura D.12: Bloque de conversión de voltaje a PWM.

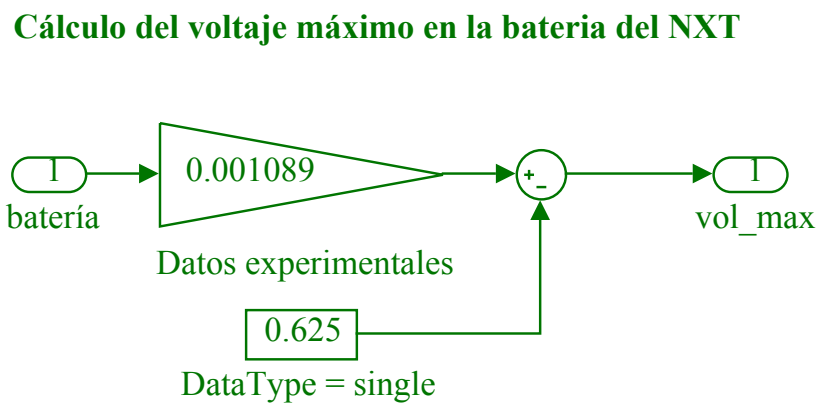


Figura D.13: Cálculo del voltaje máximo en la bateria del ladrillo NXT.

Bloque del hardware del sistema

En esta sección se incluyen los subbloques que constituyen al bloque principal del hardware del sistema NXT ballbot.

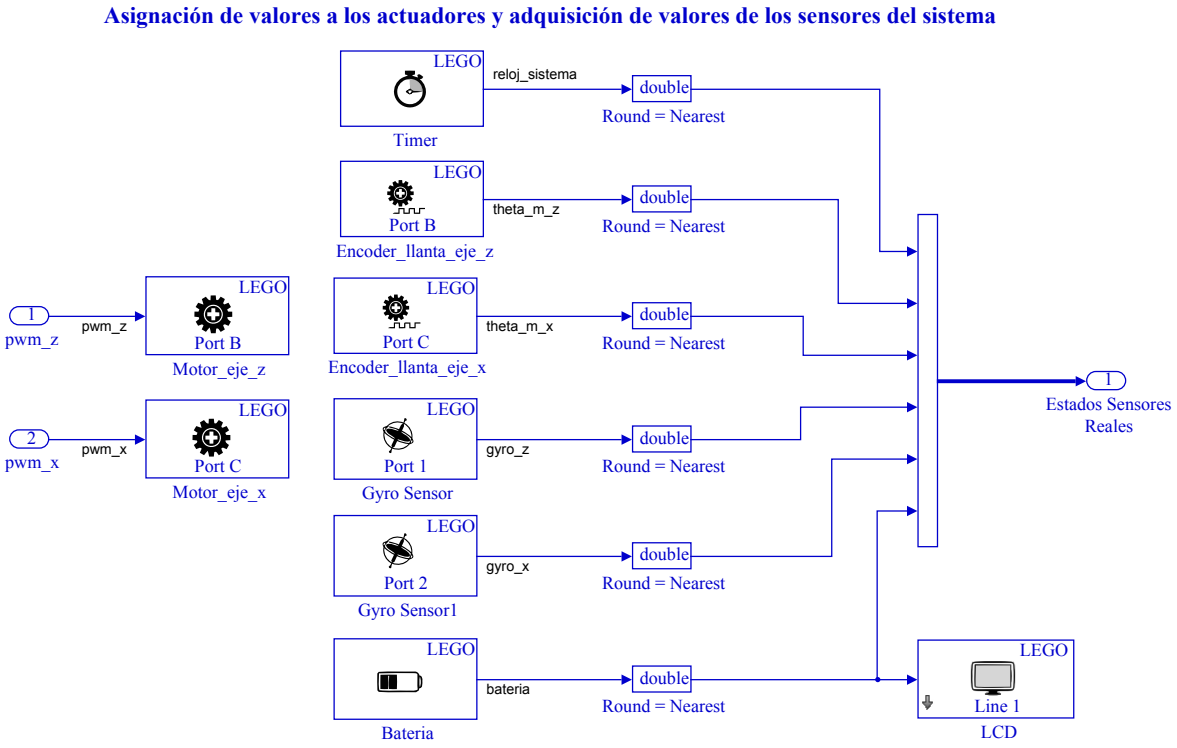


Figura D.14: Bloque de interacción con el hardware del sistema.

Bloque de la adquisición de señales del sistema

En esta sección se incluyen los subbloques que constituyen al bloque principal encargado de la adquisición de señales del hardware del sistema NXT ballbot.

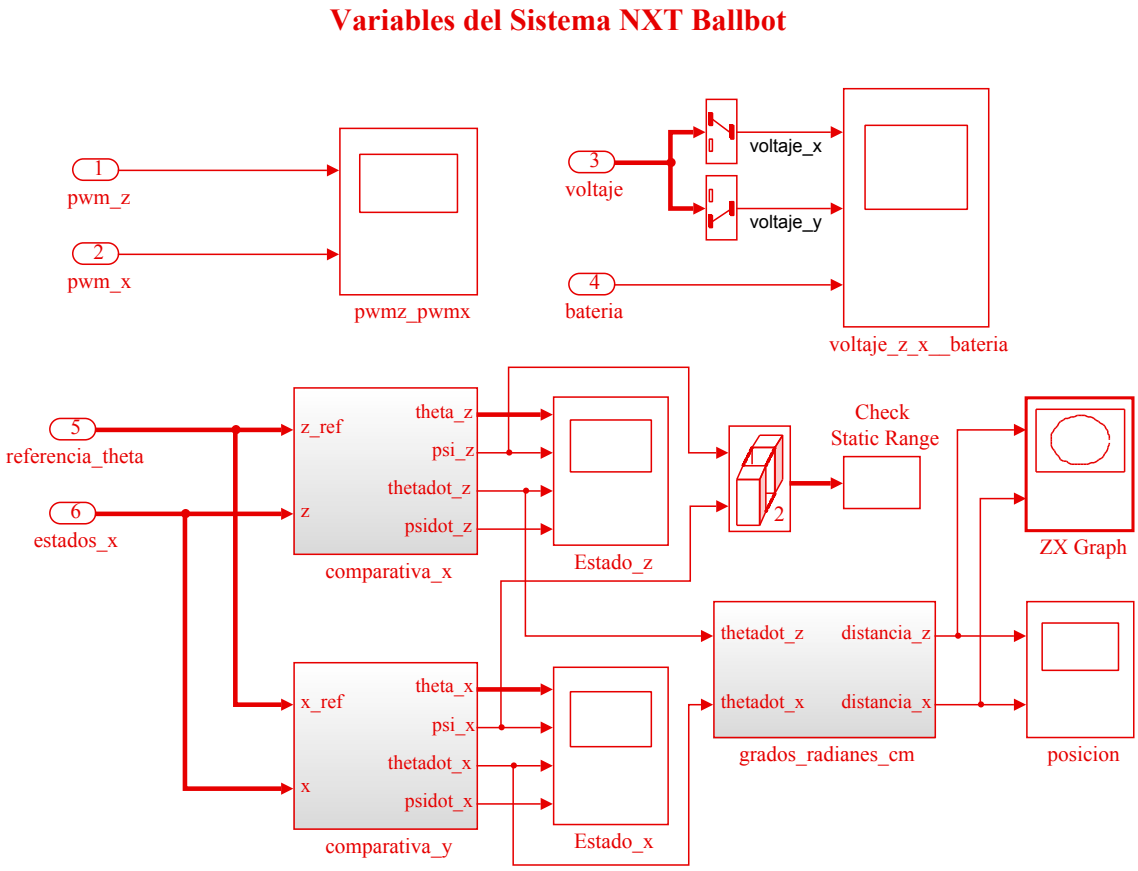


Figura D.15: Bloque para monitorear las señales del sistema.

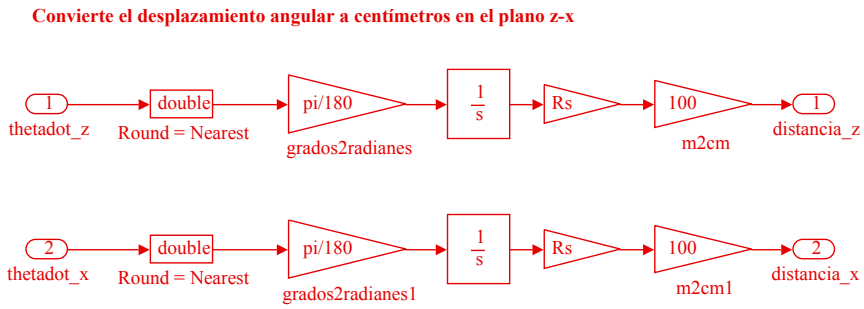


Figura D.16: Bloque de conversión del desplazamiento angular de la pelota a desplazamiento lineal.

Bloque del modelo dinámico del sistema

En esta sección se incluyen los subbloques que constituyen al bloque principal que modela la dinámica del sistema NXT ballbot.

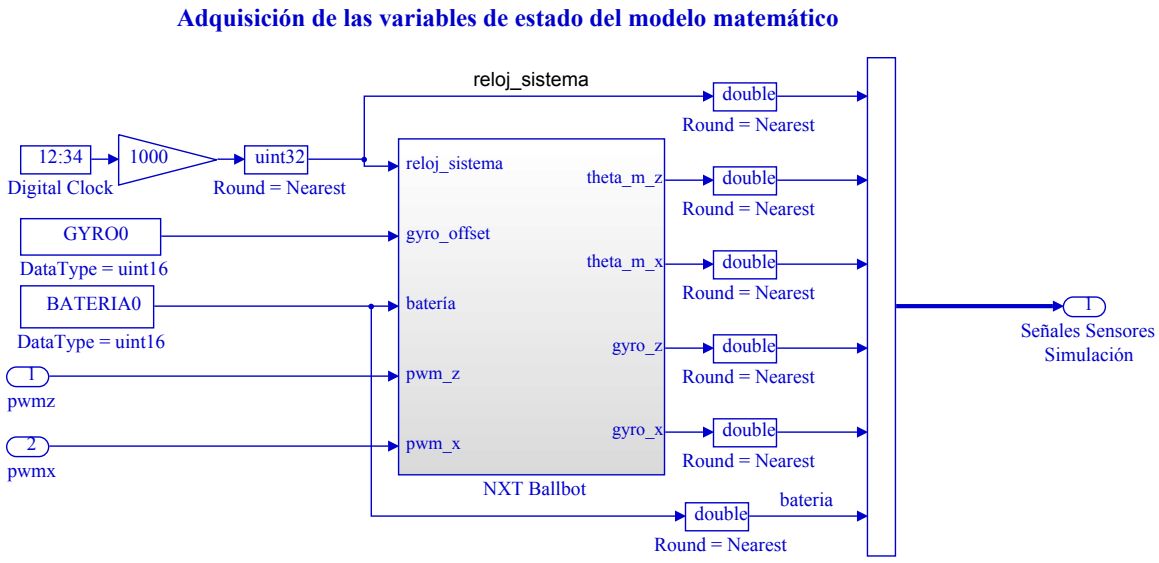


Figura D.17: Bloque de interacción con el modelo dinámico del sistema.

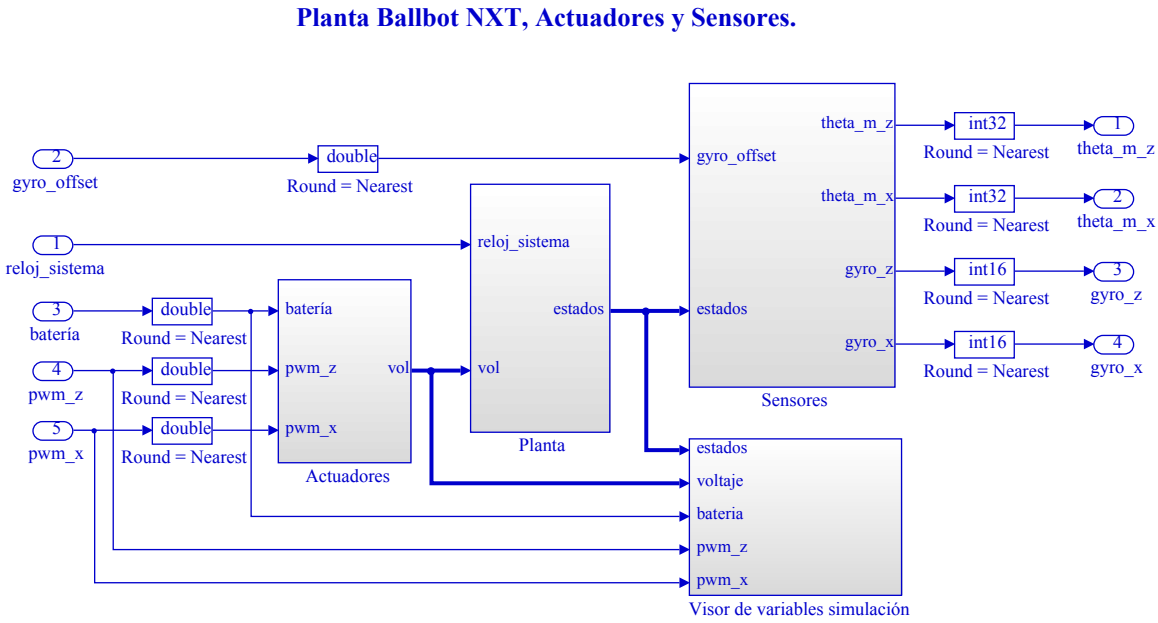


Figura D.18: Bloque que simula al hardware del sistema NXT ballbot.

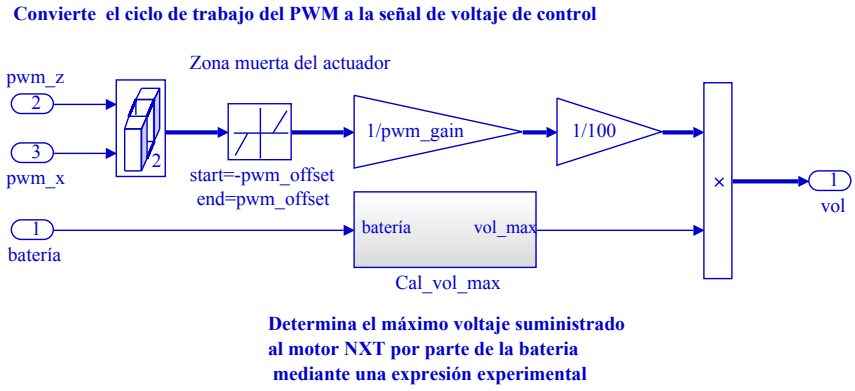


Figura D.19: Bloque de conversión de la señal PWM a voltaje.

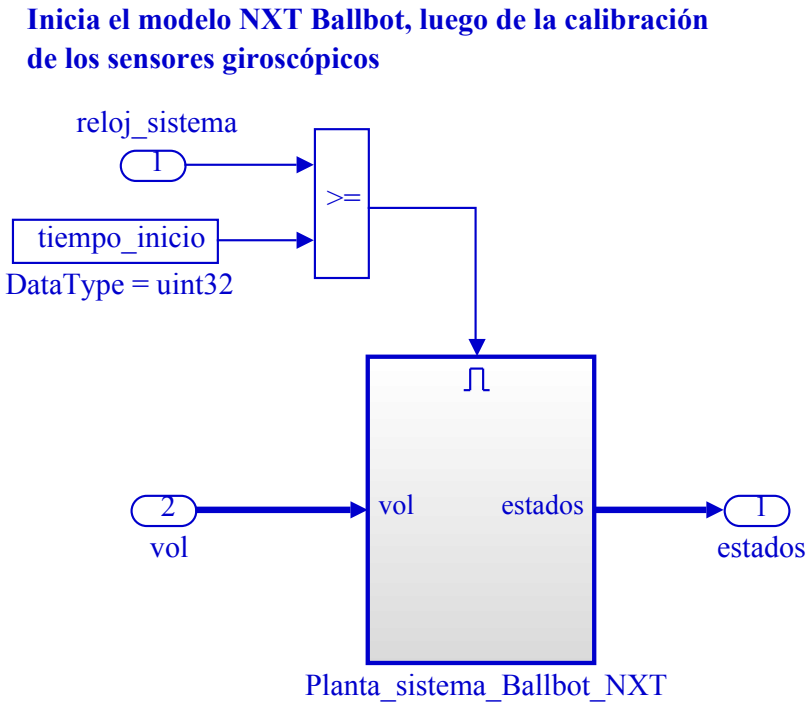


Figura D.20: Bloque de inicialización del modelo en espacio de estados del sistema.

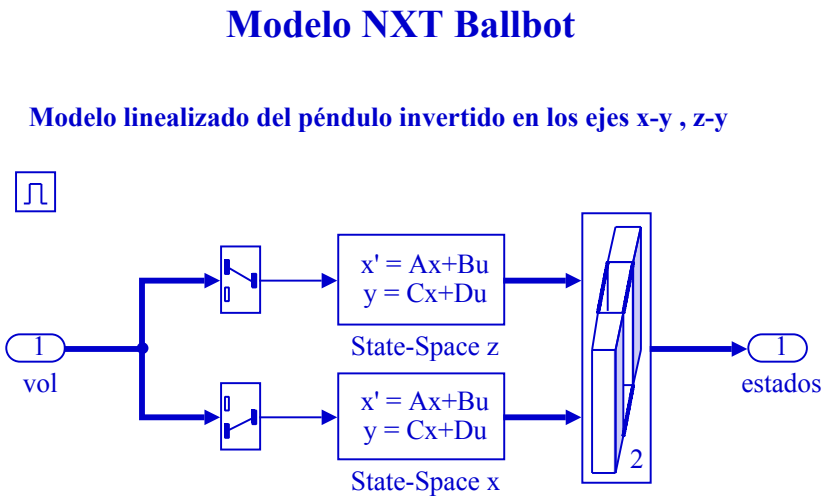


Figura D.21: Cálculo de los estados del sistema.

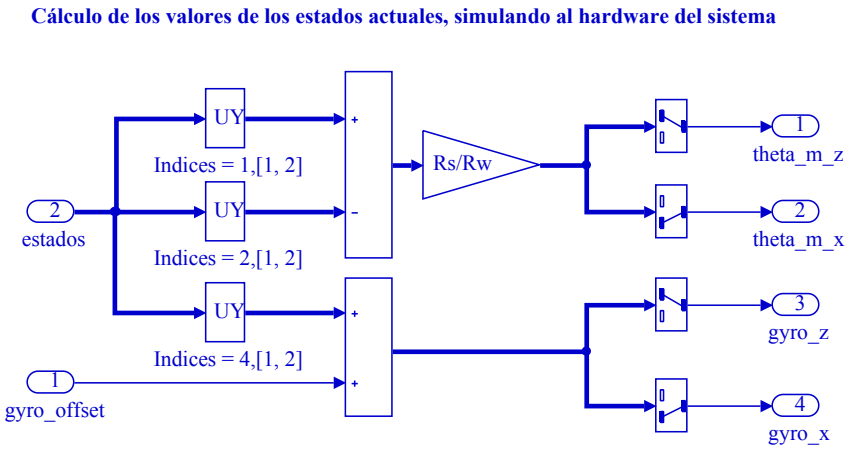


Figura D.22: Bloque para adecuar los estados actuales a los obtenidos por los sensores del sistema.

Apéndice E

Proceso de ejecución del modelo implementado en *Simulink®*

Una vez que se obtienen todas las ganancias de los controladores diseñados y el modelo del controlador para el NXT ballbot en *Simulink®*, se realiza un código en *Matlab®* que servirá para almacenar toda la información requerida por el modelo implementado en *Simulink®*, tales como: los parámetros físicos del sistema, las matrices de estado, las ganancias de los controladores, las ganancias de los filtros, los tiempos de muestreo y otros parámetros de la simulación.

Código fuente E.1: Controlador.m

```
%% Controladores para simulación e implementación
% Este archivo contiene las ganancias de los controladores para ser
% cargados en el archivo de Simulink y proceder a la simulación o ejecución
% del sistema en tiempo real.

% Obtención de datos del sistema NXT Ballbot
clear all;
Parametros_Matrices_NXT_Ballbot;
%% Ganancia de los Controladores

%- Controlador LQR
% Modelo 1
K_lqr = [-0.0150   -1.5698   -0.0270   -0.2325    0.0071];
% Modelo 2
K_lqr_v2 = [-0.0153   -2.0748   -0.0273   -0.3228    0.0071];
%- Controlador Asignación de Polos
% Modelo 1
K_polos = [ -0.0075   -1.4776   -0.0226   -0.1324];
% Modelo 2
K_polos_v2 = [ -0.1236  -23.7402   -0.2009   -2.3652];
%- Controlador LQR mediante LMIs
% Modelo 1, restricción control: mu=7
K_lqrlmi = [-0.0072   -1.3851   -0.0181   -0.2298    0.0025];
```

```
% Modelo 2, restricción control: mu=7
K_lqrlmi_v2 = [-0.0074   -0.9940   -0.0167   -0.1682    0.0023];
%- Controlador Ubicación de Polos en Regiones LMI
% Modelo 1 , parámetros: alfa= 0, r=2000, theta=85, mu=7
K_poloslmi = [-0.0026   -1.5075   -0.0175   -0.2326    0.0005];
% Modelo 2 , parámetros: alfa= 0, r=2000, theta=85, mu=7
K_poloslmi_v2 = [-0.0056   -1.2246   -0.0187   -0.1874    0.0019];

%% Matrices estados: nominal y vértices
% Se seleccionan los valores de las matrices A y B a emplear en el bloque
% de espacio de estados de Simulink

%- nominal
A = A;
B = B;
% %- vértice 1
% A = A1;
% B = B1;
% %- vértice 2
% A = A2;
% B = B1;
% %- vértice 3
% A = A3;
% B = B3;
% %- vértice 4
% A = A4;
% B = B3;

%% Parámetros para la simulación del sistema en Simulink
% Se seleccionan las ganancias a emplear en el controlador, así como se
% establecen las condiciones iniciales, ganancias de los filtros y tiempos
% de muestreo necesarios para el funcionamiento del archivo de Simulink.

%- Controlador LQR
k_f = K_lqr(1:4);        % ganancia retroalimentación
k_i = -K_lqr(5);         % ganancia integral
% k_f = K_lqr_v2(1:4);     % ganancia retroalimentación
% k_i = -K_lqr_v2(5);      % ganancia integral
%- Controlador por Asignación de Polos
% k_f = K_polos;           % ganancia retroalimentación
% k_i = 0;                 % ganancia integral
% k_f = K_polos_v2;        % ganancia retroalimentación
% k_i = 0;                 % ganancia integral
%- Controlador LQR mediante LMIs
% k_f = K_lqrlmi(1:4);     % ganancia retroalimentación
% k_i = -K_lqrlmi(5);      % ganancia integral
% k_f = K_lqrlmi_v2(1:4);  % ganancia retroalimentación
% k_i = -K_lqrlmi_v2(5);   % ganancia integral
%- Controlador por Asignación de Polos mediante LMIs
% k_f = K_poloslmi(1:4);   % ganancia retroalimentación
% k_i = -K_poloslmi(5);    % ganancia integral
% k_f = K_poloslmi_v2(1:4);% ganancia retroalimentación
% k_i = -K_poloslmi_v2(5);  % ganancia integral
```

```
77
78 % Tasas de muestreo de las tareas
79 ts1 = 0.004;              % ts1 tiempo de muestreo [sec]
80 % Variables de tiempo
81 tiempo_inicio = 2000;
82 % parámetros de fricción de Coulomb y fricción viscosa
83 pwm_gain = 1;        % ganancia pwm
84 pwm_offset = 0;      % offset pwm
85 % Coeficientes de los filtros pasa bajas
86 a_b = 0.8;           % valor promedio de la bateria
87 a_d = 0.8;           % supresor del ruido en la velocidad
88 a_r = 0.996;         % señal de referencia suave
89 a_gc = 0.8;          % offset calibración del sensor de giro
90 a_gd = 0.999;        % compensación de la deriva del sensor de giro
91 % Valores configurados por el usuario
92 k_thetadot = 0.12/Rs*180/pi;     % ganancia de la velocidad (0.12 [m/seg])
93 psi_dif_umb = 45;    % umbral del ángulo de inclinación del cuerpo [grados]
94 theta_dif_umb = 4*360; % umbral del ángulo de rodadura de la esfera [grados]
95 %simulacion
96 GYRO0 = 600;         % valor inicial del sensor giroscópico
97 BATERIA0 = 8200;     % valor inicial de la bateria [mV]
98 % valores iniciales
99 PSI0 = 2;            % Valor inicial del ángulo pitch del cuerpo [grados]
100 XIV = [             % Estado inicial
101     0
102     PSI0
103     0
104     0
105     ];
106 % Tasas de muestreo
107 TS = 0.004;
108 ts = 0.001;
109 START_POS = [0, 0];
110 % Condiciones para la detención de la simulación
111 BODY_ANGLE_MAX = 35;     % ángulo máximo del cuerpo
112 BODY_ANGLE_MIN = -35;    % ángulo mínimo del cuerpo
```

En el modelo en *Simulink®* hacemos clic derecho y seleccionamos *Model Properties*, de la ventana emergente nos dirigimos a *Callbacks* y luego seleccionamos *InitFcn* donde escribimos el nombre del archivo creado en *Matlab®* que servirá de enlace, en este caso *Controlador* como se muestra en la Figura E.1.

Para realizar la simulación del modelo solo basta con dirigirnos a la pestaña de *Simulation*, luego *Mode* y seleccionar la opción *Normal*, establecer el tiempo de simulación y teclear la combinación de teclas *Ctrl+T*. Para generar el código ejecutable dentro del ladrillo NXT e iniciar la comunicación en tiempo real entre el sistema NXT ballbot y *Simulink®* nos dirigimos nuevamente a la pestaña de *Simulation*, luego *Mode* y seleccionar la opción *External*, Luego nos dirigimos a la pestaña *Tools*, luego *Run on Target Hardware* y seleccionamos *Prepare to Run* (Figura E.2(a)) para que se abra la ventana de configuración de parámetros, aquí se establece el tipo de conexión para la descarga del código ejecutable que puede ser mediante

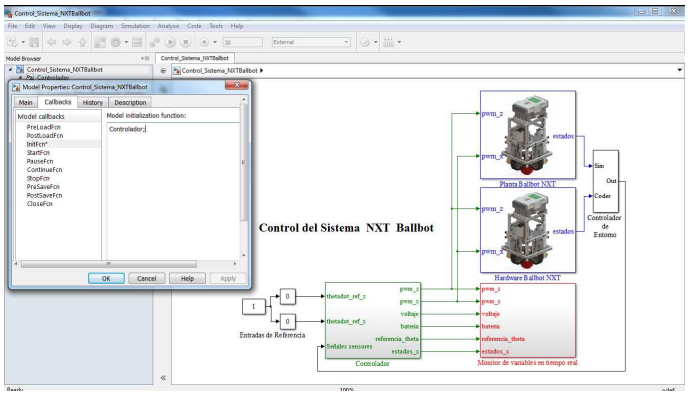
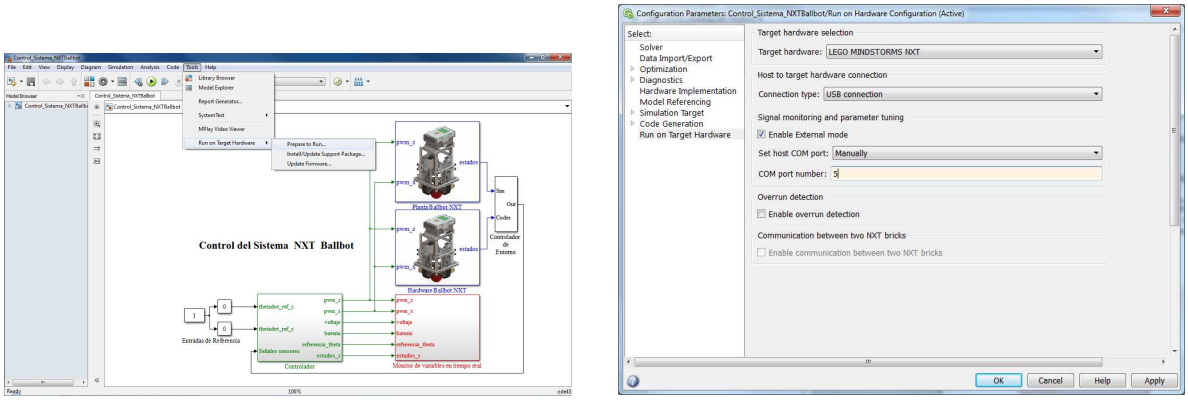


Figura E.1: Enlace entre el código de *Matlab*® y el modelo de *Simulink*®.



(a) (b)

Figura E.2: (a) Configuración para la implementación física del controlador en el sistema NXT ballbot; (b) activación del monitoreo de las señales en tiempo real.

conexión *usb* o *bluetooth*, y y para habilitar el monitoreo de las señales activamos la casilla *Enable External Mode* como se muestra en la Figura E.2(b). Luego dentro de la misma ventana emergente nos dirigimos a *Solver* donde establecemos el tiempo de simulación, como opciones del *Solver* establecemos *Fixed-step*, *ode4(Runge-Kutta)* y un tamaño de paso de 0.001. Por último aceptamos las configuraciones haciendo clic en *OK* y nuevamente nos dirigimos a la pestaña *Tools*, luego *Run on Target Hardware* y esta vez aparecerá la opción *Run* como se muestra en la Figura E.3.

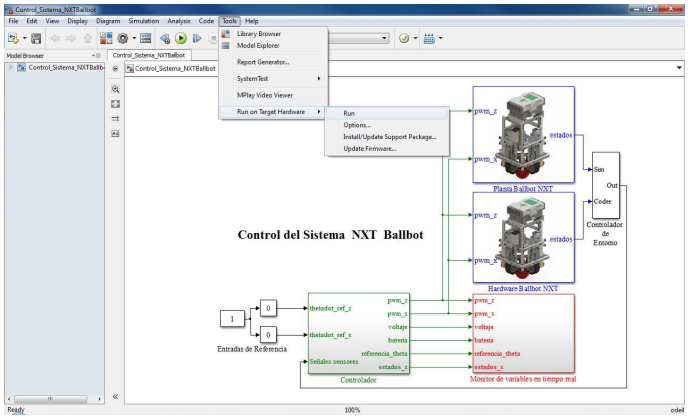


Figura E.3: Ejecución del controlador en el sistema NXT ballbot.

Apéndice F

Pruebas experimentales con perturbaciones

En este apéndice se muestran los resultados experimentales de 2 pruebas que se realizaron al sistema NXT ballbot empleando el controlador mediante ubicación de polos en regiones LMI y en las cuales se aplicaron pequeñas perturbaciones a la estructura del sistema NXT ballbot durante su operación. Esto con el fin de mostrar la robustez del controlador implementando la técnica LMI.

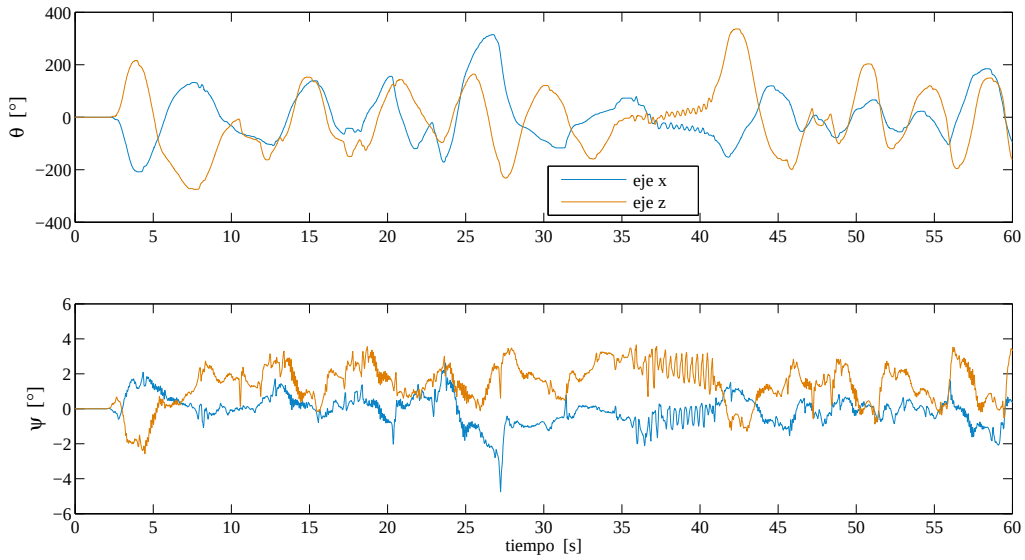


Figura F.1: Comparativa entre los estados del sistema en los ejes x y z ante el controlador por ubicación de polos en regiones LMI en el modelo 1 y la aplicación de pequeñas perturbaciones durante su operación.

En la Figura F.1 se observan pequeños picos en la trayectoria del robot, los cuales corresponden a las perturbaciones realizadas, así también, se observa que en el intervalo de 35 a 40 segundos donde se obstruyó el desplazamiento del NXT ballbot, se presentan ligeras y con-

tínuas oscilaciones como en el caso del controlador LQR. En cuanto al ángulo de inclinación, en ambos ejes se observa que las perturbaciones no fueron mayores a los 5°.

Analizando el esfuerzo de control requerido para estabilizar al sistema NXT ballbot, en la Figura F.2 se observa como en ambos planos la demanda en el esfuerzo de control es semejante, presentando picos en la magnitud del ciclo de trabajo requerido para compensar las perturbaciones y durante el intervalo de 35 y 40 segundos se observa una respuesta similar al presentado por el controlador LQR.

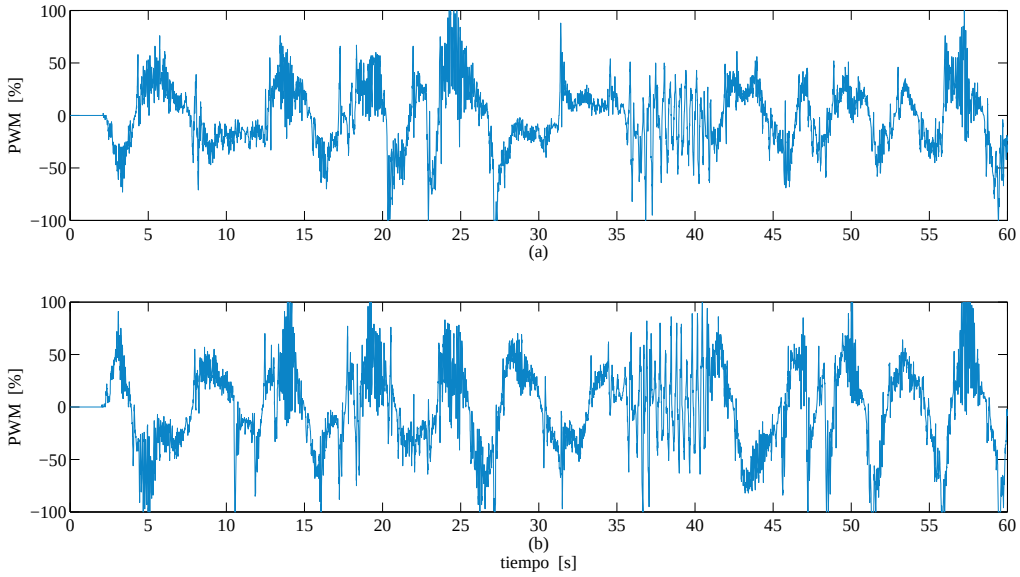


Figura F.2: Comparativa de las señales que muestran el ciclo de trabajo del PWM del sistema ante el controlador por ubicación de polos en regiones LMI en el modelo 1; (a) eje x, (b) eje z.

En cuanto a la segunda prueba, esta vez se le aplicó al sistema NXT ballbot una perturbación de mayor magnitud, la cual provocó un ángulo de inclinación en el eje z mayor a 25°, cómo se observa en la Figura F.3 provocando que el controlador no pudiera estabilizar al sistema NXT ballbot y este cayera al suelo.

Analizando el esfuerzo de control requerido para estabilizar al sistema NXT ballbot, en la Figura F.4 se observa como la demanda en el esfuerzo de control es mayor en el eje z, donde además llega a saturarse en el momento en el que se le aplica la perturbación de mayor amplitud, lo que generó que el controlador no pudiera cumplir con su objetivo.

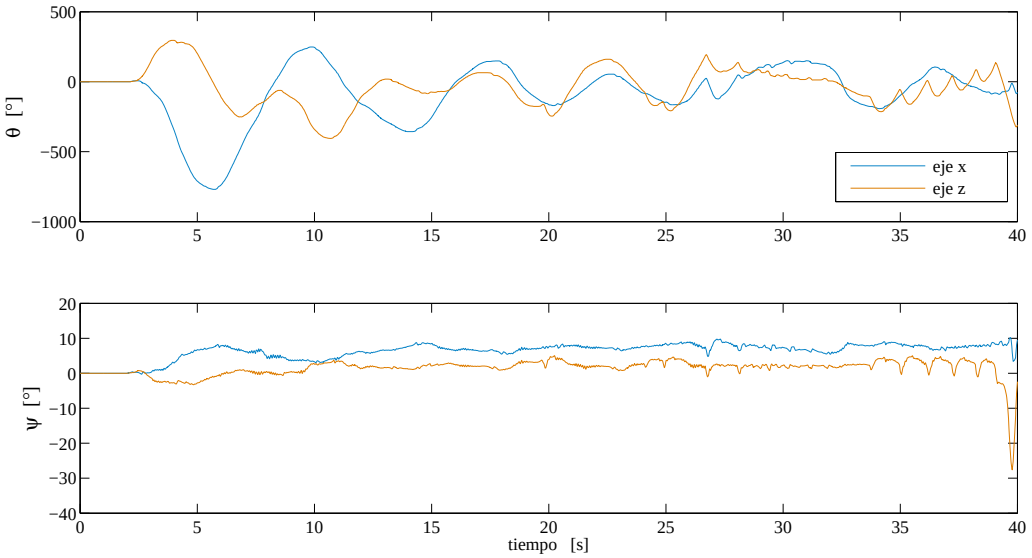


Figura F.3: Comparativa entre los estados del sistema en los ejes x y z ante el controlador por ubicación de polos en regiones LMI en el modelo 1 y la aplicación de pequeñas perturbaciones durante su operación.

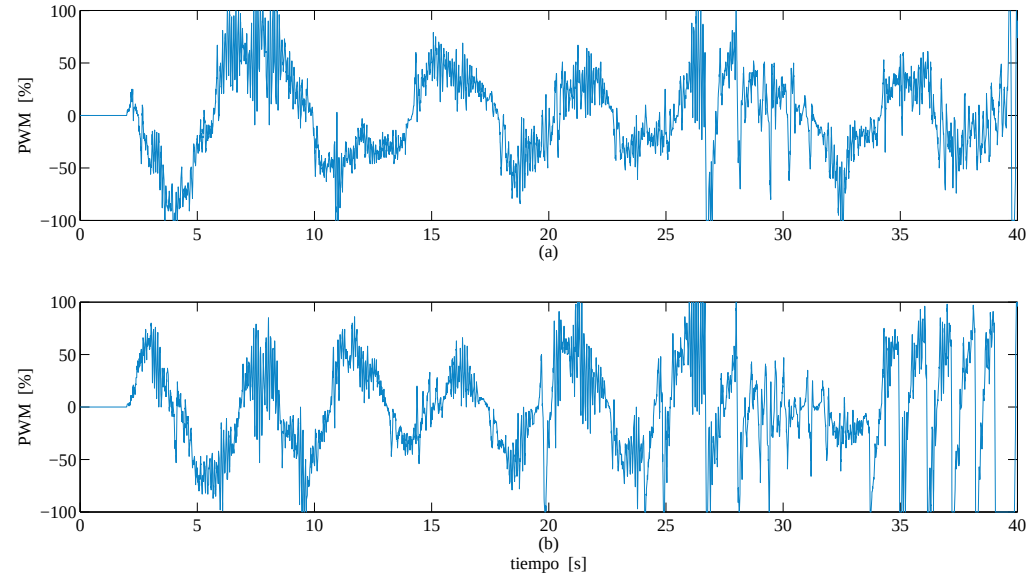


Figura F.4: Comparativa de las señales que muestran el ciclo de trabajo del PWM del sistema ante el controlador por ubicación de polos en regiones LMI en el modelo 1; (a) eje x, (b) eje z.

Apéndice G

Trabajo derivado

En este apéndice se muestra un artículo derivado del presente trabajo de investigación, realizado en colaboración con el Dr. Manuel Arias Montiel. Dicho artículo fue aceptado el 3 de Junio del presente año, y se expondrá en el 5º simposio internacional sobre sistemas multicuerpo y mecatrónica 2014 (MuSMe, *Multibody Systems and Mechatronics*, por sus siglas en inglés) que se llevará a cabo del 21 al 24 de Octubre del presente año en las instalaciones de la Universidad del Mar, campus Huatulco.

Noname manuscript No.
(will be inserted by the editor)

A robust control scheme against some parametric uncertainties for the NXT ballbot

R.A. García-García · **M. Arias-Montiel**

Received: date / Accepted: date

Abstract In this work a Linear Matrix Inequalities (LMIs) approach to provide robustness to an optimal control scheme for the NXT ballbot is proposed. The mathematical model of the system is obtained taking into account the actuators dynamics and in this overall model uncertain parameters appear. These uncertainties can affect (in a negative form) the control algorithm performance, so we define a polytopic model considering some parameters vary within some bounded sets. This polytopic model is used to synthesize a robust control scheme against parameters variation using LMIs. Numerical results show a significant improvement in the closed loop system in comparison with a classic Linear Quadratic Regulator (LQR) control. Finally, some experimental results are presented to show the viability of implementing the control strategy proposed.

Keywords NXT ballbot · Linear Matrix Inequalities (LMIs) · Uncertain parameters

1 Introduction

A ballbot is a thin, agile and omnidirectional robot that balances on, and is propelled by, a single spherical wheel. The system relies on active balancing and thus is dynamically stable [1]. Dynamical characteristics and potential applications have made ballbot an interesting object of study for engineers and researchers in automatic control, mechatronics and robotics areas [2] - [7].

An experimental platform used in order to study the dynamical behavior of the ballbot is the *LEGO Mindstorms NXT*® [8], [9]. *LEGO Mindstorms NXT*® is a

R.A. García-García
Institute of Electronics and Mechatronics, Universidad Tecnológica de la Mixteca
Tel.: +52-953-5320214
E-mail: rafagarcia.0810@gmail.com

M. Arias-Montiel
Institute of Electronics and Mechatronics, Universidad Tecnológica de la Mixteca
E-mail: mam@mixteco.utm.mx

programmable robotics kit with appropriate characteristics to be used in education and research activities [10] - [12].

Yamamoto [9] developed a version of a ballbot, called NXT ballbot, based on a *LEGO Mindstorms NXT*® kit. He presented a linearized model of the NXT ballbot and he proposed a Linear Quadratic Regulator (LQR) in order to achieve the stabilization of the system in a vertical position. Prieto et al. [8] built a ballbot using a *LEGO Mindstorms NXT*® kit with light structural modifications in relation to Yamamoto's. In addition, they proposed some procedures in order to obtain actuators parameters and to characterize gyro sensors used to measure angular rate. The control algorithm used to stabilize the ballbot was a LQR.

In order to control the NXT ballbot, actuators dynamics must be included in the model. Some authors have reported different values for physical parameters for NXT servomotors based on experimental tests [8], [9], [13], [14], so the parametric uncertainties are present and they can affect in a negative form the control algorithm performance. In this work an LMIs approach to provide robustness against parametric uncertainties to a LQR control scheme for the balancing problem in a NXT ballbot is proposed. A polytopic model considering variations within some bounded sets for inertial and friction parameters for system actuators is defined. This polytopic model is used to synthesize a robust control scheme against these parameters variations using LMIs. Numerical results are presented to show the significant improvement in the closed loop system using robust control in comparison with a classic LQR control. Some experiments were carried out and the obtained results demonstrate the viability of implementing the robust control strategy proposed.

2 System description and modeling

2.1 The NXT ballbot

In Fig. 1(a) the mechanical structure for the NXT ballbot is shown. It consists of standard structural parts, a plastic spherical ball, two wheels and two servomotors, all of them included in the basic *LEGO Mindstorms NXT*® kit, and in addition, two *Hitechnic*® gyrosensors. The rubber wheels called driven wheels transmit the motion from the servomotors to the spherical ball in order to vary its rotation axis and thus, equilibrate the system. Gyrosensors measure pitch and yaw angles rate and this information is used by the controller to actuate the servomotors providing the control forces to the closed loop system.

2.2 The NXT ballbot dynamic model

Dynamic model for NXT ballbot is based on a spherical wheeled inverted pendulum considering that the motions in xy plane and zy plane are decoupled (see Fig. 1(b)). Under this assumption, two identical motion equations are obtained, so only one plane of motion is analyzed. In Fig. 1(b) NXT ballbot body is modeled as an inverted pendulum with mass center coordinates x_b, y_b, mass M_b and mass moment

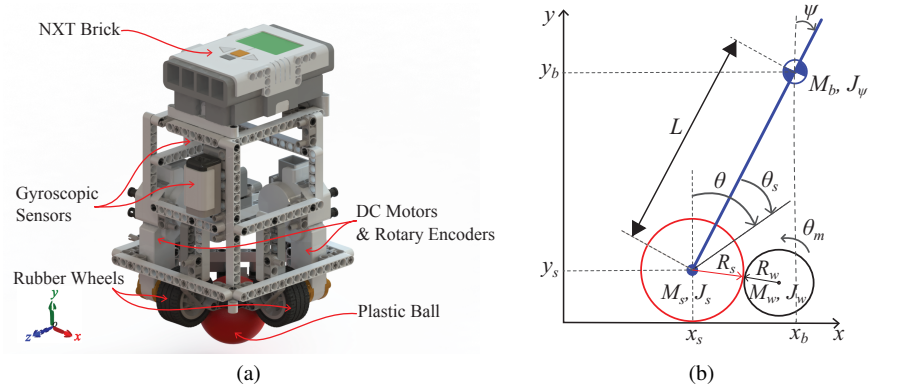


Fig. 1 (a) Render view of the NXT ballbot; (b) Schematic diagram for NXT ballbot

of inertia J_ψ. M_s, J_s and R_s are the mass, moment of inertia and radius of the spherical ball, and M_w, J_w and R_w are the mass, moment of inertia and radius of driven wheels, respectively.

The variables to describe the motion in xy plane are

- ψ *: body inclination angle in relation to vertical axis.*
- θ *: spherical ball angle in relation to vertical axis.*
- θ_s *: spherical ball angle due to the contact with the driving wheel.*
- θ_m *: driving wheel angle imposed by the actuator*

By inspection of Fig. 1(b), the next relationships can be obtained

$$\theta = \psi + \theta_s \tag{1}$$

$$R_\omega \theta_m = R_s \theta_s \tag{2}$$

Considering θ and ψ as the generalized coordinates and applying the Euler-Lagrange method, motion equations are given as

$$\begin{aligned}\left[(M_s+M_b)R_s^2+\frac{R_s^2}{R_\omega^2}(J_m+J_\omega)+J_s\right]\ddot{\theta}+\left[M_bR_sL\cos\psi-\frac{R_s^2}{R_\omega^2}(J_m+J_\omega)\right]\ddot{\psi}\\ -M_bR_sL\dot{\psi}^2\sin\psi=F_\theta\end{aligned} \tag{3}$$

$$\begin{aligned}\left[M_bR_sL\cos\psi-\frac{R_s^2}{R_\omega^2}(J_m+J_\omega)\right]\ddot{\theta}+\left[M_bL^2+J_\psi+\frac{R_s^2}{R_\omega^2}(J_m+J_\omega)\right]\ddot{\psi}\\ -M_bgL\sin\psi=F_\psi\end{aligned} \tag{4}$$

with J_m = moment of inertia of servomotor and g = gravity acceleration constant.

The generalized forces F_θ y F_ψ correspond to servomotor torques and they can be modeled as

$$F_\theta = \alpha e_a - \beta(\dot{\theta}-\dot{\psi}) \tag{5}$$

$$F_\psi = -\alpha e_a + \beta(\dot{\theta}-\dot{\psi}) \tag{6}$$

with $\alpha = \frac{k_b}{R_a}; \beta = k(\frac{k_p k_b}{R_a} + b_e)$, $k = \frac{R_s}{R_w}$, k_p is the torque constant, b_e is the friction coefficient inside the motor, R_a is the armature resistance, k_b is the back electromotriz force constant and e_a is the input voltage.

As we can see, system model expressed by equations (3) and (4) is non linear. In order to synthesize a linear control algorithm, this model is linearized about an equilibrium point [15], i.e. when the NXT ballbot is in vertical position ($\psi = 0$).

Linearized model is

$$\left[(M_s+M_b)R_s^2 + \frac{R_s^2}{R_\omega^2}(J_m+J_\omega)+J_s\right]\ddot{\theta} + \left[M_b R_s L - \frac{R_s^2}{R_\omega^2}(J_m+J_\omega)\right]\ddot{\psi} = F_\theta \quad (7)$$

$$\left[M_b R_s L - \frac{R_s^2}{R_\omega^2}(J_m+J_\omega)\right]\ddot{\theta} + \left[M_b L^2 + J_\psi + \frac{R_s^2}{R_\omega^2}(J_m+J_\omega)\right]\ddot{\psi} - M_b g L\psi = F_\psi \quad (8)$$

2.3 The NXT ballbot model in state space

Equations (7) and (8) can be rewritten in matricial form as

$$\mathbf{E}\ddot{\phi} + \mathbf{F}\dot{\phi} + \mathbf{G}\phi = \mathbf{H}e_a \quad (9)$$

where

$$\phi = \begin{bmatrix}\theta \\ \psi\end{bmatrix}$$

$$\mathbf{E} = \begin{bmatrix}(M_s+M_b)R_s^2 + k^2(J_m+J_\omega)+J_s & M_b R_s L - k^2(J_m+J_\omega) \\ LM_b R_s - k^2(J_m+J_\omega) & M_b L^2 + J_\psi + k^2(J_m+J_\omega)\end{bmatrix}$$

$$\mathbf{F} = \begin{bmatrix}\beta & -\beta \\ -\beta & \beta\end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix}0 & 0 \\ 0 & -M_b g L\end{bmatrix}, \quad \mathbf{H} = \begin{bmatrix}\alpha \\ -\alpha\end{bmatrix}.$$

By defining the state vector $\mathbf{x} = [\theta \quad \psi \quad \dot{\theta} \quad \dot{\psi}]^T$ and the input $u = e_a$, the NXT ballbot dynamics (9) can be described in state space form as follows

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}u \quad (10)$$

with

$$\mathbf{A} = \begin{bmatrix}0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{E_{12}G_{22}}{E_{11}E_{22}-E_{12}^2} & \frac{E_{12}F_{21}-E_{22}F_{11}}{E_{11}E_{22}-E_{12}^2} & \frac{E_{12}F_{22}-E_{22}F12}{E_{11}E_{22}-E_{12}^2} \\ 0 & \frac{-E_{11}G_{22}}{E_{11}E_{22}-E_{12}^2} & \frac{E_{21}F_{11}-E_{11}F_{21}}{E_{11}E_{22}-E_{12}^2} & \frac{E_{21}F_{12}-E_{11}F_{22}}{E_{11}E_{22}-E_{12}^2}\end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix}0 \\ 0 \\ \frac{E_{22}H_{11}-E_{12}H_{21}}{E_{11}E_{22}-E_{12}^2} \\ \frac{E_{11}H_{21}-E_{21}H_{11}}{E_{11}E_{22}-E_{12}^2}\end{bmatrix}.$$

3 NXT ballbot polytopic modeling and control

The LMIs approach is used in system and control theory in order to solve a wide variety of problems (i.e. construction of quadratic Lyapunov functions for stability and performance analysis of linear differential inclusions, optimization over an affine family of transfer matrices, including synthesis of multipliers for analysis of linear systems with unknown parameters, inverse problem of optimal control, multicriterion LQG/LQR, etc.). The use of LMIs is based on polytopic models which can be defined if model matrices include terms with linear dependence of uncertain parameters and these parameters vary within some bounded sets [16], [17].

Different values for the NXT ballbot actuators (NXT servomotors) have been reported in literature; some of these values are presented in Table 1.

Table 1 Physical parameters for servomotors reported in literature

Parameter	Reference[8]	Reference[14]	Reference[13]	Reference[9]
$R_a[\Omega]$	4.6	6.85	5.0012	6.69
$L_a[\mathrm{H}]$	3.7×10^{-3}	*neglected*	1×10^{-3}	*neglected*
$J_m[\mathrm{kg\cdot m^2}]$	45×10^{-9}	$----$	2.4589×10^{-6}	10×10^{-6}
$b_e[\mathrm{N\cdot m\cdot s}]$	1.85×10^{-3}	1.1278×10^{-3}	38.745×10^{-6}	2.2×10^{-3}
$K_p[\frac{\mathrm{N\cdot m}}{A}]$	0.48	0.3179	0.5246	0.317
$K_b[\frac{\mathrm{V\cdot s}}{rad}]$	0.48	0.46389	0.5246	0.468

According to data presented in Table 1, moment of inertia J_m and friction coefficient b_e are the parameters with the widest rank of variation, so the polytopic model for the NXT ballbot will be obtained in terms of these two parameter. The uncertain parameters vector is defined as

$$\rho = (b_e \quad J_m) \tag{11}$$

with the vertices number of resultant polytope $L = 4$, limited by the minimum and maximum values

$$b_e \in [b_{e\min} \quad b_{e\max}], \quad J_m \in [J_{m\min} \quad J_{m\max}] \tag{12}$$

These uncertain parameters are present in matrices **A** and **B** of the equation (10), and these matrices depend on the uncertain parameters in a linear form, so it is possible to define a polytope with 4 vertices which contains all the possible values for the uncertain parameters within their minimum and maximum values. The 4 vertices of polytopic model for matrices **A** and **B** can be defined as

$$\mathbf{A}_i = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{E_{Jm12}G_{22}}{E_{Jm11}E_{Jm22}-E_{Jm12}^2} & \frac{E_{Jm12}F_{be21}-E_{Jm22}F_{be11}}{E_{Jm11}E_{Jm22}-E_{Jm12}^2} & \frac{E_{Jm12}F_{be22}-E_{Jm22}F_{be12}}{E_{Jm11}E_{Jm22}-E_{Jm12}^2} \\ 0 & \frac{-E_{Jm11}G_{22}}{E_{Jm11}E_{Jm22}-E_{Jm12}^2} & \frac{E_{Jm21}F_{be11}-E_{Jm11}F_{be21}}{E_{Jm11}E_{Jm22}-E_{Jm12}^2} & \frac{E_{Jm21}F_{be12}-E_{Jm11}F_{be22}}{E_{Jm11}E_{Jm22}-E_{Jm12}^2} \end{bmatrix}$$

$$\mathbf{B}_i = \begin{bmatrix} 0 \\ 0 \\ \frac{E_{Jm22}H_{11} - E_{Jm12}H_{21}}{E_{Jm11}E_{Jm22} - E_{Jm12}^2} \\ \frac{E_{Jm11}H_{21} - E_{Jm21}H_{11}}{E_{Jm11}E_{Jm22} - E_{Jm12}^2} \end{bmatrix}, \quad i = 1,\ldots,4 \tag{13}$$

where values of J_m and b_e depend on vertex of polytopic model

$$\begin{aligned} [J_{m\min} \quad b_{e\min}] &\in \mathbf{A}_1, \mathbf{B}_1, \\ [J_{m\min} \quad b_{e\max}] &\in \mathbf{A}_2, \mathbf{B}_2, \\ [J_{m\max} \quad b_{e\min}] &\in \mathbf{A}_3, \mathbf{B}_3, \\ [J_{m\max} \quad b_{e\max}] &\in \mathbf{A}_4, \mathbf{B}_4. \end{aligned}$$

In order to be able to solve the tracking problem and to assure the error in steady state converges to zero, another state variable ξ is added to the polytopic model which corresponds to integral of position error for variable θ. The extended polytopic model is expressed by

$$\dot{\mathbf{x}}_a = \mathbf{A}_a\mathbf{x}_a + \mathbf{B}_a u + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mathrm{r} \tag{14}$$

where r is the reference value to track and

$$\mathbf{x}_a = \begin{bmatrix} \mathbf{x} & \xi \end{bmatrix}^T, \quad \mathbf{A}_a = \begin{bmatrix} \mathbf{A}_i & \mathbf{0} \\ -\mathbf{C} & 0 \end{bmatrix}, \quad \mathbf{B}_a = \begin{bmatrix} \mathbf{B}_i \\ 0 \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \quad i = 1,\ldots,L.$$

The extended polytopic model (14) is used to synthesize a LQR control law to stabilize the NXT ballbot in spite of the uncertainties in inertia and friction parameters for the actuators by LMI formulations. The augmented gain vector obtained will be $\mathbf{K}_a = \begin{bmatrix} \mathbf{K} & K_\xi \end{bmatrix}$ with $\mathbf{K} \in \mathbb{R}^{m\times n}$ and $K_\xi \in \mathbb{R}$.

In order to compare the performance of the LMI-LQR control with the classical LQR, values for $\mathbf{Q}$ and $\mathbf{R}$ given by Yamamoto [9] were considered. To obtain the classical LQR controller gains, *Matlab*® *lqr(A,B,Q,R)* function is used.

LQR formulation can be extended to polytopic systems by LMI constrains in order to provide robustness to the controller against uncertainties in parameters. For the system described by equation (14) the problem consists of finding a control law by state feedback $\mathbf{u} = -\mathbf{K}\mathbf{x}$, so that the next cost function in closed loop can be minimized

$$J = \int_0^\infty \left(\mathbf{x}^T \left(\mathbf{Q} + \mathbf{K}^T\mathbf{R}\mathbf{K}\right)\mathbf{x}\right) dt \tag{15}$$

Using the trace operator $\mathbf{Tr}(\cdot)$, which satisfies $\mathbf{Tr}(A^TBC) = \mathbf{Tr}(BCA^T)$, cost function (15) can be rewritten as

$$J = \int_0^\infty \mathbf{Tr}\left(\left(\mathbf{Q} + \mathbf{K}^T\mathbf{R}\mathbf{K}\right)\mathbf{x}\mathbf{x}^T\right) dt = \mathbf{Tr}\left(\left(\mathbf{Q} + \mathbf{K}^T\mathbf{R}\mathbf{K}\right)\mathbf{P}\right)$$

where $\mathbf{P} = \int_0^\infty \left(\mathbf{x}\mathbf{x}^T\right) dt$ is a symmetric positive definite matrix which satisfies the constrain

$$(\mathbf{A}_i - \mathbf{B}_i\mathbf{K})\,\mathbf{P} + \mathbf{P}\,(\mathbf{A}_i - \mathbf{B}_i\mathbf{K})^T + \mathbf{I} = 0, \quad i = 1,\ldots,L. \tag{16}$$

where subindex i represents each vertex of the polytpic model.

The optimal feedback gain can be obtained by

$$\begin{aligned} \min_{\mathbf{P},\mathbf{K},\mathbf{X}} \quad & \mathbf{Tr}\,(\mathbf{QP}) + \mathbf{Tr}\,(\mathbf{X}) \\ \text{subject to} \quad & \\ & \mathbf{P} > 0, \\ & \mathbf{A}_i\mathbf{P} + \mathbf{P}\mathbf{A}_i^T - \mathbf{B}_i\mathbf{Y} - \mathbf{Y}^T\mathbf{B}_i^T + \mathbf{I} < 0, \\ & \begin{bmatrix} \mathbf{X} & \mathbf{R}^{\frac{1}{2}}\mathbf{Y} \\ \mathbf{Y}^T\mathbf{R}^{\frac{1}{2}} & \mathbf{P} \end{bmatrix} > 0. \end{aligned} \tag{17}$$

When the constrains given by (17) are solved, the optimal feddback gain for the robust controller is obtained by $\mathbf{K} = \mathbf{Y}\mathbf{P}^{-1}$ [16], [17], [18].

The numerical problem described by constrains in equation (17) is solved by *Matlab*® LMI Toolbox [19]. Obtained gains controllers are

$$\mathbf{K}_{\text{lqr}} = [-1.504\times10^{-2} \; -1.569 \; -2.700\times10^{-2} \; -2.325\times10^{-1} \; -7.125\times10^{-3}] \tag{18}$$

$$\mathbf{K}_{\text{lmi-lqr}} = [-7.241\times10^{-3} \; -1.385 \; -1.808\times10^{-2} \; -2.298\times10^{-1} \; -2.549\times10^{-3}] \tag{19}$$

The control law is based on full state feedback and it is proposed as follows

$$u = -\mathbf{K}\mathbf{x}_\mathbf{a} \tag{20}$$

where $\mathbf{K}$ is the gains vector given by equation (18) or (19) and $\mathbf{x}_\mathbf{a}$ is the extended vector state.

4 Numerical and experimental results

In order to get some numerical results and to compare the performance of classical LQR with the LMI-LQR rosbust control scheme, nominal parameters values for the system given by Yamamoto [9] are considered. First, the extended polytopic model given by equation (14) with nominal parameters values and with the control input given by (20) is numerically solved. In Fig. 2(a)-(b), a comparison of closed loop system response for classical non robust LQR and for the robust LMI-LQR control is presented. As we can observe, in this case there is not any significant difference in the response of the four state variables. Something similar can be noted in the control effort shown in Fig. 2(c).

Now, we explore the variation in control performance when inertial and friction parameters of the actuators change using some vertices of the extended polytopic

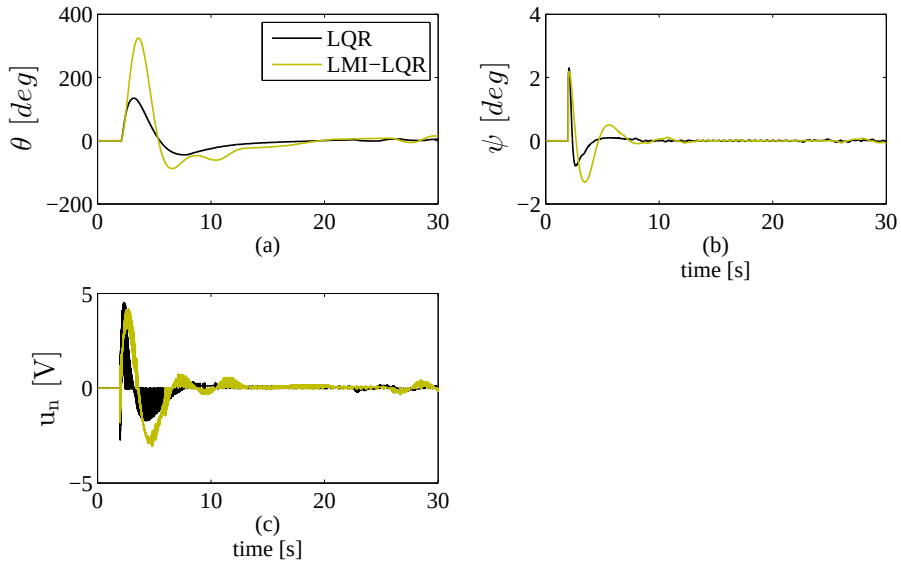


Fig. 2 State variables and control effort of the closed loop system with nominal parameters

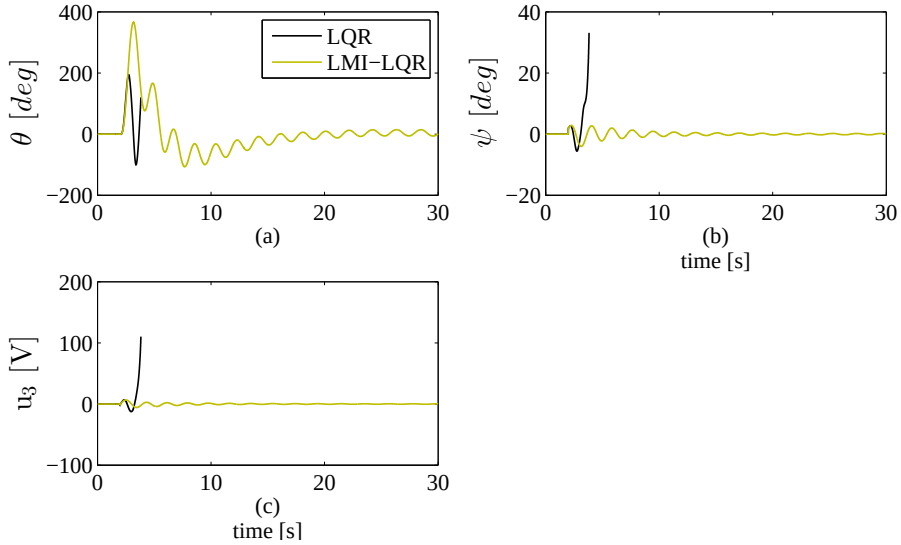


Fig. 3 State variables and control effort of the closed loop system for vertice 3 of the extended polytopic model

model which represent the extreme values (minimum or maximum) for the uncertain parameters, given in Table 1. In Fig. 3(a)-(b) closed loop system response for vertice 3 (i.e. matrices $\mathbf{A}_3$ and $\mathbf{B}_3$) of the extended polytopic model is depicted. The comparison between the non robust LQR and the robust LMI-LQR control shows the first one is not able to stabilize the NXT ballbot, at around 4s the system falls down losing

the equilibrium. In Fig. 3(c) the voltage input obtained from each controller is shown. A similar behavior is obtained at the other three vertices.

In order to demonstrate the viability of practically applying the robust control scheme proposed, some experiments were carried out. Data were acquired by gyrosensors in both planes of motion. Data acquisition was carried out by transfer information in real time via bluetooth from NXT brick to a personal computer equiped with *Simulink® Support Package for LEGO MINDSTORMS NXT® hardware*.

Experimental results for the state variables and for the measured input voltage are presented in Fig. 4.

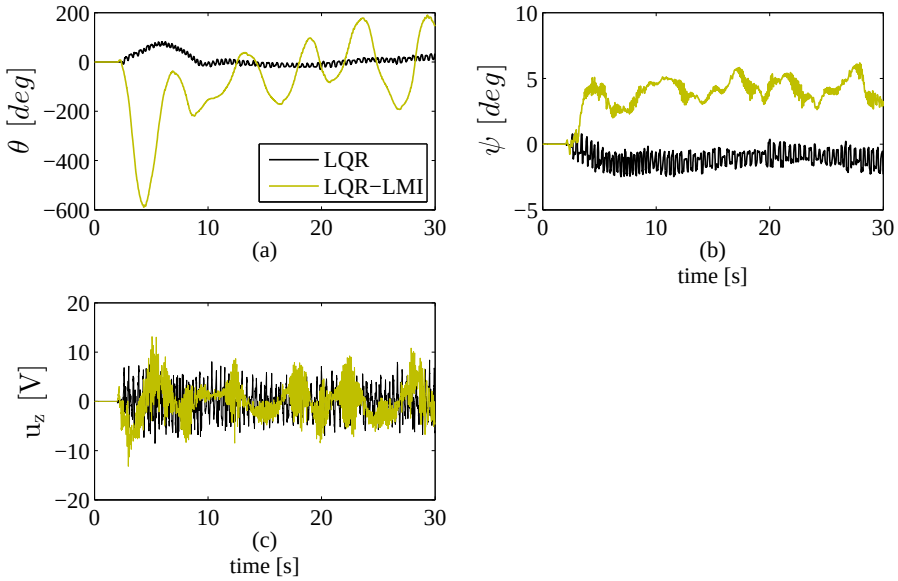


Fig. 4 Experimental response in the *xy* plane

It is important to mention that implemented controllers were tuned with gains given by (18) and (19) and we were not able to physically vary the inertial and friction parameters for the actuators, so that the experimental results presented are equivalent to numerical results for the nominal model shown in Fig. 2. In Fig. 4(a) we can observe the response oscilates around $\theta = 0$. This means ballbot needs displacements in *zx* plane in order to get the vertical equilibrium. This effect can be reduced by increasing the available control effort limited by LMI constraints.

5 Conclusions

Because of its potential applications, ballbot system is an interesting object of study for engineers and researchers in automatic control, mechatronics and robotics areas. In this paper a robust control scheme against parametric uncertainty for a ballbot

system was proposed. Robust controller is based on a optimal linear quadratic regulator (LQR) in combination with linear matrix inequalities (LMIs) theory which uses a polytopic model in order to consider the rank variation for some parameter values. Numerical results show an important improvement in the closed loop system behavior of the robust controller in comparison with the classical LQR reported in literature when the system parameters values change. In order to prove the dynamic behavior of the proposed control scheme, we use the experimental platform *LEGO Mindstorms NXT*® kit to built a ballbot system. Experimental results show the viability of practical implementation of the robust control based on LMIs approach. This control scheme can be extended in order to include uncertainties in other system parameters and even to take into account the disturbance rejection problem.

References

1. Hollis, R., Ballbots, Scientific American, Vol. 295, No. 4, pp. 72-77 (2006).
2. Asgari, P., Zarafshan, P., Moosavian, S.A.A., Dynamics modelling and stable motion control of a ballbot equipped with a manipulator, Proceedings of the 2013 International Conference on Robotics and Mechatronics, pp. 259-264 (2013).
3. Fankhauser, P., Gwerder, C., Modeling and control of a ballbot, Bachelor Thesis, Autonomous Systems Lab, Swiss Federal Institute of Technology Zurich (2010).
4. Fong, J., Uppill, S., Design and build a ballbot, Bachelor Thesis, Faculty of Engineering, Computer & Mathematical Sciences, University of Adelaide (2009).
5. Mampetta, A., Automatic transition of ballbot from statically stable state to dynamically stable state, Master's Thesis, Robotics Institute, Carnegie Mellon University (2006).
6. Nagarajan, U., Kantor, G., Hollis, R., The ballbot: An omnidirectional balancing mobile robot, The International Journal of Robotics Research, Published online 13 November 2013, pp. 1-14.
7. Skrabel, C., Mechanical design of a ballbot platform, Bachelor Thesis, Autonomous Systems Lab, Swiss Federal Institute of Technology Zurich (2013).
8. Prieto, S., Navarro, T., Plaza, M., Polo, O., A monoball robot based on lego mindstorm, IEEE Control Systems, Vol. 32, No.2, pp. 71-83 (2012).
9. Yamamoto, Y., NXT ballbot model-based design -control of self-balancing robot on a ball, built with Lego Mindstorm NXT, Cynernet Systems CO., LTD., first edition (2009).
10. Bradley, P., De la Puente, J., Zamorano, J., Brosnan, D., A platform for real-time control education with lego mindstorms, Proceedings of the 9th IFAC Symposium Advances in Control Education, pp. 112-117 (2012).
11. Kim, Y., Control systems lab using a lego mindstorms nxt motor system, IEEE Transactions on Education, Vol. 54, No.3, pp. 452-461 (2011).
12. Pinto, M., Moreira, A.P., Matos, A., Localization of mobile robots using an extended Kalman filter in a lego nxt, IEEE Transactions on Education, Vol. 55, No.1, pp. 135-144 (2012).
13. Pinto-Salamanca, M.L., Bermúdez-Bohórquez, G.R., Parameters determination of a mobil robot of lego minstorms, Ingeniería Investigación y Desarrollo, Vol. 5, No. 2, pp. 7-13 (2007).
14. Hurbain P., Hurbain P., Gasperi M., NXT motor, The International Fan-Created *Lego*® Users Group Network, Extracted from World Wide Web: `http://web.archive.org/web/20110328142611/http://web.mac.com/ryowatanabe/iWeb/RyosHoliday/NXTMotor.html`. (2007).
15. Vukic, Z., Kuljaca, L., Donlagic, D., Tesnjak, S., Nonlinear control systems, Marcel Dekker, Inc., Basel, Switzerland (2003).
16. Boyd, S., El Ghaoui, L., Feron, E., Balakrishnan, V., Linear matrix inequalities in system anc control theory, Society for Industrial and Applied Mathematics SIAM, Philadelphia, USA (1994).
17. Ostertag, E., Mono and multivariable control and estimation - Linear, quadratic and LMI methods, Springer Verlag, Berlin, Germany (2011).
18. Olalla, C., Leyva, R., El-Aroudi, A. y Queinnec, I., obust lqr control for pwm converters: An lmi approach, IEEE Transactions on Industrial Electronics, Vol. 56, No.7, pp. 2548-2558 (2009).
19. Erkus, B., Lee, Y.J., Linear Matrix Inequalities and MATLAB LMI Toolbox, University of Southern California Group Meeting Report, Los Angeles, California, (2004).

Bibliografía

[1] Abreu, V. V. *Balance-bot*. Tesis de Maestría, Universidad de Madeira, Funchal, Portugal, 2009.

[2] Andreev, F., Auckly, D., Gosavi, S., Kapitanski, L., Kelkar, A. y White, W. Matching, linear systems, and the ball and beam. *Automatica*, vol. 38(12), páginas 2147 – 2152, 2002.

[3] Aracil, J. y Gordillo, F. El péndulo invertido: un desafío para el control no lineal. *Revista Iberoamericana de Automática e Informática Industrial (RIAI)*, vol. 2(2), páginas 8–19, 2005.

[4] Astrom, K. y Furuta, K. Swinging up a pendulum by energy control. *Automatica*, vol. 36(2), páginas 287 – 295, 2000.

[5] Baker, G. y Blackburn, J. *The Pendulum: A Case Study in Physics*. OUP Oxford, New York, United States, 2009.

[6] Bjarenstam, M. J. y Lennartsson, M. *Development of a ball-balancing robot with omni-wheels*. Tesis de maestría, Lund University, Lund, Sweden, 2012.

[7] Block, D. *Mechanical Design and Control of the Pendubot*. Tesis de maestría, University of Illinois, Urbana-Champaign,Il,USA, 1996.

[8] Block, D., Åström, K. y Spong, M. *The Reaction Wheel Pendulum*, vol. 1 de *Synthesis lectures on control and mechatronics*. Morgan & Claypool Publishers, Urbana-Champaign,Il,USA, 2007.

[9] Bobtsov, A., Pyrkin, A., Kolyubin, S., Chepinskiy, S., Shavetov, S., Kapitanyuk, Y., Kapitonov, A., Titov, A., Surov, M. y Bardov, V. Using of lego mindstorms nxt technology for teaching of basics of adaptive control theory. En *Proceedings of the 18th IFAC World Congress*, vol. 18, páginas 9818–9823. 2011.

[10] Bortoff, S. y Spong, M. Pseudolinearization of the acrobot using spline functions. En *Proceedings of the 31st IEEE Conference on Decision and Control, 1992*, vol. 1, páginas 593–598. 1992.

[11] Boubaker, O. The inverted pendulum: A fundamental benchmark in control theory and robotics. En *Proceedings of the International Conference on Education and e-Learning Innovations (ICEELI), 2012*, páginas 1–6. 2012.

[12] Boyd, S., Ghaoul, L., Feron, E. y Balakrishnan, V. *Linear Matrix Inequalities in System and Control Theory*, vol. 15 de *Studies in Applied Mathematics*. Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania, 1994.

[13] Bradley, P., de la Puente, J., Zamorano, J. y Brosnan, D. A platform for real-time control education with lego mindstorms. En *Proceedings of the 9th IFAC Symposium Advances in Control Education, 2012. IFAC.*, vol. 9, páginas 112–117. 2012.

[14] Brown, S. y Passino, K. Intelligent control for an acrobot. *Journal of Intelligent and Robotic Systems*, vol. 18(3), páginas 209–248, 1997.

[15] Burns, R. *Advanced Control Engineering*. Elsevier Science, Massachusetts, United States, 2001.

[16] Chilali, M. y Gahinet, P. H_∞ design with pole placement constraints: an lmi approach. *IEEE Transactions on Automatic Control*, vol. 41(3), páginas 358–367, 1996.

[17] Corona-Miranda, J. L. *Application of Kalman Filtering and Pid Control for Direct Inverted Pendulum Control*. Tesis de Maestría, Faculty of California State University, Chicago, California , United States, 2009.

[18] Dorf, R. y Bishop, R. *Modern Control Systems*. Pearson Prentice Hall, Singapore, 11 edición, 2008.

[19] Erkus, B. y Lee, Y. J. Linear matrix inequalities and matlab lmi toolbox. Informe técnico, University of Southern California, Los Angeles, California, 2004.

[20] Fankhauser, P. y Gwerder, C. Modeling and control of a ballbot. Tesis de licenciatura, ETH Zurich, Zürich, Suiza, 2010.

[21] Fantoni, I., Lozano, R. y Spong, M. Energy based control of the pendubot. *IEEE Transactions on Automatic Control 2000.*, vol. 45(4), páginas 725–729, 2000.

[22] Fong, J. y Uppill, S. Design and build a ballbot. Tesis de licenciatura, University of Adelaide,Faculty of Engineering, Computer & Mathematical Sciences School of Mechanical Engineering, North Terrace Adelaide, Australia, 2009.

[23] Furuta, K., Yamakita, M. y Kobayashi, S. Swing-up control of inverted pendulum using pseudo-state feedback. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 206(4), páginas 263–269, 1992.

[24] Gauld, C. *The Pendulum: Scientific, Historical, Philosophical and Educational Perspectives*, capítulo Educational Perspectives, Pendulums in The Physics Education Literature: A Bibliography, páginas 505–526. Science & Education. Springer, Dordrecht, The Netherlands, 2005.

[25] Grega, W. y Pilat, A. Real-time control teaching using lego mindstorms nxt robot. En *Proceedings of the International Multiconference on Computer Science and Information Technology, 2008*, páginas 625–628. IEEE, 30-059 Krakow, Poland, 2008.

[26] Gutiérrez-Frías, O. O. *Diseño de Leyes de Control de un Péndulo Invertido montado sobre un carro*. Tesis de Maestría, Instituto Politécnico Nacional, México, D.F., 2006.

[27] Gutiérrez-Frías, O. O. *Diseño de Controladores para Sistemas Subactuados del Tipo Péndulo Invertido*. Tesis Doctoral, Instituto Politécnico Nacional, México, D.F., 2009.

[28] Gutiérrez-Frías, O. O., Aguilar Ibañez, C. y Sossa, A. H. Stabilization of the inverted spherical pendulum via lyapunov approach. *Asian Journal of Control*, vol. 11(6), páginas 587–594, 2009.

[29] Hauser, J., Sastry, S. y Kokotovic, P. Nonlinear control via approximate input-output linearization: the ball and beam example. *IEEE Transactions on Automatic Control, 1992.*, vol. 37, páginas 392–398, 1992.

[30] Hauser, J., Sastryii, S. y Meyer, G. Nonlinear control design for slightly non minimum phase systems- applications to v/stol aircraft. *Automatica*, vol. 28(4), páginas 665–679, 1992.

[31] Hernández Cifre, M. A. Capítulo 1: Conceptos y resultados básicos en la geometría de los cuerpos convexos. Sitio web, 2010. http://webs.um.es/mhcifre/apuntes/cap1.pdf [Online; accessed 08-Mayo-2014].

[32] Hernández-Cruz, P. T. Control de robots móviles: El caso del ballbot. Tesis de licenciatura, Universidad Nacional Autónoma de México, México, 2010.

[33] Hollis, R. Ballbots. *Scientific American*, vol. 295(4), páginas 72–77, 2006.

[34] Hurbain, P. Nxt® motor internals. Sitio web, 2006. http://www.philohome.com/nxtmotor/nxtmotor.htm [Online; accessed 9-January-2014].

[35] Hurbain, P. y Gasperi, M. *Extreme NXT: Extending the LEGO® MINDSTORMS® NXT to the Next Level*. Technology in action series. Apress, 2007.

[36] Hurbain, P. y Gasperi, M. Nxt motor. Sitio web, 2007. http://web.archive.org/web/20110328142611/http://web.mac.com/ryowatanabe/iWeb/RyosHoliday/NXTMotor.html [Online; accessed 08-April-2014].

[37] Jankovic, M., Fontaine, D. y Kokotovic, P. Tora example: cascade- and passivity-based control design. *IEEE Transactions on Control Systems Technology*, vol. 4(3), páginas 292–297, 1996.

[38] Kanada, T., Watanabe, Y. y Chen, G. Robust h2 control for two-wheeled inverted pendulum using lego mindstorms. En *Proceedings of the Australian Control Conference (AUCC), 2011* , páginas 136–141. IEEE, 2011.

[39] Kim, Y. Control systems lab using a lego mindstorms nxt motor system. *IEEE Transactions on Education.*, vol. 54(3), páginas 452–461, 2011.

[40] Ko, H. S., Jatskevich, J., Dumont, G. y Yoon, G. G. An advanced lmi-based-lqr design for voltage control of grid-connected wind farm. *Electric Power Systems Research*, vol. 78(4), páginas 539–546, 2008.

[41] Kosoris, B., Waning, J., Psarev, Y. y Gitego, B. Balanced robotics ballbot. Informe técnico, Southern Polytechnic State University, Georgia, Estados Unidos, 2011.

[42] Kumaga, M. y Ochiai, T. Development of a robot balanced on a ball - application of passive motion to transport. En *Proceedings of the IEEE International Conference on Robotics and Automation, 2009. ICRA '09.*, páginas 4106–4111. 2009.

[43] Kumar, A. *Convex Modeling Techniques for Aircraft Control*. Tesis de Maestría, Faculty of the Virginia Polytechnic Institute and State University, Blacksburg, Virginia, 2000.

[44] Lauwers, T., Kantor, G. y Hollis, R. One is enough En *Proceedings of 2005 International Symposium for Robotics Research*, páginas 1–10. 2005.

[45] Lauwers, T. B., Kantor, G. A. y Hollis, R. L. A dynamically stable single-wheeled mobile robot with inverse mouse-ball drive. En *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, páginas 2884–2889. Florida, United States, 2006.

[46] LEGO. Lego® mindstorms® nxt, hardware developer kit. Sitio web, 2006. http://www.lego.com/en-us/mindstorms/downloads/nxt/nxt-hdk/ [Online; accessed 19-March-2014].

[47] Li, Z., Yang, C. y Fan, L. *Advanced Control of Wheeled Inverted Pendulum Systems*. SpringerLink. Springer, London, 2012.

[48] Liu, G., Neˇsić, D. y Mareels, I. Modeling and stabilisation of a spherical inverted pendulum. *IFAC World Congress*, vol. 16, páginas 844–844, 2005.

[49] Martin, P., Devasia, S. y Paden, B. A different look at output tracking: control of a vtol aircraft. En *Proceedings of the 33rd IEEE Conference on Decision and Control, 1994.*, vol. 3, páginas 2376–2381. 1994.

[50] Matthews, M. R., Gauld, C. F. y Stinner, A. *The Pendulum: Scientific, Historical, Philosophical and Educational Perspectives*, capítulo Introduction, The Pendulum: Its Place in Science, Culture and Pedagogy, páginas 1–17. Science & Education. Springer, Dordrecht, The Netherlands, 2005.

[51] Moreno Armendariz, M. A. *Control adaptable del sistema no lineal TORA usando redes neuronales*. Tesis de maestría, CINVESTAV IPN, México, D.F., 1999.

[52] Nagarajan, U., Kantor, G. y Hollis, R. L. Trajectory planning and control of an underactuated dynamically stable single spherical wheeled mobile robot. En *Proceedings of the IEEE International Conference on Robotics and Automation, 2009. ICRA '09.*, páginas 3743–3748. 2009.

[53] Nagarajan, U., Mampetta, A., Kantor, G. A. y Hollis, R. State transition, balancing, station keeping, and yaw control for a dynamically stable single spherical wheel mobile robot. En *IEEE International Conference on Robotics and Automation, 2009. ICRA '09.*, páginas 998–1003. 2009.

[54] Naidu, D. *Optimal Control Systems*. Electrical engineering textbook series. Taylor & Francis, Florida, United States., 2003.

[55] Ochoa-Giménez, M. Modelado y control de un robot equilibrista sobre una esfera. Tesis de licenciatura, Escuela Técnica Superior de Ingeniería (ICAI), Centro Madrid, España, 2011.

[56] Ogata, K. *Ingeniería de control moderna*. Pearson educación. Pearson Educación, Ribera del Loira 28, 28042 Madrid, 2003.

[57] Olalla, C., Leyva, R., El Aroudi, A., Garces, P. y Queinnec, I. Lmi robust control design for boost pwm converters. *Power Electronics, IET*, vol. 3(1), páginas 75–85, 2010.

[58] Olalla, C., Leyva, R., El-Aroudi, A. y Queinnec, I. Robust lqr control for pwm converters: An lmi approach. *IEEE Transactions on Industrial Electronics*, vol. 56(7), páginas 2548–2558, 2009.

[59] Olfati-Saber, R. y Megretski, A. Controller design for the beam-and-ball system. En *Proceedings of the 37th IEEE Conference on Decision and Control, 1998*, vol. 4, páginas 4555–4560. 1998.

[60] Ooi, R. C. Balancing a two-wheeled autonomous robot. Tesis de licenciatura, The University of Western Australia School of Mechanical Engineering, Crawley, Australia, 2003.

[61] Ostertag, E. *Mono- and Multivariable Control and Estimation: Linear, Quadratic and LMI Methods*. Mathematical Engineering. Springer, 2011.

[62] Perdue, D. J. y Valk, L. *The Unofficial LEGO Mindstorms NXT 2.0 Inventorś Guide*. No Starch Press, San Francisco, California., 2010.

[63] Pinto, M., Moreira, A. P. y Matos, A. Localization of mobile robots using an extended kalman filter in a lego nxt. *IEEE Transactions on Education*, vol. 55(1), páginas 135–144, 2012.

[64] Pinto-Salamanca, M. L. y Bermuúdez-Bohórquez, G. R. Determinación de parámetros de un robot móvil de lego mindstorms®. *Ingeniería, Investigación y Desarrollo*, vol. 5(2), páginas 7–13, 2007.

[65] Pook, L. *Understanding Pendulums: A Brief Introduction*, vol. 12 de *History of mechanism and machine science*. Springer, Sevenoaks, United Kingdom, 2011.

[66] Prieto, S., Navarro, T., Plaza, M. y Polo, O. A monoball robot based on lego mindstorms [focus on education]. *IEEE Control Systems*, vol. 32(2), páginas 71–83, 2012.

[67] Rao, S. y Salas, R. *Vibraciones mecánicas*. Pearson Educación, 2012.

[68] Schearer, E. M. *Modeling Dynamics and Exploring Control of a Single-Wheeled Dynamically Stable Mobile Robot with Arms*. Tesis de maestría, Carnegie Mellon University, Pensilvania , United States, 2006.

[69] Spong, M. Swing up control of the acrobot. En *Proceedings of the IEEE International Conference on Robotics and Automation, 1994*, vol. 3, páginas 2356–2361. 1994.

[70] Spong, M. y Block, D. The pendubot: a mechatronic system for control research and education. En *Proceedings of the 34th IEEE Conference on Decision and Control, 1995*, vol. 1, páginas 555–556. 1995.

[71] Spong, M., Corke, P. y Lozano, R. Nonlinear control of the inertia wheel pendulum. *Automatica*, vol. 37, páginas 1845–1851, 1999.

[72] Teppa-Garrán, P. A. Control robusto de un sistema lineal de parámetros variantes (lpv): Un enfoque de las desigualdades matriciales lineales (lmi). *Revista de la Facultad de Ingeniería de la Universidad Central de Venezuela*, vol. 23(1), páginas 5–17, 2008.

[73] University, C. M. Lego® solid modeling. Sitio web, 2008. http://www.education.rec.ri.cmu.edu/content/lego [Online; accessed 4-September-2013].

[74] University, E. Z. Rezero ballbot. Sitio web, 2010. http://rezero.ethz.ch/ [Online; accessed 4-September-2013].

[75] Wan, C.-J., Bernstein, D. y Coppola, V. Global stabilization of the oscillating eccentric rotor. En *Proceedings of the 33rd IEEE Conference on Decision and Control, 1994.*, vol. 4, páginas 4024–4029. 1994.

[76] Xin, Y. J. Ballancing ballbot. Tesis de licenciatura, Faculty of Electrical Engineering Universiti Teknologi Malaysia, Skudai 81310, Johor, Malaysia, 2012.

[77] Yamamoto, Y. *NXT Ballbot Building Instructions*. Cybernet Systems CO., LTD., primera edición, 2009.

[78] Yamamoto, Y. *NXT Ballbot Model-Based Design - Control of self-balancing robot on a ball, built with LEGO Mindstorms NXT* -. Cybernet Systems CO., LTD., primera edición, 2009.

[79] Yoon, M.-G. Dynamics and stabilization of a spherical inverted pendulum on a wheeled cart. *International Journal of Control, Automation and Systems*, vol. 8(6), páginas 1271–1279, 2010.

[80] Zion, A. O. Stabilization of an underactuated robot on a ball. Tesis de licenciatura, Ben-Gurion University of the Negev, Department of Mechanical Engineering, Beerseba, Israel, 2009.